PROCEEDINGS OF THE SINGAPORE CYBER-SECURITY CONFERENCE (SG-CRC) 2016

Cryptology and Information Security Series

The Cryptology & Information Security Series (CISS) presents the latest research results in the theory and practice, analysis and design, implementation, application and experience of cryptology and information security techniques. It covers all aspects of cryptology and information security for an audience of information security researchers with specialized technical backgrounds.

Volume 14

ISSN 1871-6431 (print)
ISSN 1879-8101 (online)

Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016

Cyber-Security by Design

Edited by

Aditya Mathur

Information Systems Technology and Design, Singapore University of Technology and Design, Singapore

and

Abhik Roychoudhury

School of Computing, National University of Singapore, Singapore

IOS
Press

Amsterdam • Berlin • Washington, DC

ISBN 978-1-61499-616-3 (print)
ISBN 978-1-61499-617-0 (online)
Library of Congress Control Number: 2015960167

Publisher
IOS Press BV
Nieuwe Hemweg 6B
1013 BG Amsterdam
Netherlands
fax: +31 20 687 0019
e-mail: order@iospress.nl

Distributor in the USA and Canada
IOS Press, Inc.
4502 Rachael Manor Drive
Fairfax, VA 22032
USA
fax: +1 703 323 3668
e-mail: iosbooks@iospress.com

Preface

Welcome to the First Singapore-Cybersecurity R&D Conference (SG-CRC 2016)! This inaugural conference brings together researchers from across the globe engaged in advancing the state of the art in the broad area of cyber security. The conference is sponsored by the National Research Foundation (NRF), Singapore, and organised jointly by the Singapore University of Technology and Design and the National University of Singapore.

A total of 22 submissions were received in response to the call for papers. Each submission was reviewed by at least two members of an international programme committee. Based on the reviews, six papers were selected for presentation under the "Regular" paper category and seven in the short paper category. In addition to research paper presentations, leading researchers in cybersecurity have been invited to deliver a keynote address and make special presentations. The conference programme also includes industry tools presentations, panel discussions, and a poster session.

The paper titled "Image Region Forgery Detection: A Deep Learning Approach" focuses on the detection of tempered images. The technique proposed is independent of image format, e.g., JPEG. The proposed technique uses a two-stage deep learning approach to learn complex features of the image, in a variety of formats format. For JPEG images 87.51% tampered region localisation accuracy was obtained while for TIFF images the localisation rate was 81.91%.

Android malware is the focus of "Q-Floid: Android Malware detection with Quantitative Data Flow Graphs." This paper moves beyond the conventional signature-based malware detection techniques by using a more quantitative data-flow based technique applied to system entities such as processes, files, and sockets. The proposed approach obtained a malware detection rate of 93% for variants of known malware and up to 84% for new malware families.

"Cyber and Physical Access Control in Legacy System Using Passwords" addresses the issue of managing a large number of passwords by humans. A visual cryptography technique is proposed that allows the storage of cipher texts of passwords on mobile phones and decrypt them on demand. The proposed approach allows a simple though effective solution to the problem of access control using passwords.

"Data Driven Physical Modeling For Intrusion Detection In Cyber Physical Systems (CPS)" proposes a machine learning-based technique for intrusion detection. The proposed technique quickly detects attacks at the physical process layer. The technique was applied on a replicated version of a modern water treatment facility and found to be fast, scalable, robust to noise, and exhibiting a low false positive (FP) rate with high precision and recall.

A novel approach for detecting multi-point attacks in a CPS is proposed in "Detecting Multi-Point Attacks in a Water Treatment System Using Intermittent Control Actions." The technique makes use of control actions that are not part of the standard control algorithms implemented in the controllers. Instead, these control actions are executed on selected components to monitor the system process. The strengths and limitations of the approach were assessed experimentally in an operational water treatment plant.

Challenges in setting up Man-In-The-Middle attacks are the focus of "Attacking Fieldbus Communications in ICS: Applications to the SWaT Testbed." Attacks were successfully launched on fieldbus communications in a working water treatment plant. In such attacks, the attacker manipulates or replaces sensor data as reported from the field devices to the control components. The efficacy of the proposed framework for launching attacks is demonstrated experimentally where an adversary can intelligently design and launch attacks that remain undetected for a typical bad-data detection mechanism.

Seven short papers focus on the following areas: Directed-Tree-Transitive Signature scheme, large scale collection of information based on Juice-filming attacks, social-engineering attacks based on telephones, privacy and data aggregation, simulation of cyber attacks, steganography in the context of ECG data, and file classification. A student paper explores the idea of identifying hardware via sensor fingerprinting in a CPS. An experiment to evaluate the proposed method for hardware tampering revealed high detection rate when applied to two water level sensors in an operational CPS.

The conference also contains updates from seven projects funded by NRF under the National Cyber-security Research (NCR) program. These projects cover various themes such as software security, cyber-physical system security, mobile security and formal verification.

The success of this conference is due to the participation and contribution of a large number of people. First, we thank the many researchers who spent time in writing and submitting to this inaugural SG-CRC. Thanks to members of the Steering and Programme Committees for assisting and advising on the details of conference planning and completing on time the important task of paper reviews. Thanks to members of the organising committee who exhibited total dedication and commitment to make a successful conference. Last, but not least, our sincere thanks and appreciation to NRF and the staff who originated the idea of SG-CRC and provided constant support at all stages of organising the conference.

With best wishes for a successful conference, yours sincerely,

Aditya Mathur, Conference Co-Chair
Professor, Head of Pillar ISTD, and Centre Director iTrust
Singapore University of Technology and Design
Singapore

Abhik Roychoudhury, Co-Chair
Professor and Vice Dean, School of Computing
National University of Singapore
Singapore

Committees

Steering Committee Members

Shu Han CHEONG	Ministry of Home Affairs
Boon Kwee CHUA	Ministry of Defence
Li Ting HO	Singapore Economic Development Board
Martin KHOO	Infocomm Development Authority of Singapore
Hwee Kwang LIM (Co-chair)	National Research Foundation
Aditya MATHUR (Co-chair)	Singapore University of Technology and Design
Su Yen SHU	National Research Foundation
Chuan Yeong TAN	National Security Coordination Secretariat

Programme Committee Members

Hrishikesh ACHARYA	Indian Institute of Technology Delhi
Gail-Joon AHN	Arizona State University
Lionel BRIAND	University of Luxembourg
Alvaro CARDENAS	The University of Texas at Dallas
Mauro CONTI	University of Padua
Robert DENG	Singapore Management University
Wenliang DU	Syracuse University
Yuval ELOVICI	Ben-Gurion University of the Negev
Sandro ETALLE	Technical University of Eindhoven
Sonia FAHMY	Purdue University
Debin GAO	Singapore Management University
Stefano GALELLI	Singapore University of Technology and Design
Ananth GRAMA	Purdue University
Pieter HARTEL	University of Twente
Ralf HUUCK	NICTA
Inseok HWANG	Purdue University
Florian KERSCHBAUM	SAP
Marina KROTOFIL	Hamburg University of Technology
Xenofon KOUTSOUKOS	Vanderbilt University
Andrea LANZI	University of Milan
Emil LUPU	Imperial College
Kheng Kok MAR	Nanyang Polytechnic
Ammar MASOOD	Air University
Aditya MATHUR (Chair)	Singapore University of Technology and Design
Sjouke MAUW	University of Luxembourg
Yilin MO	Nanyang Technological University
Mattia MONGA	University of Milan
Martín OCHOA	Singapore University of Technology and Design
Bert Jan Te PASKE	TNO
Alexander PRETSCHNER	Technische Universität München
Awais RASHID	Lancaster University

Contents

Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016
A. Mathur and A. Roychoudhury (Eds.)

doi:10.3233/978-1-61499-617-0-1

Image Region Forgery Detection: A Deep Learning Approach

Ying Zhang [a,1], Jonathan Goh [a], Lei Lei Win [a] and Vrizlynn Thing [a]
[a] *Cyber Security & Intelligence Department, Institute for Infocomm Research, Singapore*

Abstract. In digital forensics, the detection of the presence of tampered images is of significant importance. The problem with the existing literature is that majority of them identify certain features in images tampered by a specific tampering method (such as copy-move, splicing, etc). This means that the method does not work reliably across various tampering methods. In addition, in terms of tampered region localization, most of the work targets only JPEG images due to the exploitation of double compression artifacts left during the re-compression of the manipulated image. However, in reality, digital forensics tools should not be specific to any image format and should also be able to localize the region of the image that was modified.

In this paper, we propose a two stage deep learning approach to learn features in order to detect tampered images in different image formats. For the first stage, we utilize a Stacked Autoencoder model to learn the complex feature for each individual patch. For the second stage, we integrate the contextual information of each patch so that the detection can be conducted more accurately. In our experiments, we were able to obtain an overall tampered region localization accuracy of 91.09% over both JPEG and TIFF images from CASIA dataset, with a fall-out of 4.31% and a precision of 57.67% respectively. The accuracy over the JPEG tampered images is 87.51%, which outperforms the 40.84% and 79.72% obtained from two state of the art tampering detection approaches.

Keywords. Image forgery detection, region localization, deep learning, feature learning

1. Introduction

In the digital era, there are an enormous volume of forged images on social media platforms such as Facebook or Flickr. The distribution of manipulated images can be shared very easily and can be used to mislead viewers from the truth. This may result in very serious consequences so the authenticity of digital images is urgently needed.

While there are a few solutions to automate image tampering detection, some of these methods are specific only to the JPEG file format [5,14,6,4,24,20,23] where they detect the tampered region based on artifacts left by multiple JPEG compressions. Other solutions [1,12,7] also identify features based on a specific tampering method such as

[1]Corresponding Author: Ying Zhang, Cyber Security & Intelligence Department, Institute for Infocomm Research, Singapore; E-mail: zhangy@i2r.a-star.edu.sg

copy-move where objects in the image are copied and pasted to hide or insert object. In these works, the duplicated parts of the image are discovered by invariant features.

There are a few techniques to automate image tampering detection. [10,19,11] determines the authenticity of the image where it is identified either as authentic or tampered [10]. However, these techniques do not identify the tampered region. Hence, a more sophisticated approach to obtain the tampered regions is required as this removes the need to manually identify suspicious regions. Currently, there are a few techniques to identify tampered regions. [5,14,6,4,24,20,23] exploits the artefacts left by multiple JPEG compressions to detect the tampered regions. However, these techniques are applicable only to the JPEG formats. Camera based methods [2,17] have also been explored where the detection is based on demosaicing regularity or sensor pattern noise where the irregularities of the sensor patterns are extracted and compared for anomalies. However, these methods are constricted to specific assumptions. For example, [16] works on the assumption that the image comes from a camera with the presence of Color Filter Array while [9] assumes that there is a presence of sensor patterns pre-obtained from specific camera models. Other methods to detect tampered regions also includes local descriptors. These methods [1,12,7] identify features that identify similar objects in the image which were copied and pasted to hide or insert object. These works are not applicable to tampering techniques such as splicing where objects are copied from one image and inserted into another.

In this paper, we address the above problems by proposing a two stage hierarchy feature learning approach for image tampered region detection. Deep learning provides a novel approach to the identification of features for tampered regions, which inherently represent characteristics of the tampered regions appearing in the dataset [18,21]. Such learning is data-driven and can be applied to images from any category of tampering. It greatly saves us time and energy to find new features from a set of images. In this work, we show that a deep learning Stacked Autoencoder (SAE) model can be used for feature learning for characterizing tampered regions. At the first stage, our experiments have shown that features of tampered regions are not only more effectively represented but it also results in a lower feature dimension. At the second stage, we provide contextual information by including neighboring patches to further improve the detection results. To the best of our knowledge, this work is the first work that utilises deep learning approach for feature learning in the field of tampered region localization.

In the following, Section 2, we will introduce our proposed method. Experimental results are shown in Section 3 and Section 4 concludes the paper.

2. Proposed Method

2.1. Basic Features Generation

As we are looking into tampered characteristics, we are unable to use pixels dependency as per traditional deep learning object recognition models [13]. Hence, we have to devise some basic features for the deep learning to learn and transform the initial features. We first converted the image into a YCrCb color space as studies [22] have shown that this color space is known to be more sensitive to tampering artifacts. We then segmented the image into 32 by 32 patches. Finally, we applied a 3 Level 2D Daubechies Wavelet

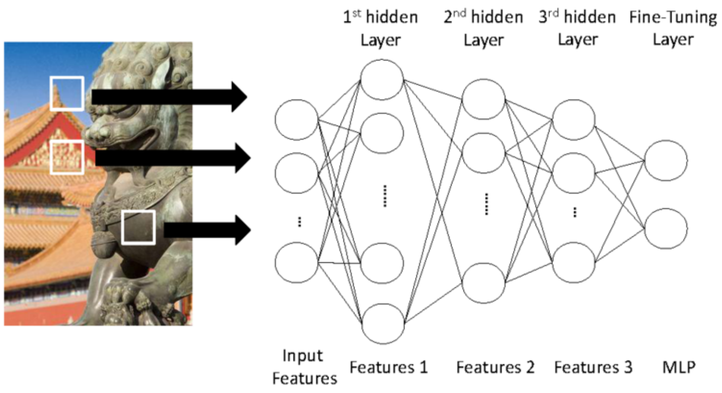


Figure 1. Stacked Autoencoder Architecture. The 1st, 2nd, 3rd hidden layers are for unsupervised learning with the 4th layer being a supervised classification network for feature fine tuning.

decomposition to each YCrCb component of the patches. We then obtained the standard deviation, mean, and the sum for each of the approximation, horizontal, vertical and diagonal coefficients to obtain 90 features. In addition, we applied Daubechies orthogonal wavelets D2-D5 to obtain a total of 450 basic features.

2.2. *Two Stage Training Hierarchy for Tampering Detection*

Based on the above raw input, we derive a complex feature with better discriminative ability to distinguish the tampered patch from the authentic ones. To achieve this, we propose a two-stage training hierarchy as follows. For the first stage, we utilize a SAE model to learn the complex feature for each individual patch. For the second stage, we integrate the contextual information of each patch so that the detection can be conducted more accurately.

2.2.1. *1st Stage: Stacked Autoencoders for Complex Feature Learning*

At the 1st stage, we use a SAE for complex feature learning. A SAE is a neural network that is built by stacking multiple layers of basic autoencoders together. The outputs of each layer are treated as inputs to the successive layer. On top of the SAE, there usually comes with an additional MLP layer to further tune its parameters. The overall structure is shown as in Fig. 1 with three hidden layers plus a MLP layer. A SAE is known to be able to learn features that can represent its input. Typically, at the 1st layer, it learns the first order features. At the 2nd layer, it learns the second order feature which corresponds to the patterns of the first order features and so forth . Consequently, high layers of the SAE tends to learn higher-order features and have very good discriminative power. Consider a SAE with parameters W^l, b^l denoting the parameters for l^{th} autoencoder. Output of the l^{th} layer is $ae^{(l)}$ with its input z^l. Then the encoding of the input feature vectors over stacked autoencoders is performed by the encoding of each layer forwards as follows:

$$ae^{(l)} = f(z^{(l)}) \tag{1}$$

$$z^{(l+1)} = W^{(l)}ae^{(l)} + b^{(l)} \tag{2}$$

where $f(\cdot)$ is an activation function and a common choice is the sigma function:

$$f(x) = \frac{1}{(1+\exp(-x))} \tag{3}$$

The decoding of SAE is performed in a reverse way, i.e., by an decoding of each layer backwards:

$$ae^{(n+l)} = f(z^{(n+l)}) \tag{4}$$

$$z^{(n+l+1)} = W^{(n-l)}ae^{(n+l)} + b^{(n-l)} \tag{5}$$

In this aspect, if we have a SAE with n layers, then the transformed complex feature y is encapsulated as the activation of the last layer:

$$y = ae^{(n)} \tag{6}$$

To obtain the parameters of the SAE, we use the greedy layer-wise training [3]. Specifically, for the first layer, we use the raw input as introduced in Section 2.1 to obtain the parameters $W^{(1)}, b^{(1)}$. Then the activation output is used as the input of the second layer to obtain the parameters $W^{(2)}, b^{(2)}$. To obtain parameters more accurately, at the top of the second layer, we additionally add a layer each two nodes indicating either tampering or authentic and unroll and trained the whole architecture as a MLP [3]. So that for a new incoming image patch, we can represent them using Eq. (6).

2.2.2. 2^{nd} Stage: Context Learning for Tampered Regions

Since tampered regions usually span across a few patches and would consist of different shapes and sizes, so the contextual information can additionally indicate if a patch is tampered or authentic. Therefore, for each patch, we introduced another layer to integrate the contextual information. To be specific, we firstly divide the image into non-overlapping 32 by 32 small patches. For each patch p, we determine its contextual neighborhood, say $\mathbf{N}(p)$, and we assume that there should exist a consistent feature pattern among patches within this neighborhood.

$$\mathbf{N}(p) = [y_p^0, y_p^1, y_p^2, ..., y_p^k] \tag{7}$$

where y_p^0 is the complex feature learnt by the introduced SAE from the patch p. And $y_p^i, i \geq 1$ is the feature of its i^{th} neighboring patch. Through this representation, the relation among spatially-close patches is preserved. The final prediction label is determined by an average value over its neighboring patches.

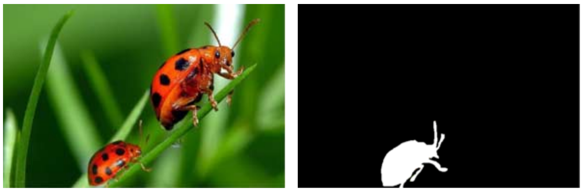
$$Prob(p) = \begin{cases} 1 & \frac{1}{k+1}\sum_{y_p^i \in \mathbf{N}(p)} MLP(y_p^i) \geq \alpha, \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

where $MLP(\cdot)$ is the probability value predicted by the MLP trained in the first stage and α is an threshold to binarize the map. In this paper, α is taken as 0.5, a midpoint of the maximal probability. For the neighborhood selection, we choose k = *3*. That is to say, for each patch, we include its right, bottom and bottom-right three patches, position of each of which overlaps with the patch itself by half patch size horizontally, vertically and both, respectively.

3. Experimental Results

3.1. Data Setup

One of the major obstacles in image region localisation is the lack of an open source database available for benchmarking. The Columbia Image Splicing Database [15] and the CASIA database [8] are the only Image Forgery database available to date. However, these databases only denote whether the images were tampered or not and do not provide ground truths of the tampered regions. In order to overcome this obstacle, we manually labelled 1000 images randomly selected from the CASIA 1 and 2 databases. For the task of labelling the ground truth, we referred to their published instructions [8] on how the images were tampered and labelled the ground truths accordingly. Fig. 2 shows an example of the tampered image and the manually labelled ground truth respectively.



(a) Tampered image (b) Labelled ground truth

Figure 2. Example of tampered region and ground truth.

For training using the SAE strategy, we randomly selected 770 images for patch level training. There are totally 36439 tampered patches and 63561 authentic patches. The data is mildly imbalanced as the tampered regions are typically smaller than the rest of the authentic regions. However, we show in our experimental results that the imbalanced training data does not affect the final overall accuracy.

The remaining 230 unseen images (where there are 135 authentic images and 95 tampered images) were then used for validating our proposed method.

3.2. Training

Using the training architecture discussed in Section 2, in the first layer, we learnt our features using a SAE with 3 hidden layers. Hence, the network's input is 450 dimensions and the number of neurons in the SAE's remaining layers are 500-256-128-2. At the second layer which integrates contextual features, the average of the MLP values of the 3 half-overlapped neighbouring patches are further used to calculate the final prediction value.

The SAE training was performed using a Core i7 PC with 24GB RAM in a Matlab environment. The total training time was 13 hours based on the amount of training data.

3.3. Evaluation

Utilising the proposed structure, we generated the detected tampered regions for each of the 230 unseen images. As our model uses patches as inputs, we evaluated our proposed method based on patches as well. Since the ground truth is labelled at pixel-level, so the corresponding grids will be labelled with the same labels, based on which all the afterwards evaluation are conducted. To be specific, the images are segmented into 32 by 32 grids, overlapped with the ground truth and labeled accordingly. Fig. 3 shows an example, the left figure is the original pixel-based ground truth while the right one is the patch-based ground truth. Based on this, there are 4 possible scenarios among the results:

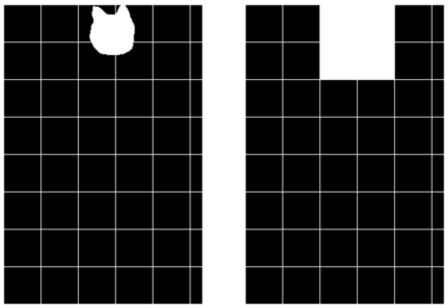


Figure 3. An illustration of patch-wise ground truth.

1) if both the output and the ground truth are labelled as tampered (in white color), we take the result as a True Positive (TP). 2) if the output grid is predicted as tampered (in white color) but the ground truth grid is labeled as authentic (in black color), the result is a False Positive (FP), 3) if the ground truth grid is authentic, but the output grid is predicted as authentic, the result is a False Negative (FN), 4) if both the output and ground truth label of a grid are authentic, the result is a True Negative (TN). Based on the above, we evaluate the results using the following criteria:

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \tag{9}$$

$$Fallout = \frac{FP}{FP+TN} \tag{10}$$

$$Precison = \frac{TP}{TP+FP} \tag{11}$$

For accuracy and Precison, the higher the values are, the better the classifier performs. For the Fall-out, the lower the value is, the better the classifier is. The statistics of the above criteria are summarized in Table 1. While our Fall-out is kept to a minimal of 4.31%, our precision is at 57.67%, which indicates that our proposed method can well predict the label of individual patch. As our method is dependent on patches, the detected tampered regions may not necessarily cover the exact parts of the ground truth grids and this results in more false positives which may effect the value of precision. However, this is can be solved by some image based post processing techniques. One possible solution is to leverage image segmentation in which way the labels of the segments are propagated from the corresponding patch labels. Therefore, we can effectively recover the tampered object since the localized region have already been identified.

Since the CASIA dataset contains both JPEG and TIFF images, and our method does not depend on specific image format, we further investigate the performance for each of them. As there is no authentic image with TIFF format, we reported the results for

Table 1. Performance Matrix for all test images.

Fall-out	4.31%
Precision	57.67%
Overall accuracy	91.09%

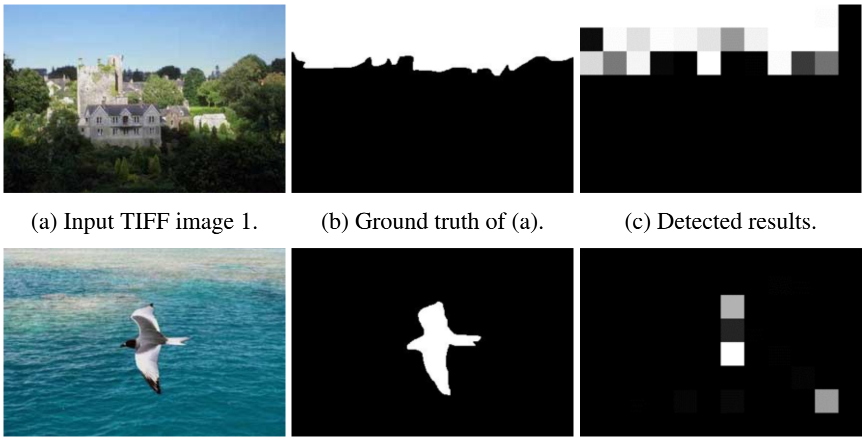

(a) Input TIFF image 1. (b) Ground truth of (a). (c) Detected results.
(d) Input TIFF image 2. (e) Ground truth of (d). (f) Detected results.

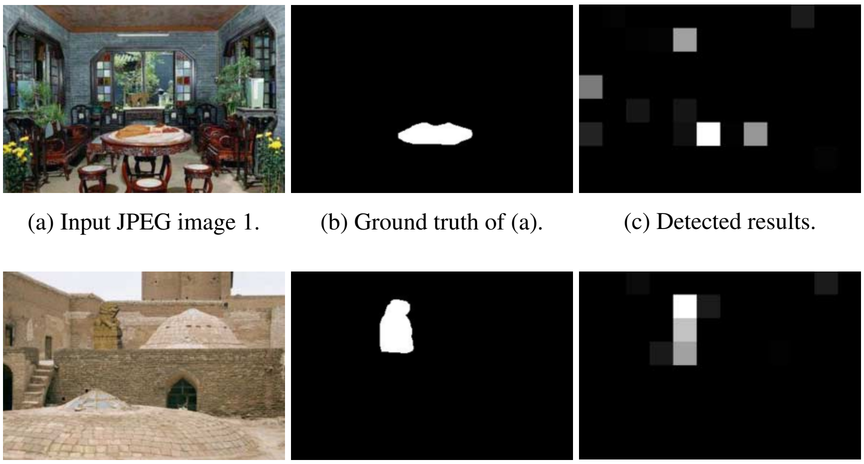
Figure 4. Detection results for TIFF image examples from CASIA 2 database.

tampered images only. Among the 95 tampered images, there are 51 JPEG and 44 TIFF, with results shown in Table 2. From the data, we can see that the performance for JPEG and TIFF are mostly similar, which indicates that our proposed method has the capability to be applied to various format images. As illustrated in table 2, the overall accuracies are consistent for each image format at 87.51% for JPEG and 81.91% for TIFF respectively. In addition, the fall-outs are kept to a minimal of 7.09% and 4.39% for JPEG and TIFF, separately.

Table 2. Performance Matrix between Tampered JPEG and TIFF images

	JPEG	TIFF
Fall-out	7.09%	4.39%
Precision	59.43%	80.65%
Overall accuracy	87.51%	81.91%

To visualize the results, we provided some examples in Fig. 4 and Fig. 5. Fig. 4 includes two examples of TIFF images while Fig. 5 illustrates two JPEG examples. Within each figure, the first column is the tampered images, the second column is the ground truth and the last column is our detection results. Note that for the detection results, we plot them in a grayscale version instead of binary values just for better visualization purpose. But all the statistics reported in the above table are based on binary label (i.e., either tampered or authentic). As shown in the these examples, our proposed method is capable of identifying the tampered regions accurately.



(a) Input JPEG image 1. (b) Ground truth of (a). (c) Detected results.

(d) Input JPEG image 2. (e) Ground truth of (d). (f) Detected results.

Figure 5. JPEG images from CASIA 2 database

3.4. Comparison with existing work

The above provides a results overview of our proposed method. In this section, we evaluate our method against existing works. Since JPEG image is the most popular image format today and majority of the work which localize tampered region such as [20,14,6,4,24] are only applicable to JPEG images, we compare our work with two existing studies in JPEG domain. Both of these works exploit the double compression artefacts left after tampering. The first method is proposed by Thing et al. [20] which leverages the double quantization effect among JPEG images for tampering detection. We choose it as it claims to be robust in practical applications even with a relatively limited or small training data set available. Additionally, this work is reported to perform well for CASIA dataset, so we believe it to be a good baseline. The second one is proposed by Bianchi et al. [4] based on an improved and unified statistical model characterizing the artifacts that appear in the presence of both A-DJPG or NA-DJPG.

Table 3. Detection accuracy comparison with existing works among all JPEG images.

Author	Accuracy
Bianchi et al. [4]	40.84%
Thing et al. [20]	79.72%
Proposed Method	**87.51%**

The experiments are conducted among the all JPEG images. For [4], we obtained their codes from the authors website. As for [20], we contacted the authors to obtain their codes for evaluation on our database. However, these existing methods are pixel based as compared to our proposed method which is patch base. Hence, in order to have a fair comparison, we took two average score from the corresponding patch in the existing method. Using the same 32 by 32 patch wise evaluation, we obtain the accuracy as shown in Table 3.

From the results, we can see that our proposed method achieves the highest accuracy at 87.51%, which outperform the second best by Thing et al. [20] by 7.79% and outperform the work by Bianchi et al.[4] by around 46.67%. In [4], the output of their algorithm is a probability map corresponding to the probability of the region being double compressed. Our results illustrate two main drawbacks on this method; firstly, the tampered regions need to be manually identified since the authors did not provide a threshold for detecting the regions. Secondly, the tampered regions are not accurately detected on our test database.

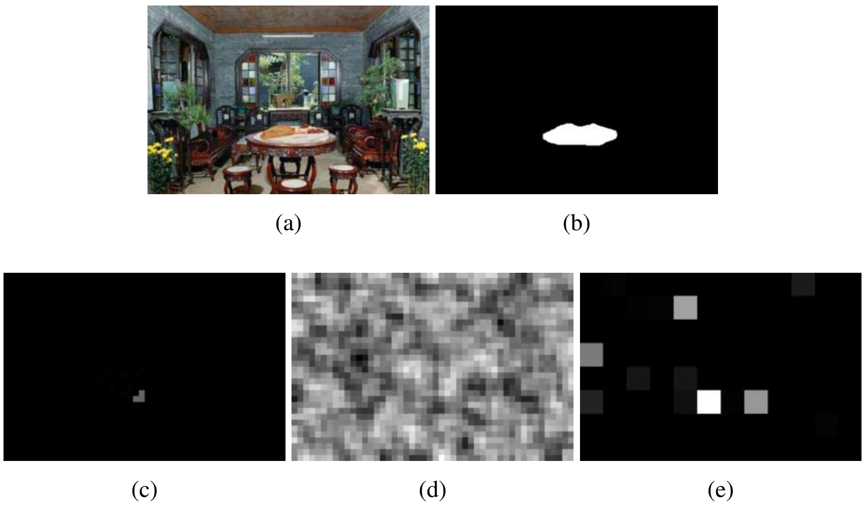
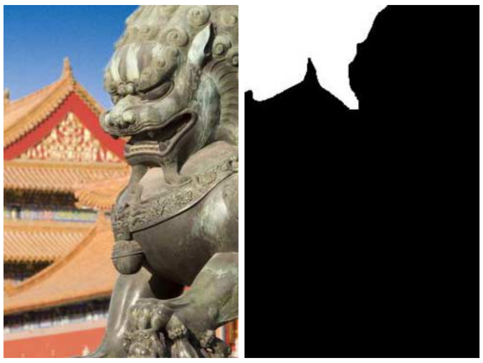


Figure 6. Comparison of detected tampering region, Example 1. (a) Input JPEG image, (b) Ground truth (white region is tampered), (c) results by Thing et al. (d) results by Bianchi et al. (e) results of our method.

We visually compare the detected regions with the existing works [20,4] as illustrated in Figs. 6 and 7. From the figures we can see that our proposed methods can detect the tampered region accurately. Furthermore, the method by [4] requires a threshold to be set in order to automatically identify the tampered region. Based on the table, we can observe that our proposed method obtained higher accuracy over existing works on the same test set. Furthermore, the advantage of our proposed method is that it is also applicable to both JPEG and TIFF image formats.

4. Conclusions

In this paper, we propose a deep learning approach for feature learning used to characterize tampered regions across multi format images. Our experiment results demonstrate that the proposed method detects tampered regions well with an overall accuracy of 91.09%. As future work, we will include other image transforms such as DCT as the base feature input. We will also investigate if other deep learning architectures such as Deep Belief Networks will improve the performance of feature learning. In addition, we will continue our efforts to manually label more ground truths from other datasets such as the Columbia Image Splicing dataset [15] as it also includes BMP file formats. This



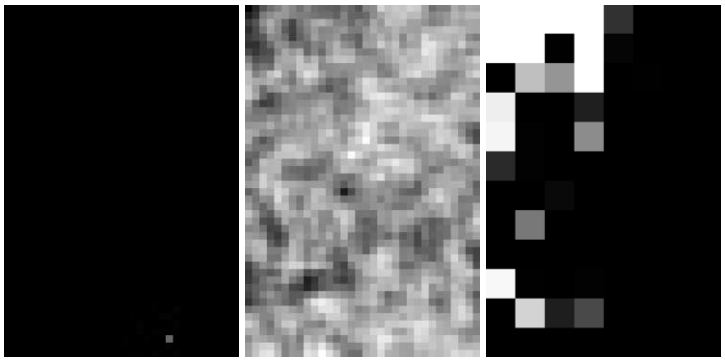


Figure 7. Comparison of detected tampering region, Example 2. (a) Input JPEG image, (b) Ground truth (white region is tampered), (c) results by Thing et al. (d) results by Bianchi et al. (e) results of our method.

will allow the deep learner to learn more characteristics of tampered regions and ensure better accuracy for tampered region localization across different image file formats.

Acknowledgement

This material is based on research work supported by the Singapore National Research Foundation under NCR Award No. NRF2014NCR-NCR001-034.

References

[1] I. Amerini, L. Ballan, R. Caldelli, A. D. Bimbo, and G. Serra. A sift-based forensic method for copy and move attack detection and transformation recovery. *IEEE Transactions on Information Forensics and Security*, 6(3):1099–1110, March 2011.

[2] I. Amerini, R. Caldelli, V. Cappellini, F. Picchioni, and A. Piva. Estimate of prnu noise based on different noise models for source camera identification. *Crime Prevention Technologies and Applications for Advancing Criminal Investigation*, page 9, 2012.

[3] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007.

[4] T. Bianchi and A. Piva. Image forgery localization via block-grained analysis of jpeg artifacts. *IEEE Transactions on Information Forensics and Security*, 7(3):1003–1017, June 2012.

[5] I. C. Chang, J. C. Yu, and C.-C. Chang. A forgery detection algorithm for exemplar-based inpainting images using multi-region relation. *Image and Vision Computing*, 31(1):57–71, January 2013.

[6] Y. L. Chen and C. T. Hsu. Detecting recompression of jpeg images via periodicity analysis of compression artifacts for tampering detection. *IEEE Transactions on Information Forensics and Security*, 6(2):396–406, June 2011.

[7] V. Christlein, C. Riess, J. Jordan, C. Riess, and E. Angelopoulou. An evaluation of popular copy-move forgery detection approaches. *IEEE Transactions on Information Forensics and Security*, 7(6):1841–1854, December 2012.

[8] J. Dong and W. Wang. Casia tampering detection dataset, 2011.

[9] J. Fridrich. Digital image forensics. *Signal Processing Magazine, IEEE*, 26(2):26–37, 2009.

[10] J. Goh and V. L. L. Thing. A hybrid evolutionary algorithm for feature and ensemble selection in image tampering detection. *International Journal of Electronic Security and Digital Forensics*, 7(1):76–104, March 2015.

[11] Z. He, W. Lu, W. Sun, and J. Huang. Digital image splicing detection based on markov features in dct and dwt domain. *Pattern Recognition*, 45(12):4292–4299, 2012.

[12] P. Kakar and N. Sudha. Exposing postprocessed copy-paste forgeries through transform-invariant featury. *IEEE Transactions on Information Forensics and Security*, 7(3):1018–1028, June 2012.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105. 2012.

[14] Z. Lin, J. He, X. Tang, and C.-K. Tang. Fast, automatic and fine-grained tampered jpeg image detection via dct coefficient analysis. *Pattern Recognition*, 42(11):2492–2501, January 2009.

[15] T. T. Ng, J. Hsu, and S. F. Chang. Columbia image splicing detection evaluation dataset, 2004.

[16] A. Popescu and H. Farid. Exposing digital forgeries in color filter array interpolated images. *Signal Processing, IEEE Transactions on*, 53(10):3948–3959, Oct 2005.

[17] K. Rosenfeld and H. T. Sencar. A study of the robustness of prnu-based camera identification. In *IS&T/SPIE Electronic Imaging*, pages 72540M–72540M. International Society for Optics and Photonics, 2009.

[18] H. C. Shin, M. R. Orton, D. J. Collins, S. J. Doran, and M. O. Leach. Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4d patient data. *IEEE Pattern Analysis and Machine Intelligence*, 35(8):1930–1943, August 2012.

[19] P. Sutthiwan, Y. Shi, H. Zhao, T.-T. Ng, and W. Su. Markovian rake transform for digital image tampering detection. In *Transactions on Data Hiding and Multimedia Security VI*, volume 6730 of *Lecture Notes in Computer Science*, pages 1–17. Springer Berlin Heidelberg, 2011.

[20] V. L. L. Thing, Y. Chen, and C. Cheh. An improved double compression detection method for jpeg image forensics. In *IEEE International Symposium on Multimedia*, pages 290–297, December 2012.

[21] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research*, 11:3371–3408, Dec. 2010.

[22] W. Wang, J. Dong, and T. Tan. Effective image splicing detection based on image chroma. In *IEEE International Conference on Image Processing*, pages 1257–1260, November 2009.

[23] W. Wang, J. Dong, and T. Tan. Exploring dct coefficient quantization effects for local tampering detection. *Information Forensics and Security, IEEE Transactions on*, 9(10):1653–1666, Oct 2014.

[24] F. Zach, C. Riess, and E. Angelopoulou. Automated image forgery detection through classification of jpeg ghosts. *Pattern Recognition*, 7476:185–194, January 2012.

Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016
A. Mathur and A. Roychoudhury (Eds.)

doi:10.3233/978-1-61499-617-0-13

Q-Floid: Android Malware detection with Quantitative Data Flow Graphs

John Henry CASTELLANOS [a], Tobias WÜCHNER [b], Martín OCHOA [b,c] and Sandra RUEDA [a]
[a] *Universidad de Los Andes, Colombia*
[b] *Technische Universität München, Germany*
[c] *SUTD, Singapore*

Abstract. Due to the rapid proliferation of Android malware, conventional anti-malware signature-based solutions face significant challenges to battle against cybercrime. In this work, we propose to use quantitative data flow profiles between system entities such as processes, files and sockets to detect malicious applications on Android, an approach which has been shown to be promising for detecting Windows malware. Our approach uses features based on graph-theoretical metrics as a basis for the analysis. Those features are trained through multiple machine learning algorithms obtaining malware detection rates of up to 93% for variants of known (trained) families and detection of new families of up to 84%.

1. Introduction

The market penetration of the Android operating system for mobile devices (with a 78% of the global market [16]), the amount of sensitive information stored in and processed by mobile devices, and the lack of security awareness of users downloading mobile apps, have attracted cyber-criminals to the Android platform, leading to an alarming proliferation of Android malware. Recent reports [24,17,12,27] coincide to estimate that the vast majority (about 98%) of mobile malware is designed to compromise Android. Moreover, a recent study [2] revealed that 97% of the top 100 paid Android Apps and the top 20 most popular free apps have been used to create trojans.

Application markets offer several mobile anti-virus solutions. Most of them operate by performing static analysis on information such as the package name and the application manifest. This type of anti-virus solutions faces limitations, such as the Android's sandboxing architecture, which does not allow apps installed by the user (including anti-virus apps) to perform scanning of other apps during execution, preventing to detect malicious components. Furthermore, they base their classification decision on static properties and thus by definition are not capable of anticipating changes of an app at run-time, i.e. through run-time deferred loading of code [19]. The limitations of such static detection techniques are emphasized by studies [11] which showed that by making small changes to the code of a malicious app and its manifest file one can successfully bypass the top 10 Android anti-virus.

The vast majority of malicious apps on mobile devices are trojans, which hijack a legitimate app and implant malign code into them. Malign apps thus often have a dual behavior because apparently they are benign but internally they hide a malicious component out of the user's perception. Such dual behavior usually does not reflect in changes to the manifest or other statically observable properties as it often is realized through run-time loading of malign code. Traditional (static) anti-virus approaches are thus often challenged by malware that hijacks benign apps.

A solution against such dual-behavior malware is dynamic behavior analysis, proposed by the research community [11,4,1,9,28,21] as an alternative to static solutions. The core idea of behavior-based detection is to discriminate malign from benign apps based on information gathered from executing the samples. This way, such techniques are more promising for coping with malware that changes its behavior at run-time.

In this work, we leverage dynamic analysis to capture the runtime behavior of an app. In contrast to related work [11,4,1,9,28,21], we do not directly use this captured behavior in terms of system or function calls, but abstract from it in form of so-called quantitative data flow graphs (QDFGs, [29]). QDFGs model program behavior as (quantitative) data flows between system entities such as processes, files and sockets, where nodes correspond to entities, and edges to the respective data transfers between them. The goal of our approach is thus to define properties of QDFGs that are characteristic for known malign or benign behavior and that can be used to correctly classify unknown apps.

To this end, we compute a wide range of metrics on QDFGs of known malware and goodware and use them as features for training supervised machine- learning classifiers. We then used the obtained classifiers to evaluate unknown samples. A prototypical implementation of our approach is evaluated from different perspectives (Section 4).

The core assumption behind the use of QDFG metrics for malware detection is that there exists a significant difference between data flow behavior of malign and benign programs that can be leveraged for discrimination. While this assumption already has been shown to hold for Windows malware [30,29], its validity has not yet been investigated for Android systems.

In sum, we tackle the problem of efficiently and effectively detecting Android malware, in particular dual-behavior malware, from a behavioral perspective. The contributions of our research thus are: **a)** We are the first to explore the application of QDFG analysis for detecting malware in Android; **b)** Our proposal achieves good results in comparison to closely related work in terms of efficiency and detection rate **c)** We show that, in contrast to similar experiments in Windows, achieving low false positive rates in Android malware by using system-call based QDFG graphs is challenging and requires further research.

2. Background

2.1. Android Architecture

The Android platform is organized in five different layers: applications, application framework, runtime and libraries, and kernel [7]. The first layer is composed of preinstalled applications, such as the ones developed by Google and device manufacturers, as well as applications installed by users.

The second layer offers support to developers performing common tasks, like accessing elements of the user interface and shared information, or providing message exchange among application components. This layer enables interaction among elements in the application layer and the runtime layer.

The runtime layer has core libraries and the Dalvik Virtual Machine (Dalvik VM). This VM was specially developed to adjust to conditions in mobile and embedded environments, like low memory and processing speed. Dalvik interacts with low-level native code using the Java Native Interface (JNI). The process responsible of starting services and applications on any Android device is Zygote; to do so, the Zygote creates a copy of itself, including a new Dalvik VM with all the libraries used by the Android framework. This element is of particular interest because the proposed system needs to learn of any transfer of information among applications, and this may be done by monitoring Zygote. Core services are those that form the underlying operating system environment and native Android components. The libraries support low-level functions required by the Android framework. Finally, the kernel layer is based on Linux.

2.2. Malware Analysis

Malware experts study malicious applications to determine the goals of these applications and their inner-workings. To this end, researchers use two types of techniques: static and dynamic analysis. Static analysis techniques study source code or binary representations of code, without running it. Dynamic analysis techniques, on the other hand, study the actual behavior of a sample while it is running in a controlled environment. Given that this work falls in the dynamic analysis category, we briefly recap related work in the area.

2.2.1. Dynamic analysis by monitoring function calls.

Functions and libraries encourage code reuse and help developers by freeing them from some implementation details. They are important for code analysis too, because they allow researchers to build an abstract representation of actions that may lead to a better understanding of a program's behavior.

In dynamic analysis, it is possible to track program behavior by monitoring system calls that support program requirements, and system call interposition is one way of achieving this. Traditional operating systems run programs in one of two modes: kernel or user, but only programs running in kernel mode have access to protected resources and the system. For example, a program running in user mode cannot create or open a file by itself, operating systems provide a system call interface to support this kind of requirements, and user-level programs must use this interface to request an operating system to execute privileged operations.

Malware programs that run at user level also have to use the system call interface to request privileged operations. Therefore, it is possible to use this interface to monitor calls and study application behavior. System tools, like `strace`, allow monitors to capture function calls from analyzed programs and record them in a log file for analysis. Malware programs that run at the kernel level can therefore evade this type of monitoring.

The result of this type of monitoring is a trace of all function calls; i.e. a sequence of functions that were invoked by the program that is being analyzed and the parameters of the call. Related proposals in the literature use such traces to build a semantic representation of malware behavior. Christodorescu et al. [5] convert traces of functions in graphs and compare them for classification purposes , Xu et al. [31] propose a method for identi-

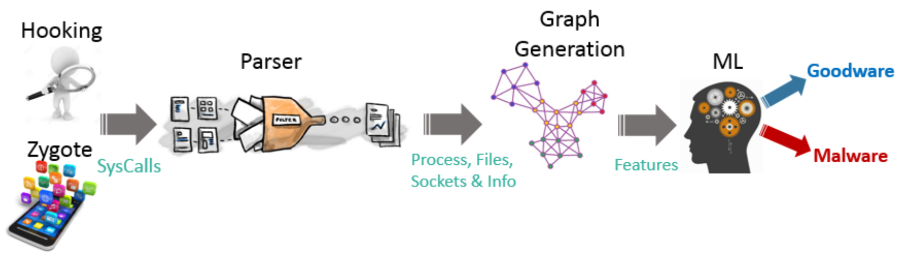


Figure 1. Overview of the classification mechanism.

fying polymorphic variations of known threats by comparing traces of function calls, and Wuechner et al. [29] detect Windows malware based on data-flow graphs that represent system calls. Some proposals have implemented similar techniques for Android [4], [10], and for intrusion detection in other platforms [14,18].

3. Approach

Our approach extends previous work in dynamic analysis for Android [4], [10], [28] in that we propose a different technique to identify malicious programs, i.e. classification based on volumes of transferred data between system entities. This idea was already successfully applied to Windows environments [29], [30] but has not been evaluated in Android yet.

For our approach we leverage dynamic analysis to monitor and log data transfers between entities, (processes, files, and sockets), in Android systems. Using the captured behavior of several executed known benign and malign apps we then create the respective QDFGs, compute characteristic metrics on them, and use them as input for training supervised machine learning classifiers to later classify unknown samples.

The intuition for the utility behind this proposal is that malware apps are likely to have higher rates of intra-system data transfers than goodware, as a consequence of data flow intensive malign behavior like messages or file transfers to leak user data to remote servers, download malicious code, or listen to SMS messages.

3.1. Overview of the Proposal

Data transfers can be tracked at the kernel level, by monitoring requests involving data exchange between processes, and to and from files and sockets. Android uses the Zygote process to support access of the Dalvik virtual machine processes, including applications, to system resources. Thus, capturing actions from the Zygote process, at kernel level, also allows capturing and logging actions from applications.

The resulting log is later processed and cleaned to build the corresponding QDFGs. These graphs are suitable to represent interactions between processes, files, and sockets, as well as how much data they carry. The information of the QDFGs is then synthesized in features (i.e. characteristics that allow separating malign from benign QDFGs).

Since the values of these features are different for different applications, it is not feasible to manually define thresholds and useful features combinations to form an effective classification model. To address this issue, we instead use supervised machine-learning

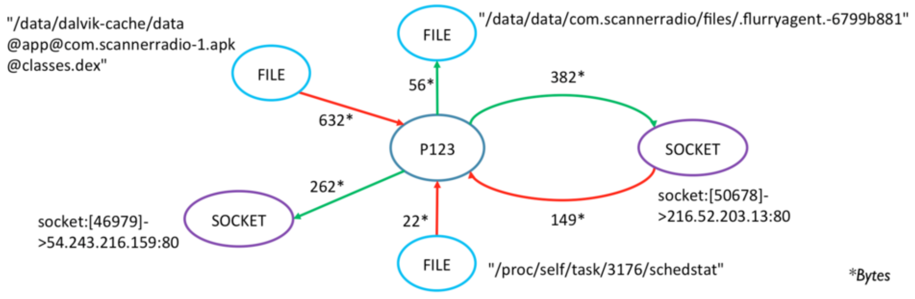


Figure 2. QDFG for one process (P123) and its flows with other entities.

algorithms, trained on features extracted from QDFGs of known malign and benign apps. Figure 1 describes the overall process.

3.2. Data Capture and QDFG generation

We used images of five different versions of Android (Froyo , Gingerbread, Honeycomb, Ice Cream Sandwich and Jelly Bean MR1) from the Android-x86 project [15] to create the data collection environment. A shell running `strace` monitors syscalls from the Zygote process and logs them.

The relevant syscalls are those that induce a data transfer between system entities like processes, files, or sockets. These include:

- Process creation and duplication syscalls: `clone()`, `fork()`, `dup()`.
- Instance creation syscalls: `open()`, `socket()`.
- Instance removal syscalls: `close()`.
- Information transfer syscalls: `read()`, `readv()`, `pread()`, `pread64()`, `write()`, `writev()`, `recvmsg()`, `sendto()`, `recvfrom()`.
- Communication control system: `pipe()`, `getsockname()`, `connect()`, `listen()`, `accept()`, `execve()`.

The key idea behind the QDFG generation is to interpret intercepted calls as data flows between system entities. Nodes in a QDFG represent system entities and flows represent data transfers between nodes. Flows are reflected in the creation or update of edges between nodes of a QDFG that model the respectively involved system entities. As all system calls described above, directly or indirectly, imply a data flow between two system entities (nodes), a QDFG evolves by creating edges between the respective nodes. Weights of the QDFG are thus obtained by accumulating the data flows of the associated monitored system calls. Figure 2 shows an example of a QDFG for one process and its interaction with other system entities.

In our prototype we constructed QDFGs based on the initial 5 minutes of application activity after starting up. In order to stimulate application activity, we also used the UI/Application Exerciser Monkey [1], an application for developers that automatically generates pseudo-random sequences of events (such as mouse clicks and touches) to simulate user activity.

[1]http://developer.android.com/tools/help/monkey.html

3.3. Feature extraction

After graph generation, we focused on identification of features, i.e. metrics, that capture behavioral characteristics in terms of differences between QDFGs of known malign and benign apps. In this section we discuss two sets of features. The first set is focused on metrics that characterize how a process shares information with all other entities of a system. The second takes a more simple viewpoint: it aims only at counting how many files and sockets are read/write from/to a process and how much data is transferred in the process.

3.3.1. Feature set 1: Process Analysis.

The first set of features is inspired by previous work on QDFG-based malware detection for Windows [30]. This approach assumes that malware applications and benign applications behave differently; malware actively look for user data to send it to remote servers, increasing the volume of data that flows between processes in a particular way. The features on this set can be categorized into three groups: direct, global, and graph-relation.

Direct Features. Features in this group measure direct interactions of a process. The extraction procedure analyzes processes as single entities, looking for actions over files and sockets, and ignoring interactions with other processes. Figure 2 shows a graph that represents direct interactions between a process and other entities in a system. This group includes the following features:

Feature	Description
DIRECT QDF	Amount of data (Bytes) shared by the process
DIRECT FROMFILES QTY	Quantity of input files
DIRECT FROMFILES QDF (%)	Data read from files / Data shared by the process
DIRECT FROMSOCKETS QTY	Number of input sockets
DIRECT FROMSOCKETS QDF (%)	Data read from sockets / Data shared by the process
DIRECT TOFILES QTY	Number of output files
DIRECT TOFILES QDF (%)	Data written to files / Data shared by the process
DIRECT TOSOCKETS QTY	Number of output sockets
DIRECT TOSOCKETS QDF (%)	Data written to sockets / Data shared by the process

Global Features. Features in this group measure direct and indirect interactions of a process, that means that activities of descendant processes are considered part of the activities of a parent process. All features in this group include activities generated by a parent and its descendants (p&d). They are:

Feature	Description
CHILD PROCESSES QTY	Number of descendant processes
GLOBAL QDF (Bytes)	Data shared by the process and their descendants
GLOBAL FROMFILES QTY	Number of input files (p&d)
GLOBAL FROMFILES QDF (%)	Data read from files (p&d) / Shared Data (p&d)
DIRECT FROMSOCKETS QTY	Number of input sockets (p&d)
DIRECT FROMSOCKETS QDF (%)	Data read from sockets (p&d) / Shared Data (p&d)
DIRECT TOFILES QTY	Number of output files (p&d)
DIRECT TOFILES QDF (%)	Data written to files (p&d) / Shared Data (p&d)

DIRECT TOSOCKETS QTY	Number of output sockets (p&d)
DIRECTED TOSOCKETS QDF (%)	Data written to sockets (p&d) / Shared Data p&d)

Graph-relation Features. Features in this group are inspired by metrics used for social network analysis as proposed in [30] that we adjusted for our Android context. Features in this group include:

- *Closeness Centrality* This feature measures how "close" a node is to other nodes in a graph. For a node in a QDFG, this value represents interactions among processes; a value closer to 1 represents more interactions. The expected value for benign applications is high as they usually have multiple components that interact to achieve a common goal, while the expected value for malicious application is low as they usually have multiple independent components.
- *Betweenness Centrality* This feature measures the centrality of a node in a graph, meaning that the node is often part of a shortest path between two any other nodes. As before, this metric allows discriminating processes based on their behavior with other system entities.

Dataset and Machine Learning Algorithm Selections. Given that different samples spawn a variable number of child processes, the aforementioned analysis would generate a non-constant number of features per sample. Although in general there are algorithms able to cope with this, most of the supervised algorithms provided by Weka [26], require a constant number of features for all samples.

Therefore, the goal is to have the same number of features per sample so they can be used to train a machine-learning algorithm. To achieve this goal, several datasets were generated. Figure 3 illustrates the context used to create different datasets; the first generation process corresponds to a Zygote (the starting process), second generation processes are children of the first generation process, third generation processes are children of any second generation process, etc.

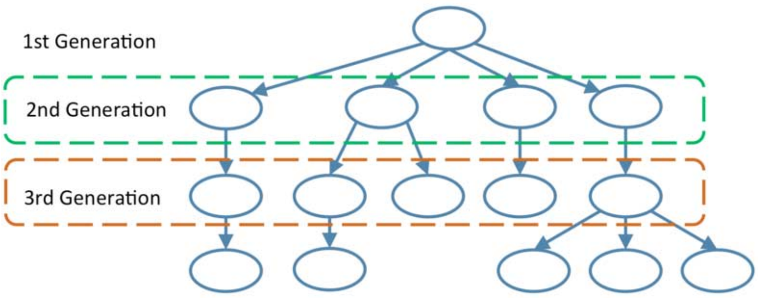


Figure 3. Hierarchical order of processes.

The following datasets were considered: Features of the two more active second generation processes, the three more active second generation processes, and the five more active third generation processes. For these features, if the number of processes was less than the maximum (2,3,5), the feature vector was artificially completed with 0s.

3.3.2. Feature set 2: Files and Sockets Analysis.

Like the previous set, the approach to define this set assumes that malware applications are more active than non-malware applications; they transfer more data and interact with many entities in a system. This set has the following features:

Feature	Description
FileCounter and SocketCounter	Number of files and sockets handled by the sample
Proc2File and QDF_Proc2File	Number of processes writing data to files, and data (bytes)
Proc2Socket and QDF_Proc2Socket	Number of processes writing data to sockets and data (bytes)
File2Proc and QDF_File2Proc	Number of processes reading data from files and data (bytes)
Socket2Proc and QDF_Socket2Proc	Number of processes reading data from sockets and data (bytes)

4. Evaluation

For this work, we considered over 2500 mobile applications, including benign and malicious programs, collected by the SIIS Laboratory from the Pennsylvania State University [25], the Genome Malware Project [32], the contagio minidump project [2] and the Android Sandbox project [3]. The applications that use techniques to detect virtualized environments were discarded from the evaluation group since they would generate trivial QDFGs. This resulted in a set of samples composed of 2115 active Android applications, of which roughly 50% were malware and 50% goodware. For the sample used, the procedure generated graphs with different sizes, ranging between 25-980 nodes and 30-1660 edges. In order to train the chosen machine-learning algorithm and to test the model, we used the Weka Framework [26].

4.1. Choosing a classifier

These datasets were evaluated with several machine learning algorithms using 10x cross-validation to determine the best dataset-machine learning algorithm combination (the best one identifying malware) The evaluated algorithms include: Naive Bayes (NB), Bayes Net (BN), Bayesian Logistic Regression (BLR), K-nearest Neighbors (KN), Locally Weighted (LW), J48, Random Forest (RF) and Decision Tree (DT).

Process related features. In the following we summarize the results for different combinations of datasets and algorithms. The best combination is a dataset with the three more active second generation processes and the Random Forest algorithm.

File and sockets related features. These features were evaluated with the same algorithms used for the proposal based on process analysis to determine the best dataset-machine learning algorithm combination. In this case, since the features were not computed per process but per sample, we only have one results row. The following table summarizes the results for process and file-sockets related features tested with multiple ML Algorithms. The Random Forest algorithm gave the best result and will be used in the following to further evaluate results.

[2]http://contagiominidump.blogspot.sg/
[3]http://www.androidsandbox.net

	Algorithms							
Dataset	NB	BN	BLR	KN	LW	J48	**RF**	DT
2 2nd generation procs.	59,7	68,4	52,2	70,2	62,6	72,9	78,7	71,8
3 2nd generation procs.	61,7	68,9	55,1	69,1	62,3	73,5	**80,7**	71,6
5 3rd generation procs.	60,9	69,8	52,0	66,3	58,4	70,4	76,4	69,0
File-Sockets	66,7	70,8	47,1	75,7	63,7	77,4	**83,6**	75,4

Table 1. Process and File-Sockets features tested with multiple ML algorithms. Values are true positive rates in percentage.

Combining feature sets. We combined the two feature sets in order to improve the precision of the analysis, however we obtain some slightly better results. These are shown on table 2.

4.2. Simulation of 0-day malware

In order to simulate a realistic application of our approach we performed the following experiment. A training phase using the Random Forest algorithm was performed on 85% of the samples, including malware families like ADRD, AnserverBot, BaseBridge, DroidKungFu, Geinimi, GoldDream and Pjapps. In a second phase we tested the generated model with the remaining samples in two different ways. One corresponds to detection of known families (families used in training phase), and a second one evaluates the detection capabilities when testing a set of unknown families or also called 0-days samples. This experiment tries to emulate new malware since these families were not used in the training phase. To make this experiment as realistic as possible, we biased the test set to contain the most recent samples. The results are depicted in Table 2.

		TPR	FPR	AUC
Known Families	Process Analysis	92,57%	49,47%	0,825
	Files and Sockets Analysis	89,71%	24,21%	0,922
	Both Features Analysis	90,85%	28,42%	0,895
0-day Families	Process Analysis	78,46%	43,44%	0,693
	Files and Sockets Analysis	83,59%	36,06%	0,813
	Both Features Analysis	75,90%	22,95%	0,809

Table 2. Detection results with known and 0-days families. *TPR*: True Positive Rate. *FPR*: False Positive Rate. *AUC*: Area Under Curve. Index which represents how reliable a classifier is, 1 is ideal.

Preliminary discussion of results Although the detection capabilities of our approach are acceptable for most experiments, the false positive rate is likely too high to be of practical value. This contrasts with previous experiments of similar approaches on windows malware [29,20,30], and has multiple possible explanations. On the one hand, by only observing system calls at the kernel level, we are potentially ignoring discriminating inter-process communication happening at a higher abstraction level (such as the Android VM). On the other hand, Android malware is typically based on existing goodware (trojans), and therefore shares common behaviors with legitimate applications, which confuses the classifier [23]. Similarly high false positive rates have for example been observed in other malware classification approaches from the literature [1]. Finally, our experiments did not use advanced stimulus/response techniques in order to collect more behavioural data from applications, which is fundamental for our analysis.

4.3. Performance

For the evaluation we used a machine equipped with an Intel Core2 Duo (2,2 GHz) processor and 2 GB of RAM, running on Lubuntu 14.10. To extract data for each sample we had the following running times.

- *VM starting* 20-45s.
- *App installation and monitoring* 301-318s.
- *Graph generation from logs* 1-18s.
- *Process feature computation* 1-275s (quadratic depending on number of edges).
- *File-sockets feature computation* 0,005 - 0,06 s.
- *Sample classification* 1,5 ms./sample.

In sum, our approach is lean and can be used for fast classification in case of isolated applications, but potentially also for running applications in real-time (after enough behavior has been collected).

5. Related Work

Malware analysis can be performed statically or dynamically. It is well known that static analysis techniques have some advantages (they do not suffer from evasion techniques such as detection of virtualized environments), but they also face disadvantages, such a fragility against code obfuscation [13].

On the early android days (2010), researchers Shabtai et al. proposed Andromaly [22], a framework that explores behavioural malware detection for android devices. Andromaly monitors multiple features such as CPU consumption, network traffic, memory status and keyboard events. The collected data is used to train machine learning algorithms. At that time malware in android was not as popular as now, so researchers had to develop their own malware specimens in order to test Andromalys accuracy. Although methodologically similar, our proposal differs from theirs basically in the kind of features considered: Q-floid takes data transferred between system calls and represents it as a Quantitative data flow graph. Later features (including graph-theoretical functions such as centrality metrics) are extracted from these graphs and used to train different machine learning algorithms. Also, our approach was evaluated against several hundred families of malware developed between 2012 and 2015.

One of the first approaches that uses system call analysis for malware detection in Android is Crowdroid [4]. This solution uses `strace` in real time to gather system calls for user activity, through an ad-hoc designed Android App, and a central server that processes information. The precision of the analysis is thus highly dependent on the amount of users installing the application and sharing their usage profile. Our work differs from theirs mainly in the profiling analysis (we build QDFGs, whereas they count the number of syscalls issued by a process), and the fact that we are independent from the number of monitored users, but rely on classified samples for training.

Park et al. [20] developed another approach for dynamic malware detection. They build behavioral graphs similar to ours, but do not take size of data transfers between entities into account. Their goal is to find a graph isomorphism between graphs that describe known malign samples and graphs of unclassified samples. Our solution, in contrast, uses machine learning on a constant number of features, such that the execution time of our detection procedure is independent of the number of malware samples used in training.

Andrubis [28] is a web-based tool for analysis of Android applications developed by the International Secure Systems Lab. This tool performs a hybrid analysis (static and dynamic) of user-supplied samples. First, Andrubis performs a static analysis on the manifest file of the application under analysis and extracts information such as permission, services, activities, package name and SDK version among others. Afterwards, it executes the sample in a controlled environment, targeting interactions with SMS messages, telephone calls and the file system. The focus of this proposal is however to aid manual analysis of suspicious samples, rather than an automatic classification as our work.

CopperDroid [10] by Fattori et al. is a dynamic analysis platform for Android applications that focuses on IPC, RPC and object interaction. Similar to Andrubis, the main goal of this approach is to aid human analysts to investigate suspicious samples. Copperdroid supports a more advanced stimulus/response technology than our proposal, which is based on annotations on the manifest file of the applications under analysis and the monkeyrunner platform.

Amos et al. proposed the STREAM [1] platform to compare different machine-learning detection approaches for Android. Different from ours, the results available on STREAM are based on features such as battery consumption and memory and network usage profiling. Although their dataset is partially similar to ours (they use the GENOMA dataset, for instance), they analyzed the behavior of a slightly bigger number of samples (1738). Our results are comparable in terms of false positive rates (22 % vs. 14,55%).

Recently, Dimjasevic et al. [6] have proposed a behavioral approach for Android malware detection based on system calls. Their features rely on a *system call dependency graph* that takes into account the order of system calls. Our approach is more abstract (we aggregate system calls into weighted data flows between entities), and there exist indications that this kind of abstraction is more resilient to advanced behavioural obfuscation than approaches relying on the order of system calls such as *n*-grams [30,3]. Although their data set is bigger (ca. 12000), and thus not directly comparable, we obtain comparable detection rates (theirs: up to 96%) and worse false positive rates (theirs: up to 5%).

Closest to ours is the work of Wuechner et al. [29,30]. Similar to them, we use graph theoretical metrics for training classifiers. However, we also considered novel features (file-sockets) and different from them we analyze QDFGs for Android apps instead of Windows apps. Due to the nature of Android Malware (majority of trojan horses), we obtain a comparatively higher rate of false positives.

6. Discussion and Future Work

Q-Floid builds and analyzes QDFGs that represent the volume of exchanged data between system entities in order to detect Android malware. Similar to most dynamic analyses, Q-Floid cannot detect malicious programs that evade monitoring or wait for specific triggers [8,21]. Some malware families (e.g. BeanBot, DroidDreamLight among others) have this behavior and thus were not included in the experiments.

We propose two sets of features to classify malware. Both sets have accuracy higher than 90% for many families on their own, but on the negative side, implied a high number of false positives. However, in sum, the detection accuracy of Q-Floid is comparable to that of the closest related work, while being by construction robust against advanced behavioral obfuscation techniques (as preliminary indicated by experiments with QDFG detection in Windows [30,3]).

To test the ability of new malware detection, some families in the sample were not used during training, and were used later to mimic 0-day malware applications. A preliminary evaluation showed that our approach detected up to 92,6% of malware not previously seen of known families, and up to 84% for unknown families. In sum, we successfully transferred the concept of QDFG-based malware detection to Android and showed that quantitative data flow analysis has the potential to significantly improve the accuracy of behavioral malware detection, however there are several points where this analysis can be improved to obtain similar results as the ones in the literature for Windows malware.

As part of future work we plan to apply our methodology to a more comprehensive malware and goodware data set to further validate our current promising results, and to further test resilience against advanced behavioral obfuscation techniques with respect to the closest related work. We also plan to explore inter-process communication and advanced stimulus-response techniques to be able to enrich the observed behaviour, and thus widen the application of our approach.

References

[1] B. Amos, H. Turner, and J. White. Applying machine learning classifiers to dynamic android malware detection at scale. In *IWCMC*, 2013.

[2] Arxan. State of Mobile App Security. Apps Under Attack. Technical report, 2014.

[3] S. Banescu, T. Wüchner, A. Salem, M. Guggenmos, M. Ochoa, and A. Pretschner. A framework for empirical evaluation of malware detection resilience against behavior obfuscation. *Malcon*, 2015.

[4] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani. Crowdroid: Behavior-Based Malware Detection System for Android. In *SPSM*, 2011.

[5] M. Christodorescu, S. Jha, and C. Kruegel. Mining specifications of malicious behavior. In *ISEC*, pages 5–14. ACM, 2008.

[6] M. Dimjaševic, S. Atzeni, I. Ugrina, and Z. Rakamaric. Android malware detection based on system calls. 2015.

[7] J. J. Drake, P. O. Fora, Z. Lanier, C. Mulliner, S. A. Ridley, and G. Wicherski. Android Security Design and Architecture. In *Android Hacker's Handbook*. 2014.

[8] M. Egele, T. Scholte, E. Kirda, and C. Kruegel. A survey on automated dynamic malware-analysis techniques and tools, 2012.

[9] W. Enck, P. Gilbert, S. Han, V. Tendulkar, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth. TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones. *ACM Trans. Comput. Syst.*, 2014.

[10] A. Fattori, K. Tam, S. J. Khan, L. Cavallaro, and A. Reina. CopperDroid: On the Reconstruction of Android Malware Behaviors. Technical report, Royal Holloway University of London, 2014.

[11] R. Fedler, J. Schütte, and M. Kulicke. On the effectiveness of Malware protection on Android: An evaluation of Android antivirus apps. Technical report, Fraunhofer AISEC, Apr. 2013.

[12] Fortinet Inc. 2014 Threat Landscape Report. Technical report, 2014.

[13] H. Gascon, F. Yamaguchi, D. Arp, and K. Rieck. Structural detection of android malware using embedded call graphs. In *AIsec*, pages 45–54. ACM, 2013.

[14] S. A. Hofmeyr, S. Forrest, and A. Somayaji. Intrusion detection using sequences of system calls. *Journal of Computer Security*, 6, 1998.

[15] C.-W. Huang and Y. Sun. Android-x86 Project - Run Android on Your PC, 2014.

[16] IDC. Smartphone OS Market Share, Q4 2014, 2015.

[17] Juniper Networks Inc. 3rd Annual Mobile Threats Report. Technical report, 2013.

[18] F. Maggi, M. Matteucci, and S. Zanero. Detecting intrusions through system call sequence and argument analysis. *IEEE TISSEC*, 2010.

[19] D. Maier, M. Protsenko, and T. Müller. A game of droid and mouse: The threat of split-personality malware on android. *Computers & Security*, 2015.

[20] Y. Park and D. S. Reeves. Deriving common malware behavior through graph clustering. *Computers and Security*, 39(PART B):419–430, 2013.

[21] T. Petsas, G. Voyatzis, E. Athanasopoulos, M. Polychronakis, and S. Ioannidis. Rage against the virtual machine. In *EuroSec*. ACM Press, 2014.

[22] A. Shabtai, U. Kanonov, Y. Elovici, C. Glezer, and Y. Weiss. Andromaly: a behavioral malware detection framework for android devices. *Journal of Intelligent Information Systems*, 38(1):161–190, 2012.

[23] A. Shabtai, L. Tenenboim-Chekina, D. Mimran, L. Rokach, B. Shapira, and Y. Elovici. Mobile malware detection through analysis of deviations in application network behavior. *Computers & Security*, 43:1–18, 2014.

[24] V. Svajcer. Sophos Mobile Security Threat Report. Technical report, 2014.

[25] Systems and Internet Infrastructure Security Lab, The Pennsylvania State University. Smartphone application analysis.

[26] The University of Waikato. Weka 3: Data Mining Software in Java, 2013.

[27] Trend Micro Inc. Turning the Tables on Cyber Attacks. Responding to Evolving Tactics. Technical report, 2014.

[28] L. Weichselbaum, M. Neugschwandtner, M. Lindorfer, Y. Fratantonio, V. V. D. Veen, and C. Platzer. ANDRUBIS: Android Malware Under The Magnifying Glass. Technical report, Vienna University of Technology, 2012.

[29] T. Wüchner, M. Ochoa, and A. Pretschner. Malware detection with quantitative data flow graphs. *ASIACCS*, 2014.

[30] T. Wüchner, M. Ochoa, and A. Pretschner. Robust and Effective Malware Detection through Quantitative Data Flow Graph Metrics. *DIMVA*, 2015.

[31] J.-Y. Xu, A. H. Sung, P. Chavez, and S. Mukkamala. Polymorphic malicious executable scanner by API sequence analysis. In *HIS*. IEEE, 2004.

[32] Y. Zhou and X. Jiang. Dissecting Android malware: Characterization and evolution. In *S&P*, 2012.

This page intentionally left blank

Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016
A. Mathur and A. Roychoudhury (Eds.)

doi:10.3233/978-1-61499-617-0-27

Cyber and Physical Access Control in Legacy System Using Passwords [1,2]

Jia XU [a], Jianying ZHOU [a] and Liming LU [b]

[a] *Infocomm Security Department*
Institute for Infocomm Research, Singapore
e-mail: {xuj,jyzhou}@i2r.a-star.edu.sg
[b] *School of Digital Media and Infocomm Technology*
Singapore Polytechnic
e-mail: LU_LIMING@sp.edu.sg

Abstract. Passwords—secret combinations of symbols—play an important role in physical world security (e.g. watchword to prevent unauthorized entry into military forbidden area) from ancient times. With emergence and advance of digital computers and computer network, passwords are also widely adopted in cyber world security protection. In most applications, password protection stands on the frontier of cyber/physical security defense. Compromise of passwords might render the whole system insecure, and make thereafter sophisticated cryptography solution ineffective. However, secure management of a large number of random passwords is a great challenge to human brains. We propose a visual cryptography technique, which allows users to store and manage ciphertexts of randomly chosen passwords in mobile phone and decrypt them *manually* on demand. The stored passwords remain confidential, even if the mobile phone is infected by spyware (Assume the spyware can capture phone screen, and monitor phone CPU and RAM). We also analyze the security and feasibility of proposed method. Leveraging on this technique, we give a simple access control system based on passwords, which provides a low cost alternative solution for legacy system besides smart card based solution.

Keywords. Password Management, Visual Cryptography, Mobile Device, Spyware, Legacy System

1. Introduction

We are interested in finding a low cost (in term of money, training, deploy time and service shutdown time) cyber and physical access control solution to protect cyber and physical assets, for large legacy systems.

[1]This work was supported by the National Research Foundation (NRF), Prime Ministers Office, Singapore, under its National Cybersecurity R&D Programme (Award No. NRF2014NCR-NCR001-31) and administered by the National Cybersecurity R&D Directorate.
[2]A full version is available at Cryptology ePrint Archive https://eprint.iacr.org/2015/1161

1.1. Legacy System

Some legacy system with long history may still rely on traditional locks (e.g. padlock) to protect each room and important device. For large legacy system, thousands or more lock keys may be required. All management (e.g. distributing keys, labeling keys, searching keys, counting keys, storing keys, backuping keys) of lock keys have to be done manually. More importantly, to revoke a lock key after a staff quits his/her job, the lock and all matching keys have to be replaced by a new pair of lock and keys, which could be expensive.

Modern design of physical door access control widely adopts smart card (e.g. contactless card) based solution. Such smart card based solutions may require network connection and consistent electricity power supply. It might be too expensive for some legacy system to upgrade to modern smart card based solution, since computer network and/or electricity power supply may not reach every corner of some legacy system with large area. In addition, smart card based solution may not be very suitable to protect device or equipment possibly due to network and electricity power requirement.

1.1.1. Access Control using Passwords

We provide a low cost alternative solution for such legacy system: Switch from keyed locks to combination locks, and manage all combination keys, i.e. the passwords (possibly including computer passwords), using staff's own phones, synchronized with a trusted central server. As a result, revocation of a password can be done by a combination lock password reset and a synchronization operation. We remark that, in this solution, the server and user phones communicate via cell phone network (e.g 2G/3G/4G). In case that cell phone network does not cover every inch of the area of legacy system, the user phone can synchronize with the central server where signal is available and then can still manage passwords locally without network connection at any place.

A natural question is that: Is it much easier to steal a password than a lock key? We have to point out that, duplication of a physical lock key may not be essentially more difficult than duplication of a password. One can duplicate a physical lock key just via a photo [5,14], without physically contacting the lock key. With a good phone camera which is widely available nowadays, attacker may even take a photo on a bunch of keys remotely, which could be more serious than peeking passwords over shoulder. Even worse, typically, people are educated to protect password privacy when typing it during login process, but not educated to hide lock keys from others' vision range.

1.2. Password Management

In most (if not all) cyber/physical security system, authentication is the first step, and also one of the most important steps of security protection. Passwords, a secret combination of symbols, is widely adopted in cyber or physical authentication solution, for a long history. Password authentication alone or with other authentication factors (e.g. fingerprint, OTP token), is still the ubiquitous authentication method.

However, the one of most important step—password authentication— is also considered as the weakest link in security protection. It is well known [18] that: (1) a good password should be chosen randomly; (2) passwords should not be reused or shared across different accounts. But, on the other side, (1) random passwords are hard to remember for human brains; (2) it is even hard, or impossible, to remember many random passwords. Consequently, many users tend to choose simple passwords [19,17], which have a certain pattern and are thus easy to remember, and share the same passwords among different accounts [12]. Unsurprisingly, simple passwords can be easily discovered by brute-force attack with a well-defined dictionary of passwords (called dictionary attack); shared password could be stolen from the account sever with the weakest security protection. Some Internet servers still store users' passwords in plaintext, instead of salted hash. It has witnessed that user password database of some of such servers have been stolen [15,2] by inside or outside attackers. Even if some system mandatorily requires user to setup a complex password, the user might write the complex password on a piece of paper strip and attach the paper strip with his/her computer, where this paper strip is not well-protected.

How to keep and manage multiple passwords securely and conveniently is an interesting and important real-world problem. Due to ubiquitous usage of portable digital device (e.g. smart phone, tablet, PDA) people have developed software (or mobile apps) to manage passwords in digital device. Unfortunately, mobile devices (including Android [4,1,16,3], jailbroken [9]/non-jailbroken [7] iOS [11] and other mobile operating systems) are suffering from stringent threat of spyware, which could log each user input (keystroke or touch points, etc) and even monitor the phone hardware (including screen, RAM, CPU, etc). For an average user, it is hard to tell whether his/her mobile device has been affected by spyware, even with anti-virus software [3].

Based on the above considerations, our goal is to design a simple method that allows users to manage passwords securely and easily in a mobile device, where the mobile device is assumed to be monitored by a spyware.

1.3. Our Contribution

Our main contributions in this work can be summarized below:

1. We propose a simple visual cryptography scheme, which allows users to decrypt a ciphertext *manually*. We also analyze the security of the proposed scheme, against adversary with unbounded computation power.
2. Based on the proposed visual cryptography scheme, we give a simple method to manage passwords in a mobile phone, where the confidentiality of stored passwords retain, even if the phone screen is monitored by possible spyware. Furthermore, we give a method to manage passwords among a large organization and implement access control to cyber or physical resource in large legacy system using passwords.

2. Related Works

2.1. Mobile Password Management

Nowadays, with the wide spread adoption of mobile devices, there is an increasing trend to manage passwords in mobile devices (e.g. smart phone) using a mobile app. Some

[3]Anti-virus's capability is limited, especially for newly emerged malware. In addition, a spyware could disguise itself as an anti-virus app in some loosely controlled app market.

examples found in Google Play Store are: "Keeper Password & Data Vault" [8], "eWallet Password Manager" [6], and "mSecure Password Manager" [13]. However, these password management apps are designed mainly focusing more on convenience than security, and suffer from at least these security issues: (1) Users have to fully trust the password management app itself in privacy of their password: users have no way to prevent the password management app from leaking their passwords to other apps installed in the same device, or from sending their passwords to some server via Internet connection. Even if the source code of password management app is available (i.e. open source program), not every user will invest time or is capable to audit the source code. (2) Still suffer from spyware, since the password will eventually display in the phone screen in some form (e.g. in typed form or handwriting form or fuzzy picture form like CAPCHA [4]), when users want to retrieve the password from the password management app.

In addition, Bojinov *et al.* [24] proposed a method to protect password database on a mobile device from attackers who may have physical access to the mobile device (e.g. stolen phones). Florêncio, Herley and Oorschot studied [27] how to group accounts and passwords for re-use, to achieve balance between security and convenience.

2.2. Visual Cryptography

Visual cryptography, which supports decryption using non-digital mechanical operation, was proposed by Naor and Shamir[29] in 1990's, and after that many subsequent works [26,20,23,28] appear. Most of these works exploit the secret sharing notion proposed by Shamir [30].

3. Our Proposed Visual Cryptography Scheme

In this section, we propose a probabilistic visual encryption scheme (KEYGEN, ENCRYPT, DECRYPT), which allows users to decrypt ciphertext manually. In addition, we also propose an algorithm CIPHERTEXTGEN that generates a random ciphertext. We will analyze the security of the proposed visual cryptography scheme.

3.1. Algorithms Description

In the basic scheme, a plaintext[5] is a string of ℓ symbols from alphabet Σ, a ciphertext is a matrix of $\texttt{NRow}$ number of rows and $\texttt{NCol}$ number of columns, and an encryption/decryption key is a list of ℓ tuples. All of $\texttt{NRow}, \texttt{NCol}$, and ℓ are public system parameters. A typical setting could be $\texttt{NRow} = 10, \texttt{NCol} = 8, \ell = 8$. In real applications, users could be allowed to choose values for these system parameters. Note that, unlike system parameters $\texttt{NRow}$ and $\texttt{NCol}$ which are constant across different ciphertexts, choice of alphabet to encrypt each password could be determined by authentication server (what symbols are allowed to form passwords), and different ciphertexts may have different alphabets.

[4] https://en.wikipedia.org/wiki/CAPTCHA

[5] Later in Section 4, we will discuss how to support plaintexts of different length.

3.1.1. Key Generation

The probabilistic key generation algorithm takes the system parameter $(\texttt{NRow}, \texttt{NCol}, \ell)$ as input, and outputs an encryption/decryption key K, which is an ordered list of ℓ randomly generated tuples (x_i, y_i, f_i). Here (x_i, y_i) is a coordinate within the grid $[0,\ \texttt{NRow}-1] \times [0,\ \texttt{NCol}-1]$, and $f_i \in \{0,1\}$ is a boolean flag. The detailed algorithm is as below.

```
1: procedure KEYGEN($\texttt{NRow}, \texttt{NCol}, \ell$)
2:     Randomly choose $\ell$ distinct tuple $(x_i, y_i)$'s from $[0,\ \texttt{NCol}-1] \times [0,\ \texttt{NCol}-1]$
3:     for $i$ from 1 upto $\ell$ do
4:         $f_i \leftarrow_R \{0,1\}$
5:     return $K := \{(x_i, y_i, f_i)\}_{i=0}^{\ell-1}$
```

The generated encryption/decryption key K can be represented *visually* as in Figure 1(a) (on page).

3.1.2. Encryption

The probabilistic encryption algorithm takes an encryption key K, an alphabet Σ, a plaintext $M \in \Sigma^\ell$, and system parameters $(\texttt{NRow}, \texttt{NCol}, \ell)$ as input. The output (i.e. the ciphertext) of the encryption algorithm is a matrix T of dimension NRow by NCol, where each cell is filled with a symbol from the alphabet Σ.

```
1: procedure ENCRYPT($K, \Sigma, M, \texttt{NRow}, \texttt{NCol}, \ell$)
2:     Create an empty matrix $T$ with dimension NRow by NCol
3:     for $i$ from 0 upto $\ell - 1$ do
4:         Parse $K[i]$ as $(x_i, y_i, f_i)$
5:         $m_i \leftarrow M[i]$
6:         if $f_i$ equals 1 and $m_i$ is a letter then
7:             $m_i \leftarrow \texttt{toggleUpperLowerCase}(m_i)$
8:         $T[x_i][y_i] \leftarrow m_i$
9:     $q \leftarrow \texttt{floor}(\texttt{NRow} \times \texttt{NCol}/|\Sigma|)$
10:    $r \leftarrow \texttt{NRow} \times \texttt{NCol}$ modulo $|\Sigma|$
11:    Randomly choose a subset $S$ of size $r$ from $\Sigma$
12:    for $i$ from 1 upto $q$ do
13:        $\Sigma_i \leftarrow \Sigma$
14:    Multiset $W \leftarrow \Sigma_1 \cup \Sigma_2 \cup \ldots \Sigma_q \cup S$
15:    Initiate set $Z \leftarrow \emptyset$
16:    for each symbol $a$ in $M$ do
17:        if $a \in W$ then
18:            $W \leftarrow W \setminus \{a\}$   /* Remove $a$ from multiset $W$ */
19:        else
20:            $Z \leftarrow Z \cup \{a\}$
21:    Randomly choose $|Z|$ elements from $W$ and replace them by $Z$
22:    Fill all elements in multiset $W$ to all empty cells of $T$
23:    Randomly permutate all cells in $T$, excepted locations specified by key $K$
24:    return $T$
```

In our application, the encryption algorithm runs in a digital computing device, e.g. mobile phone. An example of ciphertext is visually represented in Figure 1(b).

3.1.3. Decryption

The deterministic decryption algorithm takes a decryption key K, a ciphertext T, and system parameters $\texttt{NRow}, \texttt{NCol}, \ell$ as input. The output is a string of ℓ symbols from the alphabet Σ, where Σ is specified in encryption process.

```
1: procedure DECRYPT($K, T, \texttt{NRow}, \texttt{NCol}, \ell$)
2:     for $i$ from 0 upto $\ell - 1$ do
3:         Parse $K[i]$ as $(x_i, y_i, f_i)$
4:         $m_i \leftarrow T[x_i][y_i]$
5:         if $f_i$ equals 1 and $m_i$ is a letter then
6:             $m_i \leftarrow \texttt{toggleUpperLowerCase}(m_i)$
7:     return $m_0 m_1 \ldots m_{\ell-1}$
```

This decryption process is just to locate cells in the ciphertext matrix specified by decryption key, retrieve symbols in these cells, and flip the case of some letter symbol if condition meets. Such a simple procedure can be done mechanically or even manually, without any digital computing device. Figure 1(c) illustrate how to perform this decryption operation manually.

3.1.4. Random Ciphertext Generation

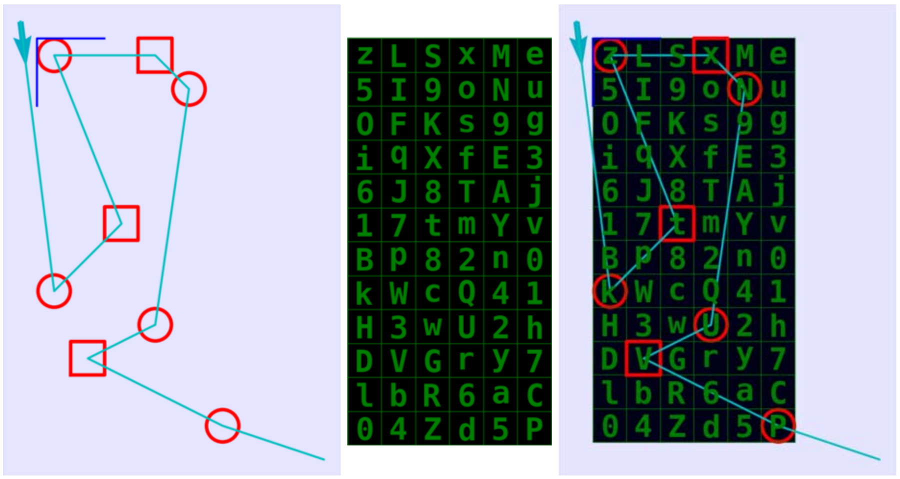
In addition, we also propose an extra algorithm CIPHERTEXTGEN, which takes as input an alphabet Σ and system parameters $\texttt{NRow}, \texttt{NCol}$, and outputs a random matrix T with cells filled by symbols from alphabet Σ.

```
1: procedure CIPHERTEXTGEN($\Sigma, \texttt{NRow}, \texttt{NCol}$)
2:     $q \leftarrow \texttt{floor}(\texttt{NRow} \times \texttt{NCol}/|\Sigma|)$
3:     $r \leftarrow \texttt{NRow} \times \texttt{NCol} \mod |\Sigma|$
4:     if $q$ equals 0 then
5:         Abort
6:     Randomly choose a subset $S$ of size $r$ from $\Sigma$
7:     for $i$ from 1 upto $q$ do
8:         $\Sigma_i \leftarrow \Sigma$
9:     Multiset $W \leftarrow \Sigma_1 \cup \Sigma_2 \cup \ldots \Sigma_q \cup S$
10:    Create an empty matrix $T$ with dimension $\texttt{NRow}$ by $\texttt{NCol}$
11:    Fill all elements of multiset $W$ into cells of matrix $T$
12:    Random permutate all cells of matrix $T$
13:    return $T$
```

3.1.5. How to Save the Decryption Key

Users may make a physical decryption key token by DIY. Here we give two examples: (1) Draw the key shown in Figure 1(a) on a piece of hard transparent plastic card. (2) Dig square or circle holes according to the key on a piece of hard paper (e.g. a name card), and draw a directed curve to connect all holes. To decrypt a ciphertext matrix displayed in a phone screen, users just put the physical decryption key token over the phone screen and align them well, then users can figure out the hidden password in mind quickly.

The unit cost of a key token is low, and can be further reduced if a lot of such physical tokens are produced in factory. The advantage of such physical decryption key



z	L	S	x	M	e
5	I	9	o	N	u
0	F	K	s	9	g
i	q	X	f	E	3
6	J	8	T	A	j
1	7	t	m	Y	v
B	p	8	2	n	0
k	W	c	Q	4	1
H	3	w	U	2	h
D	V	G	r	y	7
l	b	R	6	a	C
0	4	Z	d	5	P

(a) Encryption/Decryption Key. (b) Ciphertext (c) Visual Decryption

Figure 1. Illustration of Visual Decryption: : Align the blue color right-angle at the left-top corner of the decryption key in Figure 1(a) with the left-top corner of the ciphertext matrix in Figure 1(b), and read out message "kTzXNUvP" in Figure 1(c), where the case of letters selected by square is toggled and unchanged if selected by circle.

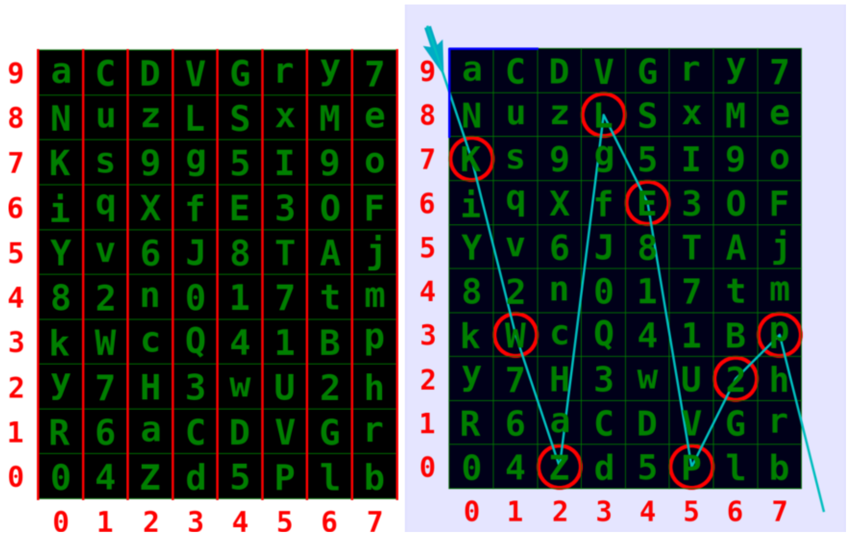
token is that users require to do a little mental effort. The drawback is that it could be lost or stolen.

3.1.6. *How to Memorize the Decryption Key*

Alternatively, users could remember the decryption key in their brains. Here we provide some suggestions to help users memorize the decryption keys. The advantage of this option is that no extra physical token is required and thus more convenient, but the entropy of the decryption key (arguably) might reduce, and we should make sure the remaining entropy will be still sufficient to protect passwords.

In order to easily remember the decryption key in human brain, one can choose the decryption key with a certain pattern. Some examples are as below:

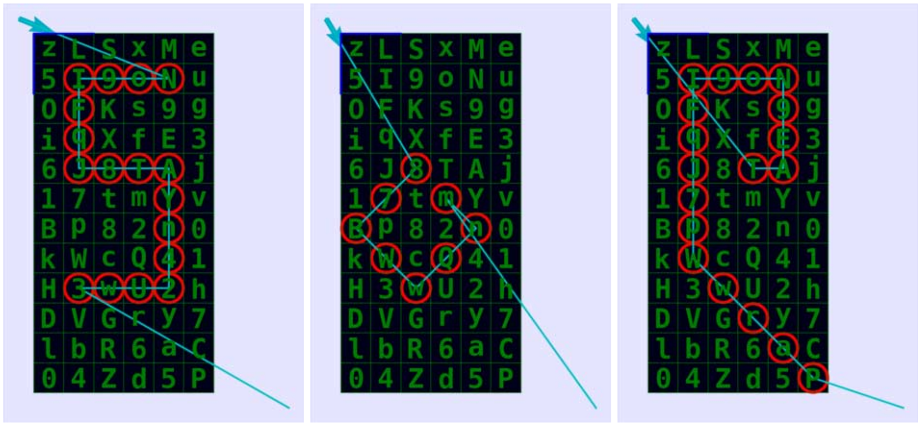
1. Encode key as numbers: Let $\texttt{NRow} = 10$, i.e. the number of rows in ciphertext matrix is 10. We can encode a subclass of decryption keys with digits. Any ℓ digital number $x_0, x_1, \ldots, x_i, \ldots, x_{\ell-1}$, each x_i in the range $[0,9]$, encode a decryption key $(0,x_0),(1,x_1),\ldots,(i \mod \texttt{NCol},\ x_i,),\ \ldots,\ (\ell-1 \mod \texttt{NCol},\ x_{\ell-1})$, ordered from left to right, where all $f_i = 0$. An example is showed in Figure 2.
2. Graphical Pattern: We may conceptually walk inside the ciphertext matrix according to some graphical pattern. For example, Figure 3(a) shows shape of digital "5"; Figure 3(b) shows shape of a diamond; Figure 3(c) shows shape of a flip version of digit "9". Similarly, one can also walk in the ciphertext matrix according to an English letter, or a character from any other language, or any shape he/she likes. To remember such decryption key, the users need only remember some anchor point positions. Note that in Figure 3, the key length may be very large. Users may take a substring of the long key as the actual decryption key. E.g. (1) substring of symbols at odd positions; (2)Deterministically derive a small



(a) A Ciphertext

(b) Visual representation of key encoded by digits "73086023"

Figure 2. Example of decryption key that can encodes to numbers. The labels along X-axis and Y-axis will help users to locate cells manually. Thick vertical lines in Figure 2(a) will mark columns obvious.



(a) 5

(b) Diamond

(c) Flip 9

Figure 3. Example of Decryption Keys with Simple Graphic Pattern

integer v, say $v \in [0,4]$, from the ciphertext label (or auxiliary) information, and let the suffix starting at position v of the long key be the actual decryption key.

It is worthy to point out that, every time the user wants to retrieve any plaintext (e.g. passwords), he/she should recall his/her decryption key mentally, in order to perform the decryption operation manually. The more frequent exercises of such manual decryption, the better that users could memorize the decryption key. Some previous works [25,21,22] showed that it is possible to remember a high entropy secret by a human brain, with a certain method (e.g. rehearsing the passwords in proper time period).

3.2. Security Analysis of Proposed Visual Cryptography Scheme

Let $\mathbf{H}(\cdot)$ denote Shannon's entropy function, and $\mathbf{H}(\cdot|\cdot)$ denote the conditional entropy. Let $\mathbf{K}, \mathbf{M}, \mathbf{T}$ denote the random variables for key, plaintext, and ciphertext, respectively. In this subsection, if not otherwise specified, we assume the encryption/decryption key $\mathbf{K}$ is chosen by probabilistic algorithm KEYGEN, the ciphertext $\mathbf{T}$ is chosen by probabilistic algorithm CIPHERTEXTGEN, and the plaintext $\mathbf{M}$ is derived deterministically from decryption key $\mathbf{K}$ and ciphertext $\mathbf{T}$ using deterministic algorithm DECRYPT. This is very different from traditional security setting of encryption scheme, where plaintext is chosen by user according to some distribution and ciphertext is (probabilistically) derived from encryption key and plaintext. The root cause of the difference is that, our encryption scheme will be applied to a special message — password, which is supposed to be random without any semantics.

We will quantify the entropy of decryption key $\mathbf{K}$ in various attack mode. High entropy decryption key $\mathbf{K}$ and random ciphertext $\mathbf{T}$, will guarantee that the message $\mathbf{M}$ (i.e. password to be protected) will have high entropy and thus secure.

Lemma 1. $\mathbf{H}(\mathbf{K}) \geq \ell \log \ell + \log \binom{\mathtt{NRow} \times \mathtt{NCol}}{\ell}$.

Lemma 2 (Ciphertext-Only Attack I)**.** $\mathbf{H}(\mathbf{K} \mid \{\mathbf{T}_i\}_i) = \mathbf{H}(\mathbf{K})$.

Lemma 3 (Ciphertext-Only Attack II)**.**
$\mathbf{H}(\mathbf{K},\ \{\mathbf{M}_i\}_{i=1}^n \mid \{\mathbf{T}_i\}_{i=1}^n) = \mathbf{H}(\mathbf{K})$*, where* $\mathbf{M}_i = \text{DECRYPT}(\mathbf{K}, \mathbf{T}_i, \mathtt{NRow}, \mathtt{NCol}, \ell)$

Lemma 4 (Known-Plaintext Attack)**.** *Suppose* $\mathtt{NRow} \times \mathtt{NCol} \geq |\Sigma| \times q$ *for some integer* q*. Suppose* $\mathbf{M} \in \Sigma^\ell$ *consists of* n_j *number of symbols* $s_j \in \Sigma$*,* $j \in [0, v]$*, where* $\sum_{j=0}^{v} n_j = \ell$ *and all symbols* s_j*'s are distinct. We have* $\mathbf{H}(\mathbf{K} \mid (\mathbf{T}, \mathbf{M})) \geq \log\left(\prod_{j=0}^{v} \binom{q}{n_j} \times n_j!\right)$.

Particularly, if plaintext $\mathbf{M}$ *consists of* ℓ *distinct symbols, we have* $\mathbf{H}(\mathbf{K} \mid (\mathbf{T}, \mathbf{M})) \geq \ell \log q$*. Additionally, if the plaintext* $\mathbf{M}$ *consists of* ℓ *distinct English letters, then* $\mathbf{H}(\mathbf{K} \mid (\mathbf{T}, \mathbf{M})) \geq \ell \log 2q$*.*

If the decryption key is encoded as ℓ numbers in $[0, 9]$ and $\mathtt{NRow} = 10$, the number of possible combinations for decryption keys is 10^ℓ , which could be sufficient (e.g. $\ell \geq 8$) to defend brute force attack, if some delay mechanism is adopted in login process, although it is smaller than the total number of all possible decryption keys. For graphical pattern key, the key space could be even larger. Theoretical and empirical analysis of space size of graphical pattern key can be found in existing works [31],[32].

In summary, if the plaintext is uniformly random distributed over its alphabet set, and the adversary can only do a limited amount (e.g. smaller than 10^6) of brute force search, then our proposed visual cryptography scheme is secure. We remark that if the plaintext is not uniformly distributed, the proposed visual cryptography scheme could be broken by frequency analysis, similar like traditional substitution cipher.

4. Managing Passwords using Visual Cryptography

In previous section, we proposed a secure visual cryptography scheme for uniformly random plaintexts. An ideal application of this scheme is to encrypt passwords: because that (1) a secure password has to be randomly generated; (2) typically, adversary is not allowed [6] to do a large number of brute force attack (e.g. passwords with 6 digits of $\log 10^6$ bits entropy could be sufficiently secure for some authentication system).

Let Σ be the alphabet for passwords, and a password is a string of ℓ symbols from Σ. We generate an encryption/decryption key and apply the encryption algorithm ENCRYPT described in previous section, to encrypt the password, and store the ciphertext in a mobile phone or print it on a piece of paper. Alternatively, we could generate a random ciphertext using algorithm CIPHERTEXTGEN and then obtain a password by decrypting the ciphertext with decryption key using algorithm DECRYPT.

4.1. Visually Encrypt Passwords

4.1.1. Short Passwords

In case that the length of a password s is smaller than ℓ, we can append a random padding suffix (randomly chosen from $\Sigma^{\ell-|s|}$) to the password, so that the resulting string has length ℓ. Then we can apply the above method to encrypt this padded password to obtain a random matrix T as ciphertext. Meanwhile the length $|s|$ of real password is recorded in plaintext. During decryption, the user just ignore the last $(\ell - |s|)$ symbols in the output of DECRYPT algorithm.

4.1.2. Long Passwords

If the length of a password s is larger than ℓ, we can divide it into blocks of length ℓ, where possibly the last (partial) block may be expanded to a full bock using random padding. Then the user can encrypt each block of length ℓ using the proposed visual encryption scheme, to get a random matrix for each block. Similar as in previous case of short password, the length $|s|$ of actual password will be recorded in plaintext.

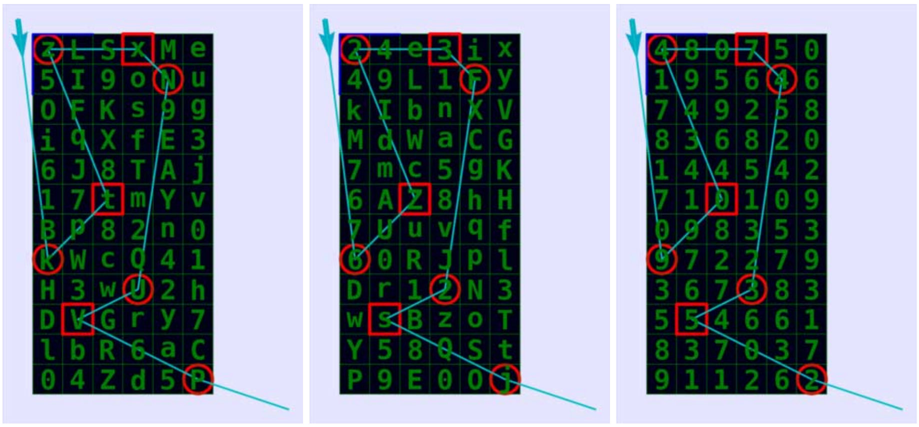
4.1.3. Multiple Passwords

We can encrypt all passwords separately using the same encryption key. Each ciphertext will be labeled with the account information (e.g. room number, device id, or domain name and id), to distinguish with each other. Figure 4 shows that one key encrypts three passwords with different alphabets.

4.1.4. New Passwords Generation

To generate a new password from alphabet Σ with length ℓ, we simply generate a random ciphertext by revoking CIPHERTEXTGEN(Σ, `NRow`, `NCol`), and visually decrypt it using the decryption algorithm DECRYPT and decryption key. The decrypted message will be the password from space Σ^{ℓ}. Shorter or longer passwords can be generated using similar ideas as above.

[6] Many authentication systems have mechanisms to essentially slow down online or even offline brute-force attack on passwords.



(a) “kTzXNUvP” (b) “6z23F2Sj” (c) “90474352”

Figure 4. Illustration of visual decryption with one key and three distinct ciphertexts, where in the first two ciphertexts, the alphabet consists of English letters and digits, and in the third ciphertext, the alphabet consists of only digits. Each sub-figure is labeled with the corresponding decrypted message.

In case that the authentication system has more restriction on passwords, e.g. every password has to contain digits, upper case letters and lower case letters, we can keep trying to generate new random passwords, until we find one that meets the requirement.

There are two approaches to generate and save a new password: (1) Generate a password using some other method and then encrypt it by running ENCRYPT algorithm using a mobile phone app, then save the ciphertext. (2) Run CIPHERTEXTGEN algorithm to generate a ciphertext using a mobile phone app, and then *manually* and *mentally* decrypt the ciphertext to get the new password on the demand. The second method will be more secure in the sense that the mobile phone app is fully ignorant to the new password, while in the first method, the mobile app has full knowledge of the new password. Note that it is not convenient to perform our encryption method manually.

4.2. Management Passwords on a Mobile Phone

We implement a mobile app in android phone, to execute our visual cryptography scheme and manage passwords. Any user can manage his/her passwords in digital devices by following the below steps:

1. User chooses and remembers a master password to lock the mobile app and all ciphertexts (this master password can be replaced by fingerprint, if fingerprint authentication is supported in the device. Such master password or fingerprint will protect users’ stored passwords, in case that both the mobile device and the physical decryption key token are stolen by attackers).
2. User randomly chooses an encryption/decryption key and keeps it private.
3. For each account, User generates a new random password by running the CIPHERTEXTGEN algorithm with the mobile app and label the newly generated ciphertext with the account information. Then User re-sets password for this account in corresponding authentication system.
4. To retrieve a designated password for a particular account,

(a) User unlocks the mobile app by providing the master password (or fingerprint);
(b) User searches the corresponding ciphertext in the digital device;
(c) Decrypt the ciphertext to get password manually.
(d) User manually types the password to the login interface on the login device, where the login device is supposed to be different from the device that stores and manages passwords.
About Step (c) and (d), we emphasize that, decrypting a ciphertext of password and typing a password into the login device could be in parallel and in one symbol by one symbol manner. That is, a user could manually decrypt the first symbol of password and type this symbol into the login device, and then proceed for the second symbol, and then for other symbols one by one.

5. Lock the mobile app and remove all unprotected version of ciphertexts from the digital device memory.

We emphasize that, very different from Google's Android graphical screen lock, in our method, **users are not supposed to touch symbols** in the ciphertext matrix, which is displayed **in the phone screen**.

4.2.1. Scope of Applications

We recommend that our method should be used in the following way:

- (**Condition 1**) the login device is different from the mobile device which manages passwords using our method. E.g. passwords of bank ATM card, safe box, combination lock, computer server root password, etc.
- (**Condition 2**) authentication servers of all passwords to be managed should have a comparable level of security protection. For example, ordinary website (e.g. some online forum, Facebook, twitter) has arguably weaker security protection on users' password than large bank corporation.

The reason behind condition 1 is that: If a mobile phone is affected by spyware, and our method is used to manage passwords of website (e.g. Facebook), the password could be captured by spyware, when a user types the password to the website login interface using the same mobile phone, although storage of passwords is secure.

The reason behind condition 2 is that: Our method has a certain level (but still limited) resistance to known-plaintext attack, if multiple plaintext-ciphertext pairs are given. Some websites may still store all users' passwords in plaintext in server side for some reason, and these passwords might be stolen by hackers and revealed in Internet. If the spyware in mobile phone has access to such leaked password database, it might be able to launch known-plaintext attack on our scheme. In physical world, no centralized authentication server is required by combination lock, which is favorite to our scheme.

We recommend users to classify all passwords in categories, according to the security levels of authentication servers and value of assets protected by each password, and protect each category of passwords using a separate visual cryptography key. For example, choose a random visual cryptography key K_0 to manage passwords of high value, e.g. ATM passwords, safe-box passwords, computer server root passwords, using our proposed method, and never use the master key K_0 to protect passwords of normal websites (e.g. Facebook, twitter, online forum).

4.2.2. Comparison

It is easy to see that, our mobile password management solution based on our visual cryptography method, is more secure than existing mobile password management apps (e.g. [8,6,13]), in the sense that, our solution can defeat spyware, and in other solutions, spyware can obtain all protected passwords by monitoring the phone screen. In any other security aspect, our solution is at least equally good as others.

4.3. Our Experiment

We implement our method in android phone and conduct survey with 7 subjects (teenage students). With little training (time for briefing and training is within 5 minutes), it takes 45.3 seconds on average for these 7 subjects to successfully login to a randomly chosen online account among a list of 100, using our mobile app, where decryption key of our visual encryption scheme is remembered in subject's brain. Here searching time, manual decryption time and login time are all included. The shortest time record is 30 seconds and the longest one is 64 seconds. In the future, we will also experiment on opening a combination lock.

4.4. Alternative Visual Cryptography for Password Management

4.4.1. Off The Grid

Steve Gibson proposed a paper based system (called as "Off The Grid") for encrypting[7] a domain name into a random looking password using Latin Square [10], where the resulting password can be used for this domain exclusively. More details of this method are given in our full version [33].

4.4.2. Comparison with Our Method

In this "Off The Grid" method, the paper with printed Latin Square plays the role of secret key and should be kept securely, the domain names (after possible substitution from non-alphabet symbols to alphabet symbols) are plaintexts, and the resulting passwords are ciphertexts. We compare this method with our method in Table 1

5. Access Control in Legacy System using Passwords

In cyber access control and authentication system, passwords are widely used to protect each computer, server and online resource. Especially, legacy systems may rely more on passwords for cyber access control, compared with newly designed system which may prefer smart card and biometric authentication (e.g. fingerprint and iris). For physical access control, some legacy system may still reply on keyed lock (e.g. padlock) to protect each room and important device. We suggest such legacy system to switch from keyed lock to combination lock, and securely and conveniently manage all passwords, for both cyber and physical access control, using our visual cryptography method in mobile phones.

[7]The author describes it as "encryption". But according to the algorithm, it is more like a hash instead of encryption method, since decryption algorithm is not required, and not provided either.

Table 1. Comparison between Our Scheme and Off The Grid Method

Our Scheme	**Off The Grid**
Probabilistic encryption. The ciphertext is a matrix of symbols.	Deterministic encryption (actually hash). The key (i.e. Latin Square) is a matrix of symbols.
Users manually decrypt a ciphertext to obtain password	Users manually encrypt/hash an input message to obtain password
Account id is labeled with the ciphertext of password	Account id is the input message to the encrypt/hash method
Dimension is flexible (Suitable for both phone screen or printed on paper)	26 by 26 Grid (Difficult to see if shown in phone screen for manual encrypting/hashing)
Encryption key and alphabet are independent (Example of one key, three ciphertexts and two distinct alphabets is given in Figure 4)	Encryption key binds to a fixed alphabet. Additional patches required to support digits and special symbols in input and/or output . The resulting output is a repeation of pattern "letter letter non-letter" (`https://www.grc.com/otg/enhancements.htm`)
Passwords remain secure if matrix of symbols is leaked	All passwords revealed if Latin Square is leaked

5.1. *Current access control in Legacy system*

In some legacy system, keyed lock is widely adopted to protect access to rooms or important devices. Some staff has to carry a lot of keys during work time, and find a correct one to open a lock to a door or device. As a physical token, keys have some inherent disadvantages: (1) it is relatively expensive to revoke or update a key; (2) all management (e.g. distributing/labeling/searching/storing/backuping keys) of keys are performed manually and are thus troublesome; (3) it is troublesome for a staff to carry a lot of keys. Some organization might share the same key for different rooms/devices, in order to reduce the number of keys and simplify key management, at the cost of security. For a large legacy system, the number of keys could be thousands or more.

5.2. *Upgrade to Password based System*

In newly designed modern systems, smart card (e.g. contactless card) based door solution is widely adopted.

Smart card solution requires persistent electronic power supply and networks. This could be the reason that it is only widely applied to protect rooms, instead of devices. Some legacy system may spread in a large area, and their existing power supply grid and computer network may not cover every room to be protected. Despite the great security and convenience that smart card based door access solution offers, the upgrading cost might be too high for such legacy system with the traditional keyed locks, considering the overall cost including power grid and network construction.

For such legacy system, we propose an alternative mitigation solution: (1) switch keyed-lock to password based combination lock (e.g. single dial combination lock); (2) end users store and manage passwords on their own mobile phones using our visual encryption method; (3) build a trusted central server, to distribute, synchronized, update, revoke all passwords, among all users.

The trusted central server could communicate with users mobile phones in a secure communication channel using cell phone network (e.g. 2G(SMS)/3G/4G), or wifi in a small area. When users move to location beyond the coverage of cell phone network of wifi, users can manage passwords locally without network connection.

The total cost of our solution will be limited: (1) Optionally, users bring their own phone to work, which means zero additional cost for end device; (2) In case that cell phone network available, just pay the additional communication data usage and no additional cost to construct the network facility; in case of wifi, the cost of wifi router is cheap; (3) Setup a central server.

6. Conclusion

We proposed a visual cryptography scheme to encrypt random messages and applied it to protect and manage passwords in digital devices. Our method protects confidentiality of passwords even if in an extreme case that the digital device is monitored by spyware. In other words, our method can protect the confidentiality of passwords, without trusting the digital device and without trusting the password management app which implements our method. Our method provides two alternative configurations: (1) higher security with a dedicated non-digital physical token (e.g. made with a piece of paper or plastic card); (2) sufficient security without any extra physical token. Our method protects the storage confidentiality of passwords, and can increase the overall security level, if the login device is different from the digital device that stores ciphertexts of passwords. Thus, our method is very suitable to manage combination lock passwords, safe box passwords, ATM passwords, computer login password, server root password, and so on. Furthermore, our method is easy to understand (so easy to audit security of our proposal) and easy to use, and the deployment cost is cheap.

In future work, we may design more sophisticated physical decryption token (say optical based) to provide a much larger key space.

References

[1] 16 million mobile devices infected with malware, http://www.esecurityplanet.com/mobile-security/16-million-mobile-devices-infected-with-malware.html

[2] 42 million unencrypted passwords leaked, http://www.computerworld.com/article/2475534/cybercrime-hacking/42-million-unencrypted-passwords-leaked-from-hacked-online-dating-site-cupid-medi.html

[3] android malware ios-apps pose greater risk leaking user data, http://tech.firstpost.com/news-analysis/android-malware-ios-apps-pose-greater-risk-leaking-user-data-says-study-219746.html

[4] android malware rises 300 percent, http://www.scmagazine.com/android-malware-rises-300-percent-report-says/article/390903/

[5] Copy your house keys with your phone, https://keysduplicated.com/

[6] eWallet Password Manager, https://play.google.com/store/apps/details?id=org.awallet.free&hl=en

[7] ios devices dont have to be jailbroken for spyware, http://www.scmagazine.com/ios-devices-dont-have-to-be-jailbroken-for-spyware-sold-by-hacking-team-to-be-installed/article/426137/

[8] Keeper Password and Data Vault, https://play.google.com/store/apps/details?id=com.callpod.android_apps.keeper&hl=en

[9] keyraider malware stole over 225000 apple credentials from jailbroken ios devices, http://www.computerworld.com/article/2978152/cybercrime-hacking/keyraider-malware-stole-over-225-000-apple-credentials-from-jailbroken-ios-devices.html
[10] Latin Square, http://mathworld.wolfram.com/LatinSquare.html
[11] Malware for iOS, https://www.theiphonewiki.com/wiki/Malware_for_iOS
[12] Measuring password re-use empirically, https://www.lightbluetouchpaper.org/2011/02/09/measuring-password-re-use-empirically/
[13] mSecure Password Manager, https://play.google.com/store/apps/details?id=com.mseven.msecure&hl=en
[14] Software can clone keys from single photo, http://www.telegraph.co.uk/news/newstopics/howaboutthat/3375872/Software-can-clone-keys-from-single-photo.html
[15] sony hacked yet again plaintext passwords posted, http://arstechnica.com/tech-policy/2011/06/sony-hacked-yet-again-plaintext-passwords-posted/
[16] spyware pre-installed on certain xiaomi, huawei and lenovo smartphones sold by third party, http://www.digit.in/mobile-phones/spyware-pre-installed-on-certain-xiaomi-huawei-and-lenovo-smartphones-27144.html
[17] the 25 most popular passwords of 2014, http://gizmodo.com/the-25-most-popular-passwords-of-2014-were-all-doomed-1680596951
[18] US-Cyber Emergency Response Readiness Team: CyberSecurity Tips. (Choosing and Protecting Passwords), http://www.us-cert.gov/cas/tips/
[19] worst passwords 2013, http://splashdata.com/press/worstpasswords2013.htm
[20] Ateniese, G., Blundo, C., De Santis, A., Stinson, D.R.: Visual cryptography for general access structures. Information and Computation 129(2), 86–106 (Sep 1996)
[21] Blocki, J., Blum, M., Datta, A.: Naturally Rehearsing Passwords. CoRR abs/1302.5122 (2013), http://arxiv.org/abs/1302.5122
[22] Blocki, J., Komanduri, S., Cranor, L.F., Datta, A.: Spaced Repetition and Mnemonics Enable Recall of Multiple Strong Passwords. CoRR abs/1410.1490 (2014), http://arxiv.org/abs/1410.1490
[23] Blundo, C., De Santis, A., Naor, M.: Visual Cryptography for Grey Level Images. Information Processing Letters 75(6), 255–259 (Nov 2000)
[24] Bojinov, H., Bursztein, E., Boyen, X., Boneh, D.: Kamouflage: Loss-resistant Password Management. In: Proceedings of the 15th European Conference on Research in Computer Security. pp. 286–302. ESORICS'10 (2010)
[25] Bonneau, J., Schechter, S.: Towards Reliable Storage of 56-bit Secrets in Human Memory. In: 23rd USENIX Security Symposium (USENIX Security 14). pp. 607–623 (2014)
[26] Droste, S.: New Results on Visual Cryptography. In: CRYPTO 96: Advances in Cryptology. pp. 401–415
[27] Florêncio, D., Herley, C., van Oorschot, P.C.: Password Portfolios and the Finite-Effort User: Sustainably Managing Large Numbers of Accounts. In: 23rd USENIX Security Symposium (USENIX Security 14). pp. 575–590 (2014)
[28] Hou, Y.C.: Visual cryptography for color images. Pattern Recognition 36(7), 1619–1629 (2003)
[29] Naor, M., Shamir, A.: Visual Cryptography. In: EUROCRYPT '94: Advances in Cryptology. pp. 1–12
[30] Shamir, A.: How to Share a Secret. Commun. ACM 22(11), 612–613 (Nov 1979)
[31] Tao, H., Adams, C.: Pass-Go: A Proposal to Improve the Usability of Graphical Passwords 7(2), 273–292 (Sep 2008)
[32] Uellenbeck, S., Dürmuth, M., Wolf, C., Holz, T.: Quantifying the Security of Graphical Passwords: The Case of Android Unlock Patterns. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer; Communications Security. pp. 161–172. CCS '13 (2013)
[33] Xu, J., Zhou, J., Lu, L.: Cyber and physical access control in legacy system using passwords. Cryptology ePrint Archive, Report 2015/1161 (2015), http://eprint.iacr.org/

Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016
A. Mathur and A. Roychoudhury (Eds.)

doi:10.3233/978-1-61499-617-0-43

Data Driven Physical Modelling For Intrusion Detection In Cyber Physical Systems

Khurum Nazir JUNEJO [a,c,1], David YAU [a,b]
[a] *Singapore University of Technology and Design, Singapore*
[b] *Advanced Digital Sciences Center, Illinois at Singapore*
[c] *Karachi Institute of Economics and Technology, Pakistan*

Abstract. Cyber physical systems are critical to the infrastructure of a country. They are becoming more vulnerable to cyber attacks due to their use of off the shelf servers and industrial network protocols. Availability on World Wide Web for monitoring and reporting, has further aggravated their risk of being attacked. Once an attacker breaches the network security, he can affect the operations of the system which may even lead to a catastrophe. Mathematical and formal models try to detect the departure of the system from its expected behaviour but are difficult to build, and are sensitive to noise. Furthermore they take a lot of time to detect the attack. We here propose a behaviour based machine learning intrusion detection approach that quickly detects attacks at the physical process layer. We validate our result on a complete replicate of the physical and control components of a real modern water treatment facility. Our approach is fast, scalable, robust to noise, and exhibits a low false positive (FP) rate with high precision and recall. The model can be easily updated to match the changing behaviour of the system and environment.

Keywords. Cyber Physical Systems Security, Machine Learning, Intrusion Detection, Fault Detection

1. Introduction

Cyber Physical Systems (CPSs) are geographically dispersed, large-scale, life-critical, and expensive systems. Smart grids, water filtration and distribution, unmanned vehicles, and pervasive health care systems are some of the examples of such systems. They comprise of physical components such as actuators, sensors, and cyber components such as network components and commodity serves. These network can be wired or wireless, and run off the shelf industrial network protocols. Furthermore they maybe available on the World Wide Web for remote monitoring and reporting. Thus leaving them prone to attacks from not only criminals, hacktivists, or disgruntled employees, but also from enemy states, such as the attack on Iran's nuclear centrifuge [5]. A hacker in US infected a water filtering plant with a software that could alter the plants water treatment operations

[1]Corresponding Author: Singapore University of Technology and Design, 8 Somapah Rd, Singapore 487372; E-mail: khurum_junejo@sutd.ed.sg.

[4]. A disgruntled ex-employee took control of Australia's Maroochy Water Services and released one million litres of untreated sewage into local parks and rivers for many days [22]. Thus intrusion detection systems (IDS) are needed to secure these systems.

Traditional IDS systems try to detect intrusion in the network traffic layer only. They do so by either a dictionary of known attacks or by modelling the normal behaviour of the traffic through Machine Learning (ML) techniques. The former requires the cumbersome task of keeping the dictionary of attacks upto date, and are totally blind to zero day attacks. The later suffer from missed detections and high false alarms. Once the adversary has breached the traffic layer, the system is completely at his mercy. An IDS system is needed at the physical layer to serve as a last line of defence. Current systems tackle this problem by deploying a mathematical or some formal model of the physical process that simulates the system and classifies deviation from the expected output as an attack or an abnormal behaviour. These techniques are referred to as behaviour-specification approaches. They require intricate understanding of the physical process, physical laws that govern this process, and the equipment specifications. They are costly to come up with, are sensitive to noise, do not update to the change of the environment or the ageing of the plant, and lastly the behaviour of the physical process may not exactly follow the specification in operational manual.

Data driven (aka behaviour based) approaches do not suffer from these problems. They do not look for something specific and nor do they require an expert to build a formal model, instead they learn the behaviour of the system through historical data. They are fast, easier to develop, and are robust to noise and can adapt to the drift in the normal behaviour of the system with time and change in weather. However these approaches are very rare in the literature because they suffer from the lack of availability of training data under an attack scenario. So they depend on simulations to generate their results. These results may not be valid on real operational systems. We on the hand use the data from Secure Water Treatment (SWaT) testbed, a complete replicate of the physical and control components of a real modern water treatment facility to learn our model and validate results.

Since behaviour based approaches learn directly from the data, they are not blind to the operational behaviour of the CPS that is sometimes different than the vendor specification. This peculiar behaviour is missed by the simulations and specification based approaches. As an example, in SWaT, the closing and opening of a valve is not immediate, as specified in the operational manual. Instead, the valve is in a transient state for a dozen of seconds until it opens completely, followed by a similar delay in the water inflow to reach its desired rate as specified in the manual. This rate of inflow at times can be even greater than 11% as specified in the specification, operating under normal conditions. These phenomenon are brushed under sensor noise tag in a behaviour-specification approach. Sensor noise itself being a nuisance that needs separate modelling, as it even sometime shows a flow of water even when the valve is closed. Behaviour based approaches do not need to model these artefacts separately. Furthermore they can fit a more tighter bound around the operational behaviour of the system e.g. we observe that under the normal behaviour the level of the tank does not go above a little over 900 litres, whereas a behaviour-specification based approach would consider an upper bound of 1100 litres as stated in the specification and encoded into the PLC coding.

In this paper, in contrast to the dictionary, and behaviour-specification based approaches, we model the physical process of SWaT through a data driven approach to detect attacks at the physical layer. Contrary to their usual critique, we show that these approaches do not suffer from a high FP rate, in fact, few of the best performing techniques generate no or very few FP with high precision and recall. In particular, we evaluate and compare the performance of nine supervised machine learning (ML) classifiers on data generated by SWaT injected with eighteen attacks of ten different types. We further compare the time to detect an attack, type of attacks detected by each approach, and time to build the model. We demonstrate that these ML approaches not only successfully detect almost all the attacks, but also detect them earlier than the behaviour-specification approaches.

The rest of the paper is organised as follows: Section 2 describes the SWaT testbed and the process that we are trying to model, followed by related work in Section 3. We describe the nine classifiers in Section 4. Dataset details and results are presented in Section 5, followed by conclusion in Section 6.

2. SWaT Testbed

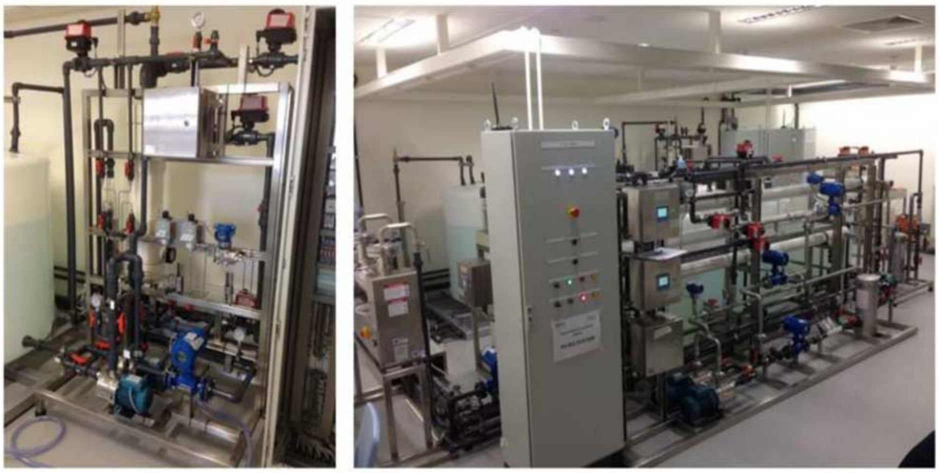


Figure 1. A View of The SWaT Testbed.

Secure Water Treatment (SWaT) is a testbed that is jointly setup by the Ministry of Defence, National Research Foundation, Singapore and Singapore University of Technology and Design (SUTD). Its purpose is to enable experimentally validated research in the design of secure and safe CPS. It replicates the physical and control components of a real modern water treatment facility, see figure 1. SWaT consists of the following six-stage filtration process:

P1 Supply and storage
P2 Pre-treatment
P3 Ultrafiltration and backwash
P4 De-Chlorination System

P5 Reverse Osmosis (RO)

P6 RO Permeate Transfer, UF Backwash and Cleaning

The process begins by taking in raw water (P1), adding necessary chemicals to it (P2), followed by filtration via an Ultra-filtration (UF) system (P3), de-chlorinating it using UV lamps (P4), and then feeding it to a Reverse Osmosis (RO) system (P5). A backwash process (P6) cleans the membranes in UF using the water produced by RO, and transfers the clean water back to the raw water tank. All the process are interdependent, with each process dependent on the previous one. The cyber portion of SWaT consists of a layered communications network, Programmable Logic Controllers (PLCs), Human Machine Interfaces (HMIs), Supervisory Control and Data Acquisition (SCADA) workstation, and a Historian. Data from sensors is available to the SCADA system and recorded by the Historian for subsequent analysis. A separate PLC is dedicated for each of the six processes. Each of these PLCs is provided with a redundant hot-standby PLC.

The communication in the SWaT takes place over multi-layer communication links, consisting of different switches and routers. This communication can take place trough either a WiFi, or an Ethernet link running various industrial communication protocols. The sensor readings, and actuator commands are all communicated with the PLC over these (wired or wireless) communication links. These protocols suffer from network attacks. In our lab we have successfully injected Man In The Middle (MITM) attacks, which can inject false sensor readings, and commands.

2.1. Pre-Treatment Process

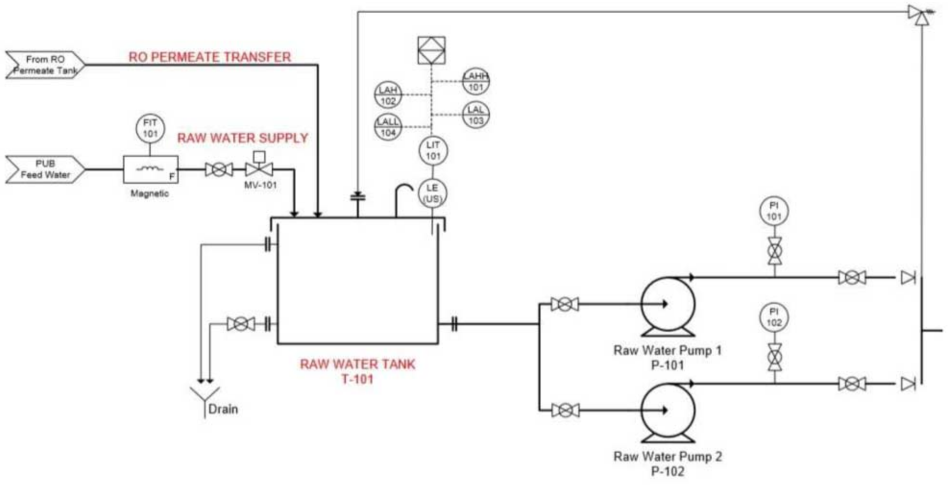


Figure 2. Details of the first stage of SWaT.

SWaT is a complex system, with six different processes interacting with each other. In this paper we try to model the intrusions in the process P1 only. Findings for this process will be extended to rest of the processes by either extending the current model to incorporate one subsequent process at a time, or by learning a separate model for each of the process.

Process control of the P1 is illustrated in figure 2. The water flows into the raw water tank from two sources, city water supply system, and RO process (P6) after cleaning. The water from the water supply system is controlled by motor valve MV-101. It is opened when the water level goes below a predefined Low (L) threshold, and is closed when it reaches a predefined High (H) threshold. If for some reason, the water level still keeps on decreasing (or increasing) past the L (or H) threshold, then alarms are set off when it reaches the Low Low (or High High) threshold, signalling an emergency situation in which the system can be thought of as insecure. The rate of this inflow is measured by the flow indicator FIT-101. The water inflow from the RO process can not be controlled by P1, therefore at a given point in time water could be flowing into the tank from none, one, or both of the sources. Pump P-101 is turned on when the water level drops below L threshold in the UF tank (P3), and is turned off when water level rises above the H threshold in the UF tank. It can also be turned off when the level in the raw water tank drops below L, or the flow indicator FIT-201 (in process P2) drops below a certain predefined threshold. Pump P-102 is only for back up, it only operates when P-101 fails to do so.

Even though process P1 is not the most financially critical process of the testbed, but all the later process depend on it. A false sensor value for LIT-101 can overflow the tank, decrease the output of the plant, or burn out the pumps. In section 5 we define ten different attacks that are injected into the system.

3. Related Work

In literature, IDS have been categorized into three categories; knowledge based, behaviour-specification based, and behaviour based [14]. Knowledge based approaches look for specific runtime patterns that match a pre-maintained dictionary of pattern of known attack types [8]. These approaches are fast and have low FP rate, but require the dictionary of attacks to be kept upto date. They are totally blind to zero day attacks. Behaviour-specification approaches model the normal behaviour of the system by formally defining the legitimate behaviour, and signal an attack when the system departs from this model [15,10]. These approaches require a human expert to define this proper system behaviour, which is very time consuming. Secondly the systems behaviour may change because of environmental changes, or with ageing, thus departing from the behaviour that was previously thought as normal. Behaviour based approaches address these issues but suffer from high FP rate, but our study shows otherwise. Some behaviour based approaches use conventional statistics based techniques [25], they test if a sensor reading or an actuator setting is within some number of standard deviations of a mean. These approaches are parametric, not fully automatic, and difficult to come up for large number of interdependent sensor and actuators. We therefore limit our focus to non parametric approaches such as data mining and ML approaches.

Most of IDS model the network traffic [26,2]. Only few try to detect attacks at physical layer, and these approaches are behaviour-specification approaches [4]. Out of the forty IDS systems reviewed, only twenty four were behaviour based. More than half of these IDS detect intrusions in non CPS systems, using internet and LAN traffic [24,11]. Other approaches try to detect attacks in the network traffic layers of CPS that use of the shelf industrial network protocols such as MODBUS, DNP3, Ethernet/IP, etc. [13]. We were not able to find an IDS that models only the physical process of a CPS using a ML approach. The closest work that we could find was done on the water storage testbed of Mississippi State University [16]. It is a simple testbed consisting a 2 litre water storage tank, a level sensor, and a pump. [7] use NN to detect attacks on this testbed with basically one network traffic attribute and one physical process attribute, namely, number of responses against a command, and water level. The only physical attacks that they simulate are the change in water level. [17] uses the same testbed but adopts a one class approach. They add network traffic features to increase the attribute set, thus rendering their approach to mostly a network traffic IDS. Whereas we use a two class approach due to the availability of attack data on SWaT, and we inject ten different attacks in the physical process. Lastly, our testbed is much more complex and realistic which validates our findings more.

4. Methodology

A data driven approach for CPS should be fast, scalable, and robust. It should be fast because the system operates in real time, it wouldn't be beneficial to signal an attack after it has happened. It should be scalable because CPS can be huge systems involving multiple readings per second from hundreds of sensors and actuators. It should be robust to sensor noise, and to the effects of weather and ageing. Therefore we choose to model this behaviour with the state of the art supervised Machine Learning (ML) techniques. Supervised techniques require training data under both the normal operating circumstances, and when under attack. It is rare to find the data when system is under attack, because these expensive and infrastructure critical plants are not available to researchers to carry out attacks. We on the other hand can generate the data under the attack scenario from the testbed. Even then, IDS problem is a skewed class problem. In all, we injected ten different types of attacks in the system. These attacks henceforth referred to as A1, A2,.., A10 are described below:

A1 FIT-101 is attacked and its value is changed to zero.
A2 MV-101 is attacked and is turned off.
A3 P-102 is attacked and is opened.
A4 P-101 is attacked and is closed down.
A5 MV-101 and P601 are closed down.
A6 FIT-101 is attacked by showing it operating value above its normal operating range.
A7 LIT-101 is attacked by increasing its value above the normal maximum level.
A8 LIT-101 is attacked by decreasing its value below the normal minimum level.
A9 LIT-101 is attacked by increasing its value more than normal rate of change.
A10 LIT-101 is attacked by decreasing its value more than normal rate of change.

The nine state of the art ML classifiers that we choose are fast and scalable, but some what sensitive to noise. We tackle this by reserving some part of the data for testing over which we tune the parameters of the algorithms to find the best performance parameters.

We choose three most widely used discriminative classifiers, namely, Support Vector Machines (SVM), Neural Network (NN), and Logistic Regression (LR). Three classifiers are decision tree based, namely, Random Forest (RF), J48, and Best First Tree. The remaining three are statistical classifiers, namely, Naive Bayes (NB), Bayes Network (BayesNet), and Bayes Logistic Regression (BayesLR). We briefly discuss each of these classifiers below.

4.1. Discriminative Approaches

Support Vector Machines (SVM) are non-probabilistic binary linear classifiers. They project the data in a higher dimensional feature space where a hyperplane is learned to discriminate between the points of the two classes. The hyperplane is such that it maximizes the margin between the closest points of the two classes, thus achieving a better generalization over the unseen data. This has led SVM to exhibit high performance on most of the real world classification problems including IDS [1].

Artificial Neural Networks (NNs) are a family of models inspired by biological neural networks and are used to approximate functions that can depend on a large number of inputs. In addition to the input and output layer, they consist of one or more hidden layer of neurons that try to learn non linear decision boundaries separating the classes. Along with SVM's, NN belongs to few of the most successful algorithms for intrusion detection [2].

Logistic Regression (LR) is an alternative to linear discriminant analysis, and can be seen as analogous to linear regression. However it is based on quite different assumptions. Firstly, the conditional distribution is a Bernoulli distribution rather than a Gaussian distribution. Secondly, the predicted values are probabilities through a logistic function. It nicely measures the relationship between the categorical dependent variable and one or more independent variables. It can outperform SVM and NN on problems with large number of features, it has been less successful on intrusion detection problems though [23].

4.2. Decision Tree Based Approaches

Decision tree have been a popular approach to ML since early nineties, mainly because they can be translated to simple IF-ELSE, OR and AND rules which are intuitive and easily understandable. They have been successfully applied to detect intrusion detection in network traffic layer [20,11] in recent years. A decision tree can be visualized as a tree structure that consist of nodes from root to leaf, in which each internal node denotes a test on an attribute, each branch represents the outcome of a test, and each leaf node represents a class label. A new record is traversed through the root node till the leaf, where it is assigned the label of the leaf node. The selection of attributes is based on some information theoretic measure. J48 uses Information Gain as the measure to select attributes at each node of the tree [19]. The tree is pruned to avoid overfitting. Best First Tree (BFTree) is similar to J48 [21], except that the best node is expanded first instead of the depth-first order. This approach allows new ways to prune the tree which make it less susceptible to overfitting.

Random forests (RF) are an ensemble approach proposed by [3]. They are based on notion of bootstrapping and selection of random subset of features i.e. they operate by constructing a multitude of decision trees at training time on a random subset of the training examples. At each candidate split in the learning process, a random subset of the features is selected. The mode of the classes of the individual trees is the predicted class label for a record. RF have been recently perform very well on many problems. RF have been shown to be more robust to overfitting than other decision tree algorithms.

4.3. Bayesian Approaches

Bayesian classifiers are one of the most popular approaches in pattern recognition. They are probabilistic classifiers that predict class by means of probabilities, such as the probability that a given sample belongs to a particular class or not. Bayesian Networks (BayesNet) and Naive Bayes (NB) variants are the two most fundamental methods that belong to these class of classifiers. They have been successfully used for intrusion detection as well [12,24]. NB classifiers assume conditional independence given the value of the class. i.e. the attributes do not influence each other given the value of the class attribute. NB classifiers are highly scalable, requiring a number of parameters linear in the number of features. They have been shown to perform nicely on large dimensional problems, and in scenarios where part of the data is unobservable. They can be learned through Maximum-likelihood method that does not require multiple iterations over the training data. BayesNet are probabilistic graphical model that represents a set of random variables and their conditional dependencies via a directed acyclic graph [6]. Unlike NB, dependencies can be laid out between the variables by adding an edge between them. Since attributes are not completely independent in the real world, they tend to outperform NB classifier when number of features are less. BayesLR is a Bayesian approach to LR that uses a Laplace prior to avoid overfitting, and is shown to generate better results than LR especially for large feature spaces [9]. Bayesian classifiers are commonly used classification algorithms because of their simplicity, computational efficiency and good performance for real-world problems.

4.4. Optimal Parameters

In this subsection we report the parameters for which the classifiers give the best performance. For SVM, we tried out several parameters, including the kernel and the regularization parameter C. Performance for linear kernel did not exceed 95% despite choosing large values of C. SVM gave best performance using Radial Basis Function (RBF) kernel with $C = 8000$. NN gave best results with seven hidden units in the hidden layer, two units in the output layer, learning rate, momentum, and number of iterations equal to 0.3, 0.2, and 1000, respectively. LR worked best for the default parameters. RF and J48, gave the best performance for 200 trees, and confidence factor of 0.6, respectively. BFTree worked best without pruning, and minimum number of objects set at 4. NB does not have parameters, while BayesLR worked best for the default parameter values. For BayesNet, we used Simple Estimator, and TAN as the search algorithm with Entropy as the Score Type. Most of the algorithms have a few more parameters for which we tried out different values, but we do not mention them here because their best value turned out to be the default values.

	No. of Attacks	Attack	Normal	Total
Training Data	18	1260	18900	20160
Test Data	18	580	8060	8640

Table 1. Details of the dataset.

5. Dataset and Results

5.1. Dataset

Each second, the state of the sensors and actuators is recorded into a Historian database. For this study, we extracted eight hours of data from the database which totals around 28800 records. Since we are using supervised ML approaches, we divided this data into two sets, training and test set. We learn the model on the training set and then evaluate its performance on the test set. We tune the parameters of each algorithm on this test set to obtain their best performance. The details of these sets are given in table 1.

Since the probability of attack in real life is small, so in order to simulate a real scenario we have injected only few attacks into the system. We have injected 10 different attacks based on the sensor and actuator values. The first eight attacks have two instances each, while each of the last two attacks have one instance only. In the training set the duration of the first and second attack is kept at 30 and 120 seconds, respectively. In reality, the duration of attacks in the test set may be different from that of the training data, therefore we also keep it different in test data. In test data, the first attack lasts 10 seconds while the second attack lasts only 30 seconds. This totals to 18 attacks in the training and test set, corresponding to 1260 and 580 rows, respectively.

An important phenomenon observed in the data that is absent in the specification is that the turning on and off of valves and pumps is not immediate. It can take upto a dozen seconds for the valve to open, during which the state of the pump is neither open nor closed, instead its in a transient state. Furthermore, the water inflow takes similar amount of time to reach its normal or desired rate of flow. This makes the problem of detecting the attacks more challenging because now the valve is not closed and yet water is not pouring in. This problem is further aggravated by the sensor noise, e.g. the valve is closed but the value of the water inflow is greater than zero. Similarly the output pump is closed but the level of the tank still decreases. These issues can result in the degradation of the performance of the IDS, but we choose not to model them explicitly. We do this for two reasons, it would further complicate the model, and secondly since no modelling is perfect, therefore it may add some new artefact in the model. Instead we try to tackle these issues directly in the learning processing through regularization.

5.2. Metrics

Being a skewed class problem, accuracy is not a sufficient metric to measure the measure the performance of the IDS. Precision and recall are more suitable measures for this class of problems. Precision is basically a measure of how accurate the classifier is when it says that it has detected an attack. A higher value corresponds to a lesser number of false alarms (FP). Whereas, recall is basically how many of the attacks did the classifier actually detect. Higher value of Recall corresponds to a lesser number of missed attacks (FN). Ideally we want a classifier with high precision and recall because that corresponds to lower FP and FN. F-Measure helps us to combine both into a single metric. It is the harmonic mean of precision and recall. It is a more conservative measure than the arithmetic mean of the two. The three measures are mathematically defined below:

$$Precision = \frac{TP}{TP+FP}$$

$$Recall = \frac{TP}{TP+FN}$$

$$F-Measure = \frac{2*Precision*Recall}{Precision+Recall}$$

where TP is the number of attacks correctly detected by the classifier. Fifth measure that we use is the Area Under the Receiver Operating Characteristic (ROC) curve (AUC). It is considered to be a more robust measure than F-Measure for highly skewed problems [18]. ROC curve is true positive rate (TPR) plotted against false positive rate (FPR) thresholded at various settings. This allows to select possibly optimal models by varying this threshold.

5.3. Pre-Processing

We treat every attribute as a numeric attribute, even the states of valve and pumps. Reason being that they are not a binary variables as they take on three values, where value 2 corresponds to the on state, 1 for the off state, and 0 for the transient state. We normalize all the attributes in the interval between zero and one. This is very important for certain algorithms e.g. normalization helped reduce SVM's training time from hours to just a few seconds, in some cases SVM also crashed on the un-normalize data. Lastly we introduced a new attribute in the data set as change in tank level, hoping that it would help us detect attacks that incur sudden changes in tank level. We will see in results that, it helps us to detect the sudden increase in the tank level, but not the sudden decrease in tank level. We calculate this attribute by subtracting the old reading from the current reading of the tank level.

5.4. Results

The results demonstrate the effectiveness of ML based algorithms to detect attacks in the physical process of a CPS. We performed various runs of each algorithm to find their parameters that would give their best performance on the test dataset. We summarize these results in table 2. All the five performance measures are reported as percentage values. The last column, Time To Build the Model (TTBM) is reported in seconds. The highest value for each metric is highlighted in bold. The results carry a bit of a surprise. In this particular problem when number of features is significantly smaller than the number of training examples, LR, and SVM with linear kernel should perform best. To the contrary LR and SVM with a linear kernel are one of the worst performers. Instead SVM with a RBF kernel takes the top spot in the accuracy section. NN on the other hand was supposed to perform equally better as SVM.

Coming to the tree based algorithms, rarely used BFTree performance is quite high, sharing the top spot with RF on three measures, and being second in the accuracy mea-

	Accurracy	Precision	Recall	F-Measure	AUC	TTBM
SVM	**99.83**	99.65	97.76	98.69	98.38	6.78
NN	99.08	99.10	99.10	99.10	96.30	54.23
LR	94.52	93.90	94.50	93.10	90.00	0.38
RF	99.78	**99.80**	**99.80**	**99.80**	99.60	10.95
J48	98.91	99.00	98.90	98.90	98.50	0.8
BFTree	99.78	**99.80**	**99.80**	**99.80**	98.30	0.15
NB	94.13	94.50	94.10	91.90	84.70	**0.03**
BayesNet	99.07	99.10	99.10	99.10	**99.70**	0.09
BayesLR	94.09	94.40	94.10	91.90	56.00	0.38

Table 2. Performance of Classifiers. All values are in percentages, except TTBM which is in seconds.

sure. Nonetheless, its AUC is significantly lower than that of RF, which demonstrates the befitting discriminative power of the RF.

Strikingly, BayesNet has the highest AUC. This means that its decision function can be tweaked to achieve better performance than rest of the algorithms. Its training time is also the lowest amongst the high performing classifiers.

It is interesting to note that in attack A7 to A10 the bounds are calculated using the normal behaviour of the system and not from the specification manual of the system. This is important, and allows us to fix more tighter bounds on the operation of the system, e.g. the specifications state that the level of the tank cannot be more than 1100 litres, so an alarm goes off if this so happens, but from the data we observe that the level of the tank never goes above 903.04 litres because MV-101 is turned off when it reaches 900 litres. Using this insight our model learns a value little more than 903.04 as an abnormal behaviour and signals it as an attack. Same is the case for the minimum level of the tank. Deriving these upper and lower bounds from data enables us to detect an attack way before it reaches the critical stage. On the contrary, for A6, the data helps us to loosen the bound on operational range of FIT-101, which otherwise would have resulted into an false alarm. Reason being that according to data, FIT-101 can operate at a 11% higher value than that stated in the specification.

5.5. *Fault Identification*

We further drill down into the results, and analyse which classifier failed to detect which type of attacks. Table 3 summarizes the type of attacks detected by each classifier, we discard the classifier with accuracy lower than 95% from this analysis. None of the classifier has been successful in detecting all of the attacks. The most difficult to detect was attack A10. RF and BFTree that seemed to perform exactly same also differ on detecting A10. In A10 attacker significantly reduces the level of the tank but still within its upper and lower bounds, hence being classified as normal behaviour by most of the classifiers. A9 is quite similar to A10, the level is increased significantly as opposed to decreasing it as in case of A10, rest of the conditions are same. A reason for its better detection could be that the change in water level is negative in this case, still it needs further analysis. Second most difficult to detect attack is A4, completely missed by NN, thus costing it position in the top rankings.

In the test data, the attack lasts from 10 to 30 seconds, so it could be that an attack is not detected for its whole duration but for a few seconds only. We therefore further

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
SVM	2	2	2	2	2	2	2	2	1	0
NN	2	2	2	0	2	2	2	2	1	1
RF	2	2	2	2	2	2	2	2	1	0
J48	2	1	2	2	2	2	2	2	1	0
BFTRee	2	2	2	2	2	2	2	2	1	1
BayesNet	2	2	2	1	2	2	2	2	0	1

Table 3. Number of Attacks Detected. Two instance of each attack upto A8. Only once instance for A9 and A10.

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
SVM	100	95.71	100	100	100	100	100	100	100	0
NN	100	100	100	0	100	100	100	100	100	10
RF	100	87.14	100	100	100	100	100	100	100	0
J48	100	85.71	100	100	100	100	100	100	100	0
BFTree	100	87.14	100	100	100	100	100	100	100	10
BayesNet	100	100	97.14	85.71	100	100	100	97.14	0	10

Table 4. Percentage of Records Detected For Each Attack.

analyse this by looking at the percentage of records detected by each classifier in table 4. This table shows a slightly different picture than 3. It shows that despite getting detected, attack A2 has given almost all the classifiers a tough time. SVM, RF, BFTree, and J48 miss some part of this attack. For attack A10, the three algorithms that eventually detected it, were only able to detect only 10% of it.

This raises a question; how long does it take to detect an attack? Delay in detecting an attack can lead to a catastrophe. We try to answer this question through table 5. A value of 0 means that the attack was detected the first second that it started, whereas ∞ means that the attack went undetected. Most of the attacks get detected at the very instant they start, or do not get detected at all. Only three attacks are detected at the fourth second. This is a good feature of ML based approaches in contrast to the specification based approaches where the model waits for the output of the system to deviate from the normal behaviour to be able to detect them.

These results demonstrate the effectiveness of using ML algorithms for modelling of the physical layer. An ensemble approach of complementing algorithms to detect all the attacks can be used to further improve performance, a direction for future exploration.

5.6. Training Time

Attack detection should be in real time, so it is necessary to train the model and classify the incoming examples as quick as possible. We report Time To Build the Model (TTBM) in table 2. Given the number of training examples, most of the algorithms were quick to learn the model. As expected, NN is the slowest, but it classifies new examples very fast. So does SVM, it took less then half a second to classify all the testing examples. TTBM should be taken with a grain of salt because it is affected by other processes running on the system. The experiments were run on a 3 GHZ Core i7 laptop with 8 GB RAM. The implementations were not parallelised, so the running times are for algorithms running on a single core. Memory footprint for each of the algorithm was also quite

	A1	**A2**	**A3**	**A4**	**A5**	**A6**	**A7**	**A8**	**A9**	**A10**
SVM	0	0	0	0	0	0	0	0	0	∞
	0	0	0	0	0	0	0	0		
NN	0	0	0	∞	0	0	0	0	0	0
	0	0	0	∞	0	0	0	0		
RF	0	3	0	0	0	0	0	0	0	∞
	0	0	0	0	0	0	0	0		
J48	0	∞	0	0	0	0	0	0	0	∞
	0	0	0	0	0	0	0	0		
BFTree	0	3	0	0	0	0	0	0	0	3
	0	0	0	0	0	0	0	0		
BayesNet	0	0	0	∞	0	0	0	0	∞	0
	0	0	0	0	0	0	0	0		

Table 5. Time (in seconds) to Detect an Attack.

small. Low memory footprint and low running time indicate that these approaches are highly scalable, and would be appropriate for detecting intrusion in all of the six stages of the SWaT testbed.

6. Conclusion

CPS are critical to the infrastructure and economy of a country. Securing them from cyber attacks that may be carried out by criminals, hacktivists, disgruntled employees, and enemy states is a high priority area for many governments. Even though many intrusion detection system (IDS) exist for the network layer, no further defences exist ones it has been breached. We therefore propose a behaviour based machine learning (ML) approach for intrusion detection that models the physical process of the CPS to detect any anomalous behaviour or attack that may try to change the behaviour of the CPS. We successfully demonstrate the effectiveness of such approaches by validating results of nine ML approaches on Secure Water and Treatment (SWaT) testbed, a complete replicate of the physical and control components of a real modern water treatment facility. We show that ML approaches are not only fast, robust, but detect attacks very early while exhibiting no or very little false positives. Furthermore our approach can be used in conjunction with the existing network traffic IDS systems.

Acknowledgment

This work was supported by the National Research Foundation (NRF), Prime Ministers Office, Singapore, under its National Cybersecurity R&D Programme (Award No. NRF2014NCR-NCR001-40) and administered by the National Cybersecurity R&D Directorate.

References

[1] Iftikhar Ahmad, Muhammad Hussain, Abdullah Alghamdi, and Abdulhameed Alelaiwi. Enhancing svm performance in intrusion detection using optimal feature subset selection based on genetic principal components. *Neural Computing and Applications*, 24(7-8):1671–1682, 2014.

[2] Omar Al-Jarrah and Ahmad Arafat. Network intrusion detection system using neural network classification of attack behavior. *Journal of Advances in Information Technology Vol*, 6(1), 2015.

[3] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[4] Alvaro A Cárdenas, Saurabh Amin, Zong-Syun Lin, Yu-Lun Huang, Chi-Yen Huang, and Shankar Sastry. Attacks against process control systems: risk assessment, detection, and response. In *Proceedings of the 6th ACM symposium on information, computer and communications security*, pages 355–366. ACM, 2011.

[5] Thomas M Chen and Saeed Abu-Nimeh. Lessons from stuxnet. *Computer*, 44(4):91–93, 2011.

[6] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. *Machine learning*, 29(2-3):131–163, 1997.

[7] Wei Gao, Thomas Morris, Bradley Reaves, and Drew Richey. On scada control system command and response injection and intrusion detection. In *Proceedings of the 5th Annual Anti-Phishing Working Group eCrime Researchers Summit (eCrime)*, pages 1–9. IEEE, 2010.

[8] Wei Gao and Thomas H Morris. On cyber attacks and signature based intrusion detection for modbus based industrial control systems. *Journal of Digital Forensics, Security and Law*, 9(1):37–56, 2014.

[9] Alexander Genkin, David D Lewis, and David Madigan. Large-scale bayesian logistic regression for text categorization. *Technometrics*, 49(3):291–304, 2007.

[10] Dina Hadžiosmanović, Robin Sommer, Emmanuele Zambon, and Pieter H Hartel. Through the eye of the plc: semantic security monitoring for industrial processes. In *Proceedings of the 30th Annual Computer Security Applications Conference*, pages 126–135. ACM, 2014.

[11] Md Al Mehedi Hasan, Mohammed Nasser, Biprodip Pal, and Shamim Ahmad. Support vector machine and random forest modeling for intrusion detection systems. *Journal of Intelligent Learning Systems and Applications*, 2014, 2014.

[12] Levent Koc, Thomas A Mazzuchi, and Shahram Sarkani. A network intrusion detection system based on a hidden naïve bayes multiclass classifier. *Expert Systems with Applications*, 39(18):13492–13500, 2012.

[13] Hui Lin, Adam Slagell, Catello Di Martino, Zbigniew Kalbarczyk, and Ravishankar K Iyer. Adapting bro into scada: building a specification-based intrusion detection system for the dnp3 protocol. In *Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop*, page 5. ACM, 2013.

[14] Robert Mitchell and Ing-Ray Chen. A survey of intrusion detection techniques for cyber-physical systems. *ACM Computing Surveys (CSUR)*, 46(4):55, 2014.

[15] Robert Mitchell and Ing-Ray Chen. Behavior rule specification-based intrusion detection for safety critical medical cyber physical systems. *Dependable and Secure Computing, IEEE Transactions on*, 12(1):16–30, 2015.

[16] Thomas Morris, Anurag Srivastava, Bradley Reaves, Wei Gao, Kalyan Pavurapu, and Ram Reddi. A control system testbed to validate critical infrastructure protection concepts. *International Journal of Critical Infrastructure Protection*, 4(2):88–103, 2011.

[17] Patric Nader, Paul Honeine, and Pierre Beauseroy. Mahalanobis-based one-class classification. In *Machine Learning for Signal Processing (MLSP), 2014 IEEE International Workshop on*, pages 1–6, 2014.

[18] David Martin Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. 2011.

[19] J. Ross Quinlan. Improved use of continuous attributes in c4. 5. *Journal of artificial intelligence research*, pages 77–90, 1996.

[20] Shailendra Sahu and BM Mehtre. Network intrusion detection system using j48 decision tree. In *Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on*, pages 2023–2026. IEEE, 2015.

[21] Haijian Shi. *Best-first decision tree learning*. PhD thesis, Citeseer, 2007.

[22] Jill Slay and Michael Miller. *Lessons learned from the maroochy water breach*. Springer, 2008.

[23] Chih-Fong Tsai, Yu-Feng Hsu, Chia-Ying Lin, and Wei-Yang Lin. Intrusion detection by machine learning: A review. *Expert Systems with Applications*, 36(10):11994–12000, 2009.

[24] Liyuan Xiao, Yetian Chen, and Carl K Chang. Bayesian model averaging of bayesian network classifiers for intrusion detection. In *Computer Software and Applications Conference Workshops (COMPSACW), 2014 IEEE 38th International*, pages 128–133. IEEE, 2014.

[25] Nong Ye, Syed Masum Emran, Qiang Chen, and Sean Vilbert. Multivariate statistical analysis of audit trails for host-based intrusion detection. *Computers, IEEE Transactions on*, 51(7):810–820, 2002.

[26] Yichi Zhang, Lingfeng Wang, Weiqing Sun, Robert C Green, Mansoor Alam, et al. Distributed intrusion detection system in a multi-layer network architecture of smart grids. *Smart Grid, IEEE Transactions on*, 2(4):796–808, 2011.

Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016
A. Mathur and A. Roychoudhury (Eds.)

doi:10.3233/978-1-61499-617-0-59

Detecting Multi-Point Attacks in a Water Treatment System Using Intermittent Control Actions

Sridhar ADEPU and Aditya MATHUR[1]
sridhar_adepu@sutd.edu.sg, aditya_mathur@sutd.edu.sg

iTrust, Center for Research in Cyber Security,
Singapore University of Technology and Design
Singapore

Abstract.
A novel technique for detecting multi-point attacks on an Industrial Control System (ICS) is described. The technique, referred to as Intermittent Control Actions (ICA), sends control signals intermittently to selected components to monitor the system using a process invariant. ICA was assessed experimentally for its effectiveness in an operational water treatment testbed. The experiments revealed (a) multi-point attack scenarios where ICA succeeds or fails to detect an attack, (b) issues in the design of key ICS components to ensure that the control actions in ICA do not lead to undesirable process behavior, and (c) constraints on the design of the physical system for safe use of ICA.

Keywords. Attack detection, multi-point cyber attacks, cyber security, design parameters for resilience, Industrial Control Systems, Intermittent Control Actions, Secure Water Treatment testbed.

1. Introduction

An experimental investigation was undertaken with the long term goal of designing robust defense mechanisms for an Industrial Control System (ICS) to improve its resiliency. An ICS includes several components including Programmable Logic Controllers (PLCs), a Supervisory Control and Data Acquisition (SCADA) workstation, a Historian, and several physical components. In the remainder of this paper the experiments are referred to as IC-Exp. The objective in IC-Exp was to understand how to detect specific cyber attacks against an ICS using correlations (invariants) across data obtained locally by a Programmable Logic Controller

[1]This work was supported by research grant 9013102373 from the Ministry of Defense and This work was supported by the National Research Foundation (NRF), Prime Minister's Office, Singapore, under its National Cybersecurity R&D Programme (Award No. NRF2014NCR-NCR001-40) and administered by the National Cybersecurity R&D Directorate.. Corresponding author: Sridhar Adepu.

(PLC) and the use of intermittent control actions against a process actuator. The experiments were performed to understand the effectiveness of the proposed detection methods in an operational mini-water treatment testbed, referred to as SWaT (Secure Water Treatment), that produces 5 gallons/minute of treated water.

Cyber attack: In this work a "cyber attack" refers to an attempt at disturbing the state of an ICS through its communication network. In the experiments reported, cyber attacks were aimed at disturbing the state of SWaT with malicious intent, i.e., degrading system productivity measured in gallons of treated water produced.

Process invariant: A "process invariant" is a mathematical relationship among "physical" and/or "chemical" properties of the process controlled by the PLCs in an ICS. Together, at a given time instant, a suitable set of such properties capture the observable state of SWaT. The relationships include, for example, correlations between the level of water in a tank and the flow rate of incoming and outgoing water across this tank, and the relationship between water pH prior to and immediately after the ultrafiltration unit. In SWaT, these properties are measured using sensors during its operation and captured by the PLCs at predetermined time instants–which is every 0.1-second. The measurements are also saved in a historian for subsequent analysis. In this paper one invariant, derived from physical properties of water flow in stage 1 of SWaT (Section 2.1), is considered for detecting cyber attacks.

Single and multi-point attacks: An ICS contains several points of entry into the system that could be exploited by an attacker to realize an intent. For example, each wireless link that connects a sensor to the Programmable Logic Controller (PLC) is a potential entry point. Each PLC itself is a potential entry point when vulnerabilities exist in its firmware. The SCADA workstation in an ICS is also an entry point when vulnerabilities exist in the operating system it uses. A *single point attack* exploits exactly one entry point to gain access to an ICS. A *multi-point attack*, considered in this work, exploits more than one entry point , e.g., two or more communications links, to disturb the state of an ICS. Single-point attacks on SWaT have been studied extensively [1]. In the experiment described in this paper, the objective was to investigate how multi-point attacks can be detected using process invariants while simultaneously using intermittent control actions.

Research questions: RQ 1: How effective is the use of ICA in detecting multi-point attacks on a water treatment system? RQ 2: What is the complexity and scalability of the attack detection approach? RQ 3: What are the impacts of using ICA on the physical design of the water treatment system?

Related work: In a survey of attack detection techniques in Cyber Physical Systems (CPS) [7], the detection techniques are classified as follows: misuse/signature-based intrusion detection, anomaly-based intrusion detection, and statefull protocol analysis. The process invariants based approach proposed in this paper does not fall in any of these three categories. Further, as implemented in the case study described here, the proposed approach differs from those mentioned above in several ways including the fact that it does not use network-based anomaly detection.

A common aspect of the techniques described in the survey [7] is that they are employed in the network used for communications among the PLCs and the SCADA workstation. The primary goal of these techniques is to detect any intrusion into the ICS by analyzing network traffic from different points of view. For example, in [3] security specifications are defined for smart meters and a security policy for the Advanced Metering Infrastructure (AMI). The authors then use a formal specification of the C12.22 protocol for data exchange among meters and other devices. A formal verification of the specifications and monitoring operations is conducted. The prototype described by the authors uses an emulation of AMI and is based on an analysis of network traffic.

A process oriented, technique for the detection of cyber attacks in PLCs is described in [6]. It is noted that this technique does not fall in any of the three categories mentioned above. Data is collected from the PLCs via network monitoring, and an autoregressive method is used to model specific process variables. Such a model is then used to detect whether the value of a process variable is suspicious. Another technique uses the physics of the process controlled by the ICS to detect cyber attacks [9]. Here the authors analyze Modbus [11] traffic focusing on "harmful" command streams. Physical constraints are integrated into an intrusion detection framework. An example of a boiler is described where out of bound values are checked against predefined constraints.

Significant work exists in detecting anomalies in network traffic in ICS across PLCs, sensors, actuators and SCADA workstation. One such technique is based on CUSUM used for change point detection [13]. This non-parametric method has been applied to detect network intrusions [16]. Given as in [17] "....A key drawback of the CUSUM algorithm is that the intensity of the anomaly needs to be known a priori,.." parametric as well as spectral [2] methods have been developed to detect network traffic anomalies.

Identification of sensor manipulation, and its impact, have been studied. In [8] a simulated Tennessee Eastman Challenge process is used o study the effectiveness of entropy-based technique for detection manipulated sensor values. The authors use correlations across sensor reading to detect malicious readings;. In IC-Exp sensor masking is used where multi-point sensor attacks mask the effects of another compromised sensor. In IC-Exp, detection is based on physics-based invariants and not noise. The Tennessee Eastman process is also used in [4] to study multi-point attacks. A CUSUM-based technique was found effective. However, when applied on a robot [15], the effectiveness of the CUSUM-based technique was below that reported in [4]. While these techniques are found effective in the environments in which they were assessed, in IC-Exp it was decided to instead use only the process property based invariants to detect anomalies arising due to a cyber attack. Doing so avoids making assumptions on probability distributions of process data. Indeed, making use of invariants is perhaps appropriate when a real or simulated process is available for experimentation, and not necessarily when only data from such process is available as for example in [6,17]. A few key advantages of invariant-based attack detection, implemented in a PLC, are as follows.

1. *Implementation context*: Detection method is implemented as a procedure and integrated directly into the PLC. Thus, PLC-SCADA network traffic is not disturbed.
2. *Physical constraints*: These are local to a PLC and based on the physics of the sub-process, e.g., the chemical dosing process in a water treatment plant, being controlled by a PLC.
3. *Detection method*: Detection is agnostic to the type of an attack; it checks if the dynamics of the sub-process matches prediction based on its physics.

Contributions: (a) Invariant-based approach, combined with ICA, for attack detection in a specific ICS. (b) Dependence of the attack detection approach on the system state, and other design parameters, when an attack is launched. (c) Impact of attack detection study on the design of the software and hardware of a specific ICS.

Organization: The remainder of this paper follows the tradition IMRaD (Introduction, Method, Results and Discussion) approach for structuring a scientific article. Section 2 describes the method used and the experiments conducted. A brief introduction to SWaT is in this section. Results from the experiment appear in Section 3. Discussion on various aspects of attack detection in an ICS and design challenges, are in Section 4. Conclusions and plans for further experimentation are in Section 5.

2. Method

2.1. Context: The SWaT testbed

SWaT is a fully operational scaled down water treatment plant for research in the design of IC resilient to cyber attacks. In a small footprint producing 5 gallons/minute of doubly filtered water, this testbed mimics large modern plants for water treatment such as those found in cities. The testbed is available for investigating the response to cyber-attacks and for conducting experiments with novel designs of physics-based and other attack detection and defense mechanisms.

Water treatment process: The treatment process (Figure 1) in SWaT consists of six distinct and cooperating sub-processes P1 through P6. Each sub-process, referred to as a *stage*, is controlled by an independent PLC. Thus, six PLCs work in concert to control the entire process. Control actions are based on the system state estimated by the PLCs using data from sensors. Stage P1 controls the inflow of water to be treated, by opening or closing a motorized valve that connects the inlet pipe to the raw water tank. Water from the raw water tank is pumped via a chemical dosing station (stage P2) to another UF (Ultra Filtration) feed water tank in stage P3. A UF feed pump in P3 sends water via the UF unit to RO (Reverse Osmosis) feed water tank in stage P4. Here an RO feed pump sends water through an ultraviolet dechlorination unit controlled by a PLC in stage P4. In stage P5, the dechlorinated water is passed through a 2-stage RO filtration unit. The filtered water from the RO unit is stored in the permeate tank and the reject in the UF backwash tank. Stage P6 controls the cleaning of the membranes in the UF unit by turning on or off the UF backwash pump.

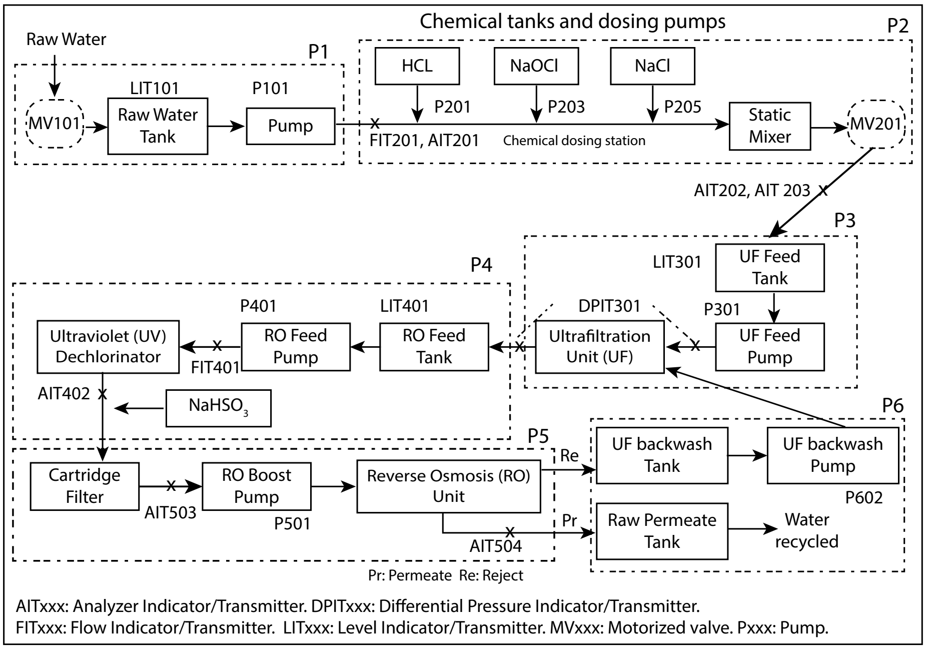


Figure 1. Sub-processes in SWaT.

Communications: Each PLC obtains data from sensors associated with the corresponding stage, and controls pumps and valves at that stage Ultrasonic level sensors in each tank inform the PLCs of water level in the corresponding tank. Several other sensors are available to check the physical and chemical properties of water flowing through the six stages. PLCs communicate with each other through a separate network. Communications among sensors, actuators, and PLCs can be via either wired or wireless links; manual switches allow switch between the wired and wireless modes.

Attacking SWaT: The wireless network in SWaT connects PLCs to sensors, actuators, and to the SCADA server and an engineering workstation. Attacks that exploit vulnerabilities in the protocol used, and in the PLC firmware, are feasible and could compromise the communications links between sensors and PLCs, PLCs and actuators, among the PLCs, and between PLC and SCDA and the historian. Having compromised one or more links, an attacker could use one of several strategies to send fake state data to one or more PLCs, or simply do reconnaissance for a possibly subsequent attack.

2.2. System state

Each PLC in SWaT controls one or more actuators, e.g., a pump. The *local* state of SWaT is comprised of the respective observable states of components under direct control of a single PLC. An observable state of a component is one that can be deduced from readings obtained via one or more sensors associated directly with that component. readings. A collection of local observable states of all six PLCs in SWaT constitutes its *global* state. It is important to note that only the observable properties of the process and its components are included, and used for attack detection, in the local and global states. In IC-Exp, only the local states of PLC 1 are used to detect cyber attacks. In IC-Exp only the local state in stage 1 of SWaT was used to detect cyber attacks. While it possible to use states from other stages, an objective here was to assess the effectiveness of IC-Exp based on state information obtained from only one stage.

2.3. Normal operation

SWaT is designed to operate continuously without any interruption. As shown in Figure 1, water to be treated is passed from tank T101 to tank T301 for ultrafiltration. Tank T101 is normally kept at or above level “L”. The motorized valve MV101 is opened by PLC 1 when T101 reaches level ”L”. Opening the valve causes water to flow in to T101 from outside of SWaT. PLC 1 keeps track of the level in tank T301. It does so by obtaining the state of T301 from PLC3 . Note that the state of T301 is available to PLC3 via LIT301. Whenever tank T301 becomes low, i.e., reaches state “L”, pump P101 is turned ON to direct water from T101 to T301. Thus, PLC 1 ensures that T101 and T301 are always in state that allows for the ultrafiltration process to operate continuously without interruption. Stages 4, 5, and 6 also operate in a continuous manner; details of these later stages are omitted here as the experiments reported do not involve any sensors from these stages.

The normal operation of SWaT can be affected by a cyber attack. Numerous such attacks are possible that may cause SWaT to either shut down, or continue to operate while producing at a less than the normal production rate, or cause a tank to overflow, or even damage some of the critical units such as the UF or RO. Numerous such attacks have already been reported [1]. In the following, two attack scenarios, not considered in earlier work, are considered and methods to detect such attacks discussed and experimentally evaluated.

2.4. Attack scenarios

Two attack scenarios labeled A1 and A2 were considered. Figure 2 shows the components of SWaT directly involved in these two scenarios. Table 1 lists the state of SWaT when an attack is launched and the sensors used in detecting these. To understand A1 and A2 and their potential impact, consider an attacker who intends to launch a cyber attack on SWaT to significantly reduce its productivity below the standard 5 gallons/minute. In A1, the attacker is able to compromise links **A**, **B**, and **C**, marked in Figure 2. Thus, the attacker can open/close MV101 at will, report to PLC 1 an incorrect flow rate and tank level of T101 which would be given by, respectively, FIT101 and LIT101. It is assumed that attacker knows the specifications of MV101 and the dimensions of T101. This knowledge allows the attacker to compute, in real time, the output of LIT101 given those of FIT101 and FIT102 and thus launch a replay attack. This computation is done to make the PLC (incorrectly) believe that the state of MV101 is what i should be. This attack could be based on pre-computed values of the sensors or values computed in real time. Scenario A2 is similar to A1 except that in A2 the attacker also compromises FIT201 marked as **D** in Figure 2. Following is a sequence of attacker-PLC actions in A1 that would lead to a reduction in water production unless the attack is detected sufficiently early.

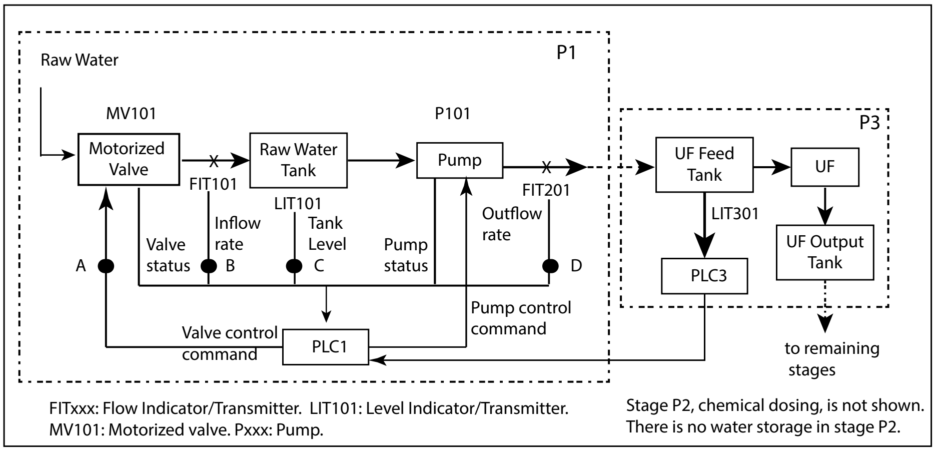


Figure 2. SWaT components involved in the attack and detection scenario. Links compromised by the attacker are indicated by a darkened circle marked A, B, C, and D. Pump P101 is used as a device to which intermittent control actions are applied for detecting cyber attacks.

Table 1. Attack scenarios

Attack Scenario	**Components**[†]	**State**	**Description**
A1	LIT101*	L[‡]	T101 is low and needs to be filled.
	MV101*	Open	Flow into T101
	FIT101*	Non-zero	Indicates flow rate into T101
	P101	OFF	No outflow from T101
	T301	H	No inflow into T301
	F201	0	No outflow from T101
A2	LIT301	H	Indicates the state of T301. All sensors used in detecting A1 are also used in detecting A2. All components states in P1 and P3 are as in A1.
	FIT201*	0	No outflow from T101

[†] Sensors not listed are not used during attack detection. * Sensors and actuators compromised.[‡]Tank states: HH=1000mm, H=800mm, L=500mm, LL=250mm.

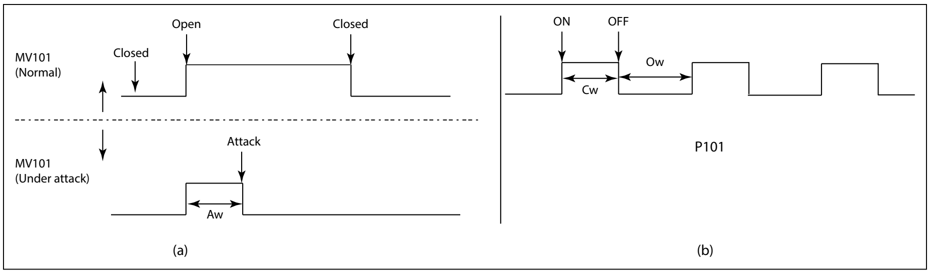


Figure 3. (a) Attack on MV101. (b) Intermittent control action on pump P101 to detect the attack (see text in Section 2.7). In the experiments reported here, $C_w = 30$ seconds, $O_w = 2$ minutes.

1. Initial state: (As in Table 1) Tank T101 level status is "L"; the Ultrafiltration (UF) process is active and tank T301 is at "H". This implies that pump P101 is OFF and hence there is no outflow from tank T101; FIT201=0.
2. PLC 1: Open MV101 because T101 level reported by LIT101 is "L" (see the upper portion of Figure 3). In a few seconds MV101 opens and FIT101 shows non-zero flow rate.
3. Attacker: (a) After some time, denoted as A$_w$, close MV101 thereby causing the correct reading of FIT101 to drop to 0 (see the lower portion of Figure 3). However, as the attacker has compromised FIT101, PLC 1 continues to receive non-zero flow rate. (b) For each request from PLC 1, set LIT101 to $L_0+\delta$, where L_k is the water level in T101 at the time instant k and δ is the increase in tank level due to the inflow of water; the attacker computes δ using the specifications of MV101 and T101.
4. PLC 1: When the attack is launched, T301 is at "H" and hence P101 is OFF. However, as the ultrafiltration process is active, after some time T301 will be in state "L". PLC 1 checks the state of T301 on a regular basis. When it reaches "L" it turns ON pump P101 but only if T101 is above "L". In this case the actual level of T101 is "L" but through LIT101 the attacker is indicating that it is above "L" and hence P101 can be started. Note that MV101 is OFF implying that no water is flowing into T101, and P101 is ON. After some time there will be no water in T101 which will either cause P101 to trip or to be damaged. This will lead to stoppage of water flow into T301 and of the ultrafiltration process hereby leading to a reduction in the net productivity of SWaT.

The steps involved in attack scenario A2 are similar to those described above except that in this case FIT201 is also compromised. Hence, the attacker can willfully send incorrect data to PLC 3 by manipulating FIT201.

2.5. Attack procedure

The following general procedure was used to launch cyber attacks in both scenarios.

1. Identify the *tags* to be manipulated in the attack. A tag is a memory location where a PLC saves the received sensor data.
2. Compromise the wireless link between the SCADA computer and PLC 1; this is the level 2 network in SWaT.
3. Manipulate the *tags*. Manipulation involves setting the tag to a value different from that received by the PLC. In the absence of any hardware or attack detection logic in PLC 1 code, PLC 1 assumes the manipulated value to represent the true state of the component that corresponds to the sensor whose output is manipulated.

2.6. Invariant for attack detection

Let x denote property p of a component of the system under attack, and y its measurement obtained from a sensor. $y(k)$ denotes the sensor measurement for $x(k)$ at instant k. $\hat{x}(k)$ is an estimate of $x(k)$. In the absence of sensor errors and no cyber attacks, $\hat{x}(k) = x(k) = y(k)$. In ICA the water level in tank T101 was considered as the property p. The level in T101 is measured by sensor LIT101 (Figure 1). Sensors FIT101 and FIT201 measure, respectively, water flow into and out of T101. These flow rates are denoted as $u_i(k)$ for inflow, and $u_o(k)$ for outflow. In SWaT, each PLC obtains sensor data at 0.1 second intervals. However, for detecting an attack, data was sampled from LIT101, FIT101, and FIT201 every second as smaller sampling intervals did not offer any benefit in attack detection in trial runs. $\hat{x}(k+1)$ was computed as follows,

$$\begin{aligned}\hat{x}(0) =& y(0)\\ \hat{x}(k+1) =& y(k) + \alpha(u_{in}(k) - u_{out}(k)),\end{aligned} \tag{1}$$

where the constant α is based on the dimensions, e.g., diameter and height, of T101 to transform flow rates into an increase/decrease in water level. In the above, the water level in tank T101 is estimated using sensor values.

At time instant $k+1$, the water level in T101 depends on the level at time instant k and the inflow and outflow at time instant k. This relationship is captured in the following idealized model of the tank,

$$x(k+1) - x(k) = \alpha(u_i(k) - u_o(k)), \tag{2}$$

where (2) assumes sensor with no noise. Eqn.2 is an ideal invariant for the water flow process in stage 1 of SWaT. To derive a practically usable invariant, SWaT was run several times without any attacks to estimate the mean μ_d and the standard deviation σ_d of $d = (\hat{x}(k) - y(k))$ over several runs, i.e., the mean and variance of the difference between the estimated tank level $\hat{x}(k)$ and its measured value ($y(k)$). In these runs $\hat{x}(k)$ was computed using Eqn. 1. Based on Eqn. 2, the statistics obtained experimentally, and converting the true process state to its estimate, the following conditions were derived to test whether or not sensor LIT101 is under attack.

$$E = \frac{\sum_{i=1}^{n}(\hat{x}(i) - y(i))}{n} > \quad \epsilon, \text{ under attack,} \tag{3}$$

$$E \le \quad \epsilon, \text{ normal.} \tag{4}$$

Figure 4. Invariant to detect attacks on LIT101.

In Eqns. 3 and 4, the average E of the difference between the estimated and the measured tank levels is tested against ϵ. Thus, a decision whether or not LIT101 is under attack is taken from n sensor readings. Selection of n ought to be done carefully as it impacts the detection effectiveness. In ICA n, was set to 10. As described earlier, based on trial runs of SWaT without attacks, ϵ was set to 0.55.

2.7. Experiments

Based on the invariant in Eqn.4, sample code to implement attack detection was added to the control algorithm already built into PLC 1. Table 2 lists four experiments conducted to assess the effectiveness of the invariant in Eqn. 3 in detecting attacks in scenarios A1 and A2.

Table 2. Experiments

Scenario	Experiment	Detection method
A1	1	Invariant
	2	Invariant and intermittent control action
A2	3	Invariant
	4	Invariant and intermittent control action

The intermittent control action was to switch P101 ON and OFF as indicated in Figure 3(b). C_w denotes width of the pulse in the intermittent control action. O_w is the time for which P101 is in the OFF state. Each experiment set consisted of several individual experiments to understand the impact of n (Eqn 3), A_w, C_w, and O_w on the attack detection effectiveness of the methods proposed. ICA was coded (Figure 7) and the code added to PLC 1 together with that to compute the invariant E.

As ICA is applied to P101, C_w should be set based on the duty cycle of P101. Pump P101 takes about 15 seconds to move from OFF to ON state and 10 seconds from ON to OFF state. Thus, C_w was set to 1 minute and O_w to 5 minutes in experiments 2 and 4. Selection of C_w and O_w must be done carefully as discussed in Section 4.

3. Results

To understand the effectiveness of the invariant and ICA, four experiments were conducted as summarized in Table 2. Additional exploratory experiments were conducted to understand the impact of A_w, O_w, and C_w on the attack detection effectiveness and on the behavior of SWaT. Results from these experiments are summarised in Table 3 and explained in detail in the following.

Table 3. Effectiveness of attack detection methods

Scenario	Detection method	Outcome
A1	Invariant	Attack detected, though delayed; intent not realized (Figure 5(a))
	Invariant+ ICA	Attack detected soon after P101 is started; intent not realized (Figure 5(b)).
A2	Invariant	Not detected; intent realized: T101 becomes empty and water output falls below 5gallons/minute (Figure 6(a)).
	Invariant + ICA	Not detected; intent realized: T101 becomes empty and water output falls below 5gallons/minute (Figure 6(b)).

RQ 1: How effective is the use of ICA in detecting multi-point attacks on a water treatment system?

Experiment 1: A1 with invariant detector: Error E is plotted against time in Figure 5(a). As shown, the attack is launched at about 201 seconds from the start of the experiment. For about next 100 seconds, E reduces significantly and condition in Eqn. 4 is true and hence the attack remains undetected. However, at around 301 seconds, pump P101 is started by PLC 1 because tank T301 reaches its "L" state. Thus, detection is delayed by as much as the time it takes for T301 to reach "L". In this the attacker intent is not realized as P101 keeps T301 above "L" and hence the UF unit continues to filter water. '

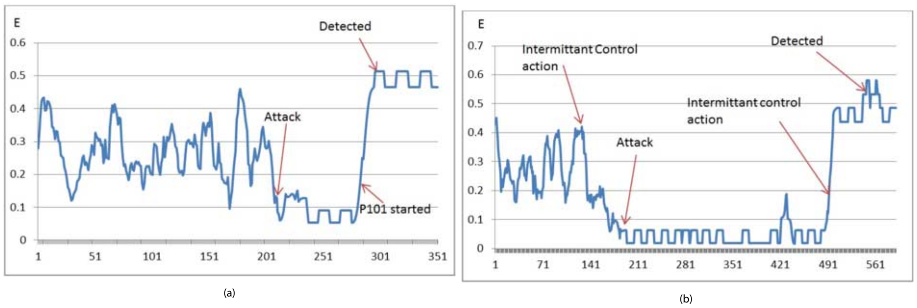


Figure 5. Error E (Eqn. 3) against time. Detection of attack scenario A1 with (a) invariant and (b) invariant and intermittent control action.

Experiment 2: A1 with invariant and ICA : This attack is detected soon after P101 is started due to the control action initiated by PLC 1, and the most recent value of E has been computed. Recall that computation of E requires n readings. Thus, after P101 is started, the attack was detected in at most n seconds. It might seem that the use of ICA is redundant as A1 is detected, as in experiment 1, by using the invariant alone. This is not always true; note that attack detection is delayed when using the invariant alone. The delay depends on the state of tank T301. If T301 is at its top, i.e., state is “HH”, then the P101 will start only after T301 reaches “L”. However, when the ICA is used, P101 is started regardless of the state of T301 and hence the attack is detected almost immediately.

Experiment 3: A2 with invariant: Recall that in this scenario the attacker has compromised sensors LIT101, FIT101 and FIT301. Hence, the attacker is essentially controlling the computation of E in Eqn. 3. As can be seen in Figure 6(a), the attack is not detected even after 400 seconds due to a very small value of E. In this case pump P101 will never turn ON and hence eventually tank T301will get to state “L”. Soon after T301 reaches state “L”, the UF unit will stop. This will cause a cascading effect and will cause the RO unit to stop once T401 becomes empty. Though the experiment was stopped at about 400 seconds after it started, the entire SWaT will shut down sometime after the attack is launched. The exact time of shutdown will depend on the state of all the intermediate tanks at the time the attack is launched.

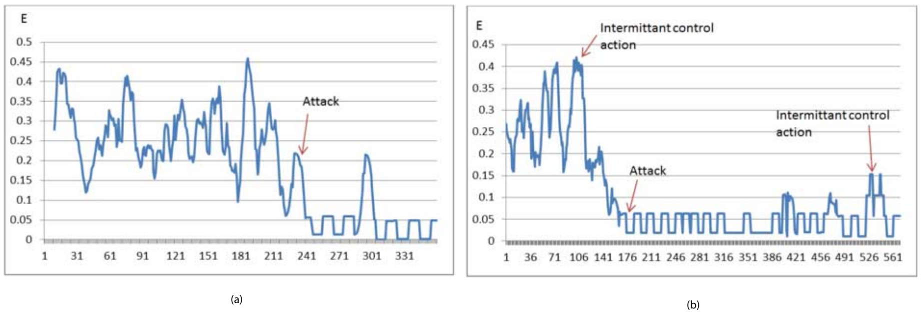


Figure 6. Error E (Eqn. 3) against time. Detection of attack scenario A2 with (a) invariant and (b) invariant and intermittent control action.

Experiment 4: A2 with invariant and ICA : As shown in Figure 6(b), A2 is not detected even when ICA is used. Again, this happens because the attacker is controlling the computation of E. In the figure the value of E does increase when P101 is started by PLC 1. This happens because the attacker does not know that P101 has been started and hence is not recomputing the value of LIT101 to match with the net inflow and outflow of water in tank T101. Note that in fact an attacker can find out the state of P101 simply by observing a change in the value sent by F201. However, this aspect was not implemented in the experiment as the attack was not detected anyway. There does exist a possibility that the attack is detected when (a) the attacker does not recompute E accounting for the state of P101 and (b) the outflow from P101 is high enough to force E such that the invariant becomes false.

Additional exploratory experiments were conducted to investigate the impact of the parameters indicated in Figure 3. These parameters are: the delay in launching the attack (A_w), duration of P101 in the ON state (C_w), and the duration of P101 in the OFF state (O_w). The outcome of these experiments is summarized next.

Impact of C_w: The duration for which P101 is kept in the ON state (C_w) does affect the attack detection effectiveness of the ICA approach. Experiments show that C_w should be computed as $min(d, n)$, where d is the duty cycle of the pump, and n the number of readings used in computing E. If C_w is less than the duty cycle plus number of readings to find E, the attack can not detected. Larger C_w may cause overflow the next stage water tank, T301 in the current study. Also, lower values of C_w might have a negative impact on the actuator's lifetime and its reliability. In the experiments reported, C_w was set to 1 minute. Additional experimentation is needed to determine the optimal C_w against several attack scenarios.

Impact of A_w: A_w does not affect the attack detection effectiveness. However, it does affect the time to realize the attacker objective. In the case of the attack scenarios considered in this work, it was observed that (a) smaller A_w leads to quicker realization of the attacker objective when the attack is not detected, and (b) larger A_w increases the time to realize the objective assuming the attack is not detected.

Impact of O_w: O_w is time when P101 is OFF, unless turned on using normal conditions, i.e., when T301 reaches state "L". O_w affects the attack detection effectiveness. Lower values of O_w lead to quicker detection but only if ICA is effective, i,.e., in scenario A1. However, low values of O_w may reduce the actuator life time just as low values of C_w may. Hence the system designer ought to select C_w and O_w such that detect time is minimized without unduly affecting the actuator reliability.

4. Discussion

RQ 2: What is the complexity and scalability of the attack detection approach?

Scalability of ICA: Industrial strength water treatment systems produce millions of gallons of water per day. The pumps used in such facilities are huge and often require a protocol, sometimes including a human, to be followed before they can be turned ON or OFF. In such situations it is impractical to apply ICA for attack detection. However, in large water treatment plants one could design a separate auxiliary system specially for applying ICA. Such an auxiliary system consists of a small pump with appropriate sensors in addition to the large pump. ICA is applied to the auxiliary system and not to the main pump. The pump, and C_w, should be designed such that the intermittent control action can detect a change in the tank level for attack detection. The use of auxiliary systems to detect cyber attacks on ICS has also been proposed using the concept of Intelligent Checkers [14] and in [19] using time variant systems.

Design of invariants and complexity: As observed in the experiments, intermittent control actions are unable to detect attack scenario A2. However, this should not be considered as an inherent limitation of the approach itself. Instead, the design of the invariant plays a significant role in the effectiveness of ICA. In experiments 3 and 4, it is feasible to detect scenario A2 if sensors in stage 3 are also used in the invariant. Doing so offers more process data to the PLC and hence more chances of detecting the attack. Using more information does increase the complexity of the invariant while likely increasing its effectiveness in attack detection.

RQ 3: What are the impacts of using ICA on the physical design of the water treatment system?

Design constraints: In scenario A2 it is possible that tank T101 is empty while PLC 1 is sending a command to turn ON P101. P101 does not have a dry-run cut off and hence after a few cycles the pump will likely get damaged. Thus, prior to turning P101 ON, PLC 1 must check, using an auxiliary mechanism, whether there is water in T101. The obvious assumption is that the auxiliary mechanism is secure.

Generalising the ICA approach: A combination of process invariant and intermittent control actions for detecting cyber attacks appears applicable to ICS other than water treatment systems. For example, in a smart grid, one could explore the use of ICA to detect an attack on the emergency battery management system by requesting energy for an emergency load. However, the choice of parameters, such as C_w and O_w ought to be done with extreme care due to the short reaction time of most power machinery.

Detection using patterns in invariant error E

A cursory examination of Figures 5 and 6 reveals an interesting pattern. First, the error E reduces significantly from its high value of about 0.4 to 0.15. Second, the entire pattern of E over the observed duration of about 200 seconds changes significantly when the attack is launched. This happens because the attacker is not accurately simulating the random variation in the sensor errors that are evident in the pattern of E prior to the attack. Instead, the attacker is either sending to the PLC pre-computed values of the compromised sensors, or computing them in real time but without simulating the physical randomness in the sensors. This change in error pattern is useful for machine learning algorithms to improve the attack detection effects in both attack scenarios A1 and A2 [5,12]. Certainly, an attacker could be more severe and replay the error pattern as in replay attacks [10,18].

5. Conclusions and future work

Two attack scenarios were considered in IC-Exp. Based on these two scenarios, a single most important conclusion from the experiments reported here is that *the ICA approach, combined with local process invariant, has its limitations unless the invariant used is extended beyond the SWaT stage that has been compromised.* Both scenarios consider attacks that are limited to a single stage of SWaT.

Additional case studies are needed to investigate the following. (a) How does an invariant(s)-based approach, combined with ICA, perform when sensors across multiple stages are compromised. (b) How best to design invariants to improve their effectiveness. Note that when stage k in SWaT is compromised, a PLC in of the uncompromized stages ($\neq k$) may be able to detect the attack. Certainly, in any case, perhaps the effectiveness of the invariant based method, when combined with ICA, reduces as the attacker manages to compromise an increasing number of sensors across stages of SWaT.

```
(* ICA: Intermittent Control Action.*)
(* November 1, 2015. Programmed by Sridhar Adepu. *)

IF HMI_PLANT.START OR HMI_PLANT.STOP THEN
    ICA_STATE:=0; (* Initialize state of the ICA processor *)
END_IF;
CASE ICA_STATE OF
    0: (* *)
        for i:=1 to 60 do (* Wait for 60 seconds. *)
            IF _SEC_P THEN (* Wait for 1 second after system start.*)
            ENDIF
        ICA_STATE := 1; (* Next state of the ICA processor.*)

    1: (* $C_w$=1 minute.*)
        HMI_P101.Auto := 0;
        HMI_P101.Status := HMI_P101.Status XOR 1
        for i:=1 to 60 do (* $C_w$: Pump ON for 60 seconds.*)
            IF _SEC_P THEN (* Wait for 1 second*)
            END_IF;
        ICA_STATE:=2; (* Next state of the ICA processor.*)

    2: (* $O_w$=5 minutes *)
        HMI_P101.Auto := 0;
        HMI_P101.Status := HMI_P101.Status XOR 1;
        for i:=1 to 300 do (* Pump OFF for 5 minutes. *)
            IF _SEC_P THEN (* $O_w$: Wait for 1 second. *)
            ENDIF
        ICA_STATE:=1; (* Set next state of ICA processor to 1. *)
    ELSE (* If ICA_STATE is not any of 0 through 2. *)
        ICA_STATE:=0;
END_CASE;
```

Figure 7. Code included in PLC 1 to detect an attack using intermittent control actions.

Acknowledgements

Thanks to Kaung Myat Aung for assistance in conducting the experiments and debugging the code added to PLC 1. Thanks to the reviewers for a careful reading and suggesting edits.

References

[1] S. Adepu and A. Mathur. An investigation into the response of a water treatment system to cyber attacks. In *Proceedings of the 17th IEEE High Assurance Systems Engineering Symposium (in Press)*, January 2016.

[2] J. K. Barford, D. Plonka, and A. Ron. A signal analysis of network traffic anomalies. In *Proc. ACM SIGCOMM IMW, France*, pages 71–82, Nov. 2002.

[3] R. Berthier and Sanders. Specification-based intrusion detection for advanced metering infrastructures. In *17th IEEE Pacific Rim International Symposium on Dependable Computing*, pages 184–193, Oct 2011.

[4] A. A. Cárdenas, S. Amin, Z.-S. Lin, Y.-L. Huang, C.-Y. Huang, and S. Sastry. Attacks against process control systems: Risk assessment, detection, and response. In *ACM Symp. Inf. Comput. Commun. Security*, 2011.

[5] M. Esmalifalak, L. Liu, N. Nguyen, R. Zheng, and Z. Han. Detecting stealthy false data injection using machine learning in smart grid. *IEEE Systems Journal*, PP(99):1–9, 2014.

[6] D. Hadžiosmanović, R. Sommer, E. Zambon, and P. H. Hartel. Through the eye of the PLC: Semantic security monitoring for industrial processes. In *Proceedings of the 30th Annual Computer Security Applications Conference*, pages 126–135, New York, NY, USA, 2014. ACM.

[7] S. Han, M. Xie, H.-H. Chen, and Y. Ling. Intrusion detection in cyber-physical systems: Techniques and challenges. *IEEE Systems Journal*, 8(4):1049–1059, Dec 2014.

[8] M. Krotofil, J. Larsen, and D. Gollmann. The process matters: Ensuring data veracity in cyber-physical systems. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*, pages 133–144, 2015.

[9] C. McParland, S. Peisert, and A. Scaglione. Monitoring security of networked control systems: It's the Physics. *IEEE Security Privacy*, 12(6):32–39, Nov 2014.

[10] Y. Mo and B. Sinopoli. Secure control against replay attacks. In *47th Annual Allerton Conference on Communication, Control, and Computing, Allerton*, pages 911–918, 2009.

[11] Modbus application protocol specification; Modbus IDA, June 2004.

[12] M. Ozay, I. Esnaola, F. Yarman Vural, S. Kulkarni, and H. Poor. Distributed models for sparse attack construction and state vector estimation in the smart grid. In *EEE Third International Conference on Smart Grid Communications (SmartGridComm)*, pages 306–311, Nov 2012.

[13] E. S. Page. Continuous inspection scheme. *Biometrika*, 41 (1/2):100–115, June 1954.

[14] G. Sabaliauskaite and A. P. Mathur. Intelligent checkers to improve attack detection in cyber physical systems. In *Proceedings of the 2nd IEEE International Workshop on Cyber Security and Privacy (CSP 2013), International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC 2013) Beijing, PRC, in press)*, 2013.

[15] G. Sabaliauskaite, G. S. Ng, J. Ruths, and A. Mathur. Experimental evaluation of stealthy attack detection in a robot (in press). In *The 21st IEEE Pacific Rim International Symposium on Dependable Computing (PRDC 2015)*, 2015.

[16] A. Tartakovsky, B. Rozovskii, R. Blazek, and H. Kim. A novel approach to detection of intrusions in computer networks via adaptive sequential and batch-sequential change-point detection methods. *Signal Processing, IEEE Transactions on*, 54(9):3372–3382, Sept 2006.

[17] G. Thatte, U. Mitra, and J. Heidemann. Parametric methods for anomaly detection in aggregate traffic. *Networking, IEEE/ACM Transactions on*, 19(2):512–525, April 2011.

[18] T. T. Tran, O. S. Shin, and J. H. Lee. Detection of replay attacks in smart grid systems. In *Proceedings of the International Conference on Computing, Management and Telecommunications*, pages 298–302, 2013.

[19] S. Weerakkody and B. Sinopoli. Detecting integrity attacks on control systems using a moving target approach (I). In *Proceedings of 54th IEEE Conference on Decision and Control (in press)*, 2015.

Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016
A. Mathur and A. Roychoudhury (Eds.)

doi:10.3233/978-1-61499-617-0-75

Attacking Fieldbus Communications in ICS: Applications to the SWaT Testbed

David URBINA [a,1], Jairo GIRALDO [a], Nils Ole TIPPENHAUER [b], and Alvaro CARDENAS [a]
[a] *University of Texas at Dallas*
[b] *ISTD, Singapore University of Technology and Design*

Abstract The study of cyber-attacks in industrial control systems is of growing interest among the research community. Nevertheless, restricted access to real industrial control systems that can be used to test attacks has limited the study of their implementation and potential impact. In this work, we discuss practical attacks applied to a room-sized water treatment testbed. The testbed includes a complete physical process, industrial communication systems, and supervisory controls. We implement scenarios in which the attacker manipulates or replaces sensor data as reported from the field devices to the control components. As a result, the attacker can change the system state vector as perceived by the controls, which will cause incorrect control decisions and potential catastrophic failures. We discuss practical challenges in setting up Man-In-The-Middle attacks on fieldbus communications in the industrial EtherNet/IP protocol and topologies such as Ethernet rings using the Device-Level-Ring protocol. We show how the attacker can overcome those challenges, and insert herself into the ring. Once established as a Man-in-the-Middle attacker, we launched a range of attacks to modify sensor measurements and manipulate actuators. We show the efficacy of the proposed methodology in two experimental examples, where an adversary can intelligently design attacks that remain undetected for a typical bad-data detection mechanism.

Keywords. ICS, Critical System, Cyber-attacks, Fieldbus

1. Introduction

In recent years, security threats to industrial control systems (ICS) have received an increasing amount of attention [7,6,15,8,10]. One area that has received particular attention are attacks against the integrity of sensor and control data in the system, also known as false data injection attacks [12,16,19,14]. If the attacker manipulates or replaces sensor data reported from the field devices, the control algorithm will take actions based on an incorrect perception of the world, which will cause incorrect control decisions and potential catastrophic failures.

So far, most of the analysis on false data injection has focused on a wide range of suspected theoretical attacks and simulation studies, and only limited work has been

[1]Corresponding Author: David Urbina, University of Texas at Dallas, 800 Campbell Rd., Richardson, 75080, TX; E-mail: david.urbina@utdallas.edu

published on the practical challenges for launching such attacks as access to real-world ICS is usually hard to obtain for security researchers.

In this work, we discuss practical attacks applied to a room-sized water treatment testbed (the SWaT testbed). The Secure Water Treatment (SWaT) testbed includes a complete physical process, the related industrial communication infrastructure, and a supervisory control network. This paper is an extention of our previous work [9], where we launched Man-in-the-Middle (MitM) attacks at the supervisory control layer of the SWaT testbed. In that work, we attacked the physical system using Ettercap spoofing on the Supervisory Control network level, which enabled us to modify the sensor information. In addition, we noted that Ettercap-based spoofing can easily be detected by more complex networking devices such as SDN-capable switches and their controllers.

In contrast to [9], our proposed methodology attacks fieldbus communications directly, which enables the adversary to have total control of the system operation, without taking into account any command coming from the higher levels such as the Human-Machine Interface (HMI). Such attacks on fieldbus communications are challenging due to the ring topology and the device specific messages used in the fieldbus, requiring more specific knowledge of the network operation. We discuss practical challenges in setting up MitM attacks on a fieldbus communication network using the EtherNet/IP protocol over an Ethernet ring (maintained with the Device-Level-Ring (DLR) protocol). Once established as a MitM attacker, we are able to launch a range of sensor and actuator attacks and demonstrate their efficacy.

We summarize our contributions as follows:

- We study fieldbus communications and implement a prototype Man-in-the-Middle (MitM) attack.
- We show how a MitM attacker can obtain sensor readings from eavesdropped packets, and craft her own spoofed sensor and actuator command traffic. We also provide details on datatypes and conversions required (e.g., from 4-20mA signal to physical measurements).
- We combine the MitM attack and detailed understanding of the EtherNet/IP protocol to demonstrate practical attacks on SWaT, and show their impact on the physical process.

This work is structured as follows: in Section 2, we state the problem setting and attacker model. In Section 3, we present our practical attacks and discuss them in detail. We conclude the paper in Section 4.

2. Background and Problem Formulation

In this section, we introduce Industrial Control Systems (ICS), their fieldbus networks, and the specific testbed we use for our experiments. Finally, we introduce our attacker model.

2.1. The SWaT Physical Process

The SWaT testbed is a water treatment plant which consists of 6 main processes to purify raw water. Each process possesses a PLC that receives the information from the sensors and compute the control actions to the actuators. SWaT is set up to have two different communication channels for many links: either wired (over IEEE 802.3 Ethernet) or wireless communications (using IEEE 802.11).

The main physical processes SWaT can be described as follows (see Figure 1):

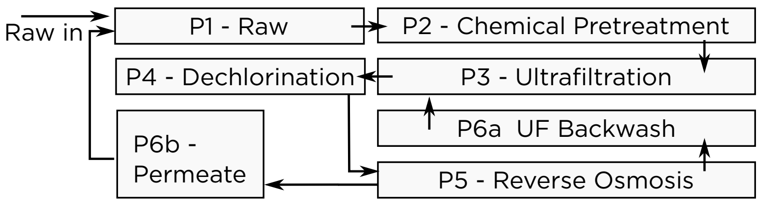


Figure 1. SWaT physical process overview

Raw water (P1) In this process, raw water is stored. P1 acts as the main water buffer supplying water to the water treatment system. It consists of one tank, an on/off valve that controls the inlet water, and a pump that transfers the water to the Ultra Filtration (UF) system's tank.

Pre-treatment (P2) While the water from P1 is pumped to the UF system, water quality properties are evaluated and pre-treated. Conductivity, pH, and ORP are measured to determine the activation of chemical dosing to maintain the quality of the water within desirable limits.

Ultra Filtration (P3) The ultra-filtration process is used to remove the bulk of the feed water solids and colloidal material by using fine filtration membranes that only allow the flow of small molecules. The accumulated contaminants are removed by back-washing away the membrane surface depending on the measure of a differential pressure sensor located at the two ends of the UF.

Dechlorinization (P4) After the small residuals are removed by the UF system, the remaining chlorines are destroyed in the ultraviolet chlorine destruction unit and by dosing a solution of sodium bisulphite.

Reverse Osmosis (P5) The RO system is designed to reduce inorganic impurities by pumping the filtrated and dechlorinated water with a high pressure through semipermeable membranes.

RO final product (P6) The last part of the water treatment process consists on storing the RO product i.e., cleaned water ready to distribute. In the SWaT case, treated water is transferred again to the raw water tank in order to reuse it.

The SWaT testbed features distributed controls among the different process stages. Each stage is controlled by a PLC (with hot-redundant counterpart), and all PLCs and the SCADA system are connected together through a common network (which we call Level 1 (L1) network). In addition, the PLCs are connected to local sensors and actuators through individual fieldbus rings called Level 0 (L0). We now give details of this network architecture.

2.2. Industrial Control Network

A modern industrial control system typically consists of several layers of networks. The SWaT industrial control network is illustrated in Figure 2. The physical process is measured by distributed sensors, and manipulated by actuators. These sensors and actuators in SWaT operate by receiving and sending analog signals (4mA). The analog signals are converted into digital signals by Remote Input/Output (RIO) modules. The digital signals are then encapsulated over *fieldbus* communication protocols (EtherNet/IP in our case over the L0 network Figure 2), and sent back and forth from PLCs. PLCs in turn communicate with a centralized Supervisory Control and Data Acquisition (SCADA) system with the L1 network in Figure 2. This central system contains the HMI and Historian. In this work, we want to show a systematic methodology to deploy cyber-attacks on industrial control systems (ICS), with a focus on fieldbus communication networks.

The reliability of such fieldbus networks is of great concern to the plant operator. For that reason, ring topologies are a popular choice to implement these topologies. The ring topology can be seen in Figure 2 at the L0 network, where there is a ring between the RIO and a primary and a backup PLC. In particular, rings can tolerate faults such as the loss of a single ring segment, without losing connectivity between any of the participating devices. If the communication uses Ethernet as medium, rings can be constructed using the device-level-ring (DLR) protocol.

In the context of an attacker who tries to insert itself as MitM, such ring topologies have interesting properties. In particular, if the attacker cuts the ring to insert her own device, the ring will automatically stop transmitting data through the "lost" segment. As a result, the attacker will not receive any traffic to eavesdrop on. In order to successfully complete the attacker, the attacker must "close" the ring again. We discuss that further in Section 3.2.2.

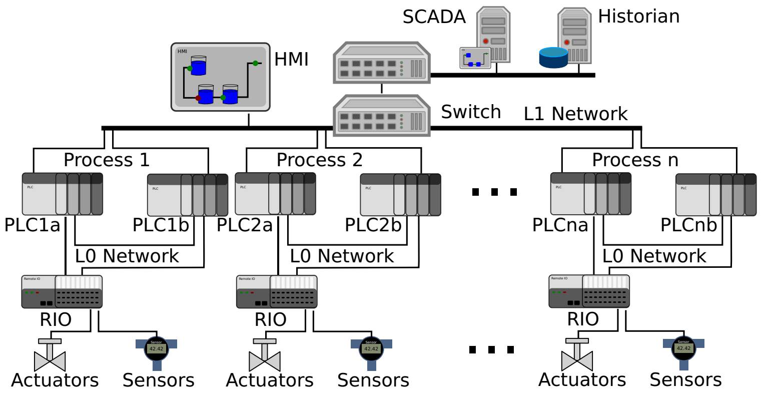


Figure 2. SWaT network architecture.

2.3. Fieldbus Communications

SWaT's ring topology in the fieldbus network contains the following four main devices (see Figure 3).

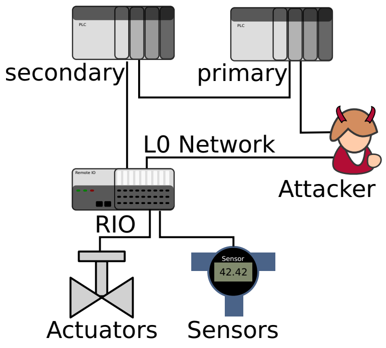


Figure 3. Example ring topology in SWaT, with an attacker inserted as Man-in-the-Middle. In this configuration, the attacker can eavesdrop and manipulate all traffic between RIO and primary PLC.

Programmable Logic Controller (PLC) This is the control device which receives sensors readings and emits control commands to the actuators. SWaT's PLCs correspond to a chassis conformed by 6 components: 1756-PA2 Control Logix AC Power Supply Unit; 1756-L71 Control Logix 2MB Controller; 1756-EN2T Ethernet I/P Module (for communications at the Supervisory Control Network level); 1756-EN2TR Dual Port Ethernet I/P Module (for fieldbus communications at the ring level); 1756-RM2 Control Logix Redundancy Module; and a 1756-RMC1 Control Logix Redundancy Fiber Optic Cable. Each SWaT's ring presents a redundant PLC, and on power up the system selects one as Primary, actively controlling the physical process, and other as Secondary, shadowing the memory state of the Primary. From that moment on, the Primary is responsible for the monitoring and control of the fieldbus communications and the relaying/receiving of data from the SCADA network. If the Primary fails, the system automatically switches over the control to the Secondary.

Remote I/O (RIO) This device is the responsible for the translation of 4-20 mA signals to/from the actuators/sensors to a stream of bytes where each byte (or bit depending on the resolution of the I/O module) corresponds to an I/O signal of the system. For the communications between the PLC and RIO, the stream is encapsulated following the Common Packet Format of the EtherNet/IP specification [18] and transported through a wired connection. The RIO is modular, and in SWaT it consists of 4 components: 1794-AENTR Flex Dual Port Ethernet I/P Adapter (for fieldbus communications at the ring level); 1794-IB32 Flex 32 Points Digital Input Module (1-bit resolution per signal); 1794-OB16 Flex 16 Points Digital Outputs Module (1-bit resolution per signal); and 1794-IE12 Flex 12 Points Analog Input Module (16-bit 2's complement per signal).

Wireless Remote I/O (WRIO) SWaT features a manual switch to turn parts of the system into "wireless mode". If the field communications at the ring level are set to wireless mode, this WRIO takes over the responsibility of scanning the analog sensors and sending updates to the PLC. Its is important to highlight that in SWaT this wireless transmission contains exclusively the analog input signals and not the digital I/O signals. The Digital input and output signals are always transmitted through the wired ring network between the RIO and PLC.

Wireless Access Point (WAP) In wireless mode, the WRIO connects to a wireless access point connect to the EtherNet/IP ring thought a 1783-ETAP 3-ports switch.

2.4. Attacker Model

Objective. The objective of an attacker depends on how much damage she wants to cause to the system. Our proposed methodology can be used to a) manipulate sensor readings that are reported from the sensors to the PLCs, and b) to manipulate control messages that are sent from the PLC to the actuators. Therefore, a wide number of attacks can be deployed, starting with a simple eavesdropping to sophisticated integrity persistent and stealthy attacks.

Resources. The attacker is assumed to either a) being able to physically access the network connecting the PLCs, sensors, and actuators, or b) able to fully compromise one of the devices attached to that network. In both cases, the attacker will have a device attached to the network, and can program the device to transmit arbitrary messages to the network, and to process any message it receives.

3. Attacking Fieldbus Communications in SWaT

Based on the details we provided on SWaT, its fieldbus topologies, and the EtherNet/IP protocol, we now show results of practical MitM attacks. We start by introducing tools we used, and among them our custom *SWaT Assault* tool.

3.1. Tools

We used several tools to launch attacks against the fieldbus communications at the SWaT testbed:

SWaT Assault We developed a command-line interpreter (CLI) application which includes a library of attack modules capable of launching diverse spoofing and bad-data-injection attacks against the sensor and actuator signals of the SWaT testbed. The attack modules can be loaded, configured, and run independently of each other, allowing the attack of sensors and actuators separately. Attack modules also can be orchestrated and assembled in teams in order to force more complex behaviors over the physical process, while maintaining a normal operational profile on the HMI. SWaT Assault consists of 439 lines of Python [3] 2.7 code and its only external dependencies are Scapy and NetFilterQueue.

Scapy Making use of the Scapy[4] packet manipulation program we developed a new protocol parser for the Rockwell Automation proprietary message protocol used for signal communication between the RIO and the PLC, and for the EtherNet/IP Common Packet Format wrapper that encapsulates it. This parser (which we chose to call SWaT message parser) is specific for the SWaT's deployment (the SWaT Ring implementation makes use of User Datagram Protocol (UDP) for the transport of EtherNet/IP I/O implicit messages among ring devices) and its implementation follows SWaT's Control Panels and Electrical Drawings manual. Scapy was also used to sniff sensor readings from the EtherNet/IP Ring and to inject manipulated data on both, sensor readings and actuation commands. Our tool also automatically recomputes the data integrity checksums used by the Transport Layer protocol to match the false-data injection attack values.

NetFilterQueue In order to avoid duplication of packets and/or race conditions between original and injected packets, we employed the NetFilterQueue [2] Python bindings for libnetfilter_queue to redirect all the EtherNet/IP I/O messages between PLC and RIO to a handling queue defined on the *mangle* table of the Linux firewall *iptables*. The queued packets are later modified using Scapy and the previously mentioned SWaT message parser, and finally released to reach their original destination i.e. PLC or RIO. Likewise, this technique allowed us to avoid disruptions on the sequence of EtherNet/IP counters, and injection of undesirable perturbations in the EtherNet/IP connections established between ring devices.

The command we use to queue packets for modification is the following:

```
iptables -t mangle -A PREROUTING -p udp --dport <port> -j NFQUEUE
```

Wireshark We used Wireshark [5] to understand the nature of the communication between devices in the ring. We also used Wireshark together with the SWaT's Control Panel and Electrical Drawings manual, to derive the exact structure of the EtherNet/IP-wrapped messages used in SWaT.

Ettercap We used Ettercap [1], a Man-In-The-Middle attack suite, on our attempts to launch wireless attacks.

3.1.1. Differences Between Parsing Supervisory Network Packets vs. Fieldbus Packets

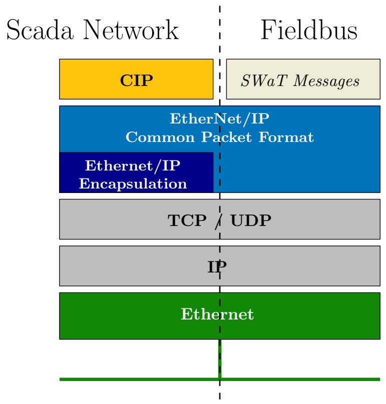


Figure 4. EtherNet/IP encapsulation for CIP and custom *SWaT Messages* for Supervisory Network and Fieldbus communications, respectively.

While the SWaT networks use EtherNet/IP at the supervisory as well as the fieldbus level, the encapsulated protocol is different at each level: the CIP protocol is used as main data payload for device communications at the Supervisory Control Network level, while a device-dependent I/O implicit message payload is employed at the Fieldbus Communications level (see Figure 4).

Parsing and injection of manipulated data at the Supervisory Control level using CIP messages or at the Fieldbus Communications level using EtherNet/IP device-dependent I/O implicit messages introduce different challenges and requirements to the attacker.

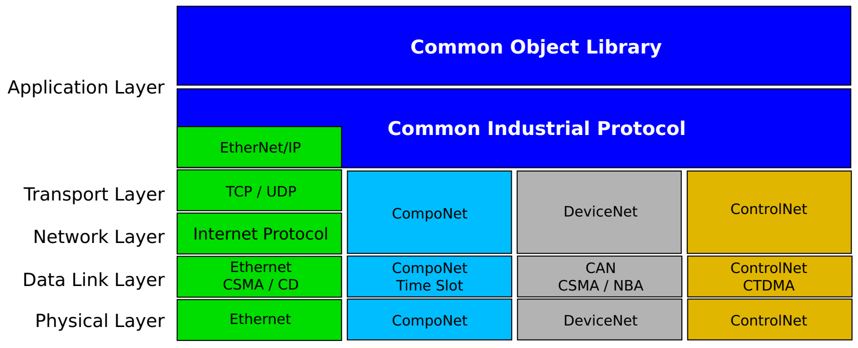


Figure 5. Common Industrial Protocol stack and its different physical layers.

Common Industrial Protocol stack The Common Industrial Protocol (CIP) network specification library was originally developed by Rockwell Automation and finally standardized and maintained by Open Device Vendors Association (ODVA) and ControlNet International. It aims to fulfill the main three needs of ICS systems: control, configuration, and collection of data [11]. It defines the CIP application layer protocol as a encapsulated object-oriented protocol for transmission of connected (I/O implicit) messages between a data producer and one or more data consumer devices, and unconnected (explicit) messages between two devices in the control network. Transmissions associated with a particular connection are assigned an unique *connection ID*. While being an application layer protocol, CIP is independent of the underlying layers, and requires an encapsulation protocol which allows abstraction from different data link and physical layers. It also includes a Common Object library defining commonly used objects, some of which are specific for a particular encapsulation protocol, and allows for extension and definition of vendor specific objects. The CIP specification library includes the definition of 4 different CIP stacks depending of the physical layer in use (see Figure 5): EtherNet/IP (over IEEE 802.3 Ethernet), CompoNet, DeviceNet, and ControlNet.

Therefore, CIP messages contain much more rich semantic information about the information being exchanged in the network, which can facilitate the understanding of the process by the attacker. CIP messages follow an object-oriented format, allowing for the transmission of a variable number of data with distinct types, which translates into highly structured packets of variable lengths (see Figure 6). The attacker must dissect each particular packet to extract the CIP object attributes containing the sensor or actuator data, which may be set at different offsets depending on the number and type of objects targeted by the packet.

On the other hand at the fieldbus level of SWaT, EtherNet/IP device-dependent I/O implicit messages follow non-standard formats of fixed lengths, partially defined by the vendor and by the control system designer, and where the analog sensor signals are encoded using 4-20 mA measurements. The attacker therefore must have detailed knowledge and understanding of the system design a priori and implementation decisions, i.e. she must have access to the devices specifications, electrical drawings, and installation layouts in order to understand the information exchanged and manipulate the sensor readings and control commands.

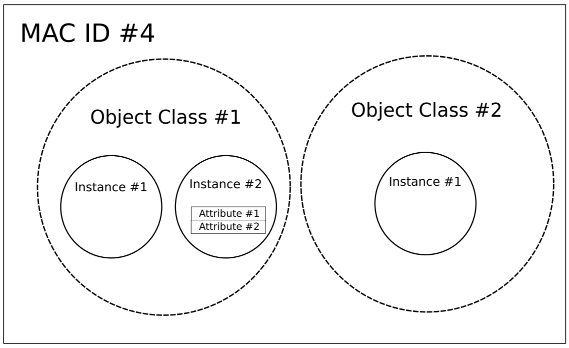


Figure 6. CIP Object-Oriented Packet Structure.

3.2. MitM of Fieldbus Communications

We attempt attacks on the wired and wireless mode fieldbus communications in SWaT. We now present results for both cases.

3.2.1. Wireless MitM

On our first try to sniff the fieldbus communications in the SWaT testbed we attempted a wireless MitM attack between the PLC and the WRIO. We set the fieldbus communications into *wireless mode* and use a laptop we connected to the WAP. Using Wireshark, we realized that we could already see one multicast EtherNet/IP connection, stacking EtherNet/IP over UDP, with the WRIO's IP as source. This multicast connection corresponded to the analog input signal which, as by SWaT's design, is the only signal transported in wireless mode. After switching on and off the wireless mode multiple times, we verified that the multicast address range corresponded to the *Organizational Local Scope* [13], as expected. The assigned UDP destination port was 2222, also defined in SWaT's design.

It is important to highlight that the attacks presented on section 3.4 could be achieved with minimum technical requirements through a *wireless MitM* if the ICS's design accounts for a complete wireless transmission of digital and analog signals. Unfortunately, the digital input and output are not reported via the wireless links, and thus cannot be compromised using the wireless MitM attack. In order to cope with this characteristic of SWaT's design, we resorted to performing a *wired MitM* directly in the EtherNet/IP ring.

3.2.2. Wired MitM

We assume an attacker who is an insider or who is able to set a physical device at any point of the EtherNet/IP ring, between the PLC and the RIO. In our experiments, for the implementation of the wired MitM, we intercepted the fieldbus communications by adding a laptop with two Ethernet ports in a segment of the EtherNet/IP ring.

DLR must be taken into consideration when attempting a MitM in the EtherNet/IP ring. If the attacker uses a DLR-unaware device, as we did in our experiments, she must disable MAC learning and Spanning Tree Protocol when bridging both Ethernet ports. Failing in carefully addressing this requirement will result in the isolation of the EtherNet/IP ring segment, as the DLR supervisor will recognize it as broken, and ultimately, in the inability to sniff and inject data into the ring messages.

Our configuration for the Ethernet ports and bridge for launching the attacks is the following:

```
auto eth0                        # Port 0
iface eth0 inet manual

auto eth1                        # Port 1
iface eth1 inet manual

# Bridge between Port 0 and Port 1
auto br0
iface br0 inet manual
        bridge_ports eth0 eth1
        bridge_stp off           # Disabling STP
        bridge_ageing 0          # Disabling MAC learning
```

3.3. Parsing and injection of manipulated data in EtherNet/IP ring packets

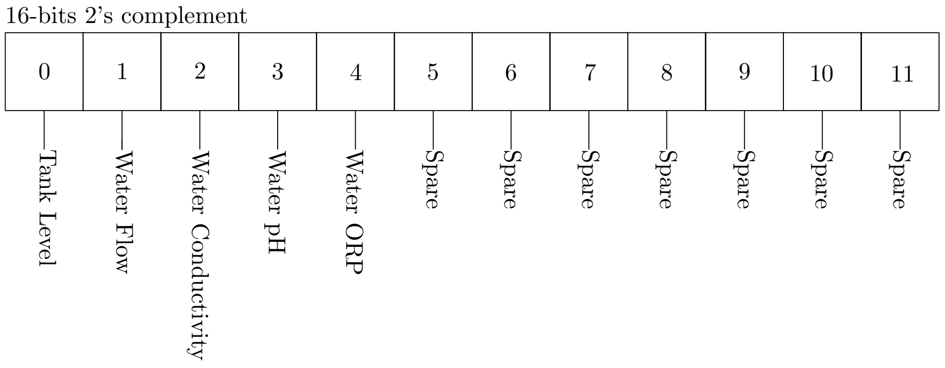


Figure 7. RIO's Analog Input Module 12 input signals (16-bits 2's complement per analog signal)

In SWaT, we identify three different device-dependent I/O implicit messages: one for each I/O module conforming the RIO. Figure 7 shows the I/O implicit message for the analog input module. It consists of a stream of 24 bytes, corresponding to 12 analog inputs channels of 16-bits. The *spare* channels are not in use by SWaT's current deployment. The digital input and output modules emit and receive bit streams, 32 bits and 16 bits respectively, where each bit corresponds to one digital signal. See table 1 for details on RIO modules.

Table 1. RIO I/O modules.

Module	Signal size (bits)	# signals	Avg. Freq. (ms)
Digital Input	1	32	50
Digital Output	1	16	60
Analog Input	16	12	80

The I/O implicit messages representing the analog signals are sent by the RIO to the PLC with an average frequency of 80 milliseconds. They transport the numeric representation of the 4-20 mA signals measured by the analog sensors. In order to scale back and forth the 4-20 mA signal to the real physical measurement we use Equation (1), which is a typical linear transformation (scaling and bias shift) to change the analog signal (4-20mA) into a physical meaningful quantity (e.g. the height of the water level in a tank being 0.5m). The constant values (*RawMin, RawMax, EUMax, EUMin*) depend on the deployment and the physical property being measured (we obtained the specific values for each constant from the HMI software of the testbed). Figure 8 shows an example for the scaling of the water flow in SWaT.

$$Out = (In - RawMin) * \frac{EUMax - EUMin}{RawMax - RawMin} + EUMin \qquad (1)$$

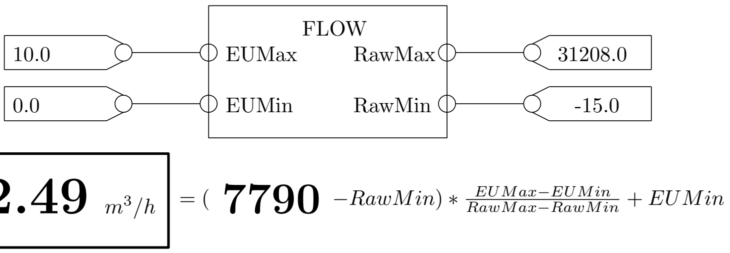


Figure 8. Scaling from 4-20 mA signals to water flow. The 4-20mA signal is scaled by the RIO to another value (7790 in this case) and this value sent over the network is the one we capture and convert to physical observations using Equation (1) with the respective constants for each signal.

3.4. Example: Stealthy Sensor Attack

We illustrate the feasibility of our proposed methodology by deploying a stealthy sensor attack in the first stage of the plant i.e., raw water storage. The raw water process consists of a storage tank with its water level sensor h_1, one valve that opens when $h_1 < 0.5\ m$ and closes when $h_1 > 0.8\ m$, and one pump whose action depends on the UF process. As a safety mechanism, if the water level in tank 1 is below 0.25 *m*, the pump is immediately Off. The attacker's goal is to overflow the water without being detected by a typical behavior-based detection mechanism using the "physics" of the system under to control to identify anomalies. Using the methodology described above, we gain access to the ring and we are able to modify the sensor and actuator information by constructing appropriate packets.

3.4.1. Attacker Action

Figure 9 depicts how an attacker can gain access to the sensor measure $h_i(k)$ and actuator command $u_i^{nom}(k)$ packets and modify them. In this example, the attack consists on injecting false information to the level sensor in tank 1. In particular, data injected is computed using $h_1^a(k) = h_1(k) + \delta(k)$. As result of this attack, the level of the water is decreasing all the time, i.e., $\delta(k) - \delta(k-1) = \Delta < 0$ is the rate at which the sensor information is modified.

3.4.2. Detection Mechanism

The detection mechanism also known as bad-data detection uses the residuals $r_i(k)$, which consists on the difference between the sensor measure $h_i(k)$ and its estimated $\hat{h}_i(k)$, such that $r_i(k) = |h_i(k) - \hat{h}_i(k)|$. For a sensor attack occurring at a time instant k^*, the residuals are then $r_i(k) = |h_i^a(k) - \hat{h}_i(k)|$ for all $k \geq k^*$. In our work, the estimated states are obtained by obtaining a mathematical approximation of the system behavior with a Luenberger observer (we refer to [16,17] for more details on system estimation and the bad-data detection method, which are out of the scope of this paper). When $r_i(k) > \tau_i$ for $\tau_i > 0$, an alarm is triggered indicating the presence of an attack. The main property of this type of detection is that it is based on the physical properties of the system and it can detect changes that violate those physical properties. For instance, Figure 10 depicts how the detection mechanism is able to trigger alarms for $\tau_1 = 0.1$ when an attack induces a sudden change in the sensor measurements, $h_1^a(k^*) = 0.1$ m, which yields to $r_1(k) > 0.1$. However, an intelligent adversary can remain stealthy by causing small changes in the sensor information.

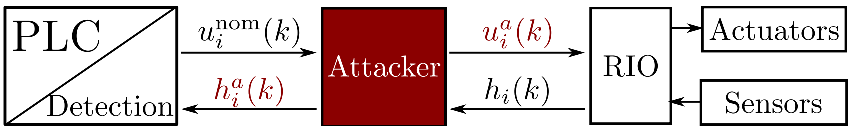


Figure 9. Detection mechanism and man in the middle attack for SWaT. $h_1^{(k)}$ is the attacked water level measure, $u_1^{nom}(k), u_1^a(k)$ are the real and attacked control command, respectively, for time instance k.

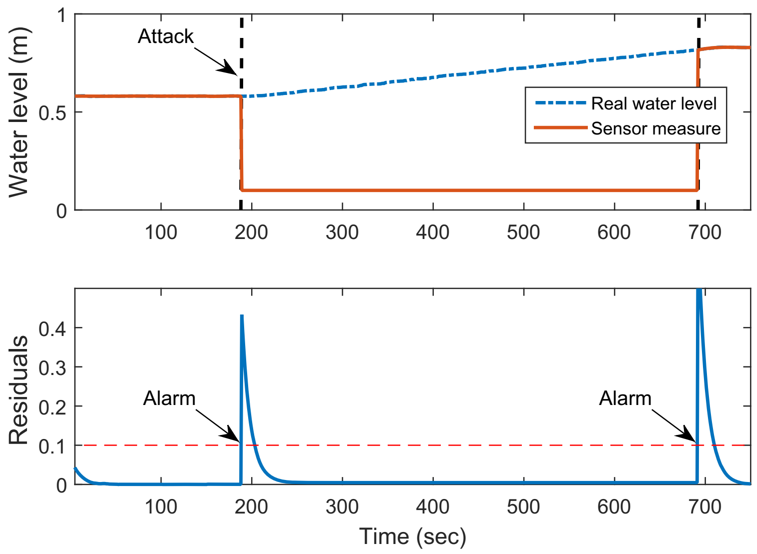


Figure 10. Sensor attack in the water level $h_1^(k)$. The attack is detected when $r_1(k) > 0.1$.

Figure 11 illustrates the effect of the stealthy sensor attack. Before the attack is launched, the valve was closed and the pump was ON, so the water level was decreasing.

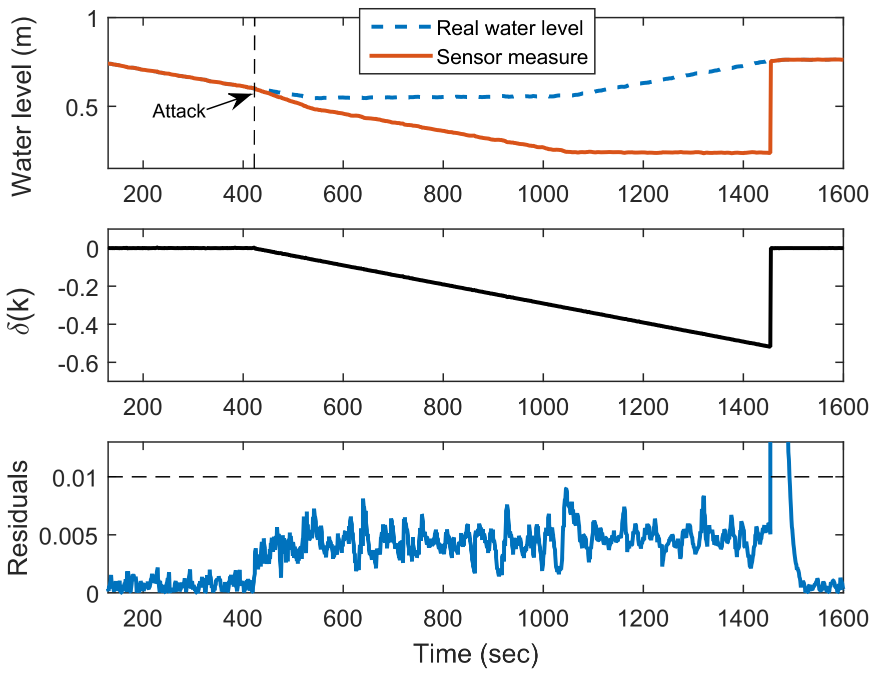


Figure 11. Effects of a stealthy sensor attack in the SWaT testbed. The PLC is connected to an intelligent verifier that analyzes the sensor measure and detects attacks based on sudden changes in the physical behavior. The attack was designed such that the bad-data detector with $\tau_1 = 0.01$ could not detect the attack.

As soon as the attack starts, the sensor measurement received keeps indicating that the water level is always decreasing (a little bit faster because the pump is ON), such that the valve opens when it reaches its minimum level $h_1^a(k) = 0.5\ m$. At that instant the pump continues pumping water such that the real water level remains constant for some time. When the pump stops ($h_1^a(k) < 0.25\ m$), the tank starts filling up with water (the water level $h(k)$ starts increasing) but the false sensor reading keeps indicating low water levels. The valve never closes and eventually it will yield to an overflow. Due to the small rate of change, the bad-data detection with a threshold $\tau_1 = 0.01$ never detects the attack. Smaller thresholds can detect the attack but also increase the number of false alarms. Besides, even if τ_1 is smaller, the attack can be designed to remain stealthy for small Δ.

4. Conclusions

In this work, we discussed practical MitM attacks on ICS Fieldbus communications. In particular, we provided details on the typical topologies (in particular, DLR) and protocols used in such a setting (EtherNet/IP). We also discussed practical challenges in setting up MitM attacks and how to overcome them, and demonstrated results of our practical attacks. We have shown that, although EtherNet/IP can be used as the overall encapsulation protocol, the protocol selection, and therefore, parsing and injection of manipulated data on the encapsulated message depends on the control system layer. SWaT presents CIP for Supervisory Control Network communications and a device-specific I/O messages for Fieldbus communications.

The MitM of an EtherNet/IP ring also presents several challenges: The attacker must verify that the attacking device incorporated into the ring does not break the ring permanently, as these events could potentially be monitored, i.e. the attacking device must not interfere with the DLR protocol. An attacker who successfully deploys a MitM device into a EtherNet/IP ring established for Fieldbus communications must be assumed to have access to all digital and analog signals. She could inject manipulated data in all sensors and actuators monitored and controlled by the ring's PLC at any time. Therefore, the adversary could effectively isolate and manipulate the physical process disregarding control actions sent by the control room or the PLCs. Using the proposed methodology to launch attacks, different detection mechanisms can be tested and improved. In addition, more robust communication infrastructures can be designed in order to decrease an attacker's impact over the system.

5. Acknowledgments

We thank SWaT's lab engineer Kaung Myat Aung for his valuable advise and help during the experimentation process. In addition, we thank Professor Aditya Mathur for giving us access to the SWaT testbed deployed at SUTD. The work from UT Dallas was supported by NIST under award 70NANB14H236 from the U.S. Department of Commerce.

References

[1] Ettercap Project. https://ettercap.github.io/ettercap/, October 2015.
[2] Python bindings for libnetfilter_queue. https://github.com/fqrouter/python-netfilterqueue, October 2015.
[3] Python Language. Version 2.7.10. https://docs.python.org/2/, October 2015.
[4] Scapy Packet Manupulation Program. Version 2.3.1. http://www.secdev.org/projects/scapy/doc/, October 2015.
[5] Wireshark Network Protocol Analyzer. https://www.wireshark.org/, October 2015.
[6] M. Abrams and J. Weiss. Malicious control system cyber security attack case study–maroochy water services, australia. Technical report, The MITRE Corporation, 2008.
[7] D. Albright, P. Brannan, and C. Walrond. Did stuxnet take out 1,000 centrifuges at the natanz enrichment plant? Technical report, Institute for Science and International Security, 2010.
[8] S. Amin, A. Cárdenas, and S. S. Sastry. Safe and secure networked control systems under denial-of-service attacks. In *Hybrid Systems: Computation and Control. Proc. 12th Intl. Conf. (HSCC '09), LNCS, Vol. 5469, Springer-Verlag*, pages 31–45, 2009.
[9] D. Antonioli and N. O. Tippenhauer. MiniCPS: A toolkit for security research on CPS networks. In *Proceedings of Workshop on Cyber-Physical Systems Security & Privay (CPS-SPC), co-located with CCS*, Oct. 2015.
[10] A. Banerjee, K. Venkatasubramanian, T. Mukherjee, and S. Gupta. Ensuring safety, security, and sustainability of mission-critical cyber-physical systems. *Proceedings of the IEEE*, 100(1):283 –299, Jan 2012.
[11] P. Brooks. EtherNet/IP: Industrial Protocol White Paper. Technical report, Rockwell Automation, 2001.
[12] A. A. Cárdenas, S. Amin, Z.-S. Lin, Y.-L. Huang, C.-Y. Huang, and S. Sastry. Attacks against process control systems: risk assessment, detection, and response. In *Proceedings of the 6th ACM symposium on information, computer and communications security*, pages 355–366. ACM, 2011.
[13] I. N. W. Group. Administratively Scoped IP Multicast. http://tools.ietf.org/html/rfc2365, October 2015.
[14] O. Kosut, L. Jia, R. Thomas, and L. Tong. Malicious data attacks on smart grid state estimation: Attack strategies and countermeasures. In *Proc. of the IEEE Conference on Smart Grid Communications (SmartGridComm)*, pages 220–225, Oct 2010.

[15] M. Krotofil and D. Gollmann. Industrial control systems security: What is happening? In *Industrial Informatics (INDIN), 2013 11th IEEE International Conference on*, pages 670–675. IEEE, 2013.

[16] Y. Liu, P. Ning, and M. K. Reiter. False data injection attacks against state estimation in electric power grids. *ACM Transactions on Information and System Security (TISSEC)*, 14(1):13, 2011.

[17] D. G. Luenberger. Observers for multivariable systems. *Automatic Control, IEEE Transactions on*, 11(2):190–197, 1966.

[18] ODVA. The CIP Networks Library Volume 2: EtherNet/IP Adaptation of CIP, 2007.

[19] L. Xie, Y. Mo, and B. Sinopoli. False data injection attacks in electricity markets. In *Proc. of the IEEE Conference on Smart Grid Communications (SmartGridComm)*, pages 226–231, Oct 2010.

Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016
A. Mathur and A. Roychoudhury (Eds.)
© 2016 The authors and IOS Press.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License.
doi:10.3233/978-1-61499-617-0-91

Directed Transitive Signature on Directed Tree[1]

Jia XU [a], Ee-Chien CHANG [b] and Jianying ZHOU [a]

[a] *Infocomm Security Department*
Institute for Infocomm Research, Singapore
e-mail: {xuj,jyzhou}@i2r.a-star.edu.sg
[b] *School of Computing*
National University of Singapore
e-mail: changec@comp.nus.edu.sg

Abstract. In early 2000's, Rivest [1,2] and Micali [2] introduced the notion of *transitive signature*, which allows a third party with public key to generate a valid signature for a composed edge (v_i, v_k), from the signatures for two edges (v_i, v_j) and (v_j, v_k). Since then, a number of works, including [2,3,4,5,6], have been devoted on transitive signatures. Most of them address the undirected transitive signature problem, and the directed transitive signature is still an open problem. S. Hohenberger [4] even showed that a directed transitive signature implies a complex mathematical group, whose existence is still unknown. Recently, a few directed transitive signature schemes [7,8] on directed trees are proposed. The drawbacks of these schemes include: the size of composed signature increases linearly with the number of nested applications of composition and the creating history of composed edge is not hidden properly. This paper presents a RSA-Accumulator [9] based scheme **DTTS**—a *Directed*-Tree-Transitive Signature scheme, to address these issues. Like previous works [7,8], **DTTS** is designed only for directed trees, however, it features with constant (composed) signature size and privacy-preserving property. We prove that **DTTS** is transitively unforgeable under adaptive chosen message attack in the standard model.

Keywords. Homomorphic Signature, Transitive Signature, Directed Transitive Signature, Redactable Signature, Privacy-Preserving

1. Introduction

In 2000, Rivest [1] introduced the notion of homomorphic signatures (formalized in [10, 11] etc.) and proposed an open problem on the existence of directed transitive signatures. Later, Micali and Rivest [2] proposed the first undirected transitive signature scheme, and raised the directed transitive signature as open problem again and officially. A transitive signature scheme aims to authenticate the transitive closure of a dynamically growing graph [7]. The scheme works in this way: a signer has a pair of public/private signing key, and is able to sign a new vertex or edge when it is generated at any time. Unlike standard digital signature, the transitive signature scheme supports a transitive property. That is,

[1]A full version is available at Cryptology ePrint Archive `https://eprint.iacr.org/2009/209`

given the signatures $\sigma_{i,j}$ and $\sigma_{j,k}$ of edges (v_i, v_j) and (v_j, v_k) respectively, anyone can produce a signature $\sigma_{i,k}$ for composed edge (v_i, v_k) using the public key only, where v_i, v_j, and v_k are vertices, and $(v_i, v_j), (v_j, v_k)$ are edges in a graph. If the graph is undirected, such scheme is called *undirected transitive signature* scheme; if the graph is directed, it is called *directed transitive signature* scheme. This paper attempts to attack the directed transitive signature problem in a restricted but meaningful setting: (1) The graph is a rooted directed tree (arborescence); (2) When composing two signatures of two adjacent edges, the second signature must be provided by the original signer.

Since Rivest's talk in 2000, a number of undirected transitive signature schemes [2,3, 5,6,12,13] have been proposed. However, the directed transitive signature is still an open problem [4,8], although some plausible directed transitive signature schemes [14,7,8] on restricted directed graphs, like directed tree, have been proposed. Y. Xun et al. [15] pointed out that Kuwakado-Tanaka transitive signature scheme [14] on directed trees is insecure under chosen message attack by proposing a forgery attack. Y. Xun [7] also proposed a transitive signature scheme **RSADTS** on directed trees , but the (composed) signature size is not constant. G. Neven [8] pointed out that it would be much easier to construct a directed transitive signature scheme (on directed tree) if the signature size is allowed to grow linearly, and gave a simple scheme as a demonstration. So far, to our knowledge, there is no known transitive signature scheme on directed trees, which is provably secure and has constant signature size. Table 1 and Table 2 compare various transitive signature schemes appeared in literatures with **DTTS** proposed in this paper, from different aspects.

Scheme	**Signing cost**	**Verification cost**	**Composi-tion cost**	**Signature size**	**Compos-ed Signature size**	**Supported Graph**
DLTS [2]	2 stand. sigs. 2 exp. in **G**	2 stand. verifs 1 exp. in **G**	2 adds in $\mathbb{Z}_q$	2 stand. sigs 2 points in **G** 2 points in $\mathbb{Z}_q$	constant	undirected graph
RSATS-1 [2]	2 stand. sigs. 2 RSA encs	2 stand. verifs 1 RSA enc.	$O(\lvert n\rvert^2)$ ops	2 stand. sigs. 3 points in $\mathbb{Z}_n^*$	constant	undirected graph
F*act***TS**-1 [6]	2 stand. sigs $O(\lvert n\rvert^2)$ ops	2 stand. verifs $O(\lvert n\rvert^2)$ ops	$O(\lvert n\rvert^2)$ ops	2 stand. sigs 3 points in $\mathbb{Z}_n^*$	constant	undirected graph
G*ap***TS**-1 [6]	2 stand. sigs 2 exp. in $\hat{\mathbb{G}}$	2 stand. verifs 1 S_{ddh}	$O(\lvert n\rvert^2)$ ops	2 stand. sigs. 3 points in $\hat{\mathbb{G}}$	constant	undirected graph
RSADTS [7]	2 stand. sigs 1 exp. in $\langle\mathscr{G}\rangle$	2 stand. verifs 1 exp. in $\langle\mathscr{G}\rangle$	$\leq \lvert M\rvert$ ops	2 stand. sigs 2 points in $\langle\mathscr{G}\rangle$ 1 label $\delta_{i,j} \leq M$	increase	directed tree
DTTS (This paper)	$\leq$ 2 stand. sigs 2 exp. in $\mathbb{Z}_n^*$	2 stand. verifs 2 exp. in $\mathbb{Z}_n^*$	1 exp. in $\mathbb{Z}_n^*$	2 stand. sigs. 3† points in $\mathbb{Z}_n^*$	constant	directed tree (Arborescence)

Table 1. Performance comparison among transitive signature schemes([6,7]). †: The left labels in a signature can be reduced using a hash function (See our full version [16]).

In **RSADTS**, each edge (i, j) is associated with a random number $r_{i,j}$ as the label. Given two adjacent edges (i, j) and (j, k) and their signatures, anyone with public key can produce a signature for the composed edge (i, k), whose label is the integer product $r_{i,j} \times r_{j,k}$. If we apply the transitive property recursively, the length of the label of the newly composed edge increases linearly with the depth of the recursion. Furthermore, the integer multiplication reveals some information about the creating history of the newly composed edge: if the original random numbers chosen by the signer are small, then adversaries could factorize the integer product; otherwise the bit-length of the product may reveal significant information about the number of multiplications, which implies the length of the path used to create the composed edge.

The directed transitive signature scheme **DTTS** on directed tree proposed in this paper, is inspired by the relation between transitive signature and redactable signature (Chang et al. [17]), and is different from previous schemes at least in these aspects: (1) It is provably secure under adaptive chosen message attack; (2) The length of signature of a composed edge is constant; (3) The creating history of a composed edge is hidden properly; (4) The directed tree supported by **DTTS** is slightly more restricted (precisely, every vertex has at most one incoming edge) than that of **RSADTS** (See Section 2); (5) When the transitive property is applied repeatedly on a path, for example path $i_1 \to i_2 \to i_3 \to i_4$, the order of nested applications is predetermined. That is, compose a signature

Scheme	**Assumptions for Provable Security**	**Privacy Preserving**	**How to grow?**	**Persis-tent Vertex?**
DLTS [2]	Security of standard signature scheme; Hardness of discrete logarithm in prime order group	Perfect,Transparent	Arbitrarily	No
RSATS-1 [2]	Security of standard signature scheme; RSA is secure against one-more-inversion attack	Perfect,Transparent	Arbitrarily	No
F*act*TS-1 [6]	Security of standard signature scheme; Hardness of factoring	Perfect,Transparent	Arbitrarily	No
G*ap*TS-1 [6]	Security of standard signature scheme; One-more gap Diffie-Hellman assumption	Perfect,Transparent	Arbitrarily	No
RSADTS [7]	Security of standard signature scheme; RSA Inversion Problem in a Cyclic Group is hard	No (due to integer multiplication)	From a single source	No
DTTS (This paper)	Security of standard signature scheme; Strong RSA Problem is hard	Computational,Non-Transparent	From a single source	Yes

Table 2. All of these schemes are transitive unforgeable under adaptive chosen-message attack in standard model [6].

for (i_1, i_3) first from signatures of edge (i_1, i_2) and edge (i_2, i_3), then compose a signature for (i_1, i_4) from signatures of edge (i_1, i_3) and edge (i_3, i_4). This is because, in **DTTS**, Comp requires the second edge is original, i.e. signed directly by the original signer. Note that the last difference does not restrict the power of transitive property of **DTTS**. Instead, this difference can be treated as a feature, and can be utilized to provide the signer with control on composition (See our full version [16] for details).

1.1. Contributions of this paper

Directed transitive signature is a hard open problem. We attack this problem from a different angle in a simplified but meaningful setting: (1) The graph is a directed tree (arborescence); (2) When composing two signatures of two adjacent edges, the second signature must not be a composed signature itself. The contributions of this paper include:

1. We present **DTTS**, a directed transitive signature scheme on directed trees with constant signature size (Section 3.1).
2. We prove that **DTTS** is transitively unforgeable under adaptive chosen message attack in standard model, and the creating history of composed signature is hidden properly (Section 3.2).

2. Definitions

Notations. Let $\mathbb{N} = \{1, 2, 3, 4, 5, \ldots\}$ be the set of integers. The notation $x \leftarrow a$ denotes that x is assigned a value a, and $x \overset{\$}{\leftarrow} S$ denotes that x is randomly selected from the set S. Let Prime be the set of all odd prime numbers.

Graph. Let $G = (V,E)$ be a simple directed graph with a set V of nodes (or vertices) v_i's and a set E of directed edges. In this paper, we focus on directed trees. Note that there exist different definitions of directed tree in the literature: (1)A directed tree is a directed graph that would be a (undirected) tree if ignoring the direction of edges; (2)A directed tree (or *Arborescence*) is a directed graph, where edges are all directed away from a particular vertex. The second definition is slightly more restricted than the first one. In this paper, we adopt the second definition for directed tree and the term "directed tree" refers to arborescence by default. Notice that Y. Xun [7] adopted the first definition of directed tree and G. Neven [8] adopted the second definition.

A *transitive closure* of a directed graph $G = (V,E)$, is a directed graph, denoted as $\widetilde{G} = (V,\widetilde{E})$, where $(v_i,v_j) \in \widetilde{E}$ if and only if there is a directed path from vertex v_i to vertex v_j in graph G.

Directed Transitive Signature Scheme. A directed transitive signature scheme **DTS** $=$ $(\mathsf{TKG}, \mathsf{TSign}, \mathsf{TVf}, \mathsf{Comp})$ is specified by four polynomial-time algorithms, and the functionality is as follows [6,7]:

- The randomized *key generation* algorithm TKG takes as input 1^k, where k is the security parameter, and returns a pair of keys (tpk,tsk), where tpk is the public key and tsk is the private key.
- The *signing* algorithm TSign could be randomized or/and stateful. TSign takes the private key tsk, two vertices v_i and v_j, and returns a value called an *original signature* of the edge (v_i,v_j) relative to tsk. If stateful, it maintains a state which it updates upon each invocation.
- The deterministic *verification* algorithm TVf, given tpk, two vertices v_i,v_j and a candidate signature σ, returns either `TRUE` or `FALSE`. We say that σ is a *valid signature* of edge (v_i,v_j) relative to tsk, if the output is `TRUE`.
- The deterministic *composition* algorithm Comp takes as input tpk, two directed edges (v_i,v_j) and (v_j,v_k) and two signatures $\sigma_{i,j}$ and $\sigma_{j,k}$, and returns either a *composed signature* $\sigma_{i,k}$ of the composed edge (v_i,v_k), or $\perp$ to indicate failure.

An edge e is called *original edge* if $e \in E$, or *composed edge* if $e \in \widetilde{E} - E$. All original edges are signed by the signer using TSign and tsk, and all composed edges could be indirectly signed by anyone using Comp and tpk.

Two different views of Transitive Signatures. Transitive signatures are originally designed to authenticate a transitively closed graph in an economic way, i.e. sign as least as possible number of vertices and edges to authenticate a transitively closed graph. Viewed from another angle, transitive signatures are actually redactable signatures on growing graph (Figure 1). The redaction operation can be implemented straightforwardly just using the composition operation Comp.

Correctness, Security and Privacy. We slightly modify the definitions of correctness and security of (directed) transitive signature scheme in [6,7] to adapt for **DTTS**. We also formalize the definition of privacy of transitive signatures when viewed as redactable signatures. Due to space constraint, we will leave details of these definitions to our full version [16].

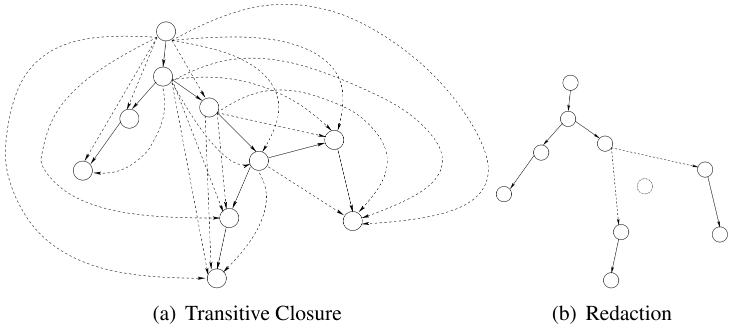


Figure 1. This graph illustrates the two different views of transitive property. In Subfigure (a), composed edges represented by dashed lines are signed indirectly by applying composition operation Comp. In this graph of 10 vertices and 29 edges, 9 original edges are signed directly using TSign, and the signatures of the other 20 composed edges (dashed line) can be saved due to transitive property. In Subfigure (b), a vertex represented by the dashed circle is redacted from the graph, and the edges connecting its parent and children are created and signed by applying Comp.

3. DTTS: Transitive Signature on Directed Tree

3.1. The scheme

Let **SDS** $= (\mathsf{SKG}, \mathsf{SSign}, \mathsf{SVf})$ be a standard signature scheme (For example, the signature scheme proposed by Goldwasser et al [18]). We define the directed transitive signature scheme **DTTS** $= (\mathsf{TKG}, \mathsf{TSign}, \mathsf{TVf}, \mathsf{Comp})$ as follows.

$\mathsf{TKG}(1^k)$. The key generation algorithm TKG taking 1^k as input, runs as follows:

1. Run $\mathsf{SKG}(1^k)$ to generate a key pair (spk, ssk).
2. Choose a RSA modulus $n = pq$, such that $p = 2p' + 1, q = 2q' + 1$, p, q, p' and q' are all prime, and $|p| = |q|$. Let Carmichael function $\lambda(n) = lcm(p-1, q-1)$.
3. Choose an element g from $\mathbb{Z}_n^*$, such that the multiplicative order of g modulo n is p'. Let $\langle g \rangle$ denote the subgroup of $\mathbb{Z}_n^*$ generated by g. Let $\mathscr{P}$ denote the set of all odd primes in $\mathbb{Z}_{p'}$, i.e. $\mathscr{P} = \mathbb{Z}_{p'} \cap \mathsf{Prime}$.
4. Output $tpk = (spk, n)$ as the public key and $tsk = (ssk, \lambda(n), p', g)$ as the private key.

$\mathsf{TSign}_{tsk}(v_i, v_j)$. The signing algorithm TSign maintains a state $(V, E, L, \Pi, \Delta, \Sigma)$:

- $V \subset \{0,1\}^*$ is a set of queried nodes;
- $E \subset V \times V$ is a set of directed edges;
- The function $L : V \to \mathscr{P} \times \mathbb{Z}_n^*$ assigns to each node $v \in V$ a public label $L(v)$, which consists of a prime (called left label, denoted as $L_{\mathscr{L}}(v)$) from $\mathscr{P}$ and an element (called right label, denoted as $L_{\mathscr{R}}(v)$) from $\mathbb{Z}_n^*$ $(L(v) \equiv (L_{\mathscr{L}}(v), L_{\mathscr{R}}(v)))$;
- The set Π records all prime numbers chosen in the signing process;
- The function $\Delta : E \to \mathbb{Z}_n^*$ assigns to each edge $(v_i, v_j) \in E$ a label $\delta_{i,j}$;
- The function $\Sigma : V \to \{0,1\}^*$ assigns to each node $v \in V$ a standard signature $\Sigma(v)$.

Initially, all of V, E and Π are empty sets. When invoked on input v_i, v_j $(v_i \neq v_j)$ and tsk, the signing algorithm TSign runs as follows:

1. Case 1: $v_i, v_j \notin V$, i.e. neither vertex v_i or vertex v_j is signed.

(a) Choose r_i randomly from $\mathscr{P} - \Pi$: $r_i \xleftarrow{\$} \mathscr{P} - \Pi$. Update Π: $\Pi \leftarrow \Pi \cup \{r_i\}$.
(b) The left label $L_{\mathscr{L}}(v_i)$ of v_i is: $L_{\mathscr{L}}(v_i) \leftarrow r_i$. The right label $L_{\mathscr{R}}(v_i)$ of v_i is: $L_{\mathscr{R}}(v_i) \leftarrow g^{r_i} \mod n$.
(c) Choose r_j randomly from $\mathscr{P} - \Pi$: $r_j \xleftarrow{\$} \mathscr{P} - \Pi$. Update Π: $\Pi \leftarrow \Pi \cup \{r_j\}$.
(d) The left label $L_{\mathscr{L}}(v_j)$ of v_j is: $L_{\mathscr{L}}(v_j) \leftarrow r_j$. The right label $L_{\mathscr{R}}(v_j)$ of v_j is: $L_{\mathscr{R}}(v_j) \leftarrow L_{\mathscr{R}}(v_i)^{r_j} \mod n$.
(e) $\Sigma(v_i) \leftarrow \mathsf{SSign}_{ssk}(v_i, r_i, L_{\mathscr{R}}(v_i))$; $\Sigma(v_j) \leftarrow \mathsf{SSign}_{ssk}(v_j, r_j, L_{\mathscr{R}}(v_j))$.
(f) The certificate of v_i is: $C(v_i) \leftarrow (v_i, r_i, L_{\mathscr{R}}(v_i), \Sigma(v_i))$. The certificate of v_j is: $C(v_j) \leftarrow (v_j, r_j, L_{\mathscr{R}}(v_j), \Sigma(v_j))$
(g) The label of the edge (v_i, v_j) is: $\Delta(v_i, v_j) \leftarrow g$.

2. Case 2: $v_i \in V, v_j \notin V$, i.e. vertex v_i is signed already but vertex v_j is not signed yet.

   (a) Let the certificate of v_i be $C(v_i) = (v_i, r_i, L_{\mathscr{R}}(v_i), \Sigma(v_i))$, where $r_i = L_{\mathscr{L}}(v_i)$.
   (b) Randomly choose r_j from $\mathscr{P} - \Pi$: $r_j \xleftarrow{\$} \mathscr{P} - \Pi$. Update Π: $\Pi \leftarrow \Pi \cup \{r_j\}$.
   (c) The left label $L_{\mathscr{L}}(v_j)$ of v_j is: $L_{\mathscr{L}}(v_j) \leftarrow r_j$. The right label of v_j is $L_{\mathscr{R}}(v_j) \leftarrow L_{\mathscr{R}}(v_i)^{r_j} \mod n$.
   (d) The certificate of vertex v_j is $C(v_j) \leftarrow (v_j, r_j, L_{\mathscr{R}}(v_j), \Sigma(v_j))$, where $\Sigma(v_j) \leftarrow \mathsf{SSign}_{ssk}(v_j, r_j, L_{\mathscr{R}}(v_j))$.
   (e) The label of the edge (v_i, v_j) is: $\Delta(v_i, v_j) \leftarrow L_{\mathscr{R}}(v_i)^{\frac{1}{r_i}} \mod n$.

3. Case 3: $v_i \notin V, v_j \in V$, i.e. vertex v_j is signed already but vertex v_i is not signed yet.

   (a) Let the certificate of v_j be $C(v_j) = (v_j, r_j, L_{\mathscr{R}}(v_j), \Sigma(v_j))$, where $r_j = L_{\mathscr{L}}(v_j)$.
   (b) Randomly choose r_i from $\mathscr{P} - \Pi$: $r_i \xleftarrow{\$} \mathscr{P} - \Pi$. Update Π: $\Pi \leftarrow \Pi \cup \{r_i\}$.
   (c) The left label $L_{\mathscr{L}}(v_i)$ of v_i is: $L_{\mathscr{L}}(v_i) \leftarrow r_i$. The right label of v_i is: $L_{\mathscr{R}}(v_i) \leftarrow L_{\mathscr{R}}(v_j)^{\frac{1}{r_j}} \mod n$.
   (d) The certificate of vertex v_i is: $C(v_i) \leftarrow (v_i, r_i, L_{\mathscr{R}}(v_i), \Sigma(v_i))$, where $\Sigma(v_i) \leftarrow \mathsf{SSign}_{ssk}(v_i, r_i, L_{\mathscr{R}}(v_i))$.
   (e) The label of the edge (v_i, v_j) is: $\Delta(v_i, v_j) \leftarrow L_{\mathscr{R}}(v_i)^{\frac{1}{r_i}} \mod n$.

For all cases, update V and E: $V \leftarrow V \cup \{v_i, v_j\}, E \leftarrow E \cup \{(v_i, v_j)\}$, and output $(C(v_i), C(v_j), \Delta(v_i, v_j))$ as the signature of (v_i, v_j).

$\mathsf{TVf}_{tpk}(v_i, v_j, \sigma_{i,j})$. The verification algorithm TVf, when revoked on input tpk, nodes v_i, v_j and a candidate signature $\sigma_{i,j}$ on directed edge (v_i, v_j), runs as follows:

1. Parse $\sigma_{i,j}$ as $(C_i, C_j, \delta_{i,j})$. Parse C_i as $(v_i, r_i, L_{\mathscr{R},i}, \sigma_i))$ and parse C_j as $(v_j, r_j, L_{\mathscr{R},j}, \sigma_j)$.
2. If $\mathsf{SVf}_{spk}((v_i, r_i, L_{\mathscr{R},i}), \sigma_i) = \mathtt{FALSE}$ or $\mathsf{SVf}_{spk}((v_j, r_j, L_{\mathscr{R},j}), \sigma_j) = \mathtt{FALSE}$, then reject.
3. Accept if $\delta_{i,j}^{r_i r_j} \equiv L_{\mathscr{R},j} \pmod{n}$.

$\mathsf{Comp}_{tpk}(v_i, v_j, v_k, \sigma_{i,j}, \sigma_{j,k})$. The composition algorithm Comp, when invoked on input tpk, nodes v_i, v_j, v_k, and two signatures $\sigma_{i,j}$ and $\sigma_{j,k}$, runs as follows:

1. Parse $\sigma_{i,j}$ as $(C_i, C_j, \delta_{i,j})$ and $\sigma_{j,k}$ as $(C'_j, C_k, \delta_{j,k})$.
2. If C_j and C'_j are different, output $\perp$ and abort.

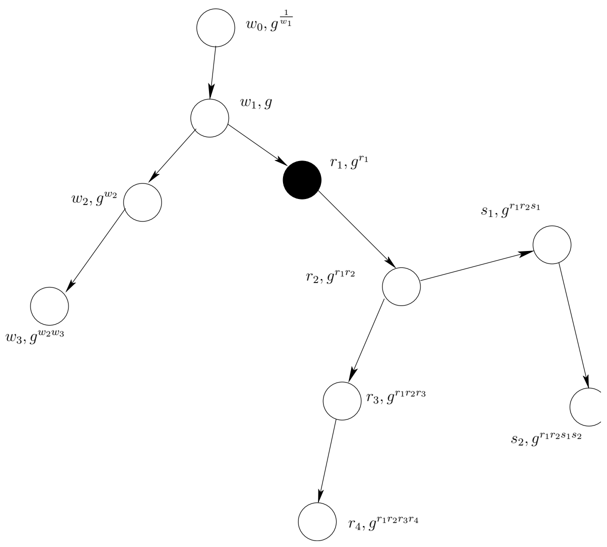


Figure 2. This figure shows the left label $L_{\mathscr{L}}(v)$ and right label $L_{\mathscr{R}}(v)$ associated with every vertex v. Note this graph grows from the vertex represented by the dark circle.

3. Parse C_i, C_j, C_k as $(v_i, r_i, L_{\mathscr{R},i}, \sigma_i), (v_j, r_j, L_{\mathscr{R},j}, \sigma_j)$ and $(v_k, r_k, L_{\mathscr{R},k}, \sigma_k)$ respectively.
4. If $\mathsf{SVf}_{spk}((v_i, r_i, L_{\mathscr{R},i}), \sigma_i) = \texttt{FALSE}$ or $\mathsf{SVf}_{spk}((v_j, r_j, L_{\mathscr{R},j}), \sigma_j) = \texttt{FALSE}$ or $\mathsf{SVf}_{spk}((v_k, r_k, L_{\mathscr{R},k}), \sigma_k) = \texttt{FALSE}$, output $\perp$ and abort.
5. If $L_{\mathscr{R}}(v_j)^{r_k} \not\equiv L_{\mathscr{R}}(v_k) \mod n$, output $\perp$ and abort[2].
6. Compute $\delta_{i,k} \leftarrow \delta_{i,j}^{r_j} \mod n$.
7. Output $(C_i, C_k, \delta_{i,k})$ as the signature of edge (v_i, v_k).

Figure 2 shows the left and right labels associated with every vertex v_i.

3.2. Security

Theorem 1. **DTTS** $= (\mathsf{TKG}, \mathsf{TSign}, \mathsf{TVf}, \mathsf{Comp})$ *as defined in Section 3.1 is transitively unforgeable under adaptive chosen message attack, assuming the standard signature scheme* **SDS** $= (\mathsf{SKG}, \mathsf{SSign}, \mathsf{SVf})$ *is unforgeable under adaptive chosen message attack and the Strong RSA problem is difficult.*

4. Conclusion

In this paper, we gave the first directed transitive signature scheme **DTTS** on directed trees, which is inspired by the relationship between transitive signatures and redactable signatures. Unlike previous schemes, **DTTS** features with constant signature size and privacy preserving property. We proved that **DTTS** is transitively unforgeable and non-transparently privacy-preserving under reasonable assumptions. In summary, we solved the open problem of directed transitive signature in a relaxed setting, although in general the directed transitive signature remains open problem.

[2]This means the Comp algorithm requires that the second edge (v_j, v_k) is an original edge, i.e. signed by the signer, instead of edge generated by composing a path.

References

[1] Rivest, R.: Two signature schemes (October 2000) Slides from talk given at Cambridge University.

[2] Micali, S., Rivest, R.L.: Transitive Signature Schemes. In: CT-RSA '02: Proceedings of the The Cryptographer's Track at the RSA Conference on Topics in Cryptology, London, UK, Springer-Verlag (2002) 236–243

[3] Bellare, M., Neven, G.: Transitive Signatures based on Factoring and RSA. In: ASIACRYPT '02: Proceedings of the 8th International Conference on the Theory and Application of Cryptology and Information Security, London, UK, Springer-Verlag (2002) 397–414

[4] Hohenberger, S.R.: The cryptographic impact of groups with infeasible inversion. Master's thesis, MIT (2003)

[5] Siamak Fayyaz Shahandashti, M.S., Mohajeri, J.: A Provably Secure Short Transitive Signature Scheme from Bilinear Group Pairs. Security and Communication Networks **3352** (2005) 60–76

[6] Bellare, M., Neven, G.: Transitive signatures: new schemes and proofs. Information Theory, IEEE Transactions on **51**(6) (June 2005) 2133–2151

[7] Yi, X.: Directed Transitive Signature Scheme. In: CT-RSA. Volume 4377 of Lecture Notes in Computer Science., Springer Berlin / Heidelberg (2007) 129–144

[8] Neven, G.: Note: A simple transitive signature scheme for directed trees. Theor. Comput. Sci. **396**(1-3) (2008) 277–282

[9] Benaloh, J., de Mare, M.: One-way accumulators: a decentralized alternative to digital signatures. In: EUROCRYPT '93: Workshop on the theory and application of cryptographic techniques on Advances in cryptology. (1994) 274–285

[10] Johnson, R., Molnar, D., Song, D.X., Wagner, D.: Homomorphic Signature Schemes. In: CT-RSA '02: Proceedings of the The Cryptographer's Track at the RSA Conference on Topics in Cryptology, London, UK, Springer-Verlag (2002) 244–262

[11] Ateniese, G., Chou, D.H., de Medeiros, B., Tsudik, G.: Sanitizable Signatures. In: ESORICS. (2005) 159–177

[12] Shahandashti, S.F., Salmasizadeh, M., Mohajeri, J.: A provably secure short transitive signature scheme from bilinear group pairs. In: Security in Communication Networks. Volume 3352. (2005) 60–76

[13] Wang, L., Cao, Z., Zheng, S., Huang, X., Yang, Y.: Transitive signatures from braid groups. In: INDOCRYPT. (2007) 183–196

[14] Kuwakado, H., Tanaka, H.: Transitive signature scheme for directed trees. IEICE Trans. Fundamentals (2003)

[15] Yi, X., Tan, C., Okamoto, E.: Security of Kuwakado-Tanaka Transitive Signature Scheme for Directed Trees. IEICE Trans. on Fundamentals (2004)

[16] Xu, J., Chang, E.C., Zhou, J.: On directed transitive signature. Cryptology ePrint Archive, Report 2009/209 (2009) http://eprint.iacr.org/.

[17] Chang, E.C., Lim, C.L., Xu, J.: Short redactable signatures using random trees. In: CT-RSA. (2009) 133–147

[18] Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM J. Comput. **17**(2) (1988) 281–308

Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016
A. Mathur and A. Roychoudhury (Eds.)

doi:10.3233/978-1-61499-617-0-99

A Framework for Large-Scale Collection of Information from Smartphone Users based on Juice Filming Attacks

Weizhi MENG [a,1], Wang Hao LEE [a] and S. P. T. KRISHNAN [a]
[a] *Infocomm Security Department, Institute for Infocom Research, Singapore*

Abstract. Cyber security refers to protecting computers, networks, programs and data from unauthorized access, change or destruction. With the wide adoption of smartphones, a lot of sensitive information can be cleaned from users' interaction with their smartphones and tablets. In our previous effort, we have developed a type of charging attacks called *juice filming attacks*, which can sniff smartphone users' information when they are interacting with their phones during charging. With the increasing number of public charging stations, we notice that this type of charging attacks may become a big threat to compromise users' privacy. In this work, we propose a framework of collecting smartphone information in large-scale based on juice filming attacks. Then, we show that such charging attacks work well not only on Android phones, but also on the newly released iPhones. Our work points out that the proposed framework is feasible to threaten users' privacy in practice.

Keywords. Cyber Security, Smartphone Security, Charging Attacks,

1. Introduction

Cyber security research aims to protect networks, computers, programs and data from attack, damage or unauthorized access. It includes applications security, information security, network security and so on. With the increasing adoption of smartphones, more recent research has focused on mobile security and the data stored in these phones has become a major target for hackers.

As an example, Xu *et al.* [13] investigated video-based vulnerabilities in 3G Smartphones and designed a video-based spyware, called Stealthy Video Capturer (SVC), which allows hackers to automatically activate the built-in camera on 3G Smartphones. They then implemented the spyware and conducted experiments on real world 3G smartphones. The results showed that SVC spyware can capture private video information with unremarkable power consumption, CPU and memory footprint. There are also several surveys about smartphone malware such as [1,2,10].

However, we find that few studies in the literature give attention to charging attacks that could steal information through charging facilities. In our previous research [9], we

[1] Corresponding Author: Infocomm Security Department, Institute for Infocom Research, Singapore; E-mail: mengw@i2r.a-star.edu.sg.

developed a new type of charging attacks called *juice filming attacks* on smartphones, which can automatically video-capture screen information of smartphone users during charging process. Based on the captured video, it is feasible to extract huge amounts of sensitive information such as PIN code, emails accounts and various passwords. Moreover, our attack is scalable on both Android OS and iOS, distinguishing this type of attacks from malware attacks.

Motivation. With the increasing demands, public charging stations are becoming popular and widely available in our daily lives. For example, Singapore Power (SP) promised to provide 200 free mobile charging stations for SG50 [12]. As it said, these stations will be launched progressively in busy locations including hospitals, tertiary institutions, libraries and supermarkets, and will become available for one to two years. More specifically, each station will be equipped with 10 individual slots, which contain multiple charging connectors such as mini and micro USBs which will fit most mobile phones and tablets. These charging facilitates can greatly benefit smartphone users, but also open a hole for hackers. This work aims to emphasize that more attention should be given to charging attacks.

Contributions. In this work, we focus on proposing a proof-of-concept framework to show how to collect a large-scale smartphone users' information based on *juice filming attacks*, aiming to explain the potential impact of such attacks on smartphone users in practice. The contributions of our work can be described as below:

- Firstly, we propose a framework to launch *juice filming attacks* in larger-scale, which can greatly threaten smartphone users' privacy in public places. In particular, we take Singapore MRT stations as an example and illustrate how to widely collect users' information with cloud computing. We point out that our framework is feasible to deploy in practice and would be even more severe when utilized by large agents or organizations.
- To illustrate the attack capability, we further verify *juice filming attacks* on recently released Android and iOS devices. It is found that our attack works well in these newly developed platforms. To mitigate it, we also discuss some countermeasures. Our effort demonstrates that charging attacks may become a major threat for cyber security and attempts to stimulate more attention to this area.

The remaining parts are organized as follows. Section 2 briefly introduce the principles of *juice filming attacks*. We describe our framework and show how to implement it through an example in Section 3. In Section 4, we validate the effect of *juice filming attacks* on newly released mobile operating systems and phones. Then, we discuss some countermeasures in Section 5. Finally, we conclude this work in Section 6.

2. Background of Juice Filming Attacks

In [9], juice filming attack was developed, which can refer users' private information through automatically video-capture smartphone screens when users are interacting with their phones during charging. It does not need to install any additional apps or ask for any permissions. Overall, it can provide six features as below:

- It is easy to implement but quite efficient.
- It is a kind of passive attack, which receives less user awareness.

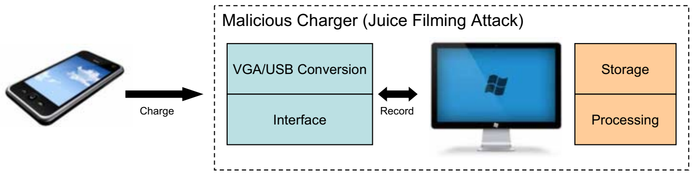


Figure 1. The high-level architecture of juice filming attacks.

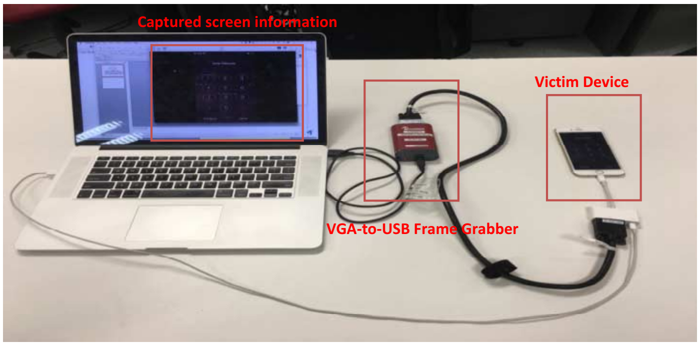


Figure 2. The real setup for juice filming attacks using VGA2USB.

- It does not need to install any additional apps or components on phones.
- It does not need to request any permissions.
- It will not be detected or notified by any current anti-malware software.
- It can be scalable and effective in both Android OS and iOS.

Threat model. The basic assumption is that most smartphone users would not treat public chargers as highly sensitive or dangerous. This assumption is reasonable since it is easy to see lots of smartphone users charge their phones in public places such as airports, subways and so on.

Basic idea. We observe that presently, no permissions will be asked when plugging iPhones or Android phones to a projector, while the projector can display the phone screen. In addition, there are no compelling notifications on the screen when the device is being plugged, or the indicators are very small and last only few seconds. Based on these, *juice filming attacks* can automatically video-record users' inputs during charging by using a VGA/USB interface. This attack reveals that the display can be leaked through a standard micro USB connector through the Mobile High-Definition Link (MHL) standard. For iPhones, the lighting connector is used.

The high-level architecture of such attacks is depicted in Figure 1. When users charge their phones to juice filming charger facilities, their phone screens can be video-captured by the malicious charger. These sensitive videos can be stored and processed in the cloud to conserve compute and network resources.

Real setup. To implement juice filming attack, choosing an appropriate VGA/USB interface is critical as there are many alternatives online. In the previous study [9], we employ a hardware interface called *VGA2USB* from Epiphan system Inc.[2] The *VGA2USB* is

[2] `http://www.epiphan.com/products/vga2usb/`.

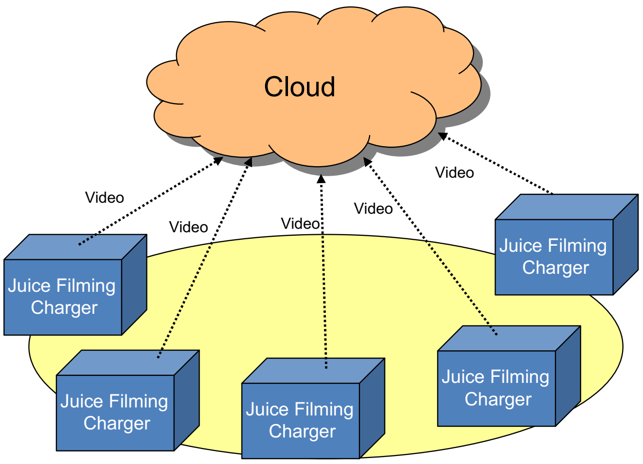


Figure 3. A framework to conduct a large-scale juice filming attack using cloud.

particularly a full-featured VGA/RGB frame grabber, which can send a digitized video signal from VGA to USB.

The real setup is shown in Figure 2. It can be seen that the connected iPhone's screen activity can be shown in the computer end. In this case, it is easy to imagine that all user interaction on the device screen would be captured by our attack. It is worth noting that the computer and other cables can be crampted into a portable charger, making the attack mobile and even more transparent.

3. Our Proposed Framework

In [9], we have shown the feasibility and effectiveness of *juice filming attacks* in a local charging station. In this section, we aim to illustrate that this type of attacks may become a big threat for cyber security through connecting to a cloud environment.

In Figure 3, we present a framework to show how to conduct *juice filming attacks* in large-scale. It is seen that the impact of such attacks can be magnified when deploying malicious chargers to various public locations such as MRT stations, airports and so on. The malicious chargers can upload recorded videos to a cloud environment and extract sensitive information. It is worth noting that this framework is very effective if utilized by a large organizations as large amount of video information can be collected through various charging points. We point out that this attack can be a big threat to users' privacy with the increasing adoption of public charger stations. We summarize some features of this framework as follows.

- *Automatic recording.* Our attack is able to automatically check whether phones are charging to the station and decide when to start video-recording periodically. The script of checking and video-capturing phone screens is shown in Figure 4.
- *Sustainable capturing.* Different from conducting *juice filming attacks* in local stations, our framework make such attack sustainable in capturing videos. For example, it is feasible to delete all videos after uploading them to a cloud, which can save much space.

```
#!/bin/bash
while [ 1 ]
do
        now="$(date +%s)"
        ./ffmpeg -probesize 32 -framerate 25 -analyzeduration 1 -f v4l2 -t 00:00:05 -i
/dev/video0 output/$now.mkv
        sleep 1
done
```

Figure 4. Source code for automatically checking and video-capturing phone screens in period time.

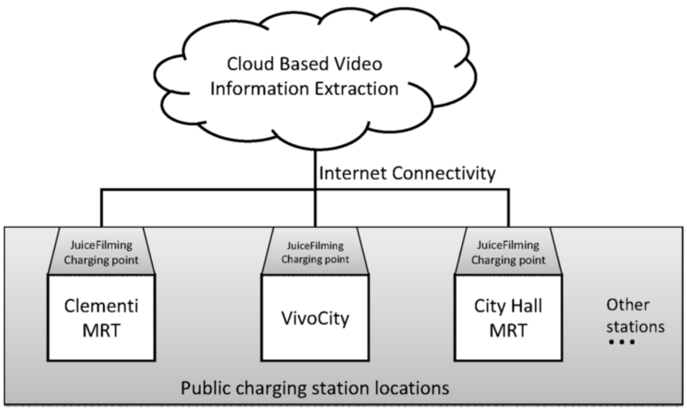

Cloud Based Video Information Extraction
Internet Connectivity
JuiceFilming Charging point
Clementi MRT
JuiceFilming Charging point
VivoCity
JuiceFilming Charging point
City Hall MRT
Other stations
Public charging station locations

Figure 5. A realization of our framework in Singapore MRT stations.

- *Large impact.* The framework employs a distributed deployment, which can collect information in large-scale. The recorded videos can be processed in one cloud or multiple clouds. Due to very large crowd in public places, our framework can make a large impact on compromising users' privacy.

Based on the above proposed framework, we take Singapore MRT as a hypothetical example to show how to conduct a large-scale *juice filming attack*. As shown in Figure 5, we deploy the juice filming charging stations to various MRT stations such as Clementi, City Hall, etc. All these malicious chargers are connected to a cloud (i.e., uploading recorded videos) through Wi-Fi. In the cloud, all videos can be processed to extract sensitive information including PIN, Android unlock patterns, account names, email content, chat history and so on. According to the surveys conducted in [9], most users (over 95%) have no idea about charging attacks so that they would pay no attention to charging station security. Due to less user awareness, our framework can be feasible in practice and make a large impact.

4. Verification on Newly Released Platforms

In this section, we validate the effect of *juice filming attacks* on various mobile platforms like Samsung Android phones, iPhone 6s and iPad, and recent operating system versions like Android OS 5.0.1 and iOS 9. Several mobile platforms are shown in Figure 6 including iPhone 6s, Samsung Note 4 and iPad.

Figure 6. Several mobile facilities used in our evaluation including iPhone 6s, Samsung Note 4 and iPad.

4.1. *iOS Validation*

Apple launched the new iPhone 6s and 6s Plus at the time of writing this paper. This motivates us to validate our attack on this newly released platform. The iPhone 6s we obtained is also updated to the latest version of iOS 9. Several screenshots are shown in Figure 7 and present that our attack works well on the iPhone 6s.

- Figure 7 (a) shows the screenshots for Facebook apps. It is easily seen that contact names and relevant information can be captured. After processing the recorded videos, it is feasible to infer private information of users and his or her friends.
- Figure 7 (b) shows that our attack can record the whole process of users' passcode input, so that we can know the actual passcode of that user. Similarly, our attack is able to record various passwords since the video would record all typing actions on the software keyboard.

4.2. *Android Validation*

For the Android platform, we employ Samsung Note 4 in this evaluation with Android version 5.0.1. Two screenshots are shown in Figure 7.

- Figure 7 (c) shows the screenshots for message apps. It is visible that all messages shown on the screens can be captured on video. After processing these videos, it is feasible to extract private dialogue and sensitive information such as credit card numbers.
- Figure 7 (d) shows that our attack can record Android unlock patterns. Similarly, all input passwords during charging process can be recorded by our attack.

It is found that the newly released Android phones and iPhones did not pay much attention to such type of charging attacks, which still opens a hole for conducting juice filming attacks in large-scale.

5. Countermeasures

As described in [9], the root cause of our attack is that Android OS and iOS allow screen mirroring *without explicit permission at the discretion of the user*. In this case, we point

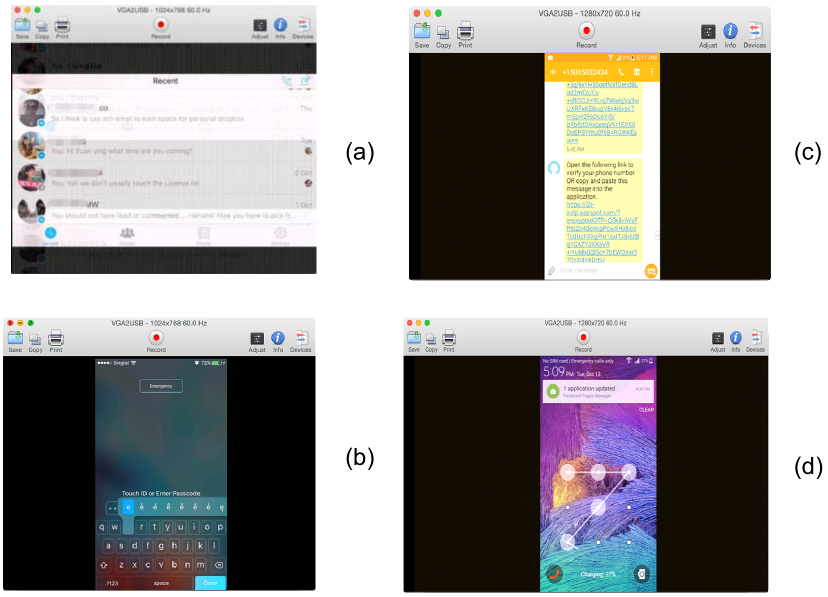

VGA2USB - 1024x768 60.0 Hz
VGA2USB - 1280x720 60.0 Hz
Save
Copy
Print
Record
Adjust
Info
Devices
Recent
+15015032434
Open the following link to verify your phone number. OR copy and paste this message into the application.
Emergency
Touch ID or Enter Passcode
5:09 PM Tue, Oct 13
1 application updated
CLEAR
Charging: 27%
(a)
(b)
(c)
(d)

Figure 7. Examples of screenshots on iPhone 6s: (a) Facebook and (b) Passcode, and Samsung Note 4: (c) Message app and (d) Android unlock patterns.

out that more attention should be given to such attacks, which may cause a huge threat for cyber security. Also, it is very critical to protect users' privacy by taking proper countermeasures to defend against such attacks. We discuss several mitigation strategies as follows.

- *Making notifications.* When connecting iPhones to a computer, iOS should prompt a notification asking whether to trust the computer or not. Similarly, to defend against our attack, the smartphone should also warn and make notifications before output of the display. This strategy can increase user awareness and let users consider their actions carefully.
- *Securing the charger interface.* As described in [9], one potential solution to defend against such charging attacks is to use a safe charger such as USB Condom [11]. This USB can prevent accidental data exchange when the device is plugged into another device with a USB cable, by cutting off the data pins in the USB cable and allowing only the power pins to connect through.
- *Employing biometrics.* It is feasible to defend such attacks by combining other techniques like biometrics. For example, behavioral biometrics can be added to the process of inputting PIN code and unlock patterns (i.e., building a fingerprint-based unlocking mechanism), then our attack cannot easily capture these secrets. Several behavioral-based authentication can be referred to [3,5,6,8].
- *Educating users.* Until patches are issued by vendors, such charging attacks are hard to control by filtration mechanisms [7], thus, user education should be a necessary countermeasure to raise users' attention on this threat. It is also a widely used method for securing systems related to human factors such as spam detection [4] & phishing.

6. Conclusion

Charging attacks are often ignored by the literature. In this paper, we proposed a framework to largely collect smartphone users' information by means of juice filming attacks. This type of attacks can extract users' private information through automatically video-capture smartphone screens when users are interacting with their phones during charging. Based on such attacks, we show the feasibility of our framework in practice, which would greatly threaten users' privacy in collaboration with cloud environments, and validate its effectiveness on recently released iPhones and Android phones. Our work aims to stimulate related research in this area and raise user awareness of such attacks.

References

[1] P. Faruki, A. Bharmal, V. Laxmi, and V. Ganmoor: Android Security: A Survey of Issues, Malware Penetration, and Defenses, *IEEE Communications Surveys and Tutorials* 17(2), pp. 998-1022, 2014.

[2] A.P. Felt, M. Finifter, E. Chin, S. Hanna, and D. Wagner: A survey of mobile malware in the wild, *Proceedings of the ACM Workshop on Security and privacy in smartphones and mobile devices (SPSM)*, pp. 3-14, ACM, New York, NY, USA, 2011.

[3] M. Frank, R. Biedert, E. Ma, I. Martinovic, and D. Song: Touchalytics: On the Applicability of Touchscreen Input as a Behavioral Biometric for Continuous Authentication, *IEEE Transactions on Information Forensics and Security* 8(1), 136-148, 2013.

[4] W. Li and W. Meng: An Empirical Study on Email Classification Using Supervised Machine Learning in Real Environments, *Proceddings of the 2015 IEEE International Conference on Communications (ICC)*, IEEE, pp. 7438-7443, 2015.

[5] Y. Meng, D.S. Wong, R. Schlegel, and L.F. Kwok: Touch Gestures Based Biometric Authentication Scheme for Touchscreen Mobile Phones, *Proceedings of the 8th China International Conference on Information Security and Cryptology (INSCRYPT)*, pp. 331-350, 2012.

[6] Y. Meng, W. Li, and L.F. Kwok: Enhancing Click-Draw based Graphical Passwords Using Multi-Touch on Mobile Phones, *Proceedings of the 28th IFIP TC 11 International Information Security and Privacy Conference (IFIP SEC)*, Springer, pp. 55-68, 2013.

[7] W. Meng, W. Li, and L.F. Kwok: EFM: Enhancing the Performance of Signature-based Network Intrusion Detection Systems Using Enhanced Filter Mechanism, *Computers & Security* 43, pp. 189-204, Elsevier, 2014.

[8] W. Meng, D.S. Wong, S. Furnell, and J. Zhou: Surveying the Development of Biometric User Authentication on Mobile Phones, *IEEE Communications Surveys & Tutorials*, 2015.

[9] W. Meng, W.H. Lee, S.R. Murali, and S.P.T. Krishnan: Charging Me and I Know Your Secrets! Towards Juice Filming Attacks on Smartphones, *Proceedings of the ACM Workshop on Cyber-Physical System Security (CPSS)*, pp. 89-98, ACM, New York, NY, USA, 2015.

[10] S. Peng, S. Yu, and A. Yang: Smartphone malware and its propagation modeling: A survey, *IEEE Communications Surveys and Tutorials* 16(2), pp. 925-941, 2014.

[11] The Original USB Condom. `http://int3.cc/products/usbcondoms.`

[12] Singapore Power to provide 200 free mobile phone charging stations for SG50. (July 27, 2015) `http://www.straitstimes.com/singapore/singapore-power-to-provide-200-free-mobile-phone-charging-stations-for-sg50`

[13] N. Xu, F. Zhang, Y. Luo, W. Jia, D. Xuan, and J. Teng: Stealthy Video Capturer: a new video-based spyware in 3g smartphones, *Proceedings of the 2nd ACM Conference on Wireless Network Security (WiSec)*, pp. 69-78, ACM New York, NY, USA, 2009.

Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016
A. Mathur and A. Roychoudhury (Eds.)

doi:10.3233/978-1-61499-617-0-107

Telephone-based social engineering attacks: An experiment testing the success and time decay of an intervention

Jan-Willem Bullée [a], Lorena Montoya [a], Marianne Junger [a] and Pieter H. Hartel [a]
[a] *University of Twente, Enschede, The Netherlands*

Abstract. The objective of this study is to evaluate the effectiveness of an information campaign to counter a social engineering attack via the telephone. Four different offenders phoned 48 employees and made them believe that their PC was distributing spam emails. Targets were told that this situation could be solved by downloading and executing software from a website (i.e. an untrusted one). A total of 46.15 % of employees not exposed to the intervention followed the instructions of the offender. This was significantly different to those exposed to an intervention 1 week prior to the attack (9.1 %); however there was no effect for those exposed to an intervention 2 weeks prior to the attack (54.6 %). This research suggests that scam awareness-raising campaigns reduce vulnerability only in the short term.

Keywords. Awareness, Decay, Scam, Social Engineering, Retention, Telephone, Time, Training

1. Introduction

Since 2008 a scam has been carried out by employees claiming to belong to Microsoft's technical department [1]. A phone call is received unexpectedly at home; the caller introduces himself and proceeds to inform the home owner that there is a virus on the PC or that the PC is distributing spam emails. To verify the claim from the caller, the victim is persuaded to open a remote desktop session to review some warnings in the system log files. In order to resolve the problem, the caller advises the victim to buy a small software tool to prevent loosing valuable data. The solution can be bought on their website and payment is possible via either credit card or PayPal. When the victim next checks the bank account, he/she discovers that the savings have disappeared. The victim thinks that he/she is securing the system whilst in fact the opposite has taken place.

Since 2011, the Dutch Fraud Help Desk (an organisation that collects fraud data) has received 4000 reports of this Microsoft technical support scam in The Netherlands. In 2014, there were 856 people who filed a complaint, and 88 (10.28 %) of them admitted to have paid the scammers. From January until September 2015 there were already 1099 complaints filed, of which 154 (14 %) involved payment. This constitutes a significant increase in both *i*) the number of filed complaints ($z = 31.396$, $p = .000$) and *ii*) the number of victims (z = 30.130, p = .000) for a short period of time. These numbers indicate that people are vulnerable to social engineering via the telephone.

Information security has been treated as a technical problem for many years, resulting mainly in technical solutions [2] and overlooking the human aspect [3]. Since information technology becomes more integrated into our daily activities, security experts propose that social engineering will be the greatest threat to any security system [2]. One example of social engineering is the technical support scam [4].

This paper therefore explores: *i*) the extent to which people are susceptible to telephone-based social engineering attacks when they are persuaded to go to a website and download a software and, *ii*) the effectiveness of an intervention to reduce the effects of social engineering over time.

1.1. Informing people to change behaviour

The Elaboration Likelihood Model (ELM) of Persuasion describes how information is processed and can be tailored to the receivers [5]. According to the ELM, people process information via either the peripheral or the central route. The peripheral route of information processing is used when there is minimal attention to the message and can involve superficial cues, such as the attractiveness of the message presented. One may like the sound of a person's voice, or that person might have gone to the same university as one did. The central route, on the other hand, involves persuasion on the basis of the message's content, such as voting for the political party with the best arguments [6]. Consistent with the ELM theory, attitudes obtained via the central route last longer, are less vulnerable to contra-argumentation and are better predictors of human behaviour. Furthermore, the effect of persuasive communication increases if the message is relevant to the audience and if surprises and repetitions are used [5].

Studies about leaflets have successfully increased the knowledge of the general public [7], as well as more specific groups such as patients [8], parents [9], and customers [10]. Experts argue that both training and education can help protect users against phishing [11]. It is noted that the currently available intervention materials can be effective as long as the user actually reads the material [11].

Gadgets can be used as subtle reminders of a campaign and are mentioned in the theory of Situational Crime Prevention as elements that 'remove excuses' [12]. Although the literature on reminder cues is limited, research suggests that these are effective [13]. In previous work we showed that exposing university personnel to both an information leaflet and a subtle reminder contributes to a significant reduction in the success rate of a social engineering attack [14].

1.2. Retention

In 1885 Herman Ebbinghaus demonstrated the existence of memory decay over time [15]. In addition, he described the 'forgetting curves', which refer to the amount of new information one is able to learn with each repetition of the same content. The amount of new information learned after each repetition decreases less steeply, meaning that more and more information stays in one's memory [15]. Retention can relate to knowledge but it can also relate to skill and is not limited to a single context as shown in the examples below.

Knowledge Ebbinghaus showed, using a list of 3 letter nonsense syllables (starting and ending with a consonant and a resonant in the middle), that over time, more syllables are forgotten [15]. Since the publication of this work, other research areas have looked into this topic as well, such as in the fields of aeronautics and medicine.

The aeronautical knowledge of 60 pilots was tested with a multiple-choice test, with items randomly selected from the Federal Aviation Administration (FAA) private pilot item bank of questions. A negative correlation was found between test scores and number of months since each pilot's last flight review (r = -.44, df = 18, t = 1.96, $p < .05$). This means that pilots who recently completed their flight reviews were associated with higher scores, compared to those with a longer time since the review [16].

In the field of medicine it was shown that the knowledge regarding Cardiopulmonary Resuscitation (CPR) dropped significantly over a period of 10 weeks. In this test 19 health care professionals (i.e. nurses) were tested via a 26 open item CPR theoretical questionnaire [17].

Skill The flight manoeuvres of a group of 192 pilots were tested in a full-motion flight simulator 6 and 12 months after training. During the test the pilots had to perform 12 emergency manoeuvres and 25 normal manoeuvres. The results showed there was a significant decay in performance between the 6 and 12 month group [18].

18 medical students were assessed on their CPR performance at three points in time: a) a pre-test to establish the baseline score, b) a training phase, c) a post-test and d) a re-test after 10 weeks. The test scores between the post- and re-test differed significantly, confirming skill decay [19].

In the context of a phishing test, the skill retention of adult subjects was measured 7 days after an information campaign [11]. During the test, the subjects had to classify 6 emails (as phishing emails or not) at 3 points in time (i.e. pre-test, post-test and re-test). This study did not find a significant decay of performance in classifying phishing emails.

This suggests that skill after training remains stable on the short term, but not on the long term. A further question to be answered is: how do the effects of training decay between 1 and 10 weeks?

1.3. Research Question

The objective of this research was to find out: "*How susceptible are people to a technical support scam?*". The following hypothesis was formulated: **H1)** "Time influences the relation between compliance and intervention."

2. Method

The sample consisted of 48 subjects of both sexes who worked in one particular building on the campus of the University of Twente. All subjects were scientific personnel of 9 research groups. Only approached were those who *i*) had an office work space and *ii*) were present to answer the telephone.

2.1. Subject Selection

Professors, secretaries, support and laboratory staff were excluded from the experiment in order to minimize disruption of primary activities. The pool of subjects therefore consisted of PhD-candidates, Post-Doc researchers as well as Assistant and Associate professors. The sample consisted of 34.29 % of all possible targets; the nationality distribution was comparable to the overall nationality distribution of the faculty, while those in the experimental sample were slightly younger (38 vs. 41 years).

2.2. Researchers

The researchers (i.e. the "offenders") consisted of 4 bachelor students (2 female and 2 male). The age of the researchers ranged between 21 and 24, the average age was 22.25 years (SD = 1.26). All researchers were Dutch nationals. There was no restriction with regards to approaching subjects of the same gender.

Procedure One-third of the potential subjects was exposed to an information campaign two weeks before the experiment whilst another one-third was exposed one week before. The research departments were randomly selected and all their staff was exposed to the intervention.

The intervention consisted of two parts: *i*) a leaflet informing staff about what constitutes a scam and describing how scammers operate, how to detect them and what to do and *ii*) a reminder in the form of a semi-transparent card holder with the university logo on one side and the text "Beware of scams. Verify all requests. Report all incidents." on the other side.

The leaflets were designed using story telling to ensure that non experts could understand it as well [20]. The leaflet represents the information medium and the card holder represents a cue to remember the message. Departmental secretaries were responsible for distributing the materials and they were unaware that this was part of an experiment. The leaflet was distributed via email, whereas the card holder was distributed in person. All subjects were approached via telephone between 0930h and 1700h, on an normal Monday during term time.

The researchers were randomly assigned to a target, however if the researcher recognised a target, this target person was randomly assigned to another researcher. Each researcher approached the subject using the following script:

> Hi this is `[name]`. We discovered that the PC you are using is distributing spam emails. This is caused by a malicious program that is running in the background. Did you notice that your PC was a bit slower lately? There is nothing to be ashamed of, there are other people who have the same problem. I already helped 3 people to fix this earlier this morning. Luckily this is easy and quick to solve. Do you have 2 or 3 minutes time, so that we can remove it together right away? Please click the link that appears in the chat window. URL: `http://removespam.utwente.info`. To proceed to the download, please enter the validation code; this is your complete employee number. The complete number can be found on the back of your employee card. Please save the file to your Desktop and execute it. After the program is finished, could you read out the completion code?

All targets were subjected to the same script and request. After the target indicated that the downloaded file was installed, the debriefing procedure was started. During the debriefing procedure the target was told that this was an experiment, and asked some demographic information, employment length, some computer characteristics and their reasons for or against downloading and installing the software. Finally, the importance of not sharing any information about the experiment with colleagues was explained; all subjects acknowledged this and agreed not to disclose any information. This was checked during the debriefing and none of the subjects stated having had prior knowledge of the experiment.

2.3. Variables and Analysis

The variables used in the analysis were: compliance, intervention, age, offender and sex. The dependent variable compliance measured whether the subject complied with the request of the offender to download and install the software package. The dichotomous variable was dummy coded as 0 = did not comply, 1 = did comply. The independent variable intervention measured whether the subject was exposed to the intervention (0 = not exposed to the intervention, 1 = exposed to the intervention 1 week before, 2 = exposed to the intervention 2 weeks before). The independent continuous variable age measured the age in whole years at the moment of the attack. The independent categorical variable offender measured who performed the attack (1 = researcher1, ..., 4 = researcher4). The independent dichotomous variable sex was measured for both the subjects and the researchers and was dummy coded (0 = female, 1 = male). The hypothesis was tested using cross-tabulation and χ^2.

3. Results

A total of 48 subjects were approached. No 'target sex' effect on compliance ($\chi^2 = .470$, $df = 1$, $p = .493$), 'offender sex' effect on compliance ($\chi^2 = .176$, $df = 1$, $p = .675$), 'offender' effect on compliance ($\chi^2 = 4.253$, $df = 3$, $p = .354$) and 'target age' effect on compliance ($OR = .997$, $p = .905$) were found and therefore these issues are not further mentioned.

The debriefing procedure was used to verify that the subjects were coded correctly (i.e. had received an intervention or had not received an intervention). The subjects who recalled receiving the intervention material were coded as intervention group whilst those who could not recall having received any intervention material were coded as control group.

3.1. H1: *"Time influences the relation between compliance and intervention."*

The compliance for the control group was 46.15 % compared to 9.09 % for those exposed to the information campaign 1 week prior to the measurement. The compliance of those exposed to the campaign 1 week prior was 9.09 % compared to 54.55 % for those exposed to the campaign 2 weeks prior to the measurement. The compliance for the control group was 46.15 % compared to 54.55 % for those exposed to the campaign 2 weeks prior to the measurement. A difference was found between the control and the 1 week group, furthermore a difference was found between the 1 week and 2 week group. However no difference was found between the control and the 2 week group. Hypothesis 1 is therefore accepted. Refer to Table 1 for descriptive statistics.

Table 1. Number of observations and percentages per intervention condition over time

		Intervention			
		No	1 week	2 weeks	Total
Complied	No	14 (53.85%)	10 (90.91%)	5 (45.45%)	29 (60.42%)
	Yes	12 (46.15%)	1 (9.09%)	6 (54.55%)	19 (39.58%)
Total		26 (100%)	11 (100%)	11 (100%)	48 (100%)

Group control = 1 week ($\chi^2 = 4.659$, $df = 1$, $p = .031$);
Group control = 2 week ($\chi^2 = 0.218$, $df = 1$, $p = .641$;
Group 1 week = 2 week ($\chi^2 = 5.238$, $df = 1$, $p = .022$);

4. Discussion

This study investigated whether an information campaign influences the compliance with a telephone request to download and install software available on the internet, over time.

An information campaign consisting of *i*) informing employees about the dangers of telephone scams and *ii*) distributing a card holder with a reminder text was effective in the short term to reduce the vulnerabilty of employees to follow a stranger's request to perform actions on their PC.

In total 9.09 % of the employees exposed to an information campaign (1 week prior to the attack) compared to 46.15 % in the control group complied with the request to download and execute software available on the internet. Those not exposed to the campaign (1 week prior) have 8.13 higher odds of complying with the offenders request. These findings are in line with the results Bullée et al. [14] in which university personnel was approached by strangers and asked to hand over their keys.

The employees exposed 2 weeks prior showed no difference to the control group. The CPR studies of Madden [19] and Broomfield [17] both showed a significant decay since the training, however they measured their decay over a period of 10 weeks time.

One explanation for the difference could be the modality of information that is transferred, (i.e. visual leaflet). The effect of different modalities on memory has been shown in an experiment where the subjects had to remember and recall a list of words or auditory representations. The results showed that the auditory representations had both a significant better *i*) recall and *ii*) recall order of the presented stimuli [21]. Glenberg showed that this auditory modality effect is also present in long term memory, this means that auditory stimuli were better remembered in the long term compared to their visual counterparts [22].

An second alternative explanation could be in the process of creating a memory. There are 3 processes that constitute memory processing: *i*) Encoding, *ii*) Consolidation and *iii*) Retrieval. Encoding is the process of forming mental representations of the materials one wants to put in memory. The process of encoding is enhanced by elaboration (interpretation of the materials and connecting them to other materials) and trying to repeat it to oneself [23]. Attention is important as well; when attention is divided as encoding will be weaker and later attempts to remember are likely to fail [24, p. 202]. The consolidation process modifies the mental representation in such a way that it becomes stable in memory. Consolidation of the declarative memory (explicit memory containing experiences and information) can be affected by sleep [25], caffeine intake [26] and age [27]. Finally, retrieval is important to access the knowledge in the brain via cues. The context (e.g. physical surrounding) in which a memory is created is important for retrieval. The theory of optimistic bias states that people believe that positive events are more likely to occur to them than to other people [28] and that negative events are more likely to occur to other people than to themselves. In the context of the information campaign, a possibility is that the subjects could not identify themselves with the material because they thought that it was not applicable to them, since there are others who were more vulnerable. In both cases the subjects will not have the appropriate attention to properly encode the material to make it last in the memory.

A third explanation could be that the information campaign is too abstract and therefore an ambiguous memory cue is created for the material. Once the subject is 'attacked' there is no proper recollection of the cue to memory and he/she fails to recollect the materials in the information campaign. This could explain the difference between the 1 and 2 week groups as the cues are simply forgotten over time. It would be too simple to say that the subjects forgot about the information campaign since there were questions to control for this, since the subjects stated they recalled having received and read the materials.

Limitations This study has 2 limitations: *i*) The current study has a limited number of observations, therefore only a limited number of variables could be tested. *ii*) All subjects were from the same organisation, replication in other organisations would be needed.

Acknowledgements

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 318003 (TREsPASS). This publication reflects only the author's views and the Union is not liable for any use that may be made of the information contained herein. In addition, we would also like to thank Christian Orriëns, Koen de Jong, Evelien Kroon, and Hilde Jansen for their valuable contribution to this research.

References

[1] C. Arthur, "Virus phone scam being run from call centres in india," 18 July 2010 2010. [Online]. Available: http://www.theguardian.com/world/2010/jul/18/phone-scam-india-call-centres

[2] M. Rouse. (2006) Definition Social Engineering. [Online]. Available: http://www.searchsecurity.techtarget.com/definition/social-engineering

[3] H.-S. Rhee, C. Kim, and Y. U. Ryu, "Self-efficacy in information security: Its influence on end users' information security practice behavior," *Computers & Security*, vol. 28, no. 8, pp. 816 – 826, 2009.

[4] D. Harley, M. Grooten, S. Burn, and C. Johnston, "My pc has 32,539 errors: how telephone support scams really work," *22nd Virus Bulletin International Conference (VB2012)*, 2012.

[5] R. E. Petty and J. T. Cacioppo, "The elaboration likelihood model of persuasion," in *Communication and Persuasion*. Springer, 1986, pp. 1–24.

[6] ——, *Attitudes and Persuasion–classic and Contemporary Approaches*. W.C. Brown Company Publishers, 1981.

[7] S. Stubbings, K. Robb, J. Waller, A. Ramirez, J. Austoker, U. Macleod, S. Hiom, and J. Wardle, "Development of a measurement tool to assess public awareness of cancer," *Br J Cancer*, vol. 101, no. S2, pp. S13–S17, 2000.

[8] J. Barlow, "Knowledge in patients with rheumatoid arthritis: a longer term follow- up of a randomized controlled study of patient education leaflets," *Rheumatology*, vol. 37, no. 4, pp. 373–376, 1998.

[9] F. Ghaderi, A. Adl, and Z. Ranjbar, "Effect of a leaflet given to parents on knowledge of tooth avulsion." *European journal of paediatric dentistry : official journal of European Academy of Paediatric Dentistry*, vol. 14, no. 1, pp. 13–6, 2013.

[10] S. M. Shim, S. H. Seo, Y. Lee, G. I. Moon, M. S. Kim, and J. H. Park, "Consumers'knowledge and safety perceptions of food additives: Evaluation on the effectiveness of transmitting information on preservatives," *Food Control*, vol. 22, no. 7, pp. 1054–1060, 2011.

[11] P. Kumaraguru, S. Sheng, A. Acquisti, L. F. Cranor, and J. Hong, "Teaching Johnny not to fall for phish," *ACM Transactions on Internet Technology*, vol. 10, no. 2, pp. 1–31, 2010.

[12] D. B. Cornish and R. V. Clarke, "Opportunities, precipitators and criminal decisions: A reply to wortley's critique of situational crime prevention," *Crime prevention studies*, vol. 16, pp. 41–96, 2003.

[13] I. Flight, C. Wilson, and J. McGillivray, "Turning intention into behaviour: The effect of providing cues to action on participation rates for colorectal cancer screening," *Colorectal Cancer-From Prevention to Patient Care. Shanghai: InTech*, pp. 67–86, 2012.

[14] J. H. Bullée, L. Montoya, W. Pieters, M. Junger, and P. H. Hartel, "The persuasion and security awareness experiment: reducing the success of social engineering attacks," *Journal of Experimental Criminology*, vol. 11, no. 1, pp. 97–115, 2015.

[15] H. Ebbinghaus, *Memory: A Contribution to Experimental Psychology*. Teachers College, Columbia University, 1913, no. 3.

[16] S. Casner, D. Heraldez, and K. Jones, "Retention of aeronautical knowledge," *International Journal of Applied Aviation Studies*, vol. 6, no. 1, pp. 71–98, 2006.

[17] R. Broomfield, "A quasi-experimental research to investigate the retention of basic cardiopulmonary resuscitation skills and knowledge by qualified nurses following a course in professional development," *Journal of Advanced Nursing*, vol. 23, no. 5, pp. 1016–1023, 1996.

[18] S. M. L. Hendrickson, T. E. Goldsmith, and P. J. Johnson, "Retention of airline pilots' knowledge and skill," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 50, no. 17, pp. 1973–1976, 2006.

[19] C. Madden, "Undergraduate nursing students acquisition and retention of CPR knowledge and skills," *Nurse Education Today*, vol. 26, no. 3, pp. 218 – 227, 2006.

[20] E. Rader, R. Wash, and B. Brooks, "Stories as informal lessons about security," in *Proceedings of the Eighth Symposium on Usable Privacy and Security*, ser. SOUPS '12. New York, NY, USA: ACM, 2012, pp. 6:1–6:17.

[21] A. Drewnowski and B. B. Murdock, "The role of auditory features in memory span for words." *Journal of Experimental Psychology: Human Learning and Memory*, vol. 6, no. 3, pp. 319 – 332, 1980.

[22] A. M. Glenberg, "A retrieval account of the long-term modality effect." *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 10, no. 1, pp. 16 – 31, 1984.

[23] F. I. Craik and R. S. Lockhart, "Levels of processing: A framework for memory research," *Journal of verbal learning and verbal behavior*, vol. 11, no. 6, pp. 671–684, 1972.

[24] E. Smith and S. Kosslyn, *Cognitive Psychology: Mind and Brain*, ser. Pearson Education. Pearson Prentice Hall, 2008.

[25] A. Ashworth, C. M. Hill, A. Karmiloff-Smith, and D. Dimitriou, "Sleep enhances memory consolidation in children," *Journal of Sleep Research*, vol. 23, no. 3, pp. 304–310, 2014.

[26] S. E. Favila and B. A. Kuhl, "Stimulating memory consolidation," *Nature Neuroscience*, vol. 17, no. 2, pp. 151–152, 02 2014.

[27] L. Cahill, B. Prins, M. Weber, and J. L. McGaugh, "β-adrenergic activation and memory for emotional events," *Nature*, vol. 371, no. 6499, pp. 702–704, 10 1994.

[28] N. D. Weinstein, "Unrealistic optimism about future life events." *Journal of personality and social psychology*, vol. 39, no. 5, p. 806, 1980.

Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016
A. Mathur and A. Roychoudhury (Eds.)

doi:10.3233/978-1-61499-617-0-115

Privacy-Preserving and Verifiable Data Aggregation

Hieu N TRAN [a], Robert H DENG [a] and HweeHwa PANG [a]
[a] *School of Information Systems, Singapore Management University*

Abstract. There are several recent research studies on privacy-preserving aggregation of time series data, where an aggregator computes an aggregation of multiple users' data without learning each individual's private input value. However, none of the existing schemes allows the aggregation result to be verified for integrity. In this paper, we present a new data aggregation scheme that protects user privacy as well as integrity of the aggregation. Towards this end, we first propose an aggregate signature scheme in a multi-user setting without using bilinear maps. We then extend the aggregate signature scheme into a solution for privacy-preserving and verifiable data aggregation. The solution allows multiple users to periodically send encrypted data to an untrusted aggregator such that the latter is able to compute the sum of the input data values and verify its integrity, without learning any other information. A formal security analysis shows that the solution is semantically secure and unforgeable.

Keywords. data aggregation, data privacy, verifiable computation, aggregate signature

1. Introduction

Data aggregation, in which an aggregator computes aggregated results over multiple sources, has been widely used in real-world applications such as network coding, smart grid, mobile sensing and cloud services. In order to prevent leakage of sensitive information, the aggregator should be able to perform aggregation operations such as `sum`, `average`, `variance` on users' data without learning their individual values. Symmetric key homomorphic encryption schemes, which are normally based on one-time pad encryption, could be applied to protect the confidentiality of user data [1]. However, such schemes assume that the aggregator is fully trusted since each user shares a secret key with the aggregator. Therefore, these schemes protect data confidentiality against outsiders, but they do not protect users' data privacy against the aggregator.

Recently, several research studies have addressed the privacy-preserving data aggregation problem under the assumption of an untrusted aggregator [2,3]. Shi et al. proposed a construction that allows the untrusted aggregator to compute the sum of time series data [2]. However, the aggregator must use brute-force search or Pollard's lambda method [4] to decrypt the sum, which is not efficient for applications requiring a large plaintext space. Li et al. [3] proposed an efficient aggregation scheme based on a novel key management technique, where only computation of key hashes is needed for encrypting data by users and for decrypting the sum of data values by the aggregator.

In many critical control and monitoring systems, ensuring the integrity of aggregation results is extremely important. In such systems, any invalid aggregation result due to tampering by malicious attackers must be detected and rejected by the aggregator. There are two generic mechanisms for verifying the integrity of aggregation results aggregate message authentication codes (MACs) [5,6] and aggregate signatures [7,8,9,10,11,12]. The notion of aggregate MAC was first proposed by Katz et al. [5]. Agrawal et al. [13] proposed a homomorphic MAC scheme, a generalization of aggregate MAC, that allows a network node to compute a MAC of combined code words in networking coding. Aggregate MAC and homomorphic MAC require shared secret keys between the data generators and aggregators. In contrast, aggregate signature schemes enable anyone with the correct public keys to verify the integrity of aggregation results. However, most of the existing aggregate signature schemes [7,8,9,10,11,12] are constructed using bilinear maps and are not suitable for resource-constrained system settings such as sensor networks.

Our Contribution. Our goal is to propose an efficient construction for privacy-preserving and verifiable data aggregation. The main contributions of this paper are as follows:

- We first introduce an aggregate signature scheme for public verification of the integrity of aggregated data in a multi-user setting. In this scheme, each user sends his data and a signature to an aggregator, where the signature is generated on the data using the user's secret key. The aggregator combines all the users' data and signatures into an aggregated result and an aggregated signature, respectively, and checks the validity of the result using public system parameters.
- Extending the aggregate signature scheme, we further propose a privacy-preserving and verifiable data aggregation (PVDA) scheme. PVDA allows multiple users to periodically send encrypted data to an untrusted aggregator such that the latter is able to compute the sum of the data values and verify its integrity but not learn any other information. Our formal security analysis shows that the solution is semantically secure and unforgeable.

2. Related Work

We survey prior work on privacy-preserving and verifiable data aggregation, aggregate MAC and aggregate signature.

Privacy-Preserving Data Aggregation Many protocols have been proposed for privacy-preserving data aggregation in wireless sensor network (WSN) or mobile sensing [1,3,14,15,16,17], without considering the integrity of aggregation result. Shi et al. [2] proposed a construction that allows a group of users to send encrypted data to an untrusted aggregator. The construction is not efficient for applications requiring a large plaintext space because the aggregator must use brute-force search to decrypt a sum. Li et al. [3] extended the construction in [2] using a new key management technique to ensure that the aggregator could only obtain the sum of the users' private data but not the private data of each user. Our PVDA construction solution is motivated by [2,3], but provide verifiable assurance in addition to data privacy.

Aggregate MAC and Aggregate Signature Aggregate MACs [5,13] allow verification of the integrity of aggregation data in private setting. Recently, Catalano and Fiore [18] proposed homomorphic MAC schemes that support a set of functionalities (i.e., polynomially-bounded arithmetic circuits); however, in their construction, the verifier possesses the user's secret key and hence is able to access users' private data. Aggregate signatures [8,9,10,11,12] also allow verification of the integrity of aggregation data but in public setting. Most of the existing aggregate signature (or more generally homomorphic signature) schemes are constructed on bilinear maps and unsuitable for resource-constrained applications.

Verifiable and Privacy-Preserving Data Aggregation Lin et al. [19] proposed a multi-dimensional privacy preserving data aggregation scheme for WSN. This scheme adapts the super-increasing sequences and permutation techniques from [1] for data aggregation. However, the scheme assumes that the aggregator is fully trusted. Papadopoulos et al. [20] presented a scheme that utilizes a combination of homomorphic encryption and secret sharing. Although their scheme works efficiently on time series data, the data privacy of each node may be breached if the querier colludes with the aggregator, since the querier has the secret key of every node.

3. A New Aggregate Signature Scheme

3.1. Overview

Our system model consists of three types of entities: a set of n users $\{u_1, ..., u_n\}$, an aggregator, and a key generation center. The key generation center creates a public parameter $\mathscr{S}$, as well as a key pair sk_i and pk_i for every user u_i. We assume that key pairs $(\mathsf{sk}_i, \mathsf{pk}_i)$ for all $i = 1, ..., n$ are distributed to the corresponding users via a secure channel at system initialization.

In each time period, each user u_i generates a private datum $m_i \in \mathbb{Z}_p$ where p is a large prime number. The aggregator is a powerful base station that performs data aggregation operations such as $\mathtt{sum}(M) = m_1 + ... + m_n$, where $M = (m_1, ..., m_n)$ is a vector of all the users' private data.

We assume that the aggregator is honest but curious. He is trusted to perform aggregation operations, but may attempt to discover individual users' private data. We also assume that users follow the correct aggregation process, do not trust each other and are curious as well. Some curious users may collude with the aggregator by revealing their secret keys in order to deduce additional information about the remaining users.

3.2. Definition

An aggregate signature scheme is a tuple of three probabilistic polynomial-time (PPT) algorithms (KeyGen, Sign, Verify) as follows:

- $\mathsf{KeyGen}(1^\lambda)$ algorithm is executed by the key generation center. It takes as input a security parameter λ and outputs a pair of public key pk_i and secret key sk_i for each user u_i along with a set of random values $\mathscr{S}$ as public parameter.

- $\mathsf{Sign}(\mathsf{sk}_i, m_i, t)$ algorithm is executed by each user u_i. It takes as input a secret key sk_i, datum m_i and time period t, and outputs a signature $\tau_i \in \mathbb{Z}_p$. User u_i then sends (m_i, τ_i) to the aggregator.
- $\mathsf{Verify}(\mathsf{pk}, \mathscr{S}, M, T, t)$ algorithm is executed by the aggregator. It takes as input a vector of public keys $\mathsf{pk} = (\mathsf{pk}_1, ..., \mathsf{pk}_n)$, the set $\mathscr{S}$, a vector of data $M = (m_1, ..., m_n)$, a vector of signatures $T = (\tau_1, ..., \tau_n)$ and a time period t. It outputs '1' *(accept)* or '0' *(reject)*.

Security. We allow the adversary to request a vector of signatures T for an arbitrary vector of data M along with period time t of his choice. Note that set $\mathscr{S}$ is a public parameter, hence anyone can compute and verify the integrity of an aggregation result. The adversary breaks the aggregate signature scheme if he is able to output a valid triple (M, T, t) where either period time t is new, or he has not previously requested signatures on the vector of data M. We define the security of our scheme in terms of the following game between a challenger and an adversary $\mathscr{A}$.

Game 1 *Let $\mathscr{H} = (KeyGen, Sign, Verify)$ denote an aggregate signature scheme.*

- *Setup: The challenger generates a set of random values $\mathscr{S}$ and n key pairs $\{(sk_i, pk_i)\}_{i=1}^n$, then sends $\mathscr{S}$ and $pk = (pk_1, ..., pk_n)$ to the adversary.*
- *Queries: The adversary submits signature queries, each of the form (M_j, t_j) where $M_j = (m_{1,j}, ..., m_{n,j}) \in \mathbb{Z}_p^n$ and $t_j \in \{0,1\}^*$. We require the time periods t_j submitted by the adversary to be distinct. The challenger performs the following:*

```
for all $i = 1, ..., n$ do
    $\tau_{i,j} \leftarrow Sign(sk_i, m_{i,j}, t_j)$
send $T_j = (\tau_{1,j}, ..., \tau_{n,j})$ to $\mathscr{A}$
```

- *Output: The adversary outputs a vector of data M^*, a vector of signatures T^* and a time period t^*. The adversary wins the security game if $1 \leftarrow Verify(pk, \mathscr{S}, M^*, T^*, t^*)$, and one of the following conditions hold:*
  - *Type 1 forgery: $t^* \neq t_j$ for all j.*
  - *Type 2 forgery: $t^* = t_j$ for some j, and $M^* \neq M_j$.*

The advantage $\mathsf{uf\text{-}adv}[\mathscr{A}, \mathscr{H}]$ of adversary $\mathscr{A}$ with the aggregate signature scheme $\mathscr{H}$ is defined as the probability that $\mathscr{A}$ wins Game 1.

Definition 1 *The aggregate signature scheme $\mathscr{H}$ is existentially unforgeable under an adaptive chosen-message attack if, for all PPT adversaries $\mathscr{A}$, the advantage uf-adv$[\mathscr{A}, \mathscr{H}]$ is negligible.*

3.3. Construction

Let $\mathbb{G}$ be a cyclic group of prime order p on which the discrete logarithm problem is hard. Let $M = (m_1, ..., m_n) \in \mathbb{Z}_p^n$ denote a vector of n users' data values, where $m_i < p$ for $i = 1, ..., n$ and $\sum_{i=1}^n m_i \leq p$. Let $f : \mathbb{Z}_p \times \{0,1\}^* \to \mathbb{Z}_p$ be a pseudo-random function (PRF). The three algorithms (KeyGen, Sign, Verify) are given as follows.

- $\mathsf{KeyGen}(1^\lambda)$: The key generation center chooses a random generator $g \in \mathbb{G}$, generates a set of $n\alpha$ random values $\mathscr{S} = (s_1, ..., s_{n\alpha}) \in \mathbb{Z}_p^{n\alpha}$ and divides them into

n disjoint subsets $\mathscr{S}_i$ such that each subset $\mathscr{S}_i$ has α values, $\mathscr{S} = \bigcup_{i=1}^{n} \mathscr{S}_i$ and $\forall i \neq j, \mathscr{S}_i \bigcap \mathscr{S}_j = \emptyset$. The key generation center assigns $\mathscr{S}_i$ to user u_i and $\mathscr{S}$ to the aggregator. Each user u_i then generates a random integer a_i and computes $h_i = g^{a_i}$. Each user u_i has a secret key $\mathsf{sk}_i = \langle \mathscr{S}_i, a_i \rangle$ and a public key $\mathsf{pk}_i = \langle g, h_i \rangle$.

- $\mathsf{Sign}(\mathsf{sk}_i, m_i, t)$: Given secret key sk_i and datum m_i for period time t, user u_i computes a signature $\tau_i = a_i \times m_i + \mathsf{F}_{\mathscr{S}_i}(t) \bmod p$, where $\mathsf{F}_{\mathscr{S}_i}(t) = \sum_{s \in \mathscr{S}_i} f_s(t) \bmod p$, then sends (m_i, τ_i) to the aggregator.
- $\mathsf{Verify}(\mathsf{pk}, \mathscr{S}, M, T, t)$: For each time period t, the aggregator computes an aggregate signature $\mathscr{T} = \sum_{i=1}^{n} \tau_i = \sum_{i=1}^{n} (a_i \times m_i + \mathsf{F}_{\mathscr{S}_i}(t)) \bmod p$. The aggregator then verifies the integrity of the aggregate signature $\mathscr{T}$ using public keys $\mathsf{pk} = (\mathsf{pk}_1, ..., \mathsf{pk}_n)$ and $\mathscr{S} = \{s_1, ..., s_{n\alpha}\}$ as follows:

  - $Y = \prod_{i=1}^{n} h_i^{m_i} \in \mathbb{G}$.
  - If $g^{\mathscr{T}} = g^{\sum_{s \in \mathscr{S}} f_s(t)} \times Y$, output '1' *(accept)*; otherwise output '0' *(reject)*.

Security. Adversary $\mathscr{A}$ is able to succeed in a forgery attack and win the security Game 1 if he knows the secret key of any of the n users. However, his chance of guessing any user's secret key is negligible, as Lemma 1 shows. We prove security assuming F is a secure PRF, and the discrete logarithm assumption holds in $\mathbb{G}$. We refer to [4] for a definition of the PRF and discrete logarithm security games. The proofs are given in the full version of this paper.

Lemma 1 *In our aggregate signature scheme, the probability of successfully discovering any user's private key through brute-force guessing is negligible.*

Theorem 1 *The aggregate signature scheme is existentially unforgeable under an adaptive chosen-message attack, assuming F is a secure PRF and the discrete logarithm problem is hard.*

4. Privacy-Preserving and Verifiable Data Aggregation

The aggregate signature scheme introduced in the previous section does not maintain *confidentiality* of users' data. We now propose a privacy-preserving and verifiable data aggregation (PVDA) scheme that extends the aggregate signature scheme.

4.1. Definition

PVDA comprises four algorithms (KeyGen, Enc, Sign, AggDec) as follows:

- $\mathsf{KeyGen}(1^\lambda)$ algorithm remains the same as in Section 3.2.
- $\mathsf{Enc}(\mathsf{sk}_i, m_i, t)$ algorithm is executed by each user u_i. It takes as input secret key sk_i, datum m_i and time period t, and outputs a ciphertext $c_i \in \mathbb{Z}_p$.
- $\mathsf{Sign}(\mathsf{sk}_i, c_i, t)$ algorithm is as in Section 3, except that m_i is replaced by ciphertext c_i. Each user sends (c_i, τ_i) to the aggregator where $\tau_i \leftarrow \mathsf{Sign}(\mathsf{sk}_i, c_i, t)$.
- $\mathsf{AggDec}(\mathsf{pk}, \mathscr{S}, C, T, t)$ algorithm is executed by the aggregator. It takes as input public keys $\mathsf{PK} = (\mathsf{pk}_1, ..., \mathsf{pk}_n)$, the set $\mathscr{S}$, a vector of ciphertexts $C = (c_1, ..., c_n)$, a vector of signatures $T = (\tau_1, ..., \tau_n)$ and time period t. It outputs an aggregation result $\mathscr{M} = m_1 + ... + m_n$ if the Verify algorithm as in Section 4.2 outputs '1'; otherwise it outputs error $\perp$.

Security. The security notion for PVDA includes *semantic security* and *unforgeability*. Let $\mathscr{P} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Sign}, \mathsf{AggDec})$ denote the PVDA scheme.

Game 2 *Semantic security of the PVDA scheme is defined by the following game between a challenger and an adversary $\mathscr{A}$:*

- *Setup: The challenger generates a set $\mathscr{S}$ of $n\alpha$ random values and n key pairs (sk_i, pk_i) for $i = 1, ..., n$. The challenger gives the public parameters $\langle \mathscr{S}, pk = (pk_1, ..., pk_n) \rangle$ to the adversary. The former also initializes a list $L = \emptyset$ for tracking the queries from $\mathscr{A}$.*
- *Queries: The adversary adaptively queries the challenger for encryption. Each query states $m \in \mathbb{Z}_p$, index i of user u_i and time period t. The query is rejected if the tuple (i,t) exists in L; otherwise, the challenger computes $c \leftarrow \mathsf{Enc}(sk_i, m, t)$ and $\tau \leftarrow \mathsf{Sign}(sk_i, c, t)$, sends (c, τ) to $\mathscr{A}$ and inserts (i,t) into L.*
- *Challenge: The adversary $\mathscr{A}$ submits m_0, $m_1 \in \mathbb{Z}_p$ along with index i^* and time period t^* to the challenger; L must not already contain the tuple (i^*, t^*). The challenger computes $c^* \leftarrow \mathsf{Enc}(sk_{i^*}, m_b, t^*)$, $\tau^* \leftarrow \mathsf{Sign}(sk_{i^*}, c^*, t^*)$ under a random bit $b \in \{0, 1\}$, and sends (c^*, τ^*) to adversary $\mathscr{A}$.*
- *Output: Adversary $\mathscr{A}$ outputs b', representing his guess for b, and wins the game if $b' = b$.*

The advantage $\mathsf{ss\text{-}adv}[\mathscr{A}, \mathscr{P}]$ of adversary $\mathscr{A}$ with respect to $\mathscr{P}$ is defined as $|\Pr[b = b'] - \frac{1}{2}|$, where the probability is taken over the random bit used by the challenger and the adversary $\mathscr{A}$.

Definition 2 *The PVDA scheme $\mathscr{P}$ is semantically secure if, for all PPT adversary $\mathscr{A}$, the advantage $\mathsf{ss\text{-}adv}[\mathscr{A}, \mathscr{P}]$ is negligible.*

Game 3 *Unforgeability of the PVDA scheme is defined by the following game between a challenger and an adversary $\mathscr{A}$:*

- *Setup: The challenger generates a set $\mathscr{S}$ of $n\alpha$ random values and n key pairs (SK_i, PK_i) for $i = 1, ..., n$. The challenger gives public parameters $\langle \mathscr{S}, PK_1, ..., PK_n \rangle$ to the adversary.*
- *Queries: The adversary adaptively submits (M_j, t_j) to the challenger as in Game 1. The challenger responds with $(c_{1,j}, \tau_{1,j}), ..., (c_{n,j}, \tau_{n,j})$ for each query, where $c_{i,j}$ is the ciphertext of $m_{i,j}$, and $\tau_{i,j}$ is the signature of $c_{i,j}$ for $i = 1, ..., n$. We require all the time periods t_j's in the queries to be distinct.*
- *Output: The adversary outputs the set of n pairs (c_1^*, τ_1^*) , ..., (c_n^*, τ_n^*) for $M^* = (m_1^*, ..., m_n^*)$ and time period t^* such that either t^* is a new time period or M^* has not been queried before. The adversary wins the game if the AggDec algorithm outputs $\mathscr{M}^* = \sum_{i=1}^{n} m_i^*$.*

The advantage $\mathsf{uf\text{-}adv}[\mathscr{A}, \mathscr{P}]$ of adversary $\mathscr{A}$ with respect to $\mathscr{P}$ is defined as the probability that $\mathscr{A}$ wins Game 3.

Definition 3 *Our PVDA scheme $\mathscr{P}$ is unforgeable if, for all PPT adversary $\mathscr{A}$, the advantage $\mathsf{uf\text{-}adv}[\mathscr{A}, \mathscr{P}]$ is negligible.*

4.2. Construction

In order to protect the privacy of users' data while providing verifiability of aggregation results, we use the technique of Li et al. [3] in encryption. However, PVDA is more practical because Li et al.'s scheme does not provide verifiability. The four algorithms (KeyGen, Enc, Sign, AggDec) in PVDA are given below.

- $\mathsf{KeyGen}(1^\lambda)$: The same as in the aggregate signature scheme.
- $\mathsf{Enc}(\mathsf{sk}_i, m_i, t)$: User u_i generates encryption keys $k_i = \mathsf{F}_{\mathscr{S}_i}(t\|1)$ and $k'_i = \mathsf{F}_{\mathscr{S}_i}(t\|2)$, where $\|$ denotes concatenation. Next, the user encrypts his datum m_i by computing a ciphertext $c_i = m_i + k_i \bmod p$.
- $\mathsf{Sign}(\mathsf{sk}_i, c_i, t)$: User u_i computes a signature τ_i on the output c_i of the Enc algorithm using his private value a_i and encryption key k'_i by computing $\tau_i = a_i \times c_i + k'_i \bmod p$. User u_i then sends (c_i, τ_i) to the aggregator.
- $\mathsf{AggDec}(\mathsf{pk}, \mathscr{S}, C, T, t)$: In each time period t, the aggregator generates decryption keys $k_0 = \mathsf{F}_{\mathscr{S}}(t\|1)$ and $k'_0 = \mathsf{F}_{\mathscr{S}}(t\|2)$. Upon receiving $C = (c_1, ..., c_n)$ and $T = (\tau_1, ..., \tau_n)$ from all the n users, the aggregator computes:

$$\mathscr{T} = \sum_{i=1}^{n} \tau_i \bmod p, \qquad \mathscr{X} = \sum_{i=1}^{n} c_i \bmod p, \qquad \mathscr{Y} = \prod_{i=1}^{n} h_i^{c_i}$$

  The aggregation signature $\mathscr{T}$ is valid if and only if $g^{\mathscr{T}} = g^{k'_0} \times \mathscr{Y}$. The aggregator outputs aggregation result $\mathscr{M} = \sum_{i=1}^{n} m_i = \mathscr{X} - k_0 \bmod p$ if signature $\mathscr{T}$ is valid; otherwise, it returns error $\perp$.

Security. We now formally state and prove the semantic security and unforgeability assurances of PVDA. We note that Lemma 1 applies to PVDA. The proofs of the security theorems are given in the full version of this paper.

Theorem 2 *The PVDA scheme $\mathscr{P}$ is semantically secure if, for all PPT adversaries $\mathscr{A}$, the advantage $\mathsf{ss\text{-}adv}[\mathscr{A}, \mathscr{P}]$ is negligible assuming F is a secure PRF.*

Theorem 3 *The PVDA scheme $\mathscr{P}$ is unforgeable if, for all PPT adversaries $\mathscr{A}$, the advantage $\mathsf{uf\text{-}adv}[\mathscr{A}, \mathscr{P}]$ of adversary $\mathscr{A}$ is negligible assuming F is a secure PRF and the discrete logarithm problem is hard.*

5. Conclusions

In this paper, we presented an aggregate signature scheme that is suitable for data aggregation in a multi-user setting. In the scheme, each user computes a signature on his data using a secret key. An aggregator computes the sum of the users' data and combines their signatures into a single aggregate signature. Using public system parameters and the aggregate signature, the aggregator is able to verify the integrity of the sum. To protect the users' data privacy, we further extended the aggregate signature scheme into a privacy-preserving and verifiable data aggregation (PVDA) scheme for time series data in the same multi-user setting. We formally proved that the PVDA scheme is semantically secure and existentially unforgeable. Compared to existing alternatives in the literature, our PVDA scheme offers two key advantages in assuming an untrusted aggregator, and being computationally efficient as no bilinear maps are used.

References

[1] C. Castelluccia, E. Mykletun, and G. Tsudik, "Efficient aggregation of encrypted data in wireless sensor networks," in *2nd Annual International Conference on Mobile and Ubiquitous Systems (MobiQuitous)*, pp. 109–117, 2005.

[2] E. Shi, T. H. Chan, E. G. Rieffel, R. Chow, and D. Song, "Privacy-preserving aggregation of time-series data," in *Proceedings of the Network and Distributed System Security Symposium, NDSS*, 2011.

[3] Q. Li and G. Cao, "Efficient and privacy-preserving data aggregation in mobile sensing," in *20th IEEE International Conference on Network Protocols, ICNP*, pp. 1–10, 2012.

[4] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*. Chapman and Hall/CRC Press, 2007.

[5] J. Katz and A. Y. Lindell, "Aggregate message authentication codes," in *Topics in Cryptology - CT-RSAs*, pp. 155–169, 2008.

[6] A. C. Chan and C. Castelluccia, "On the (im)possibility of aggregate message authentication codes," in *2008 IEEE International Symposium on Information Theory, ISIT*, pp. 235–239, 2008.

[7] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps," in *Advances in Cryptology - EUROCRYPT*, pp. 416–432, 2003.

[8] C. Gentry and Z. Ramzan, "Identity-based aggregate signatures," in *Public Key Cryptography - PKC*, pp. 257–273, 2006.

[9] D. Boneh, D. M. Freeman, J. Katz, and B. Waters, "Signing a linear subspace: Signature schemes for network coding," in *Public Key Cryptography - PKC 2009, 12th International Conference on Practice and Theory in Public Key Cryptography, Irvine, CA, USA, March 18-20, 2009. Proceedings*, pp. 68–87, 2009.

[10] N. Attrapadung and B. Libert, "Homomorphic network coding signatures in the standard model," in *Public Key Cryptography - PKC 2011 - 14th International Conference on Practice and Theory in Public Key Cryptography, Taormina, Italy, March 6-9, 2011. Proceedings*, pp. 17–34, 2011.

[11] D. Catalano, D. Fiore, and B. Warinschi, "Efficient network coding signatures in the standard model," in *Public Key Cryptography - PKC 2012 - 15th International Conference on Practice and Theory in Public Key Cryptography, Darmstadt, Germany, May 21-23, 2012. Proceedings*, pp. 680–696, 2012.

[12] D. M. Freeman, "Improved security for linearly homomorphic signatures: A generic framework," in *Public Key Cryptography - PKC 2012 - 15th International Conference on Practice and Theory in Public Key Cryptography, Darmstadt, Germany, May 21-23, 2012. Proceedings*, pp. 697–714, 2012.

[13] S. Agrawal and D. Boneh, "Homomorphic macs: Mac-based integrity for network coding," in *Applied Cryptography and Network Security, 7th International Conference, ACNSs*, pp. 292–305, 2009.

[14] J. Shi, R. Zhang, Y. Liu, and Y. Zhang, "Prisense: Privacy-preserving data aggregation in people-centric urban sensing systems," in *INFOCOM 2010. 29th IEEE International Conference on Computer Communications*, pp. 758–766, 2010.

[15] T. Feng, C. Wang, W. Zhang, and L. Ruan, "Confidentiality protection for distributed sensor data aggregation," in *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pp. 56–60, 2008.

[16] W. He, X. Liu, H. Nguyen, K. Nahrstedt, and T. F. Abdelzaher, "PDA: privacy-preserving data aggregation in wireless sensor networks," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications*, pp. 2045–2053, 2007.

[17] T. Jung, X. Mao, X. Li, S. Tang, W. Gong, and L. Zhang, "Privacy-preserving data aggregation without secure channel: Multivariate polynomial evaluation," in *Proceedings of the IEEE INFOCOM*, pp. 2634–2642, 2013.

[18] D. Catalano and D. Fiore, "Practical homomorphic macs for arithmetic circuits," in *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, pp. 336–352, 2013.

[19] X. Lin, R. Lu, and X. S. Shen, "MDPA: multidimensional privacy-preserving aggregation scheme for wireless sensor networks," *Wireless Communications and Mobile Computing*, vol. 10, no. 6, pp. 843–856, 2010.

[20] S. Papadopoulos, A. Kiayias, and D. Papadias, "Secure and efficient in-network processing of exact SUM queries," in *Proceedings of the 27th International Conference on Data Engineering, ICDE*, pp. 517–528, 2011.

Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016
A. Mathur and A. Roychoudhury (Eds.)

doi:10.3233/978-1-61499-617-0-123

Simulation of Cyber-Physical Attacks on Water Distribution Systems with EPANET

Riccardo TAORMINA [a,1], Stefano GALELLI [a], Nils Ole TIPPENHAUER [b], Elad SALOMONS [c], and Avi OSTFELD [d]

[a] *ESD Pillar, Singapore University of Technology and Design*
[b] *ISTD Pillar, Singapore University of Technology and Design*
[c] *OptiWater*
[d] *Faculty of Civil and Environmental Engineering, Technion – Israel Institute of Technology*

Abstract. In this work, we discuss the use of EPANET to simulate the effects of malicious cyber-physical attacks on water distribution systems. EPANET—a standard numerical modeling environment developed by the US Environmental Protection Agency—models hydraulic and water-quality behavior of pressurized pipe networks. EPANET promises to be well suited to show the effects of direct attacks on hydraulic actuators, such as pumps, or the effects of attacks on sensors. Using the C-Town benchmark network, we show that EPANET has some limitations when modeling these attacks, and we report the workarounds needed to overcome these limitations. In particular, we describe attacks that change the control strategies of pumps, and attacks that alter the tank water level reported by sensors.

Keywords. Water Distribution Systems, Cyber-Physical System Security, EPANET

1. Introduction

Water distribution systems (WDS) are used in all urban developments to connect potable water sources (e.g., reservoirs or treatment plants) to the end users (e.g., housing and industrial estates). Typical components of WDS are pipes, pumps, valves, and storage tanks. Traditionally, these components were operated manually—or controlled locally—. In the past few decades advanced control systems were introduced by relying on SCADA systems, which allow automated collection of distributed sensor readings and centralized control of actuators. Whilst these changes promise to improve the operating performance of WDS, they are also expected to lead to novel security vulnerabilities [6]. In particular, attacks on automated controls have been recently discussed [2, 8, 9, 18]. The US Department of Homeland Security reported that 15 percent of the responses to cyber incidents were in the water sector [12]. Countermeasures against such attacks can be implemented

[1]Corresponding Author: Riccardo Taormina, Singapore University of Technology and Design, 8 Somapah Road, 487372 Singapore; E-mail: riccardo_taormina@sutd.edu.sg

through additional security measures on the sensor, network, and SCADA layers, but the fundamental problem remains: sensors readings rely on physical layer properties, which are susceptible to manipulations of the physical layer. Furthermore, these infrastructures are typically operated for extended periods of time, possibly decades. As such, there are higher chances that one or multiple components be attacked during their life cycle.

We consider cyber-physical attacks, which include both physical and cyber attacks that target the physical system (e.g. the Aurora attack [13]). The exact effect of such cyber-physical attacks effect depends on the specific physical process being altered. For WDS, both hydraulic (e.g. water spillage, tank overflow, pipe bursts, loss of pressure) and water quality processes (pollution with chemical or metals, biological matter) should be considered [1]. Complex process-based models are thus needed to assess the robustness, resilience and vulnerability of WDS against cyber-physical attacks. This is a novel field of research, and only few studies have tackled the problem of understanding the effect of cyber attacks on complex water systems. The authors of [3,4], for example, have recently studied the problem of detection and isolation of attacks on a water network comprised of cascaded canal pools. In this work, we discuss the use of EPANET for modeling the effects of malicious attacks on water distribution systems. EPANET simulates both hydraulic and water quality processes of pressurized pipe networks, so it is in principle well suited for the tasks described above. In the remainder of the paper we introduce some technical aspects of EPANET (Section 2), and then demonstrate its use for simulating cyber-physical attacks (Section 3). In particular, we introduce attacks that manipulate the control strategies of automated pumps and attacks that change the fill level reported by sensors installed in water tanks. Concluding comments are given in Section 4.

2. EPANET

EPANET is a public-domain software developed by the US Environmental Protection Agency to simulate hydraulic and water quality behavior within pressurized pipe networks [14, 15]. EPANET comes both as a self-contained application with a graphical user interface, as well as an open-source development Toolkit (written in C) that can be interfaced with other programming languages. The version used in this work, EPANET2, is de facto the standard modeling tool for both practitioners and researchers in the field of WDS. Typical applications of EPANET include long-term planning of WDS [5], design of alternative management strategies [17], as well as water quality modeling or the assessment of consumers exposure to contaminated water [10, 11]. EPANET accurately reproduces WDS dynamics by employing a sophisticated network model and two engines for hydraulics and water-quality simulation.

2.1. Network Model

EPANET features *physical* components for designing the network topology, and *non-physical* components to define its operations. Physical components are divided into *nodes* and *links*, whereas non-physical components include *curves*, *time patterns* and *controls*.

Table 1. Properties and output variables of EPANET physical components

Component	**Principal input properties**	**Output variables**
Junctions	Coordinates and elevation Water demand Initial water quality	Hydraulic head Pressure Water quality
Reservoirs	Coordinates, Elevation Initial water quality	None (network boundary)
Tanks	Coordinates and bottom elevation Diameter (or shape) Initial, minimum and maximum level Initial water quality and mixing model	Hydraulic head Pressure Water quality
Pipes	Start and end nodes Diameter, Length, Roughness Status Bulk/Wall reaction coefficient	Flow rate, Velocity, headloss Friction factor Average reaction rate Average water quality
Pumps	Start and end nodes Pump curve Status, Speed	Flow rate Head gain Energy consumed
Valves	Start and end nodes Diameter, Setting, Status	Flow rate headloss

2.1.1. Nodes

Nodes can be either *reservoirs*, *junctions* or *tanks*, and represent source, sink and storage points of the WDS. Reservoirs are infinite external sources or sinks of water, e.g., lakes, rivers, groundwater aquifers, and tie-ins to other systems. They are treated as boundary points by the simulation engines. Junctions are points in the network where links join together, and where water enters or leaves the network. They are usually employed to represent the demand nodes in the network, but they can also model water-quality sources. Tanks are modelled as nodes with a given maximum capacity, and whose storage varies in time with the simulation.

EPANET allows the specification of several properties for each physical component, so that the modelled network closely resembles the real one. Table 1 summarizes the customizable input properties and available output variables for each component.

2.1.2. Links

The network nodes are connected by links, which are either *pipes* conveying water from one point to another or actionable components of the WDS, i.e., *pumps* and *valves*. In EPANET, all pipes are supposed to be full at all simulation time steps. The water in the pipes flows from nodes with higher energy—i.e., higher hydraulic head—to those with lower energy. When flowing, water experiences headloss due to friction with the pipe walls. The hydraulic engine of EPANET computes the headloss between the start and end node of a pipe using different empirical formulas, all expressing the headloss as a function of flow rate, length of the pipe and its roughness coefficient. The software also accounts for local headlosses due to turbulence occurring at pipes bends and fittings.

Pumps raise the hydraulic head of the water by imparting energy to it. They are modeled using a pump curve defining all possible combinations of head and flow that

Table 2. Examples of simple controls in EPANET

Control statement	Meaning
VALVE-2 CLOSED IF TANK-1 ABOVE 20	Close VALVE-2 if the level in TANK-1 exceeds 20 ft.
VALVE-2 OPEN IF NODE 130 BELOW 30	Open VALVE-2 if the pressure at Node 130 $<$ 30 psi
PUMP-3 1.5 AT TIME 16	Set relative speed of PUMP-3 to 1.5 after 16 hours
PUMP-3 CLOSED AT CLOCKTIME 10 AM	Shuts down PUMP-3 at 10 AM

a pump can produce, or as a device that supplies a constant amount of energy to the fluid flowing through it. EPANET also features pumps with variable speed—the relative velocity with respect to the original pump curve can be changed during the simulation. A change of speed affects the position and shape of the pump curve.

Valves are links limiting the pressure or flow at a specific point in the network. EPANET supports several types of valves, such as pressure reducing valves (PRV) and flow control valves (FCV). General purpose valves (GPV) can also be employed to let the user define particular components or conditions. Shutoff and check valves (CV), which completely open or close pipes, are also featured in EPANET but as a property of the pipe in which they are placed.

2.1.3. Controls

Controls are statements that determine how the network is operated over time. They determine the functioning of selected links as a function of time or the value of some nodal variables. There are two categories of controls in EPANET. *Simple controls* change the status or setting of a link as a function of either 1) the water level in a tank, 2) the pressure at a junction, 3) the time into the simulation, or 4) the time of the day. Some examples are given in Table 2. *Rule-based controls* extend simple controls with logical operators, thus allowing to command a link based on a combination of conditions.

2.2. EPANET Simulation Models

EPANET's hydraulic engine is demand-driven—consumer demands are always satisfied regardless of the pressures throughout the system. This entails that if nodal demands exceed the maximum flow, the software extrapolates the pump curves to meet the required flow, even if this produces a negative head. Such cases are regarded as unrealistic and the software returns a warning message. EPANET hydraulically balances the network for heads and flows at a particular point in time by solving simultaneously the conservation of flow equation for each junction and the headloss relationship across the links. This process is carried out using the Gradient Algorithm [16].

3. Modeling Cyber-Physical Attacks with EPANET

3.1. Attacker Model

The goal of the attacker is the disruption of the network's hydraulic behavior. The attacker can achieve this through *direct* and *indirect* attacks. In a direct attack, the attacker takes direct control of an actionable component, e.g., by physical manipulation, remote compromise of the controller, or manipulation of the control messages. In an indirect attack, the attacker manipulates the reported reading of a nodal sensor (by either physical manipulation, remote compromise of the sensor, or manipulation of the network traffic). The manipulated sensor reading then leads to incorrect control action to be taken. Both attacks are altering the normal operations of pumps and valves in the network.

3.2. Attack Impact Metric

To better characterize an attack, appropriate criteria should be defined to assess the level of service disruption in the network. For instance, pressure-related criteria can be employed to check whether the pressure at the demand nodes is within an optimal functioning range. These measures may be evaluated on the whole network or on specific zones affected by the attack. In case storage tanks are targeted, the amount of overflow witnessed during the attack can be used to estimate its impact. Other measures may consider economic costs, such as those deriving from damage to pumps and valves.

3.3. EPANET-based Implementation of Cyber-Physical Attacks

Since EPANET was not developed to study WDS behavior from a cyber-physical perspective, some workarounds are necessary to model attack scenarios. The degree of flexibility needed for such implementation is achieved through the programming Toolkit. In this study, threats are simulated by modifying the control statements commanding the targeted components in the network. The attacks strike at a chosen time during the simulation and last for a given period, during which the control statements are altered consistently. This procedure is carried out by running the hydraulic simulation in a step-by-step fashion and interposing extra conditional constructs between consecutive steps of the simulation.

3.3.1. The C-TOWN Network

The implementation of cyber-physical attacks in EPANET is illustrated with some examples for the C-Town benchmark network [7], which is depicted in Figure 1. C-Town is based on a real-world medium-sized network with two main storage tanks (T1 and T2), and a main pumping station (S1) consisting of three pumps (PU1, PU2 and PU3). The operations of PU1 and PU2 are regulated by the water levels in T1, while PU3 is a redundant pump—it is kept off during standard operating conditions. The remaining five tanks are refilled by four secondary boosting stations that pump water from T1 and T2. While T1 is directly connected to S1, the branch pertaining T2 is connected to the source through a FCV (V1) controlled by the water level in T2.

The attack scenarios described in the following sections were generated for one week-long simulations (168 hours), and performed with an hydraulic time step of 15 minutes. The demand was generated by 5 different week-long time patterns with a time step of 1 hour. The attacks start 48 hours into the simulation and last for 2 whole days.

3.3.2. Example 1: Overflow of T1 via Attack on PU1 and PU2

The first example is an indirect attack on S1 aimed at producing an overflow of T1. This scenario assumes that there is no additional overflow sensor in the tank or that it has been deactivated by the attacker. In normal conditions, the pumps PU1 and PU2 activate and deactivate according to the following controls written in the input file:

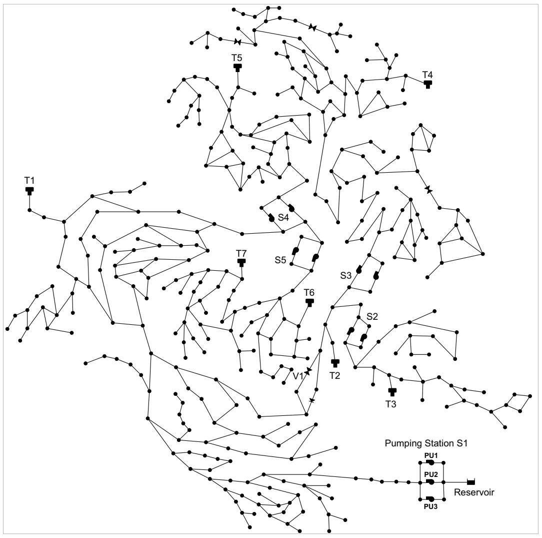


Figure 1. Subset of the C-TOWN network. We focused on tanks T1 and T2, the pumping station S1, and the main valve V1.

```
LINK  PU1  OPEN    IF  NODE  T1  BELOW  4
LINK  PU1  CLOSED  IF  NODE  T1  ABOVE  6.3

LINK  PU2  OPEN    IF  NODE  T1  BELOW  1
LINK  PU2  CLOSED  IF  NODE  T1  ABOVE  4.5
```

The attack consists in tampering the readings of the water level in T1 so that the value transmitted to the controller is always below 4.5 meters. In this way, once started, PU1 and PU2 keep pumping water to the system causing T1 to overflow when the level rises above the maximum capacity of 6.5 meters. This attack is simulated by temporarily changing the settings at the end of the second and fourth control statements to a value above 6.5 meters, so that the pumps never switch to the CLOSED status when the attack is in place. Since EPANET cannot directly model tank overflows, the original network map is modified to amend for such shortcoming. This is done by 1) duplicating the original pipe connecting the tank to the network, 2) connecting it to a dummy storage tank, and 3) including controls that keep the link closed unless the level in the original tank reaches the maximum capacity. `NEWLINK` represents this additional pipe, and the additional control statements for modeling the overflow of T1 are written as:

```
LINK  NEWLINK OPEN    IF  NODE  T2  ABOVE  6.499999
LINK  NEWLINK CLOSED  IF  NODE  T2  BELOW  6.499999
```

The amount of overflow due to the attack is therefore equal to the amount of water stored in the dummy tank at the end of the attack, or to the volume of water that flows through `NEWLINK` during the attack. The results are illustrated in Figure 2, where the attack scenario is compared against the time series expected for the same initial conditions under normal operations. Due to the system inertia and prior state, it takes around 10 hours for the attack to drive the system outside its normal regime, while almost the entire duration of the attack is needed to cause T1 to overflow.

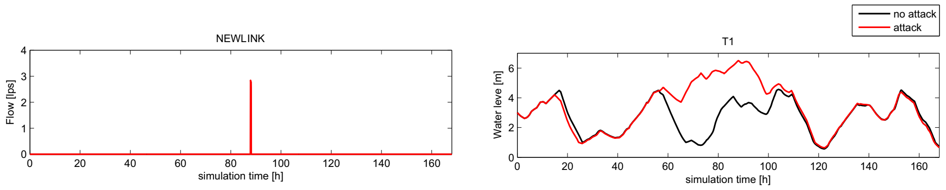


Figure 2. Trajectory of water flow in `NEWLINK` (left) and storage in T1 (right) during normal operations (black line) and under attack (red line). Results show that, when the attack strikes, a large volume of water overflows from T1.

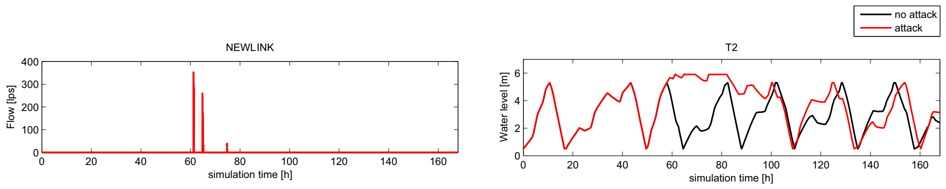


Figure 3. Trajectory of water flow in `NEWLINK` (left) and storage in T2 (right) during normal operations (black line) and under attack (red line). Results show that the attack leads to large overflow at T2.

3.3.3. Example 2: Overflow of T2 via Attack on V1

Overflow in T2 occurs if the valve connecting it to the main line of the network is forced to stay open for a sufficient amount of time. In this case, the attacker intercepts the communication to the valve controller and tampers the real readings of the water level in T2. Under normal conditions, the controller operates V1 as follows:

```
LINK  V1  OPEN    IF  NODE  T2  BELOW  0.5
LINK  V1  CLOSED  IF  NODE  T2  ABOVE  5.3
```

In this example, the valve is forced to stay in the OPEN status by substituting the setting at the end of the second statement to a value above the maximum level of T2 (for example, 5.9 meters). The effects of this attack are shown in Figure 3—under the same assumptions as for the first example, with dummy tank connected to T2.

4. Conclusion

In this work we discuss the opportunities and challenges of simulating cyber-physical attacks to WDS using EPANET—a numerical modeling environment commonly adopted to simulate WDS behavior under standard operating conditions. We present two examples, related to the overflow of storage tanks due to the direct attack to pumps controls and manipulation of valves settings. The examples show that some workarounds are necessary to implement these attacks. A limit of EPANET that surely deserve further research is the assumption that the consumers' demand is always satisfied (regardless of the pressure at the demand nodes). This implies that EPANET cannot simulate sudden shortfalls of water supply as well as other pressure-related issues, such as pipe bursts. Adopting pressure-driven numerical simulation environments seems to be a viable option; this is subject of ongoing research.

Acknowledgments

This work was supported by the National Research Foundation (NRF), Singapore, under its National Cybersecurity R&D Programme (Award No. NRF2014NCR-NCR001-40), by the Chief Scientist (OCS) Ministry of Science, Technology and Space (MOST), and by the Germany Federal Ministry of Education and Research (BMBF), under project no. 02WA1298.

References

[1] M. Abrams and J. Weiss. Malicious control system cyber security attack case study–maroochy water services, australia.

[2] S. Amin, X. Litrico, S. Sastry, and A. Bayen. Cyber security of water SCADA systems; Part I: Analysis and experimentation of stealthy deception attacks. *Control Systems Technology, IEEE Transactions on*, 21(5):1963–1970, 2013.

[3] S. Amin, X. Litrico, S. Sastry, and A. M. Bayen. Cyber security of water scada systemspart i: analysis and experimentation of stealthy deception attacks. *Control Systems Technology, IEEE Transactions on*, 21(5):1963–1970, 2013.

[4] S. Amin, X. Litrico, S. S. Sastry, and A. M. Bayen. Cyber security of water scada systemspart ii: Attack detection using enhanced hydrodynamic models. *Control Systems Technology, IEEE Transactions on*, 21(5):1679–1693, 2013.

[5] G. Fu, Z. Kapelan, J. R. Kasprzyk, and P. Reed. Optimal design of water distribution systems using many-objective visual analytics. *Journal of Water Resources Planning and Management*, 2012.

[6] W. Gao. *Cyberthreats, attacks and intrusion detection in supervisory control and data acquisition networks*. PhD thesis, Mississippi State University, December 2013.

[7] O. Giustolisi, L. Berardi, D. Laucelli, D. Savic, and Z. Kapelan. Operational and tactical management of water and energy resources in pressurized systems: Competition at WDSA 2014. *Journal of Water Resources Planning and Management*, page C4015002, 2015.

[8] O. Kosut, L. Jia, R. Thomas, and L. Tong. Malicious data attacks on smart grid state estimation: Attack strategies and countermeasures. In *Proc. of the IEEE Conference on Smart Grid Communications (SmartGridComm)*, pages 220–225, Oct 2010.

[9] Y. Liu, P. Ning, and M. K. Reiter. False data injection attacks against state estimation in electric power grids. *ACM Transactions on Information and System Security (TISSEC)*, 14(1):13, 2011.

[10] A. Ostfeld and E. Salomons. Optimal layout of early warning detection stations for water distribution systems security. *Journal of Water Resources Planning and Management*, 130(5):377–385, 2004.

[11] A. Ostfeld, J. G. Uber, E. Salomons, J. W. Berry, W. E. Hart, C. A. Phillips, J.-P. Watson, G. Dorini, P. Jonkergouw, Z. Kapelan, et al. The battle of the water sensor networks (bwsn): A design challenge for engineers and algorithms. *Journal of Water Resources Planning and Management*, 134(6):556–568, 2008.

[12] Industrial Control Systems Cyber Emergency Response Team. Malware infections in the control environment. *ICS-CERT Monitor*, pages 1–15, 2012.

[13] North American Electric Reliability Corporation. *NERC Issues AURORA Alert to Industry*. 2010.

[14] L. A. Rossman. The EPANET programmers toolkit for analysis of water distribution systems. In *Proceedings of Conference on Water Resources Planning and Management*, pages 39–48, 1999.

[15] L. A. Rossman. EPANET 2: users manual, 2000.

[16] E. Todlini and S. Pilati. A gradient method for the analysis of pipe networks. In *Proceedings of the conference on computer applications for water supply and distribution*, 1987.

[17] J. E. Van Zyl, D. A. Savic, and G. A. Walters. Operational optimization of water distribution systems using a hybrid genetic algorithm. *Journal of water resources planning and management*, 130(2):160–170, 2004.

[18] L. Xie, Y. Mo, and B. Sinopoli. False data injection attacks in electricity markets. In *Proc. of the IEEE Conference on Smart Grid Communications (SmartGridComm)*, pages 226–231, Oct 2010.

Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016
A. Mathur and A. Roychoudhury (Eds.)
© 2016 The authors and IOS Press.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License.
doi:10.3233/978-1-61499-617-0-131

ECG Steganography Using Contourlet Transform For Transmission of Secured Patient Identity

Edward Jero SAM JEEVA RAJ[a,1] and Palaniappan RAMU[b]
[a,b] *Department of Engineering Design, Indian Institute of Technology Madras, India*

Abstract. ECG steganography hides patient data inside their ECG signal to ensure the protection of patient's identity. In this work, an attempt has been made to evaluate ECG Steganography where Quantization Index Modulation (QIM) method is used to embed patient data into the Contourlet transform coefficients of ECG image. Tompkins QRS detection algorithm is used to construct ECG image and which is decomposed into a series of multiscale, local and directional image expansion using contour segments. QIM scheme is applied on the appropriate coefficients of the selected frequency sub-bands. QIM divides the selected frequency sub-bands into non overlapping blocks of matrix size 2×2. Two quantizers are used to embed binary watermark 0 and 1. The reverse Contourlet transform provides the watermarked ECG image from which 1D stego-ECG signal is re-ordered. The proposed scheme retrieves the watermark without the need of cover image. The deterioration due to the modified coefficients are reduced using adaptive selection of a coefficient to hide watermark bit. The efficiency of ECG steganography is measured using imperceptibility of watermark and Bit Error Rate of the retrieved watermark. Similarity metrics such as Peak Signal to Noise ratio, Percentage Residual Difference and Kullback-Leibler distance are used to estimate the imperceptibility of watermark on the stego-ECG signal. It is demonstrated that the proposed approach provides higher imperceptibility and less Bit Error Rate in the retrieved watermark, which is closed to zero. ECG signals obtained from MIT-BIH normal sinus rhythm data base are used to evaluate the performance of the proposed ECG Steganography approach.

Keywords. ECG steganography, ECG image, Quantization Index Modulation, Watermarking, Performance metrics

1. Introduction

Development of reliable wearable sensors and communication systems had lead to the increase of remote health monitoring systems. In this framework, the patient's bio physiological signals are to be transmitted to the care giver. During such transmissions, the bio-physiological signals are usually tagged with the patient's identity information and it is subject to unauthenticated access if not protected properly. There are different ways in which the patient data can be protected. One of the widely used data hiding technique in medical domain is steganography [1]. Patient data consists of attributes such as patient name, age and location which is referred to as watermark. Bio-physiological signals such as ECG [2] and EEG [3] are the cover signals.

[1] Edward Jero SAM JEEVA RAJ.

Steganography leads to deterioration of signal and its performance is evaluated by its ability to minimally deteriorate the cover signal, thus preserving diagnosability and retrieval of the entire watermark. Usually, watermark is embedded into the transformed coefficients of biomedical signals.

In this study, we use Contourlet transforms to identify the characteristic points of ECG signal and embed the watermark into the least significant components of ECG signal using Quantization Index Modulation (QIM) based watermarking scheme. Similarity metrics are used to estimate the performance of the current approach. The proposed method is demonstrated using 5 different ECG signals obtained from MIT-BIH database [4] and it is observed that reliable patient information transfer is possible.

2. Methodology

The proposed method consists of ECG image conversion, watermarking and extraction. Binary formatted alphanumeric patient data is referred as watermark. Two major components of ECG steganography are cover image and watermark. Cover image is a 1D ECG signal arranged in 2D matrix format according to the fiducial points obtained using QRS complexes of ECG signal [5]. Contourlet transform decomposes the cover image into frequency sub bands. Watermark is embedded into the coefficients of the selected sub band using Quantization Index Modulation (QIM) scheme. The inverse Contourlet transform results in watermarked ECG image from which 1D stego-ECG signal can be obtained. Efficiency of ECG steganography is measured using parameters such as imperceptibility of watermark and data loss due to steganography process. Similarity metrics such as PSNR, PRD and KL distance measures the imperceptibility of watermark in the stego-ECG signal and error bits in retrieved patient data is estimated using BER. Figure 1 shows the architecture of proposed Contourlet transform based ECG steganography. Hereafter, cover image and watermark are referred to 2D ECG image and binary formatted patient data respectively.

2.1. 2D conversion of ECG signal

In the current scheme, 1D ECG signal is arranged in 2D matrix using Tompkins QRS detection algorithm [5] and used as cover image. First, a recorded ECG signal passes through bandpass filter. Second, a filtered signal is subjected to differentiation, squaring and moving window integration in order, which provides QRS complex slope information. Identified peaks of R waves are referred as fiducial points. Since the sampling frequency is 128Hz, 64 samples on both sides of each fiducial points forms ECG cycles and aligned in a 2D matrix. There is negligible data loss in this conversion process.

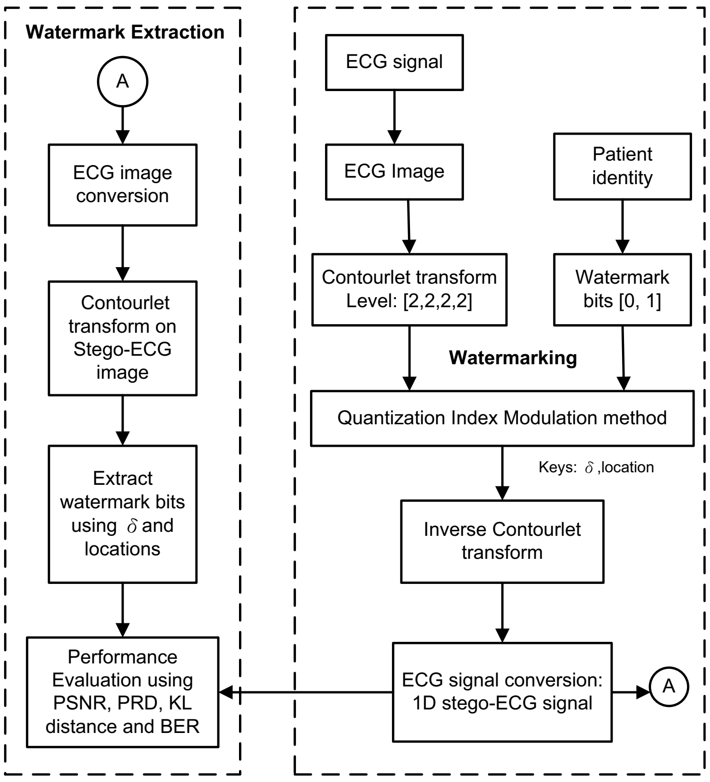

Watermark Extraction
A
ECG image conversion
Contourlet transform on Stego-ECG image
Extract watermark bits using δ and locations
Performance Evaluation using PSNR, PRD, KL distance and BER
ECG signal
ECG Image
Patient identity
Contourlet transform Level: [2,2,2,2]
Watermark bits [0, 1]
Watermarking
Quantization Index Modulation method
Keys: δ ,location
Inverse Contourlet transform
ECG signal conversion: 1D stego-ECG signal
A

Figure 1. Contourlet transform based ECG steganography architecture.

2.2. Discrete Contourlet Transform Coefficients

Contourlet transform provides multi scale and directional representation of an image using Pyramidal Directional Filter Bank (PDFB) that consists of Laplacian Pyramid (LP) filter and Directional Filter Bank (DFB) as shown in Figure 2. LP provides bandpass cover image that is the difference between cover image and the down sampled cover image. DFB captures the high frequency components of the cover image. The decomposition levels are predefined. The readers are referred to [6] for details on Contourlet transform. Characteristic points of ECG signal involved in diagnosis lies on low frequency components that need to be preserved. Therefore, high frequency sub band is the logical choice to embed the watermark which can be identified using Contourlet transform.

2.3. Quantitative Index Modulation based adaptive watermarking method

Watermark bits 0 and 1 are embedded into the Contourlet coefficients of chosen frequency sub-band using QIM method [7,8]. In order to achieve that the selected frequency band of matrix size 64×64 is divided into non-overlapping blocks of matrix size 2×2, its pictorial representation is depicted in Figure 3. The proposed scheme estimates the quantization step size of each block using Eq. (1) which is denoted as δ .

Since the possible values are 0 and 1, two quantizers are preferred for watermarking and one bit is embedded into each block.

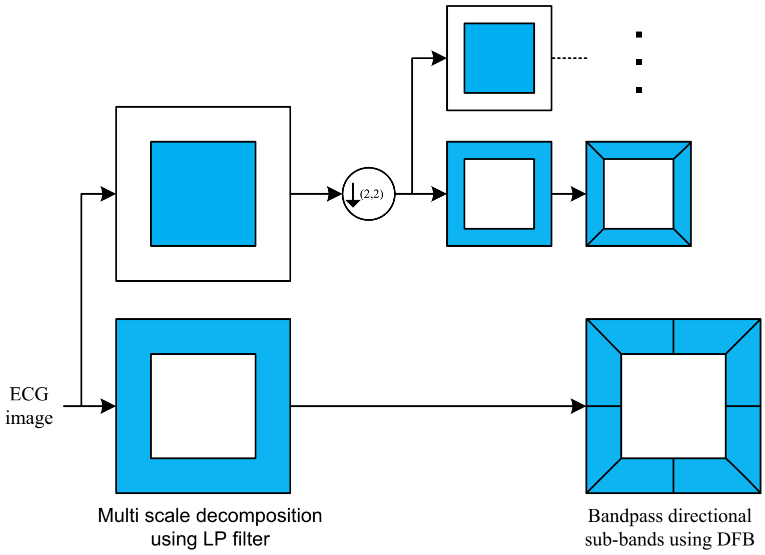

ECG image
(2,2)
Multi scale decomposition using LP filter
Bandpass directional sub-bands using DFB

Figure 2. Filter banks of Contourlet transform.

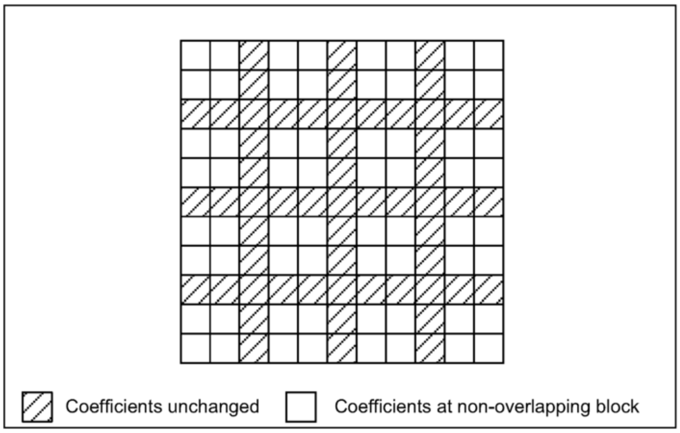


Figure 3. Pictorial representation of blocks in selected frequency band

$$\delta = (C_{\max} - C_{\min})/l \qquad (1)$$

where, $C_{\max}$ and $C_{\min}$ are the absolute highest and lowest coefficient of a block respectively. l is the length of the coefficients in each block (2×2) which is 4. The watermark bit 1 is hidden into a block using Eq. (2) where the coefficient with highest absolute magnitude is chosen to be modified in order to embed the watermark. Watermark bit zero is embedded into the coefficient with lowest absolute magnitude of a block using Eq. (3). The scaling factors 0.75 and 0.25 in Eq. (2) and (3) are obtained using experimental studies.

$$C^*_{\max} = C_{\max} + 0.75\delta \tag{2}$$

$$C^*_{\min} = 0.25\delta \tag{3}$$

where, $C^*_{\max}$ and $C^*_{\min}$ are the modified highest and lowest coefficients respectively. However, the signs of coefficients remain unchanged. The indices of the modified coefficients and δ values of each block are loaded into a scrambling matrix and provided for watermark extraction.

2.4. Watermark extraction method

Efficiency of ECG steganography is determined using the error in the extracted watermark bits from the steg-ECG signal. First, 1D steg-ECG signal is converted into 2D ECG image. The authenticated user has the key of scrambling matrix which contains the location of watermarked coefficients and δ values. Steg-ECG is subjected to Contourlet transform and the watermark bits are extracted using Eq. (4). Here, the inverse Contourlet transform alters the magnitudes of modified coefficients which affect the extraction of watermark from the steg-ECG signal. Therefore, experimental study is performed to extract the watermark with different α values through which the appropriate value is set as 0.75.

$$W^* = \begin{cases} 1, & if \;\; C^*_{ret} \geq \alpha\delta \\ 0, & else \end{cases} \tag{4}$$

2.5. Performance metrics

Efficiency of the proposed method is determined by the parameters such as imperceptibility of watermark and error in retrieved watermark. Similarity measures such as PSNR, PRD and Kullback-Leibler metrics quantifies imperceptibility of watermark. Data loss in the retrieved watermark due to ECG steganography is estimated using BER metrics. The performance of the proposed method is evaluated using these metrics. The readers are referred to [9] for detailed study of these metrics.

3. Experiment

3.1. Database

The proposed method uses ECG signals obtained from MIT-BIH normal sinus rhythm database [4] to demonstrate ECG steganography. The database contains long-term ECG signals recorded for 18 subjects with no significant arrhythmias. The sampling frequency of the signals is 128Hz.

3.2. ECG image conversion

The 2D ECG image of a 1D ECG signal (16420m) obtained using Tompkins QRS detection algorithm [5] as depicted in Figure 4. The cover image is decomposed into frequency sub-bands using Contourlet Transform.

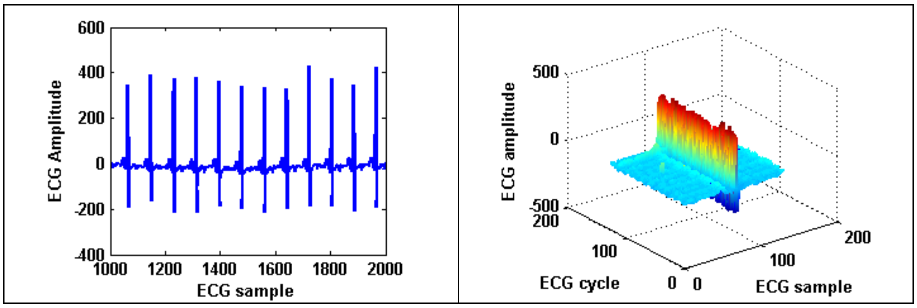


Figure 4. 16420m (a) 1D ECG signal (b) ECG image.

3.3 Frequency analysis of ECG image

In the proposed scheme, decomposition of ECG image is performed using LP and DFB filters. LP is a low pass filter and four pyramidal levels are selected for decomposition. Four levels of DFB decomposition is performed in each pyramidal level. The cover image of matrix size 128×128 is decomposed to four pyramidal levels and frequency sub-bands obtained using DFB for the fourth pyramidal level are presented in Figure 5 where each sub band of size 64×64 contains the coefficients of high frequency components and preferred to embed the watermark bits.

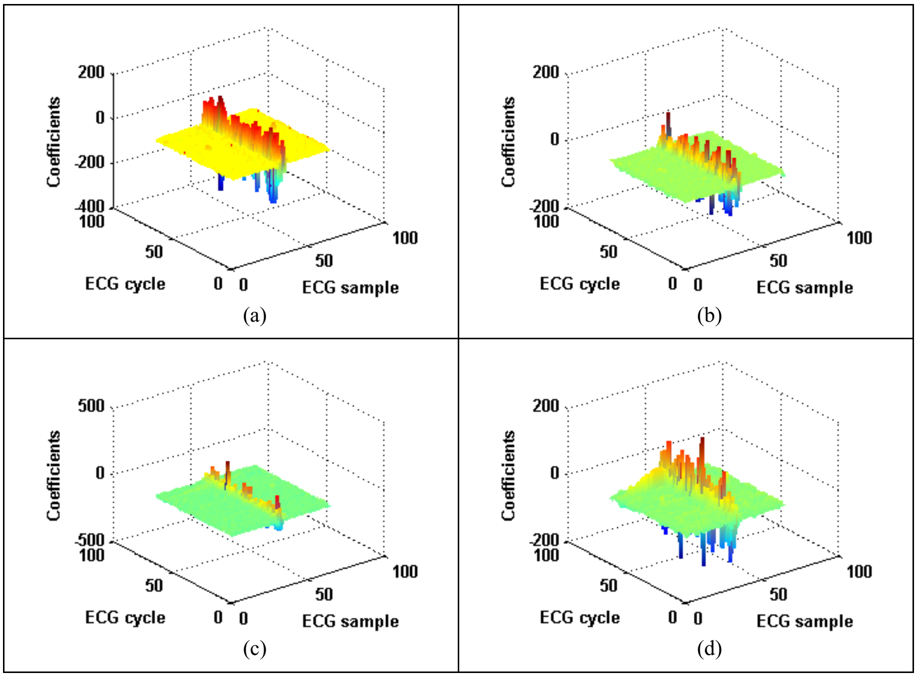


Figure 5. 16420m (a) 1D ECG signal (b) ECG image.

3.4 Performance of ECG steganography

QIM based adaptive watermarking method embed the watermark bits into the selected frequency band using quantizers. The inverse Contourlet transform of modified coefficients provides the stego-ECG image which is converted into 1D stego-ECG signal. Figure 6 shows a window of size 3000 ECG samples taken from the 1D cover (X_c) ECG signal 16420m (blue line) and its corresponding 1D stego-ECG signal (X_w) drawn on it (red color dashed line). It is observed that the cover and stego-signal are very similar. However, the efficiency of the proposed scheme is measured by quantifying the similarities between 1D cover and stego-ECG signals using PSNR, PRD and KL distance.

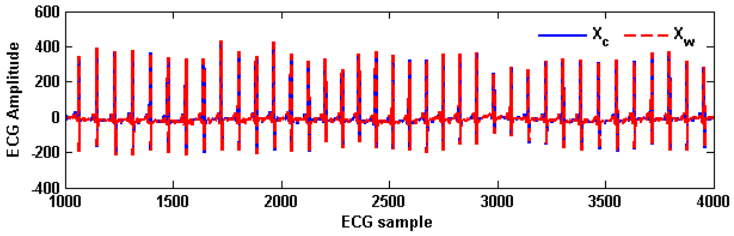


Figure 6. Original and stego-ECG signals of 16420m

The proposed method is demonstrated on 5 different signals. The watermark of size 441 bits are embedded into one frequency sub-band at a time. The corresponding performance metrics are provided in Tables 1-4. It can be readily observed that the metrics vary across the bands. However, in the fourth band, all signals allow for 0% BER. The other sub-bands allow negligible BER. It is also observed that the efficiency of the proposed scheme depends on the magnitude of the selected coefficients, size of watermark. Experimental study reveals that embedding of bit zero into the coefficient with higher magnitude and the reverse operation affects the data retrieval rate which increases the BER.

Table 1. Performance metrics of ECG steganography for the first sub-band

Signal	PSNR (db)	PRD	KL distance	BER
16272m	56.62	0.01154	0.0011	0.45
16420m	54.26	0.0145	0.0015	0.68
16483m	52.22	0.014	0.02	0.68
16539m	55.03	0.012	0.0003	0.45
17052m	56.98	0.012	0.0012	0.22

Table 2. Performance metrics of ECG steganography for the second sub-band

Signal	PSNR (db)	PRD	KL distance	BER
16272m	64.41	0.004	0.00002	0
16420m	59.06	0.008	0.001	0.22
16483m	57.57	0.0078	0.00006	0.45
16539m	62.52	0.005	0.00009	0.68
17052m	64.38	0.0053	0.013	0.22

Table 3. Performance metrics of ECG steganography for the third sub-band

Signal	PSNR (db)	PRD	KL distance	BER (%)
16272m	59.66	0.008	0.00013	0.23
16420m	58.06	0.009	0.003	0.22
16483m	54.74	0.01	0.00005	0
16539m	55.29	0.011	0.0027	0.45
17052m	62.92	0.006	0.0004	0

Table 4. Performance metrics of ECG steganography for the fourth sub-band

Signal	PSNR (db)	PRD	KL distance	BER
16272m	55.78	0.012	0.00011	0
16420m	57.91	0.009	0.0015	0
16483m	52.54	0.014	0.00015	0
16539m	55.65	0.011	0.0011	0
17052m	59.88	0.009	0.00035	0

4. Conclusions

ECG steganography ensures the protection of patient identity during storage and transmission of medical records over the internet. In this study, an attempt has been made to achieve ECG steganography using Contourlet transform based QIM scheme. The major challenge of ECG steganography are signal deterioration and patient data loss due to watermarking. The proposed method identifies the characteristic points of ECG signal such as QRS complex, P and T waves using Contourlet transform and embed the watermark bits into their least significant frequency components to preserves the diagnosability. A blind QIM scheme reduced the changes in coefficient's magnitude owing to watermarking that improves the imperceptibility of watermark in the stego-ECG signal which is demonstrated using the metric PSNR. In addition, the difference between cover and stego-ECG signals are estimated using PRD and KL distance. Zero BER is achieved for the watermark of size 441bits. Hence, the proposed method is feasible to perform ECG steganography in order to protect the patient's identity.

References

[1] R. C. Raúl, F. U. Claudia, and G. D. J. Trinidad-Blas, Data hiding scheme for medical images, *In proc:17th International Conference on Electronics, Communications and Computers, CONIELECOMP'07* (2007).

[2] A. Ibaida and I. Khalil, Wavelet-based ECG steganography for protecting patient confidential information in point-of-care systems, *IEEE Trans. Biomed. Eng.*, 60 (2013), 3322–3330.

[3] X. Kong and R. Feng, Watermarking medical signals for telemedicine, *IEEE Trans. Inf. Technol. Biomed.*, 5 (2001), 195–20.

[4] A. L. Goldberger et al., PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals, *Circulation* 101 (2000), E215–E220.

[5] J. Pan and W. J. Tompkins, A real-time QRS detection algorithm, *IEEE Trans. Biomed. Eng.*, 32 (1985), 230–236.

[6] M. N. Do and M. Vetterli, The Contourlet Transform: An Ef cient Directional Multiresolution Image Representation, *IEEE Trans. Image Process.*, 14 (2005), 1–16.

[7] B. Chen and G. W. Wornell, Quantization index modulation methods for digital watermarking and information embedding of multimedia, *J. VLSI Signal Process. Syst. Signal Image. Video Technol.*, 27 (2001), 7–33.

[8] Malik et al., Steganalysis of GIM-based data hiding using kernel density estimation. *In Proc:9th workshop on Multimedia and security - MMSec'07* (2007).

[9] S. Edward Jero, P. Ramu, and S. Ramakrishnan, Discrete Wavelet Transform and Singular Value Decomposition Based ECG Steganography for Secured Patient Information Transmission, *J. Med. Syst.*, 38 (2014).

Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016
A. Mathur and A. Roychoudhury (Eds.)

doi:10.3233/978-1-61499-617-0-139

Accurate in-network file-type classification

Dinil Mon Divakaran, Yung Siang Liau, and Vrizlynn L. L. Thing
*A*STAR Institute for Infocomm Research (I^2R), Singapore*

Abstract. Accurate classification of file types carried by network traffic aids in securing a network against various types of malicious activities such as malware infection, data exfiltration, botnet communication, etc. An important challenge here is to accurately classify files without slowing down network traffic. Therefore, the cost of accurate file-type classification has to be known. In this work, we carry out a preliminary but extensive investigation to evaluate different sets of features for file-type classification. The objective is to detect not only file types under normal scenario, but also files that are transferred with obfuscated headers. Our experiments show that the feature vector consisting of unigram frequencies leads to high accuracy; yet, combining this feature set with entropy feature vector leads to improvement in accuracies.

Keywords. Classification, files, network, real-time, n-grams, entropy

1. Introduction

Network traffic is an important medium for security breaches, and therefore for security analysis. Exfiltration of sensitive data from an enterprise, download of malicious executables, covert communications, sharing of copyrighted software, etc. happen over a network. In this work, we conduct preliminary research to detect the types of contents or files carried by traffic flows. Real-time content classification aids in securing a network, be it an enterprise network, an ISP network or a government organization. For example, a financial enterprise often blocks encrypted attachments in mails, so that the administration can monitor the contents of email exchanges. ISPs are interested in blocking malware (which are typically in binary format) before they reach end-users. Similarly, a government organization engaging in tenders might want to scrutinize files before they leave the network. Yet, another organization network, like that of a retail store, would not expect any kind of binary executables or codes in their traffic. Identifying file types is also required in forensic analysis, for example, to detect file fragments and recover files from computers and mobile systems [1].

While accuracy of file-type classification is paramount to its utility, the efficiency of classification cannot be ignored—such classification solutions are typically deployed at the perimeters of a network. Routers at the edges of a network are connected using high capacity of links, and depending on the network load, a router sees large numbers of flows and packets traversing through it. Given this realistic scenario of deployment, it is necessary to consider the cost of classifying file types carried by network traffic. Cost is in terms of, both, the number of bytes or packets that need to be buffered, and the computational runtime required for feature extraction (and classification).

In [2], the authors study a number of features (and their combinations) such as n-grams, distribution frequencies as well as Shannon's entropy and Kolmogorov complexity, for classifying file types. While the evaluation is done for 30 file types, fragments of 512 bytes (for both training and testing) are extracted from a small number of files—2,587. This naturally makes sampled fragments dependent, potentially affecting results. Overall accuracy of 73.4% is reported, using a combination of unigram and bigram frequencies. But feature vector of bigram frequencies has a dimension of 65,536, making it impractical for real-time classifications.

In a recent research work [3], block entropies (refer Section 2.3) are used for classification of files in network traffic. The work compares results of classification using the byte-streams of variable sizes from both the beginning of file and random locations. We compare block entropies in our evaluations as well (using the same feature vector as in [3]). From our experiments, we observe that block entropies (which require higher computational time) are not required for accurate classification of files using the beginning of the file; indeed unigram frequencies feature vector alone gives high accuracy in this scenario. For streams extracted from random locations, a combination of feature vectors (including block entropies) perform better than a feature vector consisting of only block entropies.

In this paper, we differentiate features based on the costs, and evaluate the accuracy of classification using different feature sets and their combinations, while varying the number of bytes required for classifying a flow. We describe the different feature vectors used for classification in the following section. Subsequently, we evaluate the goodness of different feature vectors in classifying file types, using a database of around 60,000 files and nine file types.

2. Feature vectors for classifying file types

2.1. n-gram frequencies

Given a stream of bytes, a commonly used set of features is the n-gram frequencies [2,4,5]. In case of unigram, the frequencies of bytes are stored in a vector of dimension 256. The size of the feature vector increases exponentially with increasing size n-grams (or n), and so does storage space required for computation of the feature vector. Besides, our experiments have shown that bigrams and higher orders do not bring in additional value in comparison to unigram frequencies for file-type classification. Therefore, we use a feature vector consisting of only unigram frequencies in this work. The computation of unigram frequencies is straight-forward—for a byte-stream of size m, a single pass over the m bytes gives the frequencies of unigrams.

2.2. Moments of the probability distributions

The feature vector consists of mean, standard deviation, skewness and kurtosis. Skewness for a stream of bytes is a measure of symmetry of distribution of bytes.

For sample values, the skewness is calculated as, $b_1 = \frac{1}{m}\frac{\sum_{i=1}^{m}(x_i-\bar{x})^3}{\bar{\sigma}^3}$, with $\bar{x}$ and $\bar{\sigma}$ being the mean and standard deviation, respectively, of the m samples.

The fourth measure is kurtosis, which characterises the peakedness of a distribution. The kurtosis of m samples, $k = \frac{1}{m}\frac{\sum_{i=1}^{m}(x_i-\bar{x})^4}{\bar{\sigma}^4}$. While still capturing the characteristics of the byte distribution in a given stream similar to unigram frequenies, the feature vector here has a much small dimension of four.

2.3. Block entropy

Entropy a measure of randomness of a random variable. For a discrete random variable X taking values in $\mathcal{X}$, entropy is defined as: $H(X) = -\sum_{x\in\mathcal{X}} p(x)\log(x)$, where p is the probability mass function. Following conventions, we take $0\log 0 = 0$. We measure entropy in bits, i.e., the log is taken to the base 2.

For contents in a given file, $\mathcal{X}$ is the set of all values a byte takes. However, the above computation of entropy can be generalized for byte-blocks as well. For example, when we consider a block of byte one, $|\mathcal{X}^1| = 2^8$. For blocks (or equivalently, sequences) of two bytes, $|\mathcal{X}^2| = 2^{8*2}$. Hence, we can generalize the entropy computation over a block of i bytes as: $H_i(X) = -\sum_{x\in\mathcal{X}^i} p(x)\log(x)$.

Block entropies were used in [3] for file-type classification. The complexity of computing block entropy is $\mathcal{O}(m^2)$ for a stream of m blocks.

2.4. Entropy of data from Markov information source

When the data source has a Markovian property (present is dependent on the past), as is the case with text documents, the probability of a byte in a sequence is conditional on the probabilities of occurrences of the preceding i bytes. Here i is referred to the order or the memory that needs to be recorded. For a Markov information source of order $i-1$, entropy is computed as follows;

$$H_i^M = \sum_{x_1\in\mathcal{X}} p(x_1) \sum_{x_2\in\mathcal{X}} p(x_2|x_1)... \sum_{x_i\in\mathcal{X}} p(x_i|x_1x_2...x_{i-1})\log p(x_i|x_1x_2...x_{i-1}) \qquad (1)$$

The runtime complexity of computing H_i^M is $\mathcal{O}(m^i)$ for order $i-1$, where m is the number of bytes in the stream.

3. Performance evaluation

This section presents results form the experiments conducted. Before proceeding, we discuss on the data used for evaluation purposes, and the experiment settings.

3.1. Data

We collected around 60,000 files for training and testing purposes. The different file types forming the data are: encrypted (*enc*), executables (*exe*), PDF (*pdf*), JPG (*jpg*), PNG (*png*), zip (*zip*), gzip (*gz*), MP3 (*mp3*), plain text and HTML.

Plain text and HTML are considered as a single class, called the transparent class (*trans*). Most of these files are extracted from a Linux system. All documentations were installed to extract a number of text, HTML and PDF files. A subset of PDF files also came from personal research papers collected over the years. Image files in different formats were extracted from the Linux system as well as downloaded from the Internet. Executable files consist of Linux system commands. MP3 files were downloaded from sites offering free and legal downloads of music.

The compressed files, archives and encrypted files where created by randomly sampling files from the database (MP3 and executable files were excluded from sampling). We performed sampling without replacement; therefore the total number of files used for training and test still remained the same. To obtain encrypted files, we encrypted a file using one of the following ciphers with equal probability: AES-256, CAST-128, Blowfish, Twofish, and Triple DES.

3.2. Settings

Unless otherwise specified, we use nine file types (classes) for our experiments—enc, exe, gz, jpg, mp3, pdf, png, trans and zip. Depending on the size of the byte stream being extracted, the number of files used for each class changes. For example, for the scenario where 64-byte streams are extracted from the beginning of a file, the number of files used for each class in the training phase was 5000. The number of files used for testing varied across class; the minimum is 500 and the maximum was 3700 for this particular scenario.

We use a 3:1 ratio for the number of files used for testing and training (the actual number of files is dependent on the stream-size being extracted). As a rule, we do not consider a file-type if the number of files available is less than 500 for any of the two phases of training and testing; and this will be explicitly specified below, as and when such cases arise.

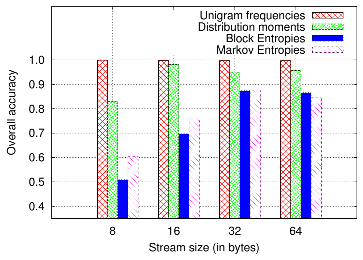
We use Random Forests for classification, as it can efficiently handle large number of features, over a dataset of considerable size. Unless otherwise specified, the four different feature vectors used in the scenarios below are i) Unigram frequencies (Section 2.1), ii) Distribution moments (Section 2.2), iii) Block entropies—feature vector being H_1, H_2, H_3, H_5, as in [3] (Section 2.3), and iv) Markov entropies—feature vector being H_1^M, H_2^M, H_3^M (Section 2.4). Note that the feature H_1 and H_1^M are the same.

3.3. Metrics for evaluation

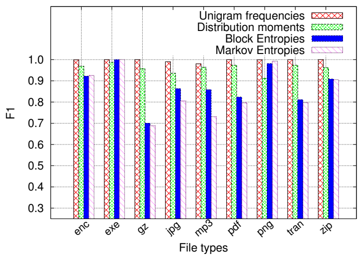
To evaluate the classifiers using different sets of features, we use two commonly and widely adopted metrics in classification, namely overall accuracy and F_1 score. Overall accuracy, or simply accuracy, is the ratio of the sum of correctly predicted files (across all file types) to the total number of files. For a file type, F_1 score $= 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$, where precision $= \frac{\text{No. of True Positive}}{\#(\text{No. of True Positive + No. of False Positive})}$, and recall $= \frac{\#\text{No. of True Positive}}{\#(\text{No. of True Positive + No. of False Negative})}$.

3.4. Results

We evaluate classifications using the different sets of features, resulting in multiple scenarios. The scenarios and the results are discussed below.



(a) Classification accuracies



(b) F_1 for file types

Figure 1. Scenario 1: byte-streams extracted from the beginning of files

Scenario 1 - Feature extraction from the beginning of a file: When a stream of contiguous bytes is extracted from the beginning of a file, naturally the 'fingerprint' (for example, header, magic number, specific strings, etc.) of the file is involved in the phases of classification. Yet, classification using the features extracted from the beginning location is not equivalent to signature matching, which requires an exact match of byte sequences.

Fig. 1(a) plots the accuracies obtained using the different feature vectors, for stream-sizes of 8, 16, 32 and 64 bytes. We observe that feature vector formed of unigram frequencies gives the best results, achieving close to 100% accuracy. While the feature vector of distribution characteristics has a small dimension of four, the classification accuracy is high, and close to the best. Both Block Entropies and Markov Entropies perform relatively worse (in comparison to unigram frequencies and distribution characteristics). The performance obtained with Block entropies is similar to that given in [3] (recall, our work and [3] use the same set of Block entropies as feature vector). However with increasing stream-size, the classification accuracies with these feature vectors are seen to increase.

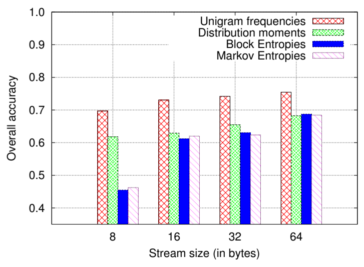
Fig. 1(b) gives the accuracies obtained in classifying each file type, in terms of the F_1 score, when trained and classified using the four different feature vectors. This figure corresponds to the accuracy results for 64 bytes in Fig. 1(a). Entropy features perform worst in classifying gz files.

Scenario 2 - Feature extraction from a fixed location: In this scenario, we skip the very beginning of the file that is highly likely to contain the fingerprints specific to file types. Thus the obfuscation of files by swapping or modifying the header part will not be able to beat the classification system.

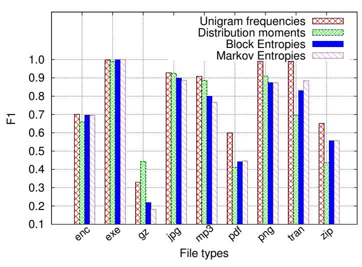
Specifically, the feature extraction phase skips the first 100 bytes of a given file, and extracts from the byte location following the 100^{th} byte. Observe that in this scenario, the location of the stream of consecutive bytes is fixed (for both the training and the testing phases).

Fig. 2(a) plots the classification accuracies. We observe that the accuracies for all the four feature vectors have come down significantly. The feature vector using unigram frequencies gives the best accuracies, ranging from 73% to 76%. The remaining three feature vectors lead to comparable accuracies. The accuracies also increase with increasing stream size.

For the experiment using streams of 64 bytes, the correspond F_1 score is given in Fig. 2(b). Comparing the classification accuracies of *gz*, *pdf* and *zip* file

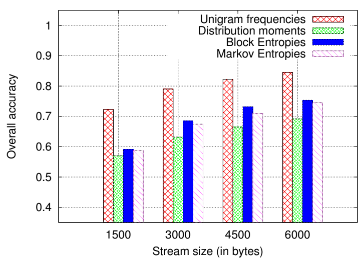


(a) Classification accuracies

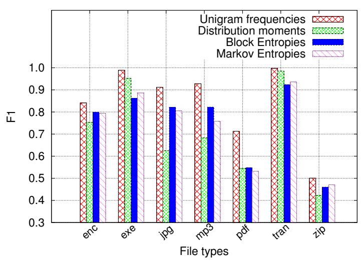


(b) F_1 for file types

Figure 2. Scenario 2: byte-streams extracted after skipping first 100 bytes of a file



(a) Classification accuracies



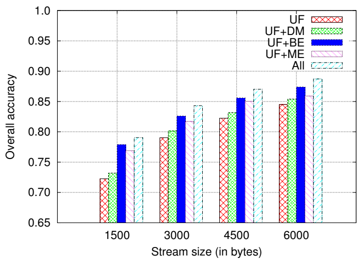
(b) F_1 for file types

Figure 3. Scenario 3: byte-streams extracted from a random location in a file

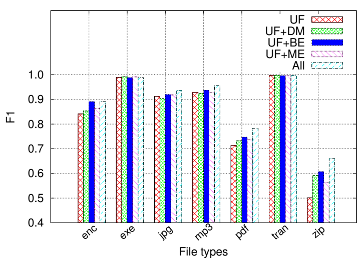
types in figures 1(b) and 2(b), it is clear that the header significantly helps in identifying these file types, as header removal brought down the classification accuracy significantly for all feature vectors.

Scenario 3 - Feature extraction from a random location in a file: Here we extract contiguous stream of bytes from a randomly chosen location. To ensure that there is no bias towards the headers of file types, the first 100 bytes are skipped, and a location in a file is selected uniformly randomly. We observed from the experiments that stream sizes in 100s of bytes (from random locations) do not give good accuracy in classifying file types. Therefore, here we provide the accuracies for stream sizes in 1000s of bytes; more specifically, we extract streams of sizes in multiples of a standard data packet size—1500 bytes. Thus the buffer space required to store packets of a single flow (or connection) can be estimated in number of packets.

As the numbers of files for types *gz* and *png* were less than the minimum requirement, we do not consider these file types in this scenario, and the scenarios following. In Fig. 3(a), we plot the accuracies for stream sizes corresponding to one, two, three and four packets. Unigram frequencies still give the best classification accuracy, and the accuracy increases with the number of contiguous bytes used for classification. However, different from the previous scenarios, entropies give better classification results than distribution characteristics in this scenario. We need to keep in mind that the computational cost of entropies is higher than the other features. From Fig. 3(b) (plotted for stream-sizes of 6000 bytes), we ob-



(a) Classification accuracies



(b) F_1 for file types

Figure 4. Scenario 4: using combinations of feature vectors

serve that for all feature vectors, pdf and zip file types pose challenge in accurate classification.

Scenario 4 - Combinations of feature vectors: In this section, we experiment combinations of feature vectors. As unigram frequencies have given the highest accuracies in all the previous scenarios, we combine feature vector of unigram frequencies with other feature vectors. UF stands for unigram frequencies, DM for distribution moments, BE for Block entropies, and ME for Markov entropies. We also study the effectiveness of using all feature vectors in classification (denoted by 'All' in the corresponding figures).

Fig. 4(a) gives the accuracies. The highest accuracy, of $\approx 89\%$, is attained for random streams of size 6000 bytes, for the combination of all the four feature vectors ('All' in the figure). In general, combining feature vector of entropies with unigram frequencies is seen to increase the classification accuracy. While adding feature vector of block entropies to the feature vector of unigram frequencies always increases the accuracies of classification of file types, this relative improvement decreases with increasing stream size. Recall that the computational cost of block entropies is a quadratic function of the number of bytes. Computing entropies for larger size streams may potentially slow down the traffic, depending on the capacity of the network and traffic characteristics (number of flows per second, number of packets per second).

Fig. 4(b) gives the F_1 score for stream sizes of 6000 bytes. We observe that, in comparison to unigram frequencies, the combination UB+BE feature vectors increases the accuracy of classification of encrypted files from 84% to 89%. The combination of all feature vectors significantly increases the accuracy of zip file classification from 50% to 66%. The combination also increases the classification accuracy of PDF files from 71% to 78%.

3.5. Discussion

To check if similar performance in terms of accuracy can be achieved when all computations are bounded within quadratic time in the number of bytes extracted, we performed one more experiment by removing the only feature that takes more runtime complexity—H_3^M. This modified set of features, all of which can be computed in linear or quadratic time, is equivalent to the features used in 'All' (UF+DF+BE+ME), but with the third feature in the ME feature vector removed. The accuracy and F_1 scores obtained were almost the same as those for the 'All' set of features. From this, one can now decide the features (and their combinations) as well as the stream-size to be used for classification depending on the resource capabilities (computational power and buffer sizes) at one's disposal and the network load. One set of features requires only linear time—UF and DF, and another set requires quadratic time—BE and ME (H_1^M, H_2^M). Increasing stream-size for classification results in increased accuracy, but at the cost of both increased buffering and computational time (where, depending on the features, the increase in computational time will be linear or quadratic).

4. Conclusions

In this paper, we conducted studies to determine the right set of features and stream-size for accurately classifying file types in network traffic. Feature vector consisting of unigram frequencies extracted from beginning of files, and which can be computed in $\mathcal{O}(m)$ time, gives the best and close to 100% accuracy. As not more than 64 bytes are required to achieve this performance, the features can be computed in constant time.

As header of files alone cannot be used a reliable source for classification, we also experimented on streams extracted from random locations (and excluding the header part). Our experiments reveal that though unigram frequencies perform good, combining all feature vectors together indeed performs the best. Depending on the resources available at the point of deployment and the network load, (in terms of link capacities, number of flows per second, number of packets per second, etc.), the stream size and feature vectors to be computed may be varied.

Acknowledgment

This material is based on research work supported in part by the Singapore National Research Foundation under NCR Award No. NRF2014NCR-NCR001-034.

References

[1] S. L. Garfinkel, "Carving Contiguous and Fragmented Files with Fast Object Validation," *Digital Investigation*, vol. 4, pp. 2–12, 2007.

[2] N. Beebe, L. Maddox, L. Liu, and M. Sun, "Sceadan: Using Concatenated N-Gram Vectors for Improved File and Data Type Classification," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 9, pp. 1519–1530, Sept 2013.

[3] A. R. Khakpour and A. X. Liu, "An Information-theoretical Approach to High-speed Flow Nature Identification," *IEEE/ACM Trans. Netw.*, vol. 21, no. 4, pp. 1076–1089, 2013.

[4] M. McDaniel and M. Heydari, "Content based file type detection algorithms," in *Proc. of the 36th Annual Hawaii International Conference on System Sciences*, Jan 2003.

[5] W. J. Li, K. Wang, S. Stolfo, and B. Herzog, "Fileprints: Identifying file types by n-gram analysis," in *Workshop on Information Assurance and security (IAW05), United States Military Academy*, 2005, pp. 64–71.

Subject Index

This page intentionally left blank

Author Index

This page intentionally left blank