JYVÄSKYLÄ STUDIES IN COMPUTING
205

Antti Juvonen

Intrusion Detection Applications Using Knowledge Discovery and Data Mining

JYVÄSKYLÄN YLIOPISTO

JYVÄSKYLÄ STUDIES IN COMPUTING 205

Antti Juvonen

Intrusion Detection Applications Using Knowledge Discovery and Data Mining

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella
julkisesti tarkastettavaksi yliopiston Agora-rakennuksen auditoriossa 2
joulukuun 11. päivänä 2014 kello 12.

Academic dissertation to be publicly discussed, by permission of
the Faculty of Information Technology of the University of Jyväskylä,
in building Agora, auditorium 2, on December 11, 2014 at 12 o'clock noon.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2014

Intrusion Detection Applications Using Knowledge Discovery and Data Mining

JYVÄSKYLÄ STUDIES IN COMPUTING 205

Antti Juvonen

Intrusion Detection Applications Using Knowledge Discovery and Data Mining

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2014

Editors
Timo Männikkö
Department of Mathematical Information Technology, University of Jyväskylä
Pekka Olsbo, Ville Korkiakangas
Publishing Unit, University Library of Jyväskylä

URN:ISBN:978-951-39-5978-4
ISBN 978-951-39-5978-4 (PDF)

ISBN 978-951-39-5977-7 (nid.)
ISSN 1456-5390

Jyväskylä University Printing House, Jyväskylä 2014

ABSTRACT

Juvonen, Antti
Intrusion Detection Applications Using Knowledge Discovery and Data Mining
Jyväskylä: University of Jyväskylä, 2014, 58 p.(+included articles)
(Jyväskylä Studies in Computing
ISSN 1456-5390; 205)
ISBN 978-951-39-5977-7 (nid.)
ISBN 978-951-39-5978-4 (PDF)
Finnish summary
Diss.

Increasing network traffic and introduction of more and more complex web services creates new vulnerabilities for attackers. As a result, attacks have become more complex and unpredictable. New and previously unseen intrusions cannot be detected using manual signature-based detection. For this reason, automatic traffic analysis and anomaly intrusion detection systems are needed. Anomaly detection faces many problems, such as high number of false alarms and lack of validation with real data. This thesis focuses on analyzing real-world network log data by using a knowledge discovery process and data mining methods. The framework combines data preprocessing, dimensionality reduction, clustering and anomaly detection. The proposed parts of the system are tested using real data. The framework is capable of producing meaningful results, such as detecting actual intrusion attempts that are present in the data. In addition, representative visualizations of high-dimensional data can be created to provide more information to the network administrator. The framework is also capable of dynamically analyzing and adding more traffic data as it's being created in the network. The study focuses on practical use and feasibility of the framework, and is backed up by practical experiments presented in the included articles.

Keywords: knowledge discovery, data mining, intrusion detection, anomaly detection, dimensionality reduction, clustering

Author

Antti Juvonen
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Supervisors

Prof. Timo Hämäläinen
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Dr. Erkki Kurkinen
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Reviewers

Prof. Andrey Garnaev
Department of Computer Modelling and Multiprocessor Systems
St. Petersburg State University
Russia

Prof. Tien V. Do
Department of Networked Systems and Services
Budapest University of Technology and Economics
Hungary

Opponent

Prof. Pekka Loula
Pori Department
Tampere University of Technology
Finland

ACKNOWLEDGEMENTS

I would like to express my gratitude to my supervisor Prof. Timo Hämäläinen for continued support and guidance over the years, and to supervisor Dr. Erkki Kurkinen for pushing me forward and making sure I could concentrate on my thesis work.

I would also like to thank my colleagues and collaborators, Dr. Tuomo Sipola, Dr. Mikhail Zolotukhin, Mr. Hannu-Heikki Puupponen, Dr. Jari Kellokoski, Mr. Joonas Koskinen, Mr. Pekka Wartiainen and all others who have worked with me or otherwise helped me along the way. Thanks are extended to Prof. Amir Averbuch and Dr. Gil David for research ideas and tips.

I am grateful to the Faculty of Information Technology of University of Jyväskylä for giving me this opportunity to work on and finish my dissertation. I would also like to thank the Nokia Foundation for supporting my research financially and giving me motivation.

Finally, I would like thank my family and friends for continued support and believing in me during my dissertation work.

GLOSSARY

$\mathbf{X}$	Original feature data matrix
$\mathbf{x}_i$	A single data point from $\mathbf{X}$
n	The number of input data points
D	The original number of dimensions in the data
k	The number of lower dimensions after dimensionality reduction
AD	Anomaly detection
AIS	Artificial immune system
ANN	Artificial neural network
APT	Advanced persistent threat
BMU	Best matching unit
CIA	Confidentiality, integrity, availability
CRA	Contructive research approach
CRISP-DM	Cross-industry standard process for data mining
DM	Diffusion map
DoS	Denial-of-service
FN	False negative
FP	False positive
GHSOM	Growing hierarchical self-organizing map
IDES	Intrusion detection expert system
IDS	Intrusion detection system
KD	Knowledge discovery
KDD	Knowledge discovery in databases
KDDM	Knowledge discovery and data mining
NIDES	Next-generation intrusion detection expert system
OOS	Out-of-sample
PCA	Principal component analysis
R2L	Remote-to-local
RE	Rule extraction
ROC	Receiver operating characteristics
RP	Random projection
SOM	Self-organizing map
SVD	Singular value decomposition
SVM	Support vector machine
TN	True negative
TP	True positive
U2R	User-to-root

LIST OF FIGURES

FIGURE 1 Paper relationships and contributions 16
FIGURE 2 NIDES intrusion detection framework 21
FIGURE 3 KDD process 23
FIGURE 4 Dimensionality reduction 26
FIGURE 5 PCA example 27
FIGURE 6 K-means clustering example 31
FIGURE 7 SOM training 34
FIGURE 8 Rule extraction example 37
FIGURE 9 Conjunctive rule set size increase 43
FIGURE 10 Overall framework 45

LIST OF TABLES

TABLE 1 Knowledge discovery process comparison 24
TABLE 2 N-gram feature matrix example 25
TABLE 3 KDD data performance metrics 43
TABLE 4 KDD testing data confusion matrix 44
TABLE 5 KDD process steps applied to this research 46

CONTENTS

ABSTRACT
ACKNOWLEDGEMENTS
GLOSSARY
LIST OF FIGURES AND TABLES
CONTENTS
LIST OF INCLUDED ARTICLES

1 INTRODUCTION 13
1.1 Research motivation 13
1.2 Research questions and approach 14
1.3 Structure of the work 15
1.4 Research contributions 15

2 INTRUSION DETECTION 18
2.1 Intrusions 18
2.1.1 Advanced persistent threat 19
2.2 Intrusion detection system 19
2.2.1 Types of intrusion detection systems 19
2.2.2 Anomaly detection systems 20

3 KNOWLEDGE DISCOVERY AND DATA MINING 22
3.1 Knowledge discovery and data mining process 22
3.2 Data acquisition and preprocessing 23
3.2.1 Data selection 24
3.2.2 N-gram analysis 24
3.3 Dimensionality reduction 25
3.3.1 Principal component analysis 26
3.3.2 Random projection 27
3.3.3 Diffusion map 28
3.4 Clustering 30
3.4.1 K-means clustering 30
3.4.2 Spectral clustering 32
3.4.3 Self-organizing Map 32
3.5 Anomaly detection 35
3.6 Out-of-sample extension 35
3.6.1 Random projection and PCA 36
3.6.2 Diffusion maps and Nyström extension 36
3.6.3 Rule extraction 37

4 RESULTS 39
4.1 Anomaly detection from real network data 39
4.2 Increasing performance and adding new data points 41
4.2.1 New unpublished results 42

4.3 Combined overall framework 44
5 CONCLUSION 48
YHTEENVETO (FINNISH SUMMARY) 49
REFERENCES 50
INCLUDED ARTICLES

LIST OF INCLUDED ARTICLES

PI Tuomo Sipola, Antti Juvonen and Joel Lehtonen. Anomaly detection from network logs using diffusion maps. *Engineering Applications of Neural Networks*, IFIP Advances in Information and Communication Technology, Vol. 363, pp. 172–181, 2011.

PII Tuomo Sipola, Antti Juvonen and Joel Lehtonen. Dimensionality reduction framework for detecting anomalies from network logs. *Engineering Intelligent Systems*, Vol. 20, Iss. 1–2, pp. 87–97, 2012.

PIII Antti Juvonen and Tuomo Sipola. Adaptive framework for network traffic classification using dimensionality reduction and clustering. *Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2012 4th International Congress on*, pp. 274–279, 2012.

PIV Mikhail Zolotukhin, Timo Hämäläinen and Antti Juvonen. Growing hierarchical self-organizing maps and statistical distribution models for online detecion of web attacks. *Web Information Systems and Technologies. Lecture Notes in Business Information Processing*, Vol. 140, pp. 281–295, 2013.

PV Antti Juvonen and Tuomo Sipola. Combining conjunctive rule extraction with diffusion maps for network intrusion detection. *The Eighteenth IEEE Symposium on Computers and Communications (ISCC 2013)*, pp. 411–416, 2013.

PVI Antti Juvonen and Timo Hämäläinen. An efficient network log anomaly detection system using random projection dimensionality reduction. *New Technologies, Mobility and Security (NTMS), 2014 6th International Conference on*, 2014.

PVII Antti Juvonen, Tuomo Sipola and Timo Hämäläinen. Online anomaly detection using dimensionality reduction techniques for http log analysis. *Submitted to Computer Networks*, Elsevier, 2014.

1 INTRODUCTION

This chapter explains the background and motivation for the research, as well as the research questions to be answered. In addition, the overall structure of the thesis and included articles and the author's contributions in them are covered.

1.1 Research motivation

Any system or organization needs many layers of security to ensure safe, reliable and efficient operation. Organizations security layers include network and information security (Whitman and Mattord, 2011). These have become increasingly important in recent times, since systems and services are becoming more and more complex, offering new possibilities for attackers.

Information security can be generally summarized by using the CIA triad:

- Confidentiality
- Integrity
- Availability

Confidentiality means that only authorized parties have access to information or data, integrity that the data are not modified in an unauthorized way, and availability that the data are available to the parties that are authorized. These principles are constantly threatened by a multitude of attacks.

At the same time, the amount of data in networks is increasing. This creates new requirements for data analysis solutions and algorithms. There is a large amount of data, often high-dimensional, leading to the curse of dimensionality, which challenges many data mining methods (Houle et al., 2010). Using dimensionality reduction might help solve the problems associated with the curse of dimensionality in intrusion detection. For data analysis, a process called knowledge discovery in databases (KDD) can be used (Fayyad et al., 1996). This kind of process is sometimes referred to as knowledge discovery and data mining (KDDM), since data mining is one of the most important steps in the analysis.

Intrusion detection systems (IDS) (Patcha and Park, 2007) combine attack detection and data mining methods into a system that dynamically finds intrusions from large amounts of traffic. Often the IDS, using anomaly detection methods, will try to find abnormal behavior. However, even though there has been active research in the field of intrusion and anomaly detection, not many anomaly detection systems are actually in use in real systems (Sommer and Paxson, 2010). This could be due to impractical algorithms, poor performance or lack of validation with real data. A lot of research focuses on providing better performance metrics and ROC curves, which leads to algorithms and systems that are customized for specific data sets but do not work in practical real-life situations.

The combination of varied attacks, increasing amounts of data and lack of practical anomaly detection systems makes securing organizations, systems and services harder than ever before. If we ever wish to see anomaly intrusion detection systems in operational use, the above-mentioned problems must be addressed.

1.2 Research questions and approach

To tackle the previously mentioned issues and to increase network security, this research attempts to answer the following research questions:

1. How to detect anomalies from network logs automatically by using KDD process and data mining algorithms?
2. Can we successfully incorporate dimensionality reduction as a part of intrusion detection framework to facilitate anomaly detection and to provide meaningful visualizations?
3. How to dynamically and effectively add new traffic data for online detection?
4. Can the resulting system deal with large amounts of real network traffic data and provide meaningful results to the administrator?
5. Is the system practical and usable in real-world applications?

To answer these questions, this research uses Constructive Research Approach (CRA) (Lukka, 2006; Piirainen and Gonzalez, 2013). The aim is to develop a solution (construction) to a real problem by using existing theoretical knowledge in a way that also contributes to the particular field of science where the method is being applied. In this research, the problem of web server intrusion and anomaly detection is solved with the help of a construction of overall intrusion detection framework using knowledge discovery and data mining methods. In addition, existing theories and algorithms of the field are tested using real-world data.

1.3 Structure of the work

The rest of this thesis is organized as follows. First, the theoretical background on intrusion and anomaly detection as well as knowledge discovery and data mining is introduced. Then, the results obtained in the research articles and new unpublished results are explained. Finally, the thesis is concluded, including future steps and research directions.

Chapter 2, with more detailed explanations, covers potential intrusions that threaten systems. These intrusions include new types of attacks referred to as advanced persistent threats. Also general information about intrusion detection systems is provided. In addition, anomaly detection as a subset of intrusion detection is explained more thoroughly in its own subsection.

Chapter 3 explains the theoretical background of knowledge discovery process and its steps. Data mining, including dimensionality reduction, clustering and anomaly detection, is covered along with the algorithms used in this research. Many of these algorithms can be used to solve different kinds of problems, so each of the sections focuses on how they are used particularly in this research.

Chapter 4 contains the results presented in the included research articles, divided into meaningful section describing the different major results obtained in the articles. In addition, some new results that have not yet been published are added to support the research work.

Finally, Chapter 5 summarizes the work and provides future research directions as well as some discussion about the results.

1.4 Research contributions

The authors general contribution to the included articles consists of the design and implementation of the overall system framework with co-authors, and acquisition and preprocessing of real network log data for analysis. Figure 1 shows the relationships between the included individual articles and their place in the overall scheme. Papers PI; PII; PIII; PIV; PVI deal with anomaly detection from real log data in general, including data preprocessing, dimensionality reduction and anomaly detection. Articles PV; PVII extend the capabilities of the framework by implementing and evaluating methods that can be used to dynamically add new data samples to the system to enable online detection.

Article PI implements the basic idea of using n-gram preprocessing, diffusion map dimensionality reduction and spectral clustering to detect anomalies from network logs. In the experimental part, real-world network logs are analyzed to find intrusions from traffic data. The author is responsible for implementing the preprocessing software, acquiring network logs from the partner company and contributing to the overall design of the framework as well as to

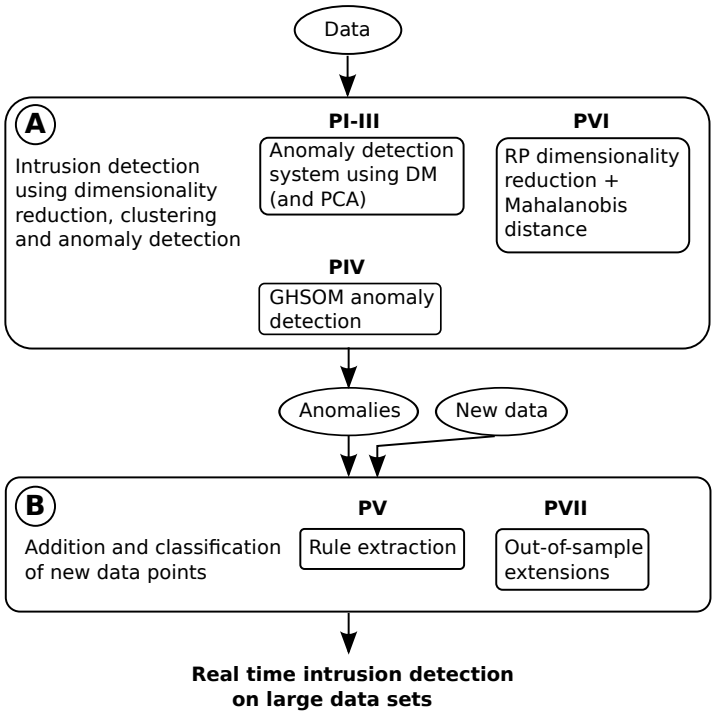

FIGURE 1 The relationships and contributions of different papers.

the analysis process and writing.

Article PII directly expands the framework used in the previous article. Several real-data sets are used to analyze network traffic, detect intrusion dynamically and present meaningful visualizations to the user. In addition, DM methodology is compared to well-known PCA dimensionality reduction for more validation. The author implemented the preprocessing components, performed n-gram analysis, acquired the real data and was involved with the overall framework design and results interpretation and presentation.

Article PIII uses a similar framework but focuses more on the clustering and visualization aspects. As before, n-gram feature matrices are analyzed using DM and PCA algorithms for reduced dimensionality. In addition, k-means clustering is brought in to reveal more information about the underlying structures in the data. The paper reveals significant differences in analysis results between DM and PCA in some data sets. The author acquired the data from a real-life web server used by a company, implemented and performed the n-gram preprocessing, performed the k-means clustering, and contributed to the framework design, article writing and results analysis and interpretation.

In Article PIV, growing hierarchical self-organizing map methodology is used to find anomalies from preprocessed network logs. In addition, statistical distribution models are applied for finding anomalous HTTP header fields. These approaches are combined to detect different kinds of anomalies. The author ac-

quired the data set and contributed to the ideas and overall design of the system, especially for n-gram preprocessing.

Article PV expands the diffusion map-based framework explored previously. Preprocessed data is analyzed using dimensionality reduction and clustering. Subsequently, the input data and output classification results are used to create a rule set with the help of a conjunctive rule extraction algorithm. The rule set then classifies the traffic to normal and anomalous in an efficient way and without the need for heavy algorithms. This creates a hybrid intrusion system that can be dynamically updated and facilitates online detection. The rules are also easy to understand and might provide more semantic understanding to the administrator using the system. The author performed the real network data acquisition and preprocessing, contributed to the overall system design and analysis, and designed and implemented the rule extraction system with the co-author.

Article PVI aims to improve speed and efficiency of the anomaly detection system by using a fast random projection dimensionality reduction. Based on the low-dimensional coordinates, Mahalanobis distance-based anomaly score is calculated so that the data points with a score higher than a preset threshold will be flagged as anomalies. The system is quick and can function in real time. The author preprocessed real network logs by using n-gram analysis, implemented the anomaly detection system that uses RP and Mahalanobis distance, and performed the data analysis and results visualization and interpretation.

Finally, Article PVII combines many of the features individually explored in other articles, compares several dimensionality reduction techniques as a part of an IDS and adds out-of-sample extension capabilities to the system for facilitating dynamic addition of new data points. The author acquired and preprocessed the data set, implemented random projection algorithm and was involved in the overall design and implementation of the framework as well as data analysis process and results interpretation.

2 INTRUSION DETECTION

This chapter provides some background knowledge about intrusion detection. First, different types of intrusions as well as more recent threats known as APTs are explored. Then, intrusion detection systems as well as some of their characteristics and classifications are explored with a more detailed focus on anomaly detection systems.

2.1 Intrusions

An intrusion can be defined as any malicious activity that aims to compromise the security principles presented in the CIA triad in Section 1.1. An attack consists of several stages, which are often summarized in the literature as follows (Asaka et al., 1999):

- Surveillance/probing stage
- Activity stage
- Mark stage
- Masquerading stage

In the first stage (probing), the attacker gathers information about the target system and possibly carries out password cracking to find potential vulnerabilities. The actual exploitation takes place during the second stage (activity) to get free access to the system. Then, in the mark stage, the attacker may steal information, destroy data or plant viruses or spyware on the target system. Finally, in the final stage, the intruder attempts to hide the traces of the attack.

From these stages, we can derive a popular taxonomy of intrusions often found in the literature (Lippmann et al., 2000):

- Denial of Service (DoS)
- Remote to Local (R2L)
- User to Root (U2R)

– Surveillance/Probing

Surveillance attacks work as already explained above. DoS attacks create abnormal amounts of traffic or corrupt messages or packets so that the target system will not be available or working properly. R2L means that an attacker who does not have access to a system sends certain packages to gain local access to the victim machine. Finally, when carrying out U2R attacks, the intruder already has local access but is also able to gain root privileges to the target system, e.g., by using some exploits or vulnerabilities. These attack types have been generally used as a basis for theoretical knowledge in research literature for a long time. However, the range of attacks is always constantly, and new types of intrusions threaten services and systems.

2.1.1 Advanced persistent threat

Security landscape is always changing, and many new and complex attacks have been discovered in recent years. A big and severe class of intrusions is that of advanced persistent threat (APT). The word "advanced" refers to a variety of methods used in combination to enable successful intrusion, and "persistent" means that the goal is to achieve and maintain long-term access to the target system for gaining information to perform malicious activities (Tankard, 2011). Another way to understand APTs is that they are stealthy, targeted and data-focused, to differentiate them from traditional threats (Cole, 2012).

Perhaps the best-known example of an APT is Stuxnet. It is highly complex, targeted towards specific infrastructure and spread by using several propagation methods (Virvilis and Gritzalis, 2013).

These new threats mean that conventional security measures like firewalls and anti-virus software are not enough, since APTs can avoid them. Because of this, sophisticated intrusion detection systems and log analyses comparing log data to baseline traffic should be used (Tankard, 2011).

2.2 Intrusion detection system

An intrusion detection system (IDS) is a system or a tool for detecting unauthorized traffic, malware and potential intrusions (Patcha and Park, 2007). An IDS is not a firewall or anti-virus software, but a dynamic tool used to complement these. The system monitors network traffic like a sniffer and analyses it to find intrusions by using manually generated signatures or automatic anomaly detection.

2.2.1 Types of intrusion detection systems

Intrusion detection systems work using different principles, behavior and audit trail location. Therefore, there are many types of systems. One way to classify

these systems is the following (Engen, 2010):

Audit source location: host based of network based.
Detection method: misuse or anomaly detection.
Detection behavior: passive or active.
Usage frequency: online (real-time) or offline.

An IDS can receive its audit trails from different locations. A host based system works in a single host, whereas a network based IDS monitors the entire network traffic. These two approaches complement each other, since not all malicious activity generates network traffic. One of the most important classifications for an IDS is the detection method. Misuse or signature-based systems use manually generated signatures to find already known intrusions. Most current tools work using this principle because it is accurate. Anomaly detection systems are theoretically capable of detecting unknown intrusions, but they might give more false positives. Most systems work in a passive manner, only notifying the administrator about potential intrusions, but it is also possible to take automatic actions to prevent the attack or mitigate the damage. However, this might sometimes create problems if the counter measures are directed at a legitimate user (Kemmerer and Vigna, 2002). Finally, the IDS can work offline (analyzing historical data) or online (analyzing in real time). It is obvious that online operation has several benefits, even though it can be challenging especially in larger networks (Engen, 2010).

2.2.2 Anomaly detection systems

An anomaly detection system aims to find abnormal behavior that deviates from a normal profile (Patcha and Park, 2007). Anomaly detection offers tools to detect previously unknown threats, but, if the normal behavior is not correctly measured, the detection results can be poor. One of the early examples of intrusion and anomaly detection is Denning's popular model for statistical anomaly detection to find abnormal behavior (Denning, 1987).

From the anomaly detection point-of-view, there are four types of traffic classified by the system (Kumar and Spafford, 1994):

- Intrusive but not anomalous: False negative (FN).
- Not intrusive but anomalous: False positive (FP).
- Not intrusive and not anomalous: True negative (TN).
- Intrusive and anomalous: True positive (TP).

False negatives are intrusions that are not detected but classified as normal, and false positives are false alarms created by non-intrusive behavior. True negatives means traffic correctly classified as normal, and true positives are actual intrusions detected. Many false positives or false negatives can render the anomaly detection system unusable.

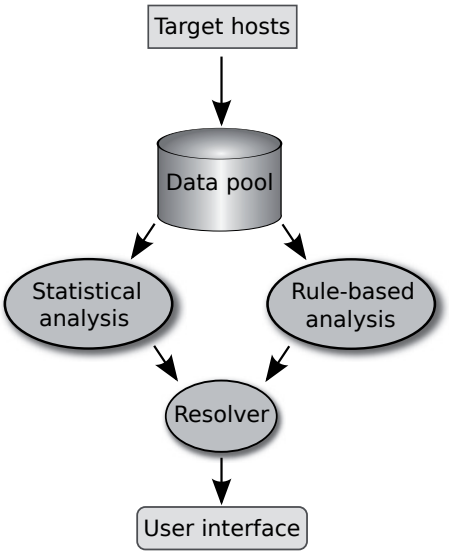

FIGURE 2 NIDES intrusion detection framework (Lunt, 1993).

Anomaly detection can also be combined with signature detection to form a hybrid system which uses some kind of hybrid model to make decisions about normal and intrusive traffic (Patcha and Park, 2007). Perhaps the best-known hybrid IDS, IDES (Lunt, 1990), uses a rule-based system in combination with anomaly detection to find both known and unknown intrusions. This was later followed by the next version of the system, NIDES (Lunt, 1993). A framework diagram of the NIDES model can be seen in Figure 2. More recently proposed systems use different data mining methods for anomaly detection, e.g., clustering analysis (Palnaty and Rao, 2013), change point detection (Tartakovsky et al., 2013), support vector machines (SVM) (Kim et al., 2014), neural networks (Panchev et al., 2014) and artificial immune systems (AIS) (Parashar et al., 2013). The data mining concepts and algorithms are explained in more detail in Chapter 3.

3 KNOWLEDGE DISCOVERY AND DATA MINING

This chapter first introduces the overall knowledge discovery process and its phases and then explains the different components and some algorithms in more detail. The focus is on data mining. The featured algorithms are the ones that are used in this research and are relevant to the goals of this thesis. It is important to note that even though many algorithms can be used for many different tasks, for dimensionality reduction or clustering for example, the algorithm is placed under the section that represents its use in this particular research and the overall framework.

3.1 Knowledge discovery and data mining process

Knowledge discovery (KD) aims to find new knowledge about an application area. It contains several steps, each step completing a single task, data mining being one of the steps (Klosgen and Zytkow, 1996). Knowledge discovery in databases (KDD) is the same process applied to databases. Since the data mining phase is an integral part of knowledge discovery, knowledge discovery and data mining (KDDM) has become a popular and descriptive name for the process.

The KDD process is illustrated in Figure 3. The process steps and the actions performed in each step can be summarized as follows (Fayyad et al., 1996):

Selection:
Selection of target data set and a subset of features or samples to be analyzed.

Preprocessing:
Data cleaning, removal of noise and handling of missing data.

Transformation:
Reduction and projection of the data. Dimensionality reduction and transformation to reduce the number of analyzed variables.

Data mining:
Choosing the data mining algorithm(s) to use based on the goal (regression,

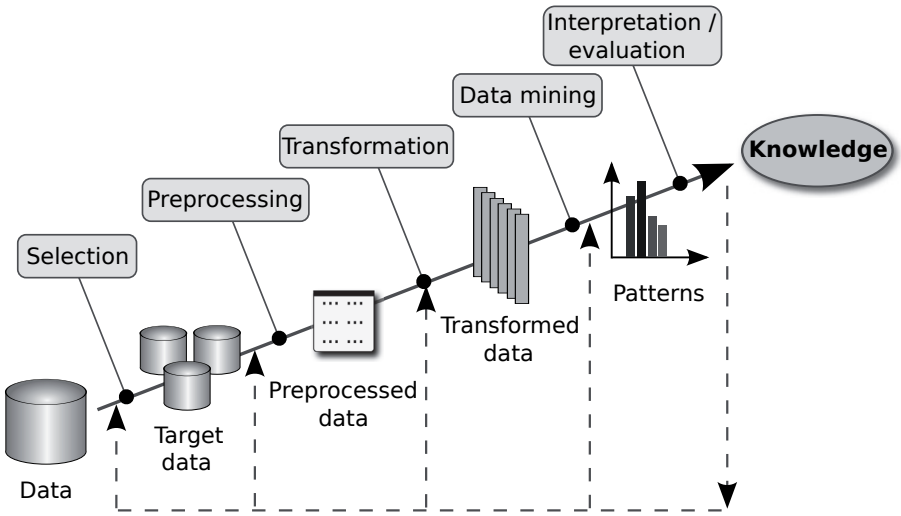

FIGURE 3 KDD process (Fayyad et al., 1996).

classification, clustering etc.), model selection. Performing the data mining analysis resulting in patterns, trees, clusters and so on.

Interpretation:

Interpreting the results obtained in the previous step. Using the discovered knowledge to perform actions and report the results to the involved parties.

As we can see in Figure 3, the process is iterative, and it is possible to loop back to a previous step at any point in the analysis if needed (Fayyad et al., 1996). The full process actually contains 9 phases, but these have been combined to better correspond with the figure.

The process by Fayyad et al. is of academic nature, but there are many other process models, some of them for industrial applications. In Table 1, there is a comparison between the KDD process used and an industrial KD process model called CRISP-DM (Shearer, 2000). It can be seen that the processes are very similar, whereas the focus is slightly different.

When looked at more closely, data mining could be understood as analyzing data present in databases, automatically solving problems there, finding patterns from the data and using these patterns for prediction (Witten and Frank, 2005). Problem solving in this case often involves using automated machine learning algorithms.

3.2 Data acquisition and preprocessing

In this section, data selection and its challenges are explained in the context of the KDD process. In addition, a preprocessing and feature extraction method called n-gram analysis is described with an illustrative example.

TABLE 1 A comparison of two different knowledge discovery processes (Kurgan and Musilek, 2006).

Model	Fayyad et al.	CRISP-DM
Steps	1. Developing and understanding application domain	1. Business understanding
	2. Creating a target data set	2. Data understanding
	3. Data cleaning and preprocessing	3. Data preparation
	4. Data reduction and projection	
	5. Matching KDD process with the data mining method	
	6. Choosing the data mining algorithm	
	7. Data mining	4. Modeling
	8. Interpretation of mined patterns	5. Evaluation
	9. Consolidating discovered knowledge	6. Deployment

3.2.1 Data selection

Once an understanding of the application domain and the goal for the process has been reached, a target data set must be created by acquiring the data as well as possibly focusing on a subset of variables in the data (Fayyad et al., 1996). This phase is not trivial, since acquiring real network data can be difficult due to security concerns and for legal reasons (Sommer and Paxson, 2010). This leads to the use of public test data sets or simulated data, and both of these approaches contain problems, e.g., the traffic might not represent real network accurately. In this research, real network data was acquired, mitigating many of the mentioned issues.

3.2.2 N-gram analysis

In the context of this research, n-grams can be understood as a way to transform textual data into numerical matrices for further analysis. An n-gram can be defined as a consecutive sequence of n characters (Damashek et al., 1995). N-grams are obtained by moving a sliding window with size n through a string or sequence. The parameter n can be freely selected. If we choose $n = 1$, we get a simple character distribution. In many cases, it is feasible to select $n > 1$. If we choose $n \geq 3$, we get *higher order n-grams* (Hubballi et al., 2010). The bigger the parameter n is, the more distinct theoretical n-grams we get. For example, using ASCII character strings (256 different characters), the theoretical maximum number of different 2-grams in the data is $256^2 = 65536$. It is easy to see that the

TABLE 2 An example of n-gram feature matrix $\mathbf{X}$.

	co	om	mp	pu	ut	te	er	ti	in	ng
$\mathbf{x}_{computer}$	1	1	1	1	1	1	1	0	0	0
$\mathbf{x}_{computing}$	1	1	1	1	1	0	0	1	1	1

maximum number goes up rapidly as n increases. However, in a practical situation, most of the possible n-grams never appear in the data, which reduces the size of the feature matrix. In anomaly detection applications, the choice of n is a compromise between the feature matrix size and detection accuracy.

Feature vector and matrix generation can be illustrated with a practical example. For example, let us assume that we have two strings: `computer` and `computing`. If we choose $n = 2$ (sometimes called *bigrams*), we get distinct n-grams `co`, `om`, `mp`, `pu`, `ut`, `te`, `er`, `ti`, `in` and `ng`. For each string, we calculate the frequencies of occurrences of an individual n-gram in that particular string. These form the feature vectors $\mathbf{x}_{computer}$ and $\mathbf{x}_{computing}$, which in turn form the entire feature matrix $\mathbf{X}$. The calculated feature matrix for this example case can be seen in Table 2.

N-grams have been used in several application areas. They are particularly helpful in the area of natural language and text processing (Suen, 1979). Another application area related to language is speech recognition (Hirsimaki et al., 2009). N-grams have also been used for anomaly detection purposes, e.g., finding abnormal sequences of system calls (Hubballi et al., 2011), detecting new and unknown malicious code (Abou-Assaleh et al., 2004) and detecting anomalies from packet payloads by using layered higher order n-grams (Hubballi et al., 2010). In addition, n-grams can be useful for protein sequence analysis (Ganapathiraju et al., 2002).

3.3 Dimensionality reduction

Many times data exist in a high-dimensional feature space. Dimensionality reduction aims to represent this data by using fewer dimensions while minimizing the introduced error (Roweis and Saul, 2000). This may prove essential in dealing with the curse of the dimensionality and in visualizing high-dimensional data as well (Lee and Verleysen, 2007). Dimensionality reduction can be divided into feature selection and feature extraction (Pudil and Novovičová, 1998). Feature selection means selecting a subset from the original features which describe the data accurately enough. Feature extraction (or feature generation) is the process of creating completely new features based on the original ones. Both of these approaches achieve a similar goal: less dimensions with minimal error. An example of dimensionality reduction where dimensions are reduced from 3 to 2 is presented in Figure 4. In the figure, the distances between points are preserved even in the low-dimensional representation. This thesis uses feature extraction meth-

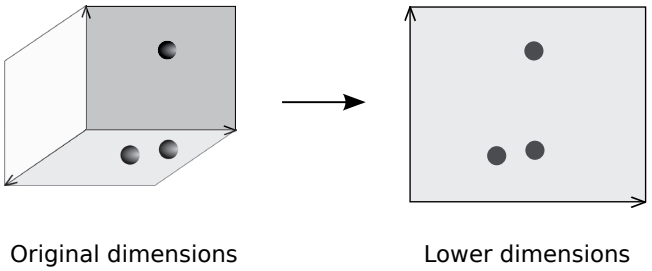

FIGURE 4 The key idea behind dimensionality reduction: An example from 3D to 2D.

ods, because these methods are transformative, follow the KDD process steps and provide us with meaningful visualizations. In this section, different dimensionality reduction methodologies are introduced.

3.3.1 Principal component analysis

Principal component analysis (PCA) is one of the best-known and widely used dimensionality reduction techniques (Jolliffe, 2002). It has a long history and it has been further developed by several researchers over the years (Pearson, 1901; Hotelling, 1933). It has been used in various applications, and is especially useful in the cases where there are many independent variables (dimensions) relative to the number of observations or where the independent variables are highly correlated (Rencher and Christensen, 2012).

PCA is defined as an orthogonal linear transformation that maps the data into a coordinate system so that the first coordinate captures most of the variance, the second coordinate captures the second highest variance, and so on (Jolliffe, 2002). An example of a dataset, with two dimensions and two principal components and their directions, is presented in Figure 5.

Before performing PCA, the data must be centered (i.e., it must have a zero mean), which can be done by subtracting the sample mean from the elements of the data matrix (Lee and Verleysen, 2007). After this, covariance matrix $\mathbf{\Sigma}$ is calculated. Subsequently, eigenvalues and corresponding eigenvectors are obtained from $\mathbf{\Sigma}$. These vectors are then ordered and put into the matrix $\mathbf{W}$ ordered according to the eigenvalues, because eigenvalues are variances of the principal components (Rencher and Christensen, 2012). Only k components are selected, so that they capture a large proportion of the variance (Rencher and Christensen, 2012) and $k \ll D$, where D is the number of original dimensions in the data. Given the original data matrix $\mathbf{X}$, we get a low-dimensional representation of the data (matrix $\mathbf{X}_{PCA}$) by performing the multiplication presented in Equation 1. This way we will get the n original data points presented in k dimensions.

$$\mathbf{X}_{PCA}^{n \times k} = \mathbf{X}^{n \times D} \mathbf{W}^{D \times k} \tag{1}$$

Deciding how many components to retain (i.e., deciding k) is not trivial. Some methods include the following (Rencher and Christensen, 2012):

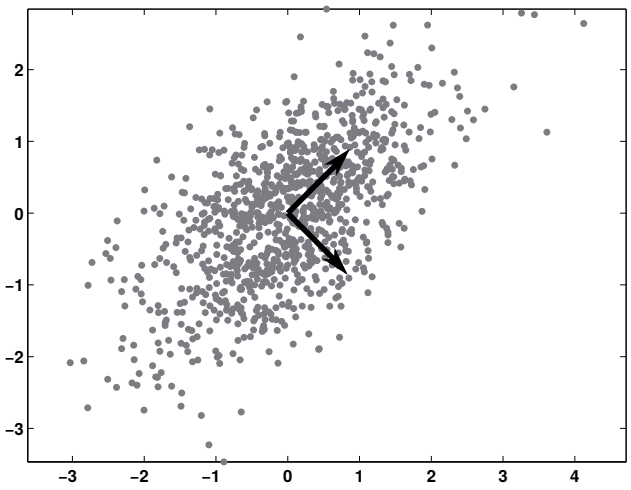

FIGURE 5 An example of PCA. The data points are grey dots, the principal components are black arrows.

1. Retain components that capture a certain percentage of the variance.
2. Retain components whose eigenvalues are greater than the mean of all eigenvalues.
3. Plot the eigenvalues to a graph and look for a gap between larger and smaller eigenvalues
4. Test the significance of larger eigenvalues

PCA has some limitations. The latent variables are assumed to have a Gaussian distribution, and the method fails when dealing with data with nonlinear dependencies (Lee and Verleysen, 2007). However, PCA can be extended to handle nonlinear problems, e.g., by using the Kernel PCA algorithm (Muller et al., 2001).

3.3.2 Random projection

Random projection is an efficient and fast dimensionality reduction method. It's key idea is based on the Johnson-Lindenstrauss lemma, which states that points in a vector space can be projected onto a subspace with suitably high dimensions while approximately preserving the distances between points (Johnson and Lindenstrauss, 1984).

If we have the original data matrix $\mathbf{X}^{n\times D}$ consisting of n data points in D dimensions, random projection is obtained by multiplying matrix $\mathbf{X}$ with random matrix $\mathbf{R}$ (Li et al., 2006). This can be seen in equation 2:

$$\mathbf{X}_{RP}^{n\times k} = \mathbf{X}^{n\times D}\mathbf{R}^{D\times k} \tag{2}$$

This way we end up with the new matrix $\mathbf{X}_{RP}$, which contains the data

points using k dimensions so that $k \ll D$. The projection is computationally very simple, as its complexity is of order $O(Dkn)$ (Bingham and Mannila, 2001). If the data matrix $\mathbf{X}$ is sparse with approximately c nonzero entries per column, the method becomes even simpler and the complexity is of order $O(ckn)$ (Papadimitriou et al., 1998). This means that random projection is especially efficient for sparse data matrices.

The key question with this method deals with random matrix generation. Random projection is an actual projection only if the matrix $\mathbf{R}$ is orthogonal. However, orthogonalizing is computationally complex, and without orthogonalization significant distortions can be introduced in the data (Bingham and Mannila, 2001). Thankfully, a result is found in the literature stating that in a high-dimensional space there exists larger number of almost orthogonal directions than orthogonal directions (Hecht-Nielsen, 1994). This means that in practice vectors with random directions can produce sufficiently accurate results, which is also backed up by experiments (Bingham and Mannila, 2001).

A random matrix can be generated in a way that is fast and easy to implement. Let's denote an individual element of random matrix $\mathbf{R}$ as r_{ij}. These elements could be generated as shown in Equations 3 and 4 (Achlioptas, 2001).

$$r_{ij} = \begin{cases} +1 & \text{with probability } \frac{1}{2} \\ -1 & .. \quad \frac{1}{2} \end{cases} \tag{3}$$

$$r_{ij} = \sqrt{3} \times \begin{cases} +1 & \text{with probability } \frac{1}{6} \\ 0 & .. \quad \frac{2}{3} \\ -1 & .. \quad \frac{1}{6} \end{cases} \tag{4}$$

Alternatively, random matrix elements can be generated with the help of a more general probability distribution as demonstrated in Equation 5 (Li et al., 2006). With this distribution, Equations 3 and 4 proposed by Achlioptas et al. are obtained by choosing $s = 1$ and $s = 3$, respectively. If we choose $s \gg 3$, e.g., $s = \sqrt{D}$ or even $s = \frac{D}{\log D}$ we get a *very sparse random projection* (Li et al., 2006).

$$r_{ij} = \sqrt{s} \times \begin{cases} +1 & \text{with probability } \frac{1}{2s} \\ 0 & .. \quad 1 - \frac{1}{s} \\ -1 & .. \quad \frac{1}{2s} \end{cases} \tag{5}$$

From the above results it can be seen that random projection offers many opportunities for significant computational speedup, as long as proper care is taken to ensure that not too much error is introduced when performing the projection (e.g., choosing s too aggressively).

3.3.3 Diffusion map

Diffusion map is a geometric manifold learning method that embeds high-dimensional data into low-dimensional diffusion space. The methodology is nonlinear and focuses on finding the underlying manifold on which the data points lie.

The low-dimensional embedding facilitates subsequent tasks such as visualization and regression (Coifman et al., 2005). Furthermore, anomaly detection and clustering are easier in this embedded space (Coifman and Lafon, 2006). For overcoming the scaling problem in many datasets, localized diffusion folders have also been introduced (David, 2009; David et al., 2010; David and Averbuch, 2012).

The diffusion map is constructed as follows. Let our original data be $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}, \mathbf{x}_i \in \mathbb{R}^D$, where n is the number of measurements and D denotes the original dimensions of the data, i.e., the data is in a matrix with D features as columns and n samples as rows. We begin by constructing affinity matrix $\mathbf{W}$, using Gaussian kernel and Euclidean distance in this case, as seen in Equation 6 (Coifman and Lafon, 2006; Nadler et al., 2008). When calculating $\mathbf{W}$, we must choose the parameter ϵ for defining the affinity neighborhood, which is not a trivial task (Schclar et al., 2010).

$$w_{ij} = exp\left(-\frac{||x_i - x_j||^2}{\epsilon}\right) \tag{6}$$

Next, we generate diagonal matrix $\mathbf{D}$, which contains the row sums of $\mathbf{W}$ on its diagonal. Using $\mathbf{D}$, we normalize $\mathbf{W}$ so that the sum of each row is 1, which results in matrix $\mathbf{P}$ (Equation 7).

$$\mathbf{P} = \mathbf{D}^{-1}\mathbf{W} \tag{7}$$

After this, we obtain the eigenvalues from the transition probability matrix (the probability of changing from one state to another). By substituting $\mathbf{P}$ in Equation 8 with the one from Equation 7, we get the symmetric probability matrix seen in Equation 9.

$$\tilde{\mathbf{P}} = \mathbf{D}^{\frac{1}{2}}\mathbf{P}\mathbf{D}^{-\frac{1}{2}} \tag{8}$$

$$\tilde{\mathbf{P}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}} \tag{9}$$

The matrix $\tilde{\mathbf{P}}$ is then decomposed using singular value decomposition (SVD): $\tilde{\mathbf{P}} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^*$. The eigenvalues on the diagonal of $\mathbf{\Lambda}$ correspond to the eigenvectors of $\tilde{\mathbf{P}}$, and the matrix $\mathbf{U}$ contains the eigenvectors of $\tilde{\mathbf{P}}$ in its columns. To get the eigenvectors of $\mathbf{P}$ in the columns of $\mathbf{V}$, we use Equation 10.

$$\mathbf{V} = \mathbf{D}^{-\frac{1}{2}}\mathbf{U} \tag{10}$$

Finally, to get the new data coordinates in the embedded space in matrix $\mathbf{\Psi}$ using eigenvalues in $\mathbf{\Lambda}$ and eigenvectors in $\mathbf{V}$, we use Equation 11. If the parameter ϵ is chosen correctly, only k ($k \ll D$) components are needed and not too much information is lost. Now the dimensionality has been reduced from D to k.

$$\mathbf{\Psi} = \mathbf{V}\mathbf{\Lambda} \tag{11}$$

3.4 Clustering

Any clustering algorithm attempts to automatically find the natural groupings of data that is not labeled. We can outline the clustering problem as follows: Based on representations of n observations, try to group them into k groups (clusters) according to a similarity measure in such a way that (Jain, 2010):

1. The similarities between data points *within a group* are high.
2. The similarities between data points *in different groups* are low.

The similarity (or dissimilarity) can mean different things, e.g., it can be the distance between the data points.

In this section, some clustering algorithms relevant to this research are introduced.

3.4.1 K-means clustering

K-means clustering is a famous and widely used clustering algorithm. Since its introduction, many other clustering algorithms have been introduced, but k-means has retained its popularity (Jain, 2010). Even though the algorithm has its limitations, its use in this research is justified when used together with diffusion map dimensionality reduction (Lafon and Lee, 2006).

Given the original data matrix $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, the goal is to cluster the n data points into k clusters. Now an individual data point is $\mathbf{x}_i$, and the center (i.e., the mean point) of cluster c_k is μ_k. The squared error (distance) between data points of c_k and μ_k is (Jain, 2010):

$$J(c_k) = \sum_{x_i \in c_k} ||x_i - \mu_k||^2 \tag{12}$$

If all the clusters are denoted by C, the k-means algorithm aims to minimize the sum of all squared errors within clusters (Jain, 2010):

$$J(C) = \sum_{k=1}^{K} \sum_{x_i \in c_k} = ||x_i - \mu_k||^2 \tag{13}$$

In an optimal solution, the clusters lie in their respective Voronoi regions and the centroids are at the center of the mass of data points within one cluster (Ostrovsky et al., 2006). The k-means algorithm (and thus the solution to the problem described above) can be broken down into these steps (Jain and Dubes, 1988):

1. Select k initial cluster centers (centroids).
2. Based on the closest cluster centroid, assign each data point to a cluster.
3. Calculate the new cluster centroids by using the points assigned in the previous step.

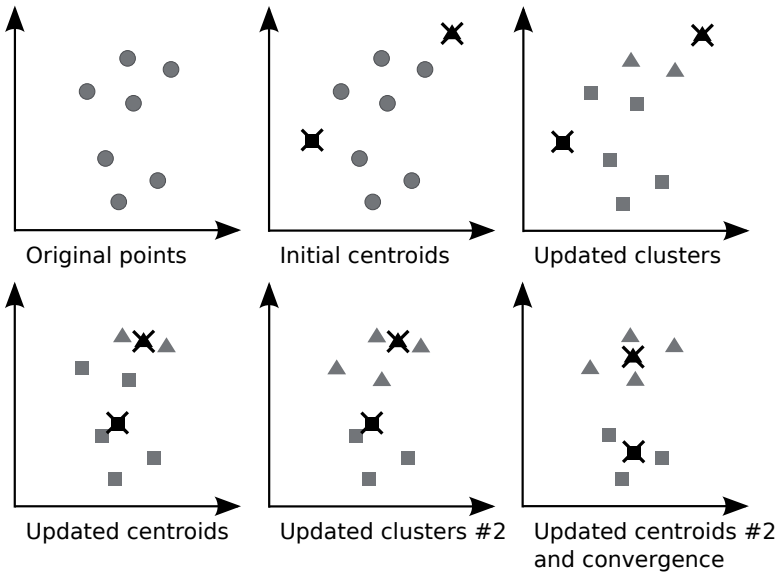

FIGURE 6 A simple example of k-means clustering. The clusters are separated using different symbols and the cluster centroids are black.

4. Repeat steps 2 and 3 until convergence takes place.

In practice convergence means meeting a certain stopping criterion, which usually is satisfied when there are no changes to clustering or centroids after the previous iteration. At this point, a local optimum solution will have been been found. A simple example with just few data points is illustrated in Figure 6.

Since k-means converges to a local minimum, cluster centroid initialization impacts the clustering result greatly. If the initialization is done well, it is possible to find a global minimum solution. Usually the aim is to find an initialization that gives the most stable clustering (Bubeck et al., 2012). The most common approach is to use random initialization (Forgy, 1965). However, this can sometimes lead to unsatisfactory results. Many initialization methods have been introduced in attempts to overcome this problem (Ostrovsky et al., 2006; Arthur and Vassilvitskii, 2007; Barakbah and Kiyoki, 2009; Bubeck et al., 2012).

Normally, before using k-means, some decision of the number of clusters (k) must be made. However, it is possible to use the algorithm even when k is unknown (Pham et al., 2005). In that case, a some kind of methodology for choosing the appropriate number of clusters can be used, e.g., Davies-Bouldin index (Davies and Bouldin, 1979) or the silhouette method (Rousseeuw, 1987).

Many extensions to k-means have been introduced to extend its capabilities and overcome certain problems and shortcomings. For example, Kernel k-means can be used even when the clusters are separated only nonlinearly in the original input space (Dhillon et al., 2004). In addition, k-medoids is a more robust alternative to k-means (Kaufman and Rousseeuw, 1987). It aims to minimize the sum of dissimilarities between cluster data points and its medoid.

3.4.2 Spectral clustering

Spectral clustering has become a very popular approach to clustering problems, because it is easy to implement and solve, and it often offers feasible results (Von Luxburg, 2007). Spectral clustering does not refer to a single algorithm, but rather a family of different algorithms that follow the same approach. For example, the algorithms can differ in the way they generate affinity matrices or how the Laplacian L is calculated.

Let us consider an example of a spectral clustering algorithm. Given a data matrix $\mathbf{X} = \mathbf{x}_1, \ldots, \mathbf{x}_n$ that we want to divide into k clusters, we can use the following algorithm (Ng et al., 2002):

1. Calculate the affinity matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$ so that $\mathbf{W} = \exp(-||x_i - x_j||^2/2\sigma^2)$ when $i \neq j$, $\mathbf{W}_{ii} = 0$. Here σ^2 is a scaling parameter that controls how quickly the affinity falls off as the distance between x_i and x_j increases.
2. Using diagonal matrix $\mathbf{D}$, whose element (i, i) is the sum of the row i in $\mathbf{W}$, calculate the matrix $\mathbf{L} = \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}$.
3. Find the k largest eigenvectors of $\mathbf{L}$, denoted by $\mathbf{u}_1, \ldots, \mathbf{u}_k$. Form the matrix $\mathbf{U} = [\mathbf{u}_1 \ldots \mathbf{u}_k] \in \mathbb{R}^{n \times k}$ so that eigenvectors are stacked in its columns.
4. Generate the matrix $\mathbf{T}$ by normalizing the rows of $\mathbf{U}$ to unit lengths by performing $\mathbf{T}_{ij} = \mathbf{U}_{ij}/(\sum_j \mathbf{U}_{ij}^2)^{\frac{1}{2}}$.
5. Using each row of $\mathbf{T}$ as a point in $\mathbb{R}^k$, cluster the points by using k-means or some other algorithm that tries to minimize distortion.
6. Assign the original data point $\mathbf{x}_i$ to cluster j if the row i of matrix $\mathbf{T}$ was assigned to cluster j.

It is important to note that even though the algorithm uses k-means or a similar algorithm in one of its steps, spectral clustering is not the same thing as using k-means clustering on its own (Ng et al., 2002).

We can clearly see that the diffusion map methodology is very similar to the above algorithm. They are indeed closely connected.

3.4.3 Self-organizing Map

Self-organizing map (SOM) (Kohonen, 1982) is a neural network based on unsupervised learning and creates a low-dimensional representation (map) of the input vectors. It aims to preserve the topological relations in the data, and it is a popular algorithm used in clustering and visualization (Shao et al., 2009). It also works well in dimensionality reduction applications. SOM has been researched extensively, with thousands of papers published on the subject (Kangas and Kaski, 1998).

The map can be constructed in the following way (Shao et al., 2009). Just as previously, we have the data points in matrix $\mathbf{X}$ in D dimensions and an individual point is $\mathbf{x}_j$. The basic version of SOM consists of a layer of neurons that exist on a low-dimensional lattice. The neurons are represented by weight vector

$\mathbf{w}_i = \{w_{i1}, \ldots, w_{iD}\}$. In the beginning of each training step, a single data point $\mathbf{x}_j$ is selected randomly. Subsequently, one of the neurons is selected as a best matching unit (BMU) c according to Equation 14.

$$||\mathbf{w}_c - \mathbf{x}_j|| = \min_i ||\mathbf{w}_i - \mathbf{x}_j|| \tag{14}$$

After this, the BMU and its neighboring nodes are updated and moved towards $\mathbf{x}_j$ by the following:

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \alpha(t) \cdot h_{ci}(t) \cdot (\mathbf{x}_j - \mathbf{w}_i(t)). \tag{15}$$

Now the $\alpha(t)$ is the learning rate and $h_{ci}(t)$ is a neighborhood kernel that defines a neurons neighborhood. One option for $h_{ci}(t)$ is:

$$h_{ci}(t) = e^{\frac{||p_c - p_i||^2}{2\sigma(t)^2}} \tag{16}$$

Where p_c is the position of the BMU in the lattice, p_i is the corresponding position of BMU's neighbor, and $\sigma(t)$ is the width of the neighborhood function. Both $\alpha(t)$ and $\sigma(t)$ need to decrease over time, otherwise the algorithm might not converge.

An example of SOM training with different number of iterations is presented in Figure 7. The units are moving and conforming to the data in the input space. In this example the data has only two dimensions.

A well-known method for visualizing and classifying data points into a cluster is the U-matrix (Ultsch and Siemon, 1990). U-matrix is constructed in a way that shows the distances between the nodes in the input space while using the low-dimensional output space. Different colors or a 3-D bump map can be used for the visualization. The points that fall into a "valley" on the map belong to the same cluster. Alternative methods for visualizing and detecting clusters include the Adaptive Coordinates and Cluster Connections method (Merkl and Rauber, 1997).

Over the years, many extensions to SOM have been introduced. One method creating a more efficient mapping is Growing Grid, which starts with a small rectangular grid of nodes (e.g., 2×2) and expands the grid by inserting rows or columns so that the map adapts to the shape of the analyzed data (Fritzke, 1995). However, this and other similar extensions do not take the inherent hierarchical structure of the data into account (Rauber et al., 2002). To overcome this problem, Growing Hierarchical Self-Organizing Map has been introduced (Dittenbach et al., 2000). It starts as a small map and generates new mappings with different sizes on different layers of hierarchy as needed. The sizes and structures of these independent mappings are determined by an unsupervised learning process.

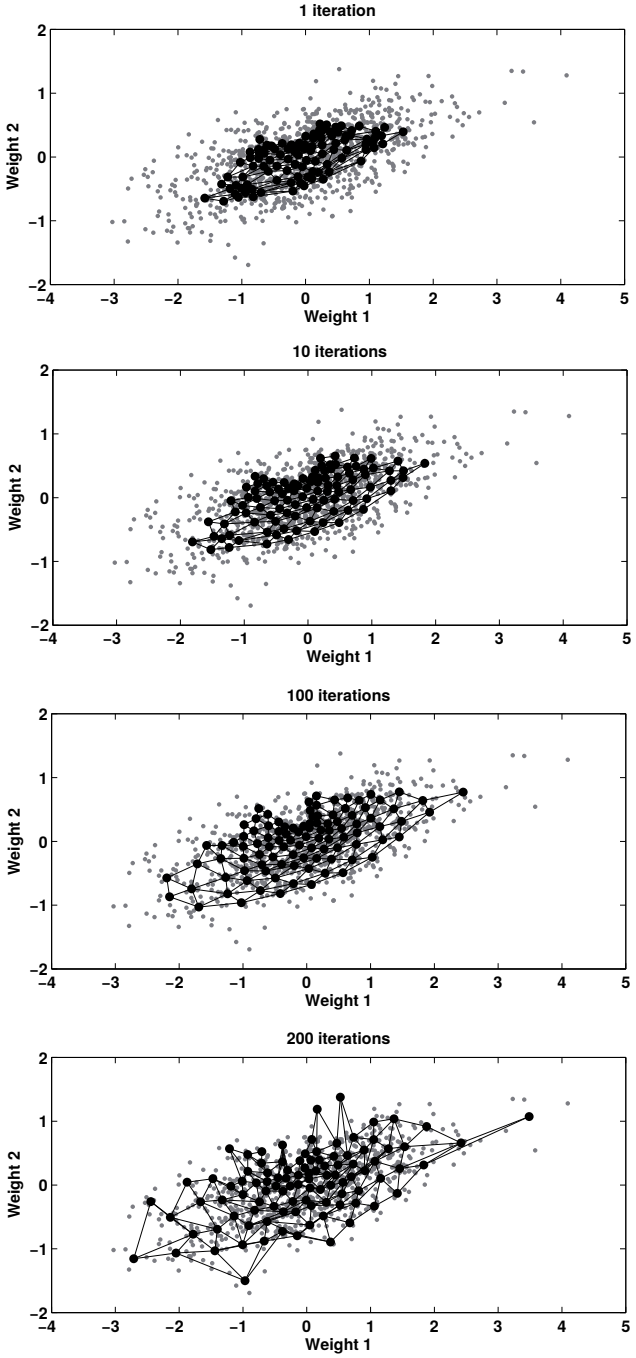

FIGURE 7 An example of SOM training with 1, 10, 100 and 200 iterations. The data points are in grey, the SOM units and their topological connections in black.

3.5 Anomaly detection

An anomaly is a pattern in data that does not conform to normal behavior or a region (Chandola et al., 2009). Anomalies are introduced for different reasons, e.g., by malicious network activity. Anomalies can also be single points or collective anomalies consisting of abnormal sequences of events. Many machine learning methods, such as the support vector machine (SVM) (Jiang and Yasakethu, 2013; Kim et al., 2014) and artificial neural networks (ANN) (Panchev et al., 2014) have been used to detect anomalies in various applications.

One way used in this study to detect anomalies uses a statistical anomaly detection method. First, some kind of anomaly score must be calculated for each point. This score can be, for example, the distance from the center of the normal cluster of points. If we assume that the data follows a Gaussian distribution, we can use an anomaly indicator function:

$$g(y) = \begin{cases} 1 & \text{, if } distance(y, center) > \mu + m\sigma \\ 0 & \text{otherwise.} \end{cases} \tag{17}$$

Here σ is the standard deviation and μ is the mean of the distances from the mean point of the data. If the data point in low dimensions y is more than $\mu + m\sigma$ away from the mean, the point is classified as an anomaly. The choice of m is not trivial, but in the case of a Gaussian distribution choosing $m = 3$ should cover 99.7% of the data. This kind of statistical distance-based method can be taken advantage of with the help of any dimensionality reduction method mentioned in Section 3.3.

The SOM algorithm is often considered a dimensionality reduction or a clustering algorithm, but it can also be used for anomaly detection. For example, a data point is classified as normal if it is close to its BMU, and as anomalous if the distance between the point and the BMU is larger than the pre-defined threshold (Ramadas et al., 2003). In addition to SOM, GHSOM has also been successfully used to solve anomaly detection problems (Huang and Huang, 2013). GHSOM is also used for network anomaly detection in this research in Article PIV.

3.6 Out-of-sample extension

If we consider dimensionality reduction, as explained in Section 3.3, as the training stage, meaning that the data present in the initial dimensionality reduction phase is the training data, we have to choose what happens when new data that was not present in training is introduced to the system. The new points must be projected to the same subspace as the original training data. This is known as out-of-sample extension. Some different methods for performing dimensionality reduction are explained in this section.

3.6.1 Random projection and PCA

The addition of new data points can be carried out very easily and similarly for both random projection and PCA. Random projection is especially simple, since it does not need a training data set at all. Random matrix $\mathbf{R}$ is generated independent of the data anyway. If we get a new data point $\mathbf{y}_i$, we can very simply project that point into the same subspace by performing the following multiplication:

$$\mathbf{y}_{RP} = \mathbf{y}_i\mathbf{R}. \tag{18}$$

In the simplest case, the same procedure for PCA is basically the same. If we have the ordered eigenvectors of the covariance matrix in the columns of matrix $\mathbf{W}$ as explained in 3.3.1, we can get a low-dimensional representation of the new data point in the following way:

$$\mathbf{y}_{PCA} = \mathbf{y}_i\mathbf{W}. \tag{19}$$

3.6.2 Diffusion maps and Nyström extension

When performing out-of-sample extension for diffusion map dimensionality reduction (explained in 3.3.3), the aim is to interpolate the coordinates of new data points that were unavailable at the training stage based on the mapping of the training data. One option is to use the Nyström extension (Belongie et al., 2002; Bengio et al., 2004; Fowlkes et al., 2004). It can be used together with many dimensionality reduction methods. In this research, it is coupled with diffusion maps.

Let us denote an original data point present in the training data as $\mathbf{x}_i$. Now a new added data point is $\mathbf{y}_j \in \mathbb{R}^D$, where D is the number of original high dimensions. We collect the distances between the new data point and existing training points into $\bar{\mathbf{W}}$ as shown in Equation 20.

$$\bar{\mathbf{W}} = \exp\left(-\frac{||\mathbf{x}_i - \mathbf{y}_j||^2}{\epsilon}\right) \tag{20}$$

Once we calculated the column sums of $\bar{\mathbf{W}}$ into diagonal matrix $\bar{\mathbf{D}}_{ii} = \sum_{i=1}^{N} \bar{\mathbf{W}}_{ij}$, the transition probabilities are

$$\mathbf{B} = \bar{\mathbf{W}}\bar{\mathbf{D}}^{-1}. \tag{21}$$

Then, we get the eigenvectors as columns of matrix $\bar{\mathbf{V}}$ (the eigenvalues $\mathbf{\Lambda}$ are the same as in the training phase):

$$\bar{\mathbf{V}} = \mathbf{B}^T\mathbf{V}\mathbf{\Lambda}^{-1}. \tag{22}$$

Now, if the low-dimensional coordinates for the new points are $\bar{\mathbf{Y}} = \bar{\mathbf{V}}\mathbf{\Lambda}$, we get the extended coordinate approximations in the columns of $\bar{\mathbf{Y}}_{DM}$ (Equation 23).

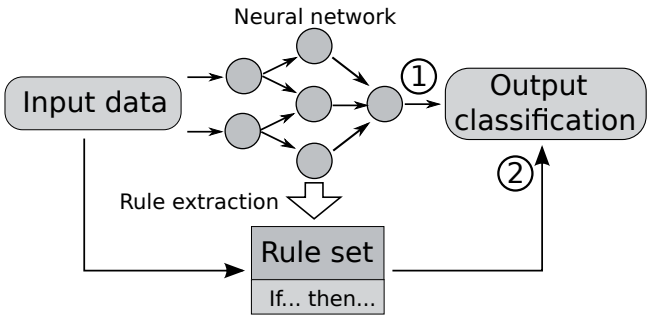

FIGURE 8 An example of the rule extraction process. Instead of using neural network classification (1), the rule set is generated and it performs the classification task (2).

$$\bar{\mathbf{Y}} = \mathbf{B}^T\mathbf{V} \tag{23}$$

3.6.3 Rule extraction

In the context of rule extraction, a rule can be defined as a symbolic rule that describes a certain classification result (Craven and Shavlik, 1994). Rule extraction is the rule generation phase. Once the rules have been extracted, the rule set is expected to replicate the classification results of any algorithm in an efficient way. Another motivation for using rule extraction is the fact that algorithms may work as a black box. Since in many applications it is important for the user to understand how the algorithm makes its decisions (Craven and Shavlik, 1994), well constructed symbolic rules can provide this information in a meaningful and understandable way. Some example algorithms and applications include generating rules from a support vector machine for medical diagnosis (Barakat and Diederich, 2004), rule extraction from trained neural networks for credit card fraud detection (Ryman-Tubb and d'Avila Garcez, 2010) and automatic medical database classification (De Falco, 2013). The rules can also be represented as decision trees (Craven and Shavlik, 1996).

Figure 8 illustrates the main idea behind rule extraction. A rule set is extracted from a neural network classification, and the rule set, instead of the network, is then used to save computational time and make the classification more understandable for humans. The rule set also facilitates out-of-sample extension, since new data points can be classified using the rules but without using a heavy classification algorithm. Even though rule extraction based on a neural network is very common, any classification algorithm can be used.

A rule extraction algorithm can be roughly divided into two main categories: decompositional and pedagogical (Andrews et al., 1995). Decompositional algorithms take the underlying classification algorithm into account. For example, if rules are extracted based on artificial neural network classification, different layers as well as hidden and output units of the network are used to

construct the rules. Pedagogical approaches treat the classification algorithm as a black box, meaning that only inputs and outputs are taken into account. Because of this, pedagogical methods are more general and can work with any classification methodology. On the other hand, the possible rule space can be large and this can lead to inefficient rule generation (Ryman-Tubb, 2011).

Let us consider an example of symbolic rules. If we have four binary features, a, b, c and d, this also means that we have four dimensions or four columns in the feature matrix. In this case, a symbolic rule determines whether a feature must be true, false or whether its value does not matter. Let's look at an example rule set:

$$\begin{cases} r_1 = a & \text{for class} \quad c_1, \\ r_2 = \neg a \wedge b \wedge c & \text{for class} \quad c_1, \\ r_3 = a \wedge \neg b \wedge d & \text{for class} \quad c_2. \end{cases}$$

All of the rules form whole rule set R. For example, rule r_2 means that a feature vector matches the rule if its symbol a is false or 0, its symbols b and c are 1, and the value of symbol d does not matter. Binary feature vectors matching rule r_2 would be $\mathbf{v}_1 = [0, 1, 1, 0]$ and $\mathbf{v}_2 = [0, 1, 1, 1]$. If a vector matches r_2, that data point will be classified to group c_1.

A simple way to construct this kind of rule set is conjunctive rule extraction (Craven and Shavlik, 1994). The algorithm requires training data as well as the classes obtained using some classification algorithm. The algorithm can be presented in a simplified way as follows:

1. Get observation $\mathbf{x}_i$ from data set $\mathbf{X}$. Use it as a basis for a new rule.
2. Check if the observation already matches a rule in rule set R. If not, continue to Step 3. If the observation is already covered by a rule, skip to Step 1.
3. Drop a symbol from $\mathbf{x}_i$. Again classify using the existing rule set. If the classification does not change, that symbol is not needed and can be omitted. This is repeated for all the symbols. After this only the necessary symbols are left and the rule pruned.
4. Add newly generated rule r_i to set R.
5. Repeat from Step 1 until all the training observations have been used.

In short, to construct new rules, this greedy algorithm uses all the observations that are not already covered by a rule. From these rules, all the unneeded symbols (the ones that do not affect the classification result in the training data) are removed. Most rule extraction algorithms are more complicated than this, but conjunctive rule extraction is simple and efficient, and it can be used with any classification algorithm.

4 RESULTS

This chapter presents the results obtained. The results are presented using a bottom-up approach, meaning that the final overall framework and findings are presented in the end of the chapter, because this approach follows the chronological order of the research papers more accurately.

4.1 Anomaly detection from real network data

All of the included articles deal with real-world network log data. This is one of the most important aspects since any intrusion detection system needs to be evaluated using real data to ensure its feasibility (Sommer and Paxson, 2010). Web server logs were acquired from several companies, and they use the following format:

```
127.0.0.1 - -
[01/January/2012:00:00:01 +0300]
"GET /resource.php?
parameter1=value1&parameter2=value2
HTTP/1.1" 200 2680
"http://www.address.com/webpage.html"
"Mozilla/5.0 (SymbianOS/9.2;...)"
```

The format includes information such as the IP address and timestamp, but the most interesting part is HTTP query and it's parameters, since it is possible to inject malicious code into them or detect other intrusions from the query part. For example, an attacker could inject SQL statements into the request parameter values to log in to a system without knowing the password, or to wipe out a database completely (SQL injection). In addition, more complex intrusions could leave traces into HTTP logs. Many password crackers and vulnerability scanning software, which are meant to be used for security auditing, are actually used for malicious scanning and will leave entries in the log files.

Article PI lays the foundations for anomaly intrusion detection. In the paper, we analyze the above-mentioned real-life log data with the help of n-gram analysis (as explained in Section 3.2.2), project the points to low-dimensional space by using diffusion maps, and detect anomalies with the help of spectral clustering. Normal traffic forms a dense cluster and anomalous traffic is rather simple to separate in this case. The results are compared to those of support vector machine (SVM), and all the methods give practically feasible outputs, since all of the real intrusions are found. Because all the tested methodologies perform well, we can conclude that the n-gram feature extraction creates a data set where separating normal from anomalous is rather simple.

The results are extended in Article PII. The focus is on further comparison between DM and PCA methodologies and on using more real data for better testing of the efficiency of the system. For this purpose, two different data sets are used. The first one is a smaller one and is labeled manually to measure the accuracy. The second data set is totally unknown, and we use a more exploratory approach for this. For both data sets, real intrusions are found and representative visualizations are acquired. The most important observation is related to the two dimensionality reduction methods, DM and PCA. We find that, for the first data set, both methodologies give almost identical results. At this point, PCA seems a better choice due to its better performance. However, the second data reveals that PCA gives unsatisfactory results while DM works much better. This might indicate that there are nonlinear dependencies in the data, and PCA does not work in that case (Lee and Verleysen, 2007).

In Article PIII, the analysis is extended with a more elaborate cluster analysis using k-means algorithm. Even though k-means has its limitations, there is justification for its use in combination with diffusion maps (Lafon and Lee, 2006). Again, the comparison between DM and PCA gives results similar to those already found in the previous papers. One of the new discoveries is that one data file is separated into two clear clusters with similar size. In this case, we found that a visualization can represent an underlying traffic structure that cannot be directly seen by looking at the log files. This extends the system's capabilities beyond anomaly detection.

Article PIV continues the analysis of HTTP logs with n-gram preprocessing. This time we a SOM extension known as growing hierarchal self-organizing map (GHSOM). The algorithm dynamically extends SOM by starting with a small map and hierarchically growing it. GHSOM is used for detecting anomalous HTTP queries. In addition, another detection method is used in combination with GHSOM detection. a statistical distribution model is applied to find abnormal HTTP headers by analyzing header lengths and non-alphanumeric symbols that appear in them (Corona and Giacinto, 2010). Both methods when used together increase the accuracy of detection. Many different types of intrusions, such as SQL injections, directory traversals or buffer overflow attacks are detected with good accuracy.

In the articles mentioned above, different methods are used for real-life log analysis and anomaly detection. While the accuracy is good, there are still some

concerns related to the speed and efficiency of the system, and new data points must be added dynamically. These problems are addressed in the following section.

4.2 Increasing performance and adding new data points

The anomaly detection framework gives good results, but the analysis and addition of new incoming data points might not be fast enough. Article PV addresses this problem by introducing conjunctive rule extraction, explained in Section 3.6.3. With unsupervised approach using the same preprocessing as before and diffusion maps, training traffic is first classified into normal and abnormal. The input data and its classification result is then used as a basis for a conjunctive rule extraction algorithm (Craven and Shavlik, 1994). The end result is a simple $if \ldots then$ rule set that classifies any incoming traffic. This approach offers the following benefits:

1. Rule based classification is much faster the use of a heavy machine learning algorithm
2. The classification results approximate the algorithm classification, so detection of intrusions and anomalies is efficient
3. The rules can be manually inspected, offering insight to the system administrator

The advantage of the first point is clear, as we can use complex algorithms for initial training and then classify the traffic with a compact rule set. The benefit referred to in the second point is huge, but it can be argued that if the training data set used to create the rules is different from new incoming traffic, classification accuracy suffers dramatically. Finally, as explained in the third point, the rules can sometimes give the user more detailed information about the classification than an application of a "black box" algorithm. However, this also depends largely on the data, and the rules can be too complex to be readable for humans.

Despite its benefits, rule extraction also has some problems that limit its use in intrusion detection. It might be beneficial to look for performance increase elsewhere. In Article PVI, instead of using a heavy algorithm and rule extraction, random projection (RP) dimensionality reduction is used to perform similar tasks but with less computational complexity. HTTP logs with n-gram feature extraction produce very sparse feature matrices, since the probability for any individual n-gram appearing in a single log file is low, due to the large number of possible n-grams. This means that most entries in the matrix are zeros. RP works especially well in these situations (Papadimitriou et al., 1998). Computational times are also significantly faster than with PCA. Anomalies are detected using a distance-based anomaly score, so that data points with score higher than the threshold value are classified as anomalies. Out of more than a million log lines, only 0.02% are flagged as anomalies and actual intrusion attempts are found.

However, more experiments are needed to ensure the accuracy of the RP method. In addition, since there is a random element apparent in the RP algorithm, the results are not always stable. This could be solved by using multiple RP mappings and combining the results (Fern and Brodley, 2003).

Article PVII combines the RP, PCA and DM methodologies and addition of new data points to each of them and compares the results. For RP and PCA, new data point addition is a simple matter of matrix and vector multiplication. For DM, we need an algorithm such as Nyström extension. This requires more calculation, but it accurately projects new points into the low-dimensional subspace. Statistical anomaly detection is used for all the methods, and test results are obtained using a simulated data set as well as a real-world one. Real intrusions are once again found, and based on the findings it is clear that RP scales with large data sets much better than the other methods do. Another major result is that all the methods scale linearly with out-of-sample extension. This means the system could be scaled up to potentially be used in big data applications.

4.2.1 New unpublished results

Some new but unpublished results have been obtained to support the findings in Article PV. To supplement a real but undisclosed log data set, we used the popular KDD Cup 99 data (Bache and Lichman, 2013). The data set does have many problems and limitations, e.g., the data is completely artificial and synthetic and might not represent real network traffic, and there are many redundant records which is problematic for machine learning algorithms (Tavallaee et al., 2009). On the other hand, the data set is widely used and is useful for comparisons.

First, the original measurement data is input to the DM algorithm and clustered to normal and anomalous sets with k-means. Since conjunctive rule extraction requires binary features and some of the measurements in KDD Cup data are continuous, the data is binarized by dividing individual features into n bins (e.g., $n = 10$) and using the binary features to describe into which bin the value of a specific feature falls. Some information is lost during binarization, but the binary features are only used for rule generation.

After performing unsupervised classification and rule extraction, the system is tested with new data. Some results for a small test set can be seen in Table 3, including normal performance metrics with the addition of the Matthews correlation coefficient (Matthews, 1975). The corresponding confusion matrix is shown in Table 4.

One important notion is that the rule based classification gets slower as the rule set size increases. The rule set growth behavior was tested using more data. The growth of the set size is seen in Figure 9. As expected, when more and more data is added, it seems that the rule set size converges and does not increase after a certain point, i.e., all the new incoming data points are already covered by a finite rule set. Regardless of the promising results, rule extraction and its accuracy need more validation before the feasibility is confirmed in a more general setting.

TABLE 3 Performance metrics for the testing KDD data lines.

Metric	Value %
Sensitivity (TPR)	98.44
FPR	1.25
Specificity (TNR)	98.75
Accuracy	98.50
Precision	99.70
Matthews corr. coef.	95.31

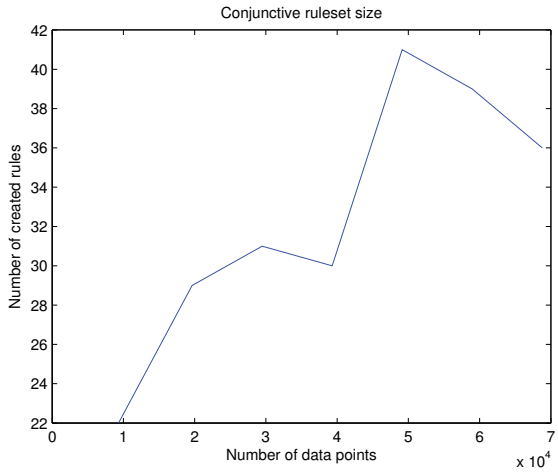

FIGURE 9 The rule set size increase with increasing number of log lines.

TABLE 4 Confusion matrix for testing data lines.

		predicted	
		normal	attack
actual	normal	947	12
	attack	63	3,978

4.3 Combined overall framework

Finally, the components explained in the articles can be combined to form the overall system framework, which is the major contribution of this thesis. Figure 10 shows the framework. The idea is that there are many "paths" that all finally lead to real-time intrusion and anomaly detection. Based on results found in the literature and in the included articles in this thesis, there are rules and justifications for choosing specific components. First, preprocessing is quite similar in all the situations. Second, one of the dimensionality reduction methods must be selected. PCA is a tried and true method and can be treated as a default option. If the data is very sparse (contains many zero values) or high speed is required, RP is the logical choice. On the other hand, if the data contains nonlinear dependencies and other methods give unfeasible results, DM is the best choice for accurate results. GHSOM methodology combines dimensionality reduction and clustering, and is the ideal option if the data has natural hierarchical properties. After dimensionality reduction, there is the clustering phase. With DM, spectral clustering or k-means will both work since there is justification for their use found in the literature and their use is supported by our results as well. With RP or PCA, k-means is a good choice, even though clustering is optional. Finally, anomalies are detected using statistical anomaly detection or a SOM-based anomaly detection technique. Using multiplication method (RP and PCA) or Nyström extension (DM), new data can be added after training. Rule extraction is an option that can be used with every algorithm. The end result is a real-time anomaly detection framework that has the capabilities to work in a variety of situations with different kinds of data.

No matter which of the presented algorithms are selected, the overall system follows the steps described in the KDD process mentioned in Section 3.1. Table 5 shows how the different phases of the process are applied to intrusion detection in this research.

The final resulting anomaly detection system contains preprocessing, dimensionality reduction, clustering, anomaly detection and out-of-sample extension components. The system offers choices for algorithms mentioned previously, and the selection can be based on different rules and justifications depending on

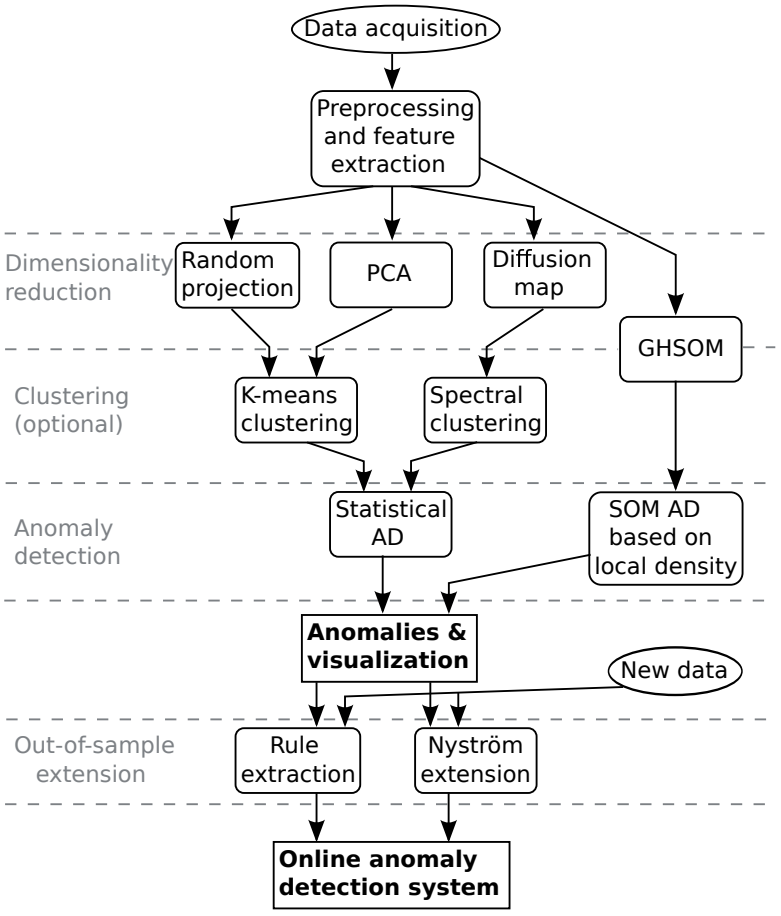

FIGURE 10 The entire anomaly detection framework.

TABLE 5 A comparison of how the different phases of the KDD process are used in this research.

KDD model step	**This research**
Selection	Data acquisition Selecting a subset of features
Preprocessing	Removal of unneeded lines Handling of missing data N-gram analysis
Transformation	Dimensionality reduction: RP, PCA, DM
Data mining	Clustering: K-means Spectral clustering SOM
	Anomaly detection: SOM Statistical
	Out-of-sample extension: OOS for RP, PCA Nyström extension
Interpretation/evaluation	Alerts Actions

the data. The system also follows the KDD process, and has the following main benefits:

- Anomalies are detected automatically
- Representative visualizations are created
- New data points can be added dynamically when new traffic is generated in real-time
- The system supports large data sets
- Choosing the right algorithm from the overall framework is not random, and can be based on existing theoretical and practical knowledge and justifications
- The system is tested and works with real network data

The system is almost exclusively tested on real data, since it was available from real companies and any intrusion detection system must be validated using real data (Sommer and Paxson, 2010). The aim was not to make the system use the most complex algorithms, but create a framework that works in a practical situation. It can be argued that the system in Figure 10 is too complex, but not all of the algorithms are used at the same time, making the end results simpler in a practical scenario. Currently the algorithm selection process is not automated, which is a shortcoming since some data mining knowledge is needed to optimize the system's efficiency. However, the presented guidelines help in this matter.

5 CONCLUSION

The amount of network traffic has increased, giving rise to massive amounts of data. In addition, web services and systems have become more complex, and as a result they face new attacks threatening their security. For this reason, more intelligent data analysis, anomaly detection and visualization methods are needed to facilitate dynamic online intrusion detection.

This thesis approaches the above-described problem from two main viewpoints. Firstly, the knowledge discovery process is used as a theoretical background framework along with relevant data mining algorithms. Secondly, the research is closely tied to practical situations and real-world data, including acquiring real log data sets and evaluating the systems feasibility in a practical scenario.

Practical experiments include evaluation of the framework with data acquisition, preprocessing, dimensionality reduction and anomaly detection. The proposed methods used together form a dynamic system that is capable of detecting network anomalies from real data, as well as providing meaningful visualizations that represent the structure of the data. Different algorithms can be selected based on the nature and characteristics of the data. The results emphasize the practical usefulness and impact of the system instead of slightly better detection metrics.

Despite the useful results from real data, the undisclosed nature of the data sets makes it difficult to reproduce the results in different settings. Therefore, more evaluation with the help of the whole framework is needed to ensure the feasibility, even if the smaller components presented in the articles already give good results. In addition, automatic selection of the proper algorithms should be added to make the system easier to use for network administrators who are not data mining experts. The automated algorithm selection could make the system slower at least in the initial training phase. For future research, the system should be adapted for massive big-data analysis to make it more relevant for current needs.

YHTEENVETO (FINNISH SUMMARY)

Tämä väitöskirja, *Hyökkäysten havainnoinnin sovellutuksia käyttäen tietämyksen löytämisprosessia ja tiedonlouhintaa*, käsittelee automaattista hyökkäysten löytämistä suurista tietoliikennemassoista käyttäen hyväksi tietämyksen löytämisen prosessin vaiheita ja ennen kaikkea tiedonlouhinta-algoritmeja. Tietoverkkojen ja verkkopalveluiden kasvava liikennemäärä ja monimutkaisuus ovat avanneet mahdollisuuksia uusille hyökkäyksille. Monet hyökkäyksistä ovat ennestään tuntemattomia, ja ne pystytään havaitsemaan vasta vahinkojen tapahduttua. Tämä tutkimus esittelee ja yhdistelee tapoja löytää nämä hyökkäykset automaattisesti tiedonlouhintaa hyväksikäyttäen.

Tutkimus esittelee ensin teoreettisen pohjatiedon, joka sisältää tietoa hyökkäysten havaitsemisjärjestelmistä ja hyökkäyksistä itsestään, sekä esittelee tietämyksenlöytämisprosessin ja erityisesti siihen olennaisena osana kuuluvia tiedonlouhinta-algoritmeja. Analyysi sisältää datan keräämisen, sen esikäsittelyn, ulottuvuuksien pienentämisen ja ryhmittelyn eli klusteroinnin sekä poikkeavuuksien havaitsemisen. Nämä kaikki vaiheet alusta loppuun voidaan suorittaa täysin automaattisesti siten, että liikennedataa analysoidaan sitä mukaa kun sitä kertyy. Tiedonlouhintavaiheessa käytetään esimerkiksi diffuusiokarttaa, pääkomponenttianalyysia, satunnaisprojektiota, k-means-klusterointia ja itseorganisoituvaa karttaa analyysin eri vaiheissa.

Tutkimus testaa järjestelmän käytettävyyttä käyttäen oikean maailman lokidataa. Esimerkkitapauksissa löydetään oikeita hyökkäysyrityksiä ja pystytään visualisoimaan aluksi korkeaulotteista tietoa käyttäen ulottuvuuksien vähentämisalgoritmeja. Tuloksena ovat hälytykset uusista hyökkäyksistä sekä kaksi- tai kolmeulotteinen kuva liikenteen rakenteesta. Käytetyt algoritmit muodostavat käytännöllisen järjestelmän, joka lisää turvallisuutta ja tarjoaa lisää tietämystä verkon ylläpitäjälle. Suuria datamassoja voidaan analysoida automaattisesti siten, että järjestelmä toimii käytännön tasolla ja antaa lisäarvoa sen käyttäjälle.

REFERENCES

Abou-Assaleh, T., Cercone, N., Keselj, V. & Sweidan, R. 2004. N-gram-based detection of new malicious code. In Computer Software and Applications Conference, 2004. COMPSAC 2004. Proceedings of the 28th Annual International, Vol. 2. IEEE, 41–42.

Achlioptas, D. 2001. Database-friendly random projections. In Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems. ACM, 274–281.

Andrews, R., Diederich, J. & Tickle, A. B. 1995. Survey and critique of techniques for extracting rules from trained artificial neural networks. Knowledge-based systems 8 (6), 373–389.

Arthur, D. & Vassilvitskii, S. 2007. k-means++: The advantages of careful seeding. In Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics, 1027–1035.

Asaka, M., Taguchi, A. & Goto, S. 1999. The implementation of ida: An intrusion detection agent system. In Proceedings of the 11th FIRST Conference, Vol. 6. Citeseer.

Bache, K. & Lichman, M. 2013. UCI Machine Learning Repository. ⟨URL:http://archive.ics.uci.edu/ml⟩.

Barakat, N. & Diederich, J. 2004. Learning-based rule-extraction from support vector machines. In The 14th International Conference on Computer Theory and applications ICCTA'2004.

Barakbah, A. R. & Kiyoki, Y. 2009. A pillar algorithm for k-means optimization by distance maximization for initial centroid designation. In Computational Intelligence and Data Mining, 2009. CIDM'09. IEEE Symposium on. IEEE, 61–68.

Belongie, S., Fowlkes, C., Chung, F. & Malik, J. 2002. Spectral partitioning with indefinite kernels using the nyström extension. In Computer Vision—ECCV 2002. Springer, 531–542.

Bengio, Y., Paiement, J.-F., Vincent, P., Delalleau, O., Le Roux, N. & Ouimet, M. 2004. Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering. Advances in neural information processing systems 16, 177–184.

Bingham, E. & Mannila, H. 2001. Random projection in dimensionality reduction: applications to image and text data. In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 245–250.

Bubeck, S., Meilă, M. & von Luxburg, U. 2012. How the initialization affects the stability of the k-means algorithm. ESAIM: Probability and Statistics 16, 436–452.

Chandola, V., Banerjee, A. & Kumar, V. 2009. Anomaly detection: A survey. ACM Computing Surveys (CSUR) 41 (3), 15.

Coifman, R. R., Lafon, S., Lee, A. B., Maggioni, M., Nadler, B., Warner, F. & Zucker, S. W. 2005. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. Proceedings of the National Academy of Sciences of the United States of America 102 (21), 7426–7431.

Coifman, R. R. & Lafon, S. 2006. Diffusion maps. Applied and computational harmonic analysis 21 (1), 5–30.

Cole, E. 2012. Advanced Persistent Threat: Understanding the Danger and How to Protect Your Organization. Newnes.

Corona, I. & Giacinto, G. 2010. Detection of server-side web attacks. In Proceedings of JMLR: Workshop on Applications of Pattern Analysis, 160–166.

Craven, M. & Shavlik, J. W. 1994. Using sampling and queries to extract rules from trained neural networks. In Proceedings of the Eleventh International Conference on Machine Learning. Morgan Kaufmann, 37–45.

Craven, M. W. & Shavlik, J. W. 1996. Extracting tree-structured representations of trained networks. Advances in neural information processing systems, 24–30.

Damashek, M. et al. 1995. Gauging similarity with n-grams: Language-independent categorization of text. Science 267 (5199), 843–848.

David, G., Averbuch, A. & Coifman, R. R. 2010. Hierarchical clustering via localized diffusion folders. In Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI) Symposium Series.

David, G. & Averbuch, A. 2012. Hierarchical data organization, clustering and denoising via localized diffusion folders. Applied and computational harmonic analysis 33 (1), 1–23.

David, G. 2009. Anomaly Detection and Classification via Diffusion Processes in Hyper-Networks. Tel-Aviv University. Ph. D. Thesis.

Davies, D. L. & Bouldin, D. W. 1979. A cluster separation measure. Pattern Analysis and Machine Intelligence, IEEE Transactions on 2, 224–227.

Denning, D. E. 1987. An intrusion-detection model. Software Engineering, IEEE Transactions on 2, 222–232.

De Falco, I. 2013. Differential evolution for automatic rule extraction from medical databases. Applied Soft Computing 13 (2), 1265–1283.

Dhillon, I. S., Guan, Y. & Kulis, B. 2004. Kernel k-means: spectral clustering and normalized cuts. In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 551–556.

Dittenbach, M., Merkl, D. & Rauber, A. 2000. The growing hierarchical self-organizing map. In Neural Networks, IEEE-INNS-ENNS International Joint Conference on, Vol. 6, 15–19.

Engen, V. 2010. Machine learning for network based intrusion detection: an investigation into discrepancies in findings with the KDD cup'99 data set and multi-objective evolution of neural network classifier ensembles from imbalanced data. Bournemouth University. Ph. D. Thesis.

Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P. et al. 1996. Knowledge discovery and data mining: Towards a unifying framework. In KDD, Vol. 96, 82–88.

Fern, X. Z. & Brodley, C. E. 2003. Random projection for high dimensional data clustering: A cluster ensemble approach. In ICML, Vol. 3, 186–193.

Forgy, E. W. 1965. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. Biometrics 21, 768–769.

Fowlkes, C., Belongie, S., Chung, F. & Malik, J. 2004. Spectral grouping using the nystrom method. Pattern Analysis and Machine Intelligence, IEEE Transactions on 26 (2), 214–225.

Fritzke, B. 1995. Growing grid—a self-organizing network with constant neighborhood range and adaptation strength. Neural Processing Letters 2 (5), 9–13.

Ganapathiraju, M., Weisser, D., Rosenfeld, R., Carbonell, J., Reddy, R. & Klein-Seetharaman, J. 2002. Comparative n-gram analysis of whole-genome protein sequences. In Proceedings of the second international conference on Human Language Technology Research. Morgan Kaufmann Publishers Inc., 76–81.

Hecht-Nielsen, R. 1994. Context vectors: general purpose approximate meaning representations self-organized from raw data. Computational intelligence: Imitating life, 43–56.

Hirsimaki, T., Pylkkonen, J. & Kurimo, M. 2009. Importance of high-order n-gram models in morph-based speech recognition. Audio, Speech, and Language Processing, IEEE Transactions on 17 (4), 724–732.

Hotelling, H. 1933. Analysis of a complex of statistical variables into principal components. Journal of educational psychology 24 (6), 417.

Houle, M. E., Kriegel, H.-P., Kröger, P., Schubert, E. & Zimek, A. 2010. Can shared-neighbor distances defeat the curse of dimensionality? In Scientific and Statistical Database Management. Springer, 482–500.

Huang, S.-Y. & Huang, Y.-N. 2013. Network traffic anomaly detection based on growing hierarchical som. In Dependable Systems and Networks (DSN), 2013 43rd Annual IEEE/IFIP International Conference on. IEEE, 1–2.

Hubballi, N., Biswas, S. & Nandi, S. 2010. Layered higher order n-grams for hardening payload based anomaly intrusion detection. In Availability, Reliability, and Security, 2010. ARES'10 International Conference on. IEEE, 321–326.

Hubballi, N., Biswas, S. & Nandi, S. 2011. Sequencegram: n-gram modeling of system calls for program based anomaly detection. In Communication Systems and Networks (COMSNETS), 2011 Third International Conference on. IEEE, 1–10.

Jain, A. K. & Dubes, R. C. 1988. Algorithms for clustering data. Prentice-Hall, Inc.

Jain, A. K. 2010. Data clustering: 50 years beyond k-means. Pattern Recognition Letters 31 (8), 651–666.

Jiang, J. & Yasakethu, L. 2013. Anomaly detection via one class svm for protection of scada systems. In Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2013 International Conference on. IEEE, 82–88.

Johnson, W. B. & Lindenstrauss, J. 1984. Extensions of lipschitz mappings into a hilbert space. Contemporary mathematics 26 (189-206), 1.

Jolliffe, I. 2002. Principal component analysis (2nd edition). Springer New York. Springer Series in Statistics.

Kangas, J. & Kaski, S. 1998. 3043 works that have been based on the self-organizing map (SOM) method developed by Kohonen. Helsinki University of Technology.

Kaufman, L. & Rousseeuw, P. 1987. Clustering by means of medoids. Statistical Data Analysis Based on the L1 Norm and Related Methods, 405–416.

Kemmerer, R. A. & Vigna, G. 2002. Intrusion detection: A brief history and overview (supplement to computer magazine). Computer 35 (4), 27–30.

Kim, G., Lee, S. & Kim, S. 2014. A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. Expert Systems with Applications 41 (4), 1690–1700.

Klosgen, W. & Zytkow, J. 1996. Knowledge discovery in databases terminology. Advances in Knowledge Discovery and Data Mining, 573–592.

Kohonen, T. 1982. Self-organized formation of topologically correct feature maps. Biological Cybernetics 43 (1), 59-69.

Kumar, S. & Spafford, E. H. 1994. An application of pattern matching in intrusion detection, The COAST Project. Purdue University.

Kurgan, L. A. & Musilek, P. 2006. A survey of knowledge discovery and data mining process models. The Knowledge Engineering Review 21 (01), 1–24.

Lafon, S. & Lee, A. B. 2006. Diffusion maps and coarse-graining: A unified framework for dimensionality reduction, graph partitioning, and data set parameterization. Pattern Analysis and Machine Intelligence, IEEE Transactions on 28 (9), 1393–1403.

Lee, J. A. & Verleysen, M. 2007. Nonlinear dimensionality reduction. Springer.

Li, P., Hastie, T. J. & Church, K. W. 2006. Very sparse random projections. In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 287–296.

Lippmann, R. P., Fried, D. J., Graf, I., Haines, J. W., Kendall, K. R., McClung, D., Weber, D., Webster, S. E., Wyschogrod, D., Cunningham, R. K. et al. 2000. Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation. In DARPA Information Survivability Conference and Exposition, 2000. DISCEX'00. Proceedings, Vol. 2. IEEE, 12–26.

Lukka, K. 2006. Konstruktiivinen tutkimusote: luonne, prosessi ja arviointi. Rolin, K., Kakkuri-Knuuttila, ML ja Henttonen, E.(Ed.) Soveltava yhteiskuntatiede ja filosofia. Gaudeamus, Helsinki.

Lunt, T. 1993. Detecting intruders in computer systems. In In Proceedings of the 1993 Conference on Auditing and Computer Technology. Citeseer.

Lunt, T. F. 1990. Ides: An intelligent system for detecting intruders. In Proceedings of the Symposium: Computer Security, Threat and Countermeasures, 30–45.

Matthews, B. W. 1975. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. Biochimica et Biophysica Acta (BBA)-Protein Structure 405 (2), 442–451.

Merkl, D. & Rauber, A. 1997. Alternative ways for cluster visualization in self-organizing maps. In Proceedings of the Workshop on Self-Organizing Maps (WSOM97). Citeseer, 4–6.

Muller, K., Mika, S., Ratsch, G., Tsuda, K. & Scholkopf, B. 2001. An introduction to kernel-based learning algorithms. Neural Networks, IEEE Transactions on 12 (2), 181–201.

Nadler, B., Lafon, S., Coifman, R. & Kevrekidis, I. G. 2008. Diffusion maps-a probabilistic interpretation for spectral embedding and clustering algorithms. In Principal manifolds for data visualization and dimension reduction. Springer, 238–260.

Ng, A. Y., Jordan, M. I., Weiss, Y. et al. 2002. On spectral clustering: Analysis and an algorithm. Advances in neural information processing systems 2, 849–856.

Ostrovsky, R., Rabani, Y., Schulman, L. J. & Swamy, C. 2006. The effectiveness of lloyd-type methods for the k-means problem. In Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on. IEEE, 165–176.

Palnaty, R. P. & Rao, A. 2013. Jcads: Semi-supervised clustering algorithm for network anomaly intrusion detection systems. In Advanced Computing Technologies (ICACT), 2013 15th International Conference on. IEEE, 1–5.

Panchev, C., Dobrev, P. & Nicholson, J. 2014. Detecting port scans against mobile devices with neural networks and decision trees. In Engineering Applications of Neural Networks. Springer, 175–182.

Papadimitriou, C. H., Tamaki, H., Raghavan, P. & Vempala, S. 1998. Latent semantic indexing: A probabilistic analysis. In Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems. ACM, 159–168.

Parashar, A., Saurabh, P. & Verma, B. 2013. A novel approach for intrusion detection system using artificial immune system. In Proceedings of All India Seminar on Biomedical Engineering 2012 (AISOBE 2012). Springer, 221–229.

Patcha, A. & Park, J.-M. 2007. An overview of anomaly detection techniques: Existing solutions and latest technological trends. Computer Networks 51 (12), 3448–3470.

Pearson, K. 1901. Liii. on lines and planes of closest fit to systems of points in space. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2 (11), 559–572.

Pham, D. T., Dimov, S. S. & Nguyen, C. 2005. Selection of k in k-means clustering. Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 219 (1), 103–119.

Piirainen, K. A. & Gonzalez, R. A. 2013. Seeking constructive synergy: design science and the constructive research approach. In Design Science at the Intersection of Physical and Virtual Design. Springer, 59–72.

Pudil, P. & Novovičová, J. 1998. Novel methods for feature subset selection with respect to problem knowledge. In Feature Extraction, Construction and Selection. Springer, 101–116.

Ramadas, M., Ostermann, S. & Tjaden, B. 2003. Detecting anomalous network traffic with self-organizing maps. In Recent Advances in Intrusion Detection. Springer, 36–54.

Rauber, A., Merkl, D. & Dittenbach, M. 2002. The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data. Neural Networks, IEEE Transactions on 13 (6), 1331–1341.

Rencher, A. C. & Christensen, W. F. 2012. Methods of multivariate analysis, Vol. 709. John Wiley & Sons.

Rousseeuw, P. J. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of computational and applied mathematics 20, 53–65.

Roweis, S. T. & Saul, L. K. 2000. Nonlinear dimensionality reduction by locally linear embedding. Science 290 (5500), 2323–2326.

Ryman-Tubb, N. F. & d'Avila Garcez, A. 2010. Soar—sparse oracle-based adaptive rule extraction: Knowledge extraction from large-scale datasets to detect credit card fraud. In Neural Networks (IJCNN), The 2010 International Joint Conference on. IEEE, 1–9.

Ryman-Tubb, N. F. 2011. Neural-symbolic processing in business applications: Credit card fraud detection. Computational Neuroscience for Advancing Artificial Intelligence: Models, Methods and Applications, 270.

Schclar, A., Averbuch, A., Rabin, N., Zheludev, V. & Hochman, K. 2010. A diffusion framework for detection of moving vehicles. Digital Signal Processing 20 (1), 111–122.

Shao, C., Zhang, X., Wan, C. & Shang, W. 2009. A som-based method for manifold learning and visualization. In Computational Sciences and Optimization, 2009. CSO 2009. International Joint Conference on, Vol. 2, 312-316.

Shearer, C. 2000. The crisp-dm model: the new blueprint for data mining. Journal of data warehousing 5 (4), 13–22.

Sommer, R. & Paxson, V. 2010. Outside the closed world: On using machine learning for network intrusion detection. In Security and Privacy (SP), 2010 IEEE Symposium on. IEEE, 305–316.

Suen, C. Y. 1979. N-gram statistics for natural language understanding and text processing. Pattern Analysis and Machine Intelligence, IEEE Transactions on 2, 164–172.

Tankard, C. 2011. Advanced persistent threats and how to monitor and deter them. Network security 2011 (8), 16–19.

Tartakovsky, A. G., Polunchenko, A. S. & Sokolov, G. 2013. Efficient computer network anomaly detection by changepoint detection methods. Selected Topics in Signal Processing, IEEE Journal of 7 (1), 4–11.

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 Symposium on Computational Intelligence for Security and Defence Applications 2009.

Ultsch, A. & Siemon, H. P. 1990. Kohonen's self organizing feature maps for exploratory data analysis. In Proceedings of the International Neural Network Conference (INNC-90). Kluwer, 305–308.

Virvilis, N. & Gritzalis, D. 2013. The big four-what we did wrong in advanced persistent threat detection? In Availability, Reliability and Security (ARES), 2013 Eighth International Conference on. IEEE, 248–254.

Von Luxburg, U. 2007. A tutorial on spectral clustering. Statistics and computing 17 (4), 395–416.

Whitman, M. & Mattord, H. 2011. Principles of information security. Cengage Learning.

Witten, I. H. & Frank, E. 2005. Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann.

ORIGINAL PAPERS

PI

ANOMALY DETECTION FROM NETWORK LOGS USING DIFFUSION MAPS

by

Tuomo Sipola, Antti Juvonen and Joel Lehtonen 2011

Engineering Applications of Neural Networks, IFIP Advances in Information and Communication Technology, Vol. 363, pp. 172–181

Anomaly Detection from Network Logs Using Diffusion Maps

Tuomo Sipola, Antti Juvonen, and Joel Lehtonen*

Department of Mathematical Information Technology
University of Jyväskylä, Finland
tuomo.sipola@jyu.fi antti.juvonen@jyu.fi joel.lehtonen@iki.fi

Abstract. The goal of this study is to detect anomalous queries from network logs using a dimensionality reduction framework. The fequencies of 2-grams in queries are extracted to a feature matrix. Dimensionality reduction is done by applying diffusion maps. The method is adaptive and thus does not need training before analysis. We tested the method with data that includes normal and intrusive traffic to a web server. This approach finds all intrusions in the dataset.

Keywords: intrusion detection, anomaly detection, n-grams, diffusion map, data mining, machine learning

1 Introduction

The goal of this paper is to present an adaptive way to detect security attacks from network log data. All networks and systems can be vulnerable to different types of intrusions. Such attacks can exploit e.g. legitimate features, misconfigurations, programming mistakes or buffer overflows [15]. This is why *intrusion detection systems* are needed. An intrusion detection system gathers data from the network, stores this data to logfiles and analyzes it to find malicious or anomalous traffic [19]. Systems can be vulnerable to previously unknown attacks. Because usually these attacks differ from the normal network traffic, they can be found using anomaly detection [2].

In modern networks clients request and send information using queries. In HTTP traffic these queries are strings containing arguments and values. It is easy to manipulate such queries to include malicious attacks. These injection attacks try to create requests that corrupt the server or collect confidential information [18]. Therefore, it is important to analyze data collected from logfiles.

An *anomaly* is a pattern in data that is different from the well defined normal data [2]. In network data, this usually means an intrusion. There are two main approaches for detecting intrusions from network data: *misuse detection* and *anomaly detection* [19]. Misuse detection means using predefined attack signatures to detect the attacks, which is usually accurate but detecting new types of

* Now with C2 SmartLight Oy.

attacks is not possible. In anomaly detection the goal is to find actions that somehow deviate from normal traffic. This way it is possible to detect previously unknown attacks. However, not all anomalous traffic is intrusive. This means there might be more false alarms. Different kinds of machine learning based methods, such as self-organizing maps and support vector machines, have been used in anomaly detection [20, 23]. Information about other anomaly detection methods can be found in the literature [19]. *Unsupervised anomaly detection* techniques are most usable in this case, because no normal training data is required [2].

This study takes the approach of dimensionality reduction. Diffusion map is a manifold learning method that maps high-dimensional data to a low-dimensional diffusion space [5]. It provides tools for visualization and clustering [6]. The basic idea behind any manifold learning method is the eigen-decomposition of a similarity matrix. By unfolding the manifold it reveals the underlying structure of the data that is originally embedded in the high-dimensional space [1]. Diffusion maps have been applied to various data mining problems. These include vehicle classification by sound [21], music tonality [10], sensor fusion [12], radio network problem detection [25] and detection of injection attacks [8]. Advantages of this approach are that the dimensionality of the data is reduced and that it can be used unsupervised [2].

2 Method

2.1 Feature extraction

First let us define an n-gram as a consecutive sequence of n characters [7]. For example, the string *ababc* contains unique 2-grams *ab*, *ba* and *bc*. The 2-gram *ab* appears twice, thus having frequency of 2. A list of tokens of text can be represented with a vector consisting of n-gram frequencies [7]. Feature vector describing this string would be $x_{ababc} = [2,\ 1,\ 1]$. The only features extracted are n-gram frequencies. Furthermore, syntactic features of the input strings might reveal the differences between normal and anomalous behavior. Computed n-grams can extract features that describe these differences.

The frequencies are collected to a feature matrix X whose rows correspond to lines in logfiles and columns to features. These n-gram frequencies are key-value fields, variable-length by definition. Key strings are ignored and 2-grams are produced from each parameter value. The count of occurrences of every occurring 2-gram is summed. In practice n-gram tables produced from real-life data are very sparse, containing columns in which there are only zero occurrences. To minimize the number of columns, the processing is done in two passes. If a column contains no variation between entries, that column is not present in the final numeric matrix X. That makes it reasonable to use diffusion maps to process n-gram tables directly with no further preprocessing.

2.2 Dimensionality reduction

The number of extracted features is so large that dimensionality reduction is performed using diffusion maps. It is a manifold learning method that embeds the

original high-dimensional space into a low-dimensional diffusion space. Anomaly detection and clustering are easier in this embedded space [6].

The recorded data describe the behavior of the system. Let this data be $X = \{x_1, x_2, \ldots, x_N\}, x_i \in \mathbb{R}^n$. Here N is the number of samples and n the dimension of the original data. In practice the data is a $N \times n$ matrix with features as columns and each sample as rows.

At first, an affinity matrix W is constructed. This calculation takes most of the computation time. The matrix describes the distances between the points. This study uses the common Gaussian kernel with Euclidean distance measure, as in equation 1 [6, 16].

$$W_{ij} = \exp\left(-\frac{||x_i - x_j||^2}{\epsilon}\right) \tag{1}$$

The affinity neighborhood is defined by ϵ. Choosing the parameter ϵ is not trivial. It should be large enough to cover the local neighborhood but small so that it does not cover too much of it [21].

The rows of the affinity matrix are normalized using the diagonal matrix D, which contains the row sums of the matrix W on its diagonal.

$$D_{ii} = \sum_{j=1}^{N} W_{ij} \tag{2}$$

P expresses normalization that represents the probability of transforming from one state to another. Now the sum of each row is 1.

$$P = D^{-1}W \tag{3}$$

Next we need to obtain the eigenvalues of this transition probability matrix. The eigenvalues of P are the same with the conjugate matrix in equation 4. The eigenvectors of P can be derived from $\tilde{P}$ as shown later.

$$\tilde{P} = D^{\frac{1}{2}} P D^{-\frac{1}{2}} \tag{4}$$

If we substitute the P in equation 4 with the one in equation 3, we get the symmetric probability matrix $\tilde{P}$ in equation 5. It is called the normalized graph Laplacian [4] and it preserves the eigenvalues [16].

$$\tilde{P} = D^{-\frac{1}{2}} W D^{-\frac{1}{2}} \tag{5}$$

This symmetric matrix is then decomposed with singular value decomposition (SVD). Because $\tilde{P}$ is a normal matrix, spectral theorem states that such a matrix is decomposed with SVD: $\tilde{P} = U \Lambda U^*$. The eigenvalues on the diagonal of $\Lambda = \mathrm{diag}([\lambda_1, \lambda_2, \ldots, \lambda_N])$ correspond to the eigenvalues of the same matrix $\tilde{P}$ because it is symmetric. Matrix $U = [u_1, u_2, \ldots, u_N]$ contains in its columns the N eigenvectors u_k of $\tilde{P}$. Furthermore, because $\tilde{P}$ is conjugate with P, these two matrices share their eigenvalues. However, to calculate the right eigenvectors v_k of P, we use equation 6 and get them in the columns of $V = [v_1, v_2, \ldots, v_N]$ [16].

$$V = D^{-\frac{1}{2}}U \tag{6}$$

The coordinates of a data point in the embedded space using eigenvalues in Λ and eigenvectors in V are in the matrix Ψ in equation 7. The rows correspond to the samples and the columns to the new embedded coordinates [6].

$$\Psi = V\Lambda \tag{7}$$

Strictly speaking, the eigenvalues should be raised to the power of t. This scale parameter t tells how many time steps are being considered when moving from data point to another. Here we have set it $t = 1$ [6].

With suitable ϵ the decay of the spectrum is fast. Only d components are needed for the diffusion map for sufficient accuracy. It should be noted that the first eigenvector v_1 is constant and is left out. Using only the next d components the diffusion map for original data point x_i is presented in equation 8. Here $v_k(x_i)$ corresponds to the ith element of kth eigenvector [6].

$$\Psi_d : x_i \to [\lambda_2 v_2(x_i), \lambda_3 v_3(x_i), \ldots, \lambda_{d+1} v_{d+1}(x_i)] \tag{8}$$

This diffusion map embeds the known point x_i to a d-dimensional space. Dimension of the data is reduced from n to d. If desired, the diffusion map may be scaled by dividing the coordinates with λ_1.

2.3 Anomaly detection

After obtaining the low-dimensional presentation of the data it is easier to cluster the samples. Because spectral methods reveal the manifold, this clustering is called spectral clustering. This method reveals the normal and anomalous samples [13]. Alternatively, k-means or any other clustering method in the low-dimensional space is also possible [17]. Another approach is the density-based method [25].

Only the first few low-dimensional coordinates are interesting. They contain most of the information about the manifold structure. We use only the dimension corresponding to second eigenvector to determine the anomality of the samples. At 0, this dimension is divided into two clusters. The cluster with more samples is considered normal behavior. Conversely, the points in the other cluster are considered anomalous [22, 11, 13]. The second eigenvector acts as the separating feature for the two clusters in the low-dimensional space. The second eigenvalue is the solution to the normalized cut problem, which finds small weights between clusters but strong internal ties. This spectral clustering has probabilistic interpretation: grouping happens through similarity of transition probabilities between clusters [22, 14].

3 Results

3.1 Data acquisition

The data is acquired from a large real-life web service. The logfiles contain mostly normal traffic, but they also include anomalities and actual intrusions. The logfiles are from several Apache servers and are stored in *combined log format*. Listing below provides an example of a single logline. It includes information about the user's IP-address, time and timezone, the HTTP request including used resource and parameters, Apache server response code, amount of data sent to the user, the web page that was requested and used browser software.

```
127.0.0.1 - - [01/January/2011:00:00:01 +0300]
"GET /resource?parameter1=value1&parameter2=value2 HTTP/1.1"
200 2680 "http://www.address.com/webpage.html"
"Mozilla/5.0 (SymbianOS/9.2;...)"
```

The access log of a web site contains entries from multiple, distinct URLs. Most of them point to static requests like images, CSS files, etc. We are not focused to find anomalies at those requests because it is not possible to inject code via static requests unless there are major deficiencies in the HTTP server itself. Instead, we are focused in finding anomalies from dynamic requests because those requests are handled by the Web application, which is run behind the HTTP server.

To reach this goal, the access log entries are grouped by the resource URL. That is the part between host name and parameters in the HTTP URL scheme. Those resources containing only HTTP GET requests with no parameters are ignored. Each remaining resource is converted to a separate numerical matrix. In this matrix, a row represents a single access log entry, and a column represents an extracted feature.

Feature extraction is done in two passes. In the first pass the number of features is determined, and in the second pass the resulting matrix is produced. In our study we extracted the number of occurrences of 2-grams produced from HTTP GET parameters. These frequencies are normalized with logarithm in order to scale them. This ensures that the distances between the samples are comparable.

3.2 Data analysis

To measure the effectiveness of the method the data is labeled so that classification accuracy can be measured. However, this labeling is not used for training the diffusion map. The class labels are not input for the method.

Diffusion map reveals the structure of the data, and all the anomalies are detected. The n-gram features of the data are mapped to a lower dimensions. Figure 1 shows the resulting low-dimensional diffusion space with $\epsilon = 100$. The normal behavior lies in the dense area to the lower right corner. Anomalous points are to the left of 0.

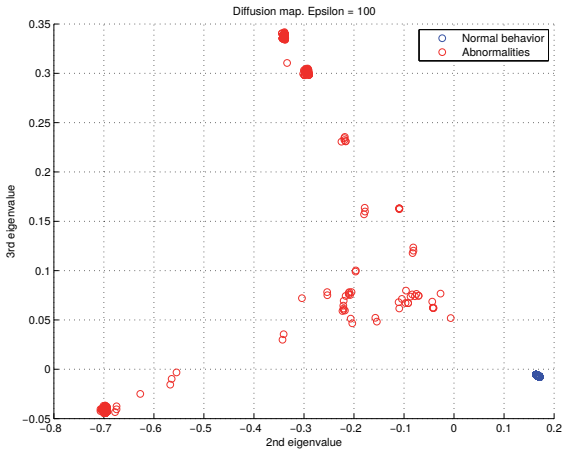

Fig. 1. Two-dimensional diffusion map of the dataset.

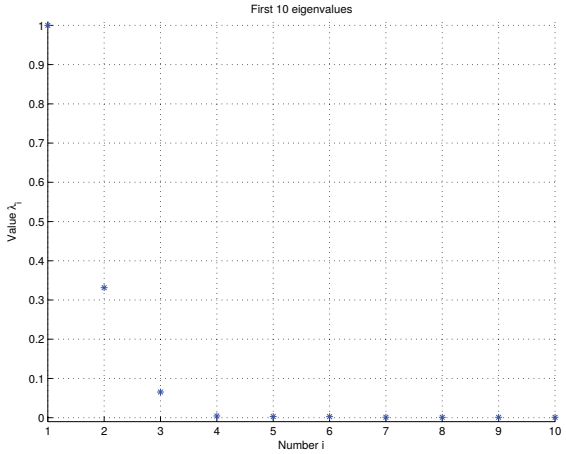

Fig. 2. Eigenvalues of transition matrix with $\epsilon = 100$.

Figure 2 shows that the eigenvalues converge rapidly with $\epsilon = 100$. This means that the first few eigenvalues and eigenvectors cover most of the differences observed in the data. The first value is 1 and corresponds to the constant eigenvector that is left out in the analysis. Eigenvalues $\lambda_2 = 0.331$ and $\lambda_3 = 0.065$ cover large portions of the data when compared to the rest that have values below 0.005.

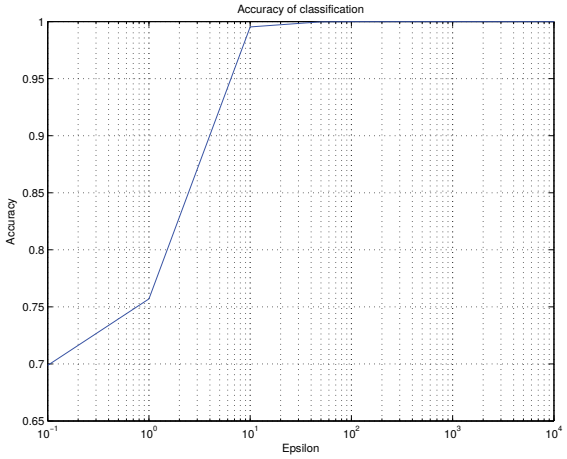

Fig. 3. Accuracy of classification changes when the parameter ϵ is changed.

Classification is tested with different values of ϵ, which defines the neighborhood for diffusion map. Accuracy of classification is defined as $accuracy = (tp + tn)/(tp + fp + fn + tn)$. Figure 3 shows how the accuracy of classification changes when ϵ is changed. Higher values of ϵ result in better accuracy. Precision of classification is defined $precision = tp/(tp + fp)$. The precision stays at 1 once any anomalies are detected, which means that all the anomalies detected are real anomalies regardless of the accuracy [9, p. 361].

For comparison, principal component analysis (PCA) is performed on the same normalized feature matrix [9, p. 79]. Results are very similar to the diffusion map approach, because of the simple structure of the feature matrix. Furthermore, PCA reaches the same accuracy and precision as diffusion map. The low-dimensional presentation is also very similar. Figure 4 shows the first two coordinates of PCA.

We also apply support vector machines (SVM) to the same data [9, p. 337–344]. LIBSVM implementation is used [3]. We use one-class SVM with RBF kernel function. A subset of the data is used in the model selection for SVM (500 lines randomly selected). Then the rest of the data is used to test the method.

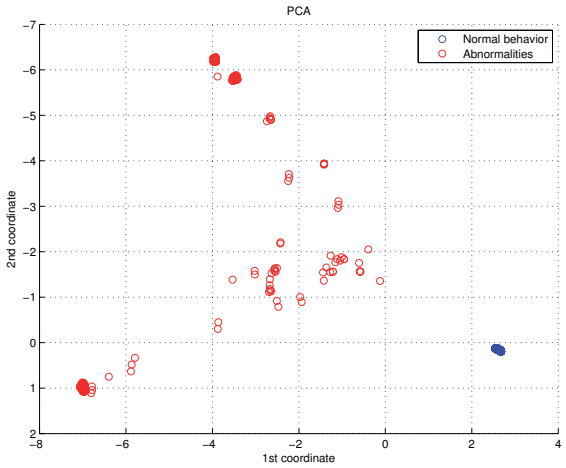

Fig. 4. PCA of the dataset, first two coordinates. The Y-axis of this figure has been reversed for better visual comparison with diffusion map.

The data labels are unknown, so the training data is not "clean" and contains some intrusions as well. It is possible to find the right parameters (ν and γ) for model selection if pre-specified true positive rate is known. The parameters which give a similar cross-validation accuracy can be selected [3]. However, this kind of information is not available. Fully automatic parameter selection for OC-SVM could be achieved by using more complicated methods, such as evolving training model method [24]. In this study the parameter selection is done manually. At best the accuracy is 0.999 and precision 0.998.

4 Conclusion

The goal of this study is to find security attacks from network data. This goal is met since all the known attacks are found. The proposed anomaly detection scheme could be used for query log analysis in real situations. In practice the boundary between normal and anomalous might not be as clear as in this example. However, the relative strangeness of the sample could indicate how severe an alert is.

The diffusion map framework adapts to the log data. It assumes that the data lies on a manifold, and finds a coordinate system that describes the global structure of the data. These coordinates could be used for further analysis of characteristics of anomalous activities.

Because all the methods perform extremely well, the data in question is rather sparse and the discriminating features are quite evident from the feature matrix.

This is the merit of n-gram feature extraction which creates a feature space that separates the normal behavior in a good manner. The features describe the data clearly, and they are easy to process afterwards.

One advantage of the diffusion map methodology is that it has only one meta-parameter, ϵ. It can be estimated with simple interval search. If for some reason the threshold sensitivity needs to be changed, ϵ gives the flexibility to adapt to the global structure. For comparison, the SVM we used has two parameters, ν and γ. Searching the best parameters for the application gets more difficult as the number of parameters increases.

The presented anomaly detection method performs well on real data. As an unsupervised algorithm this approach is well suited to finding previously unknown intrusions. This method could be applied to offline clustering as well as extended to a real-time intrusion detection system.

Acknowledgements

The authors thank Professors Amir Averbuch, Timo Hämäläinen and Tapani Ristaniemi for their continued support. Thanks are extended to Juho Knuuttila and Kristian Siljander for useful discussions and ideas.

References

1. Bengio, Y., Delalleau, O., Roux, N.L., Paiement, J.F., Vincent, P., Ouimet, M.: Feature Extraction, chap. Spectral Dimensionality Reduction, pp. 519–550. Studies in Fuzziness and Soft Computing, Springer Berlin, Heidelberg (2006)
2. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM Comput. Surv. 41(3), 1–58 (2009)
3. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001), software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
4. Chung, F.R.K.: Spectral Graph Theory, p. 2. AMS Press, Providence, R.I (1997)
5. Coifman, R.R., Lafon, S., Lee, A.B., Maggioni, M., Nadler, B., Warner, F., Zucker, S.W.: Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. In: Proceedings of the National Academy of Sciences of the United States of America. vol. 102, p. 7426 (2005)
6. Coifman, R.R., Lafon, S.: Diffusion maps. Applied and Computational Harmonic Analysis 21(1), 5–30 (2006)
7. Damashek, M.: Gauging similarity with n-grams: Language-independent categorization of text. Science 267(5199), 843 (1995)
8. David, G.: Anomaly Detection and Classification via Diffusion Processes in Hyper-Networks. Ph.D. thesis, Tel-Aviv University (2009)
9. Han, J., Kamber, M.: Data mining: concepts and techniques. Morgan Kaufmann (2006)
10. İzmirli, Ö.: Tonal-atonal classification of music audio using diffusion maps. In: 10th International Society for Music Information Retrieval Conference (ISMIR 2009) (2009)
11. Kannan, R., Vempala, S., Vetta, A.: On clusterings: Good, bad and spectral. J. ACM 51, 497–515 (May 2004)

12. Keller, Y., Coifman, R., Lafon, S., Zucker, S.: Audio-visual group recognition using diffusion maps. Signal Processing, IEEE Transactions on 58(1), 403–413 (2010)
13. von Luxburg, U.: A tutorial on spectral clustering. Statistics and Computing 17, 395–416 (2007)
14. Meila, M., Shi, J.: Learning segmentation by random walks. In: NIPS. pp. 873–879 (2000)
15. Mukkamala, S., Sung, A.: A comparative study of techniques for intrusion detection (2003)
16. Nadler, B., Lafon, S., Coifman, R., Kevrekidis, I.G.: Diffusion maps – a probabilistic interpretation for spectral embedding and clustering algorithms. In: Barth, T.J., Griebel, M., Keyes, D.E., Nieminen, R.M., Roose, D., Schlick, T., Gorban, A.N., Kégl, B., Wunsch, D.C., Zinovyev, A.Y. (eds.) Principal Manifolds for Data Visualization and Dimension Reduction, Lecture Notes in Computational Science and Engineering, vol. 58, pp. 238–260. Springer Berlin Heidelberg (2008)
17. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: Advances in Neural Information Processing Systems 14. pp. 849–856. MIT Press (2001)
18. Nguyen-Tuong, A., Guarnieri, S., Greene, D., Shirley, J., Evans, D.: Automatically hardening web applications using precise tainting. In: Sasaki, R., Qing, S., Okamoto, E., Yoshiura, H. (eds.) Security and Privacy in the Age of Ubiquitous Computing, IFIP Advances in Information and Communication Technology, vol. 181, pp. 295–307. Springer Boston (2005)
19. Patcha, A., Park, J.: An overview of anomaly detection techniques: Existing solutions and latest technological trends. Computer Networks 51(12), 3448–3470 (2007)
20. Ramadas, M., Ostermann, S., Tjaden, B.: Detecting anomalous network traffic with self-organizing maps. In: Vigna, G., Jonsson, E., Kruegel, C. (eds.) Recent Advances in Intrusion Detection. pp. 36–54. Springer (2003)
21. Schclar, A., Averbuch, A., Rabin, N., Zheludev, V., Hochman, K.: A diffusion framework for detection of moving vehicles. Digital Signal Processing 20(1), 111–122 (2010)
22. Shi, J., Malik, J.: Normalized cuts and image segmentation. Pattern Analysis and Machine Intelligence, IEEE Transactions on 22(8), 888 –905 (2000)
23. Tran, Q., Duan, H., Li, X.: One-class support vector machine for anomaly network traffic detection. China Education and Research Network (CERNET), Tsinghua University, Main Building 310 (2004)
24. Tran, Q.A., Zhang, Q., Li, X.: Evolving training model method for one-class svm. In: Systems, Man and Cybernetics, 2003. IEEE International Conference on. vol. 3, pp. 2388–2393 (2003)
25. Turkka, J., Ristaniemi, T., David, G., Averbuch, A.: Anomaly detection framework for tracing problems in radio networks. In: Proc. to ICN 2011 (2011)

PII

DIMENSIONALITY REDUCTION FRAMEWORK FOR DETECTING ANOMALIES FROM NETWORK LOGS

by

Tuomo Sipola, Antti Juvonen and Joel Lehtonen 2012

Engineering Intelligent Systems, Vol. 20, Iss. 1–2, pp. 87–97

Dimensionality Reduction Framework for Detecting Anomalies from Network Logs

Tuomo Sipola Antti Juvonen Joel Lehtonen*
tuomo.sipola@jyu.fi antti.k.a.juvonen@jyu.fi joel.lehtonen@iki.fi

Department of Mathematical Information Technology
University of Jyväskylä
Jyväskylä, Finland

Abstract

Dynamic web services are vulnerable to a multitude of intrusions that could be previously unknown. Server logs contain vast amounts of information about network traffic, and finding attacks from these logs improves the security of the services. In this research features are extracted from HTTP query parameters using 2-grams. We propose a framework that uses dimensionality reduction and clustering to identify anomalous behavior. The framework detects intrusions from log data gathered from a real network service. This approach is adaptive, works on the application layer and reduces the number of log lines that needs to be inspected. Furthermore, the traffic can be visualized.

Keywords: intrusion detection, anomaly detection, n-grams, diffusion map, data mining, machine learning

1 Introduction

The goal of this paper is to present an adaptive way to detect security attacks from network log data. All networks and systems can be vulnerable to different types of intrusions. Such attacks can exploit e.g. legitimate features, misconfigurations, programming mistakes or buffer overflows [1]. This is why *intrusion detection systems* are needed. An intrusion detection system gathers data from the network, stores these data to log files and analyzes them to find malicious or anomalous traffic [2]. Systems can be vulnerable to previously unknown attacks, commonly known as zero-day attacks [3]. Because usually these attacks differ from the normal network traffic, they can be found using anomaly detection [4].

In modern networks clients request and send information using queries. In HTTP traffic these queries are strings containing arguments and values. It is easy to manipulate such queries to include malicious attacks. These injection attacks try to create requests that corrupt the server or collect confidential information [5]. Therefore, it is important to analyze the collected data in log files. Most intrusion detection systems

*Now with C2 SmartLight Ltd.

analyze TCP packet data. There are not many application layer IDS systems available. Because the HTTP log data are very different from network packet data, they both need to be analyzed. Different attacks can be performed on different layers.

An *anomaly* is a pattern in data that is different from the well defined normal data [4]. In network data, this usually means an intrusion. There are two main approaches for detecting intrusions from network data: *misuse detection* and *anomaly detection* [2]. Misuse detection means using predefined attack signatures to detect the attacks, which is usually accurate but detecting new types of attacks is not possible. In anomaly detection the goal is to find actions that somehow deviate from normal traffic. This way it is possible to detect previously unknown attacks. However, not all anomalous traffic is intrusive. This means there might be more false alarms. Different kinds of machine learning based methods, such as self-organizing maps and support vector machines, have been used in anomaly detection [6, 7]. Information about other anomaly detection methods can be found in the literature [2]. *Unsupervised anomaly detection* techniques are most usable in this case, because no normal training data are available [4]. These techniques work without prior knowledge of attack patterns. This kind of adaptive framework is suitable for *a posteriori* network log analysis.

This study takes the approach of dimensionality reduction. Because the number of dimensions of the feature space grows large and sparse when analyzing textual information, such as log files, this is one of the most feasible techniques. Furthermore, the sparsity of data suggests about the underlying low dimensional structure. Almost the same amount of information can be represented with lower number of dimensions. Diffusion map is a manifold learning method that maps high-dimensional data to a low-dimensional diffusion space [8]. It provides tools for visualization and clustering [9]. The basic idea behind any manifold learning method is the eigendecomposition of a similarity matrix. By unfolding the manifold it reveals the underlying structure of the data that is originally embedded in the high-dimensional space [10]. Diffusion maps have been applied to various data mining problems. These include vehicle classification by sound [11], music tonality [12], sensor fusion [13], radio network problem detection [14, 15] and detection of injection attacks [16]. In addition to the advantage of reduced number of dimensions, the approach can be used for unsupervised learning [4].

2 Related research

Kruegel and Vigna [17] analyzed the parameter values of HTTP queries. The static queries with no parameters were removed. The underlying assumption is that attack patterns differ from normal traffic and that this difference can be expressed quantitatively. They used several different analyzing methods, such as attribute length and character distribution. The learning was based on previous data. The data were not labeled. The analysis of character distribution is similar to our research, because essentially the characters are n-grams with the length 1. We use 2-grams for higher detection rates, but we will also get more dimensions in the data matrices.

Hubballi et al. [18] used layered higher order n-grams for detecting intrusions. However, this analysis was not done on application layer data, but on the network packet payloads. Higher order n-grams are n-grams where $n > 2$. This means that the method is computationally more expensive, but rare events might be detected more accurately. The n-grams are organized into bins based on their frequency. The analysis starts with 1-grams, and it moves to higher n-grams incrementally to get higher accuracy. In the research the number of distinct and unique n-grams went up almost

linearly as n increased. Therefore, using higher order n-grams might not be as complex in practice as it could be theoretically. For example, the theoretical maximum number for 3-grams in ascii-characters is 256^3, which is considerably higher than the case with 2-grams.

Dimensionality reduction has been discussed in the context of anomaly detection from networks. Ringberg et al. studied the IP packet data and tried to detect anomalies using principal component analysis. They also identified the main challenges when using principal component dimensionality reduction approach. The finding about large anomalies contaminating the subspace is relevant also to our research. However, their network architecture is more complex than ours [19]. Callegari et al. analyzed similar packet data [20]. These studies used low-level IP packet datasets that need specific aggregation before they can be processed. Our research concerns the application level log data, which is text, while the IP packet datasets are numeric. In addition, we compare the results of principal component analysis and diffusion maps.

Diffusion maps have been applied in the network security context. David explored the use of diffusion map methods to find injection attacks in hyper-networks. His data included SQL injection examples that used a similar feature extraction as our research. The n-gram feature extraction was applied to tokenized SQL [16]. Our research, in contrast, focuses on the raw textual queries. Furthermore, David and Averbuch used a localized diffusion folder approach to classify network protocols, among other examples. Their data contains low-level features such as duration and the number of bytes [21]. However, our data comes from the application layer of the network, specifically web server logs. These are different from the low-level network features and contain lots of textual information in the form of queries. Moreover, we use the theoretical framework of spectral clustering as the basis of our research.

3 Methodology

Straightforward numerical methods are difficult to apply to textual data such as log files. Therefore, log data must be transformed into feature space. This mapping of textual information to numerical matrix enables mathematical analysis of the original log lines.

However, this leads to a large number of dimensions in the feature space. For efficient analysis, classification and visualization the number of dimensions must be reduced. This gives the opportunity to use a multitude of classification algorithms.

The proposed method consists of the following steps:

1. Removing lines that do not contain parameters.
2. Feature extraction from the log line using 2-grams.
3. Dimensionality reduction of the features.
4. Classifying the lines either as normal or attack.

After these steps the log file can be visualized as a figure where the attacks are more easily seen than from a text file. Furthermore, the suspected attack lines can be inspected in more depth. This facilitates finding abnormal activities because only these suspected lines are inspected, instead of thousands in the original log.

3.1 Feature extraction

The features include 2-grams from HTTP query parameters. The log files are simple text files where each line represents one query sent from the client to the server. Extracting the true intention of the query is challenging, and the text needs to be converted to a more machine-friendly format. The feature extraction essentially means converting this textual data into numerical matrices.

First let us define an *n*-gram as a consecutive sequence of *n* characters [22]. *N*-gram is a substring with length of *n*. For example, the string *ababc* contains unique 2-grams *ab*, *ba* and *bc*. The 2-gram *ab* appears twice, thus having frequency of 2. A list of tokens of text can be represented with a vector consisting of *n*-gram frequencies [22]. Feature vector describing this string would be $x_{ababc} = [2,\ 1,\ 1]$. The only features extracted are *n*-gram frequencies. Furthermore, syntactic features of the input strings might reveal the differences between normal and anomalous behavior. Computed *n*-grams can extract features that describe these differences. It is assumed that an anomalous query contains some text in the parameter part that differs from normal behaviour. This means that it must contain some *n*-grams that appear rarely in the data.

Here is an example of constructing the feature matrix using the *n*-gram analysis process with two words, *anomaly* and *analysis*. From these words we get the unique 2-grams `an`, `no`, `om`, `ma`, `al`, `ly`, `na`, `ys`, `si` and `is`. From this information we can construct a matrix with the *n*-gram frequencies.

an	no	om	ma	al	ly	na	ys	si	is
1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1

The feature matrix is constructed in this way, including all the different strings that appear in the parameter fields on each query. Each log line corresponds to a row in the matrix. From this we can see that there are 10 unique 2-grams in this example. The logs are ascii-coded, so they can contain 256 different characters. The theoretical maximum number for unique 2-grams using ascii-characters is 256^2, but in practice we did not get even near to that number. However, in a very varied and big dataset the number of dimensions could get very high.

The frequencies are collected to a feature matrix X. These *n*-gram frequencies are key-value fields, variable-length by definition. Key strings are ignored and 2-grams are produced from each parameter value. The count of occurrences of every occurring 2-gram is summed. In practice *n*-gram tables produced from real life data are very sparse, containing columns in which there are only zero occurrences. To minimize the number of columns, the processing is done in two passes, first determining the number of unique *n*-grams and then analyzing the frequencies. If a column contains no variation between entries, that column is not present in the final numeric matrix X. Therefore, only the columns that actually contain some useful information about the features are included in the analysis.

With this preprocessing technique it is possible to use *n*-grams whose value of *n* is higher than 2. However, the number of unique *n*-grams will increase and therefore the number of dimensions will increase as well.

3.2 Dimensionality reduction

The number of extracted features is so large that dimensionality reduction is performed using principal component analysis and diffusion map. Diffusion map is a manifold

learning method that embeds the original high-dimensional space into a low-dimensional diffusion space. Anomaly detection and clustering are easier in this embedded space [9].

The recorded data describe the behavior of the system. Let this data be $X = \{x_1, \ldots, x_N\}$, $x_i \in \mathbb{R}^n$. Here N is the number of samples and n the dimension of the original data. In practice the data are in a $N \times n$ matrix with features as columns and each sample as rows.

3.2.1 Principal component analysis

Principal component analysis (PCA) tries to extract orthogonal components maximizing their variance from the data. This simplifies the representation of the information within the data and also facilitates the analysis of the structure and features in the data. The principal components are linear combinations of the original features. The first principal component contains the largest amount of variance. PCA reveals the most information in terms of variance, but this does not necessarily mean that it separates different clusters in an optimal way [23, 24, 25].

PCA performs the eigendecomposition on the covariance matrix C of the centered data matrix X_c. The decomposition $C = U\Lambda U^*$ gives the eigenvectors in U that map the points in X to a low-dimensional space. This mapping can be calculated with $X_{PCA} = XU$. Another approach is to take the singular value decomposition (SVD) of the original matrix X. One way to interpret this is as rotation of axes to find the most important features. The new principal components are in the direction of most variance in the data and thus represent the most differentiating combination of features [23, 24, 25].

As with many dimensionality reduction methods using eigendecompositions, the number of selected components becomes a problem. One way to do this is to seek for the eigengap, i.e. a big change of eigenvalues. This way the eigenvalues reveal the principal components that cover most of the variance [23, 24, 25].

PCA is a linear method and has difficulties finding nonlinear dependencies between features. It has initial assumptions that restrict its use for latent variable separation and nonlinear dimensionality reduction [25].

3.2.2 Diffusion map

At first, an affinity matrix W is constructed. This calculation takes most of the computation time. The matrix describes the distances between the points. This study uses the common Gaussian kernel with Euclidean distance measure, as in equation 1 [9, 26].

$$W_{ij} = \exp\left(-\frac{||x_i - x_j||^2}{\varepsilon}\right) \tag{1}$$

The affinity neighborhood is defined by ε. Choosing the parameter ε is not trivial. It should be large enough to cover the local neighborhood but small so that it does not cover too much of it [11].

The rows of the affinity matrix are normalized using the diagonal matrix D, which contains the row sums of the matrix W on its diagonal.

$$D_{ii} = \sum_{j=1}^{N} W_{ij} \tag{2}$$

P expresses normalization that represents the probability of transforming from one state to another. Now the sum of each row is 1.

$$P = D^{-1}W \tag{3}$$

Next we need to obtain the eigenvalues of this transition probability matrix. The eigenvalues of P are the same with the conjugate matrix in equation 4. The eigenvectors of P can be derived from $\tilde{P}$ as shown later.

$$\tilde{P} = D^{\frac{1}{2}}PD^{-\frac{1}{2}} \tag{4}$$

If we substitute the P in equation 4 with the one in equation 3, we get the symmetric probability matrix $\tilde{P}$ in equation 5. It is called the normalized graph Laplacian [27] and it preserves the eigenvalues [26].

$$\tilde{P} = D^{-\frac{1}{2}}WD^{-\frac{1}{2}} \tag{5}$$

This symmetric matrix is then decomposed with singular value decomposition (SVD). Because $\tilde{P}$ is a normal matrix, spectral theorem states that such a matrix is decomposed with SVD: $\tilde{P} = U\Lambda U^*$. The singular values of this symmetric square matrix equal to its eigenvalues, which lie on the diagonal of $\Lambda = \mathrm{diag}([\lambda_1, \lambda_2, \ldots, \lambda_N])$. Matrix $U = [u_1, u_2, \ldots, u_N]$ contains in its columns the N eigenvectors u_k of $\tilde{P}$. Furthermore, because $\tilde{P}$ is conjugate with P, these two matrices share their eigenvalues. However, to calculate the right eigenvectors v_k of P, we use equation 6 and get them in the columns of $V = [v_1, v_2, \ldots, v_N]$ [26].

$$V = D^{-\frac{1}{2}}U \tag{6}$$

The coordinates of a data point in the embedded space using eigenvalues in Λ and eigenvectors in V are in the matrix Ψ in equation 7. The rows correspond to the samples and the columns to the new embedded coordinates [9].

$$\Psi = V\Lambda \tag{7}$$

Strictly speaking, the eigenvalues should be raised to the power of t. This scale parameter t tells how many time steps are being considered when moving from data point to another. Here we have set it $t = 1$ [9].

With suitable ε the decay of the spectrum is fast. Only d components are needed for the diffusion map for sufficient accuracy. It should be noted that the first eigenvector v_1 is constant and is left out. Using only the next d components the diffusion map for original data point x_i is presented in equation 8. Here $v_k(x_i)$ corresponds to the ith element of kth eigenvector [9].

$$\Psi_d : x_i \to [\lambda_2 v_2(x_i), \lambda_3 v_3(x_i), \ldots, \lambda_{d+1} v_{d+1}(x_i)] \tag{8}$$

This diffusion map embeds the known point x_i to a d-dimensional space. Dimension of the data are reduced from n to d. If desired, the diffusion map may be scaled by dividing the coordinates with λ_1.

3.3 Anomaly detection

After obtaining the low-dimensional presentation of the data it is easier to cluster the samples. Because spectral methods reveal the manifold, this clustering is called spectral clustering. This method reveals the normal and anomalous samples [28]. Alternatively, k-means or any other clustering method in the low-dimensional space is also possible [29]. Another approach is the density-based method [14].

Only the first few low-dimensional coordinates are interesting. They contain most of the information about the manifold structure. For diffusion map we use only the dimension corresponding to second eigenvector to determine the anomality of the samples. At 0, this dimension is divided into two clusters. The cluster with more samples is considered normal behavior. Conversely, the points in the other cluster are considered anomalous [30, 31, 28]. The second eigenvector acts as the separating feature for the two clusters in the low-dimensional space. The second eigenvalue is the solution to the normalized cut problem, which finds small weights between clusters but strong internal ties. This spectral clustering has probabilistic interpretation: grouping happens through similarity of transition probabilities between clusters [30, 32]. For PCA we use the first principal component in a similar way.

In practice the border between the normal and anomalous behavior might be unclear. This is the case especially with unsupervised learning, or when exploring the data for the first time. The normal cluster is usually very dense, and most of the data points lie within that cluster. The other points can be interpreted as deviating from the normal state, and thus anomalous.

4 Case 1: Validation with labeled data

4.1 Data acquisition

The data are acquired from a large real life web service. Let us call this dataset "A". This case has been presented in an earlier publication [33]. The log files contain mostly normal traffic, but they also include anomalities and actual intrusions. The log files are from several Apache servers and are stored in *combined log format*. Listing below provides an example of a single log line. It includes information about the user's IP address, time and timezone, the HTTP request including used resource and parameters, Apache server response code, amount of data sent to the user, the web page that was requested and used browser software.

```
127.0.0.1 - - [01/January/2011:00:00:01 +0300]
"GET /resource?parameter1=value1&parameter2=value2 HTTP/1.1"
200 2680 "http://www.address.com/webpage.html"
"Mozilla/5.0 (SymbianOS/9.2;...)"
```

The access log of a web site contains entries from multiple, distinct URLs. Most of them point to static requests like images, CSS files, etc. We do not focus on finding anomalies from those requests because it is not possible to inject code via static requests unless there are major deficiencies in the HTTP server itself. Instead, we focus on finding anomalies from dynamic requests because those requests are handled by the web application, which is run behind the HTTP server.

To reach this goal, the access log entries are grouped by the resource URL. That is the part between host name and parameters in the HTTP URL scheme. Resources

containing only HTTP GET requests with no parameters are ignored. Each remaining resource is converted to a separate numerical matrix. In this matrix, a row represents a single access log entry, and a column represents an extracted feature.

Feature extraction is done in two passes. In the first pass the number of features is determined, and in the second pass the resulting matrix is produced. In our study we extracted the number of occurrences of 2-grams produced from HTTP GET parameters. In the example above, the parameter values form a string `value1value2`. This string is then analyzed for 2-gram frequencies. These frequencies are normalized with logarithm in order to scale them. This ensures that the distances between the samples are comparable.

4.2 Data analysis

To measure the effectiveness of the method the data are labeled so that classification accuracy can be measured. However, this labeling is not used for training the diffusion map. The class labels are not input for the method.

Diffusion map reveals the structure of the data, and all the anomalies are detected. The n-gram features of the data are mapped to lower dimensions. Figure 1 shows the resulting low-dimensional diffusion space with $\varepsilon = 100$. The normal behavior (N=2999) lies in the dense area to the upper left corner. Anomalous points (N=1293) are to the right of 0.

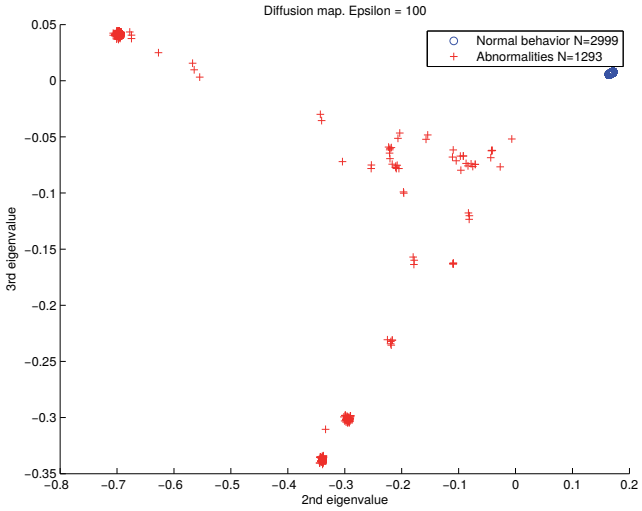

Figure 1: Two-dimensional diffusion map of the dataset A.

Figure 2 shows that the eigenvalues converge rapidly with $\varepsilon = 100$. This means that the first few eigenvalues and eigenvectors cover most of the differences observed in the data. The first value is 1 and corresponds to the constant eigenvector that is left out in the analysis. Eigenvalues $\lambda_2 = 0.331$ and $\lambda_3 = 0.065$ cover large portions of the data when compared to the rest that have values below 0.005.

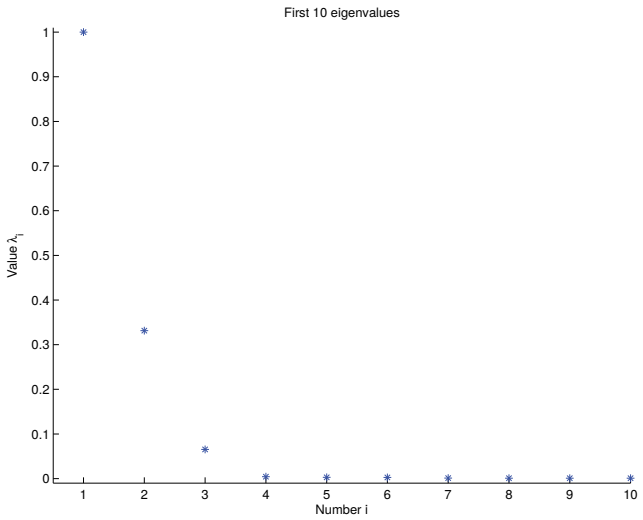

Figure 2: Eigenvalues of transition matrix with $\varepsilon = 100$ (dataset A).

Classification is tested with different values of ε, which defines the neighborhood for diffusion map. Accuracy of classification is defined as $accuracy = (tp+tn)/(tp+fp+fn+tn)$. Figure 3 shows how the accuracy of classification changes when ε is changed. Higher values of ε result in better accuracy. Precision of classification is defined $precision = tp/(tp+fp)$. The precision stays at 1 once any anomalies are detected, which means that all the anomalies detected are real anomalies regardless of the accuracy [23, p. 361].

For comparison, principal component analysis (PCA) is performed on the same normalized feature matrix [23, p. 79]. Results are very similar to the diffusion map approach, because of the simple structure of the feature matrix. This suggests that data points are linearly dependent. Furthermore, PCA reaches the same accuracy and precision as diffusion map. The low-dimensional presentation is also very similar. Figure 4 shows the first two coordinates of PCA.

5 Case 2: Unknown data

5.1 Data acquisition

After testing the methods with known data, we now analyze data that is totally unknown. We call this dataset "B". This is the realistic situation with the web service that we are trying to analyze. There is no previous information about any attacks or other anomalies. The goal is to find a small amount of interesting lines that can then be analyzed more accurately. The number of log lines is so big that it is impossible to check all the lines manually. This is why anomaly detection is needed.

We start with relatively new dataset that has about 10 million lines. However, the lines with no parameters in the HTTP queries can be filtered out, because they are

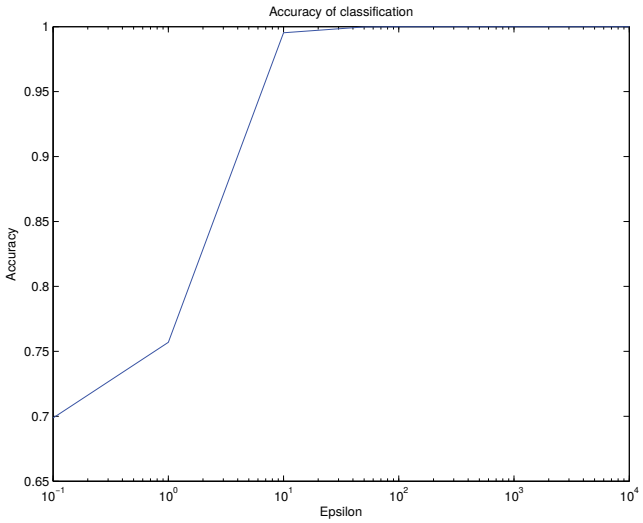

Figure 3: Accuracy of classification changes when the parameter ε is changed (dataset A).

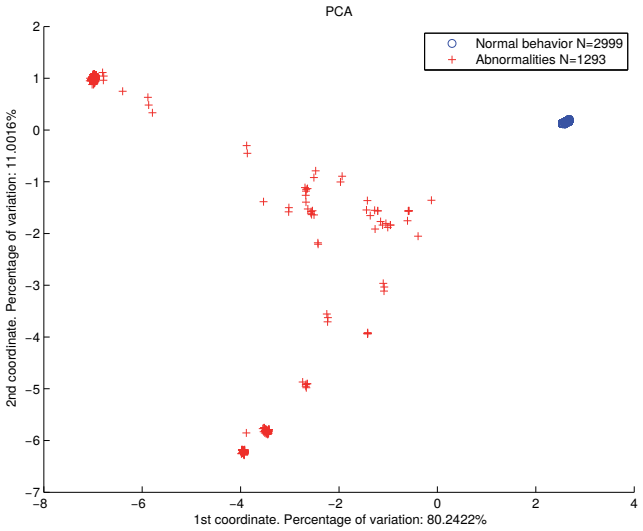

Figure 4: PCA of dataset A, first two coordinates.

not a big security risk. This leaves us with 2.5 million lines. These lines are then divided into different files based on the HTTP request URI. The entire log file cannot be analyzed at once, because different resources have very different parameters. The normal parameters for one single URI are usually quite similar, however. It makes more sense trying to analyze them individually. The number of different resources is quite high, more than 80 000, but many of the resources include just a few lines. In this case, it is sufficient to analyze some of the most frequently used resources. In addition to finding anomalies, this will give us more information about the web service traffic in general. The traffic can also be visualized.

After preprocessing the number of unique 2-grams in the most used resource URI was more than 1100. This means that dimensionality reduction is definitely needed, but the number of dimensions is not close to the theoretical maximum of 256^2 2-grams.

The new log file acquired for this case is in a different format than the log file analyzed in the first case. Some additional information, such as time in UNIX time, is also included. However, this information is not used in this analysis. The features extracted are the same as in the previous case. The feature matrix is calculated only from the parameter values. In this example the string to be analyzed would be `value`.

```
1305167880 111.222.111.222 965 633 29112
GET /path/to/resource.png?parameter=value HTTP/1.1
/path/to/resource.png parameter=value
/full/path/to/resource.png 200 + -
image/png,image/*;q=0.8,*/*;q=0.5 GB2312,utf-8;q=0.7,*;q=0.7
gzip,deflate fi-fi,fi;q=0.5 http://example.com Mozilla/5.0
(Windows; U; Windows NT 5.1; fi-FI; rv:1.9.2.15)
Gecko/20110101 Firefox/3.6.15
```

Feature matrix is constructed and logarithmic scaling applied in the same way as presented earlier with the dataset A. Even though the number of lines in the log file is quite high, the preprocessing phase takes only less than 10 minutes. Everything is written into temporary files to save memory. If more memory is available, the preprocessing could be changed to use it and it would get faster.

5.2 Data analysis

We choose two commonly used resources for analysis from the whole log data B. These resources are called "B1" and "B2". We aim to find possible intrusion attempts from them. Dimensionality reduction is performed with both PCA and diffusion map. The results are then compared. Choosing ε for diffusion map is done differently than for dataset A. The sum $L = \sum_{i,j} W_{ij}$ plotted using logarithmic scale reveals the desirable linear region for ε [34, 35]. The value is chosen from that range, however, because even small changes of ε in that area affect the resulting embedding drastically, some human discretion must be used. Classification is done using spectral clustering.

Dataset B1 turns out to be a simple case where most of the data points are similar. The few deviations are easy to find from the feature matrix. First B1 is analyzed using PCA. Figure 5 shows that most of the normal behavior (N=14206) is concentrated to a very dense cluster. Our classifier assumes the points (N=87) to the right of the normal cluster to be anomalous. This clustering is feasible because the log lines contain actual previously unknown intrusions, although not all anomalies are intrusive. The anomalous points also seem to form clusters. These could indicate different types of

attacks that happen frequently. This information could be used to further protect the service in the future.

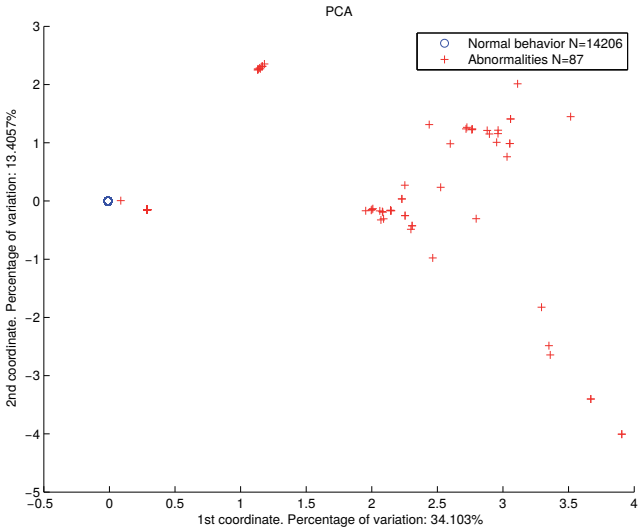

Figure 5: PCA of dataset B1.

Dataset B1 is also analyzed with diffusion map. The classification results are similar to PCA, even though the Figure 6 looks different. The normal behavior (N=14216) is concentrated leaving the anomalies (N=77) to the right of zero based on the first coordinate. The most differing anomalies are very far from the normal cluster. Some anomalies are very close to the normal data points. This means that the border between anomalous and normal traffic is not very clear. For this reason, 10 intrusion attempts that PCA detected were not discovered by diffusion map. This explains the difference in the number of found anomalies. Finding an optimal ε value would improve the result. However, this is a difficult task because of the unsupervised approach. Even though the low dimensional picture of PCA does not look as clear as the diffusion map, the result for PCA is better due to 10 false negatives that diffusion map fails to detect.

Dataset B2 contains more difficult and complex queries. This set is an example where the low dimensional representation by PCA and diffusion map are clearly different. Figure 7 shows the PCA of this dataset. The structure of the dataset is seen from the figure but the exact location of anomalies is difficult to find. This is because even the normal query lines include long and dynamically changing strings. The sparse left side is actually normal traffic, but there seems to be a lot of variation in the normal traffic alone. The anomalies found by diffusion map are situated in the upper right corner of the PCA representation. Data points do not form a distinct cluster, making anomaly detection and clustering very difficult with this representation. The used simple spectral clustering clearly does not work in this case. Further clustering with more advanced algorithms might reveal what types of queries the log file contains. Most variance is captured by the first principal component. However, two first principal components do not contain most of the total cumulative variance. Even this kind of visualization

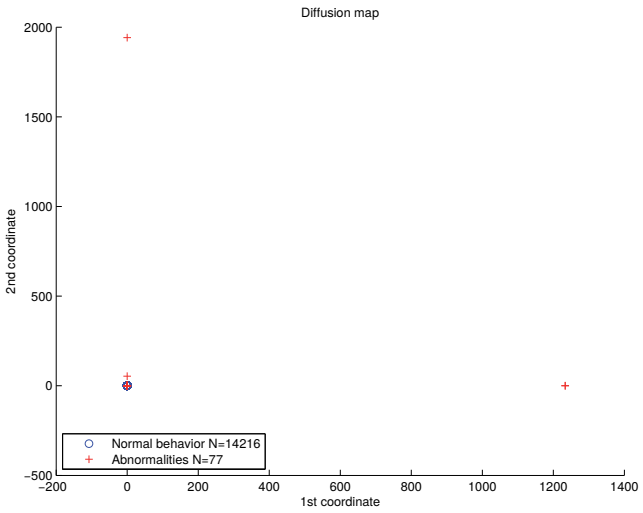

Figure 6: Diffusion map of dataset B1, $\varepsilon = 7$.

facilitates the analysis of huge text files (N=21406).

Diffusion map finds anomalies from dataset B2. The first two coordinates capture almost all of the difference between normal and anomalous queries (Figure 8). In addition, the clusters are very clearly separated and the normal traffic is easy to distinguish. This dataset shows a clear difference between PCA and diffusion map results. The anomalous cluster (N=173) contains the points on the far left and the anomalous points near the normal cluster. Again, the normal cluster (N=21233) is very dense. The found anomalies contain 88 real intrusions. The intrusions are related to injecting malicious SQL queries or scripts into the HTTP query. Some non-intrusive queries are also included, but they can be manually screened afterwards. The number of log lines is small enough so that system administrator can inspect the anomalous lines and easily find the intrusion attempts. Anomaly detection seems to find attacks from a large and varying dataset. The anomalous traffic forms two distinct clusters, one of which contains the intrusions. Diffusion map with a correctly selected ε helps in finding anomalies and automatically detecting normal cluster. Larger values of ε make the diffusion map behave more like PCA. These approachees are more suitable for visual inspection and multicluster analysis.

6 Conclusion

The goal of this study is to find security attacks from network data. The proposed anomaly detection scheme could be used for query log analysis in real life situations. We concentrate on web server log data, which contains text queries that are the focus of our analysis. In these kinds of practical situations the boundary between normal traffic and intrusions is not always very clear. However, the relative strangeness of the sample could indicate how severe an alert is.

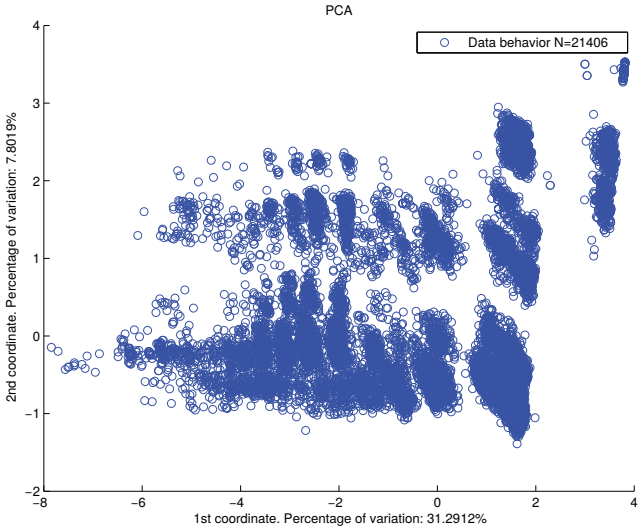

Figure 7: PCA of dataset B2.

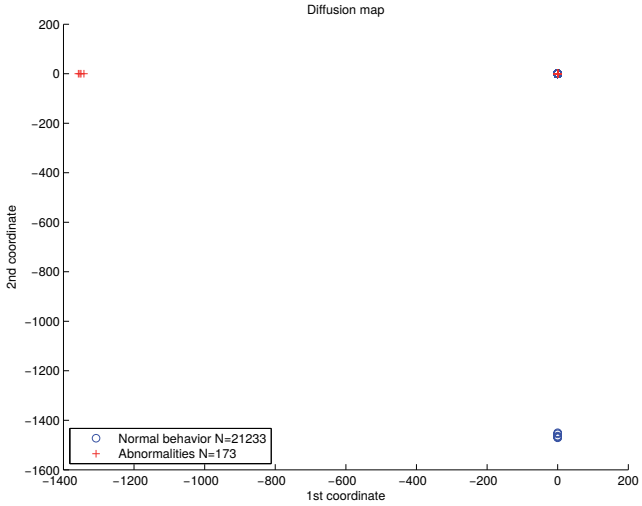

Figure 8: Diffusion map of dataset B2, $\varepsilon = 7$.

The dimensionality reduction framework adapts to the log data. It assumes that only few variables are needed to express the interesting information, and finds a coordinate system that describes the global structure of the data. These coordinates could be used for further analysis of characteristics of anomalous activities.

The main benefits of this framework include:

- The amount of log lines that needs to be inspected is reduced. This is useful for system administrators trying to identify intrusions. The number of interesting log lines is low compared to the total number of lines in the log file.
- The unsupervised nature and adaptiveness of the framework. The proposed methods adapt to the structure of the data without training or previous knowledge. This makes it suitable for exploration and analysis of data without prior examples or attack signatures. This means that the framework also detects zero-day attacks.
- It works on the application layer in the network. The attacks themselves must in some way target the actual applications running on the computer. These logs might be more available than pure low-level network packet data.
- Visualization of text log data. It is much easier to analyze the structure of traffic using visualizations than it is to read raw textual log.

The data in question are rather sparse and the discriminating features are quite evident from the feature matrix. This is the merit of n-gram feature extraction which creates a feature space that separates the normal behavior in a good manner. The features describe the data clearly, and they are easy to process afterwards. Still, an attacker might take advantage of the features used by the intrusion detection system. If the n-gram frequencies of the attack query are similar enough to normal behavior, the currently proposed system could not detect the attacks. Also, if most of the traffic in a single log file consists of attack queries, they will be considered to be normal. This might be a problem in rarely used services.

One advantage of the diffusion map methodology is that it has only one metaparameter, ε. There exists estimation methods for finding the optimal value. If for some reason the threshold sensitivity needs to be changed, ε gives the flexibility to adapt to the global structure. However, the quality of the results is sensitive to changes of this parameter. Values that are too small or large give non-desirable results.

The presented anomaly detection framework performs well on real data. Several actual intrusions are detected. As an unsupervised algorithm this approach is well suited for finding previously unknown intrusions. This method could be applied to offline systems, as well as extended to a real-time intrusion detection system.

There are several points in this framework that could benefit from further research. The feature extraction from the web log is currently done with n-grams. However, this is only one method for it and other text-focused features might better describe the dataset. Furthermore, the dimensionality reduction scheme could be developed to adapt to this kind of data more efficiently, and the quality of the reduction could also be evaluated. The classification method may be improved or changed altogether to another algorithm. Finally, automated root cause detection would make the system more usable in practice.

Acknowledgements

The authors thank Professors Amir Averbuch, Timo Hämäläinen and Tapani Ristaniemi for their continued support. Thanks are extended to Juho Knuuttila and Kristian Siljander for useful discussions and ideas.

References

[1] Mukkamala, S. and Sung, A. *A comparative study of techniques for intrusion detection* (2003).

[2] Patcha, A. and Park, J. *An overview of anomaly detection techniques: Existing solutions and latest technological trends*. *Computer Networks*, 51(12):3448–3470 (2007).

[3] Pietro, R. and Mancini, L. *Intrusion detection systems*. Advances in information security. Springer (2008).

[4] Chandola, V., Banerjee, A., and Kumar, V. *Anomaly detection: A survey*. *ACM Comput. Surv.*, 41(3):1–58 (2009).

[5] Nguyen-Tuong, A., Guarnieri, S., Greene, D., Shirley, J., and Evans, D. *Automatically hardening web applications using precise tainting*. In R. Sasaki, S. Qing, E. Okamoto, and H. Yoshiura, editors, *Security and Privacy in the Age of Ubiquitous Computing*, volume 181 of *IFIP Advances in Information and Communication Technology*, pp. 295–307. Springer Boston (2005).

[6] Ramadas, M., Ostermann, S., and Tjaden, B. *Detecting anomalous network traffic with self-organizing maps*. In G. Vigna, E. Jonsson, and C. Kruegel, editors, *Recent Advances in Intrusion Detection*, pp. 36–54. Springer (2003).

[7] Tran, Q., Duan, H., and Li, X. *One-class support vector machine for anomaly network traffic detection*. *China Education and Research Network (CERNET), Tsinghua University, Main Building*, 310 (2004).

[8] Coifman, R.R., Lafon, S., Lee, A.B., Maggioni, M., Nadler, B., Warner, F., and Zucker, S.W. *Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps*. In *Proceedings of the National Academy of Sciences of the United States of America*, volume 102, p. 7426 (2005).

[9] Coifman, R.R. and Lafon, S. *Diffusion maps*. *Applied and Computational Harmonic Analysis*, 21(1):5–30 (2006).

[10] Bengio, Y., Delalleau, O., Roux, N.L., Paiement, J.F., Vincent, P., and Ouimet, M. *Feature Extraction*, chapter Spectral Dimensionality Reduction, pp. 519–550. Studies in Fuzziness and Soft Computing. Springer Berlin, Heidelberg (2006).

[11] Schclar, A., Averbuch, A., Rabin, N., Zheludev, V., and Hochman, K. *A diffusion framework for detection of moving vehicles*. *Digital Signal Processing*, 20(1):111–122 (2010).

[12] İzmirli, Ö. *Tonal-atonal classification of music audio using diffusion maps*. In *10th International Society for Music Information Retrieval Conference (ISMIR 2009)* (2009).

[13] Keller, Y., Coifman, R., Lafon, S., and Zucker, S. *Audio-visual group recognition using diffusion maps*. *Signal Processing, IEEE Transactions on*, 58(1):403–413 (2010).

[14] Turkka, J., Ristaniemi, T., David, G., and Averbuch, A. *Anomaly detection framework for tracing problems in radio networks*. In *Proc. to ICN 2011* (2011).

[15] Chernogorov, F., Turkka, J., Ristaniemi, T., and Averbuch, A. *Detection of sleeping cells in LTE networks using diffusion maps*. In *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*, pp. 1–5. IEEE (2011).

[16] David, G. *Anomaly Detection and Classification via Diffusion Processes in Hyper-Networks*. Ph.D. thesis, Tel-Aviv University (2009).

[17] Kruegel, C. and Vigna, G. *Anomaly detection of web-based attacks*. In *Proceedings of the 10th ACM conference on Computer and communications security*, pp. 251–261. ACM (2003).

[18] Hubballi, N., Biswas, S., and Nandi, S. *Layered higher order n-grams for hardening payload based anomaly intrusion detection*. In *Availability, Reliability, and Security, 2010. ARES'10 International Conference on*, pp. 321–326. IEEE (2010).

[19] Ringberg, H., Soule, A., Rexford, J., and Diot, C. *Sensitivity of PCA for traffic anomaly detection*. *ACM SIGMETRICS Performance Evaluation Review*, 35(1):109–120 (2007).

[20] Callegari, C., Gazzarrini, L., Giordano, S., Pagano, M., and Pepe, T. *A novel PCA-based network anomaly detection*. In *Communications (ICC), 2011 IEEE International Conference on*, pp. 1–5. IEEE (2011).

[21] David, G. and Averbuch, A. *Hierarchical data organization, clustering and denoising via localized diffusion folders*. *Applied and Computational Harmonic Analysis* (2011).

[22] Damashek, M. *Gauging similarity with n-grams: Language-independent categorization of text*. *Science*, 267(5199):843 (1995).

[23] Han, J. and Kamber, M. *Data mining: concepts and techniques*. Morgan Kaufmann (2006).

[24] Abdi, H. and Williams, L. *Principal component analysis*. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(4):433–459 (2010).

[25] Lee, J. and Verleysen, M. *Nonlinear dimensionality reduction*. Springer Verlag (2007).

[26] Nadler, B., Lafon, S., Coifman, R., and Kevrekidis, I.G. *Diffusion maps – a probabilistic interpretation for spectral embedding and clustering algorithms*. In T.J. Barth, M. Griebel, D.E. Keyes, R.M. Nieminen, D. Roose, T. Schlick, A.N. Gorban, B. Kégl, D.C. Wunsch, and A.Y. Zinovyev, editors, *Principal Manifolds for Data Visualization and Dimension Reduction*, volume 58 of *Lecture Notes in Computational Science and Engineering*, pp. 238–260. Springer Berlin Heidelberg (2008).

[27] Chung, F.R.K. *Spectral Graph Theory*, p. 2. AMS Press, Providence, R.I (1997).

[28] von Luxburg, U. *A tutorial on spectral clustering*. *Statistics and Computing*, 17:395–416 (2007).

[29] Ng, A.Y., Jordan, M.I., and Weiss, Y. *On spectral clustering: Analysis and an algorithm*. In *Advances in Neural Information Processing Systems 14*, pp. 849–856. MIT Press (2001).

[30] Shi, J. and Malik, J. *Normalized cuts and image segmentation*. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888 –905 (2000).

[31] Kannan, R., Vempala, S., and Vetta, A. *On clusterings: Good, bad and spectral*. *J. ACM*, 51:497–515 (2004).

[32] Meila, M. and Shi, J. *Learning segmentation by random walks*. In *NIPS*, pp. 873–879 (2000).

[33] Sipola, T., Juvonen, A., and Lehtonen, J. *Anomaly detection from network logs using diffusion maps*. In L. Iliadis and C. Jayne, editors, *Engineering Applications of Neural Networks*, volume 363 of *IFIP Advances in Information and Communication Technology*, pp. 172–181. Springer Boston (2011).

[34] Hein, M. and Audibert, J. *Intrinsic dimensionality estimation of submanifolds in* $\mathbb{R}^d$. In *Proceedings of the 22nd international conference on Machine learning*, pp. 289–296. ACM (2005).

[35] Coifman, R., Shkolnisky, Y., Sigworth, F., and Singer, A. *Graph Laplacian tomography from unknown random projections*. *Image Processing, IEEE Transactions on*, 17(10):1891–1899 (2008).

PIII

ADAPTIVE FRAMEWORK FOR NETWORK TRAFFIC CLASSIFICATION USING DIMENSIONALITY REDUCTION AND CLUSTERING

by

Antti Juvonen and Tuomo Sipola 2012

Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2012 4th International Congress on, pp. 274–279

Adaptive Framework for Network Traffic Classification Using Dimensionality Reduction and Clustering

Antti Juvonen, Tuomo Sipola
Department of Mathematical Information Technology
University of Jyväskylä
Jyväskylä, Finland
{*antti.k.a.juvonen, tuomo.sipola*}*@jyu.fi*

***Abstract*—Information security has become a very important topic especially during the last years. Web services are becoming more complex and dynamic. This offers new possibilities for attackers to exploit vulnerabilities by inputting malicious queries or code. However, these attack attempts are often recorded in server logs. Analyzing these logs could be a way to detect intrusions either periodically or in real time. We propose a framework that preprocesses and analyzes these log files. HTTP queries are transformed to numerical matrices using n-gram analysis. The dimensionality of these matrices is reduced using principal component analysis and diffusion map methodology. Abnormal log lines can then be analyzed in more detail. We expand our previous work by elaborating the cluster analysis after obtaining the low-dimensional representation. The framework was tested with actual server log data collected from a large web service. Several previously unknown intrusions were found. Proposed methods could be customized to analyze any kind of log data. The system could be used as a real-time anomaly detection system in any network where sufficient data is available.**

***Keywords*-intrusion detection; anomaly detection; n-grams; diffusion map; k-means; data mining; machine learning**

I. Introduction

Most web servers log their traffic. This log data is rarely used, but it could be analyzed in order to find anomalies or to visualize the traffic structure. Acquiring the data does not require any modifications to the actual web service, because data logging is usually done by default. Different kinds of log files are created, but for this study the most interesting log is the one containing HTTP queries.

One important application for network traffic analysis is anomaly detection. This is done using *intrusion detection systems* (IDS) [1]. Many of these analyze the transport layer, mostly TCP packet data. However, we try to find anomalies and other information from application layer log files. HTTP queries include this information. Many attacks, such as SQL injections, can be detected from this layer.

Log files are in textual form. Therefore, some preprocessing is needed to transform query strings into numerical matrices. This can be done using information about n-gram analysis, which is described in section III-A. Calculating the frequencies of individual substrings in the data results in a numerical data matrix.

After preprocessing, many data mining methods can be used to visualize and analyze the logs. We perform dimensionality reduction and clustering. After visualizing the results it is possible to interpret the findings and make more detailed analysis about the web service traffic.

We propose a framework that processes textual log files in order to visualize them. We are trying to find patterns and anomalies using only log files containing HTTP queries. The framework is adaptive, and individual parts of it can be changed. For example, the choice of dimensionality reduction method or clustering algorithm can be done based on current needs.

The proposed methods use data mining principles, and they work as an IDS and network traffic visualization and analysis tool. Using the framework, we are trying to find whether the textual HTTP query logs actually include some information about the traffic structure. This information could then be used to classify users and individual queries and to find anomalies and intrusion attempts.

II. Related work

We have previously researched log data preprocessing and anomaly detection [2], [3]. This research focused on finding intrusions from log data. We now extend this methodology to further analyze and cluster the structure of the traffic. This is done by adding more accurate clustering algorithms into the framework.

Principal component analysis has been widely used in network intrusion detection and traffic analysis. Xu et al. used PCA and support vector machine to reduce dimensions and classify network traffic in order to find intrusions [4]. Taylor et al. used PCA and clustering analysis to find network anomalies and perform traffic screening [5].

Diffusion methods have been applied in network traffic analysis. These studies have concentrated on low-level IP packet features. These features are numerical and the network architecture differs from our study [6] [7]. Network server logs have also been analyzed using diffusion maps and spectral clustering [2] [3].

This is the authors' postprint version of the article. The original print version appeared as: A. Juvonen and T. Sipola, "Adaptive Framework for Network Traffic Classification Using Dimensionality Reduction and Clustering," *Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2012 4th International Congress on*, 2012, pp. 274–279. Available online at http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6459678&isnumber=6459638

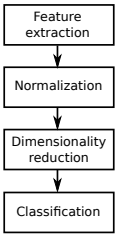

Figure 1. The data mining process

III. Methodology

Our overall approach is rooted in the data mining process [8], [9]. This approach is method-centric as our research is focused on the data processing and not business aspects. The data mining process of our study flows as follows:

1) Data selection.
2) Extract n-gram features from the text data.
3) Normalize the feature matrix.
4) Reduce the number of dimensions to obtain low-dimensional features.
5) Classify or cluster the low-dimensional data presentation.
6) Interpret the found patterns or anomalies.

The process is presented in figure 1.

A. Feature extraction

The log files are in text format. Therefore, it is necessary to transform the log lines into numerical vectors which then can be used in further mathematical analysis. We use n-gram analysis to process log files into numerical matrices. It has been used e.g. in judging similarity in text documents [10], analyzing protein sequences [11] and detecting malicious code [12].

N-grams are consecutive sequences of n characters [10]. Each log line corresponds to a feature vector containing the frequencies of each individual n-gram found in the data. The list of n-grams appearing in the data can be found using n-character-wide sliding window moved along the string one character at a time [10].

Let us consider the following example. Having two strings containing the words `anomaly` and `analysis`, we can construct the feature matrix in the following way:

an	no	om	ma	al	ly	na	ys	si	is
1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1

In this study, 2-grams are used. However, it is possible to use longer n-grams as well. This will of course results in more dimensions in the matrix, because there are more unique n-grams. The theoretical maximum number of individual 2-grams using ASCII-characters is $256^2 = 65536$, but in practice this is usually not the case. This is due to the fact that many characters are never actually used [10].

B. Normalization

Normalization ensures that the features of the input data are in the same scale. We use logarithm for this purpose. To avoid complex numbers, the input must be above zero. The normalization function for a point x_i in the dataset is

$$f_n(x_i) = \log(x_i - X_{min} + 1),$$

where X_{min} is the minimum of all the values in the dataset.

C. Principal Component Analysis

Principal Component Analysis (PCA) [13] is perhaps the best-known dimensionality reduction technique. It has many practical applications, such as computer vision and image compression [14].

The PCA process is explained in more detail in [14]. First we must substract the mean from the original data to make the data have zero mean. Then the covariance matrix must be calculated. From the covariance matrix we can then calculate eigenvalues and the corresponding eigenvectors. If we choose d eigenvectors that contain most of the variance, we get a lower dimension representation of the original data with d dimensions. This is done by choosing the d eigenvectors as columns for a matrix, and multiplying the mean-centered data with this matrix. For visualization purposes it is necessary to choose either 2 or 3 dimensions, ie. eigenvectors.

Calculating PCA is relatively simple, but it will only work in linear cases. If the dataset is non-linear, some other dimensionality reduction method must be used. PCA can also give inaccurate results if there are outliers in the data.

D. Diffusion Map

Diffusion map (DM) reduces the dimensions while retaining the diffusion distances in the high-dimensional space as Euclidean distances in the low-dimensional space. This reduction is non-linear. The goal is to move from n-dimensional space to a low-dimensional space with d dimensions, when $d \ll n$ [15].

One measurement $x_i \in \mathbb{R}^n$ in this study corresponds to one line in the log file. Given the dataset $X = \{x_1,\ x_2,\ x_3,\ \ldots\ x_N\}$ the affinity matrix $W(x_i, x_j) = \exp\left(\frac{-||x_i - x_j||^2}{\epsilon}\right)$ describes the affinities between measurements. Here we have used the Gaussian kernel. Matrix $P = W^{-1}K$ represents the transition probabilities between the measurements. Next, the matrix D collects the row sums to its diagonal. Using the singular value decomposition (SVD) of matrix $\tilde{P} = D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$ we obtain the eigenvectors v_k and eigenvalues λ_k.

The diffusion map maps the measurements x_i to low dimensions by giving each high-dimensional point coordinates in the low dimensions: $x_i \rightarrow [\lambda_1 v_1(x_i),\ \lambda_2 v_2(x_i) \dots\ \lambda_d v_d(x_i)]$. These new coordinates lose some of the information contained in the original dataset. However, the accuracy is usually good enough for later classification. Even though there is loss of information, the classification problem becomes easier.

E. Traffic clustering using k-means algorithm

We use cluster analysis to divide network traffic into meaningful groups. In this way we can capture the natural structure of the data [16].

K-means algorithm was introduced in 1955 and huge number of other clustering algorithms have been introduced since then, but k-means method is still widely used [17]. It is a prototype-based clustering technique [16]. Given the original data $X = x_i$, where $i = 1, .., n$, the goal is to cluster the data points into k clusters. The mean of cluster k is now μ_k, and the mean squared error (MSE) between a data point and the cluster mean is $||x_i - \mu_k||^2$. This leads to an optimization problem where the MSE for each datapoint in each cluster must be minimized.

The problem can be solved following these steps [18]:

1) Select initial centers for k clusters.
2) Assign each datapoint to its closest cluster centroid.
3) Compute the new cluster centers by calculating the mean of the datapoints in each cluster.

Steps 2 and 3 are repeated until a stopping criterion is met. Usually this means that the partitioning has not changed since the last iteration, and thus a local optimum solution for the problem has been found.

Choosing the number of clusters is not trivial, but there are many methods for calculating the number of clusters, such as Davies-Bouldin index, described in [19]. This algorithm takes into account both scatter within a cluster and separation between different clusters. Davies-Bouldin index is used in this study to determine the number of clusters for each resource.

The algorithm can give different results depending on the initialization, because it only finds the local optimal solution. This can happen especially when using random initialization. However, this problem can be overcome by running k-means multiple times and choosing the clustering results that gives the smallest squared error [17]. There are also many other algorithms for choosing the initial cluster centroids.

IV. Experimental setup

Figure 2 shows the architecture of the web service that was analyzed. It contains many servers that offer the same service to users using load balancing. Proprietary log files were acquired from this service. These files then need to be preprocessed into numerical matrices. The data and this process are described in this section.

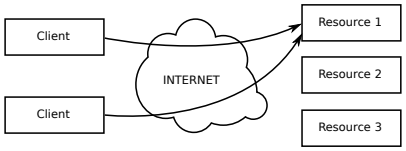

Figure 2. Experiment architecture.

A. Data acquisition

The data have been collected from a large web service. Apache web servers are used, and they log data using Combined Log Format, example of a single log line:

```
127.0.0.1 - -
[01/January/2012:00:00:01 +0300]
"GET /resource.php?parameter1=value1
&parameter2=value2
HTTP/1.1"
200 2680
"http://www.address.com/webpage.html"
"Mozilla/5.0
(SymbianOS/9.2;...)"
```

For this analysis, the HTTP query part is used because it contains the only information that a user can input. This offers possibilities for attackers. The other information, such as time, can be used when further analyzing individual log lines (e.g. for finding anomalies or attacks). On the other hand, HTTP query parameters and their values are dynamic and changing, offering valuable information about this dynamic web service. Analyzing this information will explain a lot about the structure of the traffic. The parameter values in data used in this study were dynamic and changing, and also not always human-readable. Therefore, analyzing these fields has to be done automatically with mathematical methods.

B. Data preprocessing

The first step is to select the data for analysis. The original log file contains approximately 4 million log lines. However, most of these lines contain only static queries. Static lines do not contain changing parameter values. These lines do not offer a lot of information, because they are practically identical in the used dataset. In addition, static lines do not contain information about user input, meaning it is not possible to detect attacks from those log lines alone. On the other hand, dynamic web resources are changing and also vulnerable, so dynamic lines containing parameters and parameter values are interesting and can offer more information about the web service. Therefore, static log lines are filtered out, leaving only approximately 221 000 lines to be inspected and clustered. This data selection reduces the size considerably and creates a database of the most interesting aspects of the log files.

After the first filtering stage, log files are divided into smaller files according to resource URI. This is because different resources accept different parameter values, so they do not have much to do with each other. This makes anomaly detection from full data very difficult and inaccurate. However, traffic structure inside single resource is more consistent. After this division, smaller logfiles can be analyzed independently. It makes sense to further analyze the largest log files, because some of the resources contain only a few lines. These lines have to be omitted.

Finally, in order to create data matrices out of textual log data, n-gram analysis is performed. This process is explained in III-A.

V. Results

For this research, 3 relatively large resources are selected for further analysis and clustering. Resource 1 contains 10935 lines and 414 dimensions, and is the simplest in terms of HTTP query parameters. Resource 2 contains only 2982 lines, but the number of dimensions is 3866, which makes analysis challenging. Also, the parameters are clearly not human-readable, i.e. it is impossible to say anything about the queries by looking at the parameter string alone. Resource 3 is the largest, including 21406 lines and 991 dimensions.

All the resources are analyzed using the proposed framework. The feature data are normalized with the logarithm function. PCA and diffusion map reduce the dimensionality of the normalized feature matrices. Clustering then reveals the structure of the data and facilitates the interpretation of the log files.

Resource 1 contains 10935 lines and 414 dimensions. The results for diffusion map and principal component analysis are presented in figures 3 and 4, respectively. This resource is a simple example, mainly useful in validating that the methods do give satisfactory results. The only difference is that DM separates the data points more clearly. Due to this separation we get 3 clusters, instead of 2 as in PCA. The biggest cluster contains varying parameter values. The parameters in smaller clusters are almost the same within that cluster. However, this behavior is easy to see directly from the log lines. The framework visualizes the traffic well, but in this case we do not obtain any new information about the data.

Resource 2 is the smallest in this research, containing only 2982 requests. However, the number of dimensions is 3866. This means that there are more dimensions than data points, which is always a problem in classification tasks. Despite that, we obtained clear results. The DM and PCA results presented in figures 5 and 6. The results are essentially identical, figures look slightly different but the clustering is exactly the same. This might mean that variables are linearly dependent, otherwise PCA would not work well. The log lines themselves are not human-readable, containing error

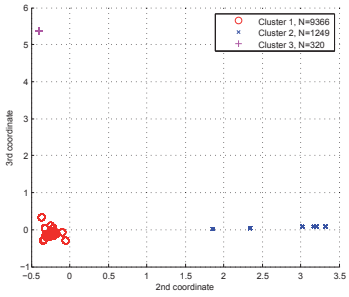

Figure 3. Resource 1, diffusion map.

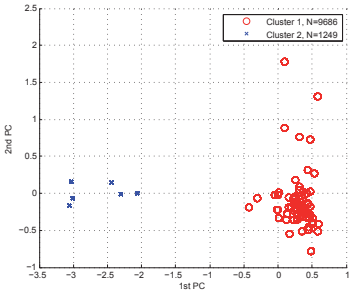

Figure 4. Resource 1, PCA.

tickets that have a seemingly random code as the parameter value. However, as can be seen from the figures, there are clearly two distinct clusters that can be seen using both dimensionality reduction methods. This behavior was not previously known and requires more detailed analysis with the administrator of the web service.

Resource 3 is the largest with 21406 lines and 996 dimensions. It also shows that DM (in figure 7) and PCA (in figure 8) can sometimes give very different results. Normal parameter values in this resource are long and varied. This results in PCA not being able to clearly distinguish any clusters. For this reason, k-means clustering was not performed for resource 3 PCA datapoints. However, with DM the results are very meaningful. Normal traffic clearly forms it's own cluster, while 2 other groups are apparent. Cluster 2 with 5 datapoints does not contain anything malicious, but is slightly different from other normal datapoints. The most interesting finding in this data is cluster 3, which contains 4 lines. All of these lines contain an SQL injection attack, where an attacker tried to include malicious SQL queries as parameter values. The 2nd DM coordinate clearly separates attacks from rest of the data, meaning that in this case only one dimension is needed for anomaly detection.

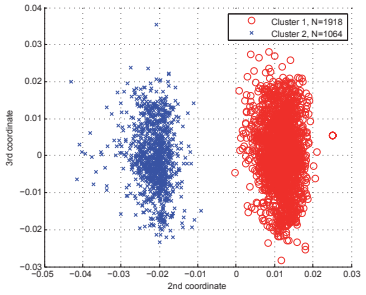

Figure 5. Resource 2, diffusion map.

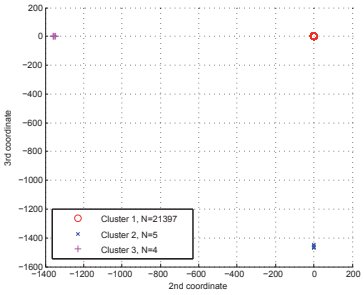

Figure 7. Resource 3, diffusion map.

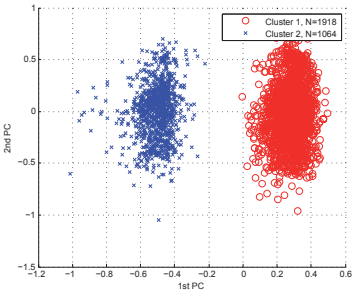

Figure 6. Resource 2, PCA.

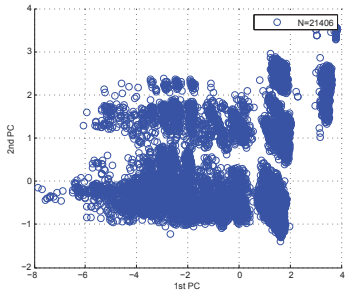

Figure 8. Resource 3, PCA.

In all of the figures, except PCA for resource 3, it can be seen that the separation of clusters is clear. A simple clustering method such as spectral clustering or decision tree could be used.

VI. Conclusion

We presented a framework for preprocessing, clustering and visualizing web server log data. This framework was used for anomaly detection, visualization and explorative data analysis based only on application layer data. Individual parts of the architecture can be changed for different results. For example, k-means clustering can be replaced with hierarchical linkage clustering method.

The results clearly indicate that there are traffic structures that can be visualized from HTTP query information. The data forms distinct clusters and contains anomalies as well. The sensitivity for outliers creates some problems for PCA, which means that it can be challenging to use it for anomaly detection. Diffusion maps give good results, but more research would have to be done to get more information about performance issues. In some cases the results for PCA and DM are nearly identical, while in other cases they differ greatly. PCA is faster but cannot be used with non-linear data. DM seems to work in most situations but can be too slow.

Traffic clustering can give new information about the users of a web service. This information could be used to categorize users more accurately. This gives opportunities for more accurate advertising or offering better content for users. Finding anomalies gives information about possible intrusion attempts and other abnormalities.

To make the framework more usable, it should be automatic and work in real-time. More research is needed to find the most generally usable algorithms for each phase in the architecture. In addition, log data tends to be high in volume, so performance issues might become a problem. For dimensionality reduction the number of dimensions is not trivial. Also, the number of clusters must be determined depending on the chosen clustering algorithm. Real-time functioning requires changes in preprocessing and limits the dimensionality reduction options. For this purpose, PCA might be a good method, since projection of new points into lower dimensions is simply a matter of matrix multiplication. However, the limitations mentioned previously still apply.

Using data mining methods, underlying structure and anomalies are found from HTTP logs and these results can be visualized and analyzed to find patterns and anomalies.

REFERENCES

[1] K. Scarfone and P. Mell, "Guide to intrusion detection and prevention systems (idps)," *NIST Special Publication*, vol. 800, no. 2007, p. 94, 2007.

[2] T. Sipola, A. Juvonen, and J. Lehtonen, "Anomaly detection from network logs using diffusion maps," in *Engineering Applications of Neural Networks*, ser. IFIP Advances in Information and Communication Technology, L. Iliadis and C. Jayne, Eds. Springer Boston, 2011, vol. 363, pp. 172–181.

[3] ——, "Dimensionality reduction framework for detecting anomalies from network logs," *Engineering Intelligent Systems*, 2012, forthcoming.

[4] X. Xu and X. Wang, "An adaptive network intrusion detection method based on pca and support vector machines," *Advanced Data Mining and Applications*, pp. 731–731, 2005.

[5] C. Taylor and J. Alves-Foss, "Nate: N etwork analysis of a nomalous t raffic e vents, a low-cost approach," in *Proceedings of the 2001 workshop on New security paradigms*. ACM, 2001, pp. 89–96.

[6] G. David, "Anomaly Detection and Classification via Diffusion Processes in Hyper-Networks," Ph.D. dissertation, Tel-Aviv University, 2009.

[7] G. David and A. Averbuch, "Hierarchical data organization, clustering and denoising via localized diffusion folders," *Applied and Computational Harmonic Analysis*, 2011.

[8] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "The KDD process for extracting useful knowledge from volumes of data," *Commun. ACM*, vol. 39, pp. 27–34, November 1996. [Online]. Available: http://doi.acm.org/10.1145/240455.240464

[9] ——, "From data mining to knowledge discovery in databases," *AI magazine*, vol. 17, no. 3, p. 37, 1996.

[10] M. Damashek, "Gauging similarity with n-grams: Language-independent categorization of text," *Science*, vol. 267, no. 5199, p. 843, 1995.

[11] M. Ganapathiraju, D. Weisser, R. Rosenfeld, J. Carbonell, R. Reddy, and J. Klein-Seetharaman, "Comparative n-gram analysis of whole-genome protein sequences," in *Proceedings of the second international conference on Human Language Technology Research*. Morgan Kaufmann Publishers Inc., 2002, pp. 76–81.

[12] T. Abou-Assaleh, N. Cercone, V. Keselj, and R. Sweidan, "N-gram-based detection of new malicious code," in *Computer Software and Applications Conference, 2004. COMPSAC 2004. Proceedings of the 28th Annual International*, vol. 2. IEEE, 2004, pp. 41–42.

[13] H. Hotelling, "Analysis of a complex of statistical variables into principal components." *Journal of educational psychology*, vol. 24, no. 6, pp. 417–441, 1933.

[14] L. Smith, "A tutorial on principal components analysis," *Cornell University, USA*, vol. 51, p. 52, 2002.

[15] R. R. Coifman and S. Lafon, "Diffusion maps," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 5–30, 2006.

[16] P. Tan, M. Steinbach, and V. Kumar, "Cluster analysis: Basic concepts and algorithms," *Introduction to data mining*, pp. 487–568, 2006.

[17] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.

[18] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Prentice Hall., 1988.

[19] D. Davies and D. Bouldin, "A cluster separation measure," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 2, pp. 224–227, 1979.

PIV

GROWING HIERARCHICAL SELF-ORGANIZING MAPS AND STATISTICAL DISTRIBUTION MODELS FOR ONLINE DETECION OF WEB ATTACKS

by

Mikhail Zolotukhin, Timo Hämäläinen and Antti Juvonen 2013

Web Information Systems and Technologies. Lecture Notes in Business Information Processing, Vol. 140, pp. 281–295

PV

COMBINING CONJUNCTIVE RULE EXTRACTION WITH DIFFUSION MAPS FOR NETWORK INTRUSION DETECTION

by

Antti Juvonen and Tuomo Sipola 2013

The Eighteenth IEEE Symposium on Computers and Communications (ISCC 2013), pp. 411–416

Combining Conjunctive Rule Extraction with Diffusion Maps for Network Intrusion Detection

Antti Juvonen, Tuomo Sipola
Department of Mathematical Information Technology
University of Jyväskylä
Jyväskylä, Finland
{antti.k.a.juvonen, tuomo.sipola}@jyu.fi

***Abstract*—Network security and intrusion detection are important in the modern world where communication happens via information networks. Traditional signature-based intrusion detection methods cannot find previously unknown attacks. On the other hand, algorithms used for anomaly detection often have black box qualities that are difficult to understand for people who are not algorithm experts. Rule extraction methods create interpretable rule sets that act as classifiers. They have mostly been combined with already labeled data sets. This paper aims to combine unsupervised anomaly detection with rule extraction techniques to create an online anomaly detection framework. Unsupervised anomaly detection uses diffusion maps and clustering for labeling an unknown data set. Rule sets are created using conjunctive rule extraction algorithm. This research suggests that the combination of machine learning methods and rule extraction is a feasible way to implement network intrusion detection that is meaningful to network administrators.**

Keywords*—*Intrusion detection, anomaly detection, n-gram, rule extraction, diffusion map, data mining, machine learning.

I. Introduction

Web services and networks have become more and more complex in the past years. This means that services and servers face new threats and attacks. *Intrusion detection systems* (IDS) are used to detect these attacks. An IDS works generally using one of two detection principles, *signature-based* and *anomaly-based* detection [1]. Signatures are predetermined attack rules that can be used to trigger an alarm when a user's behavior matches the signature. Previous information about intrusions is required for creating these rules. This leads to a low rate of false alarms, but new and unknown threats cannot be detected. On the other hand, anomaly-based detection systems try to detect traffic that deviates from the normal behavior. New attacks can be detected but this methodology will also lead to some false alarms. Both principles can also be combined to a so-called *hybrid intrusion detection system* [2]. Figure 1 shows a simplified block diagram of the different intrusion detection approaches, demonstrating how our system relates to other approaches.

Information security reseachers have been interested in intrusion detection systems extensively, and surveys describing advances in the field have been published [3], [4]. Many machine learning methods, such as self-organizing maps [5] and support vector machines [6] have been used to cluster data and detect anomalies in these systems. Various hybrid systems combining signature and anomaly-based detection have been used [2], [7]. A two-stage adaptive hybrid system for IP level intrusion detection has also been recently devised. A probabilistic classifier detects anomalies and a hidden Markov model narrows down attacker addresses [8]. Recently genetic algorithms have been widely used in anomaly detection and misuse detection [9], [10]. Artificial immune systems have raised the interest of intrusion detection researchers [11]. More traditional methods such as k nearest neighbors are also still researched because they can be combined with other methods, for example Dempster-Shafer theory of evidence [12]. A distributed environment has been proposed where intelligent agents analyze the network connections using data mining with association rule mining [13]. Moreover, in our previous work we have researched intrusion detection using dimensionality reduction and clustering to find anomalies from network traffic [14], [15].

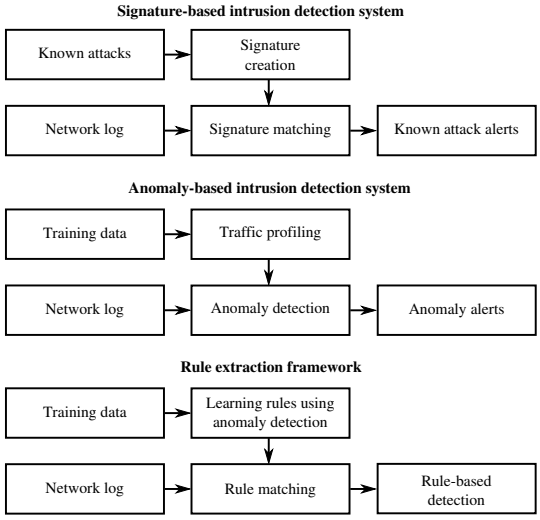

Fig. 1. Different IDS principles.

The problem with deploying anomaly detection systems in the commercial sector is that some algorithms, such as neural networks, work like a black box [16]. The systems are automated and it is difficult to know exactly how the decisions are made. To overcome this problem, *rule extraction* methods have been proposed [17]. These rules can be directly applied to the original data for efficient web traffic filtering. In addition,

This is the authors' postprint version of the article. The original print version appeared as: A. Juvonen and T. Sipola, "Combining conjunctive rule extraction with diffusion maps for network intrusion detection," in *In The Eighteenth IEEE Symposium on Computers and Communications (ISCC 2013)*. IEEE 2013. Available online at http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6754981&isnumber=6754912

this symbolic knowledge can be read and inspected by humans. This can lead to a better understanding of the data and will aid user acceptance especially in real-life company networks.

One way of extracting these rules is taking a decompositional approach [18]. This can be achieved, e.g., by decomposing a neural network architecture. However, methods of this type are algorithm dependent and the rules themselves may not be sufficiently comprehensible [16]. Another way to extract rules is by using pedagogical approach [17]. This approach takes only the input data and output results into account. Therefore, it is not specific to any particular classification method. Any suitable algorithm can be used to find anomalies or cluster data. Also, the produced rules are directly related to original data and can therefore be easily understood. Because of these reasons, we take the pedagogical approach in our system. Various methods have been used to create different kinds of rule sets and trees. Recent research seems to focus on methods based on heuristic algorithms or creating intelligent wrapper methods [19]. A less researched option is to use conjunctive rules [17]. These rule extraction methods should not be confused with association rule mining [20].

We propose a framework for detecting network anomalies and extracting rules from a data set. Figure 1 shows how it differs from other common approaches. This framework is a supplementary module for signature-based intrusion detection systems, such as next generation firewalls. In this approach, network logs or other similar data is collected and preprocessed to extract features and form numerical matrices to be analyzed further. The dimensionality of this data is reduced for more efficient clustering. After clustering the data to normal and anomalous traffic, the obtained clustering is used to create labels for the data. Subsequently, this information is used to create a rule set for the high-dimensional features. This rule set can then be used to classify traffic and detect intrusions in real time. The proposed framework enables rule creation in an unsupervised manner for previously unknown data. Our contribution is combining unsupervised data analysis with rule extraction techniques to create an online anomaly detection system.

II. Methodology

The proposed framework uses training data to create a rule set which can then be used to classify testing data or actual network traffic data. Thus, our approach is divided into two phases: *rule set learning* and *traffic classification*. The first phase takes the approach of learning the clustering of the data using dimensionality reduction and creating conjunctive rules to describe these clusters in the initial feature space. These rules will then be used to classify new incoming traffic in the second phase. This process is described in Figure 2, which shows the needed input data sets, produced rule set and classification results.

The rule set learning phase aims to find rules that describe the training data. This is done by clustering and labeling the training data set. The resulting rule set classifies data according to the obtained clustering. Architecture of the rule set learning process is shown in Figure 3, which shows the labeling and rule extraction phases in more detail. The methods in individual modules are not fixed, meaning that the specific methods

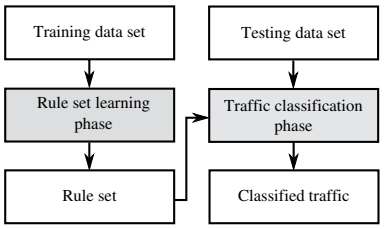

Fig. 2. Block diagram of the overall process.

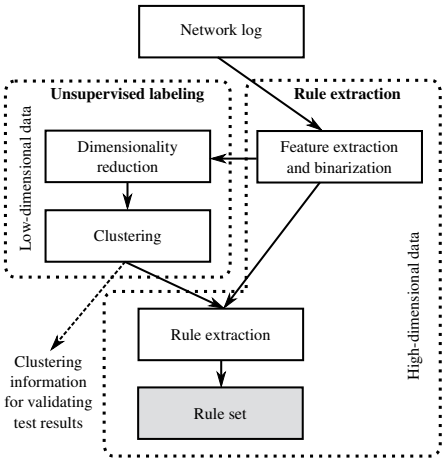

Fig. 3. Block diagram of the rule set learning process.

can be changed if better alternatives are found. The rule set learning phase consists of the following steps:

- Feature extraction from training data
- Unsupervised labeling
  - Dimensionality reduction
  - Clustering
- Rule extraction

In the traffic classification phase new incoming traffic is preprocessed and classified using the generated rule set. Because of the conjunctive nature of the rules simple matching is sufficient. This phase validates how well the rules apply to data that was not part of the training data set. The steps are as follows:

- Feature extraction from testing data
- Classification by rule matching

The following subsections describe the methods used in previously mentioned phases in detail.

A. Feature extraction

Network log files consist of text lines that need to be converted to numeric feature vectors. An n-gram is a consecutive sequence of n characters that represents extracted semantic

information [21]. Our study uses 2-gram features generated from the network logs. This approach produces a rather sparse feature matrix [14]. The rule extraction algorithm works with symbolic conjunctive rules. This means that only nominal and binarized data can be used. Converting data to this kind of format ensures that the feature matrix may be used with the overall learning pipeline.

The feature matrix consists of binary values representing whether an n-gram is present in a specific log line or not. Let us consider the following example. Having two strings containing the words `anomaly` and `analysis`, we can construct the feature matrix in the following way:

an	no	om	ma	al	ly	na	ys	si	is
1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1

In this study, 2-grams are used. However, it is possible to use longer n-grams as well. This will result in more dimensions in the matrix, because there will be more unique n-grams, slowing down the process. For the purposes of this research, 2-grams contained enough information for separating normal and anomalous traffic. Also, using $n = 1$ will give the character distribution. Single characters may not contain enough semantic information, and therefore higher values of n are often used.

B. Dimensionality reduction and clustering

Clustering high-dimensional data is facilitated by dimensionality reduction. We employ diffusion map training to identify the attacks in the training data set. The features describing the dataset are numerous and sometimes hard to interpret together. Therefore, a dimensionality reduction approach using diffusion map is taken. Diffusion map training produces a low-dimensional model of the data, which reveals the internal structure of the dataset and facilitates anomaly detection. In addition, it can cope with non-linear dependencies in the data. Diffusion map retains the diffusion distance in the initial feature space as the Euclidean distance in the low-dimensional space [22], [23], [24].

One log line is represented by feature vector $x_i \in \mathbb{R}^n$. The whole data set is $X = \{x_1,\ x_2,\ x_3,\ \ldots\ x_N\}$, from which the affinity matrix

$$W(x_i, x_j) = \exp\left(\frac{-||x_i - x_j||^2}{\epsilon}\right)$$

can be calculated. As seen, the Gaussian kernel is used for the distance matrix and the bandwith parameter ϵ is selected from the optimal region in the weight matrix sum [25]. D, which collects W's row sums on its diagonal, and the transition matrix $P = D^{-1}W$ form the symmetric matrix

$$\tilde{P} = D^{\frac{1}{2}} P D^{-\frac{1}{2}} = D^{-\frac{1}{2}} W D^{-\frac{1}{2}}.$$

The singular value decomposition (SVD) of $\tilde{P}$ yields the eigenvectors v_k and eigenvalues λ_k. Now, the low-dimensional coordinates corresponding each original log line are found: $x_i \to [\lambda_1 v_1(x_i),\ \lambda_2 v_2(x_i) \ldots\ \lambda_d v_d(x_i)]$. Most of the information is retained in the first eigenvectors and less meaningful ones are left out. Some information is lost because not all eigenvectors are used, but lower dimensionality makes clustering easier.

The k-means method is used to group the data points into clusters. This method is simple and well-known clustering algorithm and it has been used in various data mining tasks. The algorithm description and examples of use can be found in literature [26], [27], [28]. The k-means method relies heavily on the parameter k which determines the number of clusters. Silhouette expresses the quality of clustering for each data point. The optimal number of clusters for the k-means is determined using average silhouette value [29]. An alternative clustering method could be used.

The obtained clustering is believed to describe behavior of the data. If the high-dimensional features can differentiate normal and intrusive behavior, this should be apparent from the resulting low-dimensional clusters. The actual nature of the clusters should be confirmed with domain area experts.

If performance becomes an issue with larger data sets, the learning process could be expanded with out-of-sample extension. However, representative selection of training data is usually a more challenging problem.

C. Rule extraction

A rule is a way to determine the class of a data point based on certain conditions. Ideally a rule would be easily interpretable by a network administrator. All the possible rules span such a huge space that it is not feasible to go through all of them. This means that a sub-optimal but efficient method needs to be used. Such systems have been applied with neural networks [17], [16] and support vector machines [30], [31], [32].

Conjunctive rule is a logical expression containing truth values about the inclusion of binary features. These rules tell whether a symbol should be included, excluded or if it does not matter. Let us assume that we have binary features a, b, c, d, e. Thus, the feature matrix contains five columns corresponding to each binary feature. For the sake of example we have a rule set containing three rules:

$$\begin{aligned} r_1 &= \neg a && \text{for class } c_1, \\ r_2 &= a \wedge b \wedge c \wedge \neg d \wedge e && \text{for class } c_1, \\ r_3 &= a \wedge b \wedge \neg c && \text{for class } c_2. \end{aligned}$$

The rule set for class c_1 would be expressed in logical form as $R_1 = r_1 \vee r_2$. In practice, there are usually multiple rules for each class. Note that in rule r_1, values of features b, c, d and e do not matter. Similarly, for r_3 values of d and e can be anything.

For implementation purposes, the rules are expressed as vectors. The length of these vectors is equivalent to the number of features. The logical truth values are converted to 1 and -1. The values that do not matter are expressed as 0. In the previous example, the rules would be vectors of length 5. Rule r_1 is expressed as a vector $(-1 \quad 0 \quad 0 \quad 0 \quad 0)$. It is easy to match feature vectors to this kind of rule vectors. Note that in

this research a rule symbol corresponds to an n-gram feature as described in II-A.

The conjunctive rule extraction algorithm [17] finds rule-based classifier that approximates the clustering obtained in the unsupervised labeling step. Conjunctive rule extraction is presented in Algorithm II.1. Note that a rule r consists of symbols $r = s_1 \wedge s_2 \wedge s_3 \wedge \ldots \wedge s_n$.

Algorithm II.1: Conjunctive rule extraction.

Input: data points E, classes C
Output: rules R_c that cover E with classification C

```
repeat
  e := get new training observation from E
  c := get the classification of e from C
  if e not covered by the rules R_c then
    r := use e as basis for new rule r
    for all symbols s_i in r do
      r' = r with symbol s_i dropped
      if all instances covered by r are of the same class
      as e then
        r := r'
      end if
    end for
    add rule r to the rule set R_c
  end if
until all training data analyzed
```

The obtained rules separate the training data into the clusters. These rules can now be matched to new incoming data points. Their performance depends on how well the training data covers the behavior of the data. If the point matches one of the rules, the exact type of the abnormal or normal state can be interpreted. If a data point does not fall under any of the rules, then it can be considered abnormal.

Created rules are valid for the classification task while the essential profile of the data remains the same. This is often not the case for extended periods of time, especially for network traffic or similar data. Therefore, rules can be recreated periodically, e.g., daily.

III. Results

This section contains the classification results using real-world network log data. The goal is to perform preliminary validation on real data to test the feasibility of rule extraction in a practical IDS application. The previously described framework was implemented and applied to this data. Data acquisition and analysis are presented below. These results illustrate that the rule set learning phase works on a data that comes from a real-world source.

A. Data acquisition and processing

We use the same network log database that has been used in our previous related research [15]. The data comes from a real-life web server used by a company. Different kinds of intrusion attempts and other abnormal log lines are included in the data. We examine two log files that correspond to different resource URIs. The servers are using Apache server software, which logs network traffic using Combined Log Format. A single log line contains information about the HTTP query:

```
127.0.0.1 - -
[01/January/2012:00:00:01 +0300]
"GET /resource.php?parameter1=value1
&parameter2=value2
HTTP/1.1"
200 2680
"http://www.address.com/webpage.html"
"Mozilla/5.0
(SymbianOS/9.2;...)"
```

The HTTP GET request part of the log line might contain information about SQL injections and other kinds of attacks. This request part is preprocessed using the methods described in Section II-A. Consequently, we get a binary matrix representing whether an n-gram is present in a specific log line or not. The resulting data points are then clustered into normal and anomalous clusters as described in Section II-B. Because the data set is unlabeled, the unsupervised labeling is performed for the whole data set. This is information is used for test result validation as shown in Figure 3.

B. Data analysis

The first data set for initial testing contains 4292 log lines. After preprocessing we find that there are 490 unique 2-grams in the data, resulting in 4292×490 sized feature matrix. Each datapoint now has a label (normal or anomalous) based on the clustering results. This information can be used to extract the rules. We select randomly 2000 data points for rule creation. The whole data set contains 2292 log lines that are not present during rule set learning phase. These remaining lines are our testing data set.

First, we discover that the used algorithm creates 6 rules, 2 for the normal traffic cluster and 4 for the anomalous one. After testing the rules with the whole dataset, all the data points except one match the correct rules. One anomalous data point is not covered by any rule. All of the normal traffic data points match one of the rules. In this case the system works with almost 100% accuracy, which means that the training data represents the testing data well enough.

The second data set contains 10935 log lines. In this data, 414 unique n-grams are found, resulting in a matrix of size 10935×414. After dimensionality reduction, the number of clusters k is determined using the average silhouette value, as described in Section II-B. Figure 4 shows that the data seems to form 4 clusters that are found using k-means algorithm. For rule set learning phase, 8000 data points are used. Other 2935 are used for traffic classification testing. Figure 5 shows all of the data points after dimensionality reduction and clustering used for unsupervised labeling step. As we can see from this visualization, cluster c_4 contains clearly more points than the others.

Rule extraction from the training set produces 15 rules describing 3 of the classes. One class is not featured in the training data and therefore no rules were generated for this class. The testing data set does not contain any samples belonging to class c_1. Out of the 493 data points of class c_3,

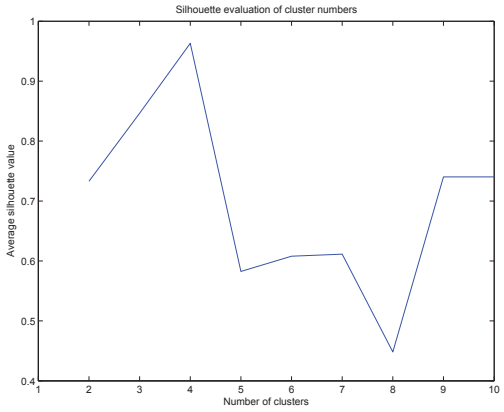

Fig. 4. Optimal cluster number for k-means.

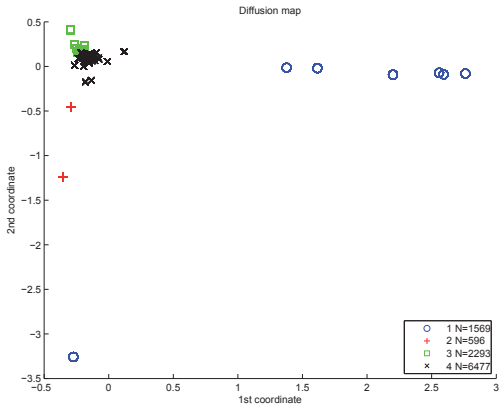

Fig. 5. Two-dimensional visualization of diffusion map of the whole data set.

the extracted rules successfully identify 349 (71%). Test data set contains 2742 data points of class c_4, out of which 1990 are found using the rules (73%). The reason these percentage figures are so low is that the training data differs from the testing data too much. However, the conjunctive rule extraction algorithm always covers the whole training data with 100% identification rate.

IV. Conclusion

Using modern data mining technology in network security context can become problematic when facing end-user needs. Even if the technology produces tangible results, the user rarely has understanding of the methodology. Therefore, this so-called black box system is not a desirable end goal. Simple conjunctive rules are easier to understand, and rule extraction from the complex data mining techniques might facilitate user acceptance. In this research, we have combined rule extraction methodology with diffusion map training framework in order to produce a rule-based network security system.

The main benefit of this framework is that the final output is a set of rules. No black box implementation is needed as the end result is a simple and easy to understand rule matching system. The training data may contain intrusions and anomalies, provided that the clustering step can differentiate them. In addition, rule matching is a fast operation compared to more complex algorithms.

The experimental data sets in this study are suitable for rule generation. The number of created rules is not too high for practical purposes and the accuracy with the first data set is high enough. Data points that do not match any rule could still be flagged as an anomaly in a practical intrusion detection system. The most important thing is to recognize normal traffic accurately. However, if new data points introduced after rule generation are very different from the training data set, the accuracy of classification using the rules might suffer considerably. Periodical rule updating will solve this issue. The second test data set demonstrates how important it is to have a training set that corresponds to the real situation as accurately as possible. If some types of data points are not featured in the rule generation phase, corresponding rules are not generated and these points will not be classified correctly. With proper training data the generated rules give much better accuracy. The created clustering may not represent reality but it is convenient while actual data labels are unknown. Another concern is overfitting of the rules, but the rules can be generalized to mitigate this problem.

The proposed framework is useful in situations where high-dimensional data sets need to be used as a basis for anomaly detection and quick classification. Such data sets are common nowadays in research environments as well as in industry, because collecting data is wide-spread. Our example case has been network security, which bears real benefits to anyone using modern communication networks. The provided tools are useful for network administrators who are trying to understand anomalous behavior in their networks.

Future topics include dynamic rule update as systems evolve, rule set optimization and using the rule set to filter real-time data sets. The modular structure of the framework enables these additions to be implemented conveniently. The applicability of the system to a wider network security context should also be tested, meaning cooperation with other security systems and components such as next-generation firewalls and other signature-based systems.

Acknowledgment

This research was supported by the Foundation of Nokia Corporation. Thanks are extended to Kilosoft Group Oy.

References

[1] K. Scarfone and P. Mell, "Guide to intrusion detection and prevention systems (idps)," *NIST Special Publication*, vol. 800, no. 2007, p. 94, 2007.

[2] M. A. Aydın, A. H. Zaim, and K. G. Ceylan, "A hybrid intrusion detection system design for computer network security," *Computers & Electrical Engineering*, vol. 35, no. 3, pp. 517–526, 2009.

[3] A. Lazarevic, V. Kumar, and J. Srivastava, "Intrusion detection: A survey," *Managing Cyber Threats*, pp. 19–78, 2005.

[4] F. Sabahi and A. Movaghar, "Intrusion detection: A survey," in *Systems and Networks Communications, 2008. ICSNC'08. 3rd International Conference on*. IEEE, 2008, pp. 23–26.

[5] M. Ramadas, S. Ostermann, and B. Tjaden, "Detecting anomalous network traffic with self-organizing maps," in *Recent Advances in Intrusion Detection*, G. Vigna, E. Jonsson, and C. Kruegel, Eds. Springer, 2003, pp. 36–54.

[6] Q. Tran, H. Duan, and X. Li, "One-class support vector machine for anomaly network traffic detection," *China Education and Research Network (CERNET), Tsinghua University, Main Building*, vol. 310, 2004.

[7] H. Om and A. Kundu, "A hybrid system for reducing the false alarm rate of anomaly intrusion detection system," in *Recent Advances in Information Technology (RAIT), 2012 1st International Conference on*, march 2012, pp. 131–136.

[8] R. Rangadurai Karthick, V. Hattiwale, and B. Ravindran, "Adaptive network intrusion detection system using a hybrid approach," in *Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on*, jan. 2012, pp. 1–7.

[9] L. Li, G. Zhang, J. Nie, Y. Niu, and A. Yao, "The application of genetic algorithm to intrusion detection in mp2p network," in *Advances in Swarm Intelligence*, ser. Lecture Notes in Computer Science, Y. Tan, Y. Shi, and Z. Ji, Eds. Springer Berlin Heidelberg, 2012, vol. 7331, pp. 390–397.

[10] M. Goyal and A. Aggarwal, "Composing signatures for misuse intrusion detection system using genetic algorithm in an offline environment," in *Advances in Computing and Information Technology*, ser. Advances in Intelligent Systems and Computing, N. Meghanathan, D. Nagamalai, and N. Chaki, Eds. Springer Berlin Heidelberg, 2012, vol. 176, pp. 151–157.

[11] A. Parashar, P. Saurabh, and B. Verma, "A novel approach for intrusion detection system using artificial immune system," in *Proceedings of All India Seminar on Biomedical Engineering 2012 (AISOBE 2012)*, ser. Lecture Notes in Bioengineering, V. Kumar and M. Bhatele, Eds. Springer India, 2013, pp. 221–229.

[12] D. Dave and S. Vashishtha, "Efficient intrusion detection with knn classification and ds theory," in *Proceedings of All India Seminar on Biomedical Engineering 2012 (AISOBE 2012)*, ser. Lecture Notes in Bioengineering, V. Kumar and M. Bhatele, Eds. Springer India, 2013, pp. 173–188.

[13] I. Brahmi, S. Yahia, H. Aouadi, and P. Poncelet, "Towards a multiagent-based distributed intrusion detection system using data mining approaches," in *Agents and Data Mining Interaction*, ser. Lecture Notes in Computer Science, L. Cao, A. Bazzan, A. Symeonidis, V. Gorodetsky, G. Weiss, and P. Yu, Eds. Springer Berlin Heidelberg, 2012, vol. 7103, pp. 173–194.

[14] T. Sipola, A. Juvonen, and J. Lehtonen, "Anomaly detection from network logs using diffusion maps," in *Engineering Applications of Neural Networks*, ser. IFIP Advances in Information and Communication Technology, L. Iliadis and C. Jayne, Eds. Springer Boston, 2011, vol. 363, pp. 172–181.

[15] ——, "Dimensionality reduction framework for detecting anomalies from network logs," *Engineering Intelligent Systems*, vol. 20, pp. 87–97, 2012.

[16] N. Ryman-Tubb and A. d'Avila Garcez, "SOAR – Sparse oracle-based adaptive rule extraction: Knowledge extraction from large-scale datasets to detect credit card fraud," in *Neural Networks (IJCNN), The 2010 International Joint Conference on*. IEEE, 2010, pp. 1–9.

[17] M. W. Craven and J. W. Shavlik, "Using sampling and queries to extract rules from trained neural networks," in *In Proceedings of the Eleventh International Conference on Machine Learning*. Morgan Kaufmann, 1994, pp. 37–45.

[18] A. d'Avila Garcez, K. Broda, and D. Gabbay, "Symbolic knowledge extraction from trained neural networks: A sound approach," *Artificial Intelligence*, vol. 125, no. 1, pp. 155–207, 2001.

[19] D. Martens, B. Baesens, and T. Van Gestel, "Decompositional rule extraction from support vector machines by active learning," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 21, no. 2, pp. 178–191, 2009.

[20] J. Hipp, U. Güntzer, and G. Nakhaeizadeh, "Algorithms for association rule mining–a general survey and comparison," *ACM SIGKDD Explorations Newsletter*, vol. 2, no. 1, pp. 58–64, 2000.

[21] M. Damashek, "Gauging similarity with n-grams: Language-independent categorization of text," *Science*, vol. 267, no. 5199, p. 843, 1995.

[22] R. Coifman, S. Lafon, A. Lee, M. Maggioni, B. Nadler, F. Warner, and S. Zucker, "Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, no. 21, pp. 7426–7431, 2005.

[23] R. R. Coifman and S. Lafon, "Diffusion maps," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 5–30, 2006.

[24] B. Nadler, S. Lafon, R. Coifman, and I. Kevrekidis, "Diffusion maps, spectral clustering and reaction coordinates of dynamical systems," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 113–127, 2006.

[25] R. Coifman, Y. Shkolnisky, F. Sigworth, and A. Singer, "Graph laplacian tomography from unknown random projections," *Image Processing, IEEE Transactions on*, vol. 17, no. 10, pp. 1891–1899, oct. 2008.

[26] P. Tan, M. Steinbach, and V. Kumar, "Cluster analysis: Basic concepts and algorithms," *Introduction to data mining*, pp. 487–568, 2006.

[27] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.

[28] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Prentice Hall., 1988.

[29] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, no. 0, pp. 53 – 65, 1987.

[30] H. Núñez, C. Angulo, and A. Català, "Rule extraction from support vector machines," in *In European Symposium on Artificial Neural Networks Proceedings*, 2002, pp. 107–112.

[31] N. Barakat and J. Diederich, "Learning-based rule-extraction from support vector machines," in *The 14th International Conference on Computer Theory and applications ICCTA'2004*, 2004.

[32] N. Barakat and A. P. Bradley, "Rule extraction from support vector machines: A review," *Neurocomputing*, vol. 74, no. 1–3, pp. 178–190, 2010.

PVI

AN EFFICIENT NETWORK LOG ANOMALY DETECTION SYSTEM USING RANDOM PROJECTION DIMENSIONALITY REDUCTION

by

Antti Juvonen and Timo Hämäläinen 2014

New Technologies, Mobility and Security (NTMS), 2014 6th International Conference on

An Efficient Network Log Anomaly Detection System using Random Projection Dimensionality Reduction

Antti Juvonen, Timo Hämäläinen
Department of Mathematical Information Technology
University of Jyväskylä
FI-40014 Jyväskylä, Finland
Email: {antti.k.a.juvonen, timo.hamalainen}@jyu.fi

***Abstract*—Network traffic is increasing all the time and network services are becoming more complex and vulnerable. To protect these networks, intrusion detection systems are used. Signature-based intrusion detection cannot find previously unknown attacks, which is why anomaly detection is needed. However, many new systems are slow and complicated. We propose a log anomaly detection framework which aims to facilitate quick anomaly detection and also provide visualizations of the network traffic structure. The system preprocesses network logs into a numerical data matrix, reduces the dimensionality of this matrix using random projection and uses Mahalanobis distance to find outliers and calculate an anomaly score for each data point. Log lines that are too different are flagged as anomalies. The system is tested with real-world network data, and actual intrusion attempts are found. In addition, visualizations are created to represent the structure of the network data. We also perform computational time evaluation to ensure the performance is feasible. The system is fast, finds real intrusion attempts and does not need clean training data.**

Keywords*—Intrusion detection, data mining, machine learning, random projection, mahalanobis distance.

I. Introduction

Web services have become more and more complicated and the amount of network traffic is increasing all the time. This makes ensuring good information security a challenge. In order to detect network attacks and improve security, *intrusion detection systems* (IDS) are used. These systems can generally be divided into two distinct categories: *signature-based* and *anomaly-based* systems [1].

Signature-based intrusion detection is still most commonly used. It uses predetermined attack rules to detect intrusive behavior. Network traffic or other actions are compared to these rules, and if there is a match an alarm is created. The benefits of this approach include fast operation and being able to distinguish different types of attacks based on the rules used to detect them. In addition, the number of false alarms is usually low. However, the attack signatures must be manually created. This means that the signatures can be one step behind attackers, and new unknown vulnerabilities can be exploited until a suitable rule is generated and the IDS rule set updated. Anomaly-based systems are based on a different principle. New incoming traffic or behavior is compared to the normal profile, and if an action deviates from the norm, it is flagged as an anomaly. Consequently, new and previously unknown intrusion attempts can be detected. The network profile can be updated periodically or in real-time, which means that the system adapts to changes in network traffic. On the other hand, some algorithms used in anomaly detection systems can be too slow for real-time detection. In addition, the number of false alarms can be unpractically high if the system is not configured properly. Many possible algorithms and methods can be found in Section II. It also possible to combine different detection principles into a hybrid IDS (HIDS) [2].

One big issue with anomaly detection systems is the efficiency and speed. If the amount of network traffic is high, it might be impossible to use complicated algorithms fast enough to detect intrusions before it's too late. Many advanced algorithms achieve a high detection rate but are too computationally complex for practical use. In addition, some intrusion detection frameworks can only do batch-analysis of the whole data or require labeled training data.

We propose an anomaly detection framework that deals with these problems. The system preprocesses web server log data and extracts numerical features from it, forming a feature matrix. Then, the dimensionality of the data is reduced using random projection methodology, and a visualization is also obtained to provide information to the network administrator. Subsequently, Mahalanobis distance is used to calculate an anomaly score for each data point. The data points (corresponding to log lines) that have a score higher than set threshold will be flagged as anomalies. The system is very fast and can function even in real-time. When new log lines are introduced, they can be visualized and the anomaly score calculated without starting the analysis from scratch, meaning that new data can be added dynamically. New data points can be added and older ones dropped from the whole dataset, so that the system adapts to changing network traffic over time.

II. Related research

Dimensionality reduction has been widely researched in the intrusion detection context. Perhaps the most well-known method is principal component analysis (PCA) [3], [4], [5]. It has been researched extensively in network anomaly detection [6], [7]. However, it has some problems, such as the fact that it cannot handle nonlinear data. It is also not as fast as random projection.

This is the authors' postprint version of the article. The original print version appeared as: A. Juvonen and T. Hämäläinen, "An Efficient Network Log Anomaly Detection System using Random Projection Dimensionality Reduction," in *New Technologies, Mobility and Security (NTMS), 2014 6th International Conference on*. IEEE 2014. Available online at http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6814006&isnumber=6813963

Surveys describing advances in the field of intrusion detection have been published [8], [9]. Many machine learning methods, such as self-organizing maps [10] and support vector machines [11] have been used to cluster data and detect anomalies in these systems. Various hybrid systems combining signature and anomaly-based detection have been used [2], [12]. A two-stage adaptive hybrid system for IP level intrusion detection has also been recently devised. A probabilistic classifier detects anomalies and a hidden Markov model narrows down attacker addresses [13]. Recently, genetic algorithms have been widely used in anomaly detection and misuse detection [14], [15]. Another recent development is using artificial immune systems (AIS) in intrusion detection [16].

The authors have already been involved in developing several network anomaly detection systems [17], [18], [19]. These papers mainly focus on diffusion map (DM) methodology for dimensionality reduction. The diffusion map serves the same purpose as random projection in this paper, and DM can be very efficient in finding anomalies and handling outliers in the data. In addition, nonlinear data is not a problem for DM. However, it's main problem is computational complexity, and this limits it's use in real-time anomaly detection. This paper focuses on random projection because of time constraints when analyzing large amounts of traffic. Random projection has not been extensively used in anomaly detection before.

Much of the research related to intrusion and anomaly detection use publicly available datasets, such as DARPA 1998 and DARPA 1999 [20] as well as KDD Cup 99 [21]. However, these datasets have many problems [22], [23] and therefore do not represent real network traffic accurately. We focus on real-world data collected from an actual network.

III. Methodology

In this section, the overall system framework and used methods are explained. Visualization of the whole system can be seen in Figure 1. The system consists of following phases:

- Data acquisition
- Preprocessing
- Feature extraction
- Random projection dimensionality reduction
- Mahalanobis distance score calculation
- Anomaly alerts based on threshold value

First the data must be collected from a network. In this study, Apache HTTP server access logs are used. Data format, preprocessing and feature extraction can vary depending on the dataset.

After acquiring and preprocessing the data, as well as extracting numerical feature matrix from the log files, the dimensionality of the matrix is reduced using random projection. Subsequently, the Mahalanobis distance for each data point from the whole dataset can be calculated. Finally, the data points with Mahalanobis distance higher than a specified threshold value are flagged as anomalies and can be inspected by the network administrator.

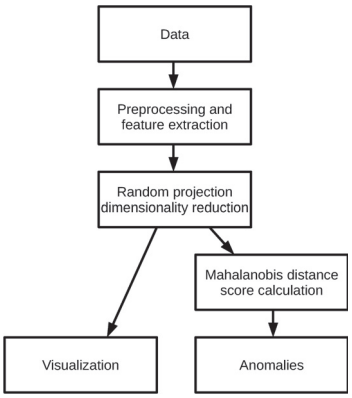

Fig. 1. Overall system framework.

Used data, methods and algorithms are described in more detail in the following subsections.

A. Data acquisition and preprocessing

We use the same network log database that has been used in our previous research [19]. The data comes from a real-life company web server. Different kinds of intrusion attempts and other abnormal log lines are included in the data. We examine a log file that is created in a web server using Apache server software. A single log line uses the following format:

```
127.0.0.1 - -
[01/January/2012:00:00:01 +0300]
"GET /resource.php?parameter1=value1
&parameter2=value2
HTTP/1.1"
200 2680
"http://www.address.com/webpage.html"
"Mozilla/5.0
(SymbianOS/9.2;...)"
```

This format is called Combined Log Format [24]. The used data is from the access logs of the server. These logs contain various information about the network traffic, such as timestamp, HTTP request and the amount of transferred bytes. The logs may contain different intrusion attempts, such as SQL injections, especially in the HTTP request part. For this analysis, the HTTP request string was analyzed and used.

After acquiring the logs, the data is ready for feature extraction. For this analysis, the character distribution is used. This simply means calculating the frequencies of individual characters in the data. These frequencies will form a feature matrix that can be used in the other steps of the analysis. Each row of the column corresponds to an individual log line, and each column corresponds to in individual character. Empty columns corresponding to characters not appearing in the dataset are omitted. This way we get a feature matrix X containing feature vectors $\boldsymbol{x} = (x_1, x_2, \ldots, x_d)$, where each symbol of a vector corresponds to a log line character frequency of one single character. There are d unique characters in the dataset, forming a d-dimensional feature matrix.

B. Random Projection

In random projection (RP), the goal is to project high-dimensional data into a lower-dimensional space using a random matrix [25]. The idea is based on Johnson-Lindenstrauss lemma [26]. It states that points can be projected to a randomly generated subspace and still the distances between points are approximately preserved.

Given the original data with d dimensions, the new subspace has k dimensions so that $k << d$. If the original data matrix is $X_{d\times N}$ and the randomly generated matrix is $R_{k\times d}$, the random projection of the data can be calculated using the following equation [25].

$$X^{RP}_{k\times N} = R_{k\times d} X_{d\times N}$$

As can be seen from the equation, the random projection method is computationally not very expensive even if the original data have a high number of dimensions. However, the generation and orthogonalization of the random matrix R can be complicated, but is not a problem in this case as explained below.

The most important phase of the method is the actual creation of the random matrix R. Basically, R should be orthogonal but unfortunately orthogonalization is computationally expensive. However, a useful result has been presented by Hecht-Nielsen [27]: *"There exists a much larger number of almost orthogonal than orthogonal directions in a high-dimensional space"*. Based on this result, we can assume that orthogonalization can be left out. The practical experimental results done in this paper also support this.

Instead of using Gaussian distributed variables, a much simpler probability distribution has been proposed by Achlioptas [28]:

$$r_{ij} = \sqrt{3} \times \begin{cases} +1 & \text{with probability} \frac{1}{6} \\ 0 & \text{with probability} \frac{2}{3} \\ -1 & \text{with probability} \frac{1}{6} \end{cases}$$

Computing the random matrix with this distribution is very efficient and easy to implement. It is possible to use random projection that is even more sparse. More generally speaking, the items in the random matrix can be calculated using the following probability distribution [29] :

$$r_{ij} = \sqrt{s} \times \begin{cases} +1 & \text{with probability} \frac{1}{2s} \\ 0 & \text{with probability} 1 - \frac{1}{s} \\ -1 & \text{with probability} \frac{1}{2s} \end{cases}$$

It is possible to choose s so that $s \gg 3$. This leads to *very sparse random projections* [29]. However, for this study we use $s = 3$, as proposed by Achlioptas [28].

C. Mahalanobis distance

The Mahalanobis distance [30] is a distance metric that is used for outlier detection in the proposed system. This distance metric takes into account the correlations of the data. The Mahalanobis distance metric is calculated for each individual data point, taking account the distance from the whole dataset. This creates basically an anomaly score. Setting a threshold for this score makes it possible to flag certain data points as anomalies.

If we have a data set X with an individual data vector being $\boldsymbol{x} = (x_1, x_2, \ldots, x_N)^T$, as well as mean $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_N)^T$ and covariance matrix S, the Mahalanobis distance score for each data point can be formally defined as follows [31]:

$$D_M = \sqrt{(\boldsymbol{x} - \boldsymbol{\mu})^T S^{-1} (\boldsymbol{x} - \boldsymbol{\mu})}$$

The outlier detection mechanism used in this paper can be changed, just like other components in the modular system. How the Mahalanobis distance works in practice in this system is described in Section IV.

D. Adding new data points

The methodology described previously is first performed on the whole available data set. However, when new traffic occurs in the network and therefore new data points are generated, the whole analysis does not have to be run from scratch. Preprocessing and character distribution are both trivial for the new log line. Random projection is performed with simple matrix multiplication with the same random matrix created previously. This way the new data point is projected into the same low-dimensional subspace as previous points. Finally, Mahalanobis distance is calculated just like for all the other points.

IV. Experimental results

Apache web server log data was acquired from a real-world company network and preprocessed as explained previously. The test data set that was received contains 1,244,025 lines, and the timespan is about one week. After preprocessing we find that 185 unique characters appear in the data, corresponding to 185 dimensions in the feature matrix.

The data are projected into a 2-dimensional subspace using random projection. Subsequently, Mahalanobis distance is calculated for each data point to form the anomaly score. These distances along with the chosen threshold value can be seen from Figure 2. The values are scaled between 0 and 1 in this figure. Some of the data points seem to be highly anomalous, while most of the points form a large normal cluster. Setting the threshold value higher will mean that only the most anomalous behavior is detected, setting it lower will mean that potentially more anomalies are found but the false alarm rate might increase as well.

Figure 3 shows the 2-dimensional RP visualization, with anomalies highlighted with red. Normal traffic is seen as a big cluster of points, and many queries are far away from the normal cluster, indicating that they are highly anomalous. Using the given threshold value, 278 log lines are flagged as anomalous, meaning that only 0.02% of the traffic is flagged. The other points (99.98%) represent normal traffic.

Because the used dataset is real network data, any prior information about possible intrusions is not available. This is

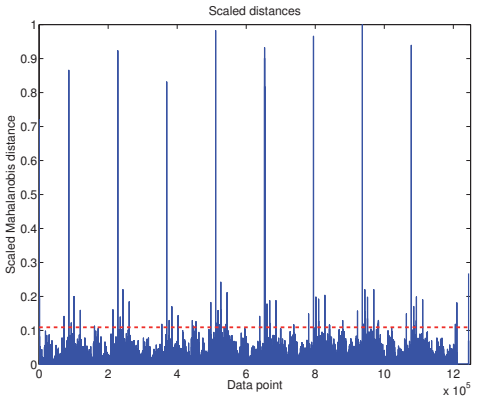

Fig. 2. Scaled Mahalanobis distances with threshold line visible.

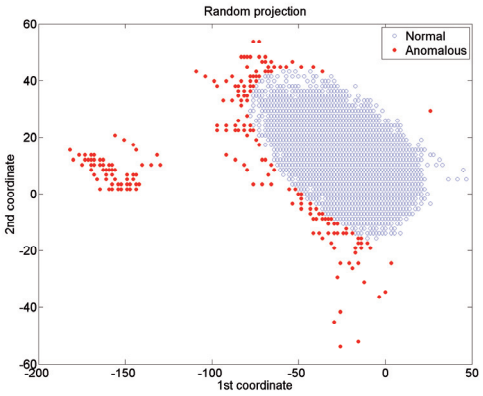

Fig. 3. Random projection visualization of the dataset.

the case in practical situations without artificially generated training data. The 278 anomalies are manually inspected to check if something intrusive is found. Upon inspection it is revealed that only 8 of these loglines are normal and non-intrusive, which equals 2.8% of all the alarms. This suggests that lowering the threshold might reveal more anomalies, even though they are potentially similar to the ones that were already found. The anomalies include GoogleBot scans, as well as several security scans using Nmap, DirBuster and Brutus AET password cracker. The scans mainly focus on finding vulnerabilities in phpMyAdmin software. These intrusion attempts are not very severe for updated systems, and therefore they do not pose a risk at this time. Still, these loglines deviate from normal traffic in a clear way. Any similar scan attempt should be easy to find using the proposed system.

Analyzing the whole data set is very fast. The Python implementation of preprocessing for the whole dataset takes the most time (minutes), while subsequent analysis phases done in Matlab are completed almost instantly. As a comparison to widely used PCA (mentioned in Section II), we calculated

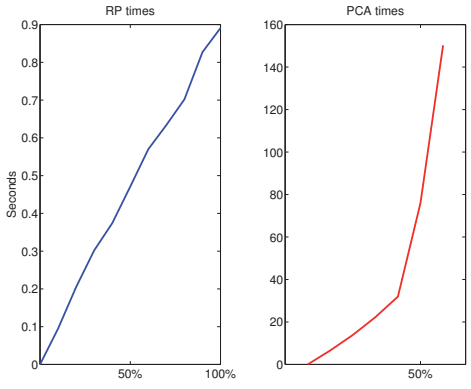

Fig. 4. Computational times for RP and PCA.

some computing times for different sized subsets of the data. This comparison can be seen in Figure 4. The purpose was to compare the time taken for dimensionality reduction, as well as the effect of data size in computational times. It is apparent that RP performs much better, the whole analysis taking less than a second. In addition, the time increase appears to be quite linear, meaning RP has good scalability. PCA analysis is performed only up to 60% of the data set, because the analysis time increases rapidly and becomes unpractically long before even analyzing 100% of the dataset. All of the runs were performed 5 times, and the times were averaged over these 5 runs. It must be noted that Mahalanobis distance calculation is not included in these performance evaluations, because the time taken would have been the same for both RP and PCA matrices.

V. Conclusion

Overall security in a network could be enhanced by using anomaly detection together with traditional signature-based intrusion detection systems. However, anomaly detection systems often use complicated and slow algorithms, which ensures high detection rate but impractically low speed. We propose a framework that can be used to analyze and visualize logs quickly, as well as find anomalous network traffic. This system is not designed to work as the only security measure, but rather as an addition to existing systems.

The system's main advantage is the simplicity and speed. It can easily analyze huge log files with relatively low-powered hardware. In addition, when new network traffic occurs, there is no need to perform the entire analysis from scratch. New data points can be added dynamically and old data can be dropped, so that the system adapts automatically to changing network profile. Also, clean traffic data (traffic that does not contain intrusions) are not needed. However, even though the system was able to generate value by finding intrusion attempts from actual real-world log files, more experiments with new data are needed to ensure that the detection rate is acceptable.

For future research, any component of the framework can be changed. Therefore, different dimensionality reduction and

outlier detection methods could be used. In addition, to make the system more general and avoid overfitting, random projection and subsequent anomaly detection could be performed several times for the same data. After this, each data point would be either flagged as normal or anomalous several times by the system. This is because random projection by definition has a certain random element to it, and might sometimes give unwanted results for individual data points. If the point is flagged as anomalous more times than normal, it will be treated as an anomaly. Bootstrapping is another technique that could be combined with this, making the system even less prone to overfitting. This way, if one random matrix gives unwanted results, it does not lessen the performance of the whole system. These features would make the system more automatic and therefore easier to use for network administrators who are not data mining experts.

ACKNOWLEDGMENT

This research was partially supported by the Nokia Foundation. Thanks are extended to Kilosoft Oy.

REFERENCES

[1] K. Scarfone and P. Mell, "Guide to intrusion detection and prevention systems (idps)," *NIST Special Publication*, vol. 800, no. 2007, p. 94, 2007.

[2] M. A. Aydın, A. H. Zaim, and K. G. Ceylan, "A hybrid intrusion detection system design for computer network security," *Computers & Electrical Engineering*, vol. 35, no. 3, pp. 517–526, 2009.

[3] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[4] H. Hotelling, "Analysis of a complex of statistical variables into principal components." *Journal of educational psychology*, vol. 24, no. 6, p. 417, 1933.

[5] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[6] H. Ringberg, A. Soule, J. Rexford, and C. Diot, "Sensitivity of PCA for traffic anomaly detection," *ACM SIGMETRICS Performance Evaluation Review*, vol. 35, no. 1, pp. 109–120, 2007.

[7] C. Callegari, L. Gazzarrini, S. Giordano, M. Pagano, and T. Pepe, "A novel PCA-based network anomaly detection," in *Communications (ICC), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1–5.

[8] A. Lazarevic, V. Kumar, and J. Srivastava, "Intrusion detection: A survey," *Managing Cyber Threats*, pp. 19–78, 2005.

[9] F. Sabahi and A. Movaghar, "Intrusion detection: A survey," in *Systems and Networks Communications, 2008. ICSNC'08. 3rd International Conference on*. IEEE, 2008, pp. 23–26.

[10] M. Ramadas, S. Ostermann, and B. Tjaden, "Detecting anomalous network traffic with self-organizing maps," in *Recent Advances in Intrusion Detection*, G. Vigna, E. Jonsson, and C. Kruegel, Eds. Springer, 2003, pp. 36–54.

[11] Q. Tran, H. Duan, and X. Li, "One-class support vector machine for anomaly network traffic detection," *China Education and Research Network (CERNET), Tsinghua University, Main Building*, vol. 310, 2004.

[12] H. Om and A. Kundu, "A hybrid system for reducing the false alarm rate of anomaly intrusion detection system," in *Recent Advances in Information Technology (RAIT), 2012 1st International Conference on*, march 2012, pp. 131–136.

[13] R. Rangadurai Karthick, V. Hattiwale, and B. Ravindran, "Adaptive network intrusion detection system using a hybrid approach," in *Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on*, jan. 2012, pp. 1–7.

[14] L. Li, G. Zhang, J. Nie, Y. Niu, and A. Yao, "The application of genetic algorithm to intrusion detection in mp2p network," in *Advances in Swarm Intelligence*, ser. Lecture Notes in Computer Science, Y. Tan, Y. Shi, and Z. Ji, Eds. Springer Berlin Heidelberg, 2012, vol. 7331, pp. 390–397.

[15] M. Goyal and A. Aggarwal, "Composing signatures for misuse intrusion detection system using genetic algorithm in an offline environment," in *Advances in Computing and Information Technology*, ser. Advances in Intelligent Systems and Computing, N. Meghanathan, D. Nagamalai, and N. Chaki, Eds. Springer Berlin Heidelberg, 2012, vol. 176, pp. 151–157.

[16] A. Parashar, P. Saurabh, and B. Verma, "A novel approach for intrusion detection system using artificial immune system," in *Proceedings of All India Seminar on Biomedical Engineering 2012 (AISOBE 2012)*, ser. Lecture Notes in Bioengineering, V. Kumar and M. Bhatele, Eds. Springer India, 2013, pp. 221–229.

[17] T. Sipola, A. Juvonen, and J. Lehtonen, "Anomaly detection from network logs using diffusion maps," in *Engineering Applications of Neural Networks*. Springer, 2011, pp. 172–181.

[18] ——, "Dimensionality reduction framework for detecting anomalies from network logs," *Engineering Intelligent Systems*, 2012.

[19] A. Juvonen and T. Sipola, "Adaptive framework for network traffic classification using dimensionality reduction and clustering," in *Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2012 4th International Congress on*. IEEE, 2012, pp. 274–279.

[20] (2013, Aug.) Mit lincoln laboratory: Communication systems and cyber security: Cyber systems and technology: Darpa intrusion detection evaluation. [Online]. Available: http://www.ll.mit.edu/mission/communications/cyber/CSTcorpora/ideval/data/

[21] (2013, Aug.) Kdd cup 1999 data. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[22] 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 Symposium on Computational Intelligence for Security and Defence Applications 2009*, 2009.

[23] J. McHugh, "Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory," *ACM transactions on Information and system Security*, vol. 3, no. 4, pp. 262–294, 2000.

[24] (2013, Aug.) Log files - apache http server. [Online]. Available: http://httpd.apache.org/docs/2.4/logs.html

[25] E. Bingham and H. Mannila, "Random projection in dimensionality reduction: applications to image and text data," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2001, pp. 245–250.

[26] W. Johnson and J. Lindenstrauss, "Extensions of lipschitz mappings into a hilbert space," *Contemporary mathematics*, vol. 26, no. 189-206, pp. 1–1, 1984.

[27] R. Hecht-Nielsen, "Context vectors: general purpose approximate meaning representations self-organized from raw data," *Computational intelligence: Imitating life*, pp. 43–56, 1994.

[28] D. Achlioptas, "Database-friendly random projections," in *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 2001, pp. 274–281.

[29] P. Li, T. Hastie, and K. Church, "Very sparse random projections," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2006, pp. 287–296.

[30] P. C. Mahalanobis, "On the generalized distance in statistics," *Proceedings of the National Institute of Sciences (Calcutta)*, vol. 2, pp. 49–55, 1936.

[31] R. De Maesschalck, D. Jouan-Rimbaud, and D. L. Massart, "The mahalanobis distance," *Chemometrics and Intelligent Laboratory Systems*, vol. 50, no. 1, pp. 1–18, 2000.

PVII

ONLINE ANOMALY DETECTION USING DIMENSIONALITY REDUCTION TECHNIQUES FOR HTTP LOG ANALYSIS

by

Antti Juvonen, Tuomo Sipola and Timo Hämäläinen 2014

Submitted to Computer Networks, Elsevier