REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
25-05-2016	Final Report	24-Jan-2012 - 23-Feb-2016

4. TITLE AND SUBTITLE
Final Report: Resource Efficient Multi-source Authentication with Split-Join One-Way Key Chain for Wireless Ad Hoc Networks

5a. CONTRACT NUMBER
W911NF-12-1-0060

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
206022

6. AUTHORS
Seonho Choi, Kun Sun

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES
Bowie State University
14000 Jericho Park Road

Bowie, MD 20715 -9465

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
60518-CS-REP.19

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for Public Release; Distribution Unlimited

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT
We developed a new scheme utilizing Combined Key Chains (CMC) for multi-source authentication in wireless ad hoc networks. It was shown that the communication overhead is small and constant, and the memory requirement at a verifier node is also minimal. We built API tracing tool for Android-based Mobile Devices which can be effectively used for malware detection and other applications. A new user authentication technique was developed based upon human episodic memory model and a prototype implementation was developed. To detect network abnormal behaviors, a new technique was developed based on the two-level monitoring method. An informal lattice

15. SUBJECT TERMS
security, network, protocol, authentication, multicast, mobile, ad hoc network

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Seonho Choi
UU	UU	UU	UU		19b. TELEPHONE NUMBER 301-860-3960

Report Title

Final Report: Resource Efficient Multi-source Authentication with Split-Join One-Way Key Chain for Wireless Ad Hoc Networks

ABSTRACT

We developed a new scheme utilizing Combined Key Chains (CMC) for multi-source authentication in wireless ad hoc networks. It was shown that the communication overhead is small and constant, and the memory requirement at a verifier node is also minimal. We built API tracing tool for Android-based Mobile Devices which can be effectively used for malware detection and other applications. A new user authentication technique was developed based upon human episodic memory model and a prototype implementation was developed. To detect network abnormal behaviors, a new technique was developed based on the two-level monitoring method. An informal lattice based partitioning approach was developed for IP watermark design and contactless verification. In addition, an FSM-level watermarking solution was developed for protection of sequential IP designs using state encoding and side channel analysis. Also, we produced research results on remotely wiping sensitive data on stolen smartphones and on reliable memory acquisition on smartphones.
An Internet Traffic Prediction Model was developed by applying a signal processing technique. A secure physical memory protection technique was developed, and a new hardware-assisted technique was developed to isolate a computing environment on mobile devices. Also, to incorporate research into our curriculum, we developed a set of project modules on Android security for a senior capstone course for computer science undergraduate program. These project modules were actually adopted and used in our Spring 2013 senior capstone course, and we obtained positive feedback from students.

Enter List of papers submitted or published that acknowledge ARO support from the start of the project to the date of this printing. List the papers, including journal references, in the following categories:

(a) Papers published in peer-reviewed journals (N/A for none)

Received		Paper
05/25/2016	18.00	Bong-Keun Jeong, Seonho Choi, Soo-Yeon Ji, Dong Hyun Jeong. A multi-level intrusion detection method for abnormal network behaviors, Journal of Network and Computer Applications (Elsevier), (02 2016): 9. doi: 10.1016/j.jnca.2015.12.004
08/31/2012	5.00	Seonho Choi, Hyeonsang Eom, Edward Jung. SECURING WIRELESS SENSOR NETWORKS AGAINST BROADCAST SERVICE ATTACKS, International Journal of Computers and Their Applications, (07 2012): 185. doi: 10.2316/Journal. 202.2012.3.202-3335
08/31/2014	11.00	Seonho Choi, Michael Bijou, Kun Sun, Edward Jung. API Tracing Tool for Android-Based Mobile Devices, International Journal of Information and Education Technology, (06 2015): 460. doi: 10.7763/IJIET.2015. V5.550
09/01/2014	13.00	Seonho Choi, Hyeonsang Eom, Edward Jung, Kun Sun. Multi-source broadcast authentication with Combined Key Chains for wireless ad hoc networks, Security and Communication Networks, (08 2014): 0. doi: 10.1002/sec.1072
09/01/2015	17.00	Soo-Yeon Ji, Seonho Choi, Dong Hyun Jeong. Designing an Internet Traffic PredictiveModel by Applying a Signal ProcessingMethod, Journal of Network and Systems Management, (09 2014): 998. doi:
TOTAL:	**5**	

Number of Papers published in peer-reviewed journals:

(b) Papers published in non-peer-reviewed journals (N/A for none)

Received Paper

TOTAL:

Number of Papers published in non peer-reviewed journals:

(c) Presentations

Number of Presentations: 0.00

Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received Paper

TOTAL:

Number of Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received		Paper
08/31/2012	4.00	Edward Jung, Chih-Cheng Hung, Seonho Choi, Ming Yang. An Efficient Locking and Unlocking Method of Sequential Systems, 2012 Research in Applied Computation Symposium . 23-OCT-12, . : ,
08/31/2013	6.00	Seonho Choi, Kun Sun, Hyeonsang Eom. Multi-Source Broadcast Authentication with Combined Key Chains for Wireless Ad Hoc Networks, The 2013 International Conference on Security and Management. 22-JUL-13, . : ,
08/31/2013	3.00	Seonho Choi, Hyeonsang Eom. Real-Time Communication Protocol with Temporally Enhanced Erasure Codes, The 18th International Conference on Parallel and Distributed Processing Techniques and Applications . 16-JUL-12, . : ,
08/31/2013	1.00	Seonho Choi, Kun Sun, Hyeonsang Eom. Resource-Efficient Multi-Source Authentication Utilizing Split-Join One-Way Key Chain, The 11th International Conference on Security and Management . 16-JUL-12, . : ,
09/01/2014	12.00	Edward Jung, Seonho Choi. An FSM-level Watermarking Solution for Protection of Sequential IP Designs using State Encoding and Side Channel Analysis, 2014 Research in Adaptive and Convergent Systems. 05-OCT-14, . : ,
09/01/2014	14.00	Brian Choi, Kun Sun, Seonho Choi. Cloud-Based User Authentication with Geo-Temporal Queries on Smartphones, The Second International Workshop on Security in Cloud Computing. 03-JUN-14, . : ,
09/01/2015	16.00	Seonho Choi, Kun Sun, Edward Jung. Optimized Design of Geo-Temporal Query Based User Authentication Scheme for Smartphones, Seoul International Conference on Applied Science and Engineering . 27-JUN-15, . : ,
09/01/2015	15.00	Seonho Choi, Edward Jung. Identification of IP control units by state encoding, IEEE Computer Society Annual Symposium on VLSI. , . : ,
TOTAL:	**8**	

Number of Peer-Reviewed Conference Proceeding publications (other than abstracts):

(d) Manuscripts

Received		Paper
08/31/2013	8.00	Edward Jung, Chih-Cheng Hung, Ming Yang, Seonho Choi. An Locking and Unlocking Primitive Function of FSM-modeled Sequential Systems Based on Extracting Logical Property, Information (01 2013)
08/31/2013	9.00	Seonho Choi, Kun Sun, Hyeonsang Eom. Multi-Source Broadcast Authentication with Combined Key Chains for Wireless Ad Hoc Networks, Security and Communication Networks (08 2013)
TOTAL:		**2**

Number of Manuscripts:

Books

Received		Book
TOTAL:		

Received		Book Chapter
08/31/2013	7.00	Seonho Choi, Kun Sun, Hyeonsang Eom. Resource-Efficient Multi-Source Authentication Utilizing Split-Join One-Way Key Chain, Waltham, MA 02451, U.S.A. : Elsevier, (10 2013)
TOTAL:		**1**

Patents Submitted

Patents Awarded

Awards

Graduate Students

NAME	PERCENT_SUPPORTED	Discipline
Wenhao Chen	0.10	
FTE Equivalent:	**0.10**	
Total Number:	**1**	

Names of Post Doctorates

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Names of Faculty Supported

NAME	PERCENT_SUPPORTED	National Academy Member
Seonho Choi	0.29	
Bo Yang	0.04	
FTE Equivalent:	**0.33**	
Total Number:	**2**	

Names of Under Graduate students supported

NAME	PERCENT_SUPPORTED	Discipline
Michael Bijou	0.32	
Cengiz Beslen	0.17	
Albert Cheng	0.31	
Dung Bui	0.07	
FTE Equivalent:	**0.87**	
Total Number:	**4**	

Student Metrics

This section only applies to graduating undergraduates supported by this agreement in this reporting period

The number of undergraduates funded by this agreement who graduated during this period:	3.00
The number of undergraduates funded by this agreement who graduated during this period with a degree in science, mathematics, engineering, or technology fields:.....	3.00
The number of undergraduates funded by your agreement who graduated during this period and will continue to pursue a graduate or Ph.D. degree in science, mathematics, engineering, or technology fields:.....	1.00
Number of graduating undergraduates who achieved a 3.5 GPA to 4.0 (4.0 max scale):.....	1.00
Number of graduating undergraduates funded by a DoD funded Center of Excellence grant for Education, Research and Engineering:.....	0.00
The number of undergraduates funded by your agreement who graduated during this period and intend to work for the Department of Defense	0.00
The number of undergraduates funded by your agreement who graduated during this period and will receive scholarships or fellowships for further studies in science, mathematics, engineering or technology fields:	0.00

Names of Personnel receiving masters degrees

NAME
Wenhao Chen
Total Number: 1

Names of personnel receiving PHDs

NAME
Total Number:

Names of other research staff

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Sub Contractors (DD882)

1 a. George Mason University 1 b. 4400 University Drive, MS 4C6

Fairfax VA 220304422

Sub Contractor Numbers (c):
Patent Clause Number (d-1):
Patent Date (d-2):
Work Description (e): Conduct research on the security of authentication protocols, security on mobile devices.
Sub Contract Award Date (f-1): 2/24/12 12:00AM
Sub Contract Est Completion Date(f-2): 8/20/14 12:00AM

---

1 a. George Mason University 1 b. 4400 University Drive, MSN 4C6

Fairfax VA 220304422

Sub Contractor Numbers (c):
Patent Clause Number (d-1):
Patent Date (d-2):
Work Description (e): Conduct research on the security of authentication protocols, security on mobile devices.
Sub Contract Award Date (f-1): 2/24/12 12:00AM
Sub Contract Est Completion Date(f-2): 8/20/14 12:00AM

---

1 a. College of William and Mary 1 b. P.O. Box 8795

Williamsburg VA 231878795

Sub Contractor Numbers (c):
Patent Clause Number (d-1):
Patent Date (d-2):
Work Description (e): Conduct research on the security of authentication protocols, security on mobile devices.
Sub Contract Award Date (f-1): 1/15/15 12:00AM
Sub Contract Est Completion Date(f-2): 8/31/15 12:00AM

---

1 a. College of William and Mary 1 b. P.O. Box 8795

Williamsburg VA 231878795

Sub Contractor Numbers (c):
Patent Clause Number (d-1):
Patent Date (d-2):
Work Description (e): Conduct research on the security of authentication protocols, security on mobile devices.
Sub Contract Award Date (f-1): 1/15/15 12:00AM
Sub Contract Est Completion Date(f-2): 8/31/15 12:00AM

Inventions (DD882)

Scientific Progress

See Attachment

Technology Transfer

Final Report

Grant No: W911NF1210060
Program Manager: Dr. Cliff X. Wang

Resource Efficient Multi-source Authentication with Split-Join One-Way Key Chain for Wireless Ad Hoc Networks

Principal Investigator:

Seonho Choi, Bowie State University

Computer Science Department

14000 Jericho Park Rd.

Bowie, MD 20715

schoi@bowiestate.edu

Office) 301-860-3967

Co-Principal Investigator:

Kun Sun, College of William & Mary

Department of Computer Science

ksun@wm.edu

Table of Contents

1. General Information

a) Contract number: W911NF1210060

b) Period of performance being reported: Jan. 24 2012– Feb. 23, 2016

c) Principal Investigator: Seonho Choi

d) PI of the Subaward: Kun Sun

e) Contracting Officer Representative: Dr. Cliff X. Wang
DEPARTMENT OF THE ARMY
US ARMY RESEARCH, DEVELOPMENT AND ENGINEERING COMMAND
ARMY RESEARCH OFFICE
Email: cliff.x.wang.civ@mail.mil
Tel: (919) 549-4207

2. Optimized Design of Geo-Temporal Query Based User Authentication Scheme for Smartphones

INTRODUCTION

Mobile devices such as smart phones have been widely used for processing monetary transactions and accessing sensitive data. The mobile device should be protected by an authentication mechanism to prevent an unfriendly user who possesses the device even for a short time from stealing valuable sensitive data. A number of authentication techniques have been proposed and developed for mobile devices. Some of the most commonly used techniques include conventional password based scheme, keystroke dynamics based technique, pattern-based scheme, physiological biometric based techniques, etc.

As more users are utilizing mobile devices for carrying out various tasks including monetary transactions, it is becoming more and more critical to protect them against malicious accesses. A lot of user authentication techniques have been proposed and developed for this purpose including physiological or behavioral biometric techniques, pattern-based scheme, conventional password based scheme, etc.

However, most of the previously developed user authentication schemes suffer from various limitations including the low resistance against some attack types such as shoulder surfing attacks, low memorability of the required information to be used for authentication such as passwords or phrases, high false negative and/or false positive rates, requirements for cumbersome training steps needed for some biometric-based schemes, or dependence on special hardware units such as fingerprint readers.

We proposed a new authentication idea utilizing a series of binary geo-temporal queries based upon user's episodic memory [16]. With the advancement of the smartphone technology it became very feasible to acquire and maintain some aspects of the user's episodic memory such as locations and times where and when the user has been. This type of personal episodic memory information may be stored along with their temporal information, and a series of geo-temporal queries may be created and asked to a user. Even though various types of geo-temporal queries may be generated we chose to focus on utilizing binary geo-temporal queries for its simplicity and applicability. A user is given a series of binary queries and he/she needs to choose true or false on each query.

Our geo-temporal query based scheme has the following characteristics [16]:

- Usability: users can easily answer a number of true/false queries for authentication purpose.
- Attack resistance: it can protect against various attack types including the shoulder surfing attack.
- Quantification and Regulation of security risks: it should allow us to (probabilistically) quantify the security risks such as the authenticity probability given user's response, expected false positive and false negative rates, etc.
- Implementation feasibility: our technique should be implementable on most mobile platforms without any special HW requirements.

However, personal-history based techniques have an inevitable drawback of privacy disclosure problem. For instance, in geo-temporal query technique queries include location and temporal information, and an adversary may obtain correct history information once the true queries – whose answers are true – are identified out of the false queries – whose answers are false – among the queries observed during an attempted authentication process. To minimize the privacy disclosure level we propose two pre-termination techniques for the query process, *success pre-termination* and *failure pre-termination*.

However, we need to formulate its design process in such a way that various security and privacy requirements may be specified and several parameters may be determined. In this report, this design process is formulated as an optimization problem and the solution approach is derived on how the optimization problem may be solved at system design time. By using our formulation and algorithm, design time decision may be made on such parameters such as number of maximum queries to be given out, the minimum number of hits needed for a successful authentication, the maximum number of misses that will automatically fail the authentication, the true (or false) query generation probability, etc. On Android 4.3, we built a prototype App, which provides user authentication to unlock a Samsung Galaxy S4 smartphone with low power consumption.

Threat Model and Assumptions

We consider someone as an attacker who has physical access to a smart phone at least for a limited time. Even though an attacker has physical access to a phone, he/she may get access to the resources in the smart phone only after successful authentication. An attacker is assumed to have knowledge on various system parameter values. We assume the mobile devices are equipped with GPS to record the location information. Most smart phones have GPS included. GPS typically cannot function indoors; however, we won't require such fine-grained geo-location information.

Geo-Temporal Authentication Framework

Overview

Figure 1 shows the geo-temporal authentication process. User's location history information is obtained by using the GPS component. The location coordinates are periodically sampled and stored along with the sampling time information. Once the coordinate values are obtained from the GPS, the coordinates and sampling time will be stored into the Location History Base. When a user attempts to unlock the smart phone, the Query Generator and Processor component will make use of the Location History Base to create a series of geo-temporal queries to be answered by the user. Given a geo-temporal query, if the user's answer is correct, then it is said that a ***hit*** occurred. Otherwise, it is said that a ***miss*** occurred. Multiple queries may be challenged to allow for some operation mistakes or vague memory.

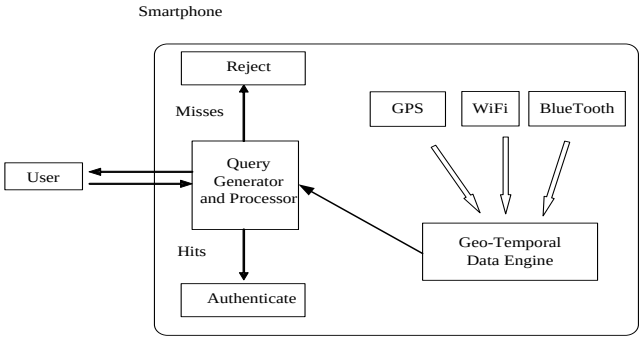


Figure 1. Geo-Temporal Authentication Framework

Geo-Temporal Data Engine

For our scheme to work we need to create and regularly update a geo-temporal database containing the location history information sampled with time information. The necessary entries in each record are sampling time, longitude, latitude, address (obtained by using the Reverse Geocoding), and duration of stay.

There are tradeoffs between GPS (or other sensors) sampling period and energy consumption. Using smaller sampling period will produce more accurate traces of user locations, while consuming more energy. Less accuracy may be achieved with bigger sampling period, but less power will be spent.

Authentication Query Generator and Processor

In this work, we focus on using the binary geo-temporal query, whose answer should be either true or false, and which contains both the location and temporal information in the query itself.

Scaled Geo-Temporal Granularities

There is a correlation between human memorability and location granularity. The finer the location granularity is, the more difficult to remember. If a user is given an address containing street number and name, he/she would have a more difficult time in remembering compared to the case when he/she is challenged only with city name. For example, it would be more difficult to answer "Did you visit 1234 Main Street, New York, NY last May?" than to answer "Did you visit New York, NY last May?"

Also, the finer the temporal granularity is, the more difficult to recall. For example, consider the following two queries: "Did you visit New York, NY, last May?" and "Did you visit New York, NY, between May 12 and 14?" In this case, the latter query would be more difficult to answer.

The relationship between location and temporal granularities may be represented by different regions in the scaled X-Y graph as is shown in Figure 2. Out of the possible regions in the graph, we will focus on those along the

correlation line. These regions will have the similar level of location and temporal granularities. "Did you visit Canada in 2009?" is an example query in region 5, and "Did you visit Quebec, Canada, in May 2009?" in region 4. As is noted in the figure, more privacy will be disclosed by the low-numbered regions. But, more accurate authentication may be provided.

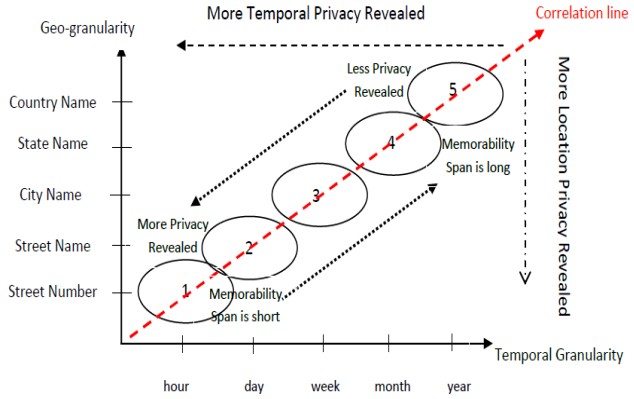


Figure 2: Scaled Geo-Temporal Granularities. Correlation line is used in creating the queries. Depending upon the temporal remoteness of the location sample used in the query, one of the regions in the Correlation Line is chosen in generating the query (Table 1).

However, human memory tends to fade away as time goes by. They can remember recent events more accurately compared to the ones in remote past. For example, the places visited yesterday may be remembered more clearly along with temporal information than the ones visited a month ago. This implies that we may utilize the queries in the low-numbered regions only from the recent location history, and from the remote location history those in the high-numbered regions may be used in the authentication process. In Figure 2 the low-numbered regions are characterized with short memorability spans and the high-numbered ones with long memorability spans. The ***temporal remoteness*** of the location samples needs to be considered in generating the queries as well as the location and temporal granularities.

When queries are generated in our approach, the temporal and location granularities of queries will be adjusted to reflect this correlation with the remoteness of the location history. We name this technique as Scaled Geo-Temporal Granularities. The following shows the example types of queries that are generated according to this principle:

- Did you visit 1234 35-th Street, New York, NY at 3:15pm today? (Region 1 in Figure 2)
- Did you visit 35-th Street, New York, NY the day before yesterday? (Region 2 in Figure 2)
- Did you visit New York, NY 3 weeks ago? (Region 3 in Fgiure 2)
- Did you visit NY state 3 last May? (Region 4 in Figure 2)
- Did you visit Canada in 2010? (Region 5 in Figure 2)

False Query Generator

False queries need to be obtained and presented to a user along with the true queries. Simple approach may be to generate totally random location and time. However, the locations used in the false queries may not be too far from the locations that have been visited by a user. For example, if an adversary acquired a phone in New York City and asked whether the user was in California that morning, it would be easy to guess the answer. For this reason we adopted to base the false query generation process upon the information stored in the Location Base, and introduce some randomness to create false queries. The basic idea is to choose a location sample from the Location Base as is done in generating the true query, and add/subtract random numbers to/from the coordinates and sampling time of the chosen sample.

Authentication Scheme

Multiple queries may be challenged to allow for some operation mistakes or vague memory. There is a constraint on the number of queries that can be given to a user. We will denote the maximum number of queries that can be given as *M*. A series of geo-temporal queries will be given until one of the following conditions met:

- *Authentication success*: enough number of hits made after *M* queries are given.
- *Authentication failure*: enough number of hits not made even after *M* queries are given.

We propose two pre-termination techniques to reduce the number of disclosed true queries: ***success pre-termination*** and ***failure pre-termination***. First, if the number of hits reaches a pre-determined threshold *H*, then the query process terminates early even before *M* queries are given out. Second, even before all of *M* queries are presented to a user, the query process may be terminated if there is no possibility that the constraints will be met even if the remaining queries are answered correct. **Figure 2** shows the basic authentication algorithm. It is assumed that $H > M-H$, which must be true for the most of the practical parameter values. This means $H > M/2$.

```
for k=1 to M do
Ask k-th query to the user.
if correct answer
      NumCorrectAnswers++
      If NumCorrectAnswers >= H
            Success pre-termination;
else
     NumIncorectAnswers++
     If NumIncorrectAnswers > M-H
            Failure pre-termination;
end do
```

Figure 2. Basic Authentication Algorithm

Suppose that *k* queries have been presented to a user, and let *h* denote the number of queries answered correct (*hits*) by the user. The number of remaining queries is *M-k*, and the maximum number of hits that may result out of *M* queries (*k* already answered and *M-k* to be answered) is $h+(M-k)$ under the assumption that all of the *M-k* remaining queries are answered correctly. If $h+(M-k) < H$, then there is no hope that *H* queries will be answered correct even after the remaining queries are given and answered. This corresponds to the failure pre-termination case.

Table 3. Terms Definition

M	Maximum number of queries to the user.
H	The number of correct answers (hit) a user needs to provide before he/she can be authenticated.
M_{max}	An upper bound on *M*, i.e., $M \leq M_{max}$. This is a design time parameter.
P_{true}	Probability that a query would be a true query.
P_{false}	Probability that a query would be a false query.
P_{auth}	Target threshold probability for a user being an authentic user
P_{FP}	Target threshold probability for false positive (valid user is not authenticated).
P_{FN}	Target threshold probability for false negative (invalid user is authenticated).
A	Event that a user is an authentic user.
C	event that a user answers correctly.
N_k	Random variable denoting the number of hits made out of *k* queries by the user
p_f	Probability that an authentic user makes a mistake in answering a query.

Security Constraints

We will design the authentication scheme in which a guarantee may be given in term of the following security constraints:

- P_{auth} : When an authentication is granted, a probability that a user is authentic should be greater than or equal to this value.
- P_{FP}: The probability that a valid user is not authenticated should be less than or equal to this.
- P_{FN}: The probability that an invalid user is authenticated should be less than or equal to this.

Given these parameter values along with M_{max}, we need to determine the values of M, H, P_{true}, and P_{false}, that will be used in the actual query generation and challenge process.

Thus, we obtain the following optimization problem.

Optimization Problem 1*: Find out values of M, H, P_{true} and P_{false} ($1 \le H \le M$) satisfying the given P_{Auth}, P_{FP}, P_{FN} and M_{max} conditions.*

It is possible that there may be more than one candidate values for H. In this case the minimum value will be chosen as H. Here, we are supposed to find minimum values for M (and H) satisfying the security constraints. $p_f = P(C^c|A)$ is the probability that an authentic user makes a mistake in answering a query. This probability may be experimentally obtained. This may vary according to different factors such as different persons (with different memory capability), different scales for Geo-Temporal Query Granularity settings. In this work, we start with some initial probability (e.g., 0.1), and dynamically adjust this value based upon the user's responses whenever a user authentication succeeds. We use Exponentially Weighted Moving Average algorithm to update this probability.
$P(A)$ is the probability that a user is authentic. It is known that 18.13% of cell phones are lost or stolen each year. So, we set this to be $P(A)=1-0.1813=0.8187$. This leads to $P(A^c)=0.1813$.

Suppose that M queries are presented to a user, and let h denote the number of queries actually answered correct by the user. Then, $0 \le h \le M$ holds. The following probabilities may be obtained.

$$P(A \mid N_M \geq h) = \frac{P(N_M \geq h \mid A) \cdot P(A)}{P(N_M \geq h \mid A) \cdot P(A) + P(N_M \geq h \mid A^c) \cdot P(A^c)} \quad (1)$$

$$P(N_M \geq h \mid A) = \sum_{i=h}^{M} \binom{M}{i} (1 - p_f)^i p_f{}^{M-i} \quad (2)$$

$$P(N_M \geq h \mid A^c) = \sum_{i=h}^{M} \binom{M}{i} (1 - 2P_{true} + 2P_{true}{}^2)^i (-2P_{true}{}^2 + 2P_{true})^{M-i} \quad (3)$$

The first probability is obtained by applying the Bayes Theorem. $P(A|N_M \geq h)$ represents the probability that a user is an authentic user when the user provides h or more correct answers to M queries. Given $P(A)$ and p_f values, this is a function of M and h.

We assume the true queries whose answer should be "true" are randomly generated with a probability P_{true}, and false queries with a probability of $P_{false} = 1 - P_{true}$. These probabilities can be known to an adversary. Given a query, the probability that an adversary provides a correct answer is $(P_{true})^2 + (P_{false})^2 = 1 - 2P_{true} + 2(P_{true})^2$, and the probability for providing an incorrect answer is $1 - (P_{true})^2 - (P_{false})^2 = 2P_{true} - 2(P_{true})^2$.

The number of queries to be asked to a user varies in $[0, M]$. P_{FP} is the target threshold probability for false positives which occur when a valid user is not authenticated. If this is 0.01, then it should be guaranteed that at most 1% of the cases a valid user is rejected (e.g., by user mistakes). P_{FN} is the target threshold probability for false negative when an invalid user is authenticated. If this is 0.005, then it should be guaranteed that at most 0.5% of the cases will result in authentication even though the user is not authentic. This may happen when an adversary launches a random guessing attack. M_{max} is an upper bound on M, and it is given at a design time. This condition will be used as one of the constraints in the optimization process.

For an authentication to be granted for the user with (M, h) values – i.e., $N_M \geq h$ to be used as an authentication criteria, the following should hold.

- $P(N_M \geq h \mid A^c) \leq P_{FN}$ (4)
- $P(N_M < h \mid A) = 1 - P(N_M \geq h \mid A) \leq P_{FP}$ (5)
- $P(A \mid N_M \geq h) \geq P_{Auth}$ (6)

We may choose the minimum value of M for which at least one integer h ($1 \le h \le M$) exists satisfying the above conditions. Then, we may denote them as M and H, respectively. Again, for more than one candidate values for H we select the minimum value.

Privacy Metrics

In the authentication scheme, a series of queries are presented to a user based upon the location history file; however, this may endanger user privacy by revealing locations frequently visited or other location-related information. To overcome this limitation, we reformulate our optimization problem as follows:

Optimization Problem 2*: Find out values of M, H, P_{true} and P_{false} ($1 \le H \le M$) satisfying the given P_{Auth}, P_{FP}, P_{FN} and M_{max} conditions and minimizing the number of true queries exposed. It is possible that there may be more than one candidate values for H. In this case the minimum value will be chosen as H.*

Suppose that an adversary tries to obtain as much location history information from the smart phone as possible by launching a guess and check attack. Also, let's assume that k queries have been given to the adversary so far. Then there are three possibilities:

- Successful pre-termination condition: $k \ge H$, H correct answers and $k-H$ incorrect answers.
- Failure pre-termination condition: $k \ge M-H+1$, and there were $M-H+1$ incorrect answers and $k-M+H-1$ correct answers. $M-H$ represents a maximum number of queries that may be missed before a failure pre-termination occurs.
- Query continuation condition: the other cases except for the above two conditions.

We may divide the cases into the following three cases depending upon the values of k.

- $H \le k$: Success pre-termination and failure pre-termination are mutually exclusive events. That is, if both conditions hold at the same time, then we have $k-H=M-H+1$, which yields to $k=M+1$. However, this is a contradiction.
- $M-H+1 \le k < H$: Failure pre-termination condition and the query continuation are mutually exclusive events. Again, failure pre-termination occurs when there are $M-H+1$ incorrect answers and $k-M+H-1$ correct answers.
- $k \le M-H$: Only query continuation is the possible state.

Let x_i $(1 \le i \le k)$ denote a random binary variable which has a value of 0 when a user answers wrong for the i-th query, and 1 when a user answers correctly. And, let $y_k = x_1 + x_2 + \ldots + x_k$ denote a sum of x_1 through x_k, which represents the number of correctly answered queries out of k queries. Then, we may trace the values of y_k over the index values from 1 through k for various cases including success pre-termination, success termination, failure termination as is shown in Figure4 Through Figure 9. In these figures, it is assumed that $M=7$ and $H=5$

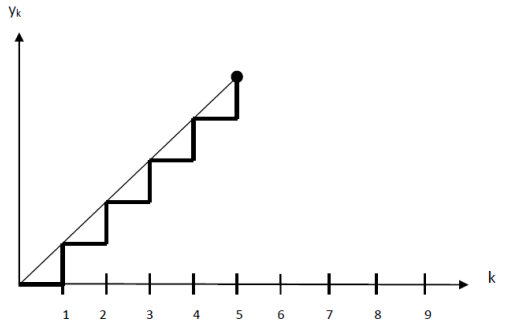


Figure 4: Success pre-termination at k=5 when the first 5 queries are answered correctly.

In Figure 4 success pre-termination case is shown. In the graph, at each k index value there are two possible cases, either y_k jumps up by one unit or stays the same as before. In this case the user answers correctly to the first 5 queries without any misses, and the query process pre-terminates only after 5 queries are given. The process terminates when $y_k=5$ is reached, which occurs for $k=5$ in this example.

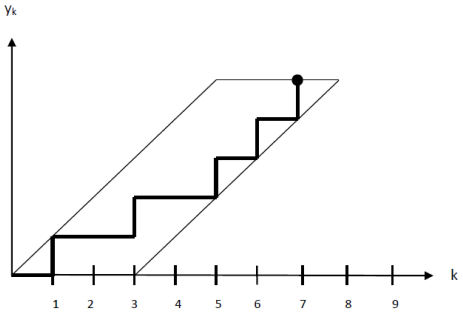


Figure 5: Success termination at k=7

Figure 5 shows a success termination case where all of the *M* queries are given out without pre-termination. In this scenario 5 queries are answered correctly, but with 2 misses in between. Because of the misses, the termination occurs at $k=7$ where $y_k=5$ is reached.

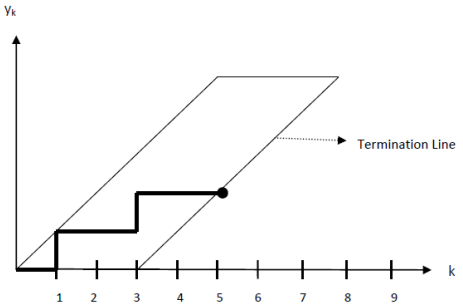


Figure 6: Failure pre-termination at *k=5*

Figure 6 shows a failure pre-termination case where the query process is terminated after giving out 5 queries. 2 queries were answered correctly and 3 queries were answered incorrectly, which yields to the condition where an authentication is impossible even if the user answers the remaining 2 queries. Note that the failure termination occurs when the trace line hits the Termination Line.

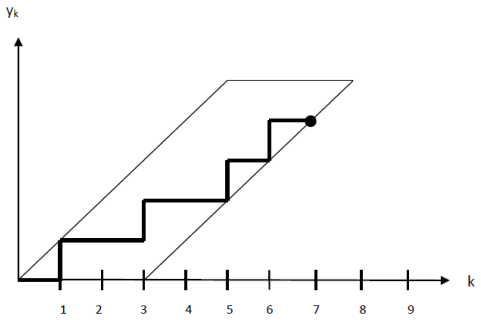


Figure 7: Failure termination at *k=7* when 4 queries are answered correctly and 3 queries are answered incorrectly.

An example scenario for failure termination is given in Figure 7. The last query q_7 must have been answered wrong. Otherwise, a failure pre-termination should have occurred at an earlier query response.

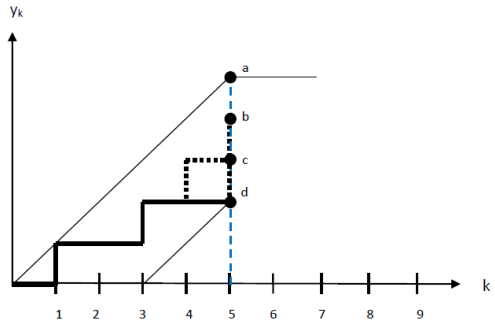


Figure 8: Possible values of trace line at *k=5*.

Different points that a trace line may end at k=5 is shown in Figure 8. In this specific scenario there are 4 such points denoted as *a*, *b*, *c*, and *d*. The possible y_5 values at *a*, *b*, *c* and *d* are 2, 3, 4 and 5 where $y_5=x_1+x_2+x_3+x_4+x_5$. Point *a* corresponds to a success pre-termination, and point *d* corresponds to a failure pre-termination. At points *b* and *c* the query process will continue without termination. It is again assumed that *H=5* and *M=7*.

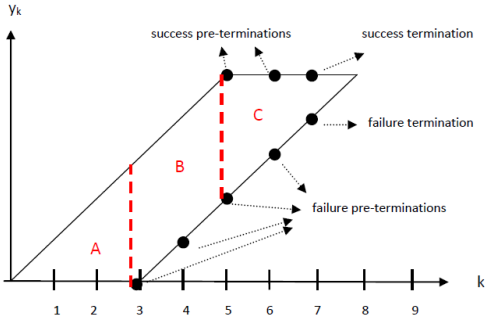


Figure 9: Possible termination points.

In Figure 9 all the possible termination points are shown for *H=5* and *M=7*. We may divide the entire area into three subareas, *A, B, and C*. Area *A* corresponds to the condition *(k ≤ M-H)*, and no termination is possible. Area B corresponds to *(M-H+1 ≤ k <H)*, and either continuation or failure pre-terminations are possible. Area *C* satisfies the condition *(H ≤ k)*, and continuation, failure pre-termination, failure termination, success pre-termination or success termination is possible.

We want to find the expected number of disclosed true queries as a privacy metrics. As is shown in Figure 9, the termination points in the (k, y_k) graph may belong to three different regions denoted as *A*, *B*, and *C*. We need to find the probabilities of the query process terminating at these points.

Probabilities in Area C ($H \leq k$)

When a *k*-th query is given and answered where $H \leq k$, the possible range of y_k values are *[k-(M-H)-1, H]*. For example, this corresponds to the point *d* in Figure 8. The minimum value *k-M+H-1* corresponds to a failure pre-termination and occurs when exactly *M-H* queries are answered wrong and *k-1-M+H* queries are answered correct to the queries, q_1, q_2, ... and q_{k-1}, and the last query q_k is answered wrong. If more than *M-H* queries are answered wrong to the queries q_1, q_2, ... *and* q_{k-1}, then a failure pre-termination must have occurred before the *k*-th query was given. The probability of this failure pre-termination happening at *k* is given as:

$$P(y_k = k - M + H - 1) = \binom{k-1}{M-H}\left(-2P_{true}{}^2 + 2P_{true}\right)^{M-H+1}\left(1 - 2P_{true} + 2P_{true}{}^2\right)^{k-1-M+H} \quad (7)$$

The probability of y_k=*k-(M-H)* – e.g., point *c* in Figure 8 – may be obtained by considering the cases where *M-H* queries are answered wrong and *k-M+H* queries are answered correct to the *k* queries q_1, q_2, ... and q_k. This is given as

$$P(y_k = k - M + H) = \binom{k}{k-M+H}\left(-2P_{true}{}^2 + 2P_{true}\right)^{M-H}\left(1 - 2P_{true} + 2P_{true}{}^2\right)^{k-M+H} \quad (8)$$

By applying the similar argument, the probability *of* y_k=*k-(M-H)+n*, where $0 \leq n \leq M-k$, may be obtained by considering the cases where *M-H-n* queries are answered wrong and *k-M+H+n* queries are answered correct to the *k* queries q_1, q_2, ... and q_k. This is given as

$$P(y_k = k - M + H + n) = \binom{k}{M-H-n}\left(-2P_{true}{}^2 + 2P_{true}\right)^{M-H-n}\left(1 - 2P_{true} + 2P_{true}{}^2\right)^{k-M+H+n} \quad (9)$$

When $n=M-k$, a success pre-termination occurs for $k<M$ and a success termination occurs for $k=M$. Hence, the probability of success termination when the k-th query is given and answered is

$$P(y_k = H) = \binom{k}{H}\left(-2P_{true}{}^2 + 2P_{true}\right)^{k-H}\left(1 - 2P_{true} + 2P_{true}{}^2\right)^{H} \quad (10)$$

Probabilities in Area B ($M-H+1 \leq k < H$)

In this case only failure pre-terminations are possible. When a k-th query is given and answered, the possible range of y_k values are $[k-(M-H)-1, k]$. The minimum value $k-M+H-1$ corresponds to a failure pre-termination and occurs when exactly $M-H$ queries are answered wrong and $k-1-M+H$ queries are answered correct to queries, q_1, q_2, ... and q_{k-1}, and the last query q_k is answered wrong. If more than $M-H$ queries are answered wrong to queries q_1, q_2, ... and q_{k-1}, then a failure pre-termination must have occurred before the k-th query was given. The probability of this happening is given as:

$$P(y_k = k - M + H - 1) = \binom{k-1}{M-H}\left(-2P_{true}{}^2 + 2P_{true}\right)^{M-H+1}\left(1 - 2P_{true} + 2P_{true}{}^2\right)^{k-1-M+H} \quad (11)$$

Probabilities in Area A ($k \leq M-H$)

In this case there is no possibility of terminations. The query process will only continue in this area.

Privacy Metric: Expected Number of True Queries Exposed

We will use the expected number of true queries that may be exposed to an adversary when he/she attempts to be authenticated. It is assumed that an adversary has knowledge on P_{true} and P_{false} values and employs a random guessing strategy in answering the queries. That is, he/she will try to answer a query with "true" with the probability of P_{true}. Then, the expected number of true queries that are given to the adversary may be found as follows from (7), (10) and (11):

$$E = \sum_{k=H}^{M} P_{true} \cdot k \cdot P(y_k = k - M + H - 1) + \sum_{k=H}^{M} P_{true} \cdot k \cdot P(y_k = H) + \sum_{k=M-H+1}^{H-1} P_{true} \cdot k \cdot P(y_k = k - M + H - 1)$$

$$E = \sum_{k=H}^{M} P_{true} \cdot k \cdot \binom{k-1}{M-H}\left(-2P_{true}{}^2 + 2P_{true}\right)^{M-H+1}\left(1 - 2P_{true} + 2P_{true}{}^2\right)^{k-1-M+H} + \sum_{k=H}^{M} P_{true} \cdot k \cdot \binom{k}{H}\left(-2P_{true}{}^2 + 2P_{true}\right)^{k-H}\left(1 - 2P_{true} + 2P_{true}{}^2\right)^{H} + \sum_{k=M-H+1}^{H-1} P_{true} \cdot k \cdot \binom{k-1}{M-H}\left(-2P_{true}{}^2 + 2P_{true}\right)^{M-H+1}\left(1 - 2P_{true} + 2P_{true}{}^2\right)^{k-1-M+H} \quad (12)$$

This E represents the expected number of true queries exposed to an adversary. This formula is used in our Optimization Problem 2 definition as a metric for privacy disclosure.

Solution Approach for the Optimization Problem

Our Optimization Problem 2 may be refined by incorporating the constraints on P_{FP}, P_{FN} and P_{auth}, and by having an optimization objective function E – given in (12) – to be minimized. One simple approach would be to exhaustively search for optimal solution by trying out different combinations of (P_{true}, M, H) values. Figure 10 shows this algorithm. P_{true} is a continuous variable and needs to be quantized with a gap of Δ. In our example Δ was chosen to be 0.001.

```
read P_auth, P_FP, P_FN, p_f

for P_true=0 to 1.0 step Δ do
  for M=1 to M_max do
    for H=M/2+1 to M do
        check inequalities (4), (5), and (6)
        if any of the inequalities doesn't hold
        then
           continue
        calculate E value using (12)
        if E is a new minimum
        then
           record this as a new minimum along with (P_true, M, H)
      end do
    end do
end do
```

output *minimum E value along with corresponding* (P_{true}, *M, H)*

Figure 10: Solving the Optimization Problem by employing an iterative exhaustive search.

Example Optimization

Given the following parameter values, we applied an optimization algorithm to find (P_{true}, M, H) values satisfying the probability constraints and minimizing the expected number of exposed true queries to a potential adversary. The input parameters are $P_{auth}= 0.9$, $P_{FP} = 0.05$, $P_{FN} = 0.05$, $p_f = 0.05$.

The optimal (M,H) was found to be $(11,9)$ when $P_{true}=0.378$ with an expected number of disclosed true queries equal to 2.391. Please note that, when $P_{true}=0.5$, the E value would be equal to 2.986.

The graph in Figure 11 shows the points on (M,H) plane meeting the probability constraints and the z axis shows the expected number of true queries to be disclosed for each (M,H) combination.

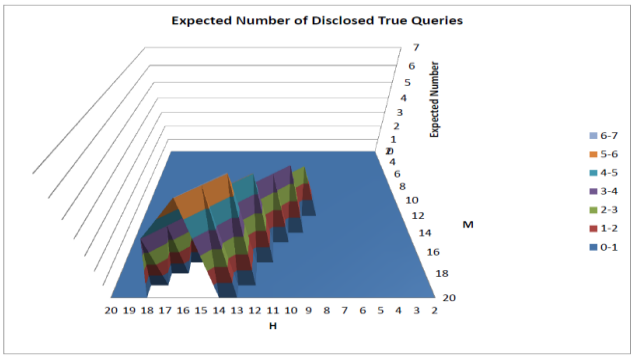


Figure 11: Expected number of disclosed true queries for different *M* and *H* for $P_{true}=0.378$.

Implementation

We constructed a prototype app on Android 4.3, and ran it for an hour to measure power consumption on Samsung Galaxy S4 smartphone. As was shown in previous research [15], invoking GPS unit and obtaining coordinates consumes a significant amount of power. Hence, we decided to invoke the GPS unit periodically with a fixed sampling period. Table 4 was obtained only by running the tracking component of the app, which will use most of the energy due to GPS samplings.

Table 4. Power consumption with different sampling periods

Sampling period	Power consumption (Wh)	Battery lifetime
10 seconds	3.002×10^{-1}	1.107 days
1 minute	5.004×10^{-2}	6.64 days
5 minutes	1.001×10^{-2}	33.2 days
15 minutes	3.337×10^{-3}	99.6 days

Figure 12 shows an example screenshot with a geo-temporal query. User is supposed to choose either the "true" or "false" button. This process is repeated between m and M times depending upon the hit & miss patterns of the user responses. Note that only one query is given each time since the termination may occur at any time and we want to minimize the number of true queries exposed.

Discussion & Future Works

Through experimentations, we need to validate the correlations used in generating the geo-temporal queries between temporal remoteness of the sample, location granularity and temporal granularity. Or, more adequate correlation models may be found through such process. The key issue is how we design the memorability model and validate it. Also, we may need to elaborate and corroborate the parameter selection for the false query generation process.

Figure 12. Example screenshot with a geo-temporal query.

To increase the memorability, we may develop several schemes such as a pre-hinting technique to remind users of the locations he/she is visiting or recently visited as a preparation for future authentication queries.

As was shown in our and previous works, GPS consumes significant energy and it would be vital to activate the unit only when necessary. To save energy for location sampling we may invoke the unit only when the user is in move by utilizing other sensors such as accelerometer sensor.

Related Works

Besides using a pre-agreed set of shared secrets, we can authenticate users by challenging them with questions dynamically generated from a wider base of personal resources, such as calendar [7], email [6], or twitter [8]. Nosseir et al. [7] propose to query the users based on their personal history and develop a prototype using automatic generation of questions from electronic Calendar information. Nishigaki and Koike [6] propose a user authentication using user's email history to generate the questions. However, it has a serious problem on leaking privacy, since the impersonators are able to read the mail content when they are trying to log on as the legitimate user. Okamoto [8] proposes to use Twitter to collect simple, memorable question/answer pairs. It requires the uses to manually input the knowledge base, so it is not practical to be used on the smart phones.

Biometric authentication has been well-studied too. Biometric authentication techniques use either physiological or behavioral features [3, 10]. Physiological biometrics utilizes static physical features of humans that are known to be unique for each individual. These may be based upon fingerprint [2], face patterns [11, 1], iris [4], or speech patterns [9], etc. Among these the most common technique is to use fingerprints to authenticate users. Built-in fingerprint readers have been integrated in iPhone 5S.

Behavioral biometrics utilizes the fact that people have distinct stable patterns on a certain behavior, such as keystroke on a keyboard. Two of the most widely studied techniques for personal computers are keystroke-based [12] and mouse movement based [13]. However, due to the size limitations it is difficult to apply these techniques to smart phones. An approach was proposed to provide authentication by capturing the finger movement patterns and comparing them against the owner's patterns [5]. Also, biometric technique based upon personal gait was proposed using accelerometer sensors [14]. The behavioral biometrics may provide usability and attack resistance, but most of them utilize complex algorithms for learning and matching the behavioral patterns and they are vulnerable to high rates of false positives and false negatives. Also, if a user's biometric is stolen, it would be impossible to change it. Because of these reasons behavioral biometrics are generally used in conjunction with a password/PIN rather than used as a sole authentication mechanism.

Conclusions

We introduced and developed an authentication scheme utilizing a series of binary geo-temporal queries based upon the users' location history information. Two pre-termination techniques were also proposed to minimize the privacy leakage level. We formulated a design time optimization problem with various security constraints and an objective function to minimize the privacy disclosure level. The iterative solution approach was presented to the optimization problem. A prototype Android app was developed and the implementation details and issues were presented along with future works.

We believe that the proposed approach satisfies the design objectives mentioned in the beginning of this report: implementation feasibility, attack resistance, quantification and regulation of security risks, and usability.

REFERENCES

[1] J. Daugman. High confidence visual recognition of persons
by a test of statistical independence. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 15(11):1148–1161, 1993.
[2] A. Jain, L. Hong, S. Pankanti, and R. Bolle. An identity-authentication system using fingerprints. Proceedings of the IEEE, 85(9), 1997.
[3] A. K. Jain, R. Bolle, and S. Pankanti, editors. Biometrics: Personal Identification in Networked Society. Kluwer Academic Publishers, 1998.
[4] S. Kurkovsky, T. Carpenter, and C. MacDonald. Experiments
with simple iris recognition for mobile phones. In Proceedings of the 2010 Seventh International Conference on Information Technology: New Generations, ITNG '10, pages 1293–1294, 2010.
[5] L. Li, X. Zhao, and G. Xue. Unobservable re-authentication for smartphones. In ISOC Network and Distributed System Security Symposium (NDSS), February 2013.
[6] M. Nishigaki and M. Koike. A user authentication based on personal history: A user authentication system using e-mail history. 47(3):945–956, 2006.
[7] A. Nosseir, R. Connor, and M. Dunlop. Internet authentication based on personal history - a feasibility test. In Proceedings of Customer Focused Mobile Services Workshop at WWW2005, 2005.
[8] M. Okamoto. Knowledge-based authentication using twitter can we use lunch menus as passwords? 5(5).
[9] K. N. Stevens, C. Williams, J. Carbonell, and B. Woods. Speaker authentication and identification: a comparison of spectrographic and auditory presentations of speech material. The Journal of the Acoustical Society of America, 44:1596, 1968.
[10] J. Wayman, A. Jain, D. Maltoni, and D. Maio. An introduction to biometric authentication systems. In J. ayman, A. Jain, D. Maltoni, and D. Maio, editors, *Biometric Systems*, pages 1–20. Springer London, 2005.
[11] J. Zhang, Y. Yan, and M. Lades. Face recognition: eigenface,
elastic matching, and neural nets. Proceedings of the IEEE, 85(9):1423–1435, 1997.
[12] F. Bergadano, D. Gunetti, and C. Picardi. User authentication through keystroke dynamics. ACM Transactions on Information and System Security (TISSEC), 5(4):367–397, 2002.
[13] N. Zheng, A. Paloski, and H. Wang. An efficient user verification system via mouse movements. In Proceedings of ACM CCS2012, pages 139–150. ACM, 2011.
[14] D. Gafurov, K. Helkala, and T. Søndrol, Biometric Gait Authentication Using Accelerometer Sensor. Journal of Computers, Vol. 1, No. 7, Oct./Nov. 2006.

[15] K. Lin, A. Kansal, D. Lymberopoulos, F. Zhao, **Energy-accuracy trade-off for continuous mobile device location.** Proceedings of the 8th international conference on Mobile systems, applications, and services. Pages 285-298. 2010.

[16] B. Choi, K. Sun, S. Choi, Cloud-Based User Authentication with Geo-Temporal Queries on Smartphones. Proceedings of the 2nd International Workshop on Security in Cloud Computing, Kyoto, Japan, June 3, 2014.

3. Implementation of Geo-Temporal Query Based Authentication Tool on Smartphones

Introduction to Android and Geounlock

Geounlock is a geotemporal user access and security system based on an original design proposed in our previous paper on geo-temporal query-based authentication technique. The system runs on Android devices on version 4.0.3 and above (API level 15). It restricts access to smartphones based on a user's location history. In today's technological society, most people carry a smartphone, be it an iPhone or an Android phone. 52.1 percent of people with smartphones in the United States and 84 percent worldwide have Android running on their smartphones, which makes it the most popular smartphone operating system in the world.

Android became so popular because of its free, open-source nature, and because it is based in Java, one of the most popular programming languages today. However, its largely open nature has also left it open to security issues. Android has many security problems ranging from malware (Android malware represents 97 percent of malware for mobile operating systems today) to malicious access from unauthorized users. Geounlock is designed to protect against malicious access in a user-friendly way.

Android security

Currently, there are six default methods of unlocking an Android device at the lockscreen. These methods serve as the "gatekeeper" to restrict access to a smart device and the sensitive data it may hold. Two, 'None' and 'Slide' offer frictionless access to the device with no security. 'Face Unlock' involves using the device's front-facing camera and Android's in-built image recognition technology to recognize the face of the owner by comparing it to a stored image of the owner's face taken during initial setup. Face Unlock is subject to false negatives, false positives, and can be fooled by a photograph or other still image of the user. 'PIN' allows a user to set a four-digit PIN, much like the ones used at bank ATMs to access checking accounts. There are 10,000 possible combinations of four digit PINs from 0000 to 9999, so it is not probable that one chosen will be unique. 'Pattern' is likely the most secure of the default unlock methods. Presented with a three by three grid of nine points, a user would create a pattern connecting at least four of the points, without visiting the same one twice. This results in 389,112 possible patterns, more likely to be unique. These, however, are susceptible to so-called 'smudge attacks', however, where a user's fingerprints or smudges left on the screens can clue in a malicious user to a device owner's set pattern. The final default method is 'Password', which is a typical alphanumeric password. Special characters can be use. The only password requirements placed on the user for unlocking their Android device is that the password be at least four characters long and have at least one letter, far below today's normal password requirements.

Introduction to Geounlock

The default device unlock methods in Android are either insecure, easily fooled, or not user-friendly. Geounlock provides an unlock method which should be secure, unique, and user-friendly. Geounlock consists of four key components: location tracking, query generation, location database, and statistics gathering. These four components, when combined, provide a series of five true or false questions in the form of 'Did you visit **location** on **date** at **time**?' Of course the application begins with default question as seen in the screen shut on image 1 until the database populated for at least 7 days of data. A series of five questions must be answered correctly to grant access to the device. If two or more are answered incorrectly, access will not be granted, and the device will be locked and go to sleep for a short period.

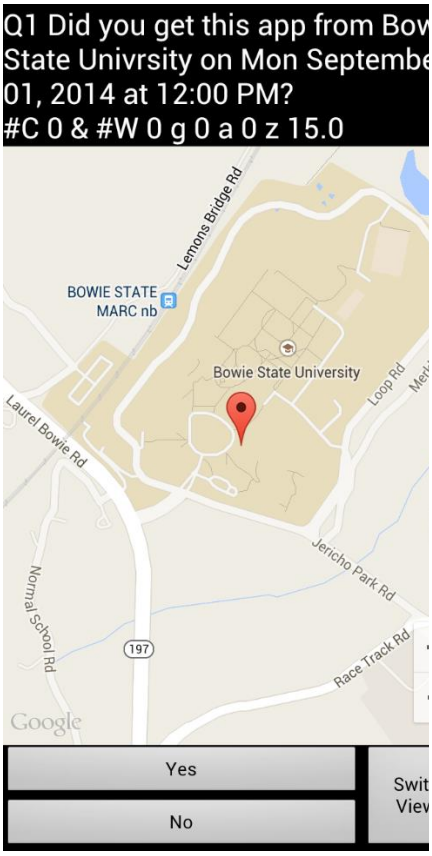


Image 1 – First time lunch of the app with default question.

Location tracking

The first of the key components utilized by the app is location tracking. Geounlock uses Android's fine device location positioning and a compatible device's GPS receiver, cellular modem, or Wi-Fi radio to determine a device's location accurate to within a few meters. The location data provided by the location service consists of a timestamp, a lattitude, and a longitude. The app takes this basic location data and uses a Google Play Services geocoding service to obtain possible addresses, city, state or province, postal code, and country for a set of given latitude and longitude. This is necessary, as latitude and longitude are generally long decimal values, which are unremarkable and unrecognizable by most people, whereas city and state pairs are generally easily remembered. This is necessary for the general usage model for the application, where users will be

required to remember general information about their travels in order to unlock their devices running Geounlock.

The fine location provider within the app is configured to collect the device's location every few seconds or whenever the device is detected to have moved from its previous location to a new location and is no longer moving. This is designed to both cut back on extraneous positions collected by the device, which might lead to situations where the app is presenting a question about a position during a user's travels in route to a destination, rather than questions about a user's actual destinations, which are more memorable.

Database

Location history is stored in the phone's default storage environment, usually a microSD card or internal storage, in the form of a SQLite relational database. The database structure within Geounlock has evolved over time, but the current structure was devised to increase the overall efficiency of the app's internal processes. Generally, the app has had two core databases, one consisting of time stamped latitude and longitude data, and the other consisting of corresponding address data reverse geocoded using the Google Maps API for Android.

Geolocation Database

0	1	2
ID	Latitude	Longitude

Address Database

	1	2		4	5	6	7	8
ID	Street Number	Street Name	City	State/Province	ZIP/ Postal Code	Country	Tag/Familiar Name	'Geo' pointer

Frequency Database

0	1	2	3
ID	Timestamp	Pointer to Geolocation DB	Pointer to Address DB

Table 1 - Current Geounlock database structure

The previous location tracking scheme, where the device was polled for updated position every few seconds, resulted in tens of thousands of entries in the latitude/longitude database and high data usage as the reverse geocoding service had to match thousands of coordinate pairs to addresses and locations on the map. The current system, by contrast, records locations a single time in the database, and instead increments an integer field in a corresponding database entry each time you revisit that location, removing the need for many duplicate entries in the database.

Query Generation

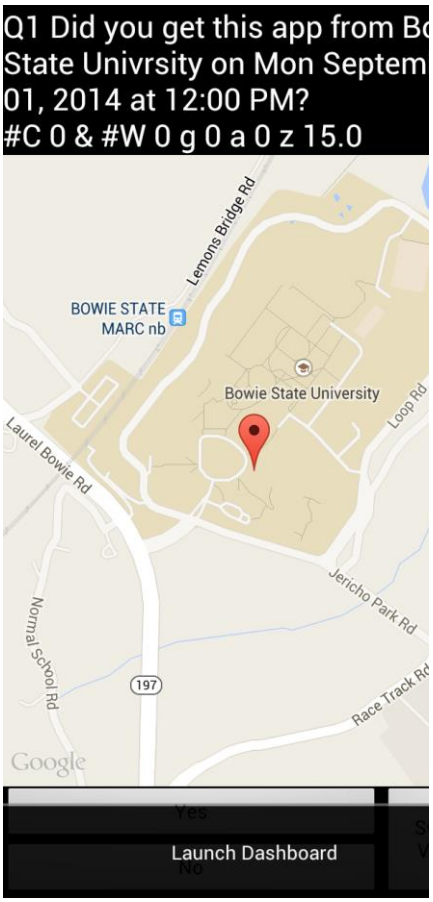
The user is asked questions about their location history by the app's query generation methods. The processes involved are both the most complex and most important in the Geounlock process flow, as they serve as the 'gatekeeper' to the device, allowing or denying access to the device after a series of questions about the location history. Five questions are randomly generated from data stored in the database. Each question has four key attributes: whether it is true or false, the latitude and longitude attached to the question, the address attached to the latitude and longitude, and the time and date attached to that particular visit.

The query generator first randomly selects whether the question will be true or false. If it is to be true, the generator will simply select a pair of coordinates randomly from a database entry, and get the corresponding address, date, and time to create a question that will be actually based on real location data gathered by the device. If the question is to be false, the process is slightly more complex. In order to create a truly random question, it randomizes the coordinates, address, date, and time to be used in a question so that they are unrelated, but still form a realistic question.

Queries generated by this process are displayed, along with a map showing the location in question, as the user's lockscreen, when the phone is woken from sleep. The first question is displayed, and when answered, advances to the next. When five questions are answered correctly, the lockscreen portion of the app is terminated, and the user is granted access to the home screen. The location tracking service will continue to operate in the background for as long as the phone is powered on. Should any two questions be answered incorrectly, a message will be displayed informing the user that too many questions were answered incorrectly, and the device will be locked and returned to sleep. When the device is woken again, five new questions will be asked.

Statistics Gathering

The last key function of Geounlock is statistics gathering. The main purpose of developing the application was as a research experiment. To support this purpose, varieties of statistics are gathered, viewable in a 'dashboard'. The dashboard also supports the changing of app-wide options and settings. The dashboard is accessible through an option in the lockscreen's option menu as seen at image 2. When the lockscreen challenge results in a success, the user is taken to the dashboard. Various settings regarding the app's operation are present for toggle. Additionally, a user is able to review his or her location history, including maps and coordinates, a 'street view' panorama of the area, and is able to export data to a file for migration or reporting, and can import from a previous backup.



Im
option m

Future Improvements

Though the prototype of Geounlock built for the purposes of this project is feature complete, some changes may be made in the future to further improve its functionality. As part of one of these changes, the database system will be modified to allow for the gathering of additional statistics. In addition to the three current databases for coordinates, address, and frequency, a fourth database will be added. Each time a question is asked, the question, time it was asked, the data source, and the answer from the user will be stored to see the differences in answering and difficulty over time. These will be reported as part of the statistics gathering function. There is a small possibility where the query generation system may create false positives and false negatives, where a false generated query may happen to generate a query which contains coordinate, address, time, and date attributes that are, in actuality, linked to a legitimate visit in the database, but were coincidentally randomly selected to form a false query. Future changes to the query

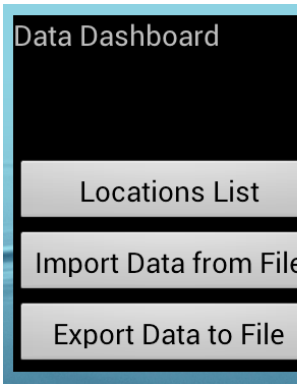


Image 3 – Dashboard menu.

generator will seek to minimize these instances by checking for a link between attributes when a false query is generated. In these instances, the linked attributes will be re-randomized.

4. Designing a Two-Level Monitoring Method to Detect Network Abnormal Behaviors

I. Introduction

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

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

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

II. Literature Review

As mentioned above, there are two globally known techniques in network anomaly detection as signature-based and feature-based detection techniques [2]. The signature-based technique is a common method used in Intrusion Detection System (IDS) that can identify attacks with referencing known signature patterns. Although signature-based approach is good for identifying network anomaly matched to pre-defined signatures, it has a limitation of detecting unknown network anomalies. To avoid this limitation, the feature-based technique is used because it does not require any prior knowledge. Numerous studies have been performed to identify effective approaches for extracting anomalous network traffic features. Among them, statistical methods and machine learning techniques are broadly used to identify anomalous network traffic features.

In the past, researchers studied on increasing the rate of detecting network attacks. Cheng et al. [3] proposed an approach of identifying normal TCP flows by using spectral analysis techniques to protect legitimate TCP flow from Denial of Service (DoS) attacks. Wang et al. [4] proposed statistics-based approach to detect TCP SYN flood attacks, which uses a nonparametric cumulative sum (CUSUM) method. Utilization of statistical approaches is good for maintaining high accuracy with spending reasonably short detection times because it approximately calculates normal traffic patterns to perform a comparison with abnormal traffics. However, it has a difficulty of detecting anomalies caused by network system failures. To resolve this difficulty, Thottan and Ji [5] used a statistical data analysis method with a signal processing technique together to quantify network behaviors to understand network anomalies. They classified network anomalies into two categories as network performance anomalies (e.g. file server failures, paging across the network, broadcast storms) and security-related problems or attacks. They showed that their approach of integrating a signal processing technique is effective for detecting several network anomalies. However, there was no accurate statistical model that was utilized to detect different abnormal traffic patterns. Due to this limitation, researcher started applying various techniques including neural networks, machine learning techniques, data mining, and so forth.

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

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

III. METHODOLOGY

Identifying abnormal behavior from network traffic is critical to maintaining a network environment secure. Abnormal behaviors detection should be performed with satisfying performance and reliability. Specifically, detailed information about detected abnormal behaviors should be provided when notifying detected abnormal behaviors to protect computing infrastructures efficiently. With our proposed predictive model, identification of exact attack types is performed to address this need. Our approach begins with generating reliable rules for detecting network abnormality (normal/abnormal) with classification and regression trees (CART). Once its abnormality is detected, an over-lapping sliding window operation is applied to extract features. Although the extracted features might contain the characteristics of abnormal behaviors, it is important to utilize only significant features to increase the accuracy

of determining exact types of abnormal behaviors. In our study, we used Statistical Analysis System (SAS) to identify significant features, with which our proposed predictive model is designed. Figure 1 illustrates how the predictive model is designed. A detailed explanation about each step is included in the following subsections.

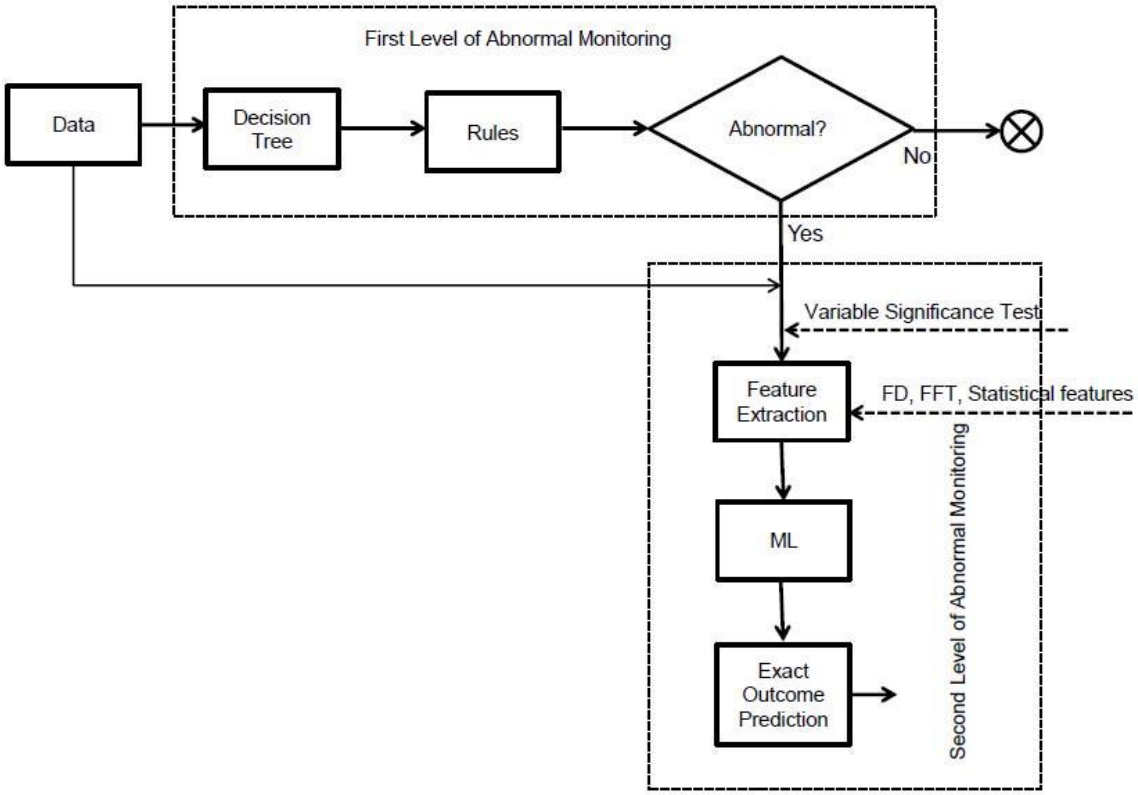


Fig. 1. The entire process of the proposed predictive model consisting of two levels.

A. Dataset

In this study, we used a new version of KDD dataset, NSLKDD dataset, which is publicly available for researchers [14], [15]. Since the KDD Cup'99 intrusion detection dataset includes a lot of redundant records, this redundancy causes an inefficient learning process. For instance, the difference between unauthorized access from a remote machine (R2L) and unauthorized access to local administrative privileges (U2L) is not clear. Due to this reason, it is difficult to distinguish them precisely. To resolve this issue, repeated records are removed from the KDD Cup'99 dataset [14]. We used total 125,973 records for training. For the testing, the rest of the data records (22,544 records) were used. The NSL-KDD dataset includes total 41 attributes (three nominal, six binary, and thirty-two numeric attributes) and four types of attacks (Denial of service (DoS), user to root (U2R), remote to user (R2L), and Probes). DOS attack indicates attempts to disabling network access to remote machines (or computing resources). R2L represents that a remote user gains access to local user accounts by sending packets to a computing machine over the network. U2R explains that an attacker accesses normal users' accounts by exploring the system as a root-user. Probing represents that the network is scanned to gather information to find known vulnerabilities.

TABLE I. DATA DISTRIBUTION OF THE NSL-KDD TRAINING AND TESTING DATASETS.

	Normal	Abnormal			
	-	DoS	U2R	R2L	Probes
Training	53.50%	37.20%	0.04%	0.08%	10.80%
Testing	43.10%	33.10%	0.20%	12.80%	10.70%

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

B. Methods

As mentioned above, our proposed network traffic detection is designed consisting of two levels. First, reliable rules are generated by using a decision tree method. Then, designing a predictive model is performed with utilizing the reliable rules.

Level 1: abnormal detection In this level, detection of abnormal network traffic behaviors is executed.

	Normal	Abnormal			
	-	DoS	U2R	R2L	Probes
Training	53.50 %	37.20 %	0.04 %	0.08%	10.80 %
Testing	43.10 %	33.10 %	0.20 %	12.80 %	10.70 %

Specifically, normal and abnormal behaviors are determined. To perform this, rules are generated with classification and regression tree (CART) [16], which uses information theoretic concepts to create a decision tree that captures complex patterns in data. We used four features (duration, protocol type, service, and flag) to create a tree and to support a rapid decision. There are three attribute values in the protocol type, seventy attributes in the service, and eleven attribute values in the flag. The service feature denotes specific network services (e.g. HTTP, FTP, etc.) on the destination. The flag indicates network connection status representing how each connection is instantiated and terminated. Total 13 connection states can be identified from network traffic [17].

CART builds a tree for predicting continuous dependent variables (regression) and categorical predictor variables (classification). It is broadly used due to its efficiency in dealing with multiple data types and missing values. CART expression forms explicit and transparent grammatical rules [18], [19]. Also, it uses an exhaustive search of all variables and split values to find optimal splitting values for each node. This splitting stops at the pure node containing fewer examples. Advantages of using CART are a) it does not require any distributional assumptions for dependent and independent variables, b) it deals with multiple types of numerical and categorical variables as inputs and outputs, c) it is not affected by outliers, and d) it efficiently handles high dimensional data. Among the generated trees with CART, only the trees that give high training accuracy are selected. Rules are extracted from the selected trees and tested with the testing dataset. With these rules, abnormality of network traffic is determined.

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

We used both fractal domain and statistical measurements to extract features. A sliding window (=20 data points) with 67% of overlapping is applied to examine the variation of network traffic flow. Fractal dimension (FD), called a non-linear dynamical method, is based on fractal theory concept. This method is applied broadly in many areas including biology, image segmentation, audio signal analysis, and medicine [20], [21], [22]. FD is a useful method for detecting rapid variations from data [21]. It also presents a self-similarity measurement of data. The self-similarity can be connected with “fractals” and the fractals within the network traffic display shape similarities in time scales. Since fractal features express the degree of self-similarity of the data, any unusual patterns can be detected. There are several algorithms to calculate the FD such as box-counting [23], Katz’s [24], and Higuchi [25].

In this study, Higuchi FD is used as a fractal feature extraction method because it requires less computational power, guarantees accurate estimation, supports memory efficiency than other methods. To the best of our knowledge, Higuchi FD has not been used to extract features for determining exact attack types. The Higuchi method first re-generates original input data as a finite time series subset based on pre-defined window size (k=5). For the given the input data, new finite time-series is constructed as follows:

$$x(m), x(m+k), x(m+2k), ..., x\Big(m + \left[\frac{N-m}{k}\right] \cdot k\Big), \quad m = 1, 2, ..., k$$

where "[ ]" denotes the Gauss' notation, the largest integer in the neighborhood of the number, and both k and m indicate internal and initial time, respectively. The length of the curve x^m_k is defined as

$$L_m(k) = \frac{1}{K}\left\{\left(\sum_{i=1}^{\frac{N-m}{k}} |x(m+ik) - x(m+(i-1)\cdot k|\right)\frac{N-1}{\lfloor\frac{N-m}{k}\rfloor \cdot k}\right\}$$

N is the total length of the signal. <L(k) > defines the length of the curve for the time series k and < L_m(k) > denotes the average value over k.

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

IV. Experimental Results

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

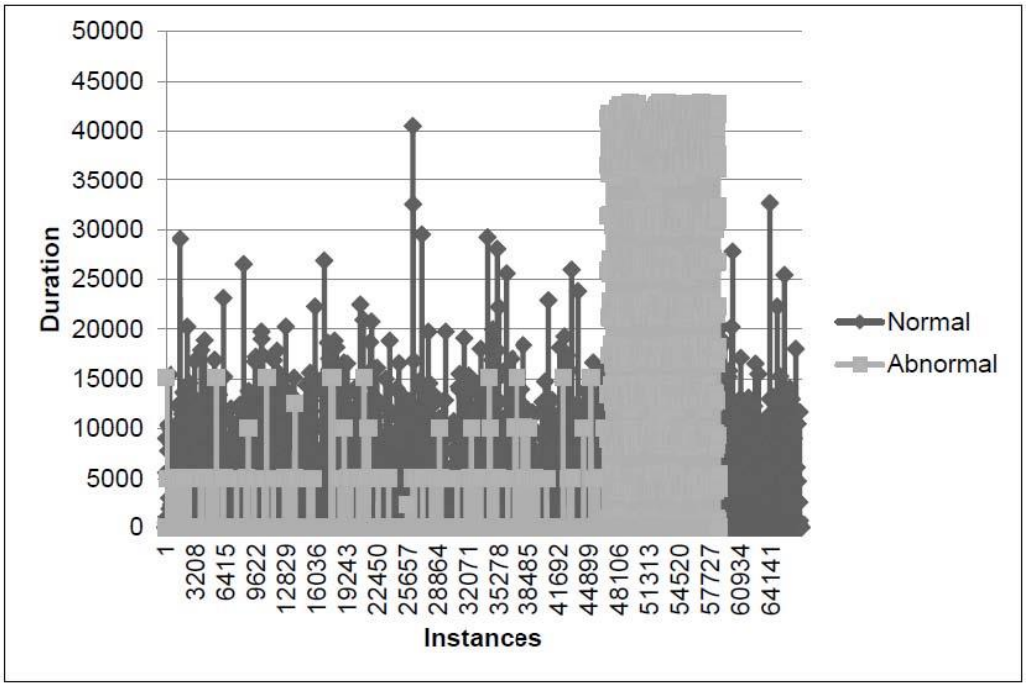
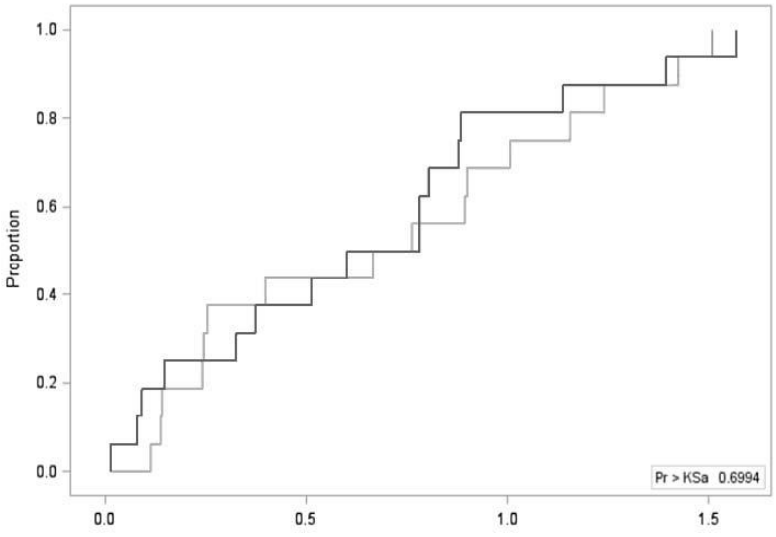


Fig. 2. A raw feature ('duration') distribution comparison of the training dataset.

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



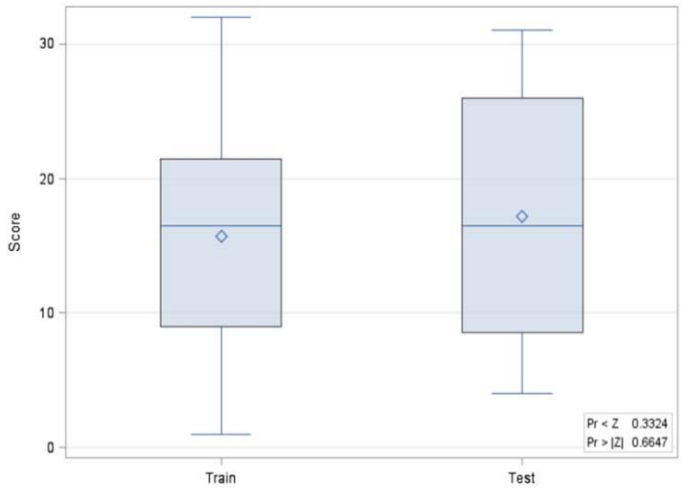


Fig. 3. Distribution test results of the feature ('duration') by (a) Kolmogorov-Smirnov test (dark and light gray indicate the training and testing datasets, respectively) and (b) Wilcoxon rank-sum test.

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

As mentioned previously, CART is considered for rule extraction. All generated rules with the training dataset are tested with the testing dataset. As a result, overall testing accuracy was 80.73%. Only the rules with high accuracy and

TABLE II. SOME OF THE GENERATED RULES MAINTAINING HIGH ACCURACY ARE PRESENTED.

Rules	Outcome (Accuracy - records)
$If(PT \neq \gamma \& PT \neq \mu \& (FLG = \epsilon \vert FLG = \varepsilon \vert FLG = \theta \vert FLG = \vartheta) \& SRV \neq \alpha \& SRV \neq \beta)$	Abnormal (99.7% - 6548/6556)
$If(PT = \gamma \& FLG \neq \delta \& FLG \neq \varepsilon)$	Normal (88.0% - 6669/7573)
$If(PT \neq \gamma \& PT \neq \mu \& FLG = \rho \& duration < 8.5)$	Abnormal (72% - 89/124)

where SRV:service,FLG:flag, PT:protocol type,α:SMTP, β:x11,γ:HTTP,δ:S1,ε:RSTR,ϵ:OTH, θ:SH,ϑ:REJ,μ:IRC,ρ:RSTO.

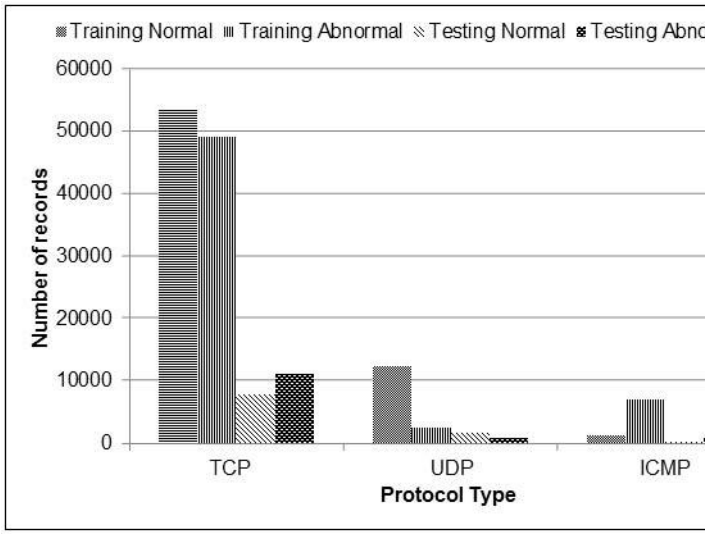
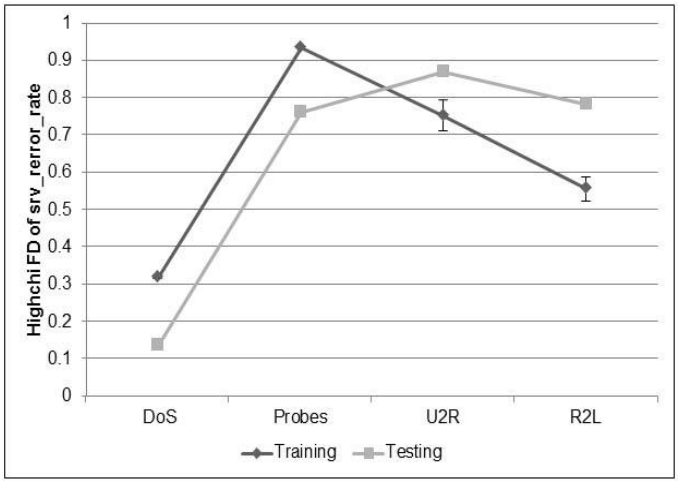


Fig. 4. Distribution of network traffic records depending on the protocol types in the training and testing datasets.

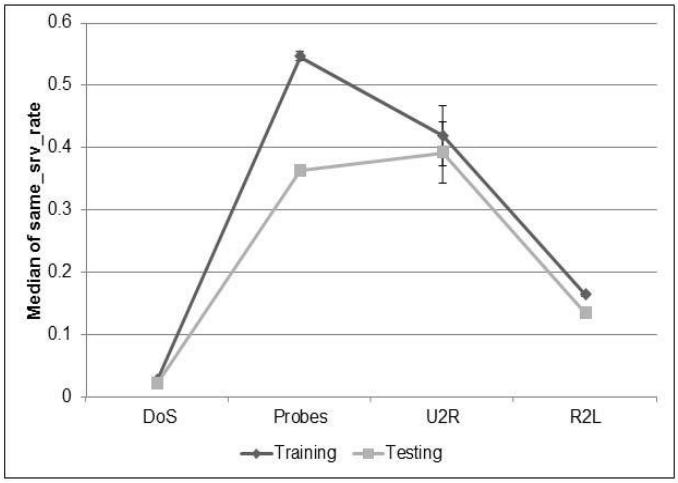
large number of examples are used to form a rule base. Some of the rules are listed in Table II.

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

Since the predictive model is designed with the training dataset, it is important to see how the proposed predictive model is efficient for detecting network anomalies from the testing dataset. To see the effectiveness of the model, we focused on understanding the difference between the training and testing datasets in consideration of the extracted features. Figure 5(a) and 5(b) show Higuchi fractal dimension feature of the network feature (srv rerror rate) and the statistical measurement feature (i.e. median) of the network feature pattern of the variable "srv rerror rate" between the training and testing datasets. The variable "srv rerror rate" indicates connections that have REJ errors ins services. Although there are differences among attacks, the extracted features maintain similar trends in the training and testing datasets. It addresses that detecting abnormal network traffics in the testing dataset can be performed precisely. Figure 5(b) represents mean and the standard error of the mean (SEM) of the feature (same srv rate). This feature denotes connections that have the same services on the network. This figure indicates that all attack types except Probes retain similar characteristics in the training and testing datasets. It explains us that our predictive model will maintain high accuracy when identifying abnormal behaviors in network traffic. In addition, we examine the input feature to see the difference between before and after applying Higuchi FD. For this, their mean are compared.



(a)



(b)

Fig. 5. Comparative results (mean ± SEM) of the features from (a) Higuchi fractal dimension and (b) statistical measurement.

Figure 6 shows the result of the feature (srv rerror rate) for the DoS attack between the raw feature and its Higuchi feature in the training and testing datasets. It explains that since the Higuchi FD feature well maintains its characteristics compared to the raw feature, the predictive model designed with the extracted Higuchi FD features can successfully distinguish new incoming exact types of attacks.

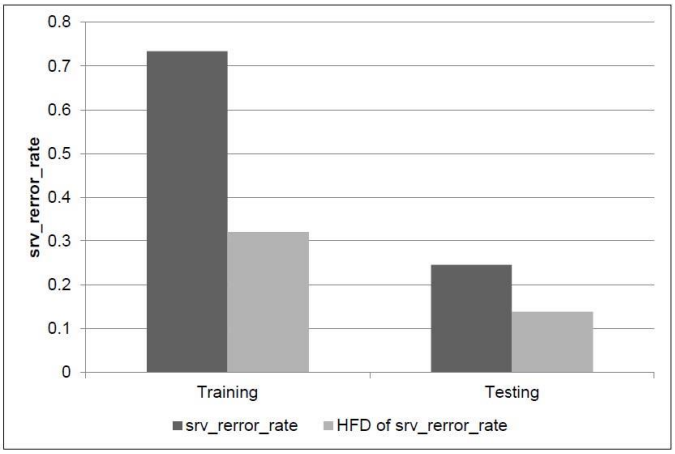


Fig. 6. An example result about the mean of the feature (srv rerror rate) for the DoS attack, in consideration of the training and testing datasets.

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

	Accuracy	ROC area
SVM	77.10%	0.879
NN	51.90%	0.643

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

V. DISCUSSION AND CONCLUSION

Understanding network traffic is important for securing our computing infrastructures. However, it is difficult to differentiate normal network traffic from abnormal network behaviors. This project contributes to designing a two-level abnormal network traffic monitoring method. Since the most critical advantage of adapting rule-based method is that transparent rules can provide reasons behind the predictions, we generated rules with CART to differentiate normal and abnormal. We also proposed a predictive model to identify exact attack types of abnormal network traffic. Instead of using the raw feature for the predictive model, Higuchi fractal dimension and statistical measures are applied to extract significant features. Prior to applying them directly to design the predictive model, all extracted features were analyzed to determine their statistical significances. As explained in Section IV, we found that the performance accuracy of

detecting network anomaly was high when using the extracted features.

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

REFERENCES

[1] A. Kind, M. Stoecklin, and X. Dimitropoulos, "Histogram-based traffic anomaly detection," *Network and Service Management, IEEE Transactions on*, vol. 6, no. 2, pp. 110–121, June 2009.

[2] A. Patcha and J. Park, "An overview of anomaly detection techniques: Existing solutions and latest technological trends," *Computer Networks*, vol. 51, no. 12, pp. 3448–3470, Aug. 2007.

[3] C.-M. Cheng, H. T. Kung, and K.-S. Tan, "Use of spectral analysis in defense against dos attacks." in *GLOBECOM*. IEEE, 2002, pp. 2143–2148.

[4] H. Wang, D. Zhang, and K. Shin, "Detecting syn flooding attacks," in
INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, vol. 3, June 2002, pp. 1530–1539.

[5] M. Thottan and C. Ji, "Anomaly detection in ip networks," *Signal Processing, IEEE Transactions on*, vol. 51, no. 8, pp. 2191–2204, Aug 2003.

[6] J. Cannady, "Artificial neural networks for misuse detection," in *National Information Systems Security Conference*, 1998, pp. 443–456.

[7] R. P. Lippmann and R. K. Cunningham, "Improving intrusion detection performance using keyword selection and neural networks," *Computer Networks*, vol. 34, no. 4, pp. 597 – 603, 2000, recent Advances in Intrusion Detection Systems.

[8] S. T. Sarasamma, Q. A. Zhu, and J. Huff, "Hierarchical kohonenen net for anomaly detection in network security." *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 35, no. 2, pp. 302–312, 2005.

[9] S.-J. Han, K.-J. Kim, and S.-B. Cho, "Evolutionary learning programs behavior in neural networks for anomaly detection," in *Neural Information Processing*, ser. Lecture Notes in Computer Science, N. Pal, N. Kasabov, R. Mudi, S. Pal, and S. Parui, Eds. Springer Berlin Heidelberg, 2004, vol. 3316, pp. 236–241.

[10] T. Shon and J. Moon, "A hybrid machine learning approach to network anomaly detection," *Inf. Sci.*, vol. 177, no. 18, pp. 3799–3821, Sep. 2007.

[11] R. Jain and N. Abouzakhar, "A comparative study of hidden markov model and support vector machine in anomaly intrusion detection," *Journal of Internet Technology and Secured Transactions (JITST)*, vol. 2, no. 1/2/3/4, pp. 176–184, 2013.

[12] A. Hofmann and B. Sick, "Evolutionary optimization of radial basis function networks for intrusion detection," in *Neural Networks, 2003. Proceedings of the International Joint Conference on*, vol. 1, July 2003, pp. 415–420 vol.1.

[13] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," in *In ACM SIGCOMM*, 2004, pp. 219–230.

[14] "NSL-KDD dataset," http://nsl.cs.unb.ca/NSL-KDD/, 2014, [Online; accessed 2-April-2014].

[15] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *Proceedings of the Second IEEE International Conference on Computational Intelligence for Security and Defense Applications*, ser. CISDA'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 53–58.

[16] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. New York: Chapman & Hall, 1984.

[17] V. Paxson, “Bro: A system for detecting network intruders in real-time,” *Comput. Netw.*, vol. 31, no. 23-24, pp. 2435–2463, Dec. 1999.

[18] W.-Y. Loh and N. Vanichsetakul, “Tree-structured classification via generalized discriminant analysis,” *Journal of the American Statistical Association*, vol. 83, no. 403, pp. pp. 715–725, 1988.

[19] C. Y. Fu, “Combining loglinear model with classification and regression tree (cart): an application to birth data,” *Computational Statistics & Data Analysis*, vol. 45, no. 4, pp. 865 – 874, 2004.

[20] U. R. Acharya, P. K. Joseph, N. Kannathal, C. M. Lim, and J. S. Suri, “Heart rate variability: a review.” *Med. Biol. Engineering and Computing*, vol. 44, no. 12, pp. 1031–1051, 2006.

[21] M. Liu, Y. He, Q. Meng, and Z. Wang, “Research on anomaly detection of network traffic based on fractal technology and vector quantization,” in *Education Technology and Computer Science (ETCS), 2010 Second International Workshop on*, vol. 2, March 2010, pp. 428–431.

[22] C. G´omez, A. Mediavilla, R. Hornero, D. Ab´asolo, and A. Fern´andez, “Use of the higuchi’s fractal dimension for the analysis of meg recordings from alzheimer’s disease patients.” *Med Eng Phys*, vol. 31, no. 3, pp. 306–13, 2009.

[23] J. Li, Q. Du, and C. Sun, “An improved box-counting method for image fractal dimension estimation,” *Pattern Recognition*, vol. 42, no. 11, pp. 2460 – 2469, 2009.

[24] M. J. Katz, “Fractals and the analysis of waveforms,” *Computers in Biology and Medicine*, vol. 18, no. 3, pp. 145 – 156, 1988.

[25] T. Higuchi, “Approach to an irregular time series on the basis of the fractal theory,” *Physica D: Nonlinear Phenomena*, vol. 31, no. 2, pp. 277 – 283, 1988.

[26] V. N. Vapnik, *Statistical Learning Theory*. Wiley-Interscience, 1998.

[27] S. Idicula-Thomas, A. J. Kulkarni, B. D. Kulkarni, V. K. Jayaraman, and P. V. Balaji, “A support vector machine-based method for predicting the propensity of a protein to be soluble or to form inclusion body on overexpression in escherichia coli.” *Bioinformatics*, vol. 22, no. 3, pp. 278–284, 2006.

[28] T. Joachims, “Text categorization with suport vector machines: Learning with many relevant features,” in *Proceedings of the 10th European Conference on Machine Learning*, ser. ECML ’98. London, UK, UK: Springer-Verlag, 1998, pp. 137–142.

[29] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” *ACM Comput. Surv.*, vol. 41, no. 3, pp. 15:1–15:58, Jul. 2009.

[30] E. L. Lehmann and H. D’Abrera, *Nonparametrics : statistical methods based on ranks*, ser. Holden-Day series in probability and statistics. San Francisco: Holden-Day New York Dusseldorf Johannesbourg, 2006, revised edition.

5. FSM Watermarks Based on Ordering of Flip Flops

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

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

General Terms
Security, Design, Algorithms

Keywords
FSM watermarking, flip-flop arrangement, cyclic property

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

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

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

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

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

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

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

PRELIMINARY

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

Definitions and Assumptions

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

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

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

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

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

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

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

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

Illustrative Example

Table 1 shows an example of a sequential design that is represented in a state table [5]. In D_1, there are six states. When a stimulating external input value is applied on the present state, it is deterministically moving to the next state while generating an external output.

One possible complete flip-flop arrangement is: {(A, 000), (B, 001), (C, 010), (D, 011), (E, 100), (F, 101)}. The corresponding logical equations *using* this particular assignment are: $Y_1 = y_2$, $Y_2 = y_1'y_2'$, $Y_3 = xy_3 + xy_2 + x'y_2'y_3' + y_2y_3$, and $z = xy_3'$, where y_i and Y_i represent the present state and the next state, respectively.

Note that D_1 has the following interesting properties. It contains a component that is both (1) independent of the external input, and (2) three states forming a cycle. Consider three combined states {α; β; γ} = {A+B; C+D; E+F}, where "+" is used to denote an operator to combine two states. Then, the transition function of this component is defined as $(\alpha, -) = \beta$, $\delta(\beta, -) = \gamma$, and $\delta(\gamma, -) = \alpha$, where "−" denotes a *don't-care* condition.

Table 1. Sequential Design (D_1)

Present State Q(t)	Next State Q(t+1), Output z	
	Input x = 0	Input x = 1
A	D, 0	C, 1
B	C, 0	D, 0
C	E, 0	F, 1
D	F, 0	F, 0
E	B, 0	A, 1
F	A, 0	B, 0

Putting it all together, the actual flip-flop arrangement can be made as follows: {(A, 000), (B, 001), (C, 010), (D, 011), (E, 100), (F, 101)} with {(α, 00), (β, 01), (γ, 10)} and {(a, 0), (b,1)}. Note that the original states are realized by two internal states: *A* = (α, a), *B* = (α, b), *C* = (β, a), *D* = (β, b), *E* = (γ, a), *F* = (γ, b). For instance, the state "*A*" is realized by two internal states "*α*" and "*a*" using three flip flops.

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

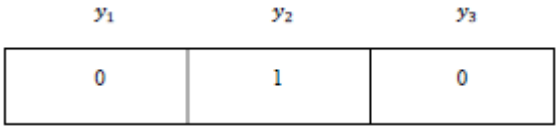


Figure 1. Ordering of flip-flop arrangement.

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

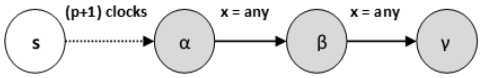


Figure 2. Verification of a cycle of states (with periodicity = 3)

WATERMARKING ALGORITHM

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

Watermark Creation (δ-WFSM)

The watermarked FSM is created using a specific property. The specific property chosen as an example is maximal input-independent periodicity representing a cyclic behavior of a sequential FSM. This specific property is denoted by P*. The watermarked FSM, denoted by δ-WFSM**,** is defined with the four parameters.

WATERMARKING ALGORITHM

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

Watermark Creation (δ-WFSM)

The watermarked FSM is created using a specific property. The specific property chosen as an example is maximal input-independent periodicity representing a cyclic behavior of a sequential FSM. This specific property is denoted by P*. The watermarked FSM, denoted by δ-WFSM**,** is defined with the four parameters.

Procedure Creation ( )
Input: a "n"-state FSM (FSM = (I, O, S, δ, λ), $|S| = n$)
Output: a watermarked FSM (δ -WFSM = (I_w, O_w, S_w, δ_w))
1) Find the *maximal* input-independent periodicity p_{max} from FSM
2) Construct δ -WFSM = (I_w, O_w, S_w, δ_w) as follow:
   a. the input: $I_w = I$
   b. the output: $O_w = O$
   c. the set of states: $S_w = \{s_1^*, s_2^*, \dots, s_{p_{max}}^*\}$
   d. the state transition: $\delta(s_1^*, a) = s_2^*$, $\delta(s_2^*, a) = s_3^*$, $\delta(s_3^*, a) = s_4^*$,....., $\delta(s_{p_{max}}^*, a) = s_1^*$ for $\forall a \in I$

Finding p_{max} is important since it will increase the security level of the hidden watermark. Note that δ –WFSM can usually contain a unique characteristic of a cyclic behavior. Step $d$above is to extract such a cyclic behavior.

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

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

Procedure Max Periodicity ( ):
Input: a "n"-state FSM (FSM = (I, O, S, δ, λ), $|S| = n$)
Output: p_{max}
1) Find a set of a closed partition $\{\pi_1, \pi_2, \dots \dots \pi_t\}$ and a nontrivial input-independent partition τ on S, where $\pi_i \geq \tau$, for $i = 1, 2, \dots \dots$t.
2) Choose the *smallest* closed partition π_{min}

3) The number of blocks in π_{min} is the maximal periodicity p_{max}

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

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

Watermark Verification

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

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

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

ANAYLSIS

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

Existence of the Property

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

Theorem 1: The existence of a closed partition π and a nontrivial input-independent partition τ_i on S in the original FSM = (I, O, S, δ, λ), where $\pi \geq \tau_i$, is a necessary and sufficient condition for the existence of the watermark FSM δ -WFSM = (I_w, O_w, S_w, δ_w). □

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

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

Guessing the Watermark

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

Table 2. Parameter List

Parameters	Description	Conditions
n	$\lvert S \rvert$, the number of states in FSM = (I, O, S, δ, λ)	
m	$\lvert FF \rvert$, the minimum number of flip flops	$m = \lceil \log_2 n \rceil$
$C; c$	C, an ordered set of states; $c = \lvert C \rvert$ = the cardinality of C	$1 \leq c \leq n$
p^*	The maximal periodicity of δ -WFSM	$p^* = \max_{all\ cycles\ of\ C} \lvert C \rvert$ $1 \leq p^* \leq n$

m'	$\|FF'\|$, the minimum number of flip flops	$m' = \lceil \log_2 p^* \rceil$ $m' \leq n$
$\pi(c)$	The number of the ways of arranging $c = \|C\|$ states in order	$\pi(c) = \|C\|!$ $1 \leq \pi(c) \leq n!$
$\Gamma(n, c)$	The number of the ways of arranging c states out of n states	(1)
$G(n, c)$	The number of the ways of arranging both cyclic states and flip flops in order	(2A) and (2B)

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

$$\Gamma(n, c) = (\lceil \log_2 c \rceil)! \times \binom{\lceil \log_2 n \rceil}{\lceil \log_2 c \rceil} \qquad (1)$$

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

$$G(n, c) = \pi(c) \times (\lceil \log_2 c \rceil)! \times \binom{\lceil \log_2 n \rceil}{\lceil \log_2 c \rceil} \qquad (2A)$$
$$1 \leq \pi(c) \leq n!$$

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

General Case: In general, however, the maximal periodicity can be kept in secret (by the owner). In this case, the security level increases by a factor of "*n*" as shown in (2B):

$$G(n, c) = n \times \pi(c) \times (\lceil \log_2 c \rceil)! \times \binom{\lceil \log_2 n \rceil}{\lceil \log_2 c \rceil} \qquad (2B)$$
$$1 \leq \pi(c) \leq n!$$

Lower and Upper Bound: The lower and upper bound of $G(n, c)$ occur when $\pi(c) = 1$ and $\pi(c) = n!$, respectively.

$$G(n, 1) = n \qquad (3A)$$

$$G(n, n) = n \times n! \times (\lceil \log_2 n \rceil)! \qquad (3B)$$

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

Limitation

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

FEASIBILITY

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

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

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

Circuits	FFs	States (n)	Lower Bound ($c = 1$; $\pi(c) = 1$)	Upper Bound ($c = n$; $\pi(c) = n!$)
s27	3	8	8	1935360
s820 s832	5	32	32	1.010422E+39
s1488 s1492 s386 s510	6	64	64	5.8469498E+93
s208	8	256	256	8.854326E+513
s27-n3	9	512	512	6.460615E+1174
S1196 s1238	18	262144	262144	4.45277E+1306615
s991	19	524288	524288	7.115547E+2771033
s382 s400 s444 s526 s526n	21	2097152	2097152	1.576848E+12346668
s15850	597	5.1E+179	5.1E+179	Non-computable
s35932	1728	2^{1728}	2^{1728}	Non-computable

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

Circuits	States (n)	Periodicity (c)	G(n, c)
s27	8	2	48
		4	1152
s820 s832	32	2	320
		4	15360
		8	7.74144E+7
		16	8.0343513E+16
s1488 s1492 s386 s510	64	2	768
		4	46080
		8	3.096576E+8
		16	4.8206108E+17
		32	1.21250689E+40
s208	256	2	4096
		4	344064
		8	3.46816512E+9
		16	8.99847347E+18
		32	4.5266924E+41
		64	6.5485838E+95
		128	3.980343769E+222

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

CONCLUSION

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

8. ACKNOWLEDGMENTS

Our thanks go to the reviewers for the constructive suggestions.

9. REFERENCES

[1] Amr T. Abdel-Hamid, Sofiène Tahar and El Mostapha Aboulhamid. 2004. A survey on IP watermarking techniques. In Design Automation for Embedded Systems, Vol. 9, Springer Science+Business Media, Berlin, 211-227. DOI:http://dx.doi.org/10.1007/s10617-005-1395-x

[2] Franc Brglez, David Bryan and Krzysztof Kozminski. 1989. Combinational profiles of sequential benchmark circuits. In Proceedings of the International Symposium on Circuits and Systems. IEEE Computer Society, Portland, OR, 1929-1934. DOI:http://dx.doi.org/10.1109/ISCAS.1989.100747

[3] Aijiao Cui, Chip-Hong Chang, S. Tahar, and A. T. Abdel-Hamid. 2011. A Robust FSM Watermarking Scheme for IP Protection of Sequential Circuit Design. IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys. 30, 5 (May2011), 678-690.DOI:http://dx.doi.org/10.1109/TCAD.2010.2098131

[4] Juris Hartmanis and R. E Stearns. 1966. Algebraic Structure Theory of Sequential Machines (Prentice-Hall International Series in Applied Mathematics). Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[5] Zvi Kohavi. 1978. Switching and Finite Automata Theory (2nd ed.). McGraw-Hill.

[6] Farinaz Koushanfar and Yousra Alkabani. 2010. Provably secure obfuscation of diverse watermarks for sequential circuits. In Proceedings of the Inter. Symp. on Hardware-Oriented Security and Trust (HOST '10), 42-47. DOI:http://dx.doi.org/10.1109/HST.2010.5513115

[7] Farinaz Koushanfar, Inki Hong, and Miodrag Potkonjak. 2005. Behavioral synthesis techniques for intellectual property protection. ACM Trans. Des. Autom. Electron. Syst. 10, 3 (July 2005), 523-545. DOI:http://doi.acm.org/10.1145/1080334.1080338

[8] Farinaz Koushanfar and Gang Qu. 2001. Hardware metering. In Proceedings of the 38th annual Design Automation Conference (DAC '01). ACM, New York, NY, USA, 490-493. DOI:http://doi.acm.org/10.1145/378239.378568

[9] Farinaz Koushanfar. 2012. Provably Secure Active IC Metering Techniques for Piracy Avoidance and Digital Rights Management. IEEE Trans. Info. For. Sec. Vol. 7, Issue 1 (February 2012), 51-63. DOI:http://dx.doi.org/10.1109/TIFS.2011.2163307

[10] John Lach, William H. Mangione-Smith, and Miodrag Potkonjak. 1999. Robust FPGA intellectual property protection through multiple small watermarks. In Proceedings of the 36th annual ACM/IEEE Design Automation Conference (DAC '99), Mary Jane Irwin (Ed.). ACM, New York, NY, USA, 831-836. DOI:http://doi.acm.org/10.1145/309847.310080

[11] Matthew Lewandowski, Richard Meana, Matthew Morrison and Srinivas Katkoori. 2012. A Novel Method for Watermarking Sequential Circuits. In Proceedings of the IEEE International Symposium on Hardware-Oriented Security and Trust (HOST'12), San Francisco, CA. 21-24. DOI: http://dx.doi.org/10.1109/HST.2012.6224313

[12] Arlindo L. Oliveira. 1999. Robust techniques for watermarking sequential circuit designs. In Proceedings of the 36th annual ACM/IEEE Design Automation Conference (DAC '99), Mary Jane Irwin (Ed.). ACM, New York, NY, USA, 837-842. DOI:http://doi.acm.org/10.1145/309847.310082

[13] Arlindo L Oliveira. 2001. Techniques for the creation of digital watermarks in sequential circuit design. IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys. Vol. 20, Issue 9, (September 2001), 1101-1117. DOI: http://dx.doi.org/10.1109/43.945306

[14] Fabien AP Petitcolas, Ross J. Anderson and Markus G. Kuhn. 1999. Information hiding – a survey. In Proceedings of the IEEE, Vol. 87, Issue 7 (July 1999), 1062-1078. DOI:http://dx.doi.org/10.1109/5.771065

[15] Gang Qu and Miodrag Potkonjak. 1998. Analysis of watermarking techniques for graph coloring problem. In Proceedings of the 1998 IEEE/ACM international conference on Computer-aided design (ICCAD '98). ACM, New York, NY, USA, 190-193. DOI:http://doi.acm.org/10.1145/288548.288607

[16] Pramod Subramanyan, Nestan Tsiskaridze, Wenchao Li, Adrià Gascón, Wei Yang Tan, Ashish Tiwari, Natarajan Shankar, Sanjit A. Seshia and Sharad Malik. 2014. Reverse Engineering Digital Circuits Using Structural and Functional Analyses. To appear in IEEE Transactions on Emerging Topics in Computing.

[17] Mohammad Tehranipoor and Cliff Wang (Ed.). 2012. Introduction to Hardware Security and Trust. Springer Science+Business Media, LLC. Chapter 17.

[18] Ilhami Torunoglu and Edoardo Charbon. 2000. Watermarking-based copyright protection of sequential functions. IEEE Journal of Solid-State Circuits. Vol. 35, Issue 3 (2000), 434-440. DOI:http://dx.doi.org/10.1109/4.826826

[19] J. L. Wong, Gang Qu, and M. Potkonjak. 2006. Optimization-intensive watermarking techniques for decision problems. IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys. 23, 1 (November 2006), 119-127. DOI:http://dx.doi.org/10.1109/TCAD.2003.819900

[20] Lin Yuan and Gang Qu. 2004. Information hiding in finite state machine. In Proceedings of the 6th international conference on Information Hiding (IH'04), Jessica Fridrich (Ed.). Springer-Verlag, Berlin, Heidelberg, 340-354. DOI:http://dx.doi.org/10.1007/978-3-540-30114-1_24

6. Identification of IP control units by state encoding

INTRODUCTION

Hardware design reuse has been the viable solution to deal with the ever-increasing logic density in the semiconductor industry. Hardware design intellectual property (IP) is a design unit that can be viewed as an independent subcomponent of a complete design (e.g., SOC design.) Examples of design unit include abstract algorithm, technique, or methodology that can make the design better as well as physical design blocks such as embedded control units. This can result in new products to be on the market in time and at a cost-effective way. The newly developed subcomponents can also be tested and deposited as new design IPs in the IP library for future reuse. Despite the attractiveness of reuse-based design, IP owners and vendors have encountered IP piracy and infringement. In this work, we address the problem of protecting embedded hardware designs modeled as a synchronous finite state machine (FSM) and present a new type of FSM IP watermarking solution.

Most of the embedded hardware IP protection solutions [4] [7][8][10][11][12] have been developed by hiding secret information that is explicitly added to the circuit in order to prevent illegal use of the IC. Another characteristic of the previous work is to use secret information at a state level. For instance, a designer's secret information (i.e., a watermark) can be defined in terms of a set of visited states (usually non-functional states), which are traveled upon by applying specific input signals. In an analogy, this type of approach would be similar to adding tattoos (either big or small) on the external surface of the human skin to hide the designer's secret information. Then, the tattoos are verified at a later stage in order to prove the ownership of the design IP. For the survey of FSM IP watermarking, refer to [1]. For the recent work on IP protection, refer to [2].

We challenge this traditional design philosophy by raising the question: *Would it be feasible to embed a watermark at a FSM level without adding non-functional entities (e.g., states, state transitions)?* We show that, under a certain condition, this is feasible. We analyze the necessary and sufficient condition based on the theory of machine decomposition [5][6].

In general, the task of watermarking consists of two processes: (1) embedding and (2) verification. The uniqueness of the proposed method lies in non-destructiveness. Here, *non-destructiveness* is meant to satisfy the following two conditions: (1) neither new (i.e., redundant) states nor additional state transitions are added to construct any form of watermark in the embedding process, and (2) no extra circuitry within the FSM system is required to check internal states in the verification process. For the description and proposed solutions for the verification process, refer to [9]. In this work, we shall focus on the embedding method. The underlying idea is based on a FSM-level watermark using the state encoding scheme.

PRELIMINARY

Partitions on States

For convenience, the definitions are summarized below [5].

Definition 1: A *synchronous finite state machine* (or *sequential machine*) is a quintuple $M = (S, I, O, \delta, \lambda)$ where (i) S is finite nonempty set of states; (ii) I is a finite nonempty set of inputs; (iii) O is a finite nonempty set of outputs; (iv) $\delta: S \times I \rightarrow S$ is the transition function; (v) $\lambda: S \times I \rightarrow O$ $(Mealy\ type)$, $\lambda : S \rightarrow O$ $(Moore\ type)$.

A Mealy-type machine can be converted into a Moore-type machine, and vice versa. Unless it is needed to distinguish one type from the other, Mealy-type machine is assumed in the work.

Definition 2: A *state machine* is a triplet $M = (S, I, \delta)$ where (i) S is a finite nonempty set of states; (ii) I is a finite nonempty set of inputs; (iii) $\delta: S \times I \rightarrow S$ is a transition function.

Definition 3: A *partition* φ on S of the machine $M = (S, I, O, \delta, \lambda)$ is a collection of disjoint subsets of S whose set union is S. That is, $\varphi = \{B_\alpha\}$ such that $B_\alpha \cap B_\beta = \emptyset$ for $\alpha \neq \beta$ and $\cup\{B_\alpha\} = S$.

We refer to the sets of φ as *blocks* of φ and designate the block containing s by $B_\varphi(s)$. In writing out a partition, we distinguish blocks with bars and semicolons.

Example 1: If $S = \{1,2,3,4,5,6,7,8\}$ and partition φ on S has blocks {1,3,4,5}, {2,6}, and {7,8}, then we write $\varphi = \{\overline{1,3,4,5}; \overline{2,6}; \overline{7,8}\}$.

Note that φ is a set and has elements like any other set. For $\varphi_1$and φ_2 on S, we say that φ_2 is *larger than or equal to* φ_1, and write $\varphi_1 \leq \varphi_2$, if and only if every block of φ_1 is contained in a block of φ_2.

Example 2: If $\varphi_1 = \{\overline{1,2}; \overline{3,4}; \overline{5,6}; \overline{7,8}\}$ and $\varphi_2 = \{\overline{1,2,3,4}; \overline{5,6,7,8}\}$, then $\varphi_1 \leq \varphi_2$.

Definition 4: Two partitions $\varphi_1$and φ_2 are *equal*, $\varphi_1 = \varphi_2$, if and only if $\varphi_1 \leq \varphi_2$ and $\varphi_2 \leq \varphi_1$.

We write $s \equiv t(\varphi)$ if and only if s and t are contained in the same block of φ. That is, $s \equiv t(\varphi)$ if and only if $B_\varphi(s) = B_\varphi(t)$.

Definition 5: A partition φ on the set of states of $M = (S, I, O, \delta, \lambda)$ is a *closed* partition if and only if $s \equiv t(\varphi)$ implies that $\delta(s, a) \equiv \delta(t, a)(\varphi)$ for all a in I.

Now, we can define a "multiplication" operation (denoted by $\cdot$) on partitions on a set.

Definition 6: If φ_1 and φ_2 are partitions on S, then $\varphi_1 \cdot \varphi_2$ is the partition on S such that $s \equiv t(\varphi_1 \cdot \varphi_2)$ if and only if $s \equiv t(\varphi_1)$ and $s \equiv t(\varphi_2)$.

Example 3: If $\varphi_1 = \{\overline{1,2}; \overline{3,4}; \overline{5,6}; \overline{7,8,9}\}$ and $\varphi_2 = \{\overline{1,6}; \overline{2,3}; \overline{4,5}; \overline{7,8}; \overline{9}\}$, then $\varphi_1 \cdot \varphi_2 = \{\overline{1}; \overline{2}; \overline{3}; \overline{4}; \overline{5}; \overline{6}; \overline{7,8}; \overline{9}\}$.

Definition 7: If partition $\varphi_1$on S has the same singleton element as S, then the partition φ_1 is called a "*zero*" partition (denoted by $\emptyset$). If partition φ_2 on S has one block containing all elements in S, then the partition φ_2 is called an "*identity*" partition.

Definition 8: If partition φ on S is either "zero" or "identity" partition, then the partition φ is called a "*trivial*" partition. Otherwise, it is called a "*non-trivial*" partition.

Example 4: If $S = \{1,2,3,4,5,6,7,8\}$ and partition $\varphi_1 = \{\overline{1}; \overline{2}; \overline{3}; \overline{4}; \overline{5}; \overline{6}; \overline{7}; \overline{8}\}$, then $\varphi_1 = \emptyset$. If $S = \{1,2,3,4,5,6,7,8\}$ and partition $\varphi_2 = \{\overline{1,2,3,4,5,6,7,8}\}$, then φ_2 is an identity partition. Both φ_1 and φ_2 are "trivial." Both φ_1 and φ_2 in Example 3 are "non-trivial."

Definition 9: A partition $\varphi = \{\overline{B_1}; \overline{B_2}; \ldots; \overline{B_t}\}$ on S is an *input-consistent partition* if $\delta(\overline{B_i}, a) = \delta(\overline{B_i}, b)$ for *a*, *b* in *I*, and *i = 1,2,….t.*

That is, the state behavior of a component with an input-consistent partition is independent of input *I*.

Motivational Example

To illustrate the basic idea of the proposed method, we use the FSM *M* as shown in Fig. 1 [6]. Note that (*x/z*) indicates an input *x* and an output *z*, and " − " denotes a *don't-care* condition.

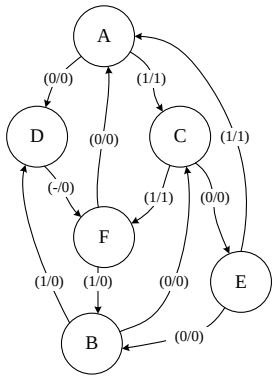


Fig. 1. An example FSM *M* [6].

The *M* has the six states {A, B, C, D, E, F}, input *x*, and output *z*. The state transition and the output function are described in the diagram. Fig. 2 shows the extraction of M_p from *M* where M_p shows a cycle of three states: α, β, γ. The successor M_s is to preserve the original functionality of *M*. This is similar to a series decomposition [6]. Note M_p possesses a cyclic state behavior. This can be viewed as extracting M_p in a given *M*. We propose to use the extracted M_p as the watermarked FSM.

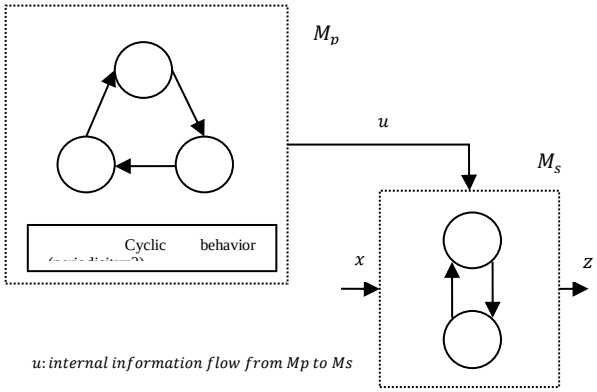


Fig. 2. Extraction of M_p from *M*.

Using the machine decomposition and state encoding methods (i.e., state assignment problem) [5][6], the following state encoding can extract M_p from M : e_1 = {(A, 000), (B, 001), (C, 010), (D, 011), (E, 100), (F, 101)} with {(α, 00), (β, 01), (γ, 10)} and {(a, 0), (b,1)}. Note that the original states are realized by two internal states: A = (α, a), B = (α, b), C = (β, a), D = (β, b), E = (γ, a), F = (γ, b). For instance, the state "A" can be realized by two internal states "α" and "a" using three flip flops.

METHOD FOR EMBEDDING WATERMARKS

State Encoding

The basic ideas lie in (1) performing a state encoding (e_i) and (2) extracting a watermarked FSM (FSM_w) possessing a property (p_j). The conceptual diagram is shown in Fig. 3.

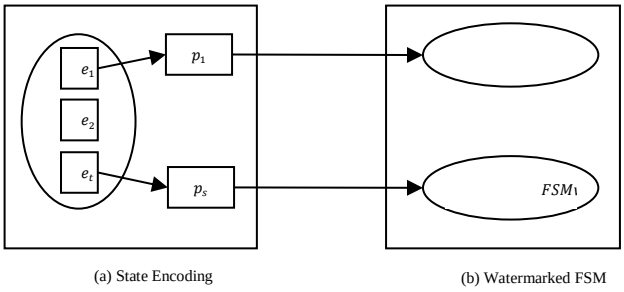


Fig. 3. Conceptual diagram of showing two main phases

For a given *n*-state FSM and *m*, the number of variables (or flip flops), there exists potentially many different state encodings. Let Ω be the set of all possible encodings: $\Omega = \{e_1, e_2, \ldots, e_t\}$. The encoding space Ω can be quite large. The number of flip flops, which is a variable (i.e., a watermark design parameter), makes a direct impact on the size of the encoding space (i.e., $|\Omega| = t$). One possible case is "$n = m$" and this is known as "hot encoding," where each state is implemented by a distinct flip flop. Another possibility is to use a "minimum" number of flip flops, which will guarantee the usage of minimum storage elements ($m = \lceil \log_2 n \rceil$). In general, $t = |\Omega| \neq s$, indicating that there is not always possible to produce a property p_i by every state encoding e_i.

In this work, we are fundamentally interested in exploring "non-destructive" solutions. Thus, we shall focus on using the minimum number of flip flops. *However, the basic idea of the proposed method is general and it can be extended for any value of "m", including "*$m \geq n$". Also, the particular property of interest, denoted by *P**, is one that has the cyclic state behavior with *maximal* periodicity.

Example 5 (n = 6, m = 3): For *M* in Fig. 1, using the minimum number of flip flops, $|\Omega| = t$ = 8 x 7 x 6 x 5 x 4 x 3 = 20,160, there are six states (A, B, C, D, E, F), three flip flops, and eight binary codes {000, 001, 010, 011, 100, 101, 110, 111} are available for the state encodings.

Hierarchical State Encoding: In a hierarchical encoding (h = level of hierarchy), multiple state encoding steps are applied. A two-tier state encoding (h = 2) is considered in this work. The basic idea can be expanded to a multi-tier state encoding, if needed. This is similar to the traditional state assignment [6].

$$e_i = e_i^1 + e_i^2 + \cdots + e_i^h = \bigcup_{j=1}^{h} e_i^j \qquad (1)$$

In a two-tier state encoding, $e_i = e_i^1 + e_i^2$. In the initial state encoding (e_i^1), the goal is to extract the property P*. Generally, in this step, a set of blocks of state is encoded. During the next step of state encoding (e_i^2), the number of states in each block is encoded. The final state encoding e_i is then the concatenation of e_i^1 and e_i^2.

Example 6 [Two-tier State Encoding]: For M in Fig. 1, $e_i^1 = \{(\alpha, 00), (\beta, 01), (\gamma, 10)\}$ and $e_i^2 = \{(a, 0), (b, 1)\}$. Then, $e_i = \{(A, 000), (B, 001), (C, 010), (D, 011), (E, 100), (F,101)\}$.

Watermarked FSM

A watermarked FSM (FSM_w) can be constructed as a state machine using e_i^1. For a given e_i^1, we can construct the corresponding state machine. The process for embedding (i.e., constructing) a watermarked FSM is described in Fig. 4. The input to the algorithm is an "n"-state FSM. The output is a watermarked FSM, FSM_w, which possesses the property P*. Note that FSM_w is a state machine with a triplet $FSM_w = (S_w, I_w, \delta_w)$.

Input: a "n"-state FSM = (S, I, O, δ, λ),$|S| = n$
Requirement: $m = \lceil \log_2 n \rceil$ /* minimum no. of binary codes */
Output: a watermarked $FSM_w = (S_w, I_w, \delta_w)$ /* state machine */

Procedure:
1. Find p_{max}, the *maximal* input-consistent periodicity
2. Construct a state machine $FSM_w = (S_w, I_w, \delta_w)$ as follow:
3. The set of states: $S_w = \{s_1^*, s_2^*, \ldots, s_{p_{max}}^*\}$
4. The input: $I_w = \Phi$
5. The state transition: $\delta_w(s_1^*, a) = s_2^*$, $\delta_w(s_2^*, a) = s_3^*$, $\delta_w(s_3^*, a) = s_4^*$,….., $\delta_w(s_{p_{max}}^*, a) = s_1^*$ for $\forall a \in I$

Fig. 4. Extracting a watermarked FSM (FSM_w).

The critical step is to find the maximal input-consistent periodicity (Step 1). The algorithm of finding p_{max} is shown in Fig. 5. The underlying idea of finding p_{max} is based on [6].

Input: a "n"-state FSM = (S, I, O, δ, λ),$|S| = n$
Output: p_{max} /* Maximal periodicity */

Procedure:
1. Find a set of a closed partition $\pi = \{\pi_1, \pi_2, \ldots . \pi_\varphi\}$ and a *nontrivial* input-consistent partition τ on S, where $\pi_i \geq \tau$, for $i = 1, 2, \ldots . \phi$.
2. Determine the *smallest* closed partition π_{min} from $\pi = \{\pi_1, \pi_2, \ldots . \pi_\varphi\}$.
3. Let $\pi_{min} = \{B_1; B_2; \ldots . ; B_q\} = \tau$.
4. The number of blocks in π_{min}, q, is the *maximal* periodicity p_{max}.

Fig. 5. Finding the maximal periodicity (p_{max})

The complexity of the algorithm for finding p_{max} is $O(n^2)$, a polynomial time, since a pair-wise state operation is required for the n states ($i.e., |S| = n$) in Step 1. Step 3 shows that the number of blocks in the smallest closed partition π_{min} is q (< n) and the nontrivial input-consistent partition τ can be set to π_{min} since this equality satisfies the condition $\pi_i \geq \tau$ in Step 1. Both determining the smallest closed partition in Step 2 and finding the number of blocks in π_{min} in Step 4 are straight forward and can be done in $O(n)$.

Example 7 [Maximal Periodicity]: For M in Fig. 1, the maximal periodicity is $p_{max} = 3$ since $\pi_{min} = \tau = \{\overline{A, B}; \overline{C, D}; \overline{E, F}\} = \{\alpha; \beta; \gamma\} = \pi_1$ and the number of blocks in π_{min} is 3. Note that there exists other candidates of input-consistent partitions, namely $\pi_2 = \{\overline{A, B, C, D}; \overline{E, F}\} = \{\overline{\alpha, \beta}; \bar{\gamma}\}$, $\pi_3 = \{\overline{A, B, E, F}; \overline{C, D}\} = \{\overline{\alpha, \gamma}; \bar{\beta}\}$, $\pi_4 = \{\overline{C, D, E, F}; \overline{A, B}\} = \{\overline{\beta, \gamma}; \bar{\alpha}\}$, and $\pi_5 = \{\overline{A, B, C, D, E, F}\} = \{\overline{\alpha, \beta, \gamma}\}$. However, π_5 is a *trivial* input-consistent

partition since it combines all states into a single block, which implies that no useful information is processed with this state partition. All other partitions above are nontrivial input-consistent partitions. However, the *smallest* closed partition is π_1.

Hierarchical (Two-tier) State Encoding: Upon determining the partition π_{min} and the maximal periodicity p_{max}, a state encoding e_i can be performed by assigning the minimum number of binary codes to the blocks of the smallest partition. In a tier-1 state encoding, the assignment with a minimum number of binary bits can be made on the blocks of state $\{B_1;\ B_2; \ldots..;\ B_q\}$ in π_{min}. This is described in Step 1 and Step 2 in Fig. 6. Pseudo-code of tier-1 state encoding (step 1 and 2) for the blocks in the smallest closed partition is associated with FSM_w. Pseudo-code of tier-2 state encoding (step 3-5) for the blocks of partition μ is associated with the residual FSM_r. Note that FSM_r is to preserve the original functionatility of *M*. Once the smallest closed partition ($\pi_{\min}$) and nontrivial input-consistent partition (τ) are determined, it is simple to perform the initial state encoding (e_i^1) using the minimum number of binary bits (in Step 1 and Step 2). The complexity of tier-1 state encoding procedure is $O(q)$. Note that Step 2 might not produce a unique state encoding.

Input: State partition $\pi_{min} = \tau = \{B_1;\ B_2; \ldots..;\ B_q\},\ q < n$.
Requirement: Use the minimum number of binary codes
Output: State encoding of (e_i^1) and (e_i^2)

Procedure:

1. Let $m_w = \lceil \log_2 q \rceil$
2. Perform one-to-one mapping using m_w *binary bits* on $\{B_1;\ B_2; \ldots..;\ B_q\}$.
3. Find another state partition μ such that $\pi_{min} \cdot \mu = \emptyset$; Let $\mu = \{b_1;\ b_2; \ldots.;\ b_r\}$.
4. Let $m_r = \lceil \log_2 r \rceil$
5. Perform one-to-one mapping using m_r *binary bits* on $\{b_1;\ b_2; \ldots.;\ b_r\}$.

Fig. 6. Pseudo-code of tier-1 and tier-2 state encoding

Example 8 [State Encoding; Tier-1]: From Example 7, $q = 3, m_w = 2, e_i^1 = \{(\alpha, 00), (\beta, 01), (\gamma, 10)\}$. Note that there exists other tier-1 state encodings possible such as $\{(\alpha, 01), (\beta, 10), (\gamma, 11)\}$.

To complete the state encoding e_i, a tier-2 state encoding e_i^2 needs to be done. Informally, e_i^2 should distinguish each state in every block B_i *in* $\{B_1;\ B_2; \ldots..;\ B_q\}$. This can be done with another partition $\mu = \{\overline{A, C, E};\ \overline{B, D, F}\}$, as an example.

A general procedure for a tier-2 state encoding e_i^2 is described in Step (3)-(5) in Fig. 6. Given $\pi_{min,}$ finding another partition μ satisfying the condition (i.e., $\pi_{min} \cdot \mu = \emptyset$) is straight forward since it can split the states in each block of π_{min} into an individual state. Step 3 can be done in polynomial time, $O(q \cdot |B_i|)$, where q is the number of blocks in π_{min} and $|B_i|$ is the maximum number of states in the block. Step 4 and 5 are similar to the tier-1 state encoding procedure.

Example 9 [State Encoding; Tier-2]: From Example 7 and Example 8, $\mu = \{\overline{A, C, E};\ \overline{B, D, F}\} = \{b_1;\ b_2\}$ since $\pi_{min} \cdot \mu = \emptyset$. Also, $r = 2, m_r = 1, e_i^2 = \{(b_1, 0), (b_2, 1)\}$.

The final state encoding e_i is made using the *concatenation* of two tiers of state encodings e_i^1 and e_i^2.

Example 10 [Final State Encoding]: $e_i = e_i^1 + e_i^2 = \{(A, 000), (B, 001), (C, 010), (D, 011), (E, 100), (F, 101)\}$.

Note that the watermarked FSM (FSM_w) is a state machine that behaves as an input-independent $\lceil log_2 q \rceil -$ bit counter.

ANALYSIS

In this section, we provide the analysis of the proposed approach. The limitation of the proposed approach and the potential solution are also discussed.

Existence of Watermarked FSM

We provide the analysis for the existence of the watermarked FSM possessing the property P* based on [6].

Theorem 1: If a closed partition π and a nontrivial input-consistent partition τ_i on *S* in a *FSM = (I, O, S, δ, λ)* which satisfy the condition of $\pi \geq \tau_i$, then the watermarked $FSM_w = (S_w, I_w, \delta_w)$ possessing the property P* exists.

Proof: If the FSM possesses a closed partition π such that $\pi \geq \tau_i$, then, for a given state S_i and every input in *I*, the next states must be in the same block of τ_i, and therefore in the same block of π. Consequently, for a given initial state, the block of π in which the state of FSM_w is contained after any finite input sequence depends only on the initial block and on the length of the sequence. Furthermore, there must exist a cycle of states such that $\delta_w(s_1, x) = \delta_w(s_2, x) = \ldots = \delta_w(s_p, x)$ for any input *x* in I_w. Then, the maximal periodicity ($p_{max} = p$) of a cycle can be chosen.

□

Attack Analysis

IP watermarking attacks can be further categorized in the main classes below [3].

Removal Attacks: Removal attacks are divided into either elimination attacks or masking attacks.
Elimination attacks: the watermark can be eliminated completely by an attacker. For instance, the attacker tries to estimate the watermark and subtract it from the watermarked design. In the proposed scheme, it will be technically infeasible to eliminate the watermarked FSM without affecting the original functionality since the watermarked FSM itself performs the sub-computation.

Masking attacks: masking attacks aim at distorting the watermark detector to disable its ability to sense the presence of the watermark. This attack is more related to the watermark verification process.

False Positive (Probability of Coincidence or Watermark Collision): In [12], the probability of coincidence was defined as "the odds that an unintended watermark is detected in a design." Often, this is used as a measure of watermark validity. In [9], we demonstrated using a set of simple FSMs that the collisions practically may not occur.

Embedding Attacks (Forging): Watermark designers need to find techniques to protect their designs from intruders to embed another watermark in the design. In the proposed scheme, it is likely that the functionality of a FSM may change and thus detectable, if another watermark is added.

Limitation

The existence of *P** is not guaranteed for *all* FSMs (see Theorem 1). The nature of the limitation is rooted from using a strong property (i.e., maximal periodicity and input-consistency). In this case, we may consider relaxing these requirements at the cost of lowing security level (i.e., to embed *less* useful data for IP protection.)

CONCLUSIONS AND FUTURE WORK

For a class of deterministic finite-state machines (D-FSMs), we have proposed the new watermarking embedding method at a FSM-level. We showed that a watermarked FSM, which possesses a property of cyclic state behavior with maximal periodicity, can be extracted using a hierarchical state encoding. The proposed work was carried out based on the observation that this property may be used as an IP owner's unique property for the FSM IP. However, the proposed method does not provide a universal solution (Section IV.C). Using state-splitting [6] may resolve this issue, but at the cost of relaxing the non-destructiveness.

Another interesting problem that should be investigated further is to formally analyze a false positive collision that may arise during the embedding process. This problem can be described as follows: *Given a* FSM *M which inherently possesses the property of a cyclic state behavior with periodicity p, does another FSM M′exist such that M′ has the same property and M′ ≠ M?* We plan to work on this important problem.

ACKNOWLEDGMENT

The author is indebted to Prof. Lilian Bossuet for his insightful, constructive comments, and many thought-provoking discussions. Also, the author would like to thank the European academic community on embedded security and IP protection for providing the opportunity to present this work.

REFERENCES

[1] Abdel-Hamid, A. T., Tahar, S., and Aboulhamid, E. M. A survey on IP watermarking techniques. In *Design Automation for Embedded Systems*, Vol. 9, (2004), Springer Science+Business Media, Berlin, 211-227.

[2] Bossuet, L. and Hely, D. Salware: Salutary hardware to design trusted IC. In *Proc. of the Trustworthy Manufacturing and Utilization of Secure Devices Workshop* (TRUDEVICE '13). Avignon, France, (2013), 30-31.

[3] Cox, I. J., Miller, M. L., Bloom, J. A., and Honsinger, C. *Digital watermarking*. (1998), Morgan Kaufmann Publishers.

[4] Cui, A., Chang, C.H, Tahar, S., and Abdel-Hamid, A.T.. A Robust FSM Watermarking Scheme for IP Protection of Sequential Circuit Design. *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.* 30, 5 (May 2011), 678-690.

[5] Hartmanis, J. and Stearns, R.E. *Algebraic Structure Theory of Sequential Machines*, Prentice-Hall, Inc., (1966), Upper Saddle River, NJ, USA.

[6] Kohavi, Z. *Switching and Finite Automata Theory* (2nd ed.). (1978), McGraw-Hill.

[7] Koushanfar, F. and Alkabani, Y. Provably secure obfuscation of diverse watermarks for sequential circuits. In *Proc. of the Inter. Symp. on Hardware-Oriented Security and Trust* (HOST '10), (2010), 42-47.

[8] Lewandowski, M., Meana, R., Morrison, M., and Katkoori, S. 2012. A Novel Method for Watermarking Sequential Circuits. In *Proc. of the IEEE Int. Symp. on Hardware-Oriented Sec. and Trust*, San Francisco, CA. (2012), 21-24.

[9] Marchand, C., Bossuet, L., and Jung, E., "IP Watermarking Verification Based On Power Consumption Analysis," *Proc. of the 27th IEEE Int. Sys.-on-Chip Conf. (IEEE SOCC '14)*, Las Vegas, September 2014.

[10] Oliveira, A. L. Robust techniques for watermarking sequential circuit designs. In *Proc. of the 36th annual ACM/IEEE Design Automation Conf.* (DAC '99), Mary Jane Irwin (Ed.). ACM, New York, NY, USA, (1999), 837-842.

[11] Oliveira, A. L. Techniques for the creation of digital watermarks in sequential circuit design. *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.* Vol. 20, Issue 9, (September 2001), 1101-1117.

[12] Torunoglu, I. and Charbon, E. Watermarking-based copyright protection of sequential functions. *IEEE Journal of Solid-State Circuits.* Vol. 35, Issue 3 (2000), 434-440.

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

J Netw Syst Manage
DOI 10.1007/s10922-014-9335-3

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

Soo-Yeon Ji · Seonho Choi · Dong Hyun Jeong

Received: 30 August 2013 / Revised: 9 September 2014 / Accepted: 22 September 2014

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

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

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

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

Published online: 30 September 2014

Keywords Internet traffic detection · Discrete wavelet transformation · Logistic regression · Area Under ROC Curve (AUC)

1 Introduction

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

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

Imbalanced data often occurs in various domains including medical, biology, and computer networks [15]. In the network security community, network features or

Keywords Internet traffic detection · Discrete wavelet transformation · Logistic regression · Area Under ROC Curve (AUC)

1 Introduction

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

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

Imbalanced data often occurs in various domains including medical, biology, and computer networks [15]. In the network security community, network features or

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

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

2 Related Work

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

Wavelet analysis is also widely used for internet traffic analysis because of its ability of identifying hidden patterns from time-frequency information by separating input data into different levels of frequencies. Applying the signal processing technique (i.e. wavelet analysis) to internet traffic helps us isolate the characteristics of the traffic by extracting hidden patterns of high and low frequency information [26]. Many researchers used wavelet analysis to identify network anomalies by reconstructing network traffic data [27, 28], compressing the data by applying two different thresholds from wavelet coefficients [29], and designing better wavelet

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

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

3 Approach

Our approach begins with applying normalization to the input data attributes as a pre-processing step. Then, discrete wavelet transformation is used to extract features. Since it is important to perform a feature validation to enhance the performance of identifying hidden abnormal behaviors, statistical analysis (ANOVA—analysis of variance) is applied to identify statistically significant features that are later used to generate a predictive model. To determine the effectiveness of the generated predictive model, the reliability of the model is tested by calculating the AUC. Figure 1 represents the overall procedure of our proposed prediction model. A detailed explanation about our method is included in Sect. 3.2.

Table 1 A summary of research reviewed in Sect. 2

Work	Goal	Dataset(s)	Used features	Major step(s)	Used algorithm(s)
Alarcon-Aquino and Barria [26]	Network anomaly detection	(1) Simulated data (2) BTnet Dial IP service data	(1) Simulated data features and (2) set of network metrics	Estimation of Nuisance parameters by maximum likelihood using wavelet coefficients	Undecimated DWT and Bayesian
Kyriakopoulos and Parish [29]	Network anomaly detection	CAIDA dataset on the Witty worm	Not known	Two thresholds for the compression and event detection tasks	Haar wavelet, adaptive threshold, and Donoho–Johnstone universal threshold (aka VisuShrink)
Barford et al.[30]	Network anomaly detection	Simulated data	Decomposed three distinct signals (low/mid/high) from SNMP and IP flow dataset	Deviation scores from three range (low, mid, and high) frequency information	Wavelet analysis and Time frequency-localization
Dainotti et al. [35]	Network anomaly detection	(1) DARPA99 (2) D-WARD (3) UNINA	Traffic traces (packet rates) and mean and std for the traces)	Generate anomaly profiles	Adaptive threshold, cumulative sum, and Continuous wavelet transform
Lu and Ghorbani [28]	Network anomaly detection	DARPA99	15 Features from TCP, UDP, ICMP data	ARX model using wavelet approximation coefficiets	DWT and AutoRegressive with eXogenous (ARX)
Kim et al. [27]	Network anomaly detection	(1) USC traces (2) Simulated virutal attacks on the University of Auckland traces	Duration, persistency, IP address	Coefficient-selective reconstruction in DWT	DWT and Threshold
Callegari et al. [31]	Network anomaly detection	(1) DARPA99 (2) UNINA (3) D-WARD	Number of IP packets received	Euclidean distance between coefficients	Wavelet Packet Transform (WPT)
Gao et al. [32]	Network anomaly detection	Simulated dataset	Data packets	A scale-adaptive method	WPT

Table 1 continued

Work	Goal	Dataset(s)	Used features	Major step(s)	Used algorithm(s)
Tan et al. [33]	Classifying network applications	Simulated dataset	69 Calculated features from TCP and UDP packets	Back-propagation (BP) neural network Particle and swarm optimization	Wavelet packet decomposition (WPD)
Ramanathan [34]	Detection of DoS attack	NLANR data	IP address, Packet size, Time stamp	Thresholding and wavelet variance computation	DWT

DARPA99: 1999 DARPA intrusion detection dataset

UNINA: University of Naples "Federico II" traffic trace dataset

D-WARD: UCLA Packet trace dataset

Only the researches utilizing wavelet transformations are presented

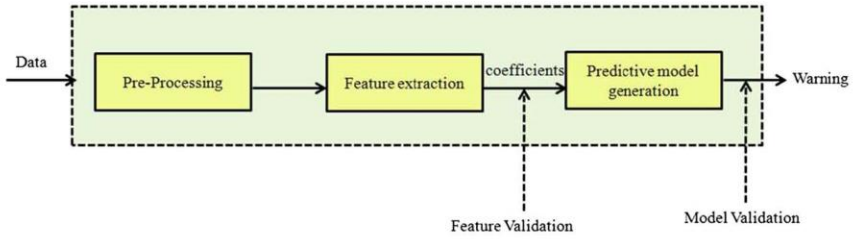


Fig. 1 A schematic diagram of generating a predictive model

3.1 Dataset

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

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

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

Table 2 A description of used normal and abnormal datasets

Class	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5
Normal					
FTP	210	210	209	209	210
Mail	209	209	210	210	209
P2P	209	209	209	209	209
Total	628	628	628	628	628
Abnormal					
Attack	628	628	628	628	628

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

Table 3 Six inter-arrival time (IAT) features for all packets (considering both directions) are used in this study

Feature name	Description
(f1) min IAT	Minimum packet inter-arrival time
(f2) q1 IAT	First quartile inter-arrival time
(f3) med IAT	Median inter-arrival time
(f4) mean IAT	Mean inter-arrival time
(f5) max IAT	Maximum packet inter-arrival time
(f6) var IAT	Variance in packet inter-arrival time

3.2 Method

3.2.1 Pre-processing

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

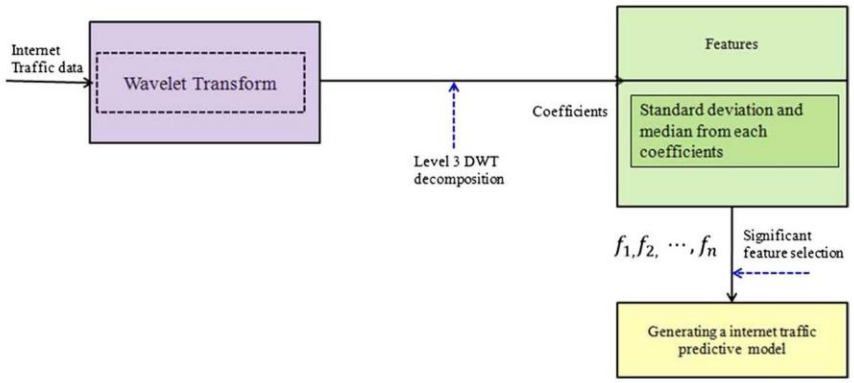


Fig. 2 A representation of feature extraction process

3.2.2 Feature Extraction

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

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

Among the extracted DWT features, feature selection is performed by examining the significance of all features. To validate their significance, a statistical analysis is performed using the SAS. In particular, one-way ANOVA is performed to identify

important features using two classes (Normal vs Abnormal) by maintaining the statistical significance ($p < .05$). From the analysis, only significant features are selected and used to generate a predictive model.

3.2.3 Generating a Predictive Model Using LR and Its Validation

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

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

where X_i is a variable with numeric value, π_i is the outcome (dichotomous; 0/1, e.g., normal/ abnormal), and the β_is are the regression coefficients that quantify the contributions of the numeric variables to the overall probability [41–43]. Unlike most regression analyses, LR does not need to assume data distributions on variables. Due to this benefit, it is commonly used when outcome is a nominal variable. To test the reliability of the predictive model, the AUC is computed. In addition, an evaluation is conducted to see the effectiveness of using the DWT features versus the raw features for generating a predictive model.

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

4 Results

This section presents the findings obtained from our study. In Sect. 4.1, we show a comparison between the raw and the DWT features by emphasizing the effectiveness of utilizing the DWT features. Since it is important to determine the usefulness of our proposed model for detecting abnormal behaviors, a performance

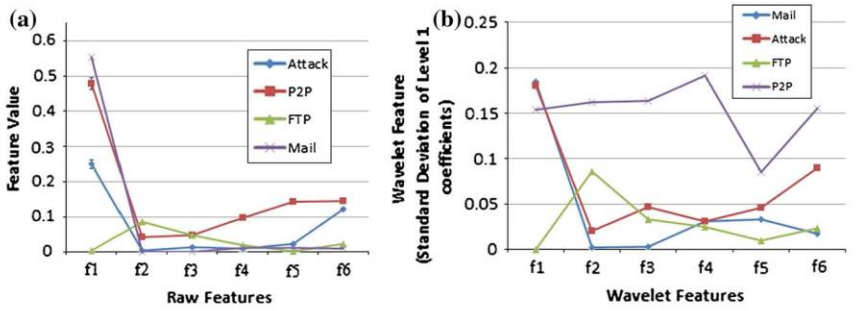


Fig. 3 A comparison between the raw and the DWT features. Average values of **a** the raw features and **b** the standard deviation of the level one wavelet coefficient are represented with identifying abnormal (Attack) and normal (P2P, FTP and Mail)

comparison is conducted by calculating accuracy, sensitivity, specificity, the ROC curve, and the AUC (see Sect. 4.2 for detail).

4.1 Feature Comparison

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

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

4.2 Classification Performance Comparison

As explained above, our proposed predictive model was designed with LR. Since the DWT features were used as inputs to LR, a classification performance comparison was conducted by generating a predictive model with the raw features. A comparison between the two models was done by computing accuracy,

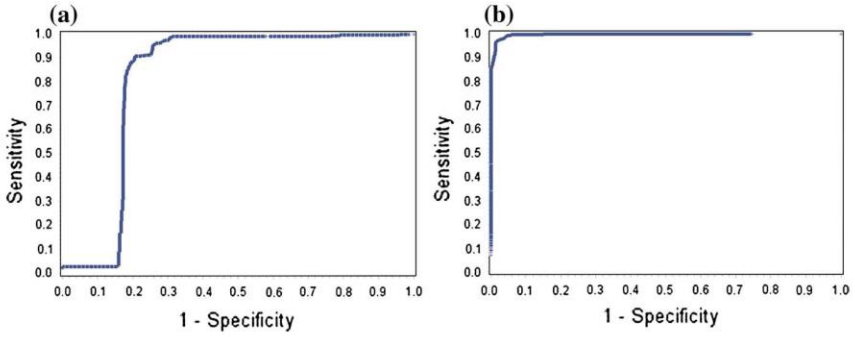


Fig. 4 ROC plots for LR using **a** the raw and **b** the DWT features. The size of 25 data points is used to the overlapping sliding window

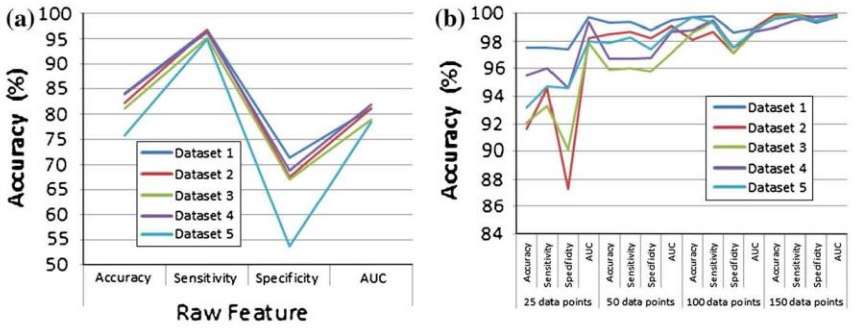


Fig. 5 A performance evaluation is conducted with **a** the raw and **b** the DWT features in different window sizes (25–150 data points)

sensitivity, specificity, and the AUCs. As an additional performance comparison, our proposed predictive model was compared to other known approaches, such as the Neural Network and SVM.

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

Performance comparison—A performance was evaluated by computing four statistical measures (accuracy, sensitivity, specificity, and the AUC). Figure 5 represents a performance comparison between the predictive model generated with the raw features (Fig. 5a) and the DWT features (Fig. 5b). The predictive model

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

	The DWT features				The raw features
window size (i.e. data points)	25	50	100	150	–
Accuracy	1.107	0.611	0.315	0.150	1.529
Sensitivity	0.712	0.635	0.181	0.073	0.408
Specificity	1.810	0.525	0.275	0.087	3.086
AUC	0.377	0.409	0.044	0.040	0.692

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

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

Each value indicates mean ± standard error mean (SEM)

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

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

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

Since it is important to perform a comparison with other approaches to determine the effectiveness of our proposed model, we conducted a study with broadly known techniques, such as Neural Network (NN) and SVM. Specifically, we measured accuracy, sensitivity, and AUC (see Table 5). From the performance comparison,

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

	Type I error (FP)	Type II error (FN)
Imbalanced	0.4878 ± 0.1329	0.0358 ± 0.0128
Balanced	0.0456 ± 0.0094	0.0830 ± 0.0130

we identified that the accuracy, sensitivity, and AUC of our method were higher compared to other approaches (NN and SVM). This explains that our predictive model (via LR) is good for identifying abnormal behaviors more accurately. In addition, we also found that utilization of the DWT features is important for increasing the performance of detecting anomalous network traffic or behaviors.

5 Discussion and Conclusion

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

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

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

frequency information as well as their specific local information in time, it is good for identifying abnormal internet traffic behaviors (see Fig. 5). To validate the reliability of the models generated with the raw and the DWT features, the AUC is calculated. From the AUC results, we identified that the model with the DWT features is much more reliable than the model with the raw features. This explains that the DWT features well preserve important characteristics of abnormal internet traffic behaviors. It is important to note that the performance difference when utilizing the raw and the DWT features could be attributed to the difference of the amount of information taken as input to generate a predictive model.

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

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

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

References

1. Han, J., Kamber, M.: Data mining: concepts and techniques. The Morgan Kaufmann Series in Data Management Systems. Elsevier Science (2011)
2. Madhukar, A., Williamson, C.: A longitudinal study of p2p traffic classification. In: Modeling, analysis, and simulation of computer and telecommunication systems, 2006. MASCOTS 2006. 14th IEEE International Symposium on, pp. 179–188 (2006). doi:10.1109/MASCOTS.2006.6

3. Dashevskiy, M., Luo, Z.: Reliable probabilistic classification and its application to internet traffic. In: Huang, II, D.S., Levine, D.C.W., Levine, D.S., Jo, K.H. (eds.) ICIC (1), Lecture notes in computer science, **5226**, pp. 380–388. Springer (2008)
4. Kim, J.T., Park, H.K., Paik, E.H.: Security issues in peer-to-peer systems. In: Advanced communication technology, 2005, ICACT 2005. The 7th International Conference on, vol. 2, 1059–1063 (2005). doi:10.1109/ICACT.2005.246141
5. Sen, S., Spatscheck, O., Wang, D.: Accurate, scalable in-network identification of p2p traffic using application signatures. In: Proceedings of the 13th International Conference on World Wide Web. WWW '04, pp. 512–521. ACM, New York, NY, USA (2004)
6. Raahemi, B., Zhong, W., Liu, J.: Peer-to-peer traffic identification by mining ip layer data streams using concept-adapting very fast decision tree. In: Tools with artificial intelligence, 2008. ICTAI '08. 20th IEEE International Conference on, vol. 1, pp. 525–532 (2008)
7. Moore, A., Papagiannaki, K.: Toward the accurate identification of network applications. In: Dovrolis, C. (ed.) Passive and active network measurement, lecture notes in computer science, vol. 3431, pp. 41–54. Springer, Berlin (2005)
8. Kushida, T., Shibata, Y.: Empirical study of inter-arrival packet times and packet losses. In: Distributed computing systems workshops, 2002. In: Proceedings. 22nd international conference on, pp. 233–238 (2002). doi:10.1109/ICDCSW.2002.1030775
9. Li, W., Canini, M., Moore, A.W., Bolla, R.: Efficient application identification and the temporal and spatial stability of classification schema. Elsevier Computer Network (2009)
10. Karagiannis, T., Broido, A., Faloutsos, M., claffy, K.: Transport layer identification of p2p traffic. In: Proceedings of the 4th ACM SIGCOMM conference on internet measurement, IMC '04, pp. 121–134. ACM, New York, NY, USA (2004). doi:10.1145/1028788.1028804
11. Xu, K., Zhang, M., Ye, M., Chiu, D.M., Wu, J.: Identify p2p traffic by inspecting data transfer behavior. Comput. Commun. **33**(10), 1141–1150 (2010)
12. Holanda Filho, R., Fontenelle do Carmo, M., Maia, J., Siqueira, G.: An internet traffic classification methodology based on statistical discriminators. In: Network operations and management symposium, 2008. NOMS 2008. IEEE, pp. 907–910 (2008). doi:10.1109/NOMS.2008.4575244
13. Williams, N., Zander, S., Armitage, G.: A preliminary performance comparison of five machine learning algorithms for practical ip traffic flow classification. SIGCOMM Comput. Commun. Rev. **36**(5), 5–16 (2006)
14. Lu, X., Duan, H., Li, X.: Identification of p2p traffic based on the content redistribution characteristic. In: Communications and information technologies, 2007. ISCIT '07. International symposium on, pp. 596–601 (2007). doi:10.1109/ISCIT.2007.4392088
15. He, H., Ma, Y.: Imbalanced learning: foundations, algorithms, and applications, 1st edn. Wiley-IEEE Press, London (2013)
16. Maloof, M.A.: Learning when data sets are imbalanced and when costs are unequal and unknown. In: ICML-2003 workshop on learning from imbalanced data sets II (2003)
17. Bhuyan, M., Bhattacharyya, D., Kalita, J.: Network anomaly detection: methods, systems and tools. Commun. Surv. Tutor. IEEE **16**(1), 303–336 (2014). doi:10.1109/SURV.2013.052213.00046
18. Estevez-Tapiador, J.M., Garcia-Teodoro, P., Diaz-Verdejo, J.E.: Anomaly detection methods in wired networks: a survey and taxonomy. Comput. Commun. **27**(16), 1569–1584 (2004). doi:10.1016/j.comcom.2004.07.002
19. Este, A., Gringoli, F., Salgarelli, L.: Support vector machines for tcp traffic classification. Comput. Netw. **53**(14), 2476–2490 (2009). doi:10.1016/j.comnet.2009.05.003
20. Li, Z., Yuan, R., Guan, X.: Accurate classification of the internet traffic based on the svm method. In: Communications, 2007. ICC '07. IEEE international conference on, pp. 1373–1378 (2007). doi:10.1109/ICC.2007.231
21. Huang, S.Y., Huang, Y.N.: Network traffic anomaly detection based on growing hierarchical som. In: 2013 43rd annual IEEE/IFIP international conference on dependable systems and networks (DSN) 0, 1–2 (2013)
22. Hoz Franco, E., Ortiz Garcia, A., Ortega Lopera, J., Hoz Correa, E., Prieto Espinosa, A.: Network anomaly detection with bayesian self-organizing maps. Advances in computational intelligence, lecture notes in computer science, vol. 7902, pp. 530–537. Springer, Berlin (2013)
23. Auld, T., Moore, A., Gull, S.: Bayesian neural networks for internet traffic classification. Neural Netw. IEEE Trans. **18**(1), 223–239 (2007)

24. Sun, R., Yang, B., Peng, L., Chen, Y., Zhang, L., Jing, S.: Traffic classification using probabilistic neural networks. In: Natural computation (ICNC), 2010 sixth international conference on, vol. 4, pp. 1914–1919 (2010)
25. Moore, A.W., Zuev, D.: Internet traffic classification using bayesian analysis techniques. SIGMETRICS Perform. Eval. Rev. **33**(1), 50–60 (2005)
26. Alarcon-Aquino, V., Barria, J.: Anomaly detection in communication networks using wavelets. Commun. IEE Proc. **148**(6), 355–362 (2001)
27. Kim, S., Reddy, A., Vannucci, M.: Detecting traffic anomalies using discrete wavelet transform. In: Kahng, H.K., Goto, S. (eds.) Information networking. Networking technologies for broadband and mobile networks. Lecture notes in computer science, vol. 3090, pp. 951–961. Springer, Berlin (2004)
28. Lu, W., Ghorbani, A.A.: Network anomaly detection based on wavelet analysis. EURASIP J. Adv. Signal Process, pp. 1–16 (2009). Hindawi Publishing Corporation, New York (2008)
29. Kyriakopoulos, K., Parish, D.: Using wavelets for compression and detecting events in anomalous network traffic. In: Systems and networks communications, 2009. ICSNC '09. Fourth international conference on, pp. 195–200 (2009)
30. Barford, P., Kline, J., Plonka, D., Ron, A.: A signal analysis of network traffic anomalies. In: Proceedings of the 2nd ACM SIGCOMM workshop on internet measurment. IMW '02, pp. 71–82. ACM, New York (2002)
31. Callegari, C., Giordano, S., Pagano, M.: Application of wavelet packet transform to network anomaly detection. In: Balandin, S., Moltchanov, D., Koucheryavy, Y. (eds.) Next generation teletraffic and wired/wireless advanced networking. Lecture notes in computer science, vol. 5174, pp. 246–257. Springer, Berlin (2008)
32. Gao, J., Hu, G., Yao, X., Chang, R.: Anomaly detection of network traffic based on wavelet packet. In: Communications, 2006. APCC '06. Asia-Pacific conference on, pp. 1–5 (2006)
33. Tan, J., Chen, Xs, Du, M., Zhu, K.: A novel internet traffic identification approach using wavelet packet decomposition and neural network. J. Cent. South Univ. **19**(8), 2218–2230 (2012). doi:10.1007/s11771-012-1266-0
34. Ramanathan, A.: WADeS: a tool for distributed denial of service attack detection. Texas A&M University, Texas (2002)
35. Dainotti, A., Pescape, A., Ventre, G.: Nis04-1: Wavelet-based detection of dos attacks. In: Global telecommunications conference, 2006. GLOBECOM '06. IEEE, pp. 1–6 (2006). doi:10.1109/GLOCOM.2006.279
36. Moore, A., Crogan, M., Moore, A.W., Mary, Q., Zuev, D., Zuev, D., Crogan, M.L.: Discriminators for use in flow-based classification. Tech. rep. (2005)
37. He, H., Garcia, E.A.: Learning from imbalanced data. IEEE Trans. Knowl. Data Eng. **21**(9), 1263–1284 (2009). doi:10.1109/TKDE.2008.239
38. Wang, W., Zhang, X., Gombault, S., Knapskog, S.: Attribute normalization in network intrusion detection. In: Pervasive systems, algorithms, and networks (ISPAN), 2009 10th international symposium on, pp. 448–453 (2009)
39. Unser, M., Aldroubi, A.: A review of wavelets in biomedical applications. Proc. IEEE **84**(4), 626–638 (1996)
40. Meyer, Y., Ryan, R.: Wavelets: Algorithms and applications. Miscellaneous Bks. Soc. Ind. Appl. Math. (1993)
41. Hasford, J., Ansari, H., Lehmann, K.: Cart and logistic regression analyses of risk factors for first dose hypotension by an ace-inhibitor. Therapie **48**(5), 479–482 (1993)
42. Kuhnert, P.M., Do, K.A., McClure, R.: Combining non-parametric models with logistic regression: an application to motor vehicle injury data. Comput. Stat. Data Anal. **34**(3), 371–386 (2000)
43. Long, W.J., Griffith, J.L., Selker, H.P., D'agostino, R.B.: A comparison of logistic regression to decision-tree induction in a medical domain. Comput. Biomed. Res. 74–97 (1993)
44. Stone, M.: Cross-validatory choice and assessment of statistical predictions. R. Stat. Soc. **36**, 111–147 (1974)
45. Cawley, G.C., Talbot, N.L.: Efficient leave-one-out cross-validation of kernel fisher discriminant classifiers. Pattern Recognit. **36**(11), 2585–2592 (2003). doi:10.1016/S0031-3203(03)00136-5
46. Chapelle, O., Vapnik, V., Bousquet, O., Mukherjee, S.: Choosing multiple parameters for support vector machines. Mach. Learn. **46**(1–3), 131–159 (2002). doi:10.1023/A:1012450327387
47. Vapnik, V., Chapelle, O.: Bounds on error expectation for support vector machines. Neural Comput. **12**(9), 2013–2036 (2000)

48. Bradley, A.P.: The use of the area under the roc curve in the evaluation of machine learning algorithms. Pattern Recognit. **30**(7), 1145–1159 (1997). doi:10.1016/S0031-3203(96)00142-2
49. King, G., Zeng, L.: Logistic regression in rare events data. Polit. Anal. **9**, 137–163 (2001)
50. Menardi, G., Torelli, N.: Training and assessing classification rules with imbalanced data. Data Min. Knowl. Discov. **28**(1), 92–122 (2014). doi:10.1007/s10618-012-0295-5

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

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

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

8. A Multi-level Intrusion Detection Method for Abnormal Network Behaviors

A Multi-level Intrusion Detection Method for Abnormal Network Behaviors

Soo-Yeon Ji[a,*], Bong-Keun Jeong[b], Seonho Choi[a], Dong Hyun Jeong[c,**]

[a]*Dept. of Computer Science, Bowie State University, 14000 Jericho Park Road, Bowie, MD 20715, USA*
[b]*Dept. of Computer Information Systems, Metropolitan State University of Denver, CO 80217, USA*
[c]*Dept. of Computer Science and Information Technology, University of the District of Columbia, 4200 Connecticut Avenue NW, Washington, DC 20008, USA*

Abstract

Abnormal network traffic analysis has become an increasingly important research topic to protect computing infrastructures from intruders. Yet, it is challenging to accurately discover threats due to the high volume of network traffic. To have better knowledge about network intrusions, this paper focuses on designing a multi-level network detection method. Mainly, it is composed of three steps as 1) understanding hidden underlying patterns from network traffic data by creating reliable rules to identify network abnormality, 2) generating a predictive model to determine exact attack categories, and 3) integrating a visual analytics tool to conduct an interactive visual analysis and validate the identified intrusions with transparent reasons.

To verify our approach, a broadly known intrusion dataset (i.e. NSL-KDD) is used. We found that the generated rules maintain high performance rate and provide clear explanations. The proposed predictive model resulted about 96% of accuracy in detecting exact attack categories. With the interactive visual analysis, a significant difference among attack categories was discovered by visually representing attacks in separated clusters. Overall, our multi-level detection method is well-suited for identifying hidden underlying patterns and attack categories by revealing the relationship among the features of network traffic data.

Keywords: Network Traffic Analysis, Discrete Wavelet Transform, Visual Analytics, Support Vector Machine

1. Introduction

Due to the advancement of Internet technologies, applications, and protocols, network traffic analysis has become more difficult since it deals with extreme amount of network traffic data. Because of the network complexity, network traffic analysis to detect unauthorized network intruders is also considered as one of the increasingly important research topics in network security.

To address the issue of protecting computing infrastructures by detecting network intruders, numerous intrusion detection (ID) techniques have been proposed. A traditionally known ID

*Principal Corresponding author
**Corresponding author
Email addresses: sji@bowiestate.edu (Soo-Yeon Ji), bjeong@msudenver.edu (Bong-Keun Jeong), schoi@bowiestate.edu (Seonho Choi), djeong@udc.edu (Dong Hyun Jeong)

system discovers threats by analyzing traffic data at the network layer. The intrusion detection system (called host-based IDS) identifies threats on computer hosts by monitoring computer system logs, system calls, network events, and files (Das and Sarkar, 2014). To detect any abnormal behaviors, it monitors network packets to find possible attack signatures and compare them to known attack patterns. Although the host-based IDS is designed to prevent intruders by changing computer system security policies, it cannot monitor network traffic effectively because it only detects intrusions based on the analysis of information such as logs or packets (Bace, 1999). The system may detect threats based on known attack signatures, but new attacks cannot be discovered (Rubin et al., 2004).

Most analysis approaches are designed to detect intrusions by conducting misuse detection and anomaly detection. The misuse detection searches for events (i.e. known attacks) that are matched to predefined signatures (Kumar and Spafford, 1994). The anomaly detection identifies abnormal behaviors on hosts or networks based on the assumption that each attack shows different behaviors compared to normal activity. Therefore, it is possible to identify any abnormal attacks without having specific knowledge. Due to this advantage, the anomaly detection is used for designing various applications in other areas such as credit cards fraud detection (Kou et al., 2004), fault detection in safety critical systems (Worden and Dulieu-Barton, 2004), and any domains that aim to detect abnormal activities including a medical field (Duftschmid and Miksch, 2001). However, the anomaly detection method may provide a high false alarm rate, and require extensive training sets to achieve a reliable performance result (Chandola et al., 2009; Eskin et al., 2002).

Abnormal behaviors are considered as different representations if they do not match to a well-defined model representing normal behaviors. To discover abnormal behaviors (i.e. intrusions or attacks), understanding their trends or patterns is essential. ID can help us to minimize further damages by providing early warnings. In this paper, we extended our two previous studies by focusing on 1) generating simple and reliable rules to identify intrusions, 2) building a predictive model to determine exact attack categories by utilizing a signal processing technique (i.e. DWT) and Support Vector Machine (SVM), and 3) visually representing the input data to support an interactive visual analysis. For the visual analysis, a visual analytics tool called iPCA (Jeong et al., 2009) was used. With this tool, an interactive visual analysis was conducted to understand the intrusions and their relationships.

The rest of this paper begins with explaining related work in Section 2, our approach including a description of the data (i.e. NSL-KDD) and methods in Section 3. Study results are provided in Section 4. Lastly, Section 5 presents implications of this study and avenue for future research.

2. Related Work

Researchers have applied various algorithms or theories such as statistics, machine learning, data mining, information theory, and spectral theory to extract patterns from attacks and design better anomaly detection techniques. Machine Learning (ML) is one of the broadly used algorithms in anomaly detection. ML techniques develop classifiers to determine possible attacks. Markou and Singh (2003a,b) proposed a detection technique with utilizing neural networks and statistical approaches. Rule-based anomaly detection techniques are introduced to capture rules that can identify network behaviors using Fuzzy (Chadha and Jain, 2015; Amini et al., 2015) or decision trees (Lee et al., 2008; Kruegel and Toth, 2003; Stein et al., 2005; Jain and Abouzakhar, 2013). Also, clustering technique (Lin et al., 2015) and SVM (Kuang et al., 2015; Wang et al., 2015; Aslahi-Shahri et al., 2015; Sani and Ghasemi, 2015) are used by numerous researchers to

detect abnormal network behaviors. For instance, Xiang et al. (2008) introduced a multiple-level hybrid classifiers combining tree classifiers and Bayesian clustering to detect network anomaly. Kuang et al. (2015) presented a hybrid classifier by integrating SVM and principal component analysis. Golmah (2014) proposed an hybrid intrusion detection method integrating both C5.0 and SVM.

To generate a reliable ID system model, feature selection and extraction are considered as critical tasks for saving computational cost as well as for discovering data patterns. The feature selection is used to select a subset of most meaningful features from the original feature. The feature extraction is necessary for converting input data to reduce dimensions. There are various techniques that can be used for the feature extraction and selection such as Genetic Algorithm (GA) (Aslahi-Shahri et al., 2015), entropy of network features (Agarwal and Mittal, 2012), Partial Least Square (PLS) (Gan et al., 2013), Kernel Principal Component Analysis (KPCA) (Kuang et al., 2015), and cuttlefish optimization algorithm (Eesa et al., 2015). When applying the feature extraction, there is an important consideration whether the characteristics of original input data are transmitted to extracted new feature sets. However, it is important to note that the generated new feature set may not maintain the same or similar patterns compared to the original input data (Yang et al., 2011). Sanei et al. (2015) addressed the potential capability of discovering important features from input data by utilizing signal processing techniques. In our previous studies (Ji et al., 2014a,b), we emphasized the importance of detecting network abnormal behaviors. More specifically, in the study (Ji et al., 2014a), two-level ID method is introduced using a publicly available internet traffic data to show its capability in classifying abnormal network traffic. Fractal dimension (FD) was applied to identify the specific attack. Our previous works focused on generating rules to detect network anomalous activities and finding the self-similarity among the attacks. While the generated rules clearly differentiated normal and abnormal behaviors, there was a limitation of providing a detailed information (i.e. reasons) about the detected abnormal behaviors. To address this limitation, the categorical variables are converted to dummy variables. In addition, a visual anlaytics approach is integrated to identify transparent reasons about detected abnormal activities.

3. Approach

3.1. Data Description

In this study, a publicly available intrusion detection dataset (called NSL-KDD dataset (NSL-KDD, 2014; Tavallaee et al., 2009)) is used. NSL-KDD dataset is the refined version of the KDD cup'99 dataset that redundant data records are removed (Tavallaee et al., 2009; NSL-KDD, 2014). The NSL-KDD dataset includes training set (125,973 records) and testing set (22,544 records). It contains 41 attributes (three nominal, six binary, and thirty-two numeric attributes), and includes normal activity and twenty-four attacks. These attacks are grouped into four major categories. Table 1 represents the four major attacks and intrusion categories. In this study, the training and testing data were combined to make a new input data. A total of 148,517 records were used as an input data.

DoS attack indicates any attempts to disable network access from remote machines (or computing resources). R2L represents that a remote user gains an access to local user accounts by sending packets to a computing machine over the network. Probe indicates that network is scanned to gather information to find known vulnerabilities. U2R denotes that an attacker accesses normal users' accounts by exploring the system as a root-user.

Table 1: Four attack categories in the NSL-KDD dataset.

Four categories	Intrusion types
DoS	back, land, neptune, pod, smurf,teardrop, mailbomb,processtable
R2L	ftp_write, imap, guess_passwd, multihop, phf, spy, warezclient, warezmaster sendmail,snmpgetattack,snmpguess,worm,xlock,xsnoop,named
U2R	buffer_overflow, loadmodule, perl, spy, rootkit,ps, xterm, sqlattack,mscan
Probe	ipsweep, nmap, portsweep, satan, saint

3.2. Methods

In this section, a brief explanation about our proposed multi-level network intrusion detection approach is provided. As shown in Figure 1, the approach is consisted of three steps; 1) generating rules to detect outcome (normal/abnormal), 2) building an abnormal network behavior model to detect exact attack categories (i.e. DoS, Probe, R2L), and 3) conducting an interactive visual analysis to provide transparent reasons. First, the input data is divided into two subsets; categorical (i.e. nominal) data and numerical data. The nominal variables are used to generate rules. To determine exact attack categories, an extraction of significant DWT features from the numerical variables is performed. Furthermore, an interactive visual analysis is conducted to find the relationship between the raw and the DWT features and to present transparent reasons about the results.

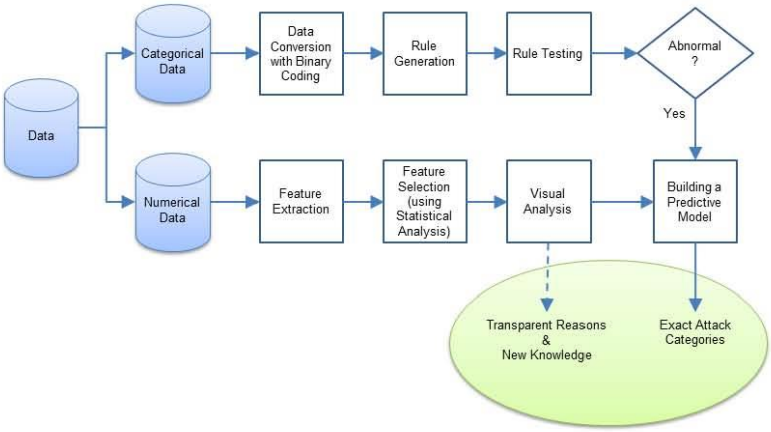


Figure 1: A schematic diagram of the proposed approach

3.2.1. Detection of abnormal behavior

Pre-Processing: As mentioned above, the NSL-KDD data set contains three nominal variables that include protocol type, service, and flag. However, each nominal variables contains many distinctive attribute values. Protocol type includes three attributes (i.e. TCP, UDP, and ICMP), service includes 70 attribute values (i.e. SMTP, HTTP, POP3, SSH, WHOIS, and among

others), and flag contains 11 attributes (i.e. SF, S2, S1, S3, REJ, RSTR, and among others). Since the nominal variables contain numerous amount of attribute values, it is difficult to extract transparent information regarding network abnormality. To resolve this issue, a binary coding scheme (Shyu et al., 2005) via the use of indicator variables is applied to the three nominal variables. Binary coding uses 1 ("one") to indicate the occurrence of a category of interest and 0 ("zero") to represents its nonoccurrence (Neter et al., 1996). For example, if the attribute value of protocol type is "TCP", it is converted to 1, and otherwise 0.

When labeling all attacks as "abnormal", total of 77,054 normal and 71,463 abnormal data are formed. To generate a rule-based method to identify abnormal behaviors, nominal and binary variables are used. By reforming the nominal variables, total of 90 features including binary variables (i.e. yes/no) are generated. Since the binary coding to the nominal variables causes an increase of data dimensions, important features are selected. For this selection, a statistical validation using SAS is performed. Then, each normal and abnormal data are randomly divided into 10 different subsets to apply ten-fold cross validation.

Rule generation with CART: To design a rule-based model, Classification and Regression Tree (CART) (Breiman et al., 1984) is used. CART applies the concept of information theory to create a decision tree that captures complex patterns of input data. It is broadly used due to its efficiency in dealing with multiple data types and missing values. CART expression forms explicit and transparent grammatical rules (Loh and Vanichsetakul, 1988; Fu, 2004). Thus, it is much simpler to understand data patterns than other models. In addition, it uses an exhaustive search of all variables and split values to find optimal splits for each node by measuring the degree of impurity for each outcome of the feature. To find the most important features for identifying network traffic abnormal behaviors, a statistical test (i.e. ANOVA) is performed. Then, trees are generated from each training set using the selected significant features. Due to the difficulty of extracting rules from the generated trees, a software application (called *TreeParser*) is designed to extract rules from the trees by navigating all branches of the generated trees. With the extracted rules, the performance of each rule is measured with a distinctive testing dataset.

3.2.2. Classification of exact attack categories

When incoming network traffic events are considered as "abnormal behaviors" or "attacks", it is important to specify their exact attack categories. Providing the exact information is critical for system administrators so that relevant actions can be taken to protect computing infrastructures. In this study, three attack categories (i.e. DoS, Probe, R2L) are considered due to insufficient amount of U2R data.

Feature extraction and selection: Since signal processing technique has a capability of discovering hidden patterns from input data, discrete wavelet transform (DWT) is used. DWT is a promising technique for time-frequency analysis by decomposing the input data until predetermined level. By decomposing the input data, further detailed information (e.g. any pattern changes) can be represented. It is beneficial to non-stationary data such as network traffic since DWT has an ability to detect any changes from the data. Due to the benefit, Wavelet Transform (WT) is commonly used to analyze data in other domains such as medicine, health, and stock. While researchers (Callegari et al., 2008; Gao et al., 2006; Tan et al., 2012; Dainotti et al., 2006) utilized WT techniques in the context of intrusion detection, they only used WT for reconstructing the data or determining a threshold for detecting intrusions. The threshold was used to make a decision to determine abnormality in their studies. However, in our study, we used DWT to extract new features representing hidden but significant patterns.

The selection of specific mother wavelet is often considered as a difficult task since results can vary depending on what mother wavelet is applied. For this study, a broadly used Daubechies' wavelet family (specifically, a db2) is utilized. A three-level decomposition is applied to the data with an overlapping sliding window (size of 100 data points) to examine rapid changes within the data. By applying DWT, three features (i.e. standard deviation of absolute values, root mean square, and energy) are calculated. The features are

$$\sigma_k = \sqrt{[(\frac{1}{N}\Sigma_{i=1}^{N}(|d_i^k| - \mu)^2]},$$
$$m_k = \sqrt{((1/N \quad \Sigma_{i=1}^{N}(d_i^k)^2)},$$
$$e_k = \Sigma_{i=1}^{N}(|d_i^k|)^2$$

where $\mu = \frac{1}{N}\Sigma_{i=1}^{N} d_i^k$, N is the size of each coefficient, d_i represents wavelet coefficients, and k indicates a decomposition level (our study uses $k = 3$).

Detection of exact attacks: Once the features are extracted, the significance of each feature is tested. Only significant features are selected to generate a classifier (i.e. learning model) that can be used to detect exact attack categories using ML algorithms. Three ML algorithms such as SVM, Neural Network (NN), and Naïve Bayes are compared. Naïve Bayes and NN are commonly used to classify data consisting of two groups (e.g. normal/abnormal). The main idea of SVM, a statistical learning theory, is finding a hyperplane that can separate the input data precisely. That is, SVM finds the optimal hyperplane by minimizing the mis-classification error. Naïve Bayes, a simplified bayesian probability model based on bayes theorem, calculated prior and conditional probabilities to generate a learning model. This learning model may cause an error because of the impacts of bias and variance, and training data noise. NN is an information processing model that is inspired by the biological nervous systems. It is composed of a large number of highly interconnected neurons. It has limitations including falling into a local solution instead of global one and having a slow convergence. In general, SVM (Vapnik, 1998) is simple, fast in operation, and has good robustness than Bayes and Neural Network. Therefore, it is widely used in different domains such as bioinformatics (Idicula-Thomas et al., 2006), data mining, pattern recognition (Shawe-Taylor and Cristianini, 2004), and text categorization (Joachims, 1998). In this study, SVM is used to generate a classifier. Also, a performance comparison with NN and Naïve Bayes is conducted.

3.2.3. Visual analysis

A visual analytics approach is utilized to perform an interactive visual analysis on network traffic data. Visual analytics has been known as a new research area that focuses on performing analytical reasoning with interactive visual interfaces (Thomas and Cook, 2006). In this study, an extended version of a visual analytics tool called iPCA (Jeong et al., 2009) is used to conduct an interactive factor analysis. iPCA is designed to represent the results of Principal Component Analysis (PCA) using multiple coordinated views and a rich set of user interactions to support interactive analysis of multivariate datasets. The network traffic data are projected onto two user-selected principal components. A parallel coordinates visualization is used to show the data in the original data dimensions. In the parallel coordinates visualization, horizontal lines represent features of the data and each line indicates an individual network traffic data. Within iPCA, the user is allowed to select data in one view and immediately see the corresponding data highlighted in the other view which helps the user to understand the relationship between

the two. To enhance the capability of interactive visual analysis within each view, several user interactions (i.e. highlighting, brushing, and filtering of data items or dimensions) are supported. A detailed explanation of conducted visual analysis with iPCA is included in Section 4.2.2 and 4.2.3.

4. Results

This section presents the generated rules to identify network abnormality, the performance of detecting exact attack categories, and the visual analysis to examine the relationship among the DWT features and its correlation analysis.

4.1. Abnormal behavior detection:

As described in Section 3.2.1, total of 77,054 normal and 71,463 abnormal data are used. After converting the nominal input variables to binary scheme indicators, total of 90 variables including six binary variables are generated. A statistical analysis (i.e. ANOVA) is performed to determine statistically significant features. As a result, 22 features (e.g. ICMP, HTTP, SMTP, domain_u, SF, private, S2, S1, IRC, REJ, land_0, login_Yes, POP3, FTP, FTP_data, x11, Host_login_Yes,urp_i, Telnet, IMAP4, Guest_login_Yes, Gopher) are found to be statistically significant ($p < .05$). Then, the 22 significant features are used to generate decision trees. Ten trees are created and tested with distinctive test datasets. Table 2 represents the samples of extracted rules maintaining the testing accuracy of 85% or above.

Table 2: Samples of the extracted rules that are used to identify abnormal network traffic behaviors.

Rules	Testing Accuracy
If(SF='NO' & http='NO' & login_Yes='YES' & IRC='NO' & S1='NO' & smtp='NO' & X11='NO') then Abnormal	5521/ 5542=99.62%
If (SF='YES' & ICMP='YES' & urp_i='NO') then Abnormal	840/ 929=90.41%
If(SF='YES' & ICMP='NO' & private='NO' & pop_3='YES') then (Abnormal)	324/ 342=94.73%
If (SF='YES' & ICMP='NO' & private='NO' & ftp='NO' & pop_3='NO' & telnet='YES' & login_No='NO') then Abnormal	506/ 507=99.80%
if(SF='NO' & http='YES' & REJ='YES') then Normal	304/ 326=93.25%
If (SF='YES' & ICMP='NO' & private='NO' & pop_3='NO' & telnet='NO' & ftp='NO' & ftp_data='YES') then Normal	560/ 633=88.46%
If(SF='YES' & ICMP='NO' & private='NO' & pop_3='NO' & telnet='NO' & ftp='NO' & ftp_data='NO' & imap4='NO' & tcp='NO') then Normal	1271/ 1297=97.99%
If(SF='NO' & http='YES' & REJ='YES') then Normal	308 / 333=92.49%
If(SF='YES' & ICMP='NO' & private='NO' & pop_3='YES') then Abnormal	324 / 342=94.73%
If (SF='YES' & ICMP='NO' & Pop_3='NO' & telnet='NO' & ftp='NO' & ftp_data='NO' & imap4='NO' & tcp='YES' & login='NO') then Normal	4799 / 4913=97.67%
If (SF='YES' & ICMP='NO' & ftp='NO' & pop_3='NO' & telnet='NO' & ftp_data='NO' & gopher='NO' & login='NO') then Normal	5744 / 6085=94.4%

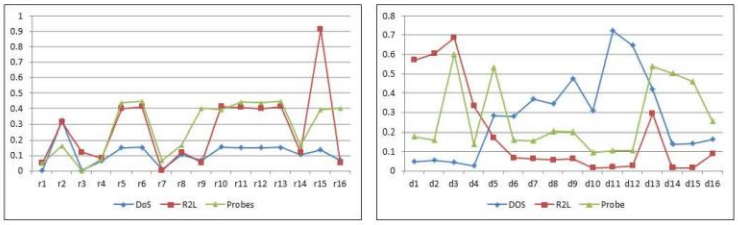
We found that "SF", one of the attribute values in "flag", is an important attribute to identify network abnormality. Also, the generated rules are complicated to present the "Abnormal" behavior. When considering the "SF" feature (indicating normal establishment and termination), if the "SF" feature is "NO", there is a higher chance that network activities are determined as abnormal behaviors. However, it is important to verify the result by checking other features. Due to this reason, the size of the rule can be longer and complex than when the "SF" feature is "Yes".

4.2. Exact attack category detection:

To detect the exact attack category, thirty-two numerical variables in abnormal data (i.e. total of 71,344) are used. A total number of 54,275 data for the DoS attack, 14,077 for the Probe attack, and 2,992 for the R2L attack are used, respectively. Since two numerical variables (i.e. urgent and num_outbound_cmds) have all zero values, they are removed from the analysis. As explained in Section 3.2.2, DWT is applied to extract features. With the DWT, total of 2,841 (2,167 for DoS, 559 for Probe, and 115 for R2L) datasets with 144 features are generated. A statistical test is applied to find a statistical significance of each feature. As a result, 77 out of 144 features were determined as statistically significant ($p < 0.05$) features.

4.2.1. Feature comparison

A feature comparison between the raw and the DWT features is performed by measuring the average of the features. Since the raw and the DWT features have different scales, a normalization between 0 and 1 is applied. As shown in Figure 2, we found that the DWT features clearly separate the attack categories while the raw features maintain similar patterns. For the raw features, we noticed that the five features (i.e. r1, r4, r7, r8, and r14) are almost identical between the two attack categories (Probe and R2L). Although the DoS attack shows a distinctive pattern among the three attacks at the features (see the features of r5, r6, r10, r11, r12, and r13), the raw features may not be useful for differentiating the three exact attacks.

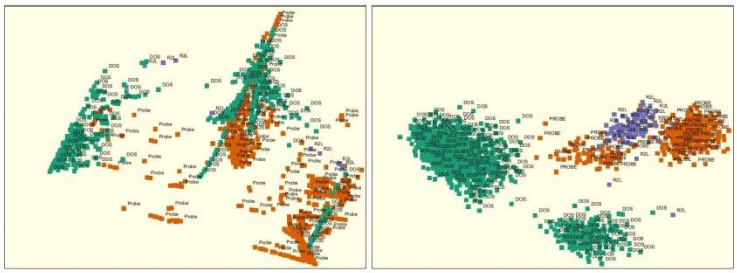


(A) (B)

Figure 2: A comparison between the raw features (A) and the DWT features (B). x-axis indicates the DWT and raw features, and y-axis presents the average value of each feature.

4.2.2. Visual comparison of the features

To project the raw and the DWT features, PCA computation is performed to identify principal components. PCA requires a high computational power to compute eigenvectors and eigenvalues, thus an approximation method based on SVD called Online SVD (Brand, 2006) is used to perform the PCA computation and maintain real-time user interactions when interacting with large scale datasets. Figure 3 represents PCA projections with two principal components on (A) the raw features and (B) the DWT features. From the projection of the raw feature (Figure 3(A)), it is difficult to identify a clear separation among the three attack categories. The DoS attacks are appeared mostly in three regions, the Probe attacks occupies two regions, and the R2L attacks are spread out all over the Projection space. This indicates that identifying the difference among the three attacks is extremely difficult due to the fact that they maintain similar patterns.



(A) (B)

Figure 3: PCA projections of (A) the raw feature and (B) the DWT feature datasets. The data are mapped with different color attributes as DoS (green), Probe (orange), and R2L (purple).

However, there was a clear separation among the attacks in the projection of the DWT features (see Figure 3(B)). The DoS attack is forming two clusters that are completed separated from other attacks. Since there is a similarity between Probe and R2L even in the DWT features, an additional analysis is conducted to determine common features appeared in both categories.

4.2.3. Dimension contribution analysis

iPCA supports the change of dimension contributions by moving slider bars where each feature provides the ability to analyze the data non-linearly. The dimension contribution analysis is performed to identify dominant features that make several attacks to become appeared within other clusters. As shown in Figure 4, when dimension contribution analysis is performed by changing the contribution of the five features (d37, d38, d68, d72, and d75) from 100% to 0%, a clear separation of pattern is emerged. Interestingly, we identified a couple of possible outliers. Figure 4 (A) indicates that a R2L attack is appeared within a DoS cluster. Figure 4 (B) represents that a DoS attack positioned in a R2L cluster. These outliers might be strongly related to the five features. To investigate the cause of the items being appeared in other attack clusters, it is important to conduct an outlier analysis. Since understanding outliers is not a primary concern of this study, we leave it as a future work.

To investigate the relationship among the features, Pearson-correlation analysis between each pair of features is conducted. Figure 5 represents the correlations of the (A) raw and (B) DWT feature datasets. In Figure 5, the diagonal displays the name of dimension as a text string. The lower triangulation shows the coefficient value between two dimensions with a color indicating positive (red), neutral (white), and negative (blue) correlations. The upper triangulation contains cells of scatter plots where all data items are projected onto the two intersecting dimensions. As we discussed above, there was no clear separation among the attacks using the raw features (see Figure 3(A)). This might be because a half of the features maintain neutral correlations (Figure 5(A)). However, positive and negative correlations are easily discovered in the DWT features (Figure 5(B)). When looking at the scatterplots having highly positive correlation coefficients ($\gamma = 0.99$) in Figure 5(C) and 5(D), we identified that they maintain different distributions. Although the scatterplot in Figure 5(C) shows vertically or horizontally increasing patterns (i.e. skew correlation), the scatterplot in Figure 5(D) presents a directly proportional pattern by show-

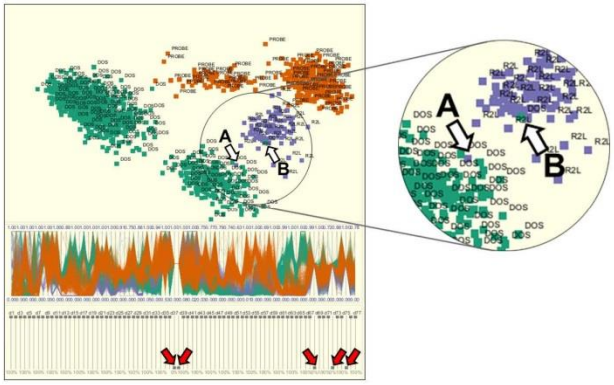


Figure 4: Dimension contribution is applied in the five DWT features (d37, d38, d68, d72, and d75) from 100% to 0% using the slider bars to make a clear separation between Probe and R2L (see the red arrows). 0% indicates that the selected variable is not used to going to contribute to the final PCA.

ing a linear relationship between the two features. In addition, the scatterplot (Figure 5(D)) displays that the attack categories are appeared by forming different patterns as the R2L attacks are mostly appeared in the lower bottom corner, the DoS attacks are forming two visible clusters, and the Probe attacks are spread out in the middle and lower regions.

4.2.4. Classification comparison

A classification is performed to determine exact attack categories with a ten-fold cross validation (CV). The performance of three ML techniques (i.e. SVM, Naïve Byes, and NN) is compared and presented in Table 3. The average accuracy to detect exact attack categories with SVM, Naïve Bayes, and NN were 95.5471%, 89.024%, and 96.67%, respectively. We found that NN shows a slightly higher accuracy than SVM. But, when measuring the standard error of the mean (SEM), there was a variation difference as SVM (0.285), Naïve Bayes(2.02), and NN (0.683). In addition, when generating a learning model with SVM and NN, it took 0.157 seconds and 13.04 seconds, accordingly.

5. Discussion and Conclusion

This study presents a multi-level network abnormality detection method by utilizing reliable rules to detect abnormal behavior, generating a predictive model to detect the exact attacks (i.e. DoS, R2L, and Probe) using the DWT features, and applying a visualization analytic tool to provide further detailed understanding and analysis for users.

Although DWT was often used by researchers to detect network abnormal behaviors, it was simply used to determine a threshold or to reconstruct data by removing noise. Unlike other studies, this study emphasizes the importance of using DWT to extract significant features for detecting network abnormal behaviors. As discussed earlier, our previous study (Ji et al., 2014a) presented decision rules for detecting network abnormal behaviors with utilizing only four variables

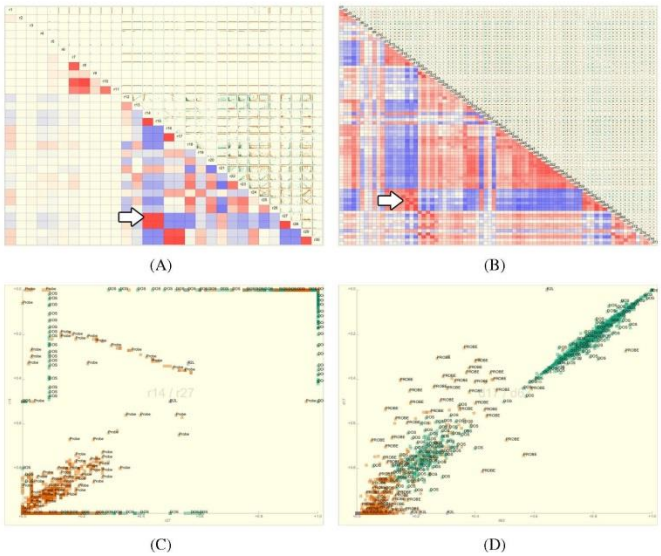


Figure 5: Correlation views of the (A) raw feature and (B) DWT feature datasets. Each color indicates positive (red), neutral (white), and negative (blue) correlations. The arrows in (A and B) indicate the scatterplots having positive correlation coefficients ($\gamma = 0.99$). Their scatterplots are presented in (C and D).

Table 3: Classification performance comparisons

	Three attack classification		
	SVM	Naïve Bayes	NN
Test 1	95.77%	91.47%	94.77%
Test 2	96.83%	91.23%	95.93%
Test 3	96.83%	95.5%	100%
Test 4	95.77%	89.77%	96.83%
Test 5	96.47%	90.47%	95.77%
Test 6	96.12%	78.2%	96.1%
Test 7	95.77%	89.1%	94.57%
Test 8	95.77%	93%	94.2%
Test 9	97.88%	76.8%	100%
Test 10	98.22%	94.7%	98.59%

(i.e. duration, protocol type, service, and flag). The rules were statistically significant to detect intrusions. While the generated rules clearly differentiated normal and abnormal behaviors, there was a limitation of providing a detailed information about the detected abnormal behaviors since each variable includes numerous attribute values. For instance, the rule ($protocol \neq HTTP$) does not provide useful information because there are about 70 attribute values indicating dif-

ferent network protocols. To avoid this ambiguity, the nominal variables are converted dummy variables to generate more accurate rules. So, the result can provide appropriate meaning about the detected network abnormal behaviors.

Based on the performance measure of each rule, only highly accurate rules were used for intrusion detection analysis. However, it is important to note that even the rules with less accuracy may provide a valuable information for detecting intrusions. For instance, the rule - if (SF='YES' & ICMP='NO' & private='YES') then Abnormal - has 72.16% of accuracy. Although the accuracy does not represent a high performance, we found that the rule is fitted to the majority of the data (306 / 424).

Among the extracted DWT features, 53.47% features are shown to be statistically significant ($p < 0.05$). Even though R2L attacks have less amount of data compared to other attacks, we identified that the true positive for the R2L with the raw feature is 59.8 % and 75% for the DWT features. One of the major concerns in many previous studies for detecting intrusions is how to reduce high false positive (FP) results. In our study, the FP rate for the raw and the DWT features were 7.9% and 2.3%, respectively. The DWT features can provide a better performance if we have a larger amount of R2L data. It is also important to note that, unlike other previous methods utilizing wavelet transform techniques, our approach includes a method of performing a mathematical calculation and a statistical validation to extract hidden underlying patterns from the input data.

In this study, we utilized a visual analytics tool to interpret the results, discover new knowledge, and find reasons efficiently. As shown in Figure 3, there was no clear separation of the raw features among DOS, Probe, and R2L. However, when using the DWT features, we identified a clear separation among the attack categories. Most importantly, the "R2L" attack was not identifiable with the raw features. When analyzing the DWT features further, we identified that there was a similarity between Probe and R2L. The dimension contribution analysis was performed with iPCA to identify specific features that make them difficult to separate. The dimension analysis with iPCA is quite challenging because the user needs to maintain an awareness of this change by the contribution since the projection of data will be modified. With carefully adjusting dimension contributions to each feature, we identified a clear separation (see Figure 4). More specifically, we identified five features as strong dimension contributors that make the Probe and R2L attacks appeared nearby in the PCA projection.

Our study has potential avenues for future research. We plan to enhance our approach by identifying possible outliers and understand their patterns as well as effectiveness for determining the abnormality precisely. In addition, we are going to test our proposed approach with different network intrusion datasets. In this study, we only focused on utilizing supervised learning algorithms. To determine the effectiveness of our approach of extracting and utilizing DWT features, we consider to compare our approach to unsupervised learning algorithms. In addition, we are going to conduct additional visual analysis to identify the cause of outliers appeared in the network traffic data. Lastly, our method can be applied to other research domains that require to detect abnormal behaviors (or activities) with providing meaningful information. Specifically, we plan to apply our proposed approach to detect abnormality in software applications.

6. Acknowledgment

This study is based on the work fully supported by U.S. Army Research Office (ARO Grant NO. W911NF-13-1-0143) and partially supported by the same agency (ARO Grant No. W911NF1210060).

References

Agarwal, B., Mittal, N., 2012. Hybrid approach for detection of anomaly network traffic using data mining techniques. Procedia Technology 6, 996–1003.

Amini, M., Rezaeenour, J., Hadavandi, E., 2015. Effective intrusion detection with a neural network ensemble using fuzzy clustering and stacking combination method. Journal of Computing and Security 1 (4).

Aslahi-Shahri, B., Rahmani, R., Chizari, M., Maralani, A., Eslami, M., Golkar, M., Ebrahimi, A., 2015. A hybrid method consisting of ga and svm for intrusion detection system. Neural Computing and Applications, 1–8.

Bace, R., 1999. An introduction to intrusion detection & assessment. ICSA Intrusion Detection Systems Consortium White Paper, 1–38.

Brand, M., 2006. Fast low-rank modifications of the thin singular value decomposition. Linear Algebra and its Applications 415 (1), 20–30.

Breiman, L., Friedman, J. H., Olshen, R. A., Stone, C. J., 1984. Classification and Regression Trees. Chapman & Hall, New York.

Callegari, C., Giordano, S., Pagano, M., 2008. Application of wavelet packet transform to network anomaly detection. In: Balandin, S., Moltchanov, D., Koucheryavy, Y. (Eds.), Next Generation Teletraffic and Wired/Wireless Advanced Networking. Vol. 5174 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 246–257.

Chadha, K., Jain, S., 2015. Hybrid genetic fuzzy rule based inference engine to detect intrusion in networks. In: Intelligent Distributed Computing. Springer, pp. 185–198.

Chandola, V., Banerjee, A., Kumar, V., 2009. Anomaly detection: A survey. ACM computing surveys (CSUR) 41 (3), 15.

Dainotti, A., Pescape, A., Ventre, G., Nov 2006. Nis04-1: Wavelet-based detection of dos attacks. In: Global Telecommunications Conference, 2006. GLOBECOM '06. IEEE. pp. 1–6.

Das, N., Sarkar, T., 2014. Survey on host and network based intrusion detection system. Int. J. Advanced Networking and Applications 6 (2), 2266–2269.

Duftschmid, G., Miksch, S., 2001. Knowledge-based verification of clinical guidelines by detection of anomalies. Artificial intelligence in medicine 22 (1), 23–41.

Eesa, A. S., Orman, Z., Brifcani, A. M. A., 2015. A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems. Expert Systems with Applications 42 (5), 2670–2679.

Eskin, E., Arnold, A., Prerau, M., Portnoy, L., Stolfo, S., 2002. A geometric framework for unsupervised anomaly detection. In: Applications of data mining in computer security. Springer, pp. 77–101.

Fu, C. Y., 2004. Combining loglinear model with classification and regression tree (cart): an application to birth data. Computational Statistics & Data Analysis 45 (4), 865 – 874.

Gan, X.-s., Duanmu, J.-s., Wang, J.-f., Cong, W., 2013. Anomaly intrusion detection based on pls feature extraction and core vector machine. Knowledge-Based Systems 40, 1–6.

Gao, J., Hu, G., Yao, X., Chang, R., Aug 2006. Anomaly detection of network traffic based on wavelet packet. In: Communications, 2006. APCC '06. Asia-Pacific Conference on. pp. 1–5.

Golmah, V., 2014. An efficient hybrid intrusion detection system based on c5. 0 and svm. International Journal of Database Theory and Application 7 (2), 59–70.

Idicula-Thomas, S., Kulkarni, A. J., Kulkarni, B. D., Jayaraman, V. K., Balaji, P. V., 2006. A support vector machine-based method for predicting the propensity of a protein to be soluble or to form inclusion body on overexpression in escherichia coli. Bioinformatics 22 (3), 278–284.

Jain, R., Abouzakhar, N., 2013. A comparative study of hidden markov model and support vector machine in anomaly intrusion detection. Journal of Internet Technology and Secured Transactions (JITST) 2 (1/2/3/4), 176–184.

Jeong, D. H., Ziemkiewicz, C., Fisher, B., Ribarsky, W., Chang, R., 2009. iPCA: An Interactive System for PCA-based Visual Analytics. Computer Graphics Forum.

Ji, S.-Y., Choi, S., Jeong, D. H., Aug 2014a. Designing a two-level monitoring method to detect network abnormal behaviors. In: Information Reuse and Integration (IRI), 2014 IEEE 15th International Conference on. pp. 703–709.

Ji, S.-Y., Choi, S., Jeong, D. H., 2014b. Designing an internet traffic predictive model by applying a signal processing method. Journal of Network and Systems Management, 1–18.

Joachims, T., 1998. Text categorization with suport vector machines: Learning with many relevant features. In: Proceedings of the 10th European Conference on Machine Learning. ECML '98. Springer-Verlag, London, UK, UK, pp. 137–142.

Kou, Y., Lu, C.-T., Sirwongwattana, S., Huang, Y.-P., 2004. Survey of fraud detection techniques. In: Networking, sensing and control, 2004 IEEE international conference on. Vol. 2. IEEE, pp. 749–754.

Kruegel, C., Toth, T., 2003. Using decision trees to improve signature-based intrusion detection. In: Recent Advances in Intrusion Detection. Springer, pp. 173–191.

Kuang, F., Zhang, S., Jin, Z., Xu, W., 2015. A novel svm by combining kernel principal component analysis and improved chaotic particle swarm optimization for intrusion detection. Soft Computing, 1–13.

Kumar, S., Spafford, E. H., 1994. A pattern matching model for misuse intrusion detection.

Lee, J.-H., Lee, J.-H., Sohn, S.-G., Ryu, J.-H., Chung, T.-M., 2008. Effective value of decision tree with kdd 99 intrusion detection datasets for intrusion detection system. In: Advanced Communication Technology, 2008. ICACT 2008. 10th International Conference on. Vol. 2. IEEE, pp. 1170–1175.

Lin, W.-C., Ke, S.-W., Tsai, C.-F., 2015. Cann: An intrusion detection system based on combining cluster centers and nearest neighbors. Knowledge-Based Systems 78, 13–21.

Loh, W.-Y., Vanichsetakul, N., 1988. Tree-structured classification via generalized discriminant analysis. Journal of the American Statistical Association 83 (403), pp. 715–725.

Markou, M., Singh, S., 2003a. Novelty detection: a reviewpart 1: statistical approaches. Signal processing 83 (12), 2481–2497.

Markou, M., Singh, S., 2003b. Novelty detection: a reviewpart 2:: neural network based approaches. Signal processing 83 (12), 2499–2521.

Neter, J., Kutner, M. H., Nachtsheim, C. J., Wasserman, W., 1996. Applied linear statistical models. Vol. 4. Irwin Chicago.

NSL-KDD, 2014. NSL-KDD dataset. `http://nsl.cs.unb.ca/NSL-KDD/`, [Online; accessed 2-April-2014].

Rubin, S., Jha, S., Miller, B. P., 2004. Automatic generation and analysis of nids attacks. In: Computer Security Applications Conference, 2004. 20th Annual. IEEE, pp. 28–38.

Sanei, S., Smaragdis, P., Ho, A. T., Nandi, A. K., Larsen, J., 2015. Guest editorial: Machine learning for signal processing. Journal of Signal Processing Systems 79 (2), 113–116.

Sani, R. A., Ghasemi, A., 2015. Learning a new distance metric to improve an svm-clustering based intrusion detection system. In: Artificial Intelligence and Signal Processing (AISP), 2015 International Symposium on. IEEE, pp. 284–289.

Shawe-Taylor, J., Cristianini, N., 2004. Kernel methods for pattern analysis. Cambridge university press.

Shyu, M.-L., Sarinnapakorn, K., Kuruppu-Appuhamilage, I., Chen, S.-C., Chang, L., Goldring, T., 2005. Handling nominal features in anomaly intrusion detection problems. In: Research Issues in Data Engineering: Stream Data Mining and Applications, 2005. RIDE-SDMA 2005. 15th International Workshop on. IEEE, pp. 55–62.

Stein, G., Chen, B., Wu, A. S., Hua, K. A., 2005. Decision tree classifier for network intrusion detection with ga-based feature selection. In: Proceedings of the 43rd annual Southeast regional conference-Volume 2. ACM, pp. 136–141.

Tan, J., Chen, X.-s., Du, M., Zhu, K., 2012. A novel internet traffic identification approach using wavelet packet decomposition and neural network. Journal of Central South University 19 (8), 2218–2230.

Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A. A., 2009. A detailed analysis of the kdd cup 99 data set. In: Proceedings of the Second IEEE International Conference on Computational Intelligence for Security and Defense Applications. CISDA'09. IEEE Press, Piscataway, NJ, USA, pp. 53–58.

Thomas, J. J., Cook, K. A., Jan. 2006. A visual analytics agenda. IEEE Comput. Graph. Appl. 26 (1), 10–13.
URL `http://dx.doi.org/10.1109/MCG.2006.5`

Vapnik, V. N., 1998. Statistical Learning Theory. Wiley-Interscience.

Wang, G., Chen, S., Liu, J., 2015. Anomaly-based intrusion detection using multiclass-svm with parameters optimized by pso.

Worden, K., Dulieu-Barton, J., 2004. An overview of intelligent fault detection in systems and structures. Structural Health Monitoring 3 (1), 85–98.

Xiang, C., Yong, P. C., Meng, L. S., 2008. Design of multiple-level hybrid classifier for intrusion detection system using bayesian clustering and decision trees. Pattern Recognition Letters 29 (7), 918–924.

Yang, W., Sun, C., Zhang, L., 2011. A multi-manifold discriminant analysis method for image feature extraction. Pattern Recognition 44 (8), 1649–1657.

9. Additional Works done by Dr. Kun Sun

(1) Now You See Me: Hide and Seek in Physical Address Space [1]

With the growing complexity of computing systems, memory based forensic techniques are becoming instrumental in digital investigations. Digital forensic examiners can unravel what happened on a system by acquiring and inspecting in-memory data. Meanwhile, attackers have developed numerous anti-forensic mechanisms to defeat existing memory forensic techniques by manipulation of system software such as OS kernel. To counter anti-forensic techniques, some recent researches suggest that memory acquisition process can be trusted if the acquisition module has not been tampered with and all the operations are performed without relying on any untrusted software including the operating system.

However, in this work, we show that it is possible for malware to bypass the current state-of-art trusted memory acquisition module by manipulating the physical address space layout, which is shared between physical memory and I/O devices on x86 platforms. This fundamental design on x86 platform enables an attacker to build an OS agnostic anti-forensic system. Base on this finding, we propose Hidden in I/O Space (HIveS) which manipulates CPU registers to alter such physical address layout. The system uses a novel I/O Shadowing technique to lock a memory region named HIveS memory into I/O address space, so all operation requests to the HIveS memory will be redirected to the I/O bus instead of the memory controller. To access the HIveS memory, the attacker unlocks the memory by mapping it back into the memory address space. Two novel techniques, Blackbox Write and TLB Camouflage, are developed to further protect the unlocked HIveS memory against memory forensics while allowing attackers to access it. A HIveS prototype is built and tested against a set of memory acquisition tools for both Windows and Linux running on x86 platform. Lastly, we propose potential countermeasures to detect and mitigate HIveS.

(2) TrustICE: Hardware-assisted Isolated Computing Environments on Mobile Devices [2]

Mobile devices have been widely used to process sensitive data and perform important transactions. It is a challenge to protect secure code from a malicious mobile OS. ARM TrustZone technology can protect secure code in a secure domain from an untrusted normal domain. However, since the attack surface of the secure domain will increase along with the size of secure code, it becomes arduous to negotiate with OEMs to get new secure code installed. We propose a novel TrustZone based isolation framework named TrustICE to create isolated computing environments (ICEs) in the normal domain. TrustICE securely isolates the secure code in an ICE from an untrusted Rich OS in the normal domain. The trusted computing base (TCB) of TrustICE remains small and unchanged regardless of the amount of secure code being protected. Our prototype shows that the switching time between an ICE and the Rich OS is less than 12 ms.

In summary, we make the following contributions in this work. We design a TrustZone-based isolation framework named TrustICE to provide isolated computing environments on mobile devices without using a hypervisor. We enhance the system security. TrustICE can reduce the attack surface of the secure domain and minimize the system's TCB by moving secure code from the secure domain to the normal domain. TrustICE's TCB only includes a Boot ROM and a small trusted domain controller, which is protected by TrustZone in the secure domain. We can ensure the isolation of secure code in the normal domain. Since all secure code will be executed in the normal domain, we ensure that no matter whether the secure code is running or suspended, the untrusted Rich OS cannot access or manipulate it. We implement a TrustICE prototype on Freescale i.MX53 QSB. The Rich OS is a customized Linux 2.6.35 and Android 2.3.4. The experimental results show that our system can switch from the Rich OS to ICE in 10.6 ms, and switch back from ICE to the Rich OS in 0.8 ms .

[1]. Ning Zhang, Kun Sun, Wenjing Lou, Y. Thomas Hou, and Sushil Jajodia. "Now You See Me: Hide and Seek in Physical Address Space". To appear in the 10th ACM Symposium on Information, Computer and Communications Security (ASIACCS), Singapore, April 14-17, 2015.

[2]. He Sun, Kun Sun, Yuewu Wang, Jiwu Jing, and Haining Wang. "TrustICE: Hardware-assisted Isolated Computing Environments on Mobile Devices". To appear in the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Rio de Janeiro, Brazil, June 22-25, 2015.