An Investigation of Data Privacy and Utility Preservation using KNN Classification as a Gauge

Kato Mivule¹ and Claude Turner PhD²
¹mivulek0220@students.bowiestate.edu, ²cturner@bowiestate.edu
Computer Science Department, Bowie State University, Bowie, MD, USA

Abstract – It is obligatory that organizations by law safeguard the privacy of individuals when handling datasets containing personal identifiable information (PII). Nevertheless, during the process of data privatization, the utility or usefulness of the privatized data diminishes. Yet achieving the optimal balance between data privacy and utility needs has been documented as an NP-hard challenge. In this study, we investigate data privacy and utility preservation using KNN machine learning classification as a gauge.

Keywords: Data Privacy Preservation, Data Utility, Machine Learning, KNN Classification.

I. INTRODUCTION

During the process of data privatization, the utility or usefulness of the privatized data diminishes. Yet achieving the optimal balance between data privacy and utility needs has been documented as an NP-hard challenge [1] [2]. In this study, we investigate data privacy and utility preservation using KNN machine learning classification as a gauge. As Cynthia Dwork succinctly and aptly stated [6]:

“Perfect privacy can be achieved by publishing nothing at all, but this has no utility; perfect utility can be obtained by publishing the data exactly as received, but this offers no privacy”.

In this study, we investigate data privacy and utility preservation using KNN machine learning classification as a gauge [4].

Noise addition: is a data privacy perturbative method that adds a random value, usually selected from a normal distribution with zero mean and a very small standard deviation, to sensitive numerical attribute values to ensure privacy [3] [8]. The general expression of noise addition as defined:

\[ X + \varepsilon = Z \]  

(1)

Where \( X \) is the original numerical dataset and \( \varepsilon \) is the set of random values (noise) with a distribution \( \varepsilon \sim N(0, \sigma^2) \) that is added to \( X \), and finally \( Z \) is the privatized dataset.

II. METHODOLOGY

In the first stage of our approach, we apply a data privacy procedure, in this case, noise addition, on the Iris dataset for privacy [7]. The privatized Iris dataset is then sent to the KNN machine learning classifier for training and testing using 10 fold cross validation; the classification error is quantified. If the classification error is lower or equal to a threshold, then better utility might be achieved, otherwise, we adjust the data privacy parameters and re-classify the results.

STEP 1: Noise Addition
• Input dataset and add noise \((X + \varepsilon = Z)\), between \(\varepsilon \sim N(0, \sigma^2)\).

STEP 2: KNN Classification
• Input Privatized dataset for learning, use 10 fold cross validation.

STEP 3: Data Utility Evaluation
• Quantify the Classification error.
• If the threshold for lower error is achieved, publish data set, else adjust noise levels and go to Step 2.

STEP 4: Publish Privatized Dataset
• If the threshold for lower classification error is achieved, then publish privatized dataset.

III. EXPERIMENT

In our experiment, we used the Iris dataset from the UCI machine learning repository as our original dataset [9]. We then privatized the dataset by using the noise addition data privacy technique. We then used KNN classification and quantified the classification error. We adjusted the noise
levels and run the privatized dataset through the KNN classifier after which we published the results. We used MATLAB for both noise addition and KNN classification.

IV. RESULTS

As shown in our initial results, only 4 percent of records from the original Iris dataset were misclassified. When noise addition was chosen between the mean and standard deviation for the privatized dataset, 32 percent of records got misclassified. However, when noise addition was reduced to mean = 0 and standard deviation = 0.1 for the privatized dataset, 26 percent of records got misclassified, a 6 point reduction in classification error.

![Fig 1: KNN classification of the original Iris dataset with classification error at 0.0400 (4 percent misclassified data)](image)

![Fig 2: KNN classification of the privatized Iris dataset with noise addition between the mean and standard deviation.](image)

![Fig 3: KNN classification of the privatized Iris dataset with reduced noise addition between mean = 0 and standard deviation = 0.1.](image)

![Fig 4: A second run of the KNN classification of the privatized Iris dataset with reduced noise addition between mean = 0 and standard deviation = 0.1.](image)

V. CONCLUSION AND DISCUSSION

The initial results from our investigation show that a reduction in noise levels does affect the classification error rate. However, this reduction in noise levels could lead to low risky privacy levels. Finding the optimal balance between data privacy and utility needs is still problematic.

ACKNOWLEDGMENT

Special thanks to Dr. Claude Turner and the Computer Science Department at Bowie State University for making this work possible.

REFERENCES


Designing an Internet Traffic Predictive Model by Applying a Signal Processing Method

Soo-Yeon Ji · Seonho Choi · Dong Hyun Jeong

Received: 30 August 2013 / Revised: 9 September 2014 / Accepted: 22 September 2014 © Springer Science+Business Media New York 2014

Abstract Detection of abnormal internet traffic has become a significant area of research in network security. Due to its importance, many predictive models are designed by utilizing machine learning algorithms. The models are well designed to show high performances in detecting abnormal internet traffic behaviors. However, they may not guarantee reliable detection performances for new incoming abnormal internet traffic because they are designed using raw features from imbalanced internet traffic data. Since internet traffic is non-stationary time-series data, it is difficult to identify abnormal internet traffic with the raw features. In this study, we propose a new approach to detecting abnormal internet traffic. Our approach begins with extracting hidden, but important, features by utilizing discrete wavelet transformation. Then, statistical analysis is performed to filter out irrelevant and less important features. Only statistically significant features are used to design a reliable predictive model with logistic regression. A comparative analysis is conducted to determine the importance of our approach by measuring accuracy, sensitivity, and the Area Under the receiver operating characteristic Curve. From the analysis, we found that our model detects abnormal internet traffic successfully with high accuracy.

S.-Y. Ji (✉) · S. Choi
Department of Computer Science, Bowie State University, 14000 Jericho Park Rd, Bowie, MD 20715, USA
e-mail: sjj@bowiestate.edu

S. Choi
e-mail: schoi@bowiestate.edu

D. H. Jeong
Department of Computer Science and Information Technology, University of the District of Columbia, 4200 Connecticut Avenue NW, Washington, DC 20008, USA
e-mail: djeong@udc.edu

Published online: 30 September 2014
1 Introduction

The Internet is a globally distributed network that supports communications among various applications and computer systems that generate different network traffic patterns [1]. With the analysis of the network traffic patterns, we are able to identify the usage of network resources. Network administrators monitor network traffic to identify possible network congestion. If needed, reallocation of network resources is performed to guarantee reliable network communication. Therefore, we are able to communicate or share data seamlessly through the Internet. Since unusual activity may slow the network communication speed down, a study of identifying anomalous network traffic or behaviors has been regarded as one of the important researches in the network security community. In network monitoring, accurate and rapid abnormal internet traffic detection is critical [2]. Thus, anomalous network traffic or behaviors should be filtered out to guarantee smooth network communication.

Identification of the applications responsible for generating internet traffic is commonly performed by locating well-known service ports obtainable from network packet header [1]. Since numerous emerging applications and services do not use well-known ports, the technique of employing known ports (e.g. 80, 22, or else.) to create a tunnel to other applications is broadly adopted. Therefore, analyzing internet traffic based on known port numbers is no longer an effective approach. More specifically, a port-based classification is ineffective for identifying the usage of P2P applications. With the port-based classification, 30–70 % of internet traffic is classified as “unknown” [3]. Instead of using known ports, many current applications use dynamic ports. Due to the limitation of identifying network flows using service ports, researcher designed new methods by considering application payload signatures as a deep packet inspection method [4] and a payload-based method [5]. These methods directly compare stored signatures to the packets coming from applications. Since new applications are emerging and existing application protocols keep upgrading, the methods have a limitation of analyzing future internet traffic. Due to this limitation, a flow-based classification receives much attention [6, 7]. This approach performs a classification based on various flow features such as the number and size distributions of internet packets in a network flow, flow duration, and inter-packet arrival time [1, 8]. To overcome the shortcomings of the approaches that use port and signature information, researchers started using statistical methods to classify internet traffic flows. They mainly focused on identifying statistically valid characteristics from the traffic flows [1, 7–14]. For instance, Moore and Papagiannaki [7] generated more than 200 features from the Internet traffic data. Later, these features have been broadly used to perform extensive studies on identifying the best analytical approaches [1, 6, 9].

Imbalanced data often occurs in various domains including medical, biology, and computer networks [15]. In the network security community, network features or
best combinations of the features from imbalanced internet traffic data are used to identify abnormal behaviors. If a predictive model is designed with the imbalanced data, it cannot classify minority class successfully because the model determines all new incoming data as majority class [16]. In addition, most previous studies mainly focused on analyzing the internet traffic data by utilizing their actual values (i.e. raw data) as input features. Therefore, there might be a limitation of detecting any sudden changes within the data as abnormal traffic behavior.

In this paper, we propose a new predictive model for detecting network anomalies (i.e. viruses and worm attacks) in local area networks (LAN). Although wavelet analysis has been applied broadly for intrusion detection, we used it differently. Specifically, discrete wavelet transformation (DWT) is used to extract important patterns from the Internet traffic time series data to identify anomalous behaviors. In this study, we primarily focus on extracting meaningful features while maintaining the characteristics of the internet traffic by applying DWT, validating the statistical significance of the features using the statistical analysis system (SAS), generating a predictive model via logistic regression (LR), and evaluating statistical significance of the predicted model. In particular, the predictive model is generated with a balanced dataset. To identify the reliability of the model, we measured accuracy, sensitivity, specificity, and the Area Under the receiver operating characteristic Curve (AUC). The rest of this paper begins with explaining related work and our approach including a description of the dataset and methods. An explication of our results is provided in Sect. 4. Then, we conclude this paper with discussing our approach’s implications in Sect. 5.

2 Related Work

Detecting internet traffic abnormality has been studied intensely for many years [17, 18]. Researchers have applied numerous analysis techniques to analyze internet traffic. In particular, machine learning methods, such as Support Vector Machine (SVM) [19–22], Decision Tree algorithm [6], Neural Network [23, 24], and Bayesian Analysis [25], are used to generate actual classifiers. A common approach to generate actual classifiers is applying machine learning methods directly to raw features. For instance, packet inter-arrival time (IAT) is a frequently used feature for identifying either abnormal internet traffic or unexpected network activities [1, 8]. Since the packet IAT is a non-stationary signal, future internet behaviors can be predicted by analyzing it. If analyzed properly, significant underlying knowledge can also be detected.

Wavelet analysis is also widely used for internet traffic analysis because of its ability of identifying hidden patterns from time-frequency information by separating input data into different levels of frequencies. Applying the signal processing technique (i.e. wavelet analysis) to internet traffic helps us isolate the characteristics of the traffic by extracting hidden patterns of high and low frequency information [26]. Many researchers used wavelet analysis to identify network anomalies by reconstructing network traffic data [27, 28], compressing the data by applying two different thresholds from wavelet coefficients [29], and designing better wavelet
filters to identify better local frequency information [30]. In the work by Barford et al. [30], wavelet transformations were used to extract flow-based traffic abnormality by splitting the input signals into different ranges of frequencies (low, mid, and high frequencies). With the frequencies, a deviation was calculated to identify anomalies by determining a threshold from wavelet coefficients at different frequency levels. Kim et al. [27] studied a traffic abnormality technique. In this study, discrete wavelet transform is used to reconstruct the signal. The only selected level coefficients are used to reconstruct the signal. Then, threshold through statistical analysis is used for abnormal detection, based on discrete wavelet transforms. Also, Wavelet Packet Transformation (WPT) was broadly used to classify internet traffic flows [31–33]. Gao et al. [32] studied WPT-based network anomaly detection to enhance the capability of high and middle frequency information. Utilization of WPT requires a high-computational power because it analyzes whole data consecutively to produce different sets of coefficients of n-level of decomposition. Due to this complexity, this approach may not be appropriate to perform rapid internet traffic monitoring. Ramanarran [34] proposed Wavelet-based Attack Detection Signatures (WADeS) to detect internet attacks by using variances of corresponding wavelet coefficients. As shown in Table 1, various techniques (as major steps) were used to detect network anomalies. Although threshold techniques or coefficient measures were commonly applied to determine input features, our approach utilizes a statistical measure to identify statistically significant features and use them to design a predictive model.

Most previous studies that utilized wavelet transform techniques focused on detecting abnormal internet traffic patterns by performing data reconstruction, two or three frequency range analysis, and filtration (or thresholding). Since one of the best advantages of applying wavelet analysis is the possibility of extracting information at different levels of signal decomposition (i.e. frequencies), extracting valuable features to discover underlying patterns is a crucial step for identifying network abnormal behaviors. However, it has been known that existing network anomaly detection methods based on wavelet transformations have a limitation of using low frequency anomaly [32]. In this paper, we address this important consideration when designing a predictive network abnormal behavior model.

3 Approach

Our approach begins with applying normalization to the input data attributes as a pre-processing step. Then, discrete wavelet transformation is used to extract features. Since it is important to perform a feature validation to enhance the performance of identifying hidden abnormal behaviors, statistical analysis (ANOVA—analysis of variance) is applied to identify statistically significant features that are later used to generate a predictive model. To determine the effectiveness of the generated predictive model, the reliability of the model is tested by calculating the AUC. Figure 1 represents the overall procedure of our proposed prediction model. A detailed explanation about our method is included in Sect. 3.2.
<table>
<thead>
<tr>
<th>Work</th>
<th>Goal</th>
<th>Dataset(s)</th>
<th>Used features</th>
<th>Major step(s)</th>
<th>Used algorithm(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alarcon-Aquino and Barria [26]</td>
<td>Network anomaly detection</td>
<td>(1) Simulated data (2) BTnet Dial IP service data</td>
<td>(1) Simulated data features and (2) set of network metrics</td>
<td>Estimation of Nuisance parameters by maximum likelihood using wavelet coefficients</td>
<td>Undecimated DWT and Bayesian</td>
</tr>
<tr>
<td>Kyriakopoulos and Parish [29]</td>
<td>Network anomaly detection</td>
<td>CAIDA dataset on the Witty worm</td>
<td>Not known</td>
<td>Two thresholds for the compression and event detection tasks</td>
<td>Haar wavelet, adaptive threshold, and Donoho–Johnstone universal threshold (aka VisuShrink)</td>
</tr>
<tr>
<td>Barford et al. [30]</td>
<td>Network anomaly detection</td>
<td>Simulated data</td>
<td>Decomposed three distinct signals (low/mid/high) from SNMP and IP flow dataset</td>
<td>Deviation scores from three range (low, mid, and high) frequency information</td>
<td>Wavelet analysis and Time frequency-localization</td>
</tr>
<tr>
<td>Dainotti et al. [35]</td>
<td>Network anomaly detection</td>
<td>(1) DARPA99 (2) D-WARD (3) UNINA</td>
<td>Traffic traces (packet rates) and mean and std for the traces</td>
<td>Generate anomaly profiles</td>
<td>Adaptive threshold, cumulative sum, and Continuous wavelet transform</td>
</tr>
<tr>
<td>Lu and Ghorbani [28]</td>
<td>Network anomaly detection</td>
<td>DARPA99</td>
<td>15 Features from TCP, UDP, ICMP data</td>
<td>ARX model using wavelet approximation coefficients</td>
<td>DWT and AutoRegressive with eXogenous (ARX)</td>
</tr>
<tr>
<td>Kim et al. [27]</td>
<td>Network anomaly detection</td>
<td>(1) USC traces (2) Simulated virtual attacks on the University of Auckland traces</td>
<td>Duration, persistency, IP address</td>
<td>Coefficient-selective reconstruction in DWT</td>
<td>DWT and Threshold</td>
</tr>
<tr>
<td>Callegari et al. [31]</td>
<td>Network anomaly detection</td>
<td>(1) DARPA99 (2) UNINA (3) D-WARD</td>
<td>Number of IP packets received</td>
<td>Euclidean distance between coefficients</td>
<td>Wavelet Packet Transform (WPT)</td>
</tr>
<tr>
<td>Gao et al. [32]</td>
<td>Network anomaly detection</td>
<td>Simulated dataset</td>
<td>Data packets</td>
<td>A scale-adaptive method</td>
<td>WPT</td>
</tr>
<tr>
<td>Work</td>
<td>Goal</td>
<td>Dataset(s)</td>
<td>Used features</td>
<td>Major step(s)</td>
<td>Used algorithm(s)</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------</td>
<td>-------------------</td>
<td>--------------------------------------</td>
<td>---------------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>Tan et al. [33]</td>
<td>Classifying network applications</td>
<td>Simulated dataset</td>
<td>69 Calculated features from TCP and UDP packets</td>
<td>Back-propagation (BP) neural network, Particle and swarm optimization</td>
<td>Wavelet packet decomposition (WPD)</td>
</tr>
<tr>
<td>Ramanathan [34]</td>
<td>Detection of DoS attack</td>
<td>NLANR data</td>
<td>IP address, Packet size, Time stamp</td>
<td>Thresholding and wavelet variance computation</td>
<td>DWT</td>
</tr>
</tbody>
</table>

DARPA99: 1999 DARPA intrusion detection dataset
UNINA: University of Naples “Federico II” traffic trace dataset
D-WARD: UCLA Packet trace dataset
Only the researches utilizing wavelet transformations are presented
3.1 Dataset

In this study, a publicly available internet traffic dataset is used [36]. The dataset is generated from both network link directions on Genome campus with three institutions on the site. A detailed explanation about this dataset can be found in [7, 36]. Since the dataset provides classification information about packets, researchers used this dataset to design new approaches to detect abnormal behaviors [1, 6, 9]. The data used in this study includes computed mean, median, maximum, minimum, and a variance of packet IATs. Throughout this paper, we call this data “raw features.”

Most previous studies commonly used imbalanced internet traffic data when designing a predictive model to detect abnormal behaviors. If a predictive model is generated with imbalanced datasets, it can detect a majority class while maintaining good accuracy. However, it has been known that it is difficult to detect a minority class due to the influence of the large majority class [16]. Specifically, it cannot classify a new incoming minority class correctly. Researchers found that the performance of existing classifiers tends to be biased towards the majority class because of unequal class distribution [37]. To overcome this limitation, balanced dataset is used to avoid possible bias caused by the majority class. In the dataset, three classes (P2P, Mail, FTP) are classified as “normal” behavior and the class (attack—virus and worm attacks) is considered as “abnormal” behavior. The original dataset is an imbalanced dataset. That is, the normal behaviors are considered as the majority class and the abnormal behavior is regarded as the minority class. To balance the normal and abnormal data, the same sample size of the normal and abnormal data is used for this study. The abnormal (i.e. attack) data contains m number of samples. To balance the normal and abnormal data, the same sample size of the normal data is used. That is, a total N number of normal samples are divided into k disjoint datasets as \( n_1, n_2, n_3, n_4, \ldots, n_k \) so that \( n_i \in N, i = 1, 2, 3, \ldots, k \) and \( n_i \cap n_j = \emptyset \). Each \( n_i \) dataset includes the same numbers of abnormal data. Thus, the balanced datasets \( d_i (i = 1, 2, \ldots, k) \) are generated by combining both normal and abnormal datasets (see Table 2).

In our study, six packet IAT information (see Table 3) is used to address the utilization of our proposed approach integrating both the signal processing technique and LR for internet traffic monitoring.
3.2 Method

3.2.1 Pre-processing

As a pre-processing step, data normalization is performed. Data normalization scales the values of each continuous attribute into a well-proportioned range so that one attribute cannot affect others. Several network variables in the network traffic dataset have large variances among them. Data normalization needs to be applied to remove such large variances. There are four data normalization approaches that are broadly utilized for network traffic data analysis as mean range, statistical normalization, ordinal normalization, and frequency normalization [38]. Mean range is a normalization technique (broadly known as min-max normalization) that performs the normalization after identifying the minimum and maximum values of given attributes. Statistical normalization is an approach of maintaining standard normal distribution. Ordinal normalization is to rank the continuous value of an attribute. Frequency normalization normalizes an attribute by considering it to the summed value proportionally. In our study, we applied the mean range normalization technique (i.e. min-max normalization) because it is a simple, effective, and relatively inexpensive technique for improving the overall performance of network abnormality detection.

<table>
<thead>
<tr>
<th>Table 2</th>
<th>A description of used normal and abnormal datasets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class</td>
<td>Dataset1</td>
</tr>
<tr>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>FTP</td>
<td>210</td>
</tr>
<tr>
<td>Mail</td>
<td>209</td>
</tr>
<tr>
<td>P2P</td>
<td>209</td>
</tr>
<tr>
<td>Total</td>
<td>628</td>
</tr>
<tr>
<td>Abnormal</td>
<td></td>
</tr>
<tr>
<td>Attack</td>
<td>628</td>
</tr>
</tbody>
</table>

Three classes (i.e. P2P, Mail, FTP) represent normal behaviors and one class (i.e. attack) indicates abnormal behavior. Each number in this table indicates the number of samples (i.e. data records).

<table>
<thead>
<tr>
<th>Table 3</th>
<th>Six inter-arrival time (IAT) features for all packets (considering both directions) are used in this study</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature name</td>
<td>Description</td>
</tr>
<tr>
<td>(f1) min IAT</td>
<td>Minimum packet inter-arrival time</td>
</tr>
<tr>
<td>(f2) q1 IAT</td>
<td>First quartile inter-arrival time</td>
</tr>
<tr>
<td>(f3) med IAT</td>
<td>Median inter-arrival time</td>
</tr>
<tr>
<td>(f4) mean IAT</td>
<td>Mean inter-arrival time</td>
</tr>
<tr>
<td>(f5) max IAT</td>
<td>Maximum packet inter-arrival time</td>
</tr>
<tr>
<td>(f6) var IAT</td>
<td>Variance in packet inter-arrival time</td>
</tr>
</tbody>
</table>

3.2 Method
3.2.2 Feature Extraction

Discrete wavelet transform is a broadly known promising method for time-frequency analysis. Since wavelet indicates a small localized wave in a time domain, any sudden or rapid changes in the data can be identified easily. Because of the characteristics that DWT has, it is advantageous to analyze non-stationary signal data (e.g. internet traffic data) with identifying significant internet traffic patterns. DWT decomposes the input data into different levels of frequency components by calculating its correlation with a set of chosen wavelet basis functions \[26, 39, 40\]. The ability of preserving both time and frequency resolutions has led to widespread use of DWT in many practical application domains \[16\]. It is particularly good for local analysis in representing fast time varying and non-stationary signals like internet traffic data. The merits of using DWT are (a) analyzing non-stationary time series data (e.g. internet traffic data), (b) capturing the non-stationary nature of the data in time-frequency domain, (c) detecting any rapid changes in the data, and (d) revealing important information from the data.

In our study, a db4 basis function that belongs to the Daubechies family is used as a wavelet basis function. A three-level decomposition with a db4 wavelet is applied to the internet traffic data utilizing an overlapping sliding window to examine rapid changes in the data. Four windows sizes (sizes of 25, 50, 100, and 150 data points) with a 70 % overlap are tested. Their results are compared to determine the most appropriate window size for internet traffic data analysis. By applying DWT, three different levels of detail coefficients (detail level 1, detail level 2, and detail level 3) and approximate coefficients are measured. Three basic statistical features (i.e. standard deviation and median) are calculated from the coefficients. The extracted DWT features \(f_1, f_2, \ldots, f_n\) are used as input features to generate a predictive model. Figure 2 shows how features are extracted.

Among the extracted DWT features, feature selection is performed by examining the significance of all features. To validate their significance, a statistical analysis is performed using the SAS. In particular, one-way ANOVA is performed to identify...
important features using two classes (Normal vs Abnormal) by maintaining the statistical significance ($p < .05$). From the analysis, only significant features are selected and used to generate a predictive model.

### 3.2.3 Generating a Predictive Model Using LR and Its Validation

The primary objective of this study is to generate a reliable, predictive model to detect abnormal behaviors. To generate such model, valuable features are extracted and used as input for LR. LR is particularly useful when the class is dichotomous (e.g. normal/abnormal) to measure the probability of classes. The logit function calculates the expected probability of a dichotomy as:

$$
\pi_i = pr(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots)}}
$$

where $X_i$ is a variable with numeric value, $\pi_i$ is the outcome (dichotomous; 0/1, e.g., normal/abnormal), and the $\beta_i$s are the regression coefficients that quantify the contributions of the numeric variables to the overall probability [41–43]. Unlike most regression analyses, LR does not need to assume data distributions on variables. Due to this benefit, it is commonly used when outcome is a nominal variable.

To test the reliability of the predictive model, the AUC is computed. In addition, an evaluation is conducted to see the effectiveness of using the DWT features versus the raw features for generating a predictive model.

To validate the performance of the predictive model, cross-validation is applied. Cross-validation [44] is commonly used to estimate the ability of a statistical classifier (i.e. the performance on previously unseen data [45]). With the cross-validation, the data is divided into $k$ disjoint sets. Each is on a different combination of $k$ partitions and is used for training. The remaining $k - 1$ partition is used for testing. In particular, leave-one-out cross-validation [45] is applied in this study. Leave-one-out cross-validation is one of the widely used cross-validation methods due to its mathematical simplicity. It provides an almost unbiased estimate of the generalization ability of a classifier [46, 47]. In this study, the leave-one-out cross-validation is applied to each dataset. Performances including accuracy, sensitivity, and specificity. Since the AUC has been proven to be highly reliable for evaluating classifiers [48], the AUC measurement is also performed to determine the performance of the classification (i.e. detecting the abnormal internet traffic behavior).

## 4 Results

This section presents the findings obtained from our study. In Sect. 4.1, we show a comparison between the raw and the DWT features by emphasizing the effectiveness of utilizing the DWT features. Since it is important to determine the usefulness of our proposed model for detecting abnormal behaviors, a performance
comparison is conducted by calculating accuracy, sensitivity, specificity, the ROC curve, and the AUC (see Sect. 4.2 for detail).

4.1 Feature Comparison

A comparative performance study was performed between the raw and the DWT features. As mentioned in Sect. 3.1, the five balanced datasets were used. We extracted a total of 48 wavelet features from the datasets. From all of the features, 42 features were determined as statically significant features ($p < 0.05$). The rest six features were statistically insignificant. They are the standard deviation of level three for $q1IAT$ ($p = 0.4880$), the median of level three for $q1IAT$ ($p = 0.1942$), the standard deviation of level three for $medIAT$ ($p = 0.5889$), the median of level one for $meanIAT$ ($p = 0.1305$), the median of approximation coefficient for $meanIAT$ ($p = 0.4595$), and the standard deviation of approximate coefficients for $maxIAT$ ($p = 0.9986$). As presented in Table 3, six raw features were used in this study.

To determine how well the features could separate the classes (i.e. attack, P2P, FTP, and Mail), the raw and the DWT features were compared across the classes by calculating their average values. Figure 3 shows two graphical representations that provide the average values of the raw and the DWT features. More specifically, Fig. 3 represents (a) the six raw features and (b) the standard deviation of the level one wavelet feature. As shown in Fig. 3a, $f3$ (first quartile IAT) cannot differentiate between ‘Mail’ and ‘Attack’. However, $f3$ separates all classes clearly when using its DWT feature (see Fig. 3b).

4.2 Classification Performance Comparison

As explained above, our proposed predictive model was designed with LR. Since the DWT features were used as inputs to LR, a classification performance comparison was conducted by generating a predictive model with the raw features. A comparison between the two models was done by computing accuracy,
sensitivity, specificity, and the AUCs. As an additional performance comparison, our proposed predictive model was compared to other known approaches, such as the Neural Network and SVM.

**ROC measure**—To validate the reliability of the proposed method in identifying the outcome (normal/abnormal) of the internet traffic data, a receiver operating characteristic (ROC), or simply the ROC curve, was computed. The ROC curve indicates a plot of sensitivity (i.e. true positive rate) versus 1-specificity (i.e. false positive rate). For reliability measure, the ROC curve for the predictive model with the raw features was compared. Figure 4 depicts the ROC curves of the predictive models designed with (a) the raw and (b) the DWT features. Figure 4 represents that our proposed predictive model shows a better performance when compared to the model with the raw features.

**Performance comparison**—A performance was evaluated by computing four statistical measures (accuracy, sensitivity, specificity, and the AUC). Figure 5 represents a performance comparison between the predictive model generated with the raw features (Fig. 5a) and the DWT features (Fig. 5b). The predictive model
with the DWT features provided outperformed results in accuracy, sensitivity, specificity, and AUC. Among the four sliding window sizes (25, 50, 100, and 150 data points), the 150 data points sliding window showed a better performance than others.

A standard error mean (SEM) is the standard deviation of the sampling distributions that measures the variability of predicted values. If the SEM value is small, it indicates that the observed values are fairly close to each other. Table 4 presents SEMs of the performance results including accuracy, sensitivity, specificity, and AUC. From the table, we can conclude that there would be a higher chance of obtaining less error when predicting future values.

Table 4 A standard error mean comparison of the predictive model using the raw and the DWT features

<table>
<thead>
<tr>
<th>window size (i.e. data points)</th>
<th>The DWT features</th>
<th>The raw features</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>25</td>
<td>50</td>
</tr>
<tr>
<td>Accuracy</td>
<td>1.107</td>
<td>0.611</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>0.712</td>
<td>0.635</td>
</tr>
<tr>
<td>Specificity</td>
<td>1.810</td>
<td>0.525</td>
</tr>
<tr>
<td>AUC</td>
<td>0.377</td>
<td>0.409</td>
</tr>
</tbody>
</table>

Table 5 A performance comparison between our proposed method (using LR) and other broadly known approaches (SVM and NN)

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
<th>Sensitivity</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>With raw features</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Logistic regression (LR)</td>
<td>84.1 ± 1.52</td>
<td>95.9 ± 0.40</td>
<td>80.3 ± 0.69</td>
</tr>
<tr>
<td>Neural network (NN)</td>
<td>75.4 ± 1.73</td>
<td>81.3 ± 2.04</td>
<td>81.6 ± 1.94</td>
</tr>
<tr>
<td>Support vector machine (SVM)</td>
<td>76.4 ± 2.81</td>
<td>75.4 ± 3.23</td>
<td>75.6 ± 2.82</td>
</tr>
<tr>
<td><strong>With DWT features</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Logistic regression (LR)</td>
<td>97.6 ± 0.61</td>
<td>97.8 ± 0.63</td>
<td>98.6 ± 0.40</td>
</tr>
<tr>
<td>Neural network (NN)</td>
<td>96.7 ± 0.51</td>
<td>96.7 ± 0.51</td>
<td>96.7 ± 0.55</td>
</tr>
<tr>
<td>Support vector machine (SVM)</td>
<td>83.9 ± 5.21</td>
<td>85.2 ± 5.69</td>
<td>89.4 ± 4.75</td>
</tr>
</tbody>
</table>

Each value indicates mean ± standard error mean (SEM)

with the DWT features provided outperformed results in accuracy, sensitivity, specificity, and AUC. Among the four sliding window sizes (25, 50, 100, and 150 data points), the 150 data points sliding window showed a better performance than others.

A standard error mean (SEM) is the standard deviation of the sampling distributions that measures the variability of predicted values. If the SEM value is small, it indicates that the observed values are fairly close to each other. Table 4 presents SEMs of the performance results including accuracy, sensitivity, specificity, and AUC. From the table, we can conclude that there would be a higher chance of obtaining less error when predicting future values.

Table 4 indicates that the predictive model (with the DWT features) showed a better performance than with the raw features. In particular, the model specificity using the raw features was much higher than the DWT features in all four sliding windows. From the reliability measure (i.e. AUC), we found that the predictive model with DWT features provides a better ability of detecting abnormal internet traffic behaviors. We also found that the reliability of the predictive model with the DWT features is almost linearly increasing when the window size is incremented.

Since it is important to perform a comparison with other approaches to determine the effectiveness of our proposed model, we conducted a study with broadly known techniques, such as Neural Network (NN) and SVM. Specifically, we measured accuracy, sensitivity, and AUC (see Table 5). From the performance comparison,
we identified that the accuracy, sensitivity, and AUC of our method were higher compared to other approaches (NN and SVM). This explains that our predictive model (via LR) is good for identifying abnormal behaviors more accurately. In addition, we also found that utilization of the DWT features is important for increasing the performance of detecting anomalous network traffic or behaviors.

### 5 Discussion and Conclusion

This study examines the capabilities of utilizing (1) Logistic Regression (LR) and (2) a DWT-based feature extraction for designing a predictive model to detect abnormal internet traffic behaviors. While many previous studies used DWT for determining threshold and reconstructing the data, we utilized DWT to extract important patterns from network traffic. With statistically significant DWT features, we designed a predictive model using LR. Although LR is a broadly known parametric technique for binary classification, it is not advised to use this technique when classes are unbalanced because the conditional probability of a rare class can be underestimated [49]. To design a robust predictive model, a balanced dataset is used to avoid possible bias caused by a majority class. To show the effectiveness of utilizing the balanced dataset, a comparative study was performed by measuring false negatives (FN) and false positives (FP) [50]. We measured Type I and II errors (i.e false positive and false negative, respectively) when using the balanced and imbalanced datasets. FP indicated that the actually abnormal class was predicted as normal. FN indicated that the actually normal class was predicted as abnormal. Table 6 shows the comparison between the balanced and imbalanced datasets with the proposed predictive model.

We found that the Type I error (FP) shows a higher mean and standard error mean when the imbalanced data is used. Interestingly, the Type II error (FN) was slightly high when using the balanced dataset. However, the difference was minor compared to the Type I error (FP). This is because the imbalanced dataset has more normal data than abnormal data. Thus, when a predictive model is generated with the imbalanced dataset, there will be a higher chance of detecting normal behaviors. Additionally, it is less likely to detect abnormal behaviors correctly for new incoming attack due to the lack of learning from the training dataset. We found that utilization of the balanced dataset presents more chances to detect abnormal behaviors for new incoming data.

To determine the effectiveness of our predictive model, a comparative analysis was performed with the model generated with the raw features. Since DWT has abilities of dealing with non-stationary signals and extracting different level of

![Table 6 Type I and II errors when utilizing either the balanced or imbalanced datasets (mean ± SEM)](image)

<table>
<thead>
<tr>
<th></th>
<th>Type I error (FP)</th>
<th>Type II error (FN)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Imbalanced</td>
<td>0.4878 ± 0.1329</td>
<td>0.0358 ± 0.0128</td>
</tr>
<tr>
<td>Balanced</td>
<td>0.0456 ± 0.0094</td>
<td>0.0830 ± 0.0130</td>
</tr>
</tbody>
</table>
frequency information as well as their specific local information in time, it is good for identifying abnormal internet traffic behaviors (see Fig. 5). To validate the reliability of the models generated with the raw and the DWT features, the AUC is calculated. From the AUC results, we identified that the model with the DWT features is much more reliable than the model with the raw features. This explains that the DWT features well preserve important characteristics of abnormal internet traffic behaviors. It is important to note that the performance difference when utilizing the raw and the DWT features could be attributed to the difference of the amount of information taken as input to generate a predictive model.

From the experimental study, we found that the DWT has an ability of extracting underlying information from the internet traffic signals. Additionally, we identified that the DWT features show a better performance than the raw features. We also found that most of the DWT features were significant (87.5 % of the DWT features were significant), while only two out of the six raw features (33.3 %) were significant. From the test of four windows sizes (25 data points, 50 data points, 100 data points, and 150 data points), we noticed that when the window size gets bigger in the DWT features, the SEM turns smaller. When we compared the small size (25 data points) to the bigger window size (150 data points), we found about 13.5 % accuracy, 10 % sensitivity, 4.8 % specificity, and 11.8 % AUC differences. Although we identified that the performance with the small window size does not perform better than with the bigger window size, it is important to note that the small size window would be more applicable in a real network environment. Finally, we found that it is important to perform a statistical validation on features for detecting abnormal internet traffic patterns accurately.

In this paper, we extracted various DWT features from the Internet traffic data and utilized them to generate a predictive model. However, it is important to identify more significant features to strengthen our designed predictive model. Therefore, in future works, we plan to identify more informative features. To increase the accuracy and stability for detecting abnormal network behaviors in a different network environment, we are going to utilize more network features, such as IP addresses, protocols (including port information), network services (e.g. HTTP, Telnet, SSH, or else.), and TCP flags. Although numerous network detection techniques have been proposed, it is still difficult to determine specific attack types. We plan to extend our research to detect attack types including DDoS (distributed denial-of-service), Buffer overflow attack, Surveillance sweep, or others.

**Acknowledgments** This study is based on the work supported by US Army Research Office (ARO) Grant W911NF1310143.

**References**


Soo-Yeon Ji is an Assistant Professor in the Department of Computer Science at Bowie State University. Her research interests include computer-aided medical decision, robust authentication using physiological signals, biomedical signal analysis, and bioinformatics. She has a Ph.D. in Computer Science from Virginia Commonwealth University.

Seonho Choi is a Professor in the Department of Computer Science at Bowie State University. His research interests include network communication, TCP protocol, and cyber security. He has a Ph.D. in Computer Science from the University of Maryland at College Park.

Dong Hyun Jeong is an Assistant Professor in the Department of Computer Science and Information Technology at the University of the District of Columbia. His research interests include information visualization, visual analytics, human-computer interaction, and information security. He has a Ph.D. in computer science from the University of North Carolina at Charlotte.
Designing a Two-Level Monitoring Method to Detect Network Abnormal Behaviors

Soo-Yeon Ji  
Bowie State University  
Bowie, Maryland 20715–3319  
Email: sji@bowiestate.edu

Seonho Choi  
Bowie State University  
Bowie, Maryland 20715–3319  
Email: schoi@bowiestate.edu

Dong Hyun Jeong  
University of the District of Columbia  
Washington, DC 20008–1122  
Email: djeong@udc.edu

Abstract—Monitoring network traffic behavior is very critical for securing computing infrastructures. In this paper, we focus on enhancing the way of detecting anomalous network traffic behaviors by proposing a new two-level detection method that consists of abnormality detection and exact attack type identification. The abnormality detection is performed with the rules generated by Classification and Regression Trees (CART). Then, Support Vector Machine (SVM) is applied to design a predictive model to identify exact attack types (among DoS, U2R, R2L, and Probes). Since feature extraction is an important step for designing an efficient predictive model, we used Higuchi fractal dimension and statistical measures (mean, median, and standard deviation) with an overlapping sliding window operation to extract features. Among the extracted features, only significant features are selected by applying statistical analysis and used to design a predictive model. As results, we found that our approach shows about 80.03% accuracy in detecting network abnormal behaviors. From a comparative study, we concluded that our proposed SVM-based predictive model is superior to a broadly known NN-based predictive model for identifying exact attack types.

I. INTRODUCTION

Since the Internet become an important part of our daily lives, it is important to secure our computing resources and infrastructures from any external cyber threats. Due to this importance, numerous researchers studied on understanding and detecting abnormal network behaviors. They proposed techniques to differentiate anomalous network patterns from heavy network traffic load situations. A traditionally known approach to network anomaly detection is based on utilizing attack signatures [1]. Although this signature-based technique produces a lower false alarms rate, it is not as effective as for detecting unknown anomalous network attacks because signature-based detection systems identify network anomalies by referencing built-in attack signatures. If signatures do not exist in the signature-based detection systems, incoming network anomalies cannot be detected. Therefore, signatures need to be updated to the systems continuously to improve the chance of detecting new types of attacks.

Feature-based detection technique is proposed to address the limitation that exists in the signature-based detection systems. It identifies network anomalies by examining network traffic features. Commonly used network traffic features are IP addresses, source/destination port numbers, and TCP flags. Known feature-based techniques detect anomalous network traffics by observing and comparing them to the features in normal traffic conditions. However, these techniques fall into drawbacks of identifying anomalous network traffic patterns correctly since the patterns include volatile information to hide their uniqueness. For instance, IP address is a unique number assigned to each computer on the network. This IP address information can be changed or hidden by hackers when penetrating network infrastructures. Although several techniques are proposed to identify different characteristics from network traffic patterns, this is still a known research challenge. A detailed explanation about known feature-based network anomaly detection techniques is included in Section 2 Literature Review.

In this paper, we propose a two-level network abnormality monitoring model to identify anomalous network behaviors. First, rules are generated to determine outcomes (normal/abnormal). Then, a predictive model is designed to identify exact attack types. Specifically, a decision tree is used to generate reliable rules and machine learning (ML) is applied to build a predictive model with utilizing only statistically significant features. With this predictive model, we found that it is possible to identify exact abnormal attack types among denial-of-service (DoS), unauthorized access from a remote machine (R2L), unauthorized access to local administrative privileges (U2L), and surveillance and other probing (Probes).

The rest of this paper consists of four sections. Literature review on network anomaly detection is included in Section II. Section III begins with explaining the network traffic data we used for this study. Then, a detailed explanation about our proposed approach is provided. We conclude this paper after providing experimental results and future works.

II. LITERATURE REVIEW

As mentioned above, there are two globally known techniques in network anomaly detection as signature-based and feature-based detection techniques [2]. The signature-based technique is a common method used in Intrusion Detection System (IDS) that can identify attacks with referencing known signature patterns. Although signature-based approach is good for identifying network anomaly matched to pre-defined signatures, it has a limitation of detecting unknown network anomalies. To avoid this limitation, the feature-based technique is used because it does not require any prior knowledge. Numerous studies have been performed to identify effective approaches for extracting anomalous network traffic features. Among them, statistical methods and machine learning techniques are broadly used to identify anomalous network traffic features.
In the past, researchers studied on increasing the rate of detecting network attacks. Cheng et al. [3] proposed an approach of identifying normal TCP flows by using spectral analysis techniques to protect legitimate TCP flow from Denial of Service (DoS) attacks. Wang et al. [4] proposed statistics-based approach to detect TCP SYN flood attacks, which uses a nonparametric cumulative sum (CUSUM) method. Utilization of statistical approaches is good for maintaining high accuracy with spending reasonably short detection times because it approximately calculates normal traffic patterns to perform a comparison with abnormal traffics. However, it has a difficulty of detecting anomalies caused by network system failures. To resolve this difficulty, Thottan and Ji [5] used a statistical data analysis method with a signal processing technique together to quantify network behaviors to understand network anomalies. They classified network anomalies into two categories as network performance anomalies (e.g. file server failures, paging across the network, broadcast storms) and security-related problems or attacks. They showed that their approach of integrating a signal processing technique is effective for detecting several network anomalies. However, there was no accurate statistical model that was utilized to detect different abnormal traffic patterns. Due to this limitation, researcher started applying various techniques including neural networks, machine learning techniques, data mining, and so forth.

Artificial neural network has been applied broadly because it has a potential of identifying and classifying unknown network activities [6]. Lippmann and Cunningham [7] utilized neural networks to design a detection model by searching for attack-specific keywords in network traffic. Sarasamma et al. [8] used multilevel hierarchical Kohonen Net (K-Map) consisting of three layers to determine different types of attacks. To increase the speed of selecting features, input dataset is divided into three feature sets based on domain knowledge. After applying single-layer K-Map onto each feature set, significant subset features are determined and used to design next hierarchical K-Map. Hand and Cho [9] proposed an approach of employing an evolutionary neural network (ENN) to overcome the limitation of designing a precise topology (i.e. domain specific neural network model) for detecting network attacks. Support Vector Machine (SVM) is also used to classify abnormal network behaviors. Since SVM supports both supervised and unsupervised learning, Shon and Moon [10] applied hybrid approach of integrating the two learning methods with emphasizing the advantages of utilizing both SVM approaches. Similarly, Jain and Abouzakkar [11] designed an approach by utilizing both Hidden Markov Model (HMM) and Support Vector Machine (SVM).

Although numerous network anomaly detection method were proposed, there is no unique solution that maintains higher detection rate and lower false positive and negative rates. Traditional approaches to network anomaly detection utilize information that is directly extracted from network packet header. However, to increase the accuracy of detecting network anomalies, integration with computational feature extraction techniques is emphasized [12]. Shon and Moon [10] used Genetic Algorithm (GA) to extract optimized information from raw internet packets. Jain and Abouzakkar [11] applied J48 decision tree algorithm to determine significant features for anomaly intrusion detection. Hofmann et al. [12] proposed an approach with the combination of evolutionary algorithm (EA) and radial basis-function networks (RBFN). The evolutionary algorithm is used to select an appropriate feature subset, optimize the number of hidden neurons, determine a number of training epochs, and choose a basis function type. Principal Component Analysis (PCA) and Singular Value Decomposition (SVD) are also broadly applied techniques for feature extraction [13]. Most proposed techniques utilize characteristics of network traffics to identify abnormalities precisely. But, performing the real-time network anomaly detection with maintaining higher accuracy is limited due to the complex nature of network traffics. In this paper, we primarily focus on enhancing the way of detecting network anomalies by integrating both a rule extraction technique and a predictive model.

III. METHODOLOGY

Identifying abnormal behavior from network traffic is critical to maintaining a network environment secure. Abnormal behaviors detection should be performed with satisfying performance and reliability. Specifically, detailed information about detected abnormal behaviors should be provided when notifying detected abnormal behaviors to protect computing infrastructures efficiently. With our proposed predictive model, identification of exact attack types is performed to address this need. Our approach begins with generating reliable rules for detecting network abnormality (normal/abnormal) with classification and regression trees (CART). Once its abnormality is detected, an over-lapping sliding window operation is applied to extract features. Although the extracted features might contain the characteristics of abnormal behaviors, it is important to utilize only significant features to increase the accuracy of determining exact types of abnormal behaviors. In our study, we used Statistical Analysis System (SAS) to identify significant features, with which our proposed predictive model is designed. Figure 1 illustrates how the predictive model is designed. A detailed explanation about each step is included in the following subsections.

![Fig. 1. The entire process of the proposed predictive model consisting of two levels.](image)

A. Dataset

In this study, we used a new version of KDD dataset, NSL-KDD dataset, which is publicly available for researchers [14],
Since the KDD Cup’99 intrusion detection dataset includes a lot of redundant records, this redundancy causes an inefficient learning process. For instance, the difference between unauthorized access from a remote machine (R2L) and unauthorized access to local administrative privileges (U2L) is not clear. Due to this reason, it is difficult to distinguish them precisely. To resolve this issue, repeated records are removed from the KDD Cup’99 dataset [14]. We used total 125,973 records for training. For the testing, the rest of the data records (22,544 records) were used. The NSL-KDD dataset includes total 41 attributes (three nominal, six binary, and thirty-two numeric attributes) and four types of attacks (Denial of service (DoS), user to root (U2R), remote to user (R2L), and Probes). DOS attack indicates attempts to disabling network access to remote machines (or computing resources). R2L represents that a remote user gains access to local user accounts by sending packets to a computing machine over the network. U2R explains that an attacker accesses normal users’ accounts by exploring the system as a root-user. Probing represents that the network is scanned to gather information to find known vulnerabilities.

Table I shows the sampling distribution of the NSL-KDD training and testing datasets. As shown, the total number of attacks are not equally distributed in the training and testing datasets. For instance, 0.08% and 12.8% of R2L attacks are included in the training and testing datasets, respectively.

### Table I. Data Distribution of the NSL-KDD Training and Testing Datasets.

<table>
<thead>
<tr>
<th></th>
<th>Normal</th>
<th>Abnormal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>53.50%</td>
<td>43.10%</td>
</tr>
<tr>
<td>DoS</td>
<td>37.20%</td>
<td>33.10%</td>
</tr>
<tr>
<td>U2R</td>
<td>0.08%</td>
<td>0.20%</td>
</tr>
<tr>
<td>R2L</td>
<td>10.70%</td>
<td>10.80%</td>
</tr>
<tr>
<td>Probes</td>
<td>0.04%</td>
<td>12.80%</td>
</tr>
</tbody>
</table>

Also, it uses an exhaustive search of all variables and split values to find optimal splitting values for each node. This splitting stops at the pure node containing fewer examples. Advantages of using CART are a) it does not require any distributional assumptions for dependent and independent variables, b) it deals with multiple types of numerical and categorical variables as inputs and outputs, c) it is not affected by outliers, and d) it efficiently handles high dimensional data. Among the generated trees with CART, only the trees that give high training accuracy are selected. Rules are extracted from the selected trees and tested with the testing dataset. With these rules, abnormality of network traffic is determined.

#### Level 2: Attack Type Identification
The abnormal monitoring depends on identifying network traffic patterns using transparent rules. In this level, we focus on identifying detailed information about abnormal behaviors by generating a predictive model. When generating a model, utilization of feature extraction from the input data is important because features may include informative knowledge representing hidden, but important patterns of the input data.

We used both fractal domain and statistical measurements to extract features. A sliding window (=20 data points) with 67% of overlapping is applied to examine the variation of network traffic flow. Fractal dimension (FD), called a non-linear dynamical method, is based on fractal theory concept. This method is applied broadly in many areas including biology, image segmentation, audio signal analysis, and medicine [20], [21], [22]. FD is a useful method for detecting rapid variations from data [21]. It also presents a self-similarity measurement of data. The self-similarity can be connected with “fractals” and the fractals within the network traffic display shape similarities in time scales. Since fractal features express the degree of self-similarity of the data, any unusual patterns can be detected. There are several algorithms to calculate the FD such as box-counting [23], Katz’s [24], and Higuchi [25]. In this study, Higuchi FD is used as a fractal feature extraction method because it requires less computational power, guarantees accurate estimation, supports memory efficiency than other methods. To the best of our knowledge, Higuchi FD has not been used to extract features for determining exact attack types. The Higuchi method first re-generates original input data as a finite time series dataset based on pre-defined window size (k=5). For the given input data, new finite time-series is constructed as follows:

\[
x(m), x(m + k), x(m + 2k), ..., x(m + \left\lceil \frac{N - m}{k} \right\rceil \cdot k),
\]

where \(\lceil \cdot \rceil\) denotes the Gauss’ notation, the largest integer in the neighborhood of the number, and both \(k\) and \(m\) indicate internal and initial time, respectively. The length of the curve \(x^m_k\) is defined as

\[
L(m) = \frac{1}{K} \left\{ \sum_{i=1}^{N/m} |x(m + i \cdot k) - x(m + (i - 1) \cdot k)| \left\lceil \frac{N - m}{k} \right\rceil \cdot k \right\}
\]

where \(\left\lceil \frac{N - m}{k} \right\rceil \cdot k\) presents the normalization factor for the
curve length and \( N \) is the total length of the signal. \(< L(k) >\) defines the length of the curve for the time series \( k \) and \(< L_m(k) >\) denotes the average value over \( k \).

Statistical features including mean, median, and standard deviation are computed to find meaningful information to determine attack types. After collecting features, a machine learning algorithm, Support Vector Machine (SVM), is used to generate a predictive model. SVM [26], [27] is a supervised machine learning algorithm. It constructs an optimal hyperplane that separates a set of positive examples from a set of negative samples with maximum margin [28]. Due to its effectiveness, SVM is broadly used in pattern recognition, regression-based statistical learning theory, and structural risk minimization. In addition, it has an ability of handling large feature space. Therefore, we utilized SVM to design the predictive model. Neural Network (NN) is also used to determine the effectiveness of our proposed (SVM-based) predictive model. Since the neural network is one of the broadly used techniques for network anomaly detection [29], a NN-based predictive model is designed and its performance is compared with the SVM-based predictive model.

IV. EXPERIMENTAL RESULTS

As described above, the training dataset consists of 125,973 network traffic records. There are 22,544 network traffics in the testing dataset. Figure 2 represents that one of the raw features (‘duration’) used in this study. It explains how the network traffic data is complex in considering of ‘normal’ and ‘abnormal’ activities in the training dataset. The duration indicates length (seconds) of network connection. It represents how long each network session was being established between source and destination.

Non-parametric t-test (Kolmogorov-Smirnov (KS) and Wilcoxon rank-sum test [30]) are performed to examine statistical distribution between the two datasets (training and testing). The hypothesis of the test is that the distribution of the two datasets is the same. Figure 3 presents the Kolmogorov-Smirnov test and the Wilcoxon rank-sum test results of the feature (‘duration’). Based on the test results, we can conclude that our hypothesis can not be rejected, indicating that the distribution of the two datasets is similar.

Figure 4 represents how the raw feature (‘protocol type’) is distributed in the training and testing datasets. TCP and UDP are the most commonly used transport layer protocols in network communication. Since TCP provides a reliable connection, it is used by majority of Internet applications including WWW, FTP, and e-mail. However, UDP is a connectionless communication method that allows the fastest and most simple way of transmitting data to the receiver. Due to this reason, UDP is widely used for online video communication or VoIP applications. Since TCP and UDP is commonly used in the Internet applications, attackers often use TCP- or UDP-based attack techniques. Attackers also use ICMP which is internet layer protocol. ICMP is a message control (or error-reporting) protocol that is used for controlling the Internet between a host server and a gateway. Therefore, ICMP is not directly used or apparent to application users. However, attackers use ICMP to attack networks by sending bad ICMP packets or overloading the targeted network’s bandwidth (often called ICMP Smurf or DDoS Attack).

As mentioned previously, CART is considered for rule extraction. All generated rules with the training dataset are tested with the testing dataset. As a result, overall testing accuracy was 80.73%. Only the rules with high accuracy and
TABLE II. SOME OF THE GENERATED RULES MAINTAIN HIGH ACCURACY ARE PRESENTED.

<table>
<thead>
<tr>
<th>Rules</th>
<th>Outcome (Accuracy - records)</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ \text{If } (\text{PT} \neq \gamma &amp; \text{PT} \neq \mu &amp; \text{FLG} = \epsilon &amp; \text{FLG} = \theta &amp; \text{SRV} \neq \alpha &amp; \text{SRV} \neq \beta) ]</td>
<td>Abnormal (99.7% - 6548/6556)</td>
</tr>
<tr>
<td>[ \text{If } (\text{PT} = \gamma &amp; \text{FLG} \neq \delta &amp; \text{FLG} = \epsilon) ]</td>
<td>Normal (88.0% - 6669/7573)</td>
</tr>
<tr>
<td>[ \text{If } (\text{PT} \neq \gamma &amp; \text{PT} \neq \mu &amp; \text{FLG} = \rho &amp; \text{duration} &lt; 8.5) ]</td>
<td>Abnormal (72% - 89/124)</td>
</tr>
</tbody>
</table>

where SRV: service, FLG: flag, PT: protocol type, \( \alpha: \text{SMTP} \), \( \beta: x_{11} \), \( \gamma: \text{HTTP} \), \( \delta: S_{1} \), \( \epsilon: \text{RSTR} \), \( \theta: \text{OTH} \), \( \epsilon: \text{RSTO} \).

After determining the abnormality of network traffic, a predictive model is generated to identify specific attack types. As mentioned in Section III-A, the abnormal attacks (i.e. DoS, U2R, R2L, and Probes) are not equally distributed in the training and testing datasets. Although we know that U2R and R2L attacks are more harmful to networks [14], it is difficult to detect these attacks because a small fraction of such attacks exists in the datasets. Total thirty-seven input features are used to design the predictive model. After applying Higuchi FD and statistical measures to the input features, 148 features are extracted. To enhance the performance of detecting exact attack types by excluding less relevant information, statistical significance of the extracted features is measured with ANOVA. From this statistical significance testing, we can identify how well the features are statistically valuable for determining exact attack types. From the ANOVA test, we found that forty-nine features are statistically significant (\( p < 0.05 \)). These significant features are used to generate a predictive model with SVM.

Since the predictive model is designed with the training dataset, it is important to see how the proposed predictive model is efficient for detecting network anomalies from the testing dataset. To see the effectiveness of the model, we focused on understanding the difference between the training and testing datasets in consideration of the extracted features. Figure 5(a) and 5(b) show Higuchi fractal dimension feature of the network feature (srv\_error\_rate) and the statistical measurement feature (i.e. median) of the network feature (same\_srv\_rate), correspondingly. The figures indicate how the extracted features from the training and testing datasets are similar. More specifically, Figure 5(a) presents a Higuchi FD pattern of the variable “srv\_error\_rate” between the training and testing datasets. The variable “srv\_error\_rate” indicates connections that have REJ errors on services. Although there are differences among attacks, the extracted features maintain similar trends in the training and testing datasets. It addresses that detecting abnormal network traffics in the testing dataset can be performed precisely. Figure 5(b) represents mean and the standard error of the mean (SEM) of the feature (same\_srv\_rate). This feature denotes connections that have the same services on the network. This figure indicates that all attack types except Probes retain similar characteristics in the training and testing datasets. It explains us that our predictive model will maintain high accuracy when identifying abnormal behaviors in network traffic. In addition, we examine the input feature to see the difference between before and after applying Higuchi FD. For this, their mean are compared.
Figure 6 shows the result of the feature (srv_error_rate) for the DoS attack between the raw feature and its Higuchi feature in the training and testing datasets. It explains that since the Higuchi FD feature well maintains its characteristics compared to the raw feature, the predictive model designed with the extracted Higuchi FD features can successfully distinguish new incoming exact types of attacks.

![Graph showing feature comparison](image)

Fig. 6. An example result about the mean of the feature (srv_error_rate) for the DoS attack, in consideration of the training and testing datasets.

<table>
<thead>
<tr>
<th></th>
<th>SVM</th>
<th>NN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>77.10%</td>
<td>51.90%</td>
</tr>
<tr>
<td>ROC area</td>
<td>0.879</td>
<td>0.643</td>
</tr>
</tbody>
</table>

Our proposed SVM-based predictive model is compared with a NN-based predictive model. The comparison is performed with the testing dataset focusing on how effectively each model can detect exact attack types among DoS, U2R, R2L, and Probes. Table III shows the comparative result between SVM-based and NN-based predictive models. From the comparison, we found that SVM-based predictive model was outperformed in terms of accuracy. Although the overall accuracy of SVM-based predictive model was 77.1%, we found that true positive of DoS attack and Probes were 99% and 100%, respectively. When generating the SVM-based predictive model with using raw features, we found that the accuracy was dropped to 69% with the area of ROC as 0.744. This explains that our proposed predictive model is able to maintain a higher chance of identifying exact attack types.

V. DISCUSSION AND CONCLUSION

Understanding network traffic is important for securing our computing infrastructures. However, it is difficult to differentiate normal network traffic from abnormal network behaviors. This paper contributes to designing a two-level abnormal network traffic monitoring method. Since the most critical advantage of adapting rule-based method is that transparent measures are applied to extract significant features. Prior to applying them directly to design the predictive model, all extracted features were analyzed to determine their statistical significances. As explained in Section IV, we found that the performance accuracy of detecting network anomaly was high when using the extracted features.

For understanding the effectiveness of our proposed predictive model, it is important to perform a comparative study with other known techniques. In this study, we considered performing a comparison between our proposed SVN-based predictive model and a broadly used NN-based predictive model. From this comparison, we identified that our proposed model shows a better performance than the NN-based model. Although the overall accuracy of the proposed model was 77.1%, true positive of DoS and Probes attacks showed over 90% of accuracy. Since U2R and R2L attacks have fewer numbers of records in the datasets, it is difficult to precisely detect U2R or R2L attacks. Due to this reason, the overall performance accuracy of detecting abnormal behaviors in network traffic was lower than expected. This accuracy can easily be increased if we add more network traffic data related to U2R and R2L attacks. For future works, we plan to find more significant features to detect network attacks with understanding details about the attacks.

TABLE III. A PERFORMANCE COMPARISON BETWEEN SVM-BASED AND NN-BASED PREDICTIVE MODELS WITH THE TESTING DATASET.

ACKNOWLEDGMENT

This work was fully supported by US Army Research Office (ARO) grant W911NF13110143, and partially supported by ARO grant W911NF1210060.

REFERENCES


709
Resource-Efficient Multi-Source Authentication Utilizing Split-Join One-Way Key Chain

Seonho Choi¹, Kun Sun², Hyoonsang Eom³
¹Department of Computer Science, Bowie State University, Bowie, Maryland, U.S.A.
²Center for Secure Information Systems, George Mason University, Fairfax, Virginia, U.S.A.
³School of Computer Science and Engineering, Seoul National University, Seoul, Korea

Abstract—In wireless ad hoc networks, most of the authentication protocols assume a single source of trust. However, in the presence of multiple trust sources (called source group in this paper), it becomes difficult to design resource (or energy) efficient authentication protocols, especially for multicast/broadcast services, utilizing multiple trust sources at the same time. Some traditional authentication approaches may be extended and used for this purpose. However, the communication overhead, e.g., may increase significantly in proportion to the number of trust sources.

In this paper, we propose a new scheme named as Multi-source Authentication with Split-Join One-Way Key Chain (SOKC). In this new technique, the communication overhead is small and constant, and the memory requirement at the verifier node is also minimal. The source group node needs to store \(n\) keys where \(n\) represents the key chain length, which may be a reasonable requirement considering that the trust sources usually have more resources compared to other regular node(s) in the network (e.g., base stations in wireless sensor networks).

Keywords—authentication, security, protocol, wireless ad hoc networks

1 Introduction

In wireless ad hoc networks, most of the authentication protocols assume a single source of trust. For example, in a wireless sensor network (WSN), it is typically assumed there is one trustworthy base station and it is the only source of the trust. However, in the presence of multiple trust sources (called source group in this paper), it becomes difficult to design resource (or energy) efficient authentication protocols utilizing multiple trust sources at the same time. Some traditional authentication approaches may be extended and used for this purpose. However, the communication overhead, e.g., may increase significantly proportional to the number of trust sources. In this paper, we propose a new scheme named as Multi-source Authentication with Split-Join One-Way Key Chain (SOKC). In this new technique, the communication overhead is small and constant, and the memory requirement at the verifier node is also small. The source node needs to store \(n\) keys where \(n\) represents the key chain length, which may be a reasonable requirement considering that the trust sources usually have more resources compared to other regular node(s) in the network (e.g., as in sensor network).

Our technique utilizes a delayed key disclosure mechanism as in TESLA and uTESLA approaches [8,9], and also extended the one-way key chain technique to achieve the goals of minimal communication overhead and minimal storage requirements at the verifier nodes.

Our SOKC scheme may be applied to both unicast and multicast/broadcast authentication services. But, the application of our scheme would be simpler for unicast cases, and for most of the cases broadcast authentication services are more important since conveying the information from the trustworthy source to other nodes may be more critical compared to the communication between non-trustworthy nodes. For example, several routing protocols were proposed based on periodic broadcasting (e.g., flooding) of routing (or beacon) messages. These include TinyOS beaconing [6], directed diffusion and its multi-path variants [10], etc. Also, several location discovery schemes have been proposed which utilize broadcasting capabilities to estimate node locations [8]. Even though more advanced broadcast techniques may be utilized in the network, simple flooding may be preferred or required due to the simplicity or instability of network connections. Our proposed approach may be applied in both cases. Hence, we will focus on developing and applying the SOKC scheme for the multicast/broadcast services.

This SOKC scheme may be applied for various authentication problems. However, to show its applicability we chose two authentication problems. For example, in wireless ad hoc networks, some attacks exploit the fact that it is hard to authenticate the actual path (or number of hops) data packets traversed - especially the attacks against the broadcast services. Sinkhole and wormhole attacks belong to this attack category [6]. With the multi-source authentication capabilities, each node would be able to detect and cope with such attacks. A new path authentication technique may be developed by utilizing our Multi-source SOKC scheme. For example, the source group keys may be duplicated and randomly distributed across the network, so that the verifier nodes may be able check whether a packet has really passed through a certain number of source group nodes along the routing path from the claimed origination point.

The SOKC scheme may also be applied to WSNs with multiple base stations. It is typically assumed that a WSN has
only one base station. However, there may exist several drawbacks. Degraded reliability may be a problem due to a single point of failure. The latency may be an issue if the number of hops in the delivery paths may become large, which may cause the reduced lifetime of the sensor nodes and, thus, the entire sensor network. The deployment of multiple base stations were proposed to overcome these limitations [1,2,12]. However, in the presence of multiple base stations, it would be more difficult to provide robust authentication services since it would be required to tolerate compromise of multiple base stations as well as sensor nodes. If we assume that the base stations can communicate with each other directly using a separate channel, then our SOKC based approach may be used in providing multi-source broadcast authentication services. It is also assumed that all the base stations need to participate in authenticating the broadcast messages to provide increased security levels. If we consider the importance of the broadcast messages in WSNs, this would be a valid assumption.

2 Related Works

Several security mechanisms for authentication and secure routing protocols in wireless ad-hoc network are based on public key cryptography ([5], [14]). However, until now, the public key cryptography is still too expensive for the resource constrained mobile nodes. Secure routing protocols based on symmetric key cryptography have been proposed (e.g., [3], [4]). SEAD [3] is a distance vector routing protocol based on DSDV. The basic idea is to use one-way hash chains to authenticate the metrics and the sequence number of a routing table. The destination node can authenticate the source node; however, it cannot authenticate the intermediate nodes along the path from the source node to the destination node. Ariadne [4] uses per-hop hashing technique and source routing techniques to prevent route misbehaviors. However, it requires a precise time synchronization among all the nodes, which is usually difficult to be achieved in the mobile networks. Moreover, the communication overhead may increase significantly when including all the identifies and corresponding MACs for all the nodes along the path.

Authenticating broadcast (or multicast) traffic in wireless ad hoc networks is also a hard problem since the traditional approaches like digital signatures may not be adequate due to the heavy resource requirements. TESLA and μTESLA approaches [7,9] were proposed as viable solutions to the authentication problem in such networks. μTESLA utilizes the delayed key disclosure and one-way key chain techniques. First the packet is broadcast with a calculated keyed Message Authentication Codes attached along with the original data portion, and only after sufficient time is elapsed for all the nodes in the network to receive it, the corresponding key will be disclosed to the network nodes for authentication of the previously sent data and MAC. TESLA and μTESLA requires loose time synchronization among the network nodes.

Researchers have proposed several mechanisms to prevent the false data injection attacks. Przydatek, Song, and Perrig propose SIA [11], a secure information aggregation scheme for sensor networks that addresses the issue of false data injection using statistical techniques and interactive proofs, ensuring that the aggregated result reported by the aggregation node is a good approximation to the true value, even if a small number of sensor nodes and the aggregation node may have been compromised. SIA focuses on the accuracy of query results reported from the base station, whereas our scheme focuses on the authenticity of the reports from sensor nodes and provides a means to filter out any injected false data as early as possible. Both schemes can be combined to make the network more robust to false data injection attacks.

SEF [13] is a statistical en-route filtering mechanism to detect and drop false reports during the forwarding process. Authenticating event reports requires that nodes share certain security information, however, attackers can obtain such information by compromising a single node. To prevent any single compromised node from breaking down the entire system, SEF carefully limits the amount of security information assigned to each node, and relies on the collective decisions of multiple sensors for false report detection. First, SEF divides a global key pool into multiple partitions and carefully assigns a certain number of keys from one partition to individual node. Given that any single node knows only a limited amount of the system secret, compromising one or a small number of nodes cannot disable the overall network from detecting bogus reports. Second, by assuming that the same event can be detected by multiple sensors, in SEF each of the detecting sensors generates a keyed message authentication code (MAC) and multiple MAC are attached to the event report. As the report is forwarded, each node along the way verifies the correctness of the MAC’s probabilistically and drops those with invalid MACs. Finally, the sink verifies the correctness of each MAC and eliminates remaining false reports that elude en-route filtering. Comparing to statistical solution provided by SEF, our solution can provide a more resource-efficient path authentication, and it cannot handle the broadcast authentication.

Zhu et al [15] present an interleaved hop-by-hop authentication scheme for addressing the false data injection attack launched by the compromised nodes. The scheme guarantees that the base station will detect any injected false data packets when no more than a certain number t nodes are compromised. To defend against false data injection attacks, at least \( t + 1 \) sensor nodes have to agree upon a report before it is sent to the base station. \( t \) is a security threshold based on the security requirements of the application under consideration and the network node density. Further, it provides an upper bound for the number of hops that a false data packet could be forwarded before it is detected and dropped, given that there are up to \( t \) colluding compromised nodes. In other words, it also attempts to filter out false data packets injected into the network by compromised nodes before they reach the base station, thus saving the energy for relaying them. [15] is the
most similar work to our proposed scheme, but their approach cannot handle the broadcast authentication. Our solution approach utilizes a new SOKC technique along with the delayed key disclosure to achieve a much smaller communication overhead.

![Figure 1. µTESLA technique [7,9] is shown with a key disclosure delay of \(d=3\) block periods. Note that keys are disclosed later while the message and MAC portions are disclose in the corresponding block.](image)

3 Methodology

3.1 Assumptions

We refer to the minimum number of hops necessary for a packet to reach from any node located at one extreme edge of the network to another node located at the opposite extreme, as the diameter of the ad hoc network. Packets may be lost or corrupted in transmission on the network. A node receiving a corrupted packet can detect the error and discard the packet. Nodes within the ad hoc network may move at any time without notice, and may even move continuously, but we assume that the speed with which nodes move is moderate with respect to the packet transmission latency and wireless transmission range of the particular underlying network hardware in use. We assume that nodes may be able to enable promiscuous receive mode on their wireless network interface hardware, causing the hardware to deliver every received packet to the network driver software without filtering based on link-layer destination address. Even though this feature is not required, by utilizing this feature, the performance of our scheme may be enhanced especially when the mobility level is high in the network.

The local clocks of the nodes are assumed to be (at least) loosely synchronized with a maximum time synchronization error \(\Delta\). Various time synchronization techniques were proposed for wireless ad hoc networks, and any of them may be utilized to achieve this requirement. Similar assumptions were made in the broadcast authentication schemes such as TESLA, µTESLA, etc. [7,9].

Also, the time line is assumed to be divided into block periods as in TESLA and µTESLA approaches [7,9]. In each block period one packet may be sent out for broadcasting by any valid broadcast originator (which may be determined by the application). Delayed key disclosure mechanism is adapted and incorporated in our scheme. Each broadcast packet contains the message, the authentication-related information, and the key information that is disclosed for the previously sent out message. The key disclosure delay is denoted as \(d\) block periods. These can be seen in Figure 1.

We assume that there is one source and one or more recipients that are involved in each session (one or more data packets are delivered in each session). That is, our authentication approach may be used for both unicast and multicast/broadcast communications. However, we will focus on developing protocols for broadcast services as mentioned before. The packets are transmitted along a multi-hop delivery path to the receiver(s). The delivery path will be determined by a routing protocol used in the network. Many routing protocols were proposed for wireless ad hoc networks, and any of them with reasonable route change rates (due to mobility) may be utilized in the network.

Finally, it is assumed that the number of different source group keys in one source group (which is denoted as \(m\)) is an odd number.

3.2 Overview of the Protocol

The protocol carries out the following three processes to provide the multi-source authentication. In this scheme it is assumed that the number of source group keys, \(m\), is an odd number.

1. Offline SOKC generation (Figure 2): SOKC is generated offline by utilizing the source seed \((Z_0)\), source keys \((a_i)\), one-way hash operation, and the bitwise EXOR operation. Source nodes with a secret source key \(a_i\) will be equipped with a chain of keys that are obtained from the intermediate keys, \(Y_j\) \((1 \leq j \leq n-1)\), by applying the EXOR operation with \(a_i\). The keys generated in this process are denoted as \(SOKC^i = \{K^i_{n-1},...,K^i_2,K^i_1\}\).

The intermediate keys that are generated in this process are named as \(Y_i\) and \(Z_0\), which will be explain in more detail in a later section.

2. Semi-encrypted key pre-distribution (Figure 3): when the original sender node (this may be one of the source group nodes or may not be one of them) has some message to send, it will first send a packet that has the following field:

- random nonce \(R_i\) of \(k\) bits - this would be used to prevent the disclosure of the next key \((Y_j)\) in the SOKC that is needed for the next round of validation.
Figure 2. SOKC generation: the entire key chain is generated offline and only the keys in the solid rectangles are stored in the nodes. The keys, $K^i$, are stored in the $i$-th source node(s). $Y_n$ is bootstrapped in each verifier node. The intermediate keys and even the secret source keys, $a_i$, are not stored in any node in the network.

Once this field is filled with a random number generated by the original source and the packet is sent out, the first source group node from $a_i$ (this may be the same as the original sender node) will apply EXOR operation with this random number to (its next key in the SOKC $\oplus$ its Message), and forward the packet to the next node in the delivery path. The next source group nodes with different source keys will carry out the same process: get the value from this field and apply EXOR operation to it. But, this process is done only once for each source key $a_i$, $1 \leq i \leq m$. In other words, if there are multiple source group nodes with the same source key in the delivery path, only the first one will carry out this process. When the packet finished traversing all the source group nodes with $m$ different source keys, then the verifier nodes in the remaining path will have the following value in the packet field:

$$\Phi_j = R_j \oplus a_1 \oplus a_2 \oplus a_3 \oplus Y_j \oplus M_j$$

$M_j$ stands for the message in the $j$-th packet. This value will be stored in the verifier nodes for later authentication purposes. Verifiers may store the other field values such as the actual message ($M_j$) depending upon the scheme.

3. delayed key disclosure with verification (Figure 3): After the key disclosure delay ($d$ block periods), the original sender of $R_j$ will start the key disclosure process by including the following fields in the packet:

- disclose the actual random nonce used $d$ block period before ($R_i$) - the original sender will initialize this field to all 0s.
- key disclosure field for accommodating the SOKC keys from the $m$ source group nodes (they will be EXORed and this field requires only $k$ bits)

Each group source nodes will apply EXOR operation between the next key in SOKC and the value from the key disclosure field mentioned above, and store back the result into the key disclosure field again. After the packet traverses all of the $m$ source group nodes with different source keys, the packet will contain the following value in its key disclosure field:

$$a_i \oplus a_2 \oplus a_3 \oplus Y_j = Z_j$$

Once the packet reaches a verifier node, and if the packet is claimed to have traversed $m$ different source group nodes, then the verifier node will carry out a sequence of steps. First, it will extract the intermediate key, $Z_j$, from the key disclosure field, and checks whether this intermediate key is really from the authentic SOKC by applying the one-way hash operation and comparing the result to the already stored $Y_{j+1}$. If they don't match, then the key disclosure packet is discarded, and the already stored message $M_j$ will not be authenticated. If they match, then the verifier extracts the intermediate key, $Y_j$, by multiplying $a_i \oplus a_2 \oplus a_3$ to $Z_j$ and stores it as $Y_j$ as a newly disclosed authentic SOKC key to be used in the next round of authentication.

Then, the verifier will check the following condition to compare to the already stored $\Phi_j$:

$$R_j \oplus a_1 \oplus a_2 \oplus a_3 \oplus Y_j \oplus M_j = \Phi_j$$

If the equality holds, then the previously (in TESLA) stored $R_j$ and $M_j$ are validated.

3.3 Basic Scheme

Notations

The following are defined for our authentication process:

- Source group: a group of nodes equipped with SOKCs generated from $m$ different source keys, $a_i$ where $1 \leq i \leq m$ , are distributed among $N_{src}$ number of source nodes ($m \leq N_{src}$). It is assumed that $m$ is an odd number in this scheme. The source nodes may or may not be located in close proximity, and some source
nodes may have the same source key if \( m < N_{\text{src}} \). The source key size is denoted as \( k \) bits.

- Verifier group: \( N_{\text{rf}} \) nodes (e.g., multicast group members or all the nodes in the broadcast case) are equipped with a verification information for authenticating a packet’s traversing of at least \( m \) source nodes with different \( a_i \) in the routing path. That is, a verifier node has the ability to verify that the packet passed through all the source group nodes with \( m \) different source keys.

The following are the information kept in the source group node and the verifier group node:

- Source node from \( a_i \) keeps the following items:
  - Split-Join One-way Key Chain from \( a_i \):
    \[
    SOKC_i^i = \{ K_{n-1}^i, ..., K_2^i, K_1^i \}
    \]
  - public source key sum \( a = a_1 \oplus a_2 \oplus \cdots \oplus a_m \)
  - cryptographic one-way hash function
- Verifier node keeps the following items:
  - public source key sum \( a_i \oplus a_2 \oplus \cdots \oplus a_m \)
  - last key, \( Y_n \), in the SOKC
  - cryptographic one-way hash function

**SOKC generation with \( m \) source keys, \( a_i \ (1 \leq i < m) \) (Figure 2):**

This process is assumed to be carried out offline before the network launch time. The possible issues that may arise in developing online SOKC generation will be addressed in the proposed research section later. For the offline case, the detailed algorithm is shown in Figure 2 and the detailed steps are listed as follows:

1. Apply a cryptographic hash function to \( Z_0 \) to generate \( Y_1 \) which is also \( k \)-bit long.
   a. That is, \( Y_1 = H(Z_0) \).
2. Calculate a key \( K_1^i \) in the chain by applying the EXOR operation with a secret source key \( a_i \), \[ K_1^i = a_i \oplus Y_1 \].
3. Calculate \[ Z_1 = a \oplus Y_1 = (a_1 \oplus a_2 \oplus \cdots \oplus a_m) \oplus Y_1. \]
4. Apply a cryptographic hash function to \( Z_1 \) to generate \( Y_2 \). That is \( Y_2 = H(Z_1) \).
5. Calculate the second key \( K_2^i \) in the chain, \[ K_2^i = a_i \oplus Y_2 \].
6. Calculate \[ Z_2 = a \oplus Y_2 = (a_1 \oplus a_2 \oplus \cdots \oplus a_m) \oplus Y_2 \].
7. Repeat steps 4 through 6 until key \( K_{n-1}^i \) is obtained.

The one-way key chain at the source is now obtained as \( SOKC_i = \{ K_{n-1}^i, ..., K_2^i, K_1^i \} \). The keys in this \( SOKC_i \) are bootstrapped in the source group nodes. These keys are used in reverse order starting from \( K_{n-1}^i \). The last key \( Y_n \) in the entire SOKC is assumed to be bootstrapped at each verifier node.

**Packet Format**

The \( j \)-th packet, \( 1 \leq j < n \), has the following packet format consisting of 5 fields:

1. SRC index bits
2. Semi-Encrypted Key (\( \Phi_j \)) pre-distribution– \( k \) bits
3. Nonce (\( R_{j,d} \)) disclosure– \( k \) bits
4. SOKC Key (for \( a \oplus Y_{j-d} \)) disclosure– \( k \) bits
5. Message (\( M_j \))

SRC index bits are used for showing which source group nodes the packet has been traversed. For example, it its \( i \)-th bit is set to 1, then it means that the packet is claiming that it has already traversed a node with the source key \( a_i \), and, if its \( i \)-th bit is 0, then it means that the packet has not traversed any such node. If another source node with the same \( a_i \) receives the packet whose \( i \)-th SRC index bit is set to 1, then it will forward the packet without modifying any of the fields, even though it can repeat the process without any adverse effect – but, it will waste resource if it does.

The other three fields following the SRC index bits field will be used in the semi-encrypted key pre-distribution and the delayed key disclosure with verification processes. Note that the nonce disclosure and SOKC key disclosure fields will disclose the values that were previously used in the \( (j-d) \)-th block period.

After the key disclosure delay (i.e., \( d \) block periods), another packet needs to be sent by the original sender and it would contain the following fields:

1. SRC index bits
2. Semi-Encrypted Key (\( \Phi_{j,d} \)) pre-distribution– \( k \) bits
3. Nonce (\( R_j \)) disclosure– \( k \) bits
4. SOKC Key (for \( a \oplus Y_j \)) disclosure– \( k \) bits
5. Message (\( M_j \))

**Semi-encrypted key pre-distribution at the source node with a source key \( a_i \) (\( 1 \leq i < m \)):**

Again, the purpose of this process is to let source group nodes to reveal their SOKC keys in a semi-encrypted form by applying the EXOR operation to the random nonce (\( R_i \)) sent out by the original sender of the packet. Let’s assume that the a \( j \)-th packet is sent out by the original sender. The process is shown in Figure 3 and the detailed steps are described as follows:
1. Original sender generates a random nonce ($R_j$) and insert it into the semi-encrypted key pre-distribution field in the packet before sending it.

2. Each source group node from a source key, $a_i$, in the delivery path will carry out the following:
   a. if the SRC index bit (whose index value is $i$) is equal to 1, then go to step 3.
   b. extract the value in the semi-encrypted key pre-distribution field (let it be denoted as $x$).
   c. extract $M_j$ from the packet.
   d. calculate $M_j \oplus K_i' \oplus x$ and store this as a new value in the semi-encrypted key pre-distribution field of the packet.
   e. set the SRC index bit (whose index value is $i$) as 1.

3. Each verifier node will perform the following steps:
   a. if all of the $m$ bits are set to 1 in the SRC index bits field, then the node will extract the value of the semi-encrypted key pre-distribution field, and store it as $\Phi_j$ to be used at the future verification time (after $d$ block periods).
   b. check whether $H(z) = Y_{j+1}$
      a) If not, then the key validation for SOKC fails, discards the packet and exit from the algorithm.
      b) If the equality holds, then store $a_1 \oplus \cdots \oplus a_m \oplus z$ as a valid $Y_j$ for future SOKC validation.
         A. calculate $R_j \oplus z \oplus M_j = R_j \oplus a_i \oplus a_j \oplus a_3 \oplus Y_j \oplus M_j$ and compare this to $\Phi_j$ that were extracted and stored in the previously received $j$-th packet.
         i. If they are the same, then the multi-source authentication succeeds for $M_j$.
         ii. If they don’t match, then the multi-source authentication fails for $M_j$.
   4. Forward the packet if it is needed.

Delayed key disclosure and verification after the key disclosure delay (Figure 3)

Because the actual SOKC keys are disclosed after the delay ($d$ block periods), the verifications of the keys and messages that were included in the $j$-th packet may be carried out when the nodes receive/process a packet in the $(j+d)$-th block period. So, we disclose the $j$-th keys from the SOKCs in the $(j+d)$-th packet, and the verifications will be carried out by the verifier nodes upon the receipt of the $(j+d)$-th packet. The process is depicted in Figure 3, and the detailed steps are described as follows:

1. Original sender discloses the nonce ($R_j$) by including in the $(j+d)$-th packet’s nonce disclosure field.
2. Each source group node from a source key, $a_i$, in the delivery path will carry out the following:
   a. if the SRC index bit (whose index value is $i$) is equal to 1, then go to step 3.
   b. extract the value from the SOKC key disclosure field (let it be denoted as $y$).
   c. calculate $K_j' \oplus y$ and store this as a new value into the SOKC key disclosure field of the packet.
   d. set the SRC index bit (whose index value is $i$) as 1.
3. Each verifier node will perform the following steps:
   a. if all of the $m$ bits are set to 1 in the SRC index bits field, then the node will perform the following steps:
      i. extract the value of the SOKC key disclosure field (let it be denoted as $z$).
   4. Forward the packet if it is needed.

Resource Requirements (Figure 3)
The resource requirements at a source group node from $a_i$ are:
- $n \times k$ bits are needed for storing the SOKC keys.
- $k$ bits for storing $\Phi_j$
- $d \times k$ bits: for storing $\Phi_j$ in at most $d$ consecutive block periods
- $2k$ bits: for temporarily storing $Z_j$ and $a_1 \oplus a_2 \oplus a_3 \oplus Y_j = Z_j$

Hence, the total memory requirement at a verifier node is $(d+3) \times k$ bits.
The communication overhead at each packet (purely needed for our scheme) consists of the following:
- $m$ bits: for SRC index bits
- $3k$ bits: for semi-encrypted key pre-distribution field, nonce disclosure field, SOKC key disclosure field

Hence, the total overhead is $m+3k$ bits for each packet.

4 Conclusion

We presented a new resource-efficient multi-source authentication scheme with Split-Join One-Way Key Chain (SOKC). In this new technique, the communication overhead is small and constant, and the memory requirement at the verifier node is also minimal. This technique may be effectively used for wireless ad hoc networks when there exist multiple trust sources to be utilized.
5 Acknowledgement

This work was supported in part by US Army Research Office (ARO) grant W911NF-12-1-0060, and in part by Seoul R&BD Program (No. JP110034).

6 References


efficient distance vector routing for mobile wireless ad
hoc networks,” in Proceedings of the 4th IEEE Workshop

on-demand routing protocol for ad hoc networks. Wireless

security in mobile ad hoc networks,” in Proceedings of the
ACM Symposium on Mobile Ad Hoc Networking and

[6] Chris Karlof and David Wagner, Secure Routing in
Wireless Sensor Networks: Attacks and Countermeasures,
In Proceedings of the First IEEE International Workshop
on Sensor Network Protocols and Applications, May
2003.

[7] Donggang Liu, Peng Ning, Multi-Level u-TESLA: A
Broadcast Authentication System for Distributed Sensor
Networks, Submitted for journal publication. Also
available as Technical Report, TR-2003-08, North
Carolina State University, Department of Computer

based Location Discovery in Ad Hoc Sensor Networks. In Proceeding (424) Wireless Networks and Emerging
Technologies - 2004

and secure source authentication for multicast. In
Proceedings of Network and Distributed System Security

Havinga A directed diffusion-based secure multicast
scheme for wireless sensor networks.First International
Workshop on Wireless Security and Privacy (WiSPr'03)

Information Aggregation in Sensor Networks. In Proc. of

for Wireless Sensor Networks with Multiple Base
Stations," Proceedings of the IEEE International
Conference on Communications, 2008, ICC’08, pp. 1625-
1629, Beijing, China, May 2008.

Filtering of Injected False Data in Sensor Networks. IEEE JOURNAL ON SELECTED AREAS IN
COMMUNICATIONS, VOL. 23, NO. 4, APRIL 2005

Network Magazine, vol. 13, no. 6, November/December
1999.

Interleaved Hop-by-Hop Authentication Scheme for
Filtering of Injected False Data in Sensor Network
FSM Watermarks Based on Ordering of Flip Flops

Edward Jung
Southern Polytechnic State Univ.
1100 S. Marietta Pkwy
Marietta, GA 30067
+1-678-915-5546
ejung@spusu.edu

Chie-Cheng Hung
Southern Polytechnic State Univ.
1100 S. Marietta Pkwy
Marietta, GA 30067
+1-678-915-3574
chung@spusu.edu

Seonho Choi
Bowie State University
14000 Jericho Park Road
Bowie, MD 20715
+1-301-860-3967
schoi@cs.bowiestate.edu

ABSTRACT
In this paper, we propose a method for constructing and verifying a watermark of finite state machines based on ordering of flip flops. The underlying idea is to use the flip-flop arrangement information as part of the IP owner’s secret. We present simple watermark construction and verification algorithms. We demonstrate the feasibility of the proposed method for a class of FSMs which satisfies certain conditions.

Categories and Subject Descriptors
B.5.1 [Register-Transfer-Level Implementation]: Design---
Styles; K.5.1 [Legal Aspects of Computing]: Hardware and Software Protection---Proprietary rights

General Terms
Security, Design, Algorithms

Keywords
FSM watermarking, flip-flop arrangement, cyclic property

1. INTRODUCTION
Watermarking is a technique that can be used to securely identify the ownership of the origin of design intellectual properties (IPs). A variety of techniques have been proposed for watermarking different steps of the design process [1][7][9][10][14][15][16][17][19][20]. There are two main classes of approaches. One approach is hardware metering [8], which allows design houses to have post-fabrication control on the produced ICs, and monitor their usage. Another popular approach to IP protection is hardware watermarking [12], in which certain identity information is inserted into behavioral specification or sequential structure of the design. Finite state machines (FSMs) are the backbone of a sequential system design. In this paper, we focus on FSM hardware watermarking.

The central idea of proposed method is based on a decomposition of FSMs. Consequently, this scheme is related to the classical problem of state assignment which had been studied extensively during the 1970s – 1980s. These studies had been conducted for designing and synthesizing sequential circuits with focused goals of reducing circuit delay, areas, power consumptions, and/or of improving testability. However, in this paper, we investigate this problem to see if it can be applicable to protect the design IPs.

This paper introduces a new method of FSM watermarking. Specifically, we focus on proposing a watermarking method which utilizes the flip-flop ordering information as part of IP owner’s secret. We present simple watermarking algorithms for constructing and for verifying the watermark. The main contributions are: (1) to the best of our knowledge, it is the first attempt to utilize the flip-flop ordering information as part of IP owner’s secret without adding new states or state transitions in order to construct and to verify the watermark, (2) we analyze the proposed scheme for estimating the chance of guessing the watermark without knowing the designer’s secret.

Related work and preliminary background are presented in Section 2 and Section 3, respectively. The watermarking algorithms are described in Section 4. Analysis is performed in Section 5. We show the feasibility in Section 6. We conclude in Section 7.

2. RELATED WORK
Most popular traditional approaches include: (a) FSM watermarking based on Unused Transitions: the authors in [18] introduced the first IP protection using FSM watermarking. The algorithm is based on extracting the unused transitions in a state transition graph (STG) of the behavioral model. In their solution, extra transitions are added to satisfy the design goals. (b) FSM watermarking by Property Implanting: the author in [13] tried to manipulate the STG of the finite state machine to implant the watermark as a property. The property was topological in nature and was defined in terms of visited states \((s_1 \rightarrow s_2 \rightarrow \cdots \rightarrow s_l)\). In order to define the topological property, the author added extra states and state transitions in a systematic way to satisfy a specific topological requirement. (c) FSM watermarking by Integration of Two Distinct FSMs: the authors in [6] designed a completely new FSM as a watermark and then the watermark FSM was combined with the original FSM to create an integrated composite FSM. Constructing a new watermark FSM was done by adding new states and transitions.

More recently, a FSM watermarking scheme by making the authorship information a non-redundant property of the FSM was proposed in [3]. In this work, the watermark bits were added into the outputs of the existing and free transitions of STG. Another method was proposed in [11]. In this work, a set of edges were added as a dummy entity. This was done by assigning state...
encoding values. The new edges created by this method were paired with an unused state input combination, and the output was specified as a don’t-care condition.

Despite these popular methods which can be effective in protecting IPs of FSMs as demonstrated in these works, these approaches are fundamentally based on expanding the original FSM to an enlarged FSM with new states and/or state transitions.

In this paper, we investigate a new method which does not depend on adding new states and/or state transitions.

3. PRELIMINARY

In this section we provide definitions and assumptions, followed by an illustrative example.

3.1 Definitions and Assumptions

Basic definitions which will be needed at a minimum level in this paper are presented below [4].

Definition 1: FSM = (I, O, S, δ, λ) where I, O, and S are finite, nonempty sets of inputs, outputs, and states, respectively. δ: I × S → S is the state transition function. λ: I × S → O is the output function.

Definition 2: A closed partition π on S of a FSM = (I, O, S, δ, λ) is defined as: if, for every two states which are in the same block of π and any input in I, the next states are in a common block of π.

Definition 3: A partition τ₁ on S of a FSM = (I, O, S, δ, λ) is input-independent, if for every state and all inputs, the next states are in the same block of τ₁.

Definition 4: A partition τ₂ on S of a FSM = (I, O, S, δ, λ) is the smallest input-independent, if τ₂ contains the maximum number of blocks in it.

Note that the states in S are encoded and then realized using a register (i.e., a set of flip flops). The binary values stored in a register can be observed by the user to check the current states of system.

Assumption 1: The internal values of flip flops can be checked by the user, if needed.

Checking the internal values of flip flops can be done using either a partial scan or a full scan.

3.2 Illustrative Example

Table 1 shows an example of a sequential design that is represented in a state table [5]. In D₁, there are six states. When a stimulating external input value is applied on the present state, it is deterministically moving to the next state while generating an external output.

One possible complete flip-flop arrangement is: {(A, 000), (B, 001), (C, 010), (D, 011), (E, 100), (F, 101)}. The corresponding logical equations using this particular assignment are: Y₁ = y₂, Y₂ = y₁y₃, Y₃ = xy₂ + xy₂ + x'y₂y₄ + y₂y₃, and Z = xy₄, where y₁ and Y₁ represent the present state and the next state, respectively.

Note that D₁ has the following interesting properties. It contains a component that is both (1) independent of the external input, and (2) three states forming a cycle. Consider three combined states \{α; β; γ\} = {A+B; C+D; E+F}, where “+” is used to denote an operator to combine two states. Then, the transition function of this component is defined as δ(α, −) = β, δ(β, −) = γ, and δ(γ, −) = α, where “−” denotes a don’t-care condition.

<table>
<thead>
<tr>
<th>Present State</th>
<th>Next State Q(t+1), Output z</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q(t)</td>
<td>Input x = 0</td>
</tr>
<tr>
<td>A</td>
<td>D, 0</td>
</tr>
<tr>
<td>B</td>
<td>C, 0</td>
</tr>
<tr>
<td>C</td>
<td>E, 0</td>
</tr>
<tr>
<td>D</td>
<td>F, 0</td>
</tr>
<tr>
<td>E</td>
<td>B, 0</td>
</tr>
<tr>
<td>F</td>
<td>A, 0</td>
</tr>
</tbody>
</table>

Putting it all together, the actual flip-flop arrangement can be made as follows: {(A, 000), (B, 001), (C, 010), (D, 011), (E, 100), (F, 101)} with {(α, 0), (β, 01), (γ, 10)} and {(a, 0), (b, 1)}. Note that the original states are realized by two internal states: A = (α, a), B = (α, b), C = (β, a), D = (β, b), E = (γ, a), F = (γ, b). For instance, the state “A” is realized by two internal states “α” and “a” using three flip flops.

Figure 1 shows an example of the three flip flops that can store the binary values in three flip flops. For instance, the state C can be realized with the binary values of flip flop = “010” as shown.

![Figure 1. Ordering of flip-flop arrangement.](image)

During the verification, the IP owner can verify his/her ownership by checking the three states (α, β, γ) in sequence, irrespective of input signals. During the (p + 1) = 4 time units, a cycle of states should be verifiable, as shown in Figure 2. The verification is done by checking the internal flip flop values.

![Figure 2. Verification of a cycle of states (with periodicity = 3)](image)

4. WATERMARKING ALGORITHM

In this section, we present the algorithms for creating and verifying a watermarked FSM.

4.1 Watermark Creation (δ-WFSM)

The watermarked FSM is created using a specific property. The specific property chosen as an example is maximal input-independent periodicity representing a cyclic behavior of a sequential FSM. This specific property is denoted by P*. The watermarked FSM, denoted by δ-WFSM, is defined with the four parameters.
Finding $p_{\text{max}}$ is important since it will increase the security level of the hidden watermark. Note that $\delta$ - WFSM can usually contain a unique characteristic of a cyclic behavior. Step d above is to extract such a cyclic behavior.

**Example 4-1**: Suppose $p_{\text{max}} = 3$ with three states $\{1, 2, 3\}$. Then, the state transition in $\delta$ - WFSM can be represented as an ordered set of states: $\{(1, 2, 3), (1, 2, 3), (2, 1, 3), (2, 1, 3), (3, 1, 2), (3, 2, 1)\}$.

The algorithm of finding $p_{\text{max}}$ can be developed as follow. The basic idea is given in [5].

**Procedure Max Periodicity ( )**:  
**Input**: a "n"-state FSM (FSM = $\{I, O, S, \delta, \lambda\}$, $|S| = n$)  
**Output**: $p_{\text{max}}$  
1) Find a set of a closed partition $(\pi_1, \pi_2, \ldots, \pi_r)$ and a nontrivial input-independent partition $\pi$ on $S$, where $\pi_i \geq \pi$, for $i = 1, 2, \ldots, r$.  
2) Choose the smallest closed partition $\pi_{\text{min}}$.  
3) The number of blocks in $\pi_{\text{min}}$ is the maximal periodicity $p_{\text{max}}$.

The complexity of finding the maximal periodicity is $O(n^2)$ since a pairwise comparisons of each state is needed in Step 1. The overall complexity of creating the watermarked FSM is $O(n^5)$.

Given a set of states $\{1, 2, 3, \ldots, p_{\text{max}}\}$, the assignment of flip-flop values in a specific order can be used to further reduce attack possibility. Suppose these states are implemented by a register $R$ with $m$ flip flops $R = [R_1, R_2, \ldots, R_m]$, where $R_i$ denotes the $i$-th flip flop and $m = \log_2 n$. Note that we need only $r = \log_2 p_{\text{max}}$ number of flip flops and $m \geq r$. Selection of an ordered subset of $r$ flip flops out of $m$ flip flops will suffice. Using the minimum number of flip flops is to ensure that there will be no additional states to be added.

### 4.2 Watermark Verification

The verification can be done using both the ordered set of states being visited, according to the maximal periodicity (Step 1 in Section 4.1), and the exact location of flip flops.

**Procedure Verification ( )**:  
**Input**: $R = [R_1, R_2, \ldots, R_m] = [FF_1, FF_2, \ldots, FF_m]$  
**Output**:  
1) Apply a random input to FSM $I^*$ input-independence $*/$  
2) Given $R = [R_1, R_2, \ldots, R_m] = [FF_1, FF_2, \ldots, FF_m]$  
   a. select the ordered subset of flip flops $< FF_{a1}, FF_{a2}, \ldots, FF_{ab} >$  
   b. check the expected ordered set of the flip-flop values  
3) If successful, the ownership is considered to be verified.

By performing this verification, the IP owner has verified the following: (1) the correct period of $\delta$-WFSM (e.g., period = 3), (2a) the exact order of cyclic states, and (2b) the specific placement of flip flops and its value. Note that these information are only known to the IP owner.

## 5. ANALYSIS

Analyzing any watermarking schemes can be broad since many different types of attacks are possible. Also, there are many well-known requirements for any watermarking solutions [1]. In this section, we focus on the most basic analysis for the proposed scheme.

### 5.1 Existence of the Property

Based on [5], we provide the results without a formal proof.

**Theorem 1**: The existence of a closed partition $\pi$ and a nontrivial input-independent partition $\pi_i$ on $S$ in the original FSM $= (I, O, S, \delta, \lambda)$, where $\pi \geq T_i$, is a necessary and sufficient condition for the existence of the watermarked FSM $\delta$ - WFSM $= (I, O, S, \delta, \lambda_o)$.

**Corollary 1**: If the number of blocks in an input independent partition $\pi_i$ is equal or greater than 2, is guaranteed to have a nontrivial periodicity of 2 or higher.

**Corollary 2**: A periodicity is maximal if the number of blocks (or elements) in $\pi_i$ is the largest.

### 5.2 Guessing the Watermark

In guessing the watermarked hidden information, analysis is performed in two different cases: (C1) the known periodicity, (C2) both the known periodicity and the known assignment of flip-flops. Table 2 summarizes the parameters used in the analysis.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
<th>Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>$</td>
<td>S</td>
</tr>
<tr>
<td>$m$</td>
<td>$</td>
<td>FF</td>
</tr>
<tr>
<td>$C; c$</td>
<td>$C$, an ordered set of states; $c =</td>
<td>C</td>
</tr>
<tr>
<td>$p^*$</td>
<td>The maximal periodicity of $\delta$ - WFSM $= \max_{\pi \in \pi_{\text{max}}}$, $</td>
<td>\pi</td>
</tr>
<tr>
<td>$m^*$</td>
<td>$</td>
<td>FF</td>
</tr>
<tr>
<td>$\pi(c)$</td>
<td>The number of the ways arranging $c =</td>
<td>C</td>
</tr>
<tr>
<td>$\Gamma(n, c)$</td>
<td>The number of the ways arranging $c$ states out of $n$ states</td>
<td>$1 \leq \pi(c) \leq n!$</td>
</tr>
<tr>
<td>$G(n, c)$</td>
<td>The number of the ways arranging both cyclic states and flip flops in order</td>
<td>(2A) and (2B)</td>
</tr>
</tbody>
</table>

Formula (1), (2A) and (2B), respectively, are derived as below.

$$\Gamma(n, c) = (\lfloor \log_2 c \rfloor)! \times \frac{\log_2 n}{\log_2 c}$$  \hspace{1cm} (1)

Note that $\Gamma(n, c)$ is determined by encoding $p^*$ states using $m^*$ ($= \lfloor FF \rfloor$) flip flops out of $m = |FF|$ flip flops as an ordered set.
\( G(n, c) \) is determined in terms of \( \pi(c) \) and \( \Gamma(n, c) \). Thus, 
\[
G(n, c) = \pi(c) \times \Gamma(n, c) 
\]
(2A)

The implication of the formula (2A) is that if both the cyclic ordering of states and the ordering of flip-flop arrangement are unknown, the adversary would try many possible ways to guess the watermark (in the worst case), provided that the adversary knows the periodicity.

**General Case:** In general, however, the maximal periodicity can be kept in secret (by the owner). In this case, the security level increases by a factor of \( "n" \) as shown in (2B):
\[
G(n, c) = n \times \pi(c) \times \left(\frac{\log_2 c}{\log_2 n}\right) 
\]
(2B)

**Lower and Upper Bound:** The lower and upper bound of \( G(n, c) \) occur when \( \pi(c) = 1 \) and \( n(c) = n! \), respectively.
\[
G(n, 1) = n 
\]
(3A)
\[
G(n, n!) = n \times n! \times \left(\frac{\log_2 n}{\log_2 c}\right) 
\]
(3B)

Other analysis such as coincidence (i.e., false positive collision) can be done, but we focus on analyzing the degree of difficulty in guessing the watermark without knowing the designer’s secret in this paper.

### 5.3 Limitation

Some FSMs may not satisfy *Theorem 1*. In this case, we should consider a weaker condition (e.g., relaxing maximal periodicity) at the cost of lower security (e.g., a higher probability of guessing the watermark).

### 6. FEASIBILITY

In this section we investigate the feasibility of the proposed method. Note that the main goal is to evaluate the degree of difficulty in guessing the watermark in the best scenario (i.e., the attackers should try many attempts to break the secret watermark information, provided that the given FSM satisfies *Theorem 1*.) In practice, however, some FSMs may not satisfy the conditions.

Table 3 shows the lower and upper bound of \( G(n, c) \) derived in (2B) in the previous section. The number of states \( n \) is from the FSM benchmarks [2]. Table 4 shows the value of \( G(n, c) \) for the more common cases. We considered the relatively low value of \( c \) as a function of \( n \). That is, approximately, \( c = \left[\frac{n}{k}\right] \) where \( k = 2, \ldots, \left[\frac{n}{2}\right] \). Note that we take a more conservative assumption (i.e., not the best scenario) in a sense that (1) the greater the value of periodicity (or the value of \( c \)), the more difficult the prediction is, and (2) we assumed the maximal periodicity does not exist beyond \( c = \left[\frac{n}{k}\right] \). Also, the subset of the benchmark circuits are selected since the system with small number of states are likely being used in a sequential design of systems (e.g., controller of embedded system).

**Table 3. Lower and upper bound of \( G(n, c) \)**

<table>
<thead>
<tr>
<th>Circuits</th>
<th>FFs</th>
<th>States (n)</th>
<th>Lower Bound ( c = 1, \pi(c) = 1 )</th>
<th>Upper Bound ( c = N, \pi(c) = n! )</th>
</tr>
</thead>
<tbody>
<tr>
<td>s27</td>
<td>3</td>
<td>8</td>
<td>8</td>
<td>1935360</td>
</tr>
<tr>
<td>s820</td>
<td>5</td>
<td>32</td>
<td>32</td>
<td>1.01042E+39</td>
</tr>
<tr>
<td>s1488</td>
<td>6</td>
<td>64</td>
<td>64</td>
<td>5.8469498E+93</td>
</tr>
<tr>
<td>s1492</td>
<td>8</td>
<td>256</td>
<td>256</td>
<td>8.854326E+51</td>
</tr>
<tr>
<td>s386</td>
<td>9</td>
<td>512</td>
<td>512</td>
<td>6.460615E+1174</td>
</tr>
<tr>
<td>s510</td>
<td>18</td>
<td>262144</td>
<td>262144</td>
<td>4.45277E+1306615</td>
</tr>
<tr>
<td>s991</td>
<td>19</td>
<td>524288</td>
<td>524288</td>
<td>7.115547E+2771033</td>
</tr>
<tr>
<td>s382</td>
<td>21</td>
<td>2097152</td>
<td>2097152</td>
<td>1.576848E+12346668</td>
</tr>
<tr>
<td>s400</td>
<td>597</td>
<td>5.1E+179</td>
<td>5.1E+179</td>
<td>Non-computable</td>
</tr>
<tr>
<td>s444</td>
<td>1728</td>
<td>21785</td>
<td>21785</td>
<td>Non-computable</td>
</tr>
</tbody>
</table>

**Table 4. \( G(n, c) \) for the various values of \( c \)**

<table>
<thead>
<tr>
<th>Circuits</th>
<th>States (n)</th>
<th>Periodicity (c)</th>
<th>( G(n, c) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>s27</td>
<td>8</td>
<td>2</td>
<td>1152</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>320</td>
</tr>
<tr>
<td>s820</td>
<td>32</td>
<td>4</td>
<td>15360</td>
</tr>
<tr>
<td>s832</td>
<td>8</td>
<td>7</td>
<td>7,74144E+7</td>
</tr>
<tr>
<td>s1488</td>
<td>64</td>
<td>2</td>
<td>768</td>
</tr>
<tr>
<td>s1492</td>
<td>8</td>
<td>4</td>
<td>46080</td>
</tr>
<tr>
<td>s386</td>
<td>16</td>
<td>3</td>
<td>3.09657E+8</td>
</tr>
<tr>
<td>s510</td>
<td>32</td>
<td>12</td>
<td>1.21250889E+40</td>
</tr>
<tr>
<td>s208</td>
<td>256</td>
<td>2</td>
<td>4096</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>4</td>
<td>344064</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>3</td>
<td>3.46816512E+9</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>8</td>
<td>8.99847347E+18</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>16</td>
<td>4.52669234E+41</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>64</td>
<td>6.5485838E+95</td>
</tr>
<tr>
<td></td>
<td>128</td>
<td>128</td>
<td>3.980343769E+222</td>
</tr>
</tbody>
</table>

Despite the limited use of the benchmark FSMs, we can make several observations in Table 3. First, for the most of the sequential circuits even with the small number of states (e.g., s27), the upper bound \( G(n, c) \) is very high, which indicates that the brute-force type guessing work can be realistically infeasible. Second, however, there are some cases that the lower bound \( G(n, c) \) is quite low. For instance, the circuits “s27” through “s27-n3” have the values of lower bound, 8 through 512. Third, as shown in Table 5, the more common cases show that \( G(n, c) \) is reasonably high for most of the FSMs.

### 7. CONCLUSION

We presented a FSM watermarking scheme which can be created by the IP owner utilizing the arrangement of flip flops. The underlying idea was to use the ordering of flip flops as part of designer’s secret. Despite the proposed method is not universal, it was illustrated that the FSM watermarking can be done using a simple flip-flop arrangement (similar to state assignments) for a class of FSMs.
8. ACKNOWLEDGMENTS

Our thanks go to the reviewers for the constructive suggestions.

9. REFERENCES


262
An Locking and Unlocking Primitive Function of FSM-modeled Sequential Systems Based on Extracting Logical Property

Edward Jung*, Chih-Cheng Hung**, Ming Yang* and Seonho Choi***

*School of Computing and Software Engineering, Center for Biometrics Research, Southern Polytechnic State University, GA, USA

** Anyang Normal University, Anyang, China
and School of Computing and Software Engineering, Center for Biometrics Research, Southern Polytechnic State University, GA, USA

*** Department of Computer Science, Bowie State University, MD, USA

Abstract

In this paper, we attempt to address the following question in the field of computer security: “can we build a primitive security function or a building block without an explicit modification of the original system for a class of systems modeled in a deterministic FSM?” As one of possible solutions, we propose a logical (or functional) property extraction-based solution (in general) and a synchronicity-based sequential locking and unlocking method (in particular). The proposed method is solely based on extracting an inherent property embedded in the original D-FSM. The property being considered is synchronicity due to the desirable characteristic that can be used for the design of an efficient sequential locking and unlocking method. The proposed method and its application to digital watermarks (as one of potential applications) are presented and analyzed. Both the feasibility and the limitation are discussed. However, the basic idea is general in a sense that (1) it can be applied to other security applications, and (2) other logical properties can be further explored.

Key Words: Property-based Security, Functional property, Synchronicity, Locking/Unlocking, Digital Watermarks, D-FSM

1. Introduction

One of core functions in designing secure systems is the mechanism of locking and unlocking (e.g., a password-based mechanism). In designing such a system (e.g., embedded systems or controllers), the system is frequently modeled using a deterministic finite-state machine (D-FSM) [8, 17]. Traditionally, the approach to the D-FSM based security solutions has focused on modifying the original D-FSM. Despite the higher-level of security that can be potentially achieved by this modification-based solution, it suffers from the increased complexity of modifying the original D-FSM and the associated overhead (e.g., performance, area). In many security applications, building an efficient locking and unlocking is important.

To address this problem, we proposed a solution ([7]) based on extracting the property (i.e., synchronicity) embedded in the original D-FSM and using that property to build a security building block (i.e., locking/unlocking). This does not require modifying the system (i.e., D-FSM) and the benefit of this approach is efficiency.
In this paper, we extend the earlier work [7] to the problem of digital watermarks as one of potential applications. Fundamentally, our work (both in this paper and [7]) is based on our belief (or principle) that any security solutions must minimize (or avoid if possible) the changes or modifications in the system solely to improve security. In doing so, we are promoting a *logical property extraction*-based security solution as much as possible (i.e., if the system possess a desirable property). In this approach, a certain type of property will be defined first, and then be extracted from the system, and finally be applied to build a certain type of security primitive functions (e.g., a locking/unlocking method) or the broader security solution itself. The contributions of this work can be summarized as follow:

- We propose fundamentally a different approach to dealing with the overheads of security solutions in a certain class of system (i.e., FSM-based sequential systems).
- We develop a locking and unlocking method as one of the security building blocks. The method is based on simply extracting the system’s inherent logical property (i.e., synchronicity) embedded in D-FSM.
- Simple algorithms are devised for both the automatic generation of key input sequence (for locking) and for using the key input sequence to unlock the system (for unlocking).
- For a higher level of security (e.g., the military application), we outline a re-locking enforcement mechanism with a minimal modification of D-FSM.
- Using digital watermarks, we show that a set of watermark properties and attack resiliency can be satisfied with the proposed method. Collision probability is estimated.

Preliminaries are provided in Section 2. The locking and unlocking method is described in Section 3. In Section 4, we consider digital watermarks as an application. An enforcement method is outlined in Section 5 and the conclusion is made in Section 6.

2. Preliminaries

2.1 Application model

Producer $P$ is an entity that has the ownership of a creative work $W$ (e.g., the design specification of a sequential system). Consumer $C$ is an entity that needs to use $W$. In the horizontal business model (e.g., an outsourcing model), the relationship between $P$ and $C$ is asymmetric: $W$ is not transparent to $C$. In the semiconductor industry, an example of $P$ and $C$ can be the design specification of IC/RTL/HDL-level blueprint (i.e., modeled in D-FSM) and a foundry which needs to use the blueprint, respectively.

2.2 Preliminary background

The following assumptions (A1, A2, and A3) are commonly used in the related literatures [2, 3, 4]. A1 is general while A2 and A3 are specific in the related security application domain.
• **A1:** A system is modeled with a deterministic finite state machine (D-FSM). D-FSM = (I, O, S, δ, λ), where I, O, and S are finite sets of inputs, outputs, and states, respectively. δ: I × S → S is the state transition function. λ: I × S → O is the output function.

• **A2:** D-FSM has both a power-up state S_p and a starting state S_s.

• **A3:** The system (i.e., D-FSM) will function *only* when it is starting from S_s.

**Synchronicity:** The synchronicity of D-FSM has been studied [8, 12]. The study was conducted based on two notions: (1) uncertainty, and (2) synchronizing sequence. The primary results can be summarized as follows:

- **Uncertainty as non-increasing function:** Initially, a machine M can be in any one of its “n” states. That is, the initial uncertainty of a machine is the entire set of states. By applying a sequence of input, the set of states that the machine can move (i.e., future uncertainty) will not increase.

- **Synchronizing sequence:** Regardless of the initial state or the output, a synchronizing sequence of M is a sequence which takes M to a specific final state, if M possesses the sequence.

The systematic way of deriving a synchronizing sequence was developed using a synchronizing tree T [8]. T is a binary tree, where a root node represents the initial uncertainty and the depth of T corresponds to the length of binary input sequence. For a given M, T can be constructed until some node is associated with an uncertainty containing just a single element known as a *singleton* state (i.e., no uncertainty). For the detailed description, refer to [8]. For the specific example relevant to the proposed work, refer to our previous paper [7].

**2.3 Summary of previous work**

Much of work done in designing D-FSM-based security solutions are based on modifying the structure of D-FSM [2, 3, 4, 6, 12]. Some has added dummy states while others have introduced additional edges between states. Despite the original functionality that is preserved, both an additional overhead and the increased level of design complexity have been reported for this approach. Another approach, possibly more promising one than changing the structure of FSMs, is to extract a *physical* property from the system [1]. However, these approaches are different from our approach which is based on extracting logical property that is inherently embedded in D-FSM. Thus, our approach is specific design-agonistic and can be used during the pre-optimization phase. With the logical level extraction, any design optimization techniques should be more effectively used.

---

1 A deterministic finite-state machine D-FSM and simply, a machine M are used interchangeably.
3. Locking and Unlocking Method - General

We describe three necessary operations that are used in related applications: (1) creating a lock (locking), (2) embedding a lock, and (3) unlocking. The first two operations are performed by the producer P while the consumer C exercises the third operation.

3.1 Creating a lock (Locking)

A lock is fundamentally secret information created by P. Many possibilities of creating a lock exist. One way is to hide S. If this approach is taken, a specific sequence of input that will transit S to S is important and thus can serve as secret known as the pass key K. In this case, a lock can be defined as a pair of < S, K >. Another possible approach is to define a lock in terms of S (solely). In this case, however, the consumer C has to derive S by running some experiment (e.g., synchronizing experiments requiring the construction of a synchronizing tree). We take the former approach since it can potentially provide more flexibility to P and also less burden on C.

In a ‘n’-state D-FSM, any state can be chosen as a locking state, if a certain condition is met. The condition is that the chosen locking state S must be singleton in the synchronizing tree T. If there are more than one singleton states, we choose the one in the lowest level. If there are more than one singleton states in the same level, choose one randomly. In this way, the singleton state S will provide the longest path from itself to the root node (or vice versa). The steps for creating a lock (by P) are as follow: Given an “n”-state D-FSM,

---

1) Construct “synchronizing tree T”;
2) FOR LOOP (Level “1” = 1, 2, ..., j)
   a. If exist, record a set of all “singleton” states {S(1, 1), S(1, 2), ...}; /* at level “1”*/
3) Determine the singleton state S* at the lowest level “1”; if more than one states, choose one randomly at the lowest level “1”;
4) Work backward from S* to the node at the root level (level 0) to construct the “longest” “synchronizing sequence”; /* this sequence becomes the pass key K*/
5) Send K to Consumer C in a secure way.

---

Note that the complexity of algorithm is O(n^3) due to the construction of the synchronization tree T. Step 5 can be done using either public key system (e.g., PKI), symmetric key system (i.e., a shared key), or other method [14]. Also, note that in Step 5 only K will be sent to C. This is due to the property of “synchronicity” which will guarantee the D-FSM move to the correct initial state S, irrespective of the power-up state on C.

3.2 Embedding a lock

In our approach, a lock is (or should be) embedded in the D-FSM itself. There is no need to
explicitly embed the lock (which is done by the traditional method.) This is one of the key advantages of the proposed method.

### 3.3 Unlocking

Consumer C can unlock the system using $K_s$, the key input sequence, received from $P$ in Step 5 (Section 3.2). The steps for unlocking are as follow:

1. Power up the system; /* The system can be in “any” one of the states; initial uncertainty */
2. Apply $K_s$ to unlock the system; /* The system should be in the starting state $S_s$ */

Note that the unlocking operation is simple. However, it is simple only for $C$ who knows $K_s$. In the following section, we provide the analysis for the length (i.e., least upper bound) of pass key ($K_s$). It is known that the unlocking step is usually the most expensive step in many security applications [9, 16].

**Limitation:** Synchronizing sequence(s) do not exist in all sequential systems or D-FSMs. Thus, the proposed method cannot be applied to a sequential system or D-FSM which does not possess a synchronizing sequence. A possible solution is provided in Section 6.

### 3.4 Analysis – Length of Synchronizing Sequence ($K_s$)

Intuitively, the longer the length of $K_s$, the higher the security level is. Since it is known that the least upper bound on the length of a synchronizing sequence is unknown, we should focus on analyzing a range of values, $[Q_{min}(least upper bound), Q_{max}(least upper bound)]$.

Based on [11], we present the results that are relevant to the study of this work below.

**Theorem 1:** If an $n$-state D-FSM has a synchronizing sequence, or sequences, then it has one such sequence whose length is at most $\frac{n(n+1)(n-1)}{6}$.

**Proof:** To reduce the initial uncertainty $S_1S_2...S_n$ to a singleton uncertainty, it takes $1 + (1 + 2) + \cdots + (1 + 2 + \cdots + n - 1) = \sum_{k=2}^{n} \frac{k(k-1)}{2}$ since $\frac{k(k-1)}{2} = 0$ for $k = 1$.

$$\sum_{k=2}^{n} \frac{k(k-1)}{2} = \frac{n(n+1)(2n+1)}{6} - \frac{3n(n+1)}{6} = \frac{n(n+1)(n-1)}{6}$$

**Theorem 2:** For every $n$, there exists an $n$-state D-FSM which has a synchronizing sequence of length $(n - 1)^2$.

**Proof:** Refer to the proof in Theorem 13-5 [8]

**Corollary 1:** The least upper bound $L$ on the length of synchronizing sequence is bounded by $(n - 1)^2 \leq L \leq \frac{n(n+1)(n-1)}{6}$. 

5
**Proof:** Directly from Theorem 1 and Theorem 2 ■

**Theorem 3:** For every \( n \), there exists \( n^2 \) ways of constructing a pair of states \( S_p, S_s \).

**Proof:** In an \( n \)-state D-FSM, there exists \( “n” \) number of possible power-up states \( S_p \) and \( “n” \) number of possible starting state \( S_s \). Thus, the total number of pairs is \( n^2 \) ■

**Corollary 2:** The least upper bound \( Q \) on the number of ways of unlocking the system by trying all possibilities is bounded by \( n^2 (n - 1)^2 \leq Q \leq \frac{n(n+1)(n-1)}{6} \).

**Proof:** Directly from Corollary 1 and Theorem 3 ■

**Implication:** Corollary 2 says that if both \( K_s \) and \( S_s \) are unknown, the adversary would try this many possible ways to unlock the system (in the worst case). The feasibility of applying the synchronicity to the benchmark systems is shown in Table 1. (For the detailed discussion, refer to [7])

Table 1. The minimum and maximum least upper bound for the sample circuits using benchmark sequential circuits [ISCAS’89][5], where *: a number at least bigger than “1.9e+25”; **: a number at least bigger than “4.5e+30”

<table>
<thead>
<tr>
<th>Circuits</th>
<th>No. of FFs</th>
<th>No. of States</th>
<th>Least Upper Bound Q (min)</th>
<th>Least Upper Bound Q (max)</th>
</tr>
</thead>
<tbody>
<tr>
<td>s27</td>
<td>3</td>
<td>8</td>
<td>3136</td>
<td>3584</td>
</tr>
<tr>
<td>s820, s832</td>
<td>5</td>
<td>32</td>
<td>984064</td>
<td>3724629</td>
</tr>
<tr>
<td>s1488, s1492, s386, s510</td>
<td>6</td>
<td>64</td>
<td>16257024</td>
<td>119275520</td>
</tr>
<tr>
<td>s208</td>
<td>8</td>
<td>256</td>
<td>4261478400</td>
<td>122166094506</td>
</tr>
<tr>
<td>s27, n3</td>
<td>9</td>
<td>512</td>
<td>68451303424</td>
<td>3909359763456</td>
</tr>
<tr>
<td>S1196, s1238</td>
<td>18</td>
<td>262144</td>
<td>4.7e+21</td>
<td>1.3e+26</td>
</tr>
<tr>
<td>s991</td>
<td>19</td>
<td>524288</td>
<td>7.5e+22</td>
<td>4.4e+27</td>
</tr>
<tr>
<td>s382, s400, s444, s526, s526n</td>
<td>21</td>
<td>2097152</td>
<td>1.9e+25</td>
<td>4.5e+30</td>
</tr>
<tr>
<td>s635, s838</td>
<td>32</td>
<td>4294967296</td>
<td>*</td>
<td>**</td>
</tr>
<tr>
<td>s15850</td>
<td>597</td>
<td>5.1e+179</td>
<td>*</td>
<td>**</td>
</tr>
<tr>
<td>s35932</td>
<td>1728</td>
<td>2^119</td>
<td>*</td>
<td>**</td>
</tr>
</tbody>
</table>

4. Digital Watermarking - Application

We consider digital watermarking as an application. First, we do the simple mapping between the general locking/unlocking method (i.e., three operations in Section 3) and the watermarking process. Then, we address the properties of watermarks and attack resiliency. Finally, we analyze the collision probability.

4.1 Mapping

A digital watermarking process is very similar to the three operations described in Section 3.1. The following three-step processes are used in digital watermarks [9, 13]: (a) signature (i.e., watermark) creation, (b) signature embedding, and (c) signature verification. The process can be directly mapped to the three operations.
4.2 Watermark Properties and Attack Resiliency

A digital watermark must satisfy a set of properties to provide security and resiliency against attacks [9]. We consider the following main properties (Table 2) and demonstrate that they can be satisfied by the methods proposed in Section 3.

Table 2. Properties of digital watermarks in a form of both general and specific properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Specific properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unobtrusiveness</td>
<td>The presence of signature (i.e., watermark) should not change/modify the original functionality of D-FSM.</td>
</tr>
<tr>
<td></td>
<td>The signature should be invisible to the design (i.e., D-FSM)</td>
</tr>
<tr>
<td>Universality</td>
<td>The same watermarking method should be applicable to all common sequential systems (i.e., D-FSMs)</td>
</tr>
<tr>
<td>Unambiguity</td>
<td>Retrieving the signature should be conclusive in proving the ownership.</td>
</tr>
<tr>
<td>Robustness</td>
<td>The embedded signature should be difficult to remove.</td>
</tr>
<tr>
<td></td>
<td>Multiple producers or individuals who create their signatures should have a unique set of characteristics (i.e., avoid the collision of watermarks)</td>
</tr>
</tbody>
</table>

In Table 3, we provide the informal reasoning for satisfying each of the watermark properties (P1.1 – P4.2) listed in Table 2.

Table 3. Satisfying the watermark properties with the reasoning

<table>
<thead>
<tr>
<th>Specific Properties</th>
<th>Reasoning</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1.1 Preservation of functionality</td>
<td>Any explicit changes to the design (i.e., D-FSM) are not made. The original functionality should be preserved.</td>
</tr>
<tr>
<td>P1.2 Invisibility</td>
<td>The signature is naturally embedded in the original design (D-FSM).</td>
</tr>
<tr>
<td>P2. Universality</td>
<td>The proposed method can be used for all common sequential systems, as long as the system possesses the synchronicity property. [NOTE: 94.6% of the benchmarks MCNC89 FSMs possess the synchronicity property [12]]</td>
</tr>
<tr>
<td>P3. Unambiguity</td>
<td>The probability of finding a valid sequence is very low without knowing the secret key pass. The retrieval of a signature is a strong proof of ownership. [For further improvement, see Section 5]</td>
</tr>
<tr>
<td>P4.1 Removal of signatures</td>
<td>The signature is an internal property of system. Removing an internal property of system will be not only difficult but will change the functionality of the system.</td>
</tr>
<tr>
<td>P4.2 Collision of signatures</td>
<td>The probability of collisions is very low. Guessing the secret key correctly (Ks) and the staring state (S0) will be very difficult. [See Section 4.3]</td>
</tr>
</tbody>
</table>

4.3 Analysis – Collision Probability

In this section, the collision probability is estimated. First, we describe the general problem. Then, the problem of estimating collisions for the proposed method is addressed using relaxed assumptions. Most of analysis in this section is performed based on [1].

General Problem: what is the probability that among K possible objects drawn from the population Ω = {1, 2, ..., n} at least two have the same value? (i.e., the birthday problem).
Specific Problem: Given a collection of $K$ binary sequences, each of length $L$, what is the probability of collision among the $K$ binary sequences?

Analysis: Consistent with the analysis in [1], the collision probability can be formulated as follows: Let $P_i$ denote the probability that the synchronizing sequence under consideration is the binary sequence $i$. There are a total of $K$ sequences with length $L$, where the minimum and maximum values of $K$ are 1 and $2^L$, respectively. Let $P = (P_1, P_2, ..., P_K)$ be the collection of probabilities of all the sequences. At least two cases should be considered:

Case 1: All the probabilities $P_i$ are equally likely (i.e., $P_1 = P_2 = \cdots = P_K = \frac{1}{K}$). Then, the probability of no match between $K$ binary sequences is given by $P \left( L, K, \frac{1}{K} \right) = \frac{\left( \frac{2^L}{K^L} \right)!}{\left( \frac{2^L}{K^L} (K^L - K)! \right)}$.

The probability of collision under this case (i.e., equally likely) is:

$$P_c^e = 1 - P \left( L, K, \frac{1}{K} \right) = 1 - \frac{\left( \frac{2^L}{K^L} \right)!}{\left( \frac{2^L}{K^L} (K^L - K)! \right)}$$  

(1)

Case 2: When the sequences are not equally likely, the probability of collision is generally given by the following formula:

$$P(collision) = 1 - P \left( L, K, P \right) = 1 - K! \sum_{1 \leq w_1 < \cdots < w_K \leq N} P(w_1)P(w_2)\cdots P(w_K)$$  

(2)

There are three sub-cases in specifying $P \left( L, K, P \right)$ in (2):

- **Sub-case 1:** The binary bits in each sequence are independent and identically distributed with probability $P(bit = 1) = \alpha$, $P(bit = 0) = 1 - \alpha$. Then, $P_i = (1 - \alpha)^{N(0,s)}\alpha^{N(1,s)}$, where the number of “zeros” and “ones” are denoted by $N(0,s)$ and $N(1,s)$, respectively.

- **Sub-case 2:** The bits in each binary sequence are independent but not identically distributed with probability $P(bit \ i = 1) = \alpha_i$, $P(bit \ i = 0) = 1 - \alpha_i$. If we define the function $G(b_i) = \alpha_i$ if $b_i = 1$, and $G(b_i) = 1 - \alpha_i$ if $b_i = 0$, then $P_i = \prod_{i=1}^{L} G(b_i)$.

- **Sub-case 3:** The bits in each binary sequence are correlated and their cumulative distribution function is $P$.

In [1], the authors suggested that the Nunnikhoven’s approximation [11] be used, since the exact solution of (2) is intractable.

**Estimation of collision probability ($L = d_{max} ln T$):** Since finding the exact collision probability is complex, we consider the worst case at level $L$ with the maximum number of nodes in the synchronizing tree $T$.

Worst-case scenario: From Theorem 1, $d_{max} = \frac{(n-1)^2n^2}{2}$. The number of nodes at level $L = d_{max}ln T$ should be between 1 and $2^d_{max}$. Since the maximum collision probability
should occur at the level where there is the highest number of nodes in $T$, all the nodes at level $L = d_{\text{max}}$ should be considered. In the worst case, there exist $2^{d_{\text{max}}}$ nodes. Suppose that the set of nodes at the last level is denoted by $V = \{v_1, v_2, \cdots, v_t\}$, where $t = 2^{d_{\text{max}}} = 2^{(n-1)^2(n)^2/2}$. Each node in level $L$ is any subset of $V$. Assuming that each element in the subset is equally likely, the probability of a node $v_i$ such that $v_i$ is singleton is:

$$P(v_i \in \text{singleton at } L = d_{\text{max}}) = p_s = \frac{t}{\sum_{i=1}^{t}{i}}$$

(3)

Then, the average number of singleton nodes at level $L = d_{\text{max}}$ will be $N(a, s) = p_s \cdot 2^{d_{\text{max}}} = p_s \cdot 2^{(n-1)^2(n)^2/2}$. Since the necessary condition for the existence of synchronizing sequence is that the node should be singleton, we can now consider only singleton nodes. Let $V^* = \{v_1^*, v_2^*, \cdots, v_N^*\}$ = a set of nodes with a singleton state at level $L = d_{\text{max}}$. Assuming that each node in $V^*$ occurs equally likely, we can apply the birthday problem mentioned above. Specifically, the maximum collision probability under the relaxed assumptions can be estimated as follow: Given $V^*$ with $N(a, s)$, the probability of collision is:

$$P(\text{collision}; L = d_{\text{max}}) = P_{\text{c}}(L = d_{\text{max}}) = 1 - P(2^{(n-1)^2(n)^2/2}, N(a, s), 1/N(a, s))$$

(4)

Note that (4) denotes the worst-case, meaning that the collision probability will not exceed this number. Due to the extreme calculation, we were unable to run the simulation even for a reasonable value of $n$. However, we expect that the collision probability is very low. This is due to the fact that the number of nodes in level $L = d_{\text{max}}$ is exponentially growing (i.e., $2^L$) and the number of possible subset of states (i.e., uncertainty) should grow accordingly. On the other hand, the ratio of the expected number of singleton nodes (i.e., the necessary condition) should decrease at level $L = d_{\text{max}}$ (i.e., the lowest level in the synchronization tree $T$).

5. Enforcement Method

For a stronger security solution, an additional security countermeasure that can prevent an adversary from attempting to break the key input sequence by a repeated trial search. The proposed method is to add a self-loop singleton state to D-FSM. That is, a minor modification to D-FSM is needed and can be done as follow:

1) Create a redundant state $S(r)$ and add it to the original D-FSM;
2) Add a self loop to $S(r)$; /* $\delta(S(r), 0) = \delta(S(r), 1) = S(r)$ */
3) Identify the path $P_c$ (critical path) = $S(1)^*$, $S(2)^*$, $\cdots$, $S(t)^*$; /* Given a derived key sequence $K_s = I(1)^*I(2)^*\cdots I(p)^*$, $P_c$ can be constructed; $S(t)^* = S_s$, $t = p + 1$ */
4) Add a set of edges from $S(1)^*$, $S(2)^*$, $\cdots$, $S(t)^*$ to $S(r)$ such that $\delta(S(1)^*$, NOT($I(1)^*$)) = $S(r)$, $\delta((S2)^*$, NOT($I(2)^*$)) = $S(r)$, etc.
Fig. 1. (a) shows the segment of an original D-FSM. The modified system is shown in Fig. 1(b). The modified system becomes a non-deterministic FSM (ND-FSM). However, any ND-FSM can be converted to a deterministic FSM [8]. Despite the number of nodes in the converted D-FSM will increase, the locking/unlocking method presented in the paper can be applied.

Fig. 1 Segments of original system and modified system where the original system is a deterministic FSM and the modified system is non deterministic FSM.

6. Conclusion

We proposed a sequential locking and unlocking method using a system’s logical property of synchronicity. The proposed method can be performed without changing or modifying the original system modeled in D-FSM, as long as the synchronicity property exists in the D-FSM. One possible solution of resolving this limitation (i.e., no existence of synchronicity) is to incorporate a partial scan so that a small subset of final states (i.e., non singleton states) is made to be directly controllable by external input.

7. Acknowledgments

This material is based upon work supported by, or in part by, the U. S. Army Research Laboratory and the U. S. Army Research Office under contract/grant number W911NF1210060.

References


*Corresponding author: Edward Jung, Ph.D.*

Department of Computer Science and Software Engineering
Southern Polytechnic State University
1100 South Marietta Parkway
Marietta, GA 30060, USA
E-mail: ejung@spsu.edu
An Overview of Data Privacy in Multi-Agent Learning Systems

Kato Mivule
Computer Science Department
Bowie State University
Bowie MD, USA
Mivulek0220@students.bowiestate.edu

Darsana Josyula, Claude Turner
Computer Science Department
Bowie State University
Bowie MD, USA
djosyula@bowiestate.edu, cturner@bowiestate.edu

Abstract—Public and private sector entities continuously produce, store, and transact in large amounts of data. However, combined with the growth of the internet, such datasets get stored and accessed on multiple devices, locations, and across the globe. Therefore, the necessity for autonomous agents that can learn across distributed systems to extract knowledge from large datasets while at the same time taking into account data privacy considerations while interacting with other agents remains a challenge. In this paper, we endeavor to provide an overview of data privacy in multi-agent learning systems, while at the same time highlighting current challenges and future areas of work and research.

Keywords: Multi-Agents; Inductive Learning; Data Privacy

I. INTRODUCTION

Public and private sector entities constantly generate, collect, and transact in large quantities of data (big data). However, with the growth of the internet, such datasets are stored and retrieved across numerous devices, and localities, across the globe. Therefore, there is necessity for artificial intelligence (AI) agents that can learn across distributed systems to extract knowledge from large datasets while at the same time taking into consideration data privacy and security issues in relation to other independent agents.

The problem of privacy and security in multi-agent systems has been an area of research interest for some time. As of 1996, Forner (1996) observed that the handling of sensitive data in multi-agent systems was still problematic due to privacy enhancing design challenges in multi-agent systems; Forner (1996) suggested cryptographic solutions to deal with privacy issues in multi-agent systems [34]. Wong et al. (2000) further addressed the problem of security and trust in multi-agent systems and proposed a security and trust architecture that ensured that agents do not act in contradiction to their designed purpose and that agents self-authenticate to ensure trust by retaining traits of correct naming and matchmaking services, secure communication channels, secure delegation when acting on behalf of other agents, and accountability [35]. However, Yu et al. (2003), succinctly and aptly observed that in multi-agent systems, privacy may have various meanings and importance for different agents; and that when designing architectures for multi-agent systems, there should be room for a diversity of perceptions and views on privacy [36]. In this article, we take this conceptual approach to privacy preservation in multi-agent systems. It is very difficult to define precisely what privacy is and therefore it becomes problematic to create a generalized solution to privacy complications.

As Spiekermann (2012) observed, one of the challenges of engineering privacy is that privacy is a fuzzy concept often confused with security, and, as such, difficult to implement [40]. Additionally, Friedewald et al. (2010) in their research on the legal characteristics of privacy, made a critical observation, that privacy is an evolving and shifting complex multi-layered concept, described differently by different people [41]. To add to this point, Katos et al. (2011) noted that privacy is a human and socially driven distinctive made up of human mannerisms, perceptions, and opinions [39]. Therefore, definitions for data utility get taken in the same light as privacy that is, data utility is the concept of how useful a privatized dataset is to the user of that particular privatized dataset [11]. Furthermore, despite various approximation methods that have been developed and designed to quantify data utility, researchers have noted that data utility varies from one scenario to the next, and, as such, problematic to have a generalized data utility gauge [12]. We believe that it is imperative that such fuzzy definitions of privacy and utility be taken into consideration when engineering privacy in multi-agent systems to avoid the pitfall of a generalized one-size-fits-all model.

Moreover Ramchurn et al. (2004) gave a detailed overview of the problem of trust in multi-agent systems due to the interactions that agents have in such environments; they observed three main aspects of problematic areas of trust in multi-agent systems: (i) how to engineer protocols for multi-agent interactions, (ii) how would agents decide who to interrelate with, and (ii) how agents decide when to cooperate with each other [37]. In their survey of security issues in multi-agent systems, Jung et al. (2012) made an important observation that multi-agent systems have become critical to autonomous computing today and therefore matters of security such as access control and trust are issues that need to be addressed [38]. This argument is further exemplified by Martins et al. (2012) in their review of security mechanisms in mobile agents, by pointing out the security threats that multi-agents face the need for agents to conform to the three canons of privacy and security, namely, confidentiality, accessibility, and integrity [26]. Lastly, Nagaraj (2012) observed that the analysis of security requirements for multi-agents, and, in this case, privacy requirements, is often neglected during the requirements phase of designing multi-agents [25]. Therefore, we believe that it is essential that any architecture, design, and
engineering of multi-agent systems seriously take privacy and security issues into consideration.

A number of data privacy enhancing algorithms have been suggested. Yet adopting the proposed algorithms for privacy preservation among autonomous agents remains a challenge. In multi-agent systems, communication and learning among the various autonomous agents involve dealing with privacy and security issues when one considers what sensitive and personal information autonomous agents can or cannot share. An example would include how multi-agents would transmit data in a health care system in which compliance to Federal and state laws require that personal identifiable information (PII) be kept confidential. Although a number of ongoing challenges exist for multi-agents in a distributed environment, in this paper we focus on data privacy issues in multi-agent learning systems as presented in current literature. The remaining part of the paper is ordered as follows: In Section II, we take a look at background of multi-agents as described in the literature. Section III deals with how multi-agents learn while in Section IV, we look at current data privacy issues in multi-agent learning systems. In Section V, we outline a conceptual architecture for privacy preserving multi-agent learning systems, and, finally, in Sections VI and VII, we provide our conclusion while highlighting future areas of research.

II. BACKGROUND

Agents: Wooldridge (2003) defined agents as computer systems that are located in a particular environment with the capability of independent and autonomous action in that particular environment so as to achieve the goals of what they were designed to do [1]. Multi-Agents: Wooldridge (2003) further described multi-agents as a group of autonomous agents combined into one system, independently solving simpler problems while communicating with each other to accomplish bigger and complex objectives [1]. Multi-agent systems (MAS): Da Silva (2005) noted that multi-agent systems are formed to deal with complex applications in a distributed systems environment. Da Silva (2005) also observed that examining data in distributed environments is a difficult problem since agents face several restrictions; for example, limited bandwidth in wireless networks and privacy issues with sensitive data [2]. MAS characteristics: However, Albashiri (2010) illuminated in his dissertation that MAS are defined by the following three traits [3]: (i) MAS essentially have to stipulate proper communication and interfacing protocols to efficiently connect with other agents; (ii) MAS need to be open and distributed with no previous information of other agents and their activities; (iii) MAS may consist of conceivably diverse agents that are distributed in that particular environment and acting independently or cooperatively to accomplish an objective. Machine Learning: Machine learning was described earlier by Samuel (1959) as the ability to encode and train computers to learn from experience and ultimately eliminate the necessity for the much exhaustive programming effort [4]. However, a more concise and commonly used formal description was given by Tom Mitchell (1997): “A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E” [5].

Big data: According to IBM, a private sector business leader in handling large amounts of data, ‘big data’ is a collection of large quantities of data that hold the following four characteristics, (i) volume, concerned with the large amounts of data, (ii) velocity, which has to do with the utilization of data as it is being produced, (iii) variety, concerned with various data types, from text, numeric, image, video, and sound, just to mention a few, (iv) veracity, as in such data must be authentic and secure for transaction [6]. Data privacy and security: Pfleeger et al. (2006), identified data privacy as a controlled disclosure in which an entity decides when and to whom to disclose its data, while security has to do with access control, as in who is allowed legitimate access to data and systems [8]. The three aspects of information security are further described by Pfleeger et al. (2006) as: (i) confidentiality, ensuring the concealment and privacy of data and systems, (ii) availability, ensuring the availability of data and systems at all times, and lastly, (iii) integrity, ensuring that data and systems are altered by only the authorized [8]. Data de-identification is the exclusion of personally identifiable information (PII) from a data set [9, 10]. PII attributes are properties that uniquely identify an individual; an example includes social security number. Data utility versus privacy is the concept of how beneficial a privatized dataset is to the user of that dataset. Achieving a balance between privacy and utility needs remains an intractable problem requiring trade-offs [11, 12, 13, 14].

III. LEARNING IN MULTI-AGENT SYSTEMS

Researchers have been fascinated by multi-agent learning for some time, and although a number of learning approaches have been proposed, in this paper we focus on two learning methods from literature to highlight the need for integration of data privacy principles in multi-agent learning systems. In an extensive review, Davies (1994) noted that Inductive Logic Programming (ILP) techniques were deployed as software agents for first order knowledge discovery in distributed databases [7]. Davies (1994) described how users are able to instruct a group of agents to discover information from particular databases. In general, a user presents an objective, and then the agents cooperate with other agents to accomplish this goal. Davis (1994) employed a combined approach with empirical first order inductive learning (inductive logic programming), data mining, and software multi-agent systems [7].

Moreover, Davies et al. (1995) explained in additional detail how agents learn in stages while discovering information in a distributed environment [15]: First phase: agents gather data in a centralized location. Second phase: agents interchange information while learning on resident data. Third phase: agents learn locally and then distribute results among fellow agents, after which the results are retuned and absorbed by other agents based on their own
data and knowledge. Davies et al. (1995) categorized agents in a distributive environment as: Non Distribution Agents: agents learn from local training examples. Incremental Theory Revision Agents: agents learn a local theory from existing training examples, and then share the learnt theory to the next agent. Simple Knowledge Integration Agents: agents learn a local theory, get tested on the training examples, and after comparison of results, the agent with the best theory is chosen. Theory Revision and Simple Knowledge Integration Agents: multiple agents learn a local theory and distribute the learned local theories to all the other agents. At this point each local agent then revises the received theories to fit local data, after which the agent tests each theory with the local training set and chooses the best theory after comparison of results [15].

Support Vector Machines (SVM), Multi-agents, and Incremental Learning: A description of how SVM based agents learn was given by Caraga et al. (2002) in which SVM based incremental learning involves an agent working on a dataset $D_1$ to produce a group of support vectors $SV_1$, the results of $SV_1$ are then added to dataset $D_2$ to produce dataset $D_2'$; after, another SVM based learning agent processes dataset $D_2'$ generating $SV_2$ results. The process continues, utilizing datasets $D_1$ and $D_2$ until a resulting classifier is learned, such that $D = D_1 U D_2$ [16]; where $D_1$ and $D_2$ are datasets, $SV_1$ and $SV_2$ are a group of produced support vectors. However, in their paper on the subject of SVM multi-agents and refuse data Ontanon et al. (2005) expounded on the cooperative learning of SVM multi-agents that utilized an ensemble effect for learning, by basically engaging in negotiating activities to improve individual agent and collective committee agent performance. Such agents have the capability of self-assessment and making decisions that some data used for learning is not needed [17]. Multi-agents situated in a distribution environment engage in communication and transaction of data and therefore questions of how such autonomous agents can learn by integrating data privacy and security principles remains a challenge.

IV. PRIVACY ISSUES IN MULTI-AGENT SYSTEMS

Privacy preserving architectures for multi-agent systems have been proposed but they mainly focus on access control rather than confidentiality. For instance, Cissee (2003) proposed a privacy preserving information filtering agent based architecture in which private or sensitive information was neither controlled by the user or provider of data gathering service but user and provider profiles could be shared between the two parties based on a trust relationship and thus filter any untrusted party [18]. In addition, Crepin et al., (2009) proposed specification for Hippocratic multi-agent systems in which each transaction of data requires a provider’s consent, limited collection of data, limited use of data, limited disclosure of data, limited retention of data, safety, and openness of data transactions by the multi-agents [19]. Another instance of access control trade-offs was the proposal by Leaute et al. (2009) in which multi-agents employ constraint satisfaction techniques, often used in resource allocation problems, and might consider trade-offs of their privacy constraints and decisions in the privacy preservation process [20]. Also, Such et al. (2012), proposed a self-disclosure system in which autonomous agents make decisions whether to disclose personal attributes to other agents mirrored after human relationships in which cost benefits are considered before disclosing private information [21]. Challenges of privacy preservation in multi-agent systems still remain an open research problem. Such et al. (2012), observed that multi-agents are vulnerable to three information-related activities: (i) information collection in which agents collect and store data about an individual, (ii) information processing whereby agents modify data that has been collected, and finally (iii) information dissemination, whereby agents publish data [22].

Klusch et al. (2003) observed then that one of the major challenges with distributed data mining was the issue of autonomy and privacy of agents in a distributed environment [23]. Albashiri (2010) indicated yet another challenge that multi heterogeneous agent systems, have to specify suitable communication and interfacing protocols and must be decentralized with new agents connecting at will by adapting to the communication protocol [3]. However, Rashvand et al. (2010) showed that multi-agent security requirements might appear in three categories: (i) service-agent protection, in which agents are protected from external threats; (ii) system vulnerability protection, in which the platforms and agents are protected from insecure internal processes; lastly, (iii) protective security services, in which the main objective of an agent is to provide security [24].

![Figure 1. Conceptual Categories of Data Privacy Multi-agents.](image-url)
distributed system is still problematic and a challenge, whereby, (i) agents have to learn how to sense privacy violations; (ii) how such a multi-agent system can be managed without centralization to deter and halt confidentiality abuses; (iii) and the need to find flexible solutions to the inapplicability of most existing privacy enhancement methodologies [28].

V. AN ARCHITECTURE FOR PRIVACY PRESERVING MULTI-AGENT LEARNING SYSTEM

Observations from our literature review on privacy issues in multi-agents, show that a number of research challenges still exist, mainly, how to integrate privacy and security principles in multi-agent learning architectures. In our conceptual contribution, we suggest an organizational structure as shown in Figure 1, that categorizes privacy preserving multi-agents as: (i) Confidentiality agents, those that handle data concealment and privacy; (ii) Integrity agents, those that handle non repudiation in data transactions, ensuring that data is altered by only authorized agents; and lastly, (iii) Availability agents, these are agents that ensure that all other agents are available for communication and that their resources are available at all times, by preventing and reporting attempted denial of service attacks.

In this way, the data privacy multi-agent system would conform to the three aspects of data privacy and security, that is, confidentiality, integrity, and availability. Communication between these multi-agents in the various levels of the architecture is a must. Secondly, we could have various data privacy roles under the specified major categories; for instance, since our focus in this paper is on data privacy preservation, under the Confidentiality category, we suggest data privacy algorithm selector multi-agents that would autonomously check what type of dataset that it is handling (categorical or numerical) as shown in Figure 2. If the data is numerical, then an agent applies Noise addition or Differential privacy data privacy algorithms [30]. If the data is categorical, the agent applies k-anonymity algorithm, Suppression, or Generalization data privacy algorithms [31, 32] on that dataset. Another agent could be employed for a hybrid solution. A different agent measures and reports on the data utility of the privatized dataset. Additionally, in this suggested framework, under the confidentiality multi-agent, we could have privacy and utility trade-off agents as shown in Figure 3. These agents would ensure the privacy and utility of privatized datasets, first, by outlining the various levels of parameters in the data privacy process.

These agents could belong to different groups based on the parameters in the data privacy process as shown in Figure 4. General overall data utility goal agents: these would ensure that the general utility or the overall goal parameters like accuracy, currency, and completeness are attained [30].

In this case, agents would ensure how accurate, how current, and how complete a privatized dataset ought to be. The data privacy enhancing algorithm parameters tuning agents: These agents would be responsible for autonomous adjustment and fine-tuning of parameters in the selected data privacy algorithm to ensure that not too much privacy is added while data utility diminishes. Finally, the machine learning parameter tuning agents: these agents would make adjustments to the parameters of the machine learner, such as increasing the number of weak learners. Even when multi-agents fully apply data privacy algorithms on data, the question of how such autonomous agents would have to...
learn to deal with the intractable problem of privacy versus utility, as illustrated in Figure 5; and how to make the trade-offs, remains open for further research.

![Data Privacy and Data Utility Diagram]

**Figure 5.** Trade-offs between privacy and utility are sought.

To illustrate this point, we added Differential privacy to a democratic political donation dataset, made public by the US Federal Election Commission and available online [33].

<table>
<thead>
<tr>
<th>Original Data</th>
<th>Data after Differential Privacy</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>125.46</td>
</tr>
<tr>
<td>100</td>
<td>122.72</td>
</tr>
<tr>
<td>100</td>
<td>145.15</td>
</tr>
<tr>
<td>25</td>
<td>106.57</td>
</tr>
<tr>
<td>5</td>
<td>66.04</td>
</tr>
<tr>
<td>5</td>
<td>69.41</td>
</tr>
<tr>
<td>100</td>
<td>131.59</td>
</tr>
<tr>
<td>30</td>
<td>62.32</td>
</tr>
<tr>
<td>50</td>
<td>123.38</td>
</tr>
<tr>
<td>30</td>
<td>99.22</td>
</tr>
</tbody>
</table>

**Table I.** Original Data Before and Synthetic Data After Privacy Enhancement

![Data Privacy versus Data Utility Chart]

**Figure 6.** Trade-offs between privacy and utility are sought.

Our goal was to create a synthetic dataset that met the requirements of differential privacy so as to conceal donations made by individuals; and while that was possible our results showed that the privacy added was at the cost of data utility. For instance, as shown in Figure 6, someone who gave a donation of US $25 is reported in the privatized database as giving US $106.57. While concealment is provided, the utility of that data diminishes.

Therefore, finding the optimal balance between privacy and utility remains a challenge for multi-agents. How autonomous agents could be trained to learn to achieve to such optimality and make trade-offs in the privacy versus utility challenge remains an open question for further investigation.

**VI. CONCLUSION**

In this article, we have endeavored to give a preliminary overview on privacy preservation in multi-agent learning systems in a distributed data systems environment. Our review of multi-agent data privacy issues from literature shows that the intractable problem of privacy in distributed data mining and machine learning is still a challenge with questions such as how can multi-agents in a distributed environment keep their autonomy and ensure privacy of data without disclosure of sensitive and personal information. The need for intelligent multi-agents that can learn how to discern private and sensitive data, and ensure confidentiality while communicating with other agents remains a challenge.

**VII. FUTURE WORK**

For future work, we plan to implement our conceptual privacy preserving multi-agent learning architecture, run simulation tests including automated software prototype, identify a data privacy and utility taxonomy for the prototype, and generate empirical results to map out the optimal balance between privacy and utility needs for various data privacy scenarios.

**ACKNOWLEDGMENT**

Special thanks to the Computer Science Department at Bowie State University for all the assistance in making this work possible.
REFERENCES


Abstract

Many organizations transact in large amounts of data often containing personal identifiable information (PII) and various confidential data. Such organizations are bound by state, federal, and international laws to ensure that the confidentiality of both individuals and sensitive data is not compromised. However, during the privacy preserving process, the utility of such datasets diminishes even while confidentiality is achieved—a problem that has been defined as NP-Hard. In this paper, we investigate a differential privacy machine learning ensemble classifier approach that seeks to preserve data privacy while maintaining an acceptable level of utility. The first step of the methodology applies a strong data privacy granting technique on a dataset using differential privacy. The resulting perturbed data is then passed through a machine learning ensemble classifier, which aims to reduce the classification error, or, equivalently, to increase utility. Then, the association between increasing the number of weak decision tree learners and data utility, which informs us as to whether the ensemble machine learner would classify more correctly is examined. As results, we found that a combined adjustment of the privacy granting noise parameters and an increase in the number of weak learners in the ensemble machine might lead to a lower classification error.

Keywords: Differential Privacy, Privacy Preserving Classification, Ensemble Machine Learning

1. Introduction

Organizations that transact in large amounts of data have to comply with state, federal, and international laws to guarantee that the privacy of individuals and other sensitive data is not compromised. However, during the privacy preserving process, when personal identifiable information (PII) is removed and privacy preserving algorithms such as differential privacy are applied, the utility of such datasets diminishes even while confidentiality is achieved—a problem that has been defined as NP-Hard [1-5]. Therefore, we investigate and present preliminary results of preserving differential privacy with machine learning ensemble classifier approach that maintains data privacy while maintaining a level of utility. In this study, we apply a strong data privacy granting technique on datasets using differential privacy. After which, we pass the perturbed data through a machine learning ensemble classifier. One of the aims of this study is to find a satisfactory threshold value that is measured by adjusting the differential privacy noise levels, with the aim of reducing the classification error. (It is important to note that a lower classification error tends to produce higher utility.) Additionally, the association between increased number of weak decision tree
learners and data utility to validate whether the proposed ensemble machine learner can classify differentially private data accurately is examined. The rest of this paper is organized as follows. Section 2 presents background and related work. Section 3 describes our methodology and experiment. Section 4 discusses results. Finally, Section 5 provides the conclusion.

2. Background and Related Work

2.1. Essential Terms

The following definitions will be essential in this paper in the context of privacy preserving classification. Privacy preservation in data mining and machine learning is the protection of private and sensitive data against disclosure during the data mining process [6, 7]. Ensemble classification is a machine learning process, in which a collection of several independently trained classifiers are merged so as to achieve better prediction. An example includes a collection of independently trained decision trees that are combined to make a more accurate prediction [8-10]. The classification error of a model \( M_i \) is the summation of weights of each record in \( D_j \) that \( M_i \) classifies incorrectly, where \( \text{err}(X_i) \) is the classification error of record \( X_i \), and \( d \) is the length of \( D_j \). If the record was misclassified, then \( \text{err}(X_i) \) is 1 otherwise \( \text{err}(X_i) \) is 0 [27]. The classification error is computed as follows:

\[
\text{Error}(M_j) = \sum_{i=1}^{d} w_{ij} \text{*err}(X_i) 
\]

(1)

AdaBoost, also known as Adaptive Boosting, is a machine learning that utilizes several successive rounds by adding weak learners to create a powerful learner. At each successive round, a new weak learner is added to the ensemble classifier by adjusting weights with emphasis placed on misclassified data in earlier iterations [11-13]. The covariance of random two random variables \( X \) and \( Y \), \( \text{Cov}(X, Y) \), is a measure of how the two random variables change jointly. If \( \text{Cov}(X, Y) \) is positive, then \( X \) and \( Y \) grow simultaneously. If the covariance is negative, then either \( X \) increases and \( Y \) decreases, or \( Y \) increases while \( X \) decreases. If the covariance is zero, the random variables are uncorrelated [26]. \( \mu_X \) and \( \mu_Y \) indicate the means of \( X \) and \( Y \) respectively, and the variances by \( \sigma_X \) and \( \sigma_Y \) respectively. Then the covariance of \( X \) and \( Y \) is defined by \( \text{Cov}(X, Y) = E[(X-\mu_X)(Y-\mu_Y)] \). If we have a series of \( n \) measurements of \( X \) and \( Y \) written as \( x_i \) and \( y_i \) of length \( N \), where \( j, k \in \{1, 2, ..., n\} \), then the sample covariance can be used to estimate the covariance \( \text{Cov}(X, Y) \). Let \( \bar{y} \) denote the sample mean of the variable \( y \). Then, the sample covariance is given by

\[
c_{jk} = \frac{1}{N-1} \sum_{i=1}^{N} (x_{ij} - \bar{x}_j)(y_{ik} - \bar{y}_k) 
\]

(2)

The correlation measures the statistical dependence between two random variables \( X \) and \( Y \). The most familiar measure of this dependence is Pearson’s product-moment correlation coefficient (or Pearson’s correlation), given by \( \rho_{xy} = \frac{\text{Cov}(X, Y)}{\sigma_X \sigma_Y} \). If \( \rho_{xy} = +1 \), then there is a positive (increasing) linear relationship between \( X \) and \( Y \) [26]. If \( \rho_{xy} = -1 \), then there is a negative (decreasing) linear relationship between \( X \) and \( Y \). Values of \( \rho_{xy} \) between 1 and -1 indicates the degree of the linear relationship between \( X \) and \( Y \), with \( \rho_{xy} = 0 \) indicating that \( X \) and \( Y \) are statistically unrelated (or uncorrelated). If we have a series of \( n \) measurements of \( X \) and \( Y \) written as \( x_i \) and \( y_i \), where \( i \in \{1, 2, ..., n\} \), with sample means then the sample correlation coefficient can be used to estimate the population Pearson correlation \( r_{xy} \) between \( X \) and \( Y \). The sample correlation coefficient is written:

\[
r_{xy} = \frac{n \sum_{i=1}^{n} (s_i - \bar{s})(v_i - \bar{v})}{\sqrt{\sum_{i=1}^{n} (s_i - \bar{s})^2 \sum_{i=1}^{n} (v_i - \bar{v})^2}}
\]

(3)

Data Utility versus Privacy. Data utility is the extent of how useful a published dataset is to the consumer of that publicized dataset. In the course of a data privacy process, original data will lose statistical value despite privacy guarantees. While attaining an optimal balance between data utility and privacy remains an NP-hard task, it is a balance that is highly desired and is continually pursued [1-5]. Differential Privacy, proposed by Dwork [14, 15], is a data privacy algorithm that works by adding Laplace noise to query answers from databases in such a way that a user of the database cannot determine if a data item has been altered. In such settings, it becomes very difficult, if not impossible, for an attacker to gain knowledge about any information in the schema. Therefore, a data privacy process, \( q_0 \), is said to achieve \( \varepsilon \)-differential privacy if the responses to any identical query ran on databases \( D_1 \) and \( D_2 \) are probabilistically alike and as long as those results satisfy the following requirement [14, 15]:

\[\varepsilon\]
\[
\frac{P[q_n(D_1) \in R]}{P[q_n(D_2) \in R]} \leq \exp(\varepsilon), \tag{4}
\]

where \( D_1 \) and \( D_2 \) are the two schemas, \( P[\cdot] \) denotes the probability, \( q_n(D_i) \) is the privacy procedure on query results from the schema \( D_i \), \( R \) is the perturbed query results from the schemas, and \( \varepsilon \) is a measure of the amount of noise.

\section*{2.2. Privacy and Utility}

In recent years, there has been considerable research interest in privacy preserving classification, with much attention given to privacy preservation distributed data mining in which associates do data mining on private data held by other associates [18, 20, 22, 23]. For example, Gorai et al [21] have proposed utilizing bloom filters for privacy preserving distributed k-NN classification. Their experiments show that bloom filters do preserve privacy while conserving the correctness of classifications. However, there is a growing interest in investigating privacy preserving data mining solutions that provide a balance between privacy and utility [24]. Kifer and Gehrke [24] conducted an unprecedented, comprehensive study of data utility in privacy preserving data publishing by employing statistical approaches. In this statistical approach, they measured the probability distributions of the original data and anonymized datasets to enhance data utility. However, for the \( m \)-confidentiality algorithm, in which privatized datasets are made public with minimum information loss, Wong [1] described how achieving global optimal privacy while maintaining utility is an NP-hard problem. Yet still, researchers have continued to study possible tradeoffs between privacy and utility in which some sensitive data is either concealed or revealed, so as to grant both data privacy and information usefulness [2, 4, 5, 6, 24]. Yet, Krause and Horvitz [25] have noted that even such an endeavor of finding the tradeoffs between privacy and utility is still an NP-hard problem. Recently, researchers have also shown that while differential privacy has been known to provide strong privacy guarantees, the utility of the privatized datasets diminishes due to too much noise [16, 17]. Therefore, finding the optimal balance between privacy and utility still remains a challenge—even with differential privacy.

\section*{3. Methodology and Experiment}

This section describes our proposed approach and experiment. In the first step of the proposed approach a strong data privacy granting technique using differential privacy is applied to a given dataset. Next, the perturbed data is passed through a machine learning ensemble classifier that performs a measure of the classification error. This procedure repeats until a satisfactory threshold is attained. If the classification error threshold is not attained, the differential privacy noise levels are re-adjusted to reduce the classification error. Since a higher classification error indicates lower utility, the intention of our approach is increasing utility by diminishing the classification error. At last, the proposed ensemble machine learner is evaluated by examining the association between increased number of weak decision tree learners and data utility. Thus, we can verify whether the proposed approach can classify data more correctly or equivalently to improve data utility.

\[\text{Fig. 1. (a) Steps in our differential privacy classifier approach; (b) A flow chart description our differential privacy utility classifier approach}\]

\begin{itemize}
  \item \textbf{Step 1: Calculate the Max Difference.}
  \item \textbf{Step 2: Generate } \( b \) \text{ in Laplace } (0, b), \text{ where }
  \begin{align*}
  b = \frac{\Delta f}{\varepsilon}. \text{ The Max Difference } / \text{ epsilons.}
  \end{align*}
  \item \textbf{Step 3: Assign a small } \( \varepsilon \), \text{ epsilon value of 0.01.}
  \item \textbf{Step 4: Divide Max Difference by the } \( \varepsilon \), \text{ epsilon value: Max Difference } / \varepsilon.
  \item \textbf{Step 5: Generate Laplace Noise Distribution.}
  \item \textbf{Step 6: Add Laplace noise to original data set.}
  \item \textbf{Step 7: Send results to AdaBoost Fit Ensemble Machine Learner.}
  \item \textbf{Step 8: Run AdaBoost Fit Ensemble on both Training and Testing data sets.}
  \item \textbf{Step 9: Get the Classifier Error.}
  \item \textbf{Step 10: If Classifier Error } \ll \text{ Threshold,}
  \item \textbf{Step 11: Refine Differential Privacy by returning the Laplace noise in step 5.}
  \item \textbf{Step 12: Repeat steps 7 to 9.}
  \item \textbf{Step 13: Stop when Classifier Error } \ll \text{ Threshold.}
\end{itemize}

\[\text{Experiment: In our experiment, a Barack Obama 2008 campaign donations dataset made public by the Federal}\]
Election Commission is used [26]. The data set contained 17,695 records of original unperturbed data and another 17,695 perturbed records of differentially private data. Two attributes, the donation amount and income status, are utilized to classify data into three groups. The three groups are low income, middle income, and high income. Low income indicates donations in the range $1 to $49, middle income for donations $50 to $80, and high income for donations greater than $81. The range of donations in this dataset was from $1 to $100. Validating our approach, the dataset is divided into two non-overlapping parts, which comprise the training and testing datasets. That is, 50 percent of the original dataset is used as training and the remainder is used for testing. The same procedure is used for the differentially private dataset. The dataset is then imported into the Oracle 11g database management system, after which it is accessed via MATLAB 2012a with MATLAB Oracle ODBC Connector. Next, a query is performed to the Oracle database via the MATLAB ODBC connector functionality. As a last step, MATLAB functionality is utilized to implement differential privacy on the query response using the pseudo code and corresponding block diagram illustrated in Fig 1 (a) and (b), respectively.

4. Results

In this paper, we applied differential privacy to the original dataset before the classification process, with the aim of investigating how the classifier would respond to a differentially privatized dataset. We found that essential statistical characteristics were kept the same for both original and differential privacy datasets a necessary requirement to publish privatized datasets. As depicted in Table 1, the mean, standard deviation, and variance of the original and differential privacy datasets remained the same. There is a strong positive covariance of 1060.8 between the two datasets, which means that they grow simultaneously, as illustrated in Fig. 2. (a). However, figure 2(b) shows that there is almost no correlation (the correlation was 0.0054) between the original and differentially privatized datasets. This indicates that there might be some privacy assurances given, as it becomes difficult for an attacker, presented only with the differential privatized dataset, to correctly infer any alterations.

After applying differential privacy, AdaBoost ensemble classifier is performed. In particular, AdaBoostM1 classifier is used since it is a better technique when few attributes are used, as is the case with our dataset. As can be seen in Table 2, the outcome of the donors’ dataset was Low, Middle, and High income, for donations 0 to 50, 51 to 80, and 81 to 100, respectively. This same classification outcome is used for the perturbed dataset to investigate whether the classifier would categorize the perturbed dataset correctly. The training dataset from the original data (Fig. 3 (a)) showed that the classification error dropped from 0.25 to 0 when the number of weak decision tree learners increased. At the same time, the results changed with the training dataset on the differentially private data when the classification error dropped from 0.588 to 0.58 as the number of weak decision tree learners increased; however, this value remained constant with the increase in the number of weak decision tree learners. When the same procedure is applied to the test dataset of the original data using AdaBoost, as illustrated in Fig. 3 (b), the classification error dropped from 0.03 to 0 as the number of weak decision tree learners increased. However, when this procedure perform on the differentially private data, the error rate did not change even with increased number of weak decision tree learners in the AdaBoost ensemble, as depicted in Fig 3. (b).

In this study, we found that while differential privacy might guarantee strong confidentiality, providing data utility still remained a challenge. However, this study is instructive in a variety of ways. Specifically, it shows that; (1) the level of Laplace noise adapted in this experiment for differential privacy does affect the classification error (See Fig 2 and Fig 3), (2) increasing the number of weak decision tree learners did not have much of a significance in correctly classifying the perturbed dataset because the classification error almost remained the same at 0.58 for both the training and testing datasets of the differentially private data. Because of these issues, adjusting the differential private Laplace noise parameters, $\epsilon$, is essential for our study.

5. Conclusion

While differential privacy in combination with AdaBoost ensemble machine learning techniques might offer strong confidentiality guarantees, our results show that providing data utility in this context still remains a challenge.
Table 1. Statistical properties before and after differential privacy.

<table>
<thead>
<tr>
<th>Statistical Property</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean of Original Data</td>
<td>54.71825148</td>
</tr>
<tr>
<td>Standard Deviation of Original Data</td>
<td>32.56985386</td>
</tr>
<tr>
<td>Variance of Original Data</td>
<td>1000.795381</td>
</tr>
<tr>
<td>Mean of Differential Privacy Data</td>
<td>54.48757564</td>
</tr>
<tr>
<td>Standard Deviation of Differential Privacy Data</td>
<td>32.44447794</td>
</tr>
<tr>
<td>Variance of Differential Privacy Data</td>
<td>1022.042202</td>
</tr>
<tr>
<td>Correlation</td>
<td>0.005590515</td>
</tr>
<tr>
<td>Covariance</td>
<td>1060.795381</td>
</tr>
</tbody>
</table>

Table 2. Expected classifications.

<table>
<thead>
<tr>
<th>Original Donations</th>
<th>Classifier Before Privacy</th>
<th>Perturbed Donations</th>
<th>Classifier After Privacy</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Low Earning</td>
<td>-2.2459327577</td>
<td>Low Earning</td>
</tr>
<tr>
<td>5</td>
<td>Low Earning</td>
<td>39.64185207</td>
<td>Low Earning</td>
</tr>
<tr>
<td>50</td>
<td>Middle Earning</td>
<td>51.90167518</td>
<td>Middle Earning</td>
</tr>
<tr>
<td>50</td>
<td>Middle Earning</td>
<td>66.610595424</td>
<td>Middle Earning</td>
</tr>
<tr>
<td>100</td>
<td>High Earning</td>
<td>81.53029477</td>
<td>High Earning</td>
</tr>
<tr>
<td>100</td>
<td>High Earning</td>
<td>93.79161352</td>
<td>High Earning</td>
</tr>
</tbody>
</table>

Fig. 2. (a) Original data verses Differential private data; (b) Correlation scatter plot of original and differential private data

Fig. 3. (a) Classification error and decision trees in Training dataset; (b) Classification error and decision trees in Testing dataset

A number of concerns arise. First, the level of Laplace noise parameter, $\epsilon$, used in this experiment for differential privacy does have an impact on the classification error. Second, increasing the number of weak decision tree learners in the ensemble did not significantly affect the classification of the perturbed datasets (as showed in Fig. 3) with lower utility. The differentially private Laplace noise parameter, $\epsilon$, might have to be adjusted to achieve higher performance of the perturbed dataset. This would require the perturbed data to be as close to the original data as possible for accurate classification. That is, high classification rate can guarantee better utility. However, since such accurate classification would come with loss of privacy, appropriate tradeoffs must be made between privacy and utility. Additionally, we found that the AdaBoost techniques used for this study can be implemented for both privacy preserving data mining and a data utility measure in differential privacy. However, achieving optimal utility by granting privacy still remains one of the most critical research topics. As a continuation of this study, we will develop an optimization algorithm for the differentially private noise parameter. Also, various ensemble classifiers, such as Bagging, will be employed and ten-fold cross validation will be used to validate the results.
Acknowledgements

This work is supported by the Department of Education’s Historically Black Graduate Institution (HBGI) award.

References

Applying Data Privacy Techniques on Published Data in Uganda

Kato Mivule¹, Claude Turner²
Computer Science Department
Bowie State University
14000 Jericho Park Road Bowie, MD 20715, USA
¹mivulek0220@students.bowiestate.edu, ²cturner@bowiestate.edu

Abstract - The growth of information technology (IT) in Africa has led to an increase in the utilization of communication networks for data transaction across that continent. Thus, many in Africa have become increasingly dependent on the Internet for data transactions. In the country of Uganda, for example, exponential growth in data transaction has presented a new challenge. Namely, what is the most efficient way to implement data privacy? While studies on data privacy have been done for developed nations such as in the European Union, studies for data privacy implementation in emerging markets have been minimal. It is with such background that we discuss data privacy challenges in Uganda. We also present an implementation of data privacy techniques for a published Ugandan dataset and suggest how this approach may be generalized to provide data privacy in the country.

Keywords: Data Privacy; Database Security; Statistical Disclosure Control; k-anonymity; Tabular data.

1. Introduction

The exponential growth of Information Technology (IT) in Africa has led to an increase in data transaction across Africa's communication networks, with 110 million Internet users and 500 million mobile phone subscriptions as of 2010[1]. In Uganda's case, higher education institutions routinely post student admission and graduation data online and grant access to student records online [2]. The Ugandan Electoral Commission posted the national voter's register online [3][4]. While the Uganda Bureau of Statistics publishes statistical data routinely, and takes great care to remove personal identifiable information (PII), a review of the published datasets from other Ugandan entities such as educational institutions and the Electoral Commission of Uganda show PII was included in published datasets. At the same time a growing number of young Ugandans are fans of large Online Social Networks (OSN) like Facebook, resulting in large amounts of PII leaked from online auxiliary data sources.

While case studies on data privacy have been done for developed nations such as in the European Union, studies for data privacy and security implementation in emerging markets such as Uganda have been minimal [48]. Yet with the growth of the globalized economy and multinational entities, demands for data privacy and security while transacting in business in the emerging markets is critical. Therefore in this paper, we take a look at current data privacy and security laws and present an implementation of data privacy techniques for a published Ugandan dataset and suggest how this approach may be generalized to provide data privacy in the country.

The rest of this paper is organized as follows. Section 2 looks at current data privacy and security policies in Uganda. Section 3 describes related work on data privacy and security in Uganda. Section 4 talks at the essential data privacy terms used in this paper. Section 5 gives an overview on data privacy techniques discussed in this paper. Section 6 discusses the implementation while Section 7 presents the results; and finally, Section 8 provides the conclusion.

2. Data Privacy and Security Policies

In developed countries like the USA, data gathering institutions are bounded by state and federal privacy laws that require that privacy of individuals be protected. One example in the USA is the Privacy Act of 1974, Health Insurance Portability and Accountability Act (HIPAA) of 1996, and the Personal Data Privacy and Security Act of 2009, requiring entities to protect and secure PII in data [5][6][7]. The Ugandan constitution defines the rights of an individual to privacy in terms of interference, stating that no person shall be subjected to interference with the privacy of that person's home, correspondence, communication or other property, however, no precise definition is given in the context of PII, data privacy, and computer security [8]. Ugandan Bureau of Statistics Act of 1998 describes Ugandan government policy on data collected by the Ugandan Bureau of Statistics (UBS). Absent from that description is how non-governmental entities collect and disseminate data. The Ugandan Bureau of Statistics Act of 1998 does not discuss what PII is in the Ugandan context. The only close reference is the “removal of identifiers” before data is granted to researchers [9]. In this case “identifiers” is ambiguous and could perhaps reference 'names' but not 'geographical location'. However, UBS with expert care
does publish de-identified micro datasets online but at the same time, many entities in Uganda publish non-de-identified tabular datasets.

A look at documents from authorities that govern communication technology in Uganda, the Uganda Communications Commission (UCC) and the Ministry of Information and Communications Technology (ICT) show that policies on data privacy and security have not been clearly formulated [9][10][11][12][13][14]. In the USA for instance, PII could include an individual’s social security number yet in Uganda, social security numbers are non-existent; thus, the set of PII in the USA differs from that in Uganda. Therefore, there is a need to expand Uganda’s policy on how government and non-government entities collect and disseminate data. To date, no clear legal and technological data privacy framework exists in Uganda. Despite the absence of any clearly formulated policy on data privacy in Uganda, this work suggest the application of data privacy techniques that could be utilized to provide basic data privacy in this context.

3. Related work on data privacy in Uganda

Our study of the literature reveals that work on data privacy in Uganda and much of sub-Saharan Africa is sparse. To date and to the best our knowledge, this work’s focus on the application of data privacy techniques to the Ugandan context might be novel. While research on computer security in Uganda exists, most of the work centers on network accessibility control methodologies [15][16][17][18][19]. For example, Mutyaba [20] and Makori [21] offer an excellent presentation on cryptographic methodologies for computer security, and Okwongale and Ogao [22] discuss data mining techniques; however, privacy preserving data mining (PPDM) methodologies are not discussed. Bakibinga [23] has articulated the need for electronic privacy in Uganda from a policy view point. Frameworks for secure management of electronic records have been proposed by Luyombya [24], Ssekibule and Mirembe [25], and Kayondo [26]; however, these works focus on data security and access control. But data privacy differs from data security in that data privacy has to do with the confidentiality of data, while data security focuses on its accessibility. Even when a database system is physically secured, an inference attack could occur on published datasets [27]. It should be noted that the Ugandan Bureau of Statistics Act of 1998 does provide a legal framework for data privacy that focuses on data gathered by the UBS. What is absent from the Ugandan computational literature is the data privacy technological framework that entities other than the Ugandan Bureau of Statistics, such as health, academia, and private business could employ [28]. To date, no work has come to our attention on if data privacy methodologies employed by UBS have been applied to private sector. Therefore, it is in this light that we make the case for data privacy in Uganda and the need for more research on data privacy and PPDM methodologies tailored to the Ugandan and African context.

4. Essential data privacy terms

The following definitions will be important in the sequel: Data privacy is the protection of an individual’s data against unauthorized disclosure while Data security is the safety of data from unauthorized access [29] [30]. Personally identifiable information (PII) is any data about an individual that could be used to construct the full identity of that individual [31][32]. Data De-identification is a process in which PII attributes are removed such that when the data is published, an individual’s identity cannot be reconstructed [33] [34]. Data utility verses privacy has to do with how useful a published dataset is to a consumer of that published dataset [35] [36]. Often the usefulness of data is lost when PII and quasi-attributes, are removed or transformed; a balance between privacy and data utility is always sought [37]. It has been determined that achieving optimal data privacy while not distorting data utility is a continual NP-hard challenge [38]. Statistical databases are published data sets that do not change, in many cases released in aggregated format [39]. Attributes in statistical databases, are field names or columns [29]. PII attributes are properties that uniquely identify an individual; an example includes social security number. Quasi-attributes are attributes not in the PII category but can be used to reconstruct an individual's identity in conjunction with external data. Confidential attributes are attributes not in the PII and quasi-attributes category but contain sensitive information, such as salary, HIV status, etc. Non confidential attributes are attributes that individuals do not consider sensitive as causing disclosure. However, non-confidential attributes can still be used to re-identify an individual given auxiliary data, thus making the explicit description of what PII is and is not even more challenging [40]. Inference and reconstruction attacks are methods of attack in which separate pieces of data are used to derive a conclusion about a subject, in this case, reconstruct their identity [41].

5. Data privacy techniques

Data privacy methods are categorized as non-perturbative techniques in which original data is not modified, some data is suppressed or some sensitive details removed while with perturbative techniques, original data is altered or disguised so as to protect PII and sensitive data [29]. While a number of data privacy techniques exist, we focus on application of k-anonymity, suppression, and generalization. Suppression is a popular data privacy method in which data values that are unique and can be used to establish an individual's identity are omitted from the published dataset [42][43]. Generalization is a data privacy method in which attributes that could cause identity disclosure are made less informative. An example includes replacing the
gender attribute value with “person” instead of “Male” or “Female” [44]. *K-anonymity* is a data privacy enhancing mechanism that utilizes *generalization*, and *suppression* as outlined extensively by Samarati [45] and Sweeney [27]. *K-anonymity* requires that for a dataset with quasi-identifier attributes in database to be published, values in the quasi-identifier attributes be repeated at least *k* times to ensure privacy; that is, *k* > 1 [27]. However, achieving the optimal *k-anonymized* dataset has been shown to be an NP-Hard problem [46].

### 6. Data privacy implementation

In this section, we describe our implementation of basic data privacy algorithms on a Ugandan dataset, utilizing open source technologies that are freely available for all to download. In this way, nations from emerging markets such as Uganda could incur minimal costs when it comes to data privacy implementation. We express our implementation using the set theory notation, relational database notation, and lastly MySQL implementation. The initial step was to de-identify a Ugandan dataset of 1200 records from a Makerere University student admission list that is published publicly online by the University, by removing PII as defined by the US data privacy laws [3]. While no explicit data privacy laws exist in Uganda, we utilized the definitions of what constitutes PII as defined by the US data privacy laws (HIPAA), considering that they could be universally applicable. We employed SQL, utilizing MySQL Sevver, an open source tool freely available for download.

![Diagram of Data De-identification procedure utilizing k-anonymity](image)

**Figure 1:** A Data De-identification procedure utilizing k-anonymity

<table>
<thead>
<tr>
<th>RegNo</th>
<th>StudentNo</th>
<th>Lname</th>
<th>Fname</th>
<th>Mname</th>
<th>Sex</th>
<th>BirthDate</th>
<th>Nationality</th>
<th>Hall</th>
<th>Program</th>
<th>IndexNo</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>09U/EVE</td>
<td>20900</td>
<td>Amnet</td>
<td>Anna</td>
<td>F</td>
<td>01/01/67</td>
<td>UGANDAN</td>
<td>AFRICA</td>
<td>LIS</td>
<td>U0166</td>
<td>2008</td>
<td></td>
</tr>
<tr>
<td>09U/EVE</td>
<td>20901</td>
<td>Green</td>
<td>RICE</td>
<td>F</td>
<td>01/01/80</td>
<td>UGANDAN</td>
<td>MARY STUART</td>
<td>ARM</td>
<td>U0166</td>
<td>2008</td>
<td></td>
</tr>
<tr>
<td>09U/EVE</td>
<td>20902</td>
<td>Timothy</td>
<td>NICE</td>
<td>F</td>
<td>01/01/81</td>
<td>KENYAN</td>
<td>MARY STUART</td>
<td>BLE</td>
<td>U0065</td>
<td>2007</td>
<td></td>
</tr>
<tr>
<td>09U/EVE</td>
<td>20903</td>
<td>Jones</td>
<td>Jane</td>
<td>F</td>
<td>01/01/73</td>
<td>TANZANIA</td>
<td>MARY STUART</td>
<td>LIS</td>
<td>U0198</td>
<td>2007</td>
<td></td>
</tr>
<tr>
<td>09U/EVE</td>
<td>20904</td>
<td>Carter</td>
<td>James</td>
<td>M</td>
<td>01/01/74</td>
<td>UGANDAN</td>
<td>RAM</td>
<td>U0160</td>
<td>2007</td>
<td></td>
<td></td>
</tr>
<tr>
<td>09U/EVE</td>
<td>20905</td>
<td>Brown</td>
<td>Britain</td>
<td>N</td>
<td>01/01/83</td>
<td>KENYAN</td>
<td>AFRICA</td>
<td>ARM</td>
<td>U0715</td>
<td>2008</td>
<td></td>
</tr>
<tr>
<td>09U/EVE</td>
<td>20906</td>
<td>Sams</td>
<td>Sam</td>
<td>F</td>
<td>01/01/84</td>
<td>TANZANIA</td>
<td>MARY STUART</td>
<td>RAM</td>
<td>U0725</td>
<td>2007</td>
<td></td>
</tr>
<tr>
<td>09U/EVE</td>
<td>20907</td>
<td>Faster</td>
<td>Master</td>
<td>M</td>
<td>01/01/85</td>
<td>UGANDAN</td>
<td>BLE</td>
<td>U1148</td>
<td>2008</td>
<td></td>
<td></td>
</tr>
<tr>
<td>09U/EVE</td>
<td>20908</td>
<td>Uhuru</td>
<td>Kenya</td>
<td>F</td>
<td>01/01/90</td>
<td>UGANDAN</td>
<td>COMPLEX</td>
<td>ARM</td>
<td>U0062</td>
<td>2007</td>
<td></td>
</tr>
<tr>
<td>09U/EVE</td>
<td>20909</td>
<td>Vineyard</td>
<td>Martha</td>
<td>M</td>
<td>01/01/88</td>
<td>KENYAN</td>
<td>AFRICA</td>
<td>ARM</td>
<td>U0117</td>
<td>2008</td>
<td></td>
</tr>
</tbody>
</table>

**Table 1:** Admission List with PII – BirthDate, IndexNo, and RegNo are generalized

Steps in the Data Privacy Procedure shown in Figure 1:

INPUT: Data from relation or schema

OUTPUT: Data privacy preserving published tabular dataset

1. Identify PII Attributes
2. Remove PII Attributes
3. Identify quasi-identifier attributes
4. Generalize or Suppress quasi-identifier attributes
5. Check that *k* > 1 in tuples
6. Check for single values that cannot be grouped together to achieve *k* > 1
7. If single values and outliers exist, Generalize or Suppress until k-anonymity at *k* > 1
8. Check for utility
9. Publish tabular dataset

We borrowed from set theory notation to describe how we implemented the data privacy procedure on the Ugandan data set as follows:

- The original Ugandan published dataset included the following attributes, in which we let the following:
  - \( A = \{ \text{RegNo, StudentNo, Lname, Fname, Mname, Sex, BirthDate, Nationality, Hall, Program, IndexNo, Year} \} \), the relation *admission list* that included all attributes in the published dataset.
  - We let \( B = \{ \text{Lname, Fname, Mname, StudentNo, IndexNo, RegNo} \} \), the set of all PII attributes that we identified in the published dataset.
  - We let \( C = \{ \text{Nationality, Sex, BirthDate} \} \), the set of all quasi-identifier attributes identified in the dataset.
  - We let \( D=\{\text{Hall, Program, Year} \} \), the set of all non-sensitive attributes.
  - Lastly, we let \( E=\{\} \), the set of all sensitive attributes.

- Thus, we have \( B \subset A \), \( C \subset A \), \( D \subset A \) and \( E \subset A \):
  - \( \text{Therefore } A = B \cup C \cup D \cup E \), and \( A \neq \{B, C, \ldots\} \).
D, E).

- By removing PII, we get \(A = \{C, D, E\}\).
- The de-identification of the Admission List set involves a complement of the PII set: 
  \(B^c = U - B = A - B = C + D + E\). Therefore, we remained with the quasi attributes, non-sensitive attributes, and sensitive attributes; where \(U\) is the universal set, which in this case is all the Admission List attributes.

- We suppressed or generalized the quasi attributes: suppress or generalize (C).
  - We then applied \(k\)-anonymity: \(k\)-anonymity (\(B^c\)).
  - Finally, we ordered values of \(B^c\).
  - If \(k = 1\), we suppressed or generalized \(C\) until \(k > 1\).

Relational model view: For a formal relational model view implementation, we applied the following notation:
- we let \(\pi <\text{attribute list}>^{(R)}\),
  - where \(\pi\) is the projection or selecting of attributes from a relation (Table),
  - \(<\text{attribute list}>\) is the list of attributes from Admission List
  - \(^{(R)}\) is the relation from which we select attributes.

The original projection with all attributes is:
- \(\pi<\text{RegNo, StudentNo, Lname, FName, Mname, Sex, BirthDate, Nationality, Hall, Program, IndexNo, Year}>^{(\text{Admission List})}\).
- The projection void of PII attributes is:
  - \(\text{To_Be_Published_List} \leftarrow \pi<\text{Sex, BirthDate, Nationality, Hall, Program, Year}>^{(\text{Admission List})}\).
  - We apply \(k\)-anonymity to the list that is to be published:
    - \(k\)-anonymity(\(\text{To_Be_Published_List}\)).

7. Results

We generalized the BirthDate attribute to further prevent any reconstruction attacks by first developing a domain generalization hierarchy (DGH). We chose the DGH based on the oldest person in the dataset, and built our DGH to \(B_4 = \{196*\}\), giving protection for the individuals born in 1967 [43], as shown in Figure 2.

\[B_4 = \{196*\}\]
\[B_3 = \{1967\}\]
\[B_2 = \{1967-09\}\]
\[B_1 = \{1967-09-08\}\]

DGH

Figure 2: Domain generalization hierarchy structure

The SQL Implementation: We implemented data de-identification in SQL by creating a SQL View and doing

SELECT on the view by choosing only attributes that remain in the Admission List after removing PII. We created SQL Views that are void of PII attributes:

```sql
CREATE VIEW V2 AS SELECT Sex, BirthDate, Nationality, Hall, Program, Year FROM Admission_List;
```

Generalization: Utilizing the SQL functions, CREATE, SELECT, and UPDATE, we further generalized the Program attribute so as not to grant such information to a researcher. We generalized the BirthDate attribute to additionally prevent any reconstruction attacks.

Table 2: Results after generalization and suppression

MySQL implementation:

```sql
CREATE table V2_Generalize1
SELECT Sex, BirthDate, Nationality, Hall, Program, Year
FROM V2;
UPDATE V2_Generalize1 set BirthDate = '1950-01-01' WHERE BirthDate BETWEEN 1950-01-01 AND 1999-12-31';
```

Suppression: In the case of achieving \(k\)-anonymity, we had to suppress some values that appeared once, yet still we had to ensure the utility of the data set, as too much suppression would kill the utility of the published dataset.

Table 3: Results after suppression, highlighted values to be further suppressed until \(k>1\)

MySQL implementation:

```sql
UPDATE V2_Generalize1 set Hall = 'WHERE Hall = 'Complex';
```
Check for $k$-anonymity that $k > 1$ by ordering data:

MySQL implementation:

```
SELECT Sex, BirthDate, Nationality, Hall, Program, Year
FROM V2 ORDER BY Sex, Program, Hall;
```

$k$-anonymity achieved at $k > 1$, where $k$ is each value in the quasi attributes repeated at least $k > 1$ times.

### Table 4: Results after we achieve $k$-anonymity at $k > 1$

<table>
<thead>
<tr>
<th>Sex</th>
<th>BirthDate</th>
<th>Nationality</th>
<th>Hall</th>
<th>Program</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>196</td>
<td>UGANDAN</td>
<td>AFRICA</td>
<td>LIS</td>
<td>2008</td>
</tr>
<tr>
<td>F</td>
<td>196</td>
<td>UGANDAN</td>
<td>MARY STUART</td>
<td>ARM</td>
<td>2008</td>
</tr>
<tr>
<td>F</td>
<td>196</td>
<td>KENYAN</td>
<td>MARY STUART</td>
<td>BLE</td>
<td>2008</td>
</tr>
<tr>
<td>F</td>
<td>196</td>
<td>TANZANIA</td>
<td>MARY STUART</td>
<td>LIS</td>
<td>2008</td>
</tr>
<tr>
<td>M</td>
<td>196</td>
<td>UGANDAN</td>
<td>RAM</td>
<td>2007</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>196</td>
<td>KENYAN</td>
<td>AFRICA</td>
<td>ARM</td>
<td>2008</td>
</tr>
<tr>
<td>F</td>
<td>196</td>
<td>TANZANIA</td>
<td>MARY STUART</td>
<td>RAM</td>
<td>2008</td>
</tr>
<tr>
<td>M</td>
<td>196</td>
<td>UGANDAN</td>
<td>BLE</td>
<td>2008</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>196</td>
<td>UGANDAN</td>
<td>ARM</td>
<td>2007</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>196</td>
<td>KENYAN</td>
<td>AFRICA</td>
<td>ARM</td>
<td>2008</td>
</tr>
</tbody>
</table>

Removing names and student numbers entirely diminishes utility, in that the data becomes meaningless to students who simply want to view it to see if their names are on the university admission list. One way this problem can be dealt with is by publishing a list that includes the student number or student names while obscuring other PII data. However, in both scenarios, the issue of balancing data utility and data privacy remain quite challenging and demands tradeoffs [47].

### 8. Conclusion

We have made the case for the need to revamp Uganda's data privacy policy to encompass both private and government sectors on how to gather and disseminate data, and the need to implement data de-identification techniques. With the growth of data transaction in Uganda, there is a need for more research on how to implement privacy preserving data publishing and privacy preserving data mining methodologies tailored to the Ugandan context, with applications ranging from academia, government, health sector, and private sector. We have shown that with freely available open source technologies, some level of data privacy can be implemented on datasets from emerging markets. However, the problem of what PII constitutes in the emerging market nations still remains. Although no set of PII has been proposed in Uganda, we suggest that PII include any information that could specifically identify an individual in the Ugandan context. This could include: full names, face, fingerprints, handwriting, genetic data such as DNA, national ID number, driver's license number, passport number, credit and debit card numbers birth-date, birth place, village of residence, city of residence, county of residence, phone number, and student examination numbers. Applying the $k$-anonymity procedure might be practicable in the Ugandan context; however, achieving optimal privacy while maximizing utility continues to be an NP-hard problem, as data is lost through generalization and suppression process. Therefore more studies need to be done on various implementations of optimal data privacy tailored to Ugandan context; with consideration that PII differs in Uganda from other geographical locations.

### 9. References


A Comparative Analysis of Data Privacy and Utility Parameter Adjustment, Using Machine Learning Classification as a Gauge

Kato Mivulea and Claude Turnerb a*

aBowie State University, 14000 Jericho Park Road, Bowie, MD, 20715

Abstract

During the data privacy process, the utility of datasets diminishes as sensitive information such as personal identifiable information (PII) is removed, transformed, or distorted to achieve confidentiality. The intractability of attaining an equilibrium between data privacy and utility needs is well documented, requiring trade-offs, and further complicated by the fact that making such trade-offs also remains problematic. Given such complexity, in this paper, we endeavor to empirically investigate what parameters could be fine-tuned to achieve an acceptable level of data privacy and utility during the data privacy process, while making reasonable trade-offs. Therefore, we present the comparative classification error gauge (Comparative x-CEG) approach, a data utility quantification concept that employs machine learning classification techniques to gauge data utility based on the classification error. In this approach, privatized datasets are passed through a series of classifiers, each of which returns a classification error, and the classifier with the lowest classification error is chosen; if the classification error is lower or equal to a set threshold then better utility might be achieved, otherwise, adjustment to the data privacy parameters are made to the chosen classifier. The process repeats x times until the desired threshold is reached. The goal is to generate empirical results after a range of parameter adjustments in the data privacy process, from which a threshold level might be chosen to make trade-offs. Our preliminary results show that given a range of empirical results, it might be possible to choose a tradeoff point and publish privacy compliant data with an acceptable level of utility.

Keywords: Data privacy; utility; machine learning classifiers; comparative data analysis

1. Introduction

During the privacy preserving process, the utility of a dataset – a measure of how useful a privatized dataset is to the user of that dataset diminishes as sensitive data such as PII is removed, transformed, or distorted to achieve confidentiality [1, 2, 3]. Yet, finding equilibrium between privacy and utility needs remains intractable, necessitating trade-offs [4, 5, 6]. For example, by using the suppression of data – that is removing or deleting sensitive attributes before publication of data, privacy might be guaranteed to an extent that the PII and other sensitive data is removed.

* Corresponding author. Tel.: 443-333-6169
E-mail address: mivulek0220@students.bowiestate.edu
However, a researcher using such a published dataset might find it difficult to fully account for some categorical entries despite confidentiality guarantees provided to that privatized dataset. In another example involving numerical data, employing high noise perturbation levels to provide privacy for sensitive numerical data might render the privatized dataset useless to a researcher as too much noise distorts the original traits of the data. Given such intricacy, in this paper, we attempt to empirically explore what parameters could be adjusted to attain an adequate level of data privacy and utility during the data privacy process, while making practical trade-offs. We present the comparative classification error gauge (Comparative x-CEG) approach, a data utility quantification model that uses machine learning classification techniques to gauge data utility based on the classification error. The remaining part of this paper is ordered as follows. Section 2 discusses background and related work. Section 3 presents our methodology and experiment. In section 4, we present preliminary results. Finally, in section 5, provides concluding remarks.

2. Background and Related Work

The task of achieving a satisfactory level of data utility while preserving privacy is a well-documented NP-hard problem that would necessitate trade-offs [7, 8]. Additionally, the problem is further complicated by the fact that attaining such trade-offs also remains intractable. Researchers have observed that the problem of preserving confidentiality while publishing useful statistical data is extensive and that the trade-off between the privacy and the utility of any anonymization technique, would largely depend on the level of the attacker’s background knowledge about a particular dataset, with higher levels indicating that it would be impossible to achieve any such trade-offs [9, 10, 11]. Li and Li (2009) indicated in their study that it is not promising to equate privacy and utility as datasets get distorted when they are privatized, leading to a decline of utility [12]. Even with the latest state of the art data privacy algorithms like differential privacy, confidentiality is guaranteed but at a major loss of data utility [13]. As Dwork (2006) concisely put it [14], “Perfect privacy can be achieved by publishing nothing at all, but this has no utility; perfect utility can be obtained by publishing the data exactly as received, but this offers no privacy”. In other words, the more confidential data is made to be, the more likely that the privatized data will become useless and decline in utility.

The privacy definition problem: On the other hand, there happens to be no specific standard metric to define privacy, as Katos, Stowell, and Bedner (2011) observed, that privacy is a human and socially driven characteristic comprised of human traits such as acuities and sentiments [15]. Dayarathna (2011) stated that to wholly comprehend the notion of data privacy, an all-inclusive methodology to outlining data privacy should include the legal, technical, and ethical facets [16]. This point is additionally exemplified by Spiekermann (2012) who noted that one of the difficulties in designing and engineering data privacy is that the idea of privacy is fuzzy, frequently confused with data security, and as a result, very problematic to implement [17]. Adding to this point, Friedewald, Wright, Gutwirth, and Mordini (2010), in their research on the legal aspects of data privacy stated that since privacy is an evolving and shifting complex multi-layered notion, being described and treasured otherwise by various people; and that empirical studies are needed to assess how different people define and value privacy [18]. As Mathews and Harel (2011) observed, the human element remains a key factor and as such data privacy is intrinsically tied to fuzziness and evolutions of how individuals view privacy, and the same applies to data utility; that is, what information about themselves that individuals determine as fit to share with others [19]. Therefore, it becomes problematic to create a generalized data privacy and utility solution; however, different individuals and entities will have different data privacy needs and thus tailored data privacy solutions. Given the complexities of defining privacy, quantifying data utility is likewise problematic. However, various methods have been employed to enumerate data utility by basically quantifying the statistical differences between the original and privatized datasets, such as, the relative query error metric, research value metric, discernibility data metric, classification error metric, the Shannon entropy, and the information loss metric (mean square error) [20, 21, 22, 23, 24, 25]. However, in this paper, we suggest using the machine learning as a gauge, by using the classification error to adjust data privacy parameters until a desired threshold is attained.

2.1. Essential Terms

While a number of data privacy algorithms exist, in this paper, the subsequent methods are used. **Noise addition**: is a data privacy algorithm in which random numbers selected from a normal distribution with zero mean and a standard deviation are added to data for obfuscation, and is expressed by [26, 27]:

-\[ \text{Noise addition: } \]
\[ X + \varepsilon = Z \]  

(1)

\( X \) is the original numerical data and \( \varepsilon \) is the set of random numbers (noise) with a distribution \( e \sim N (0, \sigma^2) \) that is added to \( X \), and \( Z \) is the privatized dataset [26, 27]. **Multiplicative noise:** random numbers whose mean \( \mu = 1 \) and a small variance \( \sigma^2 \) are produced, and then multiplied to the original data, generating confidential data, expressed as follows: [27, 28]:

\[ X_j \varepsilon_j = Y_j \]  

(2)

Where \( X_j \) is the original data; \( \varepsilon_j \) is the random numbers with mean \( \mu = 1 \) and a small variance \( \sigma^2 \); \( Y_j \) is the confidential data after multiplying \( X_j \) and \( \varepsilon_j \) [27, 28]. **Logarithmic noise:** makes a logarithmic modification of the original data as shown below [26, 28]:

\[ \ln X_j = Y_j \]  

(3)

Random values \( \varepsilon_j \) are then produced and added to the data that underwent a logarithmic alteration \( Y_j \). Lastly, the confidential data, \( Z_j \), is generated as given in Equation (4) [26, 28]:

\[ Y_j + \varepsilon_j = Z_j \]  

(4)

Note that the original data is represented by \( X_j \); \( Y_j \) is the logarithmic altered data values; and \( Z_j \) is the confidential data.

3. Methodology

*The Comparative x-CEG approach:* We present the Comparative Classification Error Gauge (x-CEG) approach. This approach is motivated by a need to investigate how a dataset that is subjected to various data privacy algorithms performs under different machine classifiers. Empirical results from this process are then gathered and a comparative analysis is performed to determine the best classifier. In the comparative x-CEG approach, a data privacy algorithm \( d_1, ..., d_n \) is applied on a dataset, where \( d_1, ..., d_n \) are the various data privacy procedures, as shown in Fig 1. The privatized datasets \( p_1, ..., p_n \) is generated, where \( p_1, ..., p_n \) are the new privatized datasets with each \( p_i \) corresponding to \( d_i \). The privatized datasets \( p_1, ..., p_n \) are passed through a series of machine learning classifiers, \( c_1, ..., c_n \) where \( c_1, ..., c_n \) is the different machine learning classifiers. Statistical properties of the privatized datasets, after applying the various data privacy algorithms \( d_1, ..., d_n \) are quantified. The classification error from each classifier \( c_i, ..., c_n \) is then measured and the classifier with the lowest classification error is chosen. If the classification error of that chosen classifier \( c_i \) is lower or equal to a set threshold, then better utility might be achieved; otherwise, an adjustment of the data privacy parameters of \( d_i \) and or the machine learning parameters of the chosen classifier. \( c_i \), is made. The results are resent to the classifier \( c_i \) and a measure of the classification error is done again. The procedure replicates until a desired classification error using the chosen classifier \( c_i \) is achieved, indicating a better utility for the user of that privatized dataset. The advantages of this approach is that multiple checks for utility using various classifiers is achieved but also a more in-depth comparative analysis is done and as such a knowledge of which machine learning classifier works best on which dataset. Secondly, since a large dataset of empirical data is generated, choosing a reasonable threshold (trade-off) point becomes feasible.

4. Experiment

This section presents the experiment setup and preliminary results to be used. The Iris Fisher multivariate dataset from the UCI repository [29], with both numeric and categorical data was used. One hundred and fifty data points were used as the original dataset. The dataset consisted of numeric attributes: sepal length, sepal width, petal length, and petal width. It also included one categorical attribute with three classes, Iris-Setosa, Iris-Versicolor, and Iris-Virginica. Three data privacy algorithms were applied on the original data set; namely, noise addition, logarithmic noise, and multiplicative noise. The comparative x-CEG algorithm was applied, using KNN, Neural Nets, Decision Trees, AdaBoost, and Naïve Bayes classification methods. MATLAB and RapidMiner were employed as tools in this experiment. The original Iris dataset with 150 records was imported into MATLAB and then vertically partitioned with the first partition containing the continuous data and the second partition containing categorical data of class labels. Data privacy algorithms were applied on the continuous portion of the data, and the categorical
portion was used in classification of the data. The datasets were then set to RapidMiner, and a series of machine learning classification techniques was applied on both the original Iris data and the various privatized Iris datasets, allowing for an implementation of the Comparative x-CEG concept.

![Diagram of the Comparative x-CEG procedure]

**Fig. 1.** The Comparative x-CEG procedure

### 4.1. Results

![The Comparative x-CEG Matrix for the privatized Iris data set](image)

**Fig. 2.** (a) The comparative x-CEG classification error results; (b) The suggested tradeoff point (threshold) at 0.2 classification error
Fig. 3. Statistical distribution for the original data and privatized data using noise addition, logarithmic, and multiplicative privacy

4.2. Discussion

As shown in Fig. 2. (a), using the original Iris dataset as a benchmark, the lowest classification error was 0.04 for the original data using KNN, with 0.06 for AdaBoost Ensemble for the highest classification error. For a normal original Iris dataset without privacy, classification error is less than 1 percent. However, after applying noise addition privacy method, the classification error is at average 0.3, approximately 30 percent misclassification with the non-adjusted mean and variance. Still, after adjusting noise addition parameters, with the mean set to 0 and variance at 0.1, the classification error drops on average to 0.04, similar to the original dataset. While such close results might be appealing on a data utility basis, the privacy of such a dataset is significantly diminished as the dataset becomes similar to the original. Additionally, the classification error for both logarithmic and multiplicative noise is on average 0.4, approximately 40 percent misclassification. On the privacy basis, both logarithmic and multiplicative noise provides better privacy. And, as shown in Fig. 3, in the scatter plots, it is very difficult to separate or cluster data after applying logarithmic and multiplicative noise. However, the utility of such datasets is not appealing when approximately 40 percent of the data is misclassified. As illustrated in Fig. 2. (b), after generating empirical data, a preferred threshold or trade-off point classification error is set at 0.2, approximately 20 percent misclassification. However, this could be lower, depending on user privacy and utility needs.

5. Conclusion

Employing the Comparative x-CEG methodology by adjusting data privacy parameters could generate adequate empirical data to assist in selecting a trade-off point for preferred data privacy and utility levels. However, more rigorous empirical studies are needed to further test this hypothesis. Furthermore, finding the optimal balance between privacy and utility needs remains an intractable problem, and, as such, for future work, we seek to study optimal trade-off points based on larger empirical datasets, and on a case by case basis. Also, we plan to conduct analytical and empirical work comparing various data privacy algorithms not covered in this study.
References

Abstract

An important step in speaker recognition is extracting features from raw speech that captures the unique characteristics of each speaker. The most widely used method of obtaining these features is the filterbank-based Mel Frequency Cepstral Coefficients (MFCC) approach. Typically, an important step in the process is the employment of the discrete Fourier transform (DFT) to compute the spectrum of the speech waveform. However, over the past few years, the discrete wavelet transform (DWT) has gained remarkable attention, and has been favored over the DFT in a wide variety of applications. This work compares the performance of the DFT with the DWT in the computation of MFCC in the feature extraction process for speaker recognition. It is shown that the DWT results in significantly lower order for the Gaussian Mixture Model (GMM) used to model speech and marginal improvement in accuracy.

© 2011 Published by Elsevier B.V. Open access under CC BY-NC-ND license.

"Keywords: Cepstral Coefficients, Speaker Recognition, Wavelets"

1. Introduction

Speaker recognition is the identification of persons from their speech or voice samples (Furui, 1997; Campbell, 1997; Bimbot et al., 2004). Speaker recognition can be divided into three specific tasks: (1) identification, (2)
verification, and (3) segmentation and clustering (Solomonoff et al., 1998). Speaker identification determines who among a group of speakers provided the input speech to the system (Furui, 1997). It has two modes of operation: closed-set and open-set. In closed-set mode, the input speech is assumed to belong to one of the speakers in the group. The open-set case allows for the situation where the input speech could belong to speakers outside of this group. Speaker verification determines if a person is whom he/she claims to be from the person’s voice samples. Synonyms for speaker verification include, voice verification or authentication, speaker authentication, talker verification or authentication, and speaker detection (Jin, 2007). In a multiple speaker scenario where the speech is a heterogeneous mixture of speech from various speakers, speaker segmentation and clustering partitions the speech into speaker-homogeneous regions. Specific speaker recognition applications include forensics, diarization (attempts to extract speaking turns of different participants from a spoken document), and telephone based systems that recognize the speaker. Speaker recognition systems may be further categorized as text-dependent or text-independent. In text dependent recognition, the phrase to be spoken is known to the system. In the text independent case, the system does not have any a priori knowledge of the phrase. Text-independent speaker recognition is the more challenging of the two categories.

At its most fundamental level, the speaker recognition problem may be cast as a pattern recognition problem (Jin, 2007). As such, it can be partitioned into two modules: (a) a training module, and (b) a classification module. The classification module can be further divided into two components: pattern matching and decision. The feature extraction module estimates a set of features from the speech signal that represent some speaker-specific information. The speaker-specific information results from complex transformations occurring at multiple levels of the speech production process: semantic, phonologic, phonetic, and acoustic (Atal, 1976; Campbell, 1997). Despite the variety among the categories of speaker-specific information, there are only a small set of criteria that they must satisfy. These are discussed by Nolan and Wolf (Nolan, 2009; Wolf, 1972). The pattern matching module is responsible for comparing the estimated features to the speaker models. There are many types of models that could be used in speaker recognition, including Gaussian Mixture Models (GMMs), Hidden Markov Models, and vector quantization (VQ). The decision module analyzes the similarity score(s), which could be either statistical or deterministic, to make a decision. The decision process is dependent on the system task. For the closed set identification task, the decision could be to select the identity associated with the model that is most similar to the test sample. In open-set applications, the systems can also require a threshold to verify whether the similarity is valid. Figure 1 depicts the speaker recognition system, and the dashed box represents the pattern matching module.

A typical speaker recognition system usually consists of two phases: an enrollment (or training) phase, and an authentication (or testing) phase. In the enrollment phase, the user speaks an appropriate phrase into a microphone or similar device attached to the system. The system then extracts speaker-specific information from the speech signal to be used to build a model for the speaker. GMM (Reynolds, 1995) is one of the most popular methods for the modeling process. The purpose of the testing phase is to determine whether the speech samples belong to one of the registered speakers. As in the training phase, speech features are extracted from the speech signal presented. The speaker is then determined by finding the speaker model which yields the maximum posterior probability for the input feature vector sequence (Reynolds, 1995).

All audio processing techniques begin with feature extraction—the conversion of the raw speech signal to acoustic vectors that characterize speaker-specific information. Although a variety of filterbank-based methods exist for feature extraction, such as Linear Prediction Cepstral Coefficients (LPCC) (Makhoul, 1975) and Perceptual
Linear Prediction Cepstral Coefficients (PLPCC) (Hermansky, 1990); the Mel Frequency Cepstral Coefficients (MFCC) (Davis & Mermelstein, 1980) approach has been the most widely employed for feature extraction (Ganchev et al., 2005). In recent years, numerous variations and improvements of the original MFCC idea have been proposed (Ganchev et al., 2005; Sigurdsson et al., 2005). This is mainly attributable to researchers’ efforts to exploit progress made in the area of psychoacoustics (Ganchev et al., 2005).

Wavelets have been employed in speaker recognition applications for over a decade with some success (Phan, 2000; Al-Ani, 2007). The goal of this work is to compare the performance of the discrete wavelet transform with the discrete Fourier transform in the speaker recognition process. Specifically, we will compare performance in terms of accuracy and efficiency between the DFT and the DWT for Daubechies’s first ten wavelets at six different decomposition levels.

2. Methods

2.1 Mel Frequency Cepstral Coefficients (MFCC)

A modular representation of a filterbank-based feature extraction model that generates the MFCC is depicted in Fig. 2. The speech signal is first pre-emphasized by applying the filter, $x(t) = y(t) - a \cdot y(t - 1)$, where $a \in [0.95,0.98]$. The goal of the filter is to enhance the high frequencies of the spectrum, which is diminished during the speech production process. Following the pre-emphasis stage is a windowing process, where a window whose size in duration is much smaller than the whole speech signal, is applied starting at the beginning of the signal, and then shifted to the right and applied, successively, until the end of the signal is reached. Two quantities must be set: the width of the window and the shift between consecutive windows. For the width of the window, two values are often used: 20ms and 30ms. These values correspond to the average duration necessary for the stationary assumption to hold. In the case of the delay, a value is chosen so that there is some overlap between consecutive windows. Ten milliseconds is often used. Once the width of the window and the shift between consecutive windows are found, the type of window can then be chosen. The Hamming and Hanning windows are most often used in speaker recognition. Next, for each of the windowed signals emerging from the windowing process, an N-point DFT is computed. Typically, N is chosen as a power of 2 and is classically 512 points, which is greater than the number of points in the window. Next, the modulus of the DFT for each of the spectral vectors is obtained, and from this the corresponding power spectrum for each is taken over 512 points. Since the signal is real valued, the spectrum is symmetric, thus only the first half plus one of it is kept--257 points.

The spectrum consists of much fluctuation. However, in this context, such details are not of interest. It is only the envelope of the spectrum that is of interest. Smoothing removes some of these details. To realize the smoothing and to get the envelope of the spectrum, the spectrum is multiplied by a filterbank. The filterbank is defined by the shape of the filters and by their frequency localization—left frequency, central frequency, and right frequency (Ganchev et al., 2005). Filters may be of a triangular or other shape, and can be differently located on the frequency scale. The Bark/Mel scale is sometimes used for frequency localization of the filters. It is an auditory scale that is similar to the frequency scale of the human ear. A commonly implemented equation for localization of the central frequencies, which is the one used in the experiments of this study, is given by:

$$f = 2595 \cdot \ln \left( 1 + \frac{f_{min}}{700} \right),$$  \hspace{1cm} (1)

The original filterbank of Davis and Mermelstein (Davis & Mermelstein, 1980), FB-20, is the one used here. It proceeds as follows: Given the N-point DFT of the discrete input signal, $x_n$, $\xi_k = \sum_{n=0}^{N-1} x_n e^{-\frac{jk\pi n}{N}}, k \in \{0,1,\ldots,N-1\}$, a filter bank with M equal height triangular filters is constructed. Each of the M equal height filters is defined by:
\[ h_i(k) = \begin{cases} 0, & k < f_{b_{i-1}} \\ \frac{k-f_{b_{i-1}}}{f_h-f_{b_{i-1}}}, & f_{b_{i-1}} \leq k < f_{b_{i}} \\ \frac{f_h-k}{f_h-f_{b_{i}}}, & f_{b_{i}} \leq k < f_{b_{i+1}} \\ 0, & k \geq f_{b_{i+1}} \end{cases} \]

\[ i \in \{1, 2, \ldots, M\}, \text{ where } i \text{ is the filter index, } f_{b_{i}} \text{ is the boundary point for the filter, } h_i, \text{ and } k \in \{1, 2, \ldots, N\} \] corresponds to the \( k \) th coefficient of the N-point Discrete Fourier Transform (DFT). Each boundary points \( f_{b_{i}} \) depends on the sampling frequency, \( f_s \), and the number of points, \( N \), in the DFT, and is given by:

\[ f_{b_{i}} = \left(\frac{N}{f_s}\right) \cdot f \cdot \left(\frac{f(f_{\text{high}}) - f(f_{\text{low}})}{M+1}\right). \]

The values \( f_{\text{low}} \) and \( f_{\text{high}} \) are, respectively, the low and high boundary frequencies for the entire filter bank, \( M \) is the number of filters, and \( \tilde{f}^{-1} \) is the inverse of (1) given by:

\[ \tilde{f}^{-1} = f_{\text{lin}} = 700 \left[ e^{\left(\frac{m}{260}\right)} - 1\right]. \]

The filter bank of Davis and Mermelstein is comprised of 20 equal height filters, which cover the frequency range [0,4600] Hz. The center frequencies for the first ten filters are linearly spaced between 100 Hz and 4000 Hz, while the next ten have center frequencies logarithmically spaced between 1000 Hz and 4000 Hz. The next step computes the logarithm of windowed signal followed by the discrete cosine transform. The process may be summarized compactly as follows:

\[ c_t = \sum_{k=0}^{M} X_k \cos \left( t \cdot (k - 1/2) \frac{\pi}{M} \right), \quad t \in \{1, 2, \ldots, J\}, \]

where

\[ X_k = \log_{10} \left( \sum_{k=0}^{M} |\tilde{h}_k|^2 \right). \]

2.2 Wavelets

Given low pass and high pass filters, \( g \) and \( h \), and associated scaling and wavelet functions, \( \varphi \) and \( \psi \), respectively, approximation and detail coefficients of \( x \) are obtained via the DWT and are given by:

\[ a_x(j + 1, k) = ((a_x(j, \cdot) * g) \downarrow 2)(k) = \sum_{m \in \mathbb{Z}} g_{2k-m} a_x(j, m), \quad d_x(j + 1, k) = ((a_x(j, \cdot) * h) \downarrow 2)(k) = \sum_{m \in \mathbb{Z}} h_{2k-m} a_x(j, m), \]

respectively, where \( j \in \{1, 2, \ldots, L\} \) and \( k \in \{0, 1, \ldots, n_j - 1\} \). * and \( \downarrow \) represent the convolution and downsampling operations, respectively, and \( n_j \) represents the number of approximation (or detail) coefficients at level \( j \). Also, \( a_x(j, \cdot) = \{a_x(j, 0), a_x(j, 1), \ldots, a_x(j, n_j - 1)\} \). We assume that \( j = 0 \) yields the pre-emphasized signal, \( x \), itself.

To obtain feature extraction filterbank coefficients using DWT, we substitute Eq. 6 for \( \tilde{x}_k \) in Eq. 5, to get:

\[ z_k = \log_{10} \left( \sum_{t=0}^{L-1} |D_t| \tilde{h}_k(t) \right), \quad \text{for some } j \in \{1, 2, \ldots, L\}. \]

The coefficients that results when the DWT is substituted for the DFT in Eq. 5 is then obtained by substituting \( z_k \) for \( X_k \) in Eq. 4, to get:

\[ w_t = \sum_{k=1}^{M} z_k \cos \left( t \cdot (k - 1/2) \frac{\pi}{M} \right), \quad t \in \{1, 2, \ldots, J\}. \]

3. Experimental Setup & Results

We used six Region 1 speakers from the TIMIT database—three males and three females—and the following single utterance from each: “She had your dark suit in greasy wash water all year.” Each speaker has a copy of this utterance stored in a file name sa1.wav. The following six speakers were used—three males and three females—from the TIMIT database: FECD0, FJSP0, FKFB0, MKLS0, MPGH0, and MPGR0. The first letter of the speaker designation tells us the gender. The next three letters following it are the first, middle and last initial of the speaker’s name. The last position makes it possible to distinguish multiple speakers with the same gender and initials—a zero indicates the first such speaker, a 1 for the second, etc. Fig. 4 gives the speech signal for FJSP0.

In the training phase of our experiment, Eq. 7 was implemented for each of the speaker signals, and a GMM was used to model the features obtained. In the testing phase, for each speaker, Eq. 7 was again used to extract the features, and then a maximum likelihood function was used to determine the model that best matched the input.
speech. The process was repeated for the first ten Daubechies’s wavelets—db1, db2, ..., db10—and for six decomposition levels of the DWT. These results were compared with the FFT approach given by Eq. 4. The value of $a$ used for the preemphasis was $a = 0.95$. The window size, and overlap used in the windowing module was, 320 and 160 samples, respectively. The filterbank was the original filterbank design of Davis and Mermelstein (Davis & Mermelstein, 1980), with 20 filters, $M = 20$, as discussed in Section 2.1. The order of the GMM (number multivariate Gaussian distributions used) was optimized using the Akaike Information Criteria (Akaike, 1974).

The results for the DWT are provided in Table I. Each value in the table is a measure of the number of times a speaker is misidentified. The best results are provided on Level 2 of the wavelet decomposition, with db1, db4, db6, and db10 identifying each speaker without error. This is an improvement over the DFT, which misidentified one out of the six speakers (error=1/6). Table II compares GMM order for the DWT versus the DFT. The row marked “DFT” gives the number of models used in the training phase for each speaker. There are four speakers, FECD0, FJSP0, MKLS0 and MPGH0, that have order 20. The other two speakers FKFB0 and MPGR0 have order 18 and 16, respectively. The results for the DWT, given on the row labeled “DWT,” show that the DFT require an order that is three to five times that of the DWT. A smaller optimal order is preferred because it leads to a GMM that is less computationally intensive to generate and use. Therefore, the DWT approach seems to provide marginal improvement over DFT in terms of its accuracy for speaker recognition with the MFCC. Further, the DWT provides significant improvement in terms of the optimal order required to generate the GMM.

<table>
<thead>
<tr>
<th>LEVEL</th>
<th>WAVELET</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>db1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
</tr>
</tbody>
</table>

4. Conclusion

This work compares the performance of the DFT with the DWT in the computation of the MFCC for feature extraction in speaker recognition. It showed that the speech features derived through the DWT resulted in a more efficient representation, in terms of order, for the GMM that used in the statistical modeling of features. It also showed marginal improvement in accuracy of the DWT over the DFT. Specifically, it was shown that the GMM order required when the DFT is used in the MFCC feature extraction process was approximately three to five times that required for the DWT. Finally, in terms of accuracy, the wavelet approach matches the DFT at decomposition Level 2. However, the DWT outperforms the DFT with an error rate of zero when the following wavelets are used: db1, db4, db6, and db10.
References


Forecasting Purchasing Managers’ Index with Compressed Interest Rates and Past Values

Anthony Joseph\textsuperscript{a}, Maurice Larrain\textsuperscript{b}, Claude Turner\textsuperscript{c} \textsuperscript{a}\textsuperscript{*}

\textsuperscript{a}Pace University, One Pace Plaza, New York 10038, USA
\textsuperscript{b}Bowie State University, 14000 Jericho Park Road, Bowie, MD 20715, USA

Abstract

The purchasing managers’ index (PMI) is a simple subjective survey about the state of the manufacturing sector of the national economy. It’s an early indicator of the nation’s economic strength with effects extending into federal monetary policy and the financial markets. It is a composite index comprising the weighted average of new orders, production, employment, supplier deliveries, and inventories. It has been established that inverted interest rates in 3-month Treasury bills is a predictor of PMI. This study extended the work on the compression of economic and financial predictor variables as well as the relative efficiency of temporal nonlinear neural network models in forecasting economic time series variables. It showed that compressed interest rates and PMI past values are also effective predictors of the future values of PMI. Less than 30% of the wavelet packets coefficients of interest rates were involved in accomplishing the forecasting task. The correlation, root mean square error, normalized root mean square error, mean absolute deviation, and Theil inequality metrics were used to determine the efficacy of the forecasts. The overall PMI forecast produced by the neural network models was relatively better than that produced by the regression models on all metrics except Theil inequality.

Introduction

Purchasing managers’ index (PMI) is a monthly seasonally adjusted weighted composite diffusion index of five indicators of economic activities in the manufacturing sector (Harris, 1991; Koenig, 2002; Lindsey & Pasvur, 2005; Raedels, 1990; Peláez, 2003). The five indicators are weighted as follows: 30% for new orders, 25% for production,
20% for employment, 15% for supplier deliveries, and 10% for inventories. The PMI is a subjective index, based on the reports from manufacturing firms’ purchasing managers. Its advantages include timeliness; reliability; and a predictor of changes in industrial production, real gross domestic product (GDP), real inventories, real sales, sales/inventory ratio, federal funds rate, foreign exchange returns, and in monetary policy (Berge & Jorda, 2011; Neely & Day, 2010; Ozylidirim et al, 2011). Some of PMI’s disadvantages are its subjective nature and the unaccounted economic impact of surveyed firms in survey responses. The PMI’s diffusion aspect makes it behave like a leading indicator. Koenig (2002) said that diffusion indices provide no information on the intensity of a time dependent relative change in values or a firms’ economic impact. Moreover, Harris (1991) found that PMI “is helpful in predicting contemporaneous manufacturing activity;” and “deserves at least part of its reputation as a key economic indicator.”

The PMI is not only subjective and simple but it is also enduring and meritorious as a forecasting tool of important economic variables (Peláez, 2003; Simpson & Ramchander, 2008). Because of the relative importance of the PMI’s ability to predict future economic activities including changes in real GDP and the growth rate, some researchers have called for improving its statistical robustness and quantification to give it greater accuracy and signaling power (Peláez, 2003). Moreover, other researchers see value in being able to accurately forecast the PMI so as to gain insight into the economy’s future direction and present models to accomplish this task (Larrain, 2007; Lindsey & Pasvur, 2005; Raedels, 1990).

Economic forecasting falls into two general groups: qualitative and quantitative, where the qualitative forecasting may merely involve intuition and speculative judgments while the quantitative forecasting could involve sophisticated statistical analyses (Granger and Newbold, 1973; Wisner & Stanley, 1994; Wolstenholme, 1999; Goodwin, 2002; Lindsey & Pasvur, 2005). The increased use of strategic planning and the need for cost containment among corporate managers necessitate hybrid forecasting strategies that integrate the right mix of qualitative and quantitative forecasting since both strategies have complementary strengths and weaknesses, and the fact that forecasting is a human activity (Wisner & Stanley, 1994; Wolstenholme, 1999; Goodwin, 2002; Lindsey & Pasvur, 2005). Nonetheless, business managers and decision makers are overly reliant on short term bias prone subjective forecasting because of their lack of training and familiarity in using more accurate statistical forecasting methods (Wolstenholme, 1999; Goodwin, 2001). When economists and business managers use statistical forecasting methods, they tend to overly favor regression analyses (Koop, 2006; Lindsey & Pasvur, 2005; Septhon, 2009; Wisner & Stanley, 1994). However, the increasing use of computers, data repositories, and ubiquitous data over the last 20 years are demanding more computational and automatic ways to efficiently mine, analyze, and forecast future economic conditions to provide information that afford a competitive advantage to firms in this ever changing dynamic business environment.

There is a limited but increasing use of neural network modeling techniques in various areas of economic forecasting (Larrain, 2007). This study is an extension of Larrain’s 2007 work on the relative forecasting of PMI with inverted 3-month T-bill interest rate using neural network models. The out-of-sample forecast in this study subsumes the 36 months covering September 2002 to August 2005 in Larrain’s (2007) work: it covers 159 months inclusive of May 1997 to July 2010. This study also differs from Larrain’s in that its forecast horizon is 12 months instead of 10; it uses past values of PMI as one of the explanatory variables; the explanatory interest rate variable is compressed to within 15% of its wavelet packet coefficients (Mix & Olejniczak, 2003); and it uses robust multiple regression (DuMouchel and O’Brien, 1989) instead of the regular linear regression as a basis of comparison for the time lagged recurrent neural network focused gamma models (Principe et al, 2000; Haykin, 1999). The performance measures used to test the accuracy of the forecasts are correlation and root mean square error for the individual forecasts and correlation (r), root mean square error (RMSE), normalized root mean square error (NRMSE), mean absolute deviation (MAD), and Theil inequality (Theil) for the overall forecasts.

Materials and Procedures

The raw interest rate and PMI data consisted of 552 monthly samples dated from August 1964 to July 2010. They were shown through scatter plots (Koop, 2006) to be generally not linearly related to each other even after preprocessing by compression, filtering, 12 month backward differencing, and mean removal. Their power spectral densities were concentrated within relatively few low frequency indices. The zero mean raw interest rate variable reached its highest power spectral density of 50.15 at index 2 and then rapidly tapered off to a spectral density of
0.93 at index 28 while the zero mean raw PMI, similarly, reached its highest power spectral density of 174.3 at index 8 and then rapidly tapered off to a spectral density of 5.53 at index 45. The power spectral densities of interest rate and PMI after the preprocessing showed similar behaviors. The power spectral density investigations were performed with Mathworks Matlab signal processing toolbox using Welch’s method inclusive of Chebyshev window. All the preprocessing and processing of the interest rate and the PMI data sets were done using a combination of Matlab version 7.12.0.635 (R2011a), Microsoft Excel 2010, and NeuroDimension NeuroSolutions version 6.06. For example, Excel and NeuroSolutions were for the neural network modeling and forecasts while Excel and Matlab for the multiple regression modeling and forecasts.

The interest rate compression and the PMI denoising were done using similar Matlab wavelet toolbox main menu setup. The one dimensional wavelet packet with the discrete approximation of Meyer wavelet (\texttt{dmey}) and the Shannon entropy criterion was used. For the interest rate compression, the decomposition level was three (3), the threshold method was balanced sparsity norm, and the global threshold was 4.07 while for the PMI denoising the decomposition level was four (4), the threshold method was the soft fixed form threshold with unscaled white noise structure, and the global threshold was 3.66. Only 12.58% of the compressed interest rate wavelet packet coefficients were required to recover 99.51% of the total energy into the compressed interest rate. Thereafter, the 12 month backward first differencing and mean removal techniques were applied to the compressed interest rate and the denoised PMI, thereby reducing the interest rate and PMI changes from 552 to 540 monthly samples.

For both levels and differences, interest rate preceded PMI by 11 months at maximum values of 0.391 and 0.601 respectively. Since the Hurst exponent value of 0.572 exceeded 0.5 for the PMI changes, the PMI changes are predictable. While the skewness with kurtosis and the Jacque-Bera test values for interest rate and PMI levels showed that they were not representing samples from normal distributions at the 1% significance level, the skewness with kurtosis and the Jacque-Bera test values for interest rate and PMI changes appeared to be near normal from the near zero skewness values of -0.027 and 0.069 and near 3.0 kurtosis values of 3.651 and 3.748 respectively, and the Jacque-Bera test being statistically significant at the 5% level. Since the absolute magnitude of PMI changes was about five times that of interest rate changes, interest rate changes were scaled up by a factor of 5 to conveniently align interest rate with PMI values to facilitate easy visual comparison.

Because of the 11 months lead of compressed interest rate changes over denoised PMI changes at their maximum correlation of 0.601, the forecast horizon for PMI was set to 12 months, where compressed interest rate changes and past values of denoised PMI changes were the predictor variables and denoised PMI was the response variable predicted to be 12 months in the future. By aligning the predictor and response variables while compensating for the forecast horizon, these variables were reduced from 540 samples to 528 samples with the response variable covering the period of August 1966 to July 2010 and the predicted response covering the out-of-sample forecast period of May 1997 to July 2010 in increments of 45, 50, 39, and 36 monthly samples for the testing subsets, where the overlapped two samples of the ending of the first and the beginning of the second increments, six samples of the ending of the second and the beginning of the third increments, and three samples of the ending of the third and the beginning of the fourth increments were averaged. The initial training subset covered the period of August 1965 to April 1996; subsequent training subsets were then increased in increments of 43, 44, and 36 monthly samples respectively making the fourth training subset 492 monthly samples. Two types of models were used in the prediction process: focused gamma time-lagged recurrent neural network and robust multiple regression (Principe et al, 2000; Haykin, 1999; DuMouchel & O’Brien, 1989). The general model describing the focused gamma neural network is the following:

\[
y(t+h) = \phi_0 \left( \sum_{j=1}^{M} \sum_{k=0}^{R} \sum_{l=1}^{D} \sum_{k=1}^{R} \sum_{l=1}^{D} \left( w_{jk}(0)x_k(t) + w_{jk}(l)((1-\mu)x_{kl}(t-1) + \mu x_{kl}(t-1)) + b_j \right) + b_0 \right)
\]

where M, R, D, b, h, and \(\mu\) are the number of processing elements in the hidden layer, number of inputs, number of taps, bias, forecast horizon, and feedback parameter, respectively. The activation functions for the hidden and output layers are \(\phi_1\) and \(\phi_0\) respectively. The focused gamma neural network models had two inputs and two input weights; input memory with four taps and tap delays of 1, 1, 1, and 2 respectively as well as a depth of six and trajectory lengths of 41, 103, 57, and 82 respectively; one hidden layer with three processing elements and 27 weights and biases; and one output layer with one processing element and four weights and biases. The hidden and output layers
used the \( \tanh \) activation function. The models were trained in supervised batch learning mode with the help of the Levenberg-Marquardt backpropagation through time algorithm. The general form of the model describing the robust regression is the following:

\[
y(t+h) = a_1 x_1(t) + a_2 x_2(t), \quad t \geq 1
\]

where the set of coefficients for \( (a_1, a_2) \) corresponding to \( x_1 \) for interest rate and \( x_2 \) for PMI over the four training intervals are \( \{(0.694, -0.402); (0.702, -0.409); (0.674, -0.414); (0.648, -0.425)\} \). The coefficients were computed using the iteratively reweighted least squares algorithm with the logistic weighting function until their estimates converged within a specified limit.

Results and Discussion

In the experiments of this study, both standard and relative performance measures were used to judge the relative efficacy of the focused gamma neural network and the robust regression models forecasts and the overall forecasts. For the overall forecasts two standard (RMSE and MAD) and three relative (\( r, \) NRMSE, Theil) performance measures were used. The performance measures and the visual examination of the forecasted PMI show that the neural network models performed better (see Table 1 and Fig. 1). Of the two sets of four PMI forecasts that make up the two overall PMI forecasts of Fig. 1, the set of four relating to the neural network models show relatively lower RMSE values on three of the four regimens (2-4) and higher correlations for two of the regimens (2 & 4). The differences in the RMSE values of neural network and regression models are 0.782, 0.653, and 3.742 in favor of the neural network models. For regimen 1, the difference in the RMSE values of 0.055 between the two types of models favors the regression model. This suggests that for the regimen 1 forecast, the regression model performed only marginally better than the neural network model, while in the other three regimens, the neural network models’ overall performance was relatively much better. A similar finding is evident in the correlation results. In regimens 2 and 4, the correlations between the PMI forecasts of the neural network models and the actual PMI exceeded those for the regression models by 0.224 and 0.386 respectively, whereas in regimen 1 and 3, the correlations relating to the neural networks models are less than the regression models by 0.007 and 0.044 respectively. The overall forecast generated by the neural network models show better performance in terms of relatively higher correlation and lower RMSE, NRMSE, and MAD (see Table 1): the related positive differences between the two forecasts in favor of neural network are 0.258, 1.763, 0.223, and 1.462.

Visual assessment of the neural network and regression forecasts confirmed the findings of most of the performance measures. While both forecasts mostly follow the general patterns of the actual PMI data, the neural network generated forecast captured more of the details as well as alignment with the PMI data. However, neither the neural network nor the regression models were able to capture the abrupt fall in PMI of -10.47 units over the six month period of July 2008 to December 2008. PMI bottomed out in December 2008, and thereafter increased over the next several months until it peaked in March 2010. The overall downturn in PMI actually started in December 2007 (see Fig. 1, month 128), the same month that the recent 2007/2009 recession started (Joseph et al, 2010). Whereas PMI started to show evidence of an upturn in the economy in January 2009, the neural network forecast of PMI showed the upturn starting much earlier, June 2008; this upturn evident in the neural network forecast reached
its peak value at the same time as the actual PMI. The regression forecast started its general upturn much earlier than the neural network forecast, March 2007, and continued until it peaked in November 2009, four months earlier than the peaks of the neural network forecast and the actual PMI. Therefore, the regression forecast did not capture any part of the recent recession evident in PMI between December 2007 and December 2008. The neural network forecast captured some of it between January 2008 and June 2008, but not much.

Figure 1: The overall out-of-sample forecasts of the purchasing managers’ index with focused gamma neural network and robust regression models.

Why did the forecasts generally miss the downturn in PMI that corresponded to the first half of the recent recession? Part of the answer might lie in the relationship between 3-month T-bills interest rate and PMI over the period of December 2005 and December 2008. The countercyclical behavior of interest rate on the economy suggests that generally low interest rates stimulate spending in the economy while high interest rates curtail it. While the maximum correlation between inverted 3-month T-bill interest rate and PMI occurred when interest rate is leading by 11 months, some of the differences between the dominant lead/lag peaks as well as troughs of these two economic variables are greater or less than 11 months. In December 2005, interest rate peaked (inverted interest rate troughed) and PMI experienced a dominant trough in December 2008; this is 21 months after interest rate peaked. Thus, a 12 month forecast horizon may have been inadequate to predict the first half of the recession, since interest rate was able to trough while PMI was declining between July 2008 and December 2008.

Using inverted T-bills interest rate as the predictor, Larrain (2007) forecasted PMI with the focused gamma neural network and regression models. His neural network models were more complex: 24 processing elements compared to three in the experiments of this work. Over the period (September 2002 to August 2005) of Larrain’s out-of-sample forecasts, the RMSE statistics for the neural network forecast in this work is greater than that obtained by Larrain by 2.31 and a correlation of 0.803 is obtained. Over the three year period of the neural network forecast, the two results are generally comparable, but the forecast here shows better alignment and trending with the actual PMI.

Conclusion

The neural network forecasts were generally better than the regression forecasts. However, neither the neural network nor the regression forecasts adequately captured the downturn in PMI that was evidence of the recent 2007/2009 economic recession. The use of more adaptive forecasting horizons reflective of the complex relationship between interest rate and PMI would probably be helpful since these variables were found to be generally not linear. Future work will investigate this premise. Furthermore, results obtained in this work for September 2002 to August 2005 are comparable to those of Larrain (2007) in showing that interest rate is a viable predictor of PMI.
Reference


Security in Computer Literacy-
A Model for Design, Dissemination, and Assessment

Claude F. Turner  
Department of Computer Science  
Bowie State University  
(301) 860-3965  
cturner@bowiestate.edu

Blair Taylor  
Department of Computer and Information Sciences  
Towson University  
(410) 704-4560  
btaylor@towson.edu

Siddharth Kaza  
Department of Computer and Information Sciences  
Towson University  
(410) 704-6310  
skaza@towson.edu

ABSTRACT
While many colleges offer specialized security courses and tracks for students in computing majors, there are few offerings in information security for the non-computing majors. Information security is becoming increasingly critical in many fields, yet most computer literacy courses insufficiently address the security challenges faced by our graduates. This paper discusses the development and impact of a set of modules designed to integrate security into computer literacy across two universities and several community colleges in the state of Maryland. Results from our comparative analyses based on pre- and post-test analysis show significant improvements in post-test results.

Categories and Subject Descriptors  
K3.2 [Computers and Education]: Computer and Information Science Education - computer science education, curriculum, information systems education.

General Terms: Security

Keywords: Security Education, Computer Science Curriculum, Information Security Curriculum Development.

1. INTRODUCTION
Computer literacy is offered as a required course by a variety of universities and colleges, where it aims to provide students with current knowledge and understanding of computers and their uses. For many college instructors, the computer literacy course is the bane of their teaching load – often the first course they teach and the first course they drop from their schedule. The disadvantages of teaching this course include: a new textbook every few years; lecture notes that quickly become obsolete; a variable, often ill-defined list of topics [4]; and large classes full of non-majors looking for an easy ‘A’. Additionally, the class seems to afford few research or publishing opportunities; since 2000, less than a dozen papers have been written on computer literacy [4].

Epperson [4] provides an excellent history of computer literacy, starting with the coinage of the term in 1972, as well as its future direction. Shelly [11] informs us that the requirements that determine computer literacy evolve as technology changes. In recent years, to help people adapt to these rapid changes, become effective users quickly, and prepare for lifelong learning, there has begun a reexamination of the general structure of computer literacy. A National Research Council (NRC) report titled, Being Fluent with Information Technology [10], has proposed a variety of recommendations, including a new term that would replace computer literacy—fluency with information technology (FIT)—or as put more succinctly by Snyder, fluency [13]. FIT is described as "a package of skills, concepts, and capabilities wrapped into a project-oriented learning approach that ensures that the content is fully integrated" [13]. Other authors including Bartholomew [2], Hoffman & Blake [6] and Sloan & Halaris [12] have discussed this subject and have provided their recommendations on providing a breadth of IT knowledge to non-computing majors.

We believe that in addition to other important topics, non-computing majors need to be exposed to security topics. Information security has become increasingly critical to many other fields, business and health for example, and college graduates from all majors will encounter information security issues in their personal and professional lives. Even so, many books on literacy and fluency often have only a chapter or section on security, and lack the active learning component that would help students internalize the concepts.

To address this issue, we added the computer literacy course to the Security Injections project [8,14,18,19] currently underway at Towson University, Bowie State University, and several other community colleges in Maryland. The objective of the project is to integrate security throughout computing curricula using a minimally invasive thread based approach [14,18,19]. The project began with a focus on the core computer science classes, but the addition of the computer literacy course had many unforeseen advantages. With apologies to Dr. Seuss, "Maybe the Computer Literacy class...perhaps...means a little bit more!" Consider the following, which holds true at many institutions, including Towson and Bowie State University:

- The computer literacy class reaches more students than any other class in our department.
- The computer literacy class reaches students in computing majors, non-computing majors and students who have not yet decided on a major.
• The computer literacy class includes more female and minority students than any other class in our department.
• The computer literacy class allows some flexibility to add interesting and relevant topics.

This work reports on our efforts to integrate security into the computer literacy [1,3,7] course using modules that encourage active learning. We have developed, deployed, and assessed a set of three modules (called security injections) for computer literacy based on the general structure proposed by Taylor et al. [18]. The modules cover the topics of phishing, passwords and cryptography and are designed to provide students with the experiential learning in computer security. In this paper, we present and discuss the collaborative process involving two and four year institutions to design, disseminate, and assess the modules. We describe the general structure of the modules and provide a specific example. We also report on our experience in deploying modules and our assessment results.

2. PROJECT OVERVIEW

2.1 Security across the Curriculum
In fall 2006, we initiated a plan to integrate security throughout the computing curriculum. The goal was to complement our security track and introduce security principles to computing majors early and often. Early efforts targeted select sections of the core computing required courses, CS0 and CS1 [18]. Using a common set of learning objectives to link the materials across the classes, we created a set of security related laboratory modules that focused on the major secure coding issues of integer overflow, buffer overflow, and input validation. In 2008, the project expanded to include five additional courses: CS2, Computer Literacy, Database, Web Programming, and Networking; and to encompass five institutions: Towson, Bowie State, and three partnering community colleges: Anne Arundel Community College, the Community Colleges of Baltimore County, and Harford Community College.

2.2 The Model
Our collaborative approach to implementing curriculum change across varying institutions is based on a model with a set of well-defined goals and repeats a process of: develop materials, pilot across institutions, evaluate, revise, and disseminate [15]. Formal evaluation includes materials review from a technical expert in security, qualitative and quantitative feedback from experienced teachers at our partnering institutions, results from ongoing pilot programs, and quantitative results from security surveys and code checks. This model and assessment results for CS0, CS1, CS2 have been reported in previous publications [8,15,16].

2.3 Security Modules
Security modules were created with the objectives of maximizing the learning experience and minimizing the burden on the instructor. To increase the effectiveness of the module, we designed it with active learning in mind [9]. The “active” components include security related exercises and a security checklist, described below. Additionally, we formalized the module structure to promote critical thinking and reflection [17]. In contrast to the traditional computer laboratory exercise which is a loosely structured set of problems or exercises that inadequately address synthetic and analytical thinking [17, 18], the security lab modules in this project are patterned after the structured labs in physics and biology. Each module contains the following components:

• **Background** - The module begins with background information, including a concise description of the topic, the risk involved, and real-life examples that include links to articles describing actual occurrences of security vulnerabilities. The purpose of this section is to set the stage for the active learning process that follows.

• **Laboratory Assignments** – There are several hands-on lab exercises related to the specific security concept. Hands-on exercises that present meaningful concepts in an engaging manner increase motivation and enhance learning. Research shows that in a “learning by doing” environment, students learn more, retain the information longer, can apply learned material in more contexts, and the environment of the classroom is more enjoyable [9].

• **Security Checklists**. A key component to each module is the security checklist. A security checklist is a well-defined set of procedures for identifying potential security concerns. Checklists have many relevant applications, including aviation, software assurance, and have recently been shown to reduce errors in emergency rooms [5]. A well-developed checklist can reduce human error, serve as a reminder list, and help ensure consistency and completeness. Additionally, checklists can reinforce security principles and help students quantify and internalize important security principles. The checklists in our modules, initially designed to find potential vulnerabilities in software, proved to be readily adaptable for passwords, phishing, and cryptography.

• **Discussion questions** require students to reflect upon the process, the results, and the security implications of the new concept.

Minimally invasive. Finally, each module was designed towards ease of use by the instructor. Using feedback from faculty and students, each module has been streamlined and formatted to allow simple insertion within a lab or for stand-alone use. Estimated completion time for most modules is 20-30 minutes. We encourage instructors to use or adapt the modules to accommodate instructional needs. We also recommend that instructors allow students to work in pairs to encourage collaborative learning. The intention of each lab is that it can be completed without class instruction and with minimal help from the instructor.

3. Computer Literacy Modules

3.1 Module Design
Developing security modules for computer literacy is a challenging activity. Many of the students are freshmen and often lack the necessary prerequisites to understand computer security principles. In our case, this is further compounded by the fact that our modules are targeted to different communities of students, ranging from community colleges, minority serving institutions, to comprehensive universities. To address these
challenges, we designed the modules to be self-contained and targeted to a freshman audience. We also hold workshops at participating institutions to help train and prepare faculty who teach using the modules.

The topics for the modules were determined through discussions in workshops attended by faculty from all partner institutions. We found the computer literacy instructors, some of whom are graduate students and adjunct instructors, to be very receptive to our approach and full of ideas for future modules. We sought to identify essential issues in information security that were topical and cut across various communities of students. Although a variety of important topics were considered, phishing, passwords and cryptography emerged at the top of the list. A module that encompassed the basic definition of security—illustrating the challenge and significance of meeting the triangular concepts of confidentiality, integrity and availability (CIA), as well as the concept of risk analysis—was also deemed very important.

3.2 Using the Security Injection Modules in your Classroom

Currently, our project includes the following three modules for Computer Literacy: (1) Passwords, (2) Phishing, and (3) Cryptography. Modules are located at: http://triton.towson.edu/~cssecinj. A compressed form of the Phishing module is shown in Table 1. These modules are currently being used at Bowie State University, Towson University, Anne Arundel Community College, Community College of Baltimore County, and Harford Community College. The procedure for using the modules, described in the ‘Faculty Access’ section of the website is as follows:

1. Administer Security Survey
2. Introduce Security Injections in Class
3. Administer Security Survey after introducing Security Injections
4. Complete Faculty Survey.

4. ASSESSMENT DESIGN AND RESULTS

The primary goals for this project are: increasing students’ general security awareness, improving students’ knowledge on the content of specific security modules, increasing faculty security awareness, and increasing the number of security-skilled students. Each of these objectives is assessed using various instruments including surveys, qualitative inputs from faculty, and institutional data. In this paper, we present the student survey design and results on student awareness and learning from the use of modules in the computer literacy courses.

4.1 Pre and Post Surveys

Each class that used our modules was given a pre-survey at the beginning of the class and a post-survey at the end of the class. An independent evaluator has reviewed all assessment materials, including pre-surveys and post-surveys. The student surveys contain demographic questions (including questions on the interest in security), and two sets of multiple choice questions—one section targets general security awareness and the other focuses on specific knowledge gained through the modules (a list of sample questions is shown in Table 2). We used parts of this survey in CS0, CS1, and CS2 previously and it was well accepted by students [8].

Table 2. Sample survey questions

<table>
<thead>
<tr>
<th>General Security Awareness</th>
</tr>
</thead>
<tbody>
<tr>
<td>What are the possible consequences of insufficient computer security?</td>
</tr>
<tr>
<td>A set of related programs, usually located at a network gateway server, that protects the resources of a private network from other networks, is known as a …</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Module specific</th>
</tr>
</thead>
<tbody>
<tr>
<td>Who is it safe to tell your password to?</td>
</tr>
<tr>
<td>Phishing is …</td>
</tr>
<tr>
<td>The following are characteristics of suspicious email:</td>
</tr>
<tr>
<td>Encryption is a special technique employed only by agencies with highly sensitive data such as the FBI or CIA.</td>
</tr>
<tr>
<td>Consider the following email: (followed by an phishing email that the students had to identify)</td>
</tr>
<tr>
<td>Using letters from a memorable phrase is a recommended way to construct a password. (T or F)</td>
</tr>
<tr>
<td>Never give out personal information upon an email request (T or F)</td>
</tr>
<tr>
<td>Encrypting your personal files requires purchasing special software. (T or F)</td>
</tr>
<tr>
<td>The conversion of data into a ciphertext that cannot be easily understood by unauthorized people is known as …</td>
</tr>
</tbody>
</table>

Based on the pre and post survey, a set of two hypotheses were proposed to test student learning.

H1: The post-survey scores will be significantly higher than pre-survey scores.

H2: The post-survey scores in specific module topics will be significantly higher than pre-survey scores.

4.2 Basic Statistics and Demographics

A total of 384 survey responses from four institutions were received with 357 responses from sections that used the modules. After data cleaning, 300 valid survey responses were analyzed. Figure 1 presents the demographics of students who took the survey.

As can be seen, the classes had a fairly even distribution between gender and ethnicity. The classes also had a majority of freshman as was expected in a literacy course. Also as expected, the classes showed a skewed distribution among majors. A large majority of students were non-computing majors – this provides us with an opportunity to spread security concepts among students in other disciplines who are not exposed to security issues that they are almost certain to face in their professional and personal lives. In addition, the even spread in gender and ethnicity offers an excellent platform to introduce exciting computer science concepts and attract women and URMs to the field.
Table 1. An abbreviated version of the Phishing module (see complete module at http://triton.towson.edu/~cssecinj)

**Phishing – “A scam to steal private information”**

**Summary** - Phishing is a type of social engineering technique in which an attacker sends an e-mail or displays a Web announcement that falsely claims to be from a legitimate organization. The intention of the messenger is to trick the user into surrendering private information.

**Description** - A more specific definition is offered by the Anti-phishing Working Group (APWG): “Phishing is a criminal mechanism employing both social engineering and technical subterfuge to steal consumers’ personal identity data and financial account credentials.” The victim in a phishing attack is asked to respond to an e-mail or is directed to a Web site to update personal information, such as passwords, credit card numbers, Social Security numbers, bank account numbers, or other information for which the legitimate organization already has a record. However, the site is actually a fraudulent Web site designed to steal the user’s information.

**Risk** - Phishing can and usually leads to online identity theft. By capturing a user’s personal information, an attacker can gain access to the user’s account on a legitimate Web site, and can engage in a number of activities resulting in substantial financial loss to the user, denial of access to e-mail, and other problems.

**Example of occurrence** - On the weekend of January 3, 2009, several users on the social network Web site, Twitter, became victims of a phishing attack. The users were deceived into giving away their passwords when they received an e-mail similar to one that they would receive from Twitter with a link that read, “hey, check out this funny blog about you…”. The link redirects to a site masquerading as the real Twitter site. Any personal information entered by the user on the fake site is then captured by the attacker.

**Anti-Phishing Training** - Training users to identify a ‘phish’ is an important component in the fight against phishing. Training has taken two forms: the first is simply to provide anti-phishing information to users through e-mail and other media. The second is to give firsthand experience to users through games, simulated phish, cartoons, etc. Recent studies seem to indicate that the latter—giving firsthand experience to users—might be more effective. The game, Anti-Phishing Phil http://wombatsecurity.com/antiphishingphil, which teaches people how to identify suspicious Web site addresses while providing the experience of being captured by a phisher, is such an example. PhishGuru in the previous section is another example. It delivers cartoon-based, anti-phishing information after a user has been deceived by simulated phishing messages.

**Anti-Phishing Technologies** - Although user ability to identify phish is an important component in the battle against phishing, combining it with technology yields better results. One of the techniques used to automatically identify phish is **filtering**. The objective of filtering is to identify (or flag) phishing attempts in e-mail or on Web pages. Filters are usually integrated into browsers or e-mail software. When a Web address is encountered the software compares it with a so-called “blacklist” of known phishing Web sites. It then takes appropriate actions, which usually include informing the user. The blacklist is updated periodically (for example, every 30 minutes) as new phishing sites become available. As with any blacklist, there is also a “whitelist” of known legitimate sites.

**Lab questions** - Consider the PayPal e-mail in the figure (an e-mail is provided to the students, please see website for the complete module). The Web address in the box, http://211.248.156/Paypal/cgi-bin/webscrecmd_login.php, appears when the user mouse-over the “Click here to verify your account” link.

1. Complete the following checklist for this e-mail.

   ![Security Checklist](image)

2. List any sentence, phrase or word that makes the e-mail a suspected phish.

**Discussion questions**

1. Play at least two games of Anti-Phishing Phil at http://wombatsecurity.com/antiphishingphil. Create a “blacklist” of the phishing Web site addresses you encountered, and a “whitelist” of the legitimate Web sites. (Hint: see the section on Anti-phishing Technologies.) Describe how the Anti-Phishing Phil experience has helped you to better recognize phishing Web sites. What are your likes and dislikes about the game? Are there any suggestion(s) that you would like to provide so as to improve it? If so, explain.

2. Take the “SonicWall Phishing and Spam IQ Test” a couple of times (http://www.sonicwall.com/phishing/). What was your maximum score? Look at the test result sheet, and give the name that appears in the “Subject” column for three of the questions. For each of the subjects, click on the “Why?” link that appears under the “Explain Answer Column.” The e-mail you viewed for that question should re-appear—this time with explanations. Copy one of the given explanations for each of the e-mails.
4.3 Survey Results

This assessment involved class sections in five institutions and we found that it was difficult to get a similar response rate for both the pre-surveys and the post-surveys. There were 300 survey responses with 86 in the pre-survey and 214 in the post-survey. In line with our Institutional Review Board requirements, the survey did not contain any information that could identify the student or pair the responses between the pre and post tests. Each question in the survey was a multiple choice question with three or more options (the T/F questions had an unsure option) and one correct option. The score of each survey was calculated by counting the number of correct answers from a total of 14 questions (including general security and module specific questions). We picked the Mann-Whitney non-parametric test to compare the mean rank of the scores in the two groups (pre and post). This was done because of two primary reasons – 1) the \( n \) for the groups is different and 2) the Kolmogorov-Smirnov and Shapiro-Wilk test showed that the scores were not normally distributed. Figure 2 summarizes the results. The average score obtained in the pre-surveys was 9.09 and the average score in the post-surveys was 10.67. It was found that this was a statistically significant increase \( (p < 0.001) \). These results look promising and support H1.

For individual modules, it was found that there was a statistically significant increase \( (p < 0.001) \) in the scores obtained in the questions targeted at each of the modules with a 34.71% increase in knowledge related to the phishing module, 25.82% increase in the cryptography module, and 9.14% increase in the password module. This supports H2.

Deeper analysis on the data showed that the significant increase in overall scores persisted across gender and ethnicity. Additionally, we found that women started significantly lower than men, yet, caught up in the post-test; and at the end of the semester there was no significant difference in their scores. Further analysis of this data is in progress.

5. Conclusion

Many colleges offer specialized security courses and tracks for students in computing majors since computer security is recognized as an essential topic for computer scientists. For the non-computing majors, there are few offerings in information security. Increasingly, information security is becoming critical in many fields and the current computer literacy courses insufficiently address the security challenges faced by our
graduates. This paper discussed the development and impact of a set of modules designed to integrate security into computer literacy across two universities and several community colleges in the state of Maryland. Results from our comparative analyses based on pre and post test results showed statistically significant improvements in the post-tests. Overall scores improved 17.37%; phishing module related scores increased 34.71%; cryptography related scores increased 25.82%, and password related scores increased 9.14%. Also of interest, females, who scored lower on the pre-tests, caught up with males on the post-tests.

6. ACKNOWLEDGMENTS
This project is partially supported by the NSF Course Curriculum and Laboratory Improvement (CLLI) grant number DUE-0817267. We greatly appreciate our colleagues and partners at Bowie State University, Harford County Community College, Community College of Baltimore County, and Anne Arundel Community College. In particular, we are grateful to Harry Hochheiser, Mike O’Leary, Shiva Azadegan, AC Chaplin, Patricia Gregory, Jack McLaughlin for their contributions.

7. REFERENCES