Intelligent Zero-Day Intrusion Detection Framework for Internet of Things

Ansam Khraisat

This thesis is submitted for

For the degree of PhD in the

School of Science, Engineering and Information Technology

Federation University,

Mt Helen, Ballarat

Australia

2020

Dedication

To the memory of my father, Ma'in Khraisat.

To my husband Dr Ammar Alazab and my children.

Your love and support have given me the biggest strength.

Acknowledgment

I would like to extend special thanks to my supervisors, family and friends who supported me during my PhD study. Importantly, I am indebted to my husband, Dr. Ammar Alazab who is an academic in ICT field. I thank him for continuous support during difficult times. I would like to thank my children (Mohammad, Sanad and Zaid) for being very patient during the busy and stressful times. Without my lovely family, I couldn't have completed my studies. Special thanks to my wonderful mum who has provided me with all the support and encouragement. Also, I would like to thank my siblings; Ahamd, Mohammad, Tharaa and Mais. A special thanks to my brother-in-law Dr. Mamoun Alazab for being supportive, as well as for his valuable advice and encouragement.

Very special recognition is due to my principle Supervisor, Professor Iqbal Gondal. I appreciate his wide skill in many areas. I would like to thank him for his patience and efforts, and for his encouragement, support and motivation throughout my studies.

Most of all, I would like to thank my supervisors, Associate Professor Peter Vamplew and Professor Joarder Kamruzzaman, for their valuable advice, comments, discussions, guidance, patience, encouragement, unconditional support, respect and belief in me. They always motivated me to do hard work. I have learned a lot from them. I have been fortunate to have them during my studies.

I would like to acknowledge that my study was supported by an Australian Government Research Training Program (RTP) Fee-Offset Scholarship and RPF scholarship through Federation University Australia.

Finally, I would like to thank Federation University Australia for this opportunity. Also, I would like to thank my colleagues at ICSL for providing a friendly atmosphere and being wonderful friends throughout the last three years.

List of Publications

1. **Khraisat, A**.; Gondal, I.; Vamplew, P.; Kamruzzaman, J.; Alazab, A., "Hybrid Intrusion Detection System Based on the Stacking Ensemble of C5 Decision Tree Classifier and One Class Support Vector Machine", Electronics Journal, Impact factor: 1.754, 2020

2. **A. Khraisat,** I. Gondal, and P. Vamplew, "An anomaly intrusion detection system using C5 decision tree classifier," in Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 149-155: Springer, 2018

3. **A. Khraisat,** I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," Cybersecurity Journal by Springer, vol. 2, no. 1, p. 20, impact factor: 1.89, 2019.

4. **A. Khraisat**, I. Gondal, P. Vamplew, and J. Kamruzzaman, " A Low-Level Intrusion Detection System Based on Hardware Performance Counters," Malware Analysis using Artificial Intelligence and Deep Learning, Springer, 2020 (Book chapter: Accepted)

5. **A. Khraisat**, I. Gondal, P. Vamplew, J. Kamruzzaman, and A. Alazab, "A Novel Ensemble of Hybrid Intrusion Detection System for Detecting Internet of Things Attacks," Electronics Journal, Impact factor: 1.754, 2020

Previously published work

6. Alazab, A., & **A. Khraisat**, A. (2016). New Strategy for Mitigating of SQL Injection Attack. International Journal of Computer Applications, 154(11), 2016

7. A.Alazab, **A. Khraisat**, A., Abawajy, J. H., & Hobbs, M., "Using Response Action with Intelligent Intrusion Detection and Prevention System against Web Application Malware, Information Management and Computer Security, April, 2014 (ERA C Rank), 2014

8. A.Alazab, **A. Khraisat**, A., Abawajy, J. H., & Hobbs, M. (2013), "Multi stage Rules for Intrusion Detection System", International Journal of Information Security and Privacy (IJISP), IGI, July, 2013 (ERA C Rank), 2013

9. **A. Khraisat**, A. Alazab, A.Abawajy, J. H., & Hobbs, M. (2013), “Trends in Crime toolkits Developments”, In A. Amine, O. Mohamed and B. Benatallah (Eds) Network Security Technologies: Design and Applications, IGI, June, 2013.

10. A.Alazab, **A. Khraisat**, A., Abawajy, J. H., & Hobbs, M. (2013), “Crime Toolkits: The current threats to web applications”, Journal of Information Privacy and Security, 9(2), 21-39, 2013

11. A. Alazab, M. Hobbs, J. Abawajy, and **A. Khraisat**, "Developing an Intelligent Intrusion Detection and Prevention System against Web Application Malware," in Advances in Security of Information and Communication Networks. vol. 381, A. Awad, A. Hassanien, and K. Baba, Eds., ed: Springer Berlin Heidelberg, , pp. 177-184, 2013

12. A.Alazab, A., Abawajy, J. H., & Hobbs, M. (2013) and **A. Khraisat**, in press “Detection and Protection of SQL injection”, Journal of Information Privacy and Security (JIPS), 2013.

13. A. Alazab, A.Abawajy, J. H., & **A. Khraisat**, (2013),” Crime Toolkits: The Productisation of Cybercrime”, Proceedings of the 10th IEEE Conference on Trust, Security and Privacy in Computing and Communications (TRUSTCOM), (ERA A Rank), 2013

Abstract

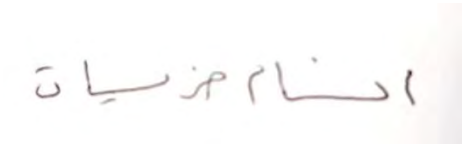
Zero-day intrusion detection system faces serious challenges as hundreds of thousands of new instances of malware are being created every day to cause harm or damage to the computer system. Cyber-attacks are becoming more sophisticated, leading to challenges in intrusion detection. There are many Intrusion Detection Systems (IDSs), which are proposed to identify abnormal activities, but most of these IDSs produce a large number of false positives and low detection accuracy. Hence, a significant quantity of false positives could generate a high-level of alerts in a short period of time as the normal activities are classified as intrusion activities.

This thesis proposes a novel framework of hybrid intrusion detection system that integrates the Signature Intrusion Detection System (SIDS) with the Anomaly Intrusion Detection System (AIDS) to detect zero-day attacks with high accuracy. SIDS has been used to identify previously known intrusions, and AIDS has been applied to detect unknown zero-day intrusions. The goal of this research is to combine the strengths of each technique toward the development of a hybrid framework for the efficient intrusion detection system. A number of performance measures including accuracy, F-measure and area under ROC curve have been used to evaluate the efficacy of our proposed models and to compare and contrast with existing approaches. Extensive simulation results conducted in this thesis show that the proposed framework is capable of yielding excellent detection performance when tested with a number of widely used benchmark datasets in the intrusion detection system domain. Experiments show that the proposed hybrid IDS provides higher detection rate and lower false-positive rate in detecting intrusions as compared to the SIDS and AIDS techniques individually.

Declaration

This thesis contains no material which has been accepted for the award of any other degree or diploma at any university or equivalent institution and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due references are made in the text of the thesis.

Ansam Khraisat



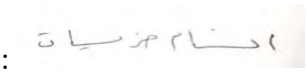
General Declaration – Published works

Four original papers (published) were part of this research in peer-reviewed ranked conferences. I was fully responsible for developing and writing all the papers in the thesis under the supervision of Prof Iqbal Gondal, A/Prof Peter Vamplew and Prof Joarder Kamruzzman.

Following table gives the level of my contribution for chapters 2, 3, 4, 5 and 6:

Thesis Chapter	Publication Title	Publication Status	Contribution	Contribution %
2	Survey of intrusion detection systems: techniques, datasets and challenges	Published	Developed analytical model and experimental setup. I wrote the initial draft and incorporated the suggestions & recommendations from the supervisors to prepare the final draft	80%
3	An Anomaly Intrusion Detection System Using C5 Decision Tree Classifier	Published	Developed analytical model and experimental setup. I wrote the initial draft and incorporated the suggestions & recommendations from the supervisors to prepare the final draft	85%
4	Hybrid Intrusion Detection System Based on the Stacking Ensemble of C5 Decision Tree Classifier and One Class Support Vector Machine per 3 title	Published	Developed analytical model and experimental setup. I wrote the initial draft and incorporated the suggestions & recommendations from the supervisors to prepare the final draft	85%
5	A Low-Level Intrusion Detection System Based on Hardware Performance Counters	Accepted	Developed analytical model and experimental setup. I wrote the initial draft and incorporated the suggestions & recommendations from the supervisors to prepare the final draft	85%

6	A novel ensemble of design hybrid intrusion detection system for detecting internet of things attacks	Published	Developed analytical model and experimental setup. I wrote the initial draft and incorporated the suggestions & recommendations from the supervisors to prepare the final draft	85%

Student Signature:  Date:6/2/2020

The undersigned hereby certify that the above declaration correctly reflects the nature and context of the student and co-authors contribution to this work.

Principal Supervisor Signature: Date:6/6/2020

Table of Contents

List of Figures

List of Tables

Abbreviations

ACSC Australian Cyber Security Centre

ADFA Australian Defense Force Academy

ANN Artificial Neural Network

API Application Program Interface

CGI Common Gateway Interface

CPCS Cyber-Physical Control System

CR Classification Rate

DoS Denial-of-Service

EM Expectation Maximization

FP False Positive

GA Genetic algorithms

HIDS Host-based Intrusion Detection System

HIDS Hybrid Intrusion Detection System

HMM Hidden Markov Model

HPCs Hardware Performance Counters

HTTPS HyperText Transfer Protocol Secure

ICMP Internet Control Message Protocol

IoT Internet of Things

KNN K-Nearest Neighbors

LAN	Local Area Network
MIB	Management Information Base
NB	Naïve Bayes
NIDS	Network Intrusion Detection System
R2L	Remote-to-Local
RF	Random Forest
SLFN	Single Hidden Layer Feed-forward Neural Network
SNMP	Simple Network Management Protocol
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
U2R	User to Root
UDP	User Datagram Protocol

Chapter 1 : Introduction

1.1 Background

With the birth of the Internet, the ability to connect computers globally also resulted in the growth of computer malicious software (malware). Nowadays the Internet is essential for daily life activities such as education, travel, business, entertainment, industry, government, defence, law enforcement, but malicious software can pose serious security threats to computer systems (Buczak & Guven, 2016). Cybercriminals are targeting online users with the use of sophisticated techniques in attacking their victims. Cybercriminals continuously innovate to evade detection, and therefore, it is a pressing need to develop effective cyberattack detection technology, which can efficiently detect the zero-day malware attacks. Zero-day attacks are

new types of attacks for which attack mitigation solutions have not been developed yet. During the zero-day time, malware can cause serious damage as demonstrated in many past attacks that spread globally, like the WannaCry attack in 2017 (Ehrenfeld, 2017).

Malware is a wide-ranging term which defines any type of malicious software. However, several definitions have been proposed to describe malware. McGraw and Morrisett describe the malware term as “a code included, changed, or erased from a product framework that causes mischief or subverting to the framework" (Idika & Mathur, 2007). Kramer and Bradfield depict malware as "programming that assaults other programming and damage it” (Kramer et al., 2010).

For the research presented in this thesis, malware is any program or code harmful to the information system to cause destruction, modify contents or access data without the proprietor's authorization. There are various kinds of malware, such as viruses, worms, trojan horses, spyware, rootkits and ransomware (Burguera, Zurutuza, & Nadjm-Tehrani, 2011). The research project in this thesis focus on zero-day malware. Numerous studies have reported the losses and the damages caused by such malware on large scales.

An intrusion could be described as a sequence of activities intended to compromise the security of a computer system. It tries to evade security detections of computer systems and threatens the availability, integrity, or confidentiality of computer resources. Intrusions are utilised by attackers to steal information from the computer system.

Intrusion detection is the procedure of distinguishing client practices that may likely alter an information system framework from an ensured state to an unprotected state. The malware detection frameworks could be equipment or programming frameworks that distinguish

suspicious practices on systems (Stakhanova, Basu, & Wong, 2007). However, the capability of current Intrusion Detection Systems (IDS) to detect up-to-date attacks is insufficient (Fan, Ye, & Chen, 2016).

IDSs generally adopt two approaches: signature-based detection and anomaly-based detection. Signature intrusion detection systems (SIDS) are based on pattern matching techniques to discover well-known intrusions; these are also known as Knowledge-based Detection or Misuse Detection. This term is borrowed from anti-virus software, which uses the detected patterns of the attacks as signatures for future detections of similar attacks.

In other words, when an attack matches with an instance of an abnormal activity from the database, an alert is raised. SIDS usually shows good accuracy for previously known intrusions. However, SIDS normally has difficulty in identifying zero-day malware attacks as the attack profile of new attacks may not be present in the current database, emphasizing the need to keep the attack profile database up-to-date. To ensure that zero-day attacks can be identified successfully, Anomaly Intrusion Detection Systems (AIDS) are used. AIDS can be effectively used to classify anomalous behaviours from normal behaviours, which would require the use of binary classifiers.

AIDSs and SIDSs are very similar in their characteristics (Stavroulakis & Stamp, 2010), as both can identify intrusions but their hybrid use can provide wide-ranging coverage against both known and zero-day attacks. Machine learning technique can be applied for both AIDS and SIDS to improve the detection accuracy. A machine learning technique's objective is to create IDS that increases the accuracy and relies on prior outcomes for training. IDS developers have been making use of machine learning methods to detect intrusions with mixed results.

AIDS methods can be categorized into three main groups (Lazarevic, Kumar, & Srivastava, 2005): Statistical based, knowledge-based, and machine learning-based. The statistical-based involves collecting and examining every data record in a set of items. Conversely, knowledge-based tries to identify the requested actions from existing system-data such as protocol specifications and network traffic instances. Classification of three subclasses (statistics-based, Knowledge-based and Machine Learning-based) with an in-depth perspective on their characteristics have been discussed by (Lazarevic et al., 2005).

The key advantage of an anomaly based intrusion detection system AIDS is the ability to identify new intrusion activities as the abnormal user profiles do not rely on a signature database. Furthermore, AIDS has several advantages: First, the internal threat can be identified. For example, if a cybercriminal exploiting a victim's customer account conducts activities that are outside the normal user's activates, it causes an alert. Second, the zero-day attack can be identified.

1.2 Research Questions

There has been a lot of research to improve the performance of IDSs, but many aspects remain to be investigated to improve the performance with the help of machine learning techniques. IDSs have to be more accurate, with the capability to detect a variety of intrusions with fewer false alarms. The majority of current IDSs are SIDS based, as these are highly efficient in identifying known intrusions. However, traditional IDSs are not effective in identifying zero-day attacks. For this reason, research in this thesis has proposed development of an intrusion detection system for detecting both known and zero-day attacks.

Most of the intrusion detection systems suffer a common problem. They produce a high number of false alerts and a huge number of false positives. To achieve this overall objective, the following research questions have been formulated in this thesis to detect the current and emerging threats to information systems:

1. Could anomaly based detection using network statistical features can be applied to detect and classify intrusions; and detect the methods that cybercriminals may use to avoid the detection provided by IDS, and how we can reduce false negatives and false positives alarms by examining different machine learning techniques?

2. Can we improve the performance of the ensemble of machine learning models e.g. hybrid of SIDS and AIDS more than the standard standalone machine learning algorithms to minimize both the number of false negatives and false positives alarm?

3. Could a combination of SIDS and AIDS detection using Hardware Performance Counters (HPCs) as features be used to efficiently detect Zero day intrusions; and how HPCs features could be selected to improve the intrusion detection rate?

4. How a combination of SIDS and AIDS could be used to detect Internet of Things (IoT) attacks without any previous knowledge?

1.3 Thesis Aim

This thesis presents a framework to identify both the well-known intrusion and the zero-day attacks with a high level of detection accuracy and low rates of false-alarm. A novel framework

is proposed for intelligent IDSs that overcomes the present weaknesses of traditional IDSs. This framework is based on the ensemble of classifiers.

1.4 Research Methodology

For the first research question (Q1), several intrusion detection techniques have been investigated to identify abnormal activities. However, most of these techniques provide lower performance in terms of high number of false alarms and an enormous number of false positives. Hence, generating a high number of false positives could create a high number of alerts in a short period of time as normal activities are classified as intrusion activities. Our approach aims to identify intrusions activities with improved accuracy of detection and decrease the false-alarm rates. One way to improve the accuracy of a classification system is to use various classifiers and combine their results. Ensembling several data mining techniques could minimize both the number of false negatives and false positives. In this thesis, we have demonstrated that ensemble-classifiers have higher efficiency for intrusion detection with the use of most common evaluation techniques and different datasets.

To address the second research question (Q2), a Hybrid IDS framework has been developed which combines SIDS and AIDS to form a hybrid IDS to detect both unknown and known attack. In our proposed framework, AIDS has been used to identify novel intrusions, while SIDS has been used to identify previously known intrusions.

To address the third research question (Q3), we explore which malware functionality and attack techniques impact the HPCs; and how HPC related features can be used to improve the detection rate by picking the top HPC features that have the greatest contribution. Our

developed Hybrid Intrusion Detection System (HIDS) has successfully detected the anomalies caused by malware exploits in attacked programs at operating systems' level.

To address the fourth question (Q4), a feature selection based on information gain has been developed. AIDS has been applied in IoT Intrusion Detection System, meaning machine learning has been employed in AIDS to detect the zero-day and complex attacks on IoT without any previous knowledge. This can be done by extracting useful features. Next, those features are integrated and then are used in our Hybrid approach to detect zero-day malware on IoT to create an effective ensemble architecture to improve the accuracy for the detection, decrease false alarm rate and storage and computational capability of IoT.

1.5 Contributions

The contributions of this thesis are as follows:

- A comprehensive survey on the intrusion detection systems is provided. We have presented, in detail, the intrusion detection system methodologies, types, and technologies with their advantages and limitations; and pinpointed the shortcomings of existing research in IDS. A Journal paper has been published on this survey work: Khraisat, A, Iqbal Gondal, Peter Vamplew, and Joarder Kamruzzaman. "Survey of intrusion detection systems: techniques, datasets and challenges." Cybersecurity Journal 2, 20 (2019).
- Anomaly intrusion detection system using C5 decision tree classifier is proposed to identify normal and anomalous activities accurately and reduce false-alarm rates. We compared C5 classifier with other classifiers and examined the efficiency of C5 published as a conference paper: A. Khraisat, I.

Gondal, and P. Vamplew, "An anomaly intrusion detection system using C5 decision tree classifier," in Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 149-155: Springer, 2018.

- A novel framework is proposed to build an intelligent IDS that overcomes the weaknesses of current IDSs. Our system integrates SIDS and AIDS to develop an efficient hybrid IDS in order to detect both unknown and known attacks. Our proposed framework is assessed using the two datasets: NSL-KDD and the Australian Defence Force Academy (ADFA) datasets. A Journal paper has been published: Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J.; Alazab, A, "Hybrid Intrusion Detection System Based on the Stacking Ensemble of C5 Decision Tree Classifier and One Class Support Vector Machine", Electronics Journal, 9, 173. 2020
- We have shown how some malware behaviour can affect HPCs in the computers and how HPC features can be used to identify malware attacks at the hardware level. Also, we have shown that most relevant HPC features can increase the accuracy and reduce false alarm rates for cyber detection for a variety of different attack types.
- We have presented a design of a low-level Intrusion detection system (IDS) based on HPCs. Our proposed approach can identify abnormalities caused by malware exploits on attacked programs. This intrusion detection system uses HPC features which are identified as normal activities. It then observes and compares the activities of the new data with normal behavior profiles to identify attacks. This chapter has been ACCEPTED on 5/1/2020 to be included in the

upcoming book entitled “Malware Analysis using Artificial Intelligence and Deep Learning”, which will be published as: A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, " A Low-Level Intrusion Detection System Based on Hardware Performance Counters," Malware Analysis using Artificial Intelligence and Deep Learning, Springer, 2020 (Accepted)

- IoT intrusion detection system proposed using the Hybrid Intrusion Detection System. Novel framework has been applied on an IoT dataset, and results have shown that it is capable of protecting IoT infrastructure and detecting any malicious activities A Journal paper has been published: A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, and A. Alazab, "A Novel Ensemble of Hybrid Intrusion Detection System for Detecting Internet of Things Attacks," vol. 8, no. 11, p. 1210, 2019.

1.6 Outline of the Thesis

This section presents an outline of the thesis. As this thesis is by publication, each contribution chapter is based on a research paper.

Chapter 2 This chapter discusses IDS approaches present in the literature that has been used for intrusion detection methods to tackle computer security threats. It is shown that the detection systems can be categorised into Signature-based Intrusion Detection System and Anomaly-based Intrusion Detection System. A number of AIDS systems have also been applied in the Network Intrusion Detection System (NIDS) and Host Intrusion Detection System (HIDS) to improve the detection performance with the use of machine learning, knowledge-based and statistical schemes. A comprehensive review, research challenges and

taxonomy of modern Intrusion Detection Systems (IDS) are presented in this chapter. Also, this chapter discusses evasion techniques that can be used by the attackers to avoid detection and future research directions in the area of IDS. This chapter is based on survey paper and was published in Security Journal by Springer (Khraisat, Gondal, Vamplew, & Kamruzzaman, 2019).

Chapter 3 We have studied and observed different machine learning techniques to minimize both false negatives and false positives alarms. We have examined the effectiveness of the C5 decision tree classifier and compared it with other classifiers. Our results have shown, with the use of C5 classifier, false negatives alerts are reduced and the accuracy is increased significantly on the intrusion detection system. This chapter is based on a conference paper (Khraisat, Gondal, & Vamplew, 2018).

Chapter 4 A hybrid IDS (HIDS) is proposed by combining the C5 decision tree classifier and One-Class Support Vector Machine. HIDS combines the advantages of both Signature Intrusion Detection Systems and Anomaly-based Intrusion Detection System. The objective of this framework is to classify both the well-known attacks and zero-day attacks resulting in low false-alarm rates and high detection accuracy. The proposed HIDS is assessed using the NSL-KDD and the Australian Defence Force Academy (ADFA) datasets and is published as a Journal paper (Khraisat, Gondal, Vamplew, Kamruzzaman, & Alazab, 2020)

Chapter 5 Proposed framework's performance is evaluated by profiling normal and malicious activities based on Hardware Performance Counters with the use of machine learning

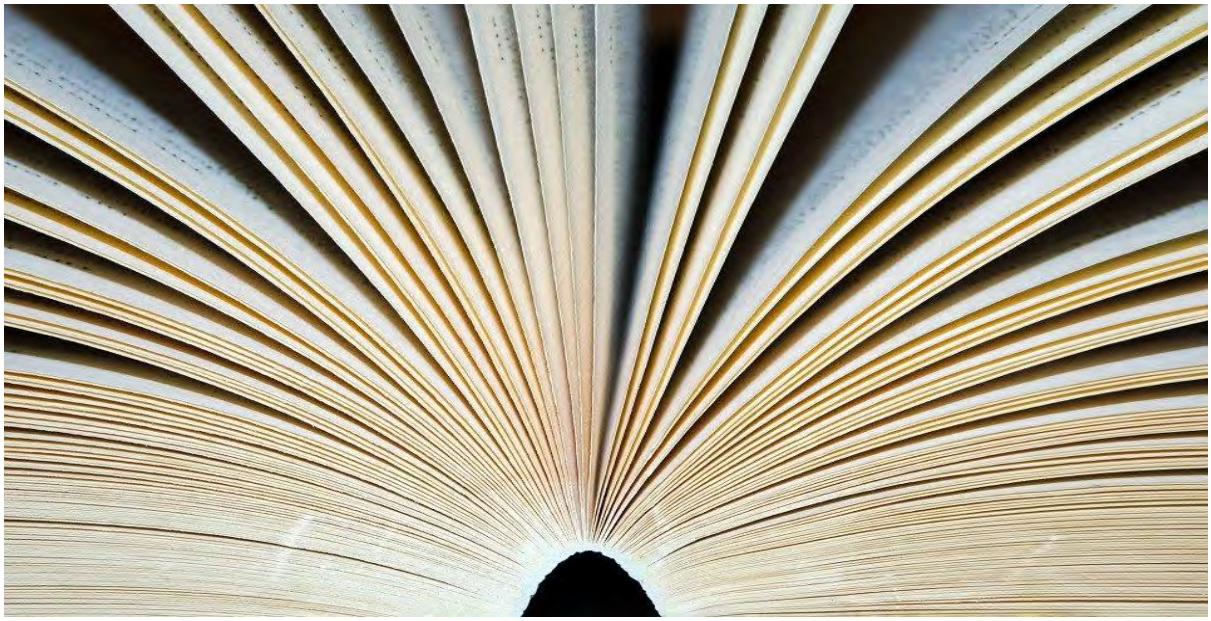
techniques. Extensive experiments are conducted to study the effectiveness of the HPC features that could distinguish between malware and normal applications. Work in this chapter has been accepted on 5/1/2020 to be included in the upcoming book entitled “Malware Analysis using Artificial Intelligence and Deep Learning”, which will be published by Springer.

Chapter 6 Our hybrid IDS architecture is applied to protect IoT infrastructure by detecting abnormal activities. The suggested IDS employs intrusion detection techniques to protect both device-to-device and device-to-gateway telecommunications. This chapter is published as a Journal paper (Khraisat, Gondal, Vamplew, Kamruzzaman, & Alazab, 2019).

Chapter 7 : The concluding chapter gives an outline of the contributions and future research directions and challenges.This chapter has presented an overview of the IDS architecture, research challenges, contribution made to address identified challenges and outline of the thesis.

Contributions of the thesis are presented in the form of publications in chapters 2-6. Each chapter gives brief commentary on the paper and then paper is included as it appeared in proceedings, Journals or books.

Chapter 2 : Intrusion Detection Systems – An Overview



This chapter presents a literature review on approaches used to tackle computer security threats using Signature-based Intrusion Detection System (SIDS) and Anomaly-based Intrusion Detection System (AIDS). A number of AIDS systems have also been developed as Network Intrusion Detection System (NIDS) and Host Intrusion Detection System (HIDS) to improve the detection performance with the use of machine learning, knowledge-based and statistical schemes. This chapter presents an overview of datasets, comprehensive reviews, research challenges and taxonomy of modern Intrusion Detection Systems (IDS) along with evasion techniques that can be used by the attackers to avoid detection.

It also discusses the limitation of intrusion detection systems that should be overcome to provide reliable intrusion detection for higher performance. This chapter is based on the survey paper published:

Khraisat, A., Gondal, I., Vamplew, P., & Kamruzzaman, J, “Survey of intrusion detection systems: techniques, datasets and challenges” Cybersecurity Journal, 2019

Khraisat et al. Cybersecurity (2019) 2:20
https://doi.org/10.1186/s42400-019-0038-7

Cybersecurity

SURVEY Open Access

Survey of intrusion detection systems: techniques, datasets and challenges

Ansam Khraisat*, Iqbal Gondal, Peter Vamplew and Joarder Kamruzzaman

Abstract

Cyber-attacks are becoming more sophisticated and thereby presenting increasing challenges in accurately detecting intrusions. Failure to prevent the intrusions could degrade the credibility of security services, e.g. data confidentiality, integrity, and availability. Numerous intrusion detection methods have been proposed in the literature to tackle computer security threats, which can be broadly classified into Signature-based Intrusion Detection Systems (SIDS) and Anomaly-based Intrusion Detection Systems (AIDS). This survey paper presents a taxonomy of contemporary IDS, a comprehensive review of notable recent works, and an overview of the datasets commonly used for evaluation purposes. It also presents evasion techniques used by attackers to avoid detection and discusses future research challenges to counter such techniques so as to make computer systems more secure.

Keywords: Malware, Intrusion detection system, NSL_KDD, Anomaly detection, Machine learning

Introduction

The evolution of malicious software (malware) poses a critical challenge to the design of intrusion detection systems (IDS). Malicious attacks have become more sophisticated and the foremost challenge is to identify unknown and obfuscated malware, as the malware authors use different evasion techniques for information concealing to prevent detection by an IDS. In addition, there has been an increase in security threats such as zero-day attacks designed to target internet users. Therefore, computer security has become essential as the use of information technology has become part of our daily lives. As a result, various countries such as Australia and the US have been significantly impacted by the zero-day attacks. According to the 2017 Symantec Internet Security Threat Report, more than three billion zero-day attacks were reported in 2016, and the volume and intensity of the zero-day attacks were substantially greater than previously (Symantec, 2017). As highlighted in the Data Breach Statistics in 2017, approximately nine billion data records were lost or stolen by hackers since 2013 (Breach_LeveL_Index, 2017). A Symantec report found that the number of security breach incidents is on the rise. In the past, cybercriminals primarily focused on bank customers, robbing bank accounts or stealing credit cards (Symantec, 2017). However, the new generation of malware has become more ambitious and is targeting the banks themselves, sometimes trying to take millions of dollars in one attack (Symantec, 2017). For that reason, the detection of zero-day attacks has become the highest priority.

High profile incidents of cybercrime have demonstrated the ease with which cyber threats can spread internationally, as a simple compromise can disrupt a business' essential services or facilities. There are a large number of cybercriminals around the world motivated to steal information, illegitimately receive revenues, and find new targets. Malware is intentionally created to compromise computer systems and take advantage of any weakness in intrusion detection systems. In 2017, the Australian Cyber Security Centre (ACSC) critically examined the different levels of sophistication employed by the attackers (Australian, 2017). So there is a need to develop an efficient IDS to detect novel, sophisticated malware. The aim of an IDS is to identify different kinds of malware as early as possible, which cannot be achieved by a traditional firewall. With the increasing volume of computer malware, the development of improved IDSs has become extremely important.

In the last few decades, machine learning has been used to improve intrusion detection, and currently there is a need for an up-to-date, thorough taxonomy and

* Correspondence: a.khraisat@federation.edu.au
Internet Commerce Security Laboratory, Federation University Australia, Mount Helen, Australia

survey of this recent work. There are a large number of related studies using either the KDD-Cup 99 or DARPA 1999 dataset to validate the development of IDSs; however there is no clear answer to the question of which data mining techniques are more effective. Secondly, the time taken for building IDS is not considered in the evaluation of some IDSs techniques, despite being a critical factor for the effectiveness of 'on-line' IDSs.

This paper provides an up to date taxonomy, together with a review of the significant research works on IDSs up to the present time; and a classification of the proposed systems according to the taxonomy. It provides a structured and comprehensive overview of the existing IDSs so that a researcher can become quickly familiar with the key aspects of anomaly detection. This paper also provides a survey of data-mining techniques applied to design intrusion detection systems. The signature-based and anomaly-based methods (i.e., SIDS and AIDS) are described, along with several techniques used in each method. The complexity of different AIDS methods and their evaluation techniques are discussed, followed by a set of suggestions identifying the best methods, depending on the nature of the intrusion. Challenges for the current IDSs are also discussed. Compared to previous survey publications (Patel et al., 2013; Liao et al., 2013a), this paper presents a discussion on IDS dataset problems which are of main concern to the research community in the area of network intrusion detection systems (NIDS). Prior studies such as (Sadotra & Sharma, 2016; Buczak & Guven, 2016) have not completely reviewed IDSs in term of the datasets, challenges and techniques. In this paper, we provide a structured and contemporary, wide-ranging study on intrusion detection system in terms of techniques and datasets; and also highlight challenges of the techniques and then make recommendations.

During the last few years, a number of surveys on intrusion detection have been published. Table 1 shows the IDS techniques and datasets covered by this survey and previous survey papers. The survey on intrusion detection system and taxonomy by Axelsson (Axelsson, 2000) classified intrusion detection systems based on the detection methods. The highly cited survey by Debar et al. (Debar et al., 2000) surveyed detection methods based on the behaviour and knowledge profiles of the attacks. A taxonomy of intrusion systems by Liao et al. (Liao et al., 2013a), has presented a classification of five subclasses with an in-depth perspective on their characteristics: Statistics-based, Pattern-based, Rule-based, State-based and Heuristic-based. On the other hand, our work focuses on the signature detection principle, anomaly detection, taxonomy and datasets.

Existing review articles (e.g., such as (Buczak & Guven, 2016; Axelsson, 2000; Ahmed et al., 2016; Lunt, 1988; Agrawal & Agrawal, 2015)) focus on intrusion detection techniques or dataset issue or type of computer attack and IDS evasion. No articles comprehensively reviewed intrusion detection, dataset problems, evasion techniques, and different kinds of attack altogether. In addition, the development of intrusion-detection systems has been such that several different systems have been proposed in the meantime, and so there is a need for an up-to-date. The updated survey of the taxonomy of intrusion-detection discipline is presented in this paper further enhances taxonomies given in (Liao et al., 2013a; Ahmed et al., 2016).

In view of the discussion on prior surveys, this article focuses on the following:

- Classifying various kinds of IDS with the major types of attacks based on intrusion methods.
- Presenting a classification of network anomaly IDS evaluation metrics and discussion on the importance of the feature selection.
- Evaluation of available IDS datasets discussing the challenges of evasion techniques.

Intrusion detection systems

Intrusion can be defined as any kind of unauthorised activities that cause damage to an information system. This

Table 1 Comparison of this survey and similar surveys: (✓: Topic is covered, ✖ the topic is not covered)

Survey	# of citation (as of 6/1/2019)	Intrusion Detection System Techniques						Dataset issue
		SIDS	AIDS				Hybrid IDS	
			Supervised learning	Unsupervised	Semi-supervised learning	Ensemble methods		
Lunt (1988)	219	✓	✖	✖	✖	✖	✖	✖
Axelsson (2000)	1039	✓	✓	✖	✖	✖	✖	✖
Liao, et al. (2013b)	505	✓	✓	✓	✖	✖	✓	✖
Agrawal and Agrawal (2015)	108	✓	✓	✓	✓	✓	✓	✖
Buczak and Guven (2016)	338	✓	✓	✓	✖	✓	✓	✓
Ahmed, et al. (2016)	181	✖	✓	✓	✖	✖	✖	✓
This survey		✓	✓	✓	✓	✓	✓	✓

means any attack that could pose a possible threat to the information confidentiality, integrity or availability will be considered an intrusion. For example, activities that would make the computer services unresponsive to legitimate users are considered an intrusion. An IDS is a software or hardware system that identifies malicious actions on computer systems in order to allow for system security to be maintained (Liao et al., 2013a). The goal of an IDS is to identify different kinds of malicious network traffic and computer usage, which cannot be identified by a traditional firewall. This is vital to achieving high protection against actions that compromise the availability, integrity, or confidentiality of computer systems. IDS systems can be broadly categorized into two groups: Signature-based Intrusion Detection System (SIDS) and Anomaly-based Intrusion Detection System (AIDS).

Signature-based intrusion detection systems (SIDS)

Signature intrusion detection systems (SIDS) are based on pattern matching techniques to find a known attack; these are also known as Knowledge-based Detection or Misuse Detection (Khraisat et al., 2018). In SIDS, matching methods are used to find a previous intrusion. In other words, when an intrusion signature matches with the signature of a previous intrusion that already exists in the signature database, an alarm signal is triggered. For SIDS, host's logs are inspected to find sequences of commands or actions which have previously been identified as malware. SIDS have also been labelled in the literature as Knowledge-Based Detection or Misuse Detection (Modi et al., 2013).

Figure 1 demonstrates the conceptual working of SIDS approaches. The main idea is to build a database of intrusion signatures and to compare the current set of activities against the existing signatures and raise an alarm if a match is found. For example, a rule in the form of "if: antecedent -then: consequent" may lead to "if (source IP address=destination IP address) then label as an attack ".

SIDS usually gives an excellent detection accuracy for previously known intrusions (Kreibich & Crowcroft, 2004). However, SIDS has difficulty in detecting zero-day attacks for the reason that no matching signature exists in the database until the signature of the new attack is extracted and stored. SIDS are employed in numerous common tools, for instance, Snort (Roesch, 1999) and NetSTAT (Vigna & Kemmerer, 1999).

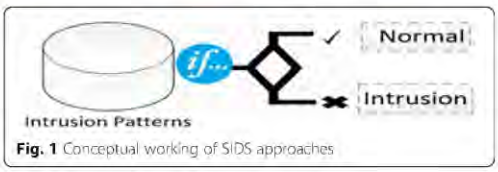


Fig. 1 Conceptual working of SIDS approaches

Traditional approaches to SIDS examine network packets and try matching against a database of signatures. But these techniques are unable to identify attacks that span several packets. As modern malware is more sophisticated it may be necessary to extract signature information over multiple packets. This requires the IDS to recall the contents of earlier packets. With regards to creating a signature for SIDS, generally, there have been a number of methods where signatures are created as state machines (Meiners et al., 2010), formal language string patterns or semantic conditions (Lin et al., 2011).

The increasing rate of zero-day attacks (Symantec, 2017) has rendered SIDS techniques progressively less effective because no prior signature exists for any such attacks. Polymorphic variants of the malware and the rising amount of targeted attacks can further undermine the adequacy of this traditional paradigm. A potential solution to this problem would be to use AIDS techniques, which operate by profiling what is an acceptable behavior rather than what is anomalous, as described in the next section.

Anomaly-based intrusion detection system (AIDS)

AIDS has drawn interest from a lot of scholars due to its capacity to overcome the limitation of SIDS. In AIDS, a normal model of the behavior of a computer system is created using machine learning, statistical-based or knowledge-based methods. Any significant deviation between the observed behavior and the model is regarded as an anomaly, which can be interpreted as an intrusion. The assumption for this group of techniques is that malicious behavior differs from typical user behavior. The behaviors of abnormal users which are dissimilar to standard behaviors are classified as intrusions. Development of AIDS comprises two phases: the training phase and the testing phase. In the training phase, the normal traffic profile is used to learn a model of normal behavior, and then in the testing phase, a new data set is used to establish the system's capacity to generalise to previously unseen intrusions. AIDS can be classified into a number of categories based on the method used for training, for instance, statistical based, knowledge-based and machine learning based (Butun et al., 2014).

The main advantage of AIDS is the ability to identify zero-day attacks due to the fact that recognizing the abnormal user activity does not rely on a signature database (Alazab et al., 2012). AIDS triggers a danger signal when the examined behavior differs from the usual behavior. Furthermore, AIDS has various benefits. First, they have the capability to discover internal malicious activities. If an intruder starts making transactions in a stolen account that are unidentified in the typical user activity, it creates an alarm. Second, it is very difficult for a cybercriminal to recognize what is a normal user

behavior without producing an alert as the system is constructed from customized profiles.

Table 2 presents the differences between signature-based detection and anomaly-based detection. SIDS can only identify well-known intrusions whereas AIDS can detect zero-day attacks. However, AIDS can result in a high false positive rate because anomalies may just be new normal activities rather than genuine intrusions.

Since there is a lack of a taxonomy for anomaly-based intrusion detection systems, we have identified five subclasses based on their features: Statistics-based, Pattern-based, Rule-based, State-based and Heuristic-based as shown in Table 3.

Intrusion data sources

The previous two sections categorised IDS on the basis of the methods used to identify intrusions. IDS can also be classified based on the input data sources used to detect abnormal activities. In terms of data sources, there are generally two types of IDS technologies, namely Host-based IDS (HIDS) and Network-based IDS (NIDS). HIDS inspect data that originates from the host system and audit sources, such as operating system, window server logs, firewalls logs, application system audits, or database logs. HIDS can detect insider attacks that do not involve network traffic (Creech & Hu, 2014a).

NIDS monitors the network traffic that is extracted from a network through packet capture, NetFlow, and other network data sources. Network-based IDS can be used to monitor many computers that are joined to a network. NIDS is able to monitor the external malicious activities that could be initiated from an external threat at an earlier phase, before the threats spread to another computer system. On the other hand, NIDSs have limited ability to inspect all data in a high bandwidth network because of the volume of data passing through modern high-speed communication networks (Bhuyan et al., 2014). NIDS deployed at a number of positions within a particular network topology, together with HIDS and firewalls, can provide a concrete, resilient, and multi-tier protection against both external and insider attacks.

Table 4 shows a summary of comparisons between HIDS and NIDS.

Creech et al. proposed a HIDS methodology applying discontinuous system call patterns, with the aim to raise detection rates while decreasing false alarm rates (Creech, 2014). The main idea is to use a semantic structure to kernel level system calls to understand anomalous program behaviour.

As shown in Table 5 a number of AIDS systems have also been applied in Network Intrusion Detection System (NIDS) and Host Intrusion Detection System (HIDS) to increase the detection performance with the use of machine learning, knowledge-based and statistical schemes. Table 5 also provides examples of current intrusion detection approaches, where types of attacks are presented in the detection capability field. Data source comprises system calls, application programme interfaces, log files, data packets obtained from well-known attacks. These data source can be beneficial to classify intrusion behaviors from abnormal actions.

Techniques for implementing AIDS

This section presents an overview of AIDS approaches proposed in recent years for improving detection accuracy and reducing false alarms.

AIDS methods can be categorized into three main groups: Statistics-based (Chao et al., 2015), knowledge-based (Elhag et al., 2015; Can & Sahingoz, 2015), and machine learning-based (Buczak & Guven, 2016; Meshram & Haas, 2017). The statistics-based approach involves collecting and examining every data record in a set of items and building a statistical model of normal user behavior. On the other hand, knowledge-based tries to identify the requested actions from existing system data such as protocol specifications and network traffic instances, while machine-learning methods acquire complex pattern-matching capabilities from training data.

Table 2 Comparisons of intrusion detection methodologies

		Advantages	Disadvantages
Detection methods	SIDS	• Very effective in identifying intrusions with minimum false alarms (FA). • Promptly identifies the intrusions. • Superior for detecting the known attacks. • Simple design	• Needs to be updated frequently with a new signature. • SIDS is designed to detect attacks for known signatures. When a previous intrusion has been altered slightly to a new variant, then the system would be unable to identify this new deviation of the similar attack. • Unable to detect the zero-day attack. • Not suitable for detecting multi-step attacks. • Little understanding of the insight of the attacks.
	AIDS	• Could be used to detect new attacks. • Could be used to create intrusion signature	• AIDS cannot handle encrypted packets, so the attack can stay undetected and can present a threat. • High false positive alarms. • Hard to build a normal profile for a very dynamic computer system. • Unclassified alerts. • Needs initial training.

Table 3 Detection methodology characteristics for intrusion-detection systems

Detection Methodology	Examples	Characteristics
Statistics based: analyzes the network traffic using complex statistical algorithms to process the information.	Bhuyan, et al. (2014)	•Needs a large amount of knowledge of statistics •Simple but less accurate •Real-time
Pattern-based: identifies the characters, forms, and patterns in the data.	Liao, et al. (2013a) Riesen and Bunke (2008)	•Easy to implement •Hash function could be used for identification.
Rule-based: uses an attack "signature" to detect a potential attack on the suspicious network traffic.	Hall, et al. (2009)	•The computational cost of rule-based systems could be very high because rules need pattern matching. •It is very hard to estimate what actions are going to occur and when •Requires a large number of rules for determining all possible attacks. •Low false positive rate •High detection rate
State-based: examines a stream of events to identify any possible attack.	Kenkre, et al. (2015a)	•Probabilistic, self-training •Low false positive rate.
Heuristic-based: identifies any abnormal activity that is out of the ordinary activity.	Abbasi, et al. (2014) Butun, et al. (2014)	•It needs knowledge and experience •Experimental and evolutionary learning

These three classes along with examples of their subclasses are shown in Fig. 2.

Statistics-based techniques

A statistics-based IDS builds a distribution model for normal behaviour profile, then detects low probability events and flags them as potential intrusions. Statistical AIDS essentially takes into account the statistical metrics such as the median, mean, mode and standard deviation of packets. In other words, rather than inspecting data traffic, each packet is monitored, which signifies the fingerprint of the flow. Statistical AIDS are employed to identify any type of differences in the present behavior from normal behavior. Statistical IDS normally use one of the following models.

Univariate: "Uni" means "one", so it means the data has only one variable. This technique is used when a statistical normal profile is created for only one measure of behaviours in computer systems. Univariate IDS look for abnormalities in each individual metric (Ye et al., 2002).

Multivariate: It is based on relationships among two or more measures in order to understand the relationships between variables. This model would be valuable if experimental data show that better classification can be achieved from combinations of correlated measures rather than analysing them separately. Ye et al. examine a multivariate quality control method to identify intrusions by building a long-term profile of normal activities (Ye et al., 2002). The main challenge for multivariate statistical IDs is that it is difficult to estimate distributions for high-dimensional data.

Time series model: A time series is a series of observations made over a certain time interval. A new observation

Table 4 Comparison of IDS technology types based on their positioning within the computer system

		Advantages	Disadvantages	Data source
Technology	HIDS	• HIDS can check end-to-end encrypted communications behaviour. • No extra hardware required. • Detects intrusions by checking hosts file system, system calls or network events. • Every packet is reassembled • Looks at the entire item, not streams only	• Delays in reporting attacks • Consumes host resources • Needs to be installed on each host. • It can monitor attacks only on the machine where it is installed.	• Audits records, log files, Application Program Interface (API), rule patterns, system calls.
	NIDS	•Detects attacks by checking network packets. •Not required to install on each host. •Can check various hosts at the same period. •Capable of detecting the broadest ranges of network protocols	•Challenge is to identify attacks from encrypted traffic. •Dedicated hardware is required. •It supports only identification of network attacks. •Difficult to analysis high-speed network. •The most serious threat is the insider attack.	•Simple Network Management Protocol (SNMP) •Network packets (TCP/UDP/ICMP), •Management Information Base (MIB) •Router NetFlow records

Table 5 Comparisons of IDS technology types, using examples from the literature. "P" indicates pre-defined attacks and "Z" indicates zero-day attacks

Detection Source			HIDS	NIDS	Capability
Detection methods	SIDS		Wagner and Soto (2002)	Hubballi and Suryanarayanan (2014)	P
	AIDS	Statistics based	Ara, Louzada & Diniz (2017)	Tan, et al. (2014); Camacho, et al. (2016)	Z
		Knowledge-based	Mitchell and Chen (2015) Creech and Hu (2014b)	Hendry and Yang (2008) Shakshuki, et al. (2013) Zargar, et al. (2013)	
		Machine learning	Du, et al. (2014) Warig, et al. (2010)	Elhag, et al. (2015); Kim, et al. (2014); Hu, et al. (2014)	
	SIDS+ AIDS		Alazab, et al. (2014); Stavroulakis and Stamp (2010); Liu, et al. (2015)		P + Z

is abnormal if its probability of occurring at that time is too low. Viinikka et al. used time series for processing intrusion detection alert aggregates (Viinikka et al., 2009). Qingtao et al. presented a method for detecting network abnormalities by examining the abrupt variation found in time series data (Qingtao & Zhiqing, 2005). The feasibility of this technique was validated through simulated experiments.

Knowledge-based techniques

This group of techniques is also referred toas an expert system method. This approach requires creating a knowledge base which reflects the legitimate traffic profile. Actions which differ from this standard profile are treated as an intrusion. Unlike the other classes of AIDS, the standard profile model is normally created based on human knowledge, in terms of a set of rules that try to define normal system activity.

The main benefit of knowledge-based techniques is the capability to reduce false-positive alarms since the system has knowledge about all the normal behaviors. However, in a dynamically changing computing environment, this kind of IDS needs a regular update on knowledge for the expected normal behavior which is a time-consuming task as gathering information about all normal behaviors is very difficult.

Finite state machine (FSM): FSM is a computation model used to represent and control execution flow.

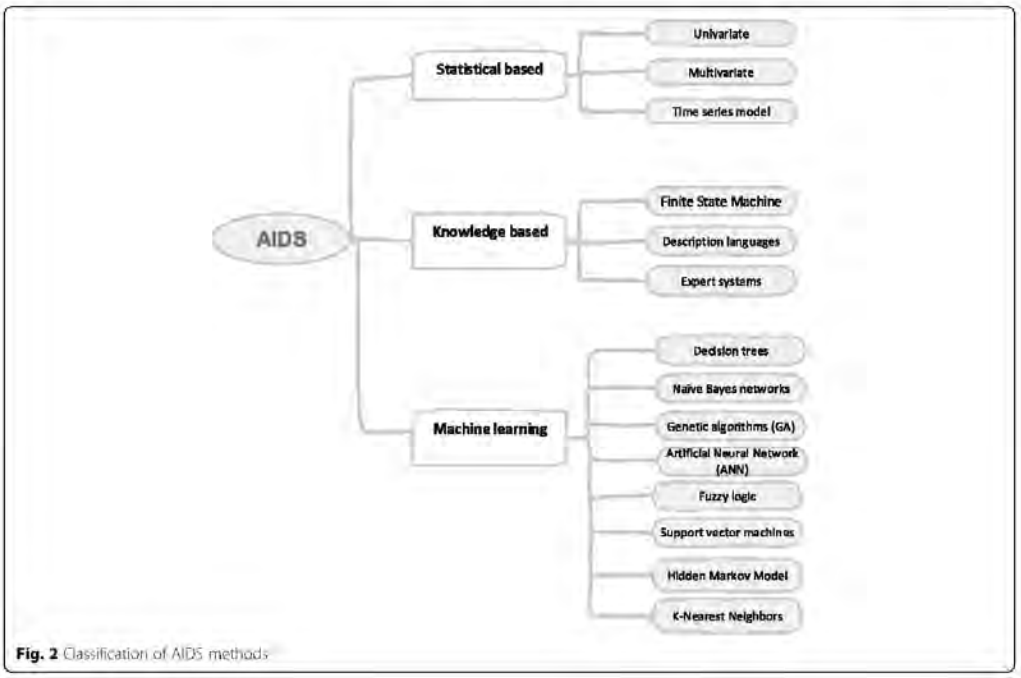


Fig. 2 Classification of AIDS methods

This model could be applied in intrusion detection to produce an intrusion detection system model. Typically, the model is represented in the form of states, transitions, and activities. A state checks the history data. For instance, any variations in the input are noted and based on the detected variation transition happens (Walkinshaw et al., 2016). An FSM can represent legitimate system behaviour, and any observed deviation from this FSM is regarded as an attack.

Description Language: Description language defines the syntax of rules which can be used to specify the characteristics of a defined attack. Rules could be built by description languages such as N-grammars and UML (Studnia et al., 2018).

Expert System: An expert system comprises a number of rules that define attacks. In an expert system, the rules are usually manually defined by a knowledge engineer working in collaboration with a domain expert (Kim et al., 2014).

Signature analysis: it is the earliest technique applied in IDS. It relies on the simple idea of string matching. In string matching, an incoming packet is inspected, word by word, with a distinct signature. If a signature is matched, an alert is raised. If not, the information in the traffic is then matched to the following signature on the signature database (Kenkre et al., 2015b).

AIDS based on machine learning techniques

Machine learning is the process of extracting knowledge from large quantities of data. Machine learning models comprise of a set of rules, methods, or complex "transfer functions" that can be applied to find interesting data patterns, or to recognise or predict behaviour (Dua & Du, 2016).

Machine learning techniques have been applied extensively in the area of AIDS. Several algorithms and techniques such as clustering, neural networks, association rules, decision trees, genetic algorithms, and nearest neighbour methods, have been applied for discovering the knowledge from intrusion datasets (Kshetri & Voas, 2017; Xiao et al, 2018).

Some prior research has examined the use of different techniques to build AIDSs. Chebrolu et al. examined the performance of two feature selection algorithms involving Bayesian networks (BN) and Classification Regression Trees (CRC) and combined these methods for higher accuracy (Chebrolu et al., 2005).

Bajaj et al. proposed a technique for feature selection using a combination of feature selection algorithms such as Information Gain (IG) and Correlation Attribute evaluation. They tested the performance of the selected features by applying different classification algorithms such as C4.5, naïve Bayes, NB-Tree and Multi-Layer Perceptron (Khraisat et al., 2018; Bajaj & Arora, 2013). A genetic-fuzzy rule mining method has been used to evaluate the importance of IDS features (Elhag et al., 2015). Thaseen et al. proposed NIDS by using Random Tree model to improve the accuracy and reduce the false alarm rate (Thaseen & Kumar, 2013). Subramanian et al. proposed classifying NSL-KDD dataset using decision tree algorithms to construct a model with respect to their metric data and studying the performance of decision tree algorithms (Subramanian et al., 2012).

Various AIDSs have been created based on machine learning techniques as shown in Fig. 3. The objective of using machine learning techniques is to create IDS with improved accuracy and less requirement for human knowledge. In the last few years, the quantity of AIDS which have used machine learning methods has been increasing. A key focus of IDS based on machine learning research is to detect patterns and build intrusion detection system based on the dataset. Generally, there are two kinds of machine learning methods, supervised and unsupervised.

Supervised learning in intrusion detection system

This section presents various supervised learning techniques for IDS. Each technique is presented in detail, and references to important research publications are presented.

Supervised learning-based IDS techniques detect intrusions by using labeled training data. A supervised learning approach usually consists of two stages, namely training and testing. In the training stage, relevant features and classes are identified and then the algorithm learns from these data samples. In supervised learning IDS, each record is a pair, containing a network or host data source and an associated output value (i.e., label), namely intrusion or normal. Next, feature selection can be applied for eliminating unnecessary features. Using the training data for selected features, a supervised learning technique is then used to train a classifier to learn the inherent relationship that exists between the input data and the labelled output value. A wide variety of supervised learning techniques have been explored in the literature, each with its advantages and disadvantages. In the testing stage, the trained model is used to classify the unknown data into intrusion or normal class. The resultant classifier then becomes a model which, given a set of feature values, predicts the class to which the input data might belong. Figure 4 shows a general approach for applying classification techniques. The performance of a classifier in its ability to predict the correct class is measured in terms of a number of metrics is discussed in Section 4.

There are many classification methods such as decision trees, rule-based systems, neural networks, support vector machines, naïve Bayes and nearest-neighbor. Each technique uses a learning method to build a classification model. However, a suitable classification approach

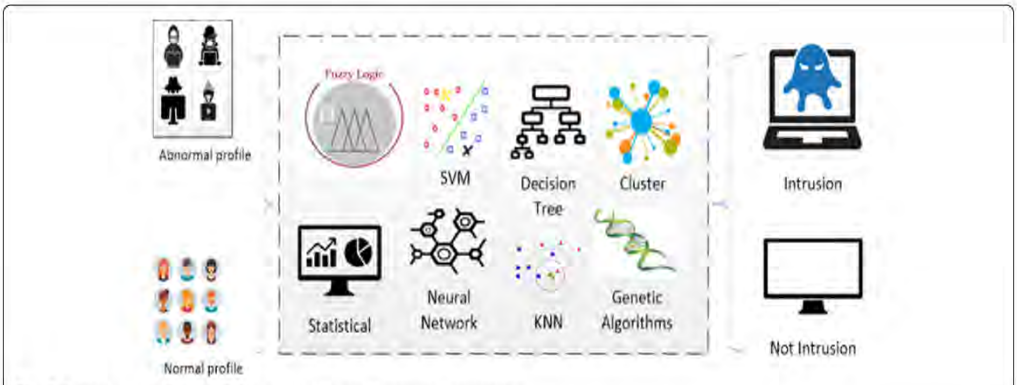


Fig. 3 Conceptual working of AIDS approaches based on machine learning

should not only handle the training data, but it should also identify accurately the class of records it has not ever seen before. Creating classification models with reliable generalization ability is an important task of the learning algorithm.

Decision trees: A decision tree comprises of three basic components. The first component is a decision node, which is used to identify a test attribute. The second is a branch, where each branch represents a possible decision based on the value of the test attribute. The third is a leaf that comprises the class to which the instance belongs (Rutkowski et al., 2014). There are many different decision trees algorithms including ID3 (Quinlan, 1986), C4.5 (Quinlan, 2014) and CART (Breiman, 1996).

Naïve Bayes: This approach is based on applying Bayes' principle with robust independence assumptions among the attributes. Naïve Bayes answers questions such as "what is the probability that a particular kind of attack is occurring, given the observed system activities?" by applying conditional probability formulae. Naïve Bayes relies on the features that have different probabilities of occurring in attacks and in normal behavior. Naïve Bayes classification model is one of the most prevalent models in IDS due to its ease of use and calculation efficiency, both of which are taken from its conditional independence assumption property (Yang & Tian, 2012). However, the system does not operate well if this independence assumption is not valid, as was demonstrated on the KDD'99 intrusion detection dataset which has complex attribute dependencies (Koc et al., 2012). The results also reveal that the Naïve Bayes model has reduced accuracy for large datasets. A further study showed that the more sophisticated Hidden Naïve Bayes (HNB) model can be applied to IDS tasks that involve high dimensionality, extremely interrelated attributes and high-speed networks (Koc et al., 2012).

Genetic algorithms (GA): Genetic algorithms are a heuristic approach to optimization, based on the principles of evolution. Each possible solution is represented as a series of bits (genes) or chromosome, and the quality of the solutions improves over time by the application of selection and reproduction operators, biased to favour fitter solutions. In applying a genetic algorithm to the intrusion classification problem, there are typically two types of chromosome encoding: one is according to clustering to generate binary chromosome coding method; another is specifying the cluster center (clustering prototype matrix) by an integer coding chromosome.

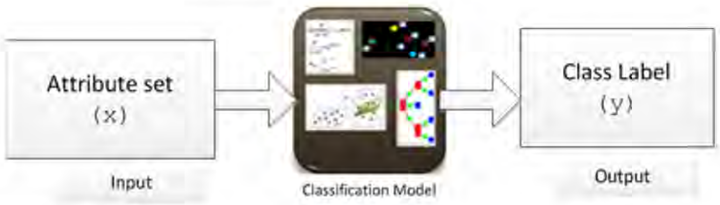


Fig. 4 Classification as the task

Murray et al., has used GA to evolve simple rules for network traffic (Murray et al., 2014). Every rule is represented by a genome and the primary population of genomes is a number of random rules. Each genome is comprised of different genes which correspond to characteristics such as IP source, IP destination, port source, port destination and 1 protocol type (Hoque & Bikas, 2012).

Artificial Neural Network (ANN): ANN is one of the most broadly applied machine-learning methods and has been shown to be successful in detecting different malware. The most frequent learning technique employed for supervised learning is backpropagation (BP) algorithm. The BP algorithm assesses the gradient of the network's error with respect to its modifiable weights. However, for ANN-based IDS, detection precision, particularly for less frequent attacks, and detection accuracy still need to be improved. The training dataset for less-frequent attacks is small compared to that of more-frequent attacks and this makes it difficult for the ANN to learn the properties of these attacks correctly. As a result, detection accuracy is lower for less frequent attacks. In the information security area, huge damage can occur if low-frequency attacks are not detected. For instance, if the User to Root (U2R) attacks evade detection, a cybercriminal can gain the authorization privileges of the root user and thereby carry out malicious activities on the victim's computer systems. In addition the less common attacks are often outliers (Wang et al., 2010). ANNs often suffer from local minima and thus learning can become very time-consuming. The strength of ANN is that, with one or more hidden layers, it is able to produce highly nonlinear models which capture complex relationships between input attributes and classification labels. With the development of many variants such as recurrent and convolutional NNs, ANNs are powerful tools in many classification tasks including IDS.

Fuzzy logic: This technique is based on the degrees of uncertainty rather than the typical true or false Boolean logic on which the contemporary PCs are created. Therefore, it presents a straightforward way of arriving at a final conclusion based upon unclear, ambiguous, noisy, inaccurate or missing input data. With a fuzzy domain, fuzzy logic permits an instance to belong, possibly partially, to multiple classes at the same time. Therefore, fuzzy logic is a good classifier for IDS problems as the security itself includes vagueness, and the borderline between the normal and abnormal states is not well identified. In addition, the intrusion detection problem contains various numeric features in the collected data and several derived statistical metrics. Building IDSs based on numeric data with hard thresholds produces high false alarms. An activity that deviates only slightly from a model could not be recognized or a minor change in normal activity could produce false alarms. With fuzzy logic, it is possible to model this minor abnormality to keep the false rates low. Elhag et al. showed that with fuzzy logic, the false alarm rate in determining intrusive actions could be decreased. They outlined a group of fuzzy rules to describe the normal and abnormal activities in a computer system, and a fuzzy inference engine to define intrusions (Elhag et al., 2015).

Support Vector Machines (SVM): SVM is a discriminative classifier defined by a splitting hyperplane. SVMs use a kernel function to map the training data into a higher-dimensioned space so that intrusion is linearly classified. SVMs are well known for their generalization capability and are mainly valuable when the number of attributes is large and the number of data points is small. Different types of separating hyperplanes can be achieved by applying a kernel, such as linear, polynomial, Gaussian Radial Basis Function (RBF), or hyperbolic tangent. In IDS datasets, many features are redundant or less influential in separating data points into correct classes. Therefore, features selection should be considered during SVM training. SVM can also be used for classification into multiple classes. In the work by Li et al., an SVM classifier with an RBF kernel was applied to classify the KDD 1999 dataset into predefined classes (Li et al., 2012). From a total of 41 attributes, a subset of features was carefully chosen by using feature selection method.

Hidden Markov Model (HMM): HMM is a statistical Markov model in which the system being modeled is assumed to be a Markov process with unseen data. Prior research has shown that HMM analysis can be applied to identify particular kinds of malware (Annachhatre et al., 2015). In this technique, a Hidden Markov Model is trained against known malware features (e.g., operation code sequence) and once the training stage is completed, the trained model is applied to score the incoming traffic. The score is then contrasted to a predefined threshold, and a score greater than the threshold indicates malware. Likewise, if the score is less than the threshold, the traffic is identified as normal.

K-Nearest Neighbors (KNN) classifier: The k-Nearest Neighbor (k-NN) techniques is a typical non-parametric classifier applied in machine learning (Lin et al., 2015). The idea of these techniques is to name an unlabelled data sample to the class of its k nearest neighbors (where k is an integer defining the number of neighbours to be considered). Figure 5 illustrates a K-Nearest Neighbors classifier where k = 5. The point X represents an instance of unlabelled date which needs to be classified. Amongst the five nearest neighbours of X there are three similar patterns from the class Intrusion and two from the class Normal. Taking a majority vote enables the assignment of X to the Intrusion class.

k-NN can be appropriately applied as a benchmark for all the other classifiers because it provides a good classification performance in most IDSs (Lin et al., 2015).

Unsupervised learning in intrusion detection system

Unsupervised learning is a form of machine learning technique used to obtain interesting information from input datasets without class labels. The input data points are normally treated as a set of random variables. A joint density model is then created for the data set. In supervised learning, the output labels are given and used to train the machine to get the required results for an unseen data point, while in unsupervised learning, no labels are given, and instead the data is grouped automatically into various classes through the learning process. In the context of developing an IDS, unsupervised learning means, use of a mechanism to identify intrusions by using unlabelled data to a train the model.

As shown in Fig. 6, once records are clustered, all of the cases that appear in small clusters are labelled as an intrusion because the normal occurrences should produce sizable clusters compared to the anomalies. In addition, malicious intrusions and normal instances are dissimilar, thus they do not fall into the identical cluster.

K-means: The K-means techniques is one of the most prevalent techniques of clustering analysis that aims to separate 'n' data objects into 'k' clusters in which each data object is selected in the cluster with the nearest mean. It is a distance-based clustering technique and it does not need to compute the distances between all combinations of records. It applies a Euclidean metric as a similarity measure. The number of clusters is determined by the user in advance. Typically several solutions will be tested before accepting the most appropriate one. Annachhatre et.al. used the K-means clustering algorithm to identify different host behaviour profiles (Annachhatre et al., 2015). They have proposed new distance metrics which can be used in the k-means algorithm to closely relate the clusters. They have clustered data into several clusters and associated them with known behavior for evaluation. Their outcomes have revealed that k-means clustering is a better approach to classify the data using unsupervised methods for intrusion detection when several kinds of datasets are available. Clustering could be used in IDS for reducing intrusion signatures, generate a high-quality signature or group similar intrusion.

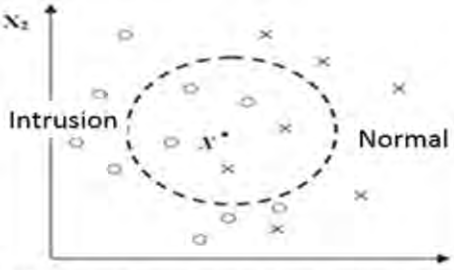


Fig. 5 An example of classification by k-Nearest Neighbour for k = 5

Hierarchical Clustering: This is a clustering technique which aims to create a hierarchy of clusters. Approaches for hierarchical clustering are normally classified into two categories:

(i) Agglomerative- bottom-up clustering techniques where clusters have sub-clusters, which in turn have sub-clusters and pairs of clusters are combined as one moves up the hierarchy.
(ii) Divisive - hierarchical clustering algorithms where iteratively the cluster with the largest diameter in feature space is selected and separated into binary sub-clusters with lower range.

A lot of work has been done in the area of the cyber-physical control system (CPCS) with attack detection and reactive attack mitigation by using unsupervised learning. For example, a redundancy-based resilience approach was proposed by Alcara (Alcaraz, 2018). He proposed a dedicated network sublayer that has the capability to handle the context by regularly collecting consensual information from the driver nodes controlled in the control network itself, and discriminating view differences through data mining techniques such as k-means and k-nearest neighbour. Chao Shen et al. proposed Hybrid-Augmented device fingerprinting for IDS in Industrial Control System Networks. They used different machine learning techniques to analyse network packets to filter anomaly traffic to detect in the intrusions in ICS networks (Shen et al., 2018).

Semi-supervised learning

Semi-supervised learning falls between supervised learning (with totally labelled training data) and unsupervised learning (without any categorized training data). Researchers have shown that semi-supervised learning could be used in conjunction with a small amount of labelled data classifier's performance for the IDSs with less time and costs needed. This is valuable as for many IDS issues, labelled data can be rare or occasional (Ashfaq et al., 2017).

A number of different techniques for semi-supervised learning have been proposed, such as the Expectation Maximization (EM) based algorithms (Goldstein, 2012), self-training (Blount et al., 2011; Lyngdoh et al., 2018), co-training (Rath et al., 2017), Semi-Supervised SVM (Ashfaq et al., 2017), graph-based methods (Sadreazami

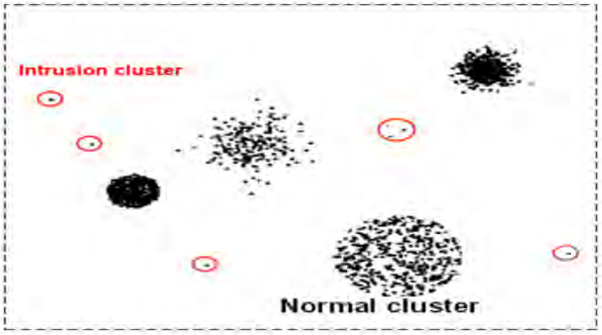


Fig. 6 Using Clustering for Intrusion Detection

et al., 2018), and boosting based semi-supervised learning methods (Yuan et al., 2016).

Rana et al. propose a novel fuzzy-based semi-supervised learning approach by applying unlabelled samples aided with a supervised learning algorithm to enhance the classifier's performance for the IDSs. A single hidden layer feed-forward neural network (SLFN) is trained to output a fuzzy membership vector, and the sample categorization (low, mid, and high fuzziness categories) on unlabelled samples is performed using the fuzzy quantity (Ashfaq et al., 2017). The classifier is retrained after incorporating each category separately into the original training set. Their experimental results using this semi-supervised of intrusion detection on the NSL-KDD dataset show that unlabelled samples belonging to low and high fuzziness groups cause foremost contributions to enhance the accuracy of IDS contrasted to traditional.

Ensemble methods

Multiple machine learning algorithms can be used to obtain better predictive performance than any of the constituent learning algorithms alone. A number of different ensemble methods have been proposed, such as Boosting, Bagging and Stacking.

Boosting refers to a family of algorithms that are able to transform weak learners to strong learners. Bagging means training the same classifier on different subsets of same dataset. Stacking combines various classification via a meta-classifier (Aburomman & Reaz, 2016). The base level models are built based on a whole training set, then the meta-model is trained on the outputs of the base level model as attributes.

Jabbar et al. proposed an ensemble classifier which is built using Random Forest and also the Average One-Dependence Estimator (AODE which solves the attribute dependency problem in Naïve Bayes classifier. Random Forest (RF) enhances precision and reduces false alarms (Jabbar et al., 2017). Combining both approaches in an ensemble results in improved accuracy over either technique applied independently.

Hybrid based techniques

Traditional IDSs have limitations: that they cannot be easily modified, inability to identify new malicious attacks, low accuracy and high false alarms. Where AIDS has a limitation such as high false positive rate. Hybrid IDS is based on the combination of SIDS and AIDS. A Hybrid IDS overcomes the disadvantage of SIDS and AIDS. Farid et al. (Farid et al., 2010) proposed hybrid IDS by using Naive Bayes and decision tree based and achieved detection rate of 99.63% on the KDD'99 dataset.

Performance metrics for IDS

There are many classification metrics for IDS, some of which are known by multiple names. Table 6 shows the confusion matrix for a two-class classifier which can be used for evaluating the performance of an IDS. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class.

IDS are typically evaluated based on the following standard performance measures:

- True Positive Rate (TPR): It is calculated as the ratio between the number of correctly predicted attacks and the total number of attacks. If all intrusions are

Table 6 Confusion Matrix for IDS System

Actual Class	Predicted Class		
	Class	Normal	Attack
	Normal	True negative (TN)	False Positive (FP)
	Attack	False Negative (FN)	True positive (TP)

detected then the TPR is 1 which is extremely rare for an IDS. TPR is also called a Detection Rate (DR) or the Sensitivity. The TPR can be expressed mathematically as

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

- False Positive Rate (FPR): It is calculated as the ratio between the number of normal instances incorrectly classified as an attack and the total number of normal instances.

$$FPR = \frac{FP}{FP + TN}$$

- False Negative Rate (FNR): False negative means when a detector fails to identify an anomaly and classifies it as normal. The FNR can be expressed mathematically as:

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

- Classification rate (CR) or Accuracy: The CR measures how accurate the IDS is in detecting normal or anomalous traffic behavior. It is described as the percentage of all those correctly predicted instances to all instances:

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

Receiver Operating Characteristic (ROC) curve: ROC has FPR on the x-axis and TPR on the y-axis. In ROC curve the TPR is plotted as a function of the FPR for different cut-off points. Each point on the ROC curve represents a FPR and TPR pair corresponding to a certain decision threshold. As the threshold for classification is varied, a different point on the ROC is selected with different False Alarm Rate (FAR) and different TPR. A test with perfect discrimination (no overlap in the two distributions) has a ROC curve that passes through the upper left corner (100% sensitivity, 100% specificity). The ROC Curve is shown in Fig. 7.

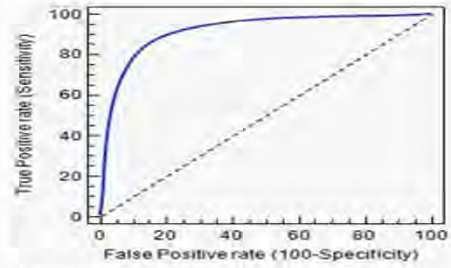


Fig. 7 ROC curve

Intrusion detection datasets

The evaluation datasets play a vital role in the validation of any IDS approach, by allowing us to assess the proposed method's capability in detecting intrusive behavior. The datasets used for network packet analysis in commercial products are not easily available due to privacy issues. However, there are a few publicly available datasets such as DARPA, KDD, NSL-KDD and ADFA-LD and they are widely used as benchmarks. Existing datasets that are used for building and comparative evaluation of IDS are discussed in this section along with their features and limitations.

DARPA / KDD Cup99

The earliest effort to create an IDS dataset was made by DARPA (Defence Advanced Research Project Agency) in 1998 and they created the KDD98 (Knowledge Discovery and Data Mining (KDD)) dataset. In 1998, DARPA introduced a programme at the MIT Lincoln Labs to provide a comprehensive and realistic IDS benchmarking environment (MIT Lincoln Laboratory, 1999). Although this dataset was an important contribution to the research on IDS, its accuracy and capability to consider real-life conditions have been widely criticized (Creech & Hu, 2014b).

These datasets were collected using multiple computers connected to the Internet to model a small US Air Force base of restricted personnel. Network packets and host log files were collected. Lincoln Labs built an experimental testbed to obtain 2 months of TCP packets dump for a Local Area Network (LAN), modelling a usual US Air Force LAN. They modelled the LAN as if it were a true Air Force environment, but interlaced it with several simulated intrusions.

The collected network packets were around four gigabytes containing about 4,900,000 records. The test data of 2 weeks had around 2 million connection records, each of which had 41 features and was categorized as normal or abnormal.

The extracted data is a series of TCP sessions starting and ending at well-defined times, between which data flows to and from a source IP address to a target IP address, which contains a large variety of attacks simulated in a military network environment. The 1998 DARPA Dataset was used as the basis to derive the KDD Cup99 dataset which has been used in Third International Knowledge Discovery and Data Mining Tools Competition (KDD, 1999). The 41 features of the KDD Cup99 dataset are presented in Table 7.

These datasets are out-of-date as they do not contain records of recent malware attacks. For example, attackers' behaviors are different in different network topologies, operating systems, and software and crime toolkits. Nevertheless, KDD99 remains in use as a benchmark within IDS research community and is still presently being used by researchers (Alazab et al., 2014; Duque & Omar, 2015; Ji et al., 2016).

CAIDA

This dataset contains network traffic traces from Distributed Denial-of-Service (DDoS) attacks, and was collected in 2007 (Hick et al., 2007). This type of denial-of-service attack attempts to interrupt normal traffic of a targeted computer, or network by overwhelming the target with a flood of network packets, preventing regular traffic from reaching its legitimate destination computer. One disadvantage of the CAIDA dataset is that it does not contain a diversity of the attacks. In addition, the gathered data does not contain features from the whole network which makes it difficult to distinguish between abnormal and normal traffic flows.

NSL-KDD

NSL-KDD is a public dataset, which has been developed from the earlier KDD cup99 dataset (Tavallaee et al., 2009). A statistical analysis performed on the cup99 dataset raised important issues which heavily influence the intrusion detection accuracy, and results in a misleading evaluation of AIDS (Tavallaee et al., 2009).

The main problem in the KDD data set is the huge amount of duplicate packets. Tavallaee et al. analyzed KDD training and test sets and revealed that approximately 78% and 75% of the network packets are duplicated in both the training and testing dataset (Tavallaee et al., 2009). This huge quantity of duplicate instances in the training set would influence machine-learning methods to be biased towards normal instances and thus prevent them from learning irregular instances which are typically more damaging to the computer system. Tavallaee et al. built the NSL-KDD dataset in 2009 from the KDD Cup'99 dataset to resolve the matters stated above by eliminating duplicated records (Tavallaee et al., 2009). The NSL-KDD train dataset consists of 125,973 records and the test dataset contains 22,544 records. The size of the NSL-KDD dataset is sufficient to make it practical to use the whole NSL-KDD dataset without the necessity to sample randomly. This has produced consistent and comparable results from various research works. The NSL_KDD dataset comprises 22 training intrusion attacks and 41 attributes (i.e., features). In this dataset, 21 attributes refer to the connection itself and 19 attributes describe the nature of connections within the same host (Tavallaee et al., 2009).

Table 7 The 41 features of KDD Cup99 dataset

Label	Network data feature	Label	Network data feature	Label	Network data feature	Label	Network data feature
A	duration	L	Logged in	W	count	AH	dst_host_same_srv_rate
B	protocol-type	M	num_comprised	X	srv_count	AI	dst_host_diff_srv_rate
C	service	N	root_shell	Y	serror_rate	AJ	dst_host_same_src_port_rate
D	flag	O	Stu attempted	Z	srv_serror_rate	AK	dst_host_srv_diff_host_rate
E	src_bytes	P	num_root	AA	rerror_rate	AL	dst_host_serror_rate
F	dst_bytes	Q	Num of file	AB	srv_rerror_rate	AM	dst_host_srv_serror_rate
G	land	R	Number of shell	AC	same_srv_rate	AN	dst_host_rerror_rate
H	wrong_fragment	S	num_access_files	AD	diff_srv_rate	AO	dst_host_srv_rerror_rate
I	urgent	T	num_outbound_cmds	AE	srv_diff_host_rate		
J	hot	U	Is host login	AF	dst_host_count		
K	num_falied_logins	V	Is guest login	AG	dst_host_srv_count		

ISCX 2012

In this dataset, real network traffic traces were analyzed to identify normal behaviour for computers from real traffic of HTTP, SMTP, SSH, IMAP, POP3, and FTP protocols (Shiravi et al., 2012). This dataset is based on realistic network traffic, which is labeled and contains diverse attacks scenarios.

ADFA-LD and ADFA-WD

Researchers at the Australian Defence Force Academy created two datasets (ADFA-LD and ADFA-WD) as public datasets that represent the structure and methodology of the modern attacks (Creech, 2014). The datasets contain records from both Linux and Windows operating systems; they are created from the evaluation of system-call-based HIDS. Ubuntu Linux version 11.04 was used as the host operating system to build ADFA-LD (Creech & Hu, 2014b). Some of the attack instances in ADFA-LD were derived from new zero-day malware, making this dataset suitable for highlighting differences between SIDS and AIDS approaches to intrusion detection. It comprises three dissimilar data categories, each group of data containing raw system call traces. Each training dataset was gathered from the host for normal activities, with user behaviors ranging from web browsing to LATEX document preparation. Table 8 shows some of the ADFA-LD features with the type and the description for each feature.

ADFA-LD also incorporates system call traces of different types of attacks. The ADFA Windows Dataset (ADFA-WD) provides a contemporary Windows dataset for evaluation of HIDS. Table 9 shows the number of systems calls for each category of AFDA-LD and AFDA-WD Table 10 describes details of each attack class in the ADFA-LD dataset. Table 11 lists the ADFA-WD Vectors and Effects.

Table 8 Features of ADFA-LD cataset (Creech, 2014)

Name	Type	Description
srcip	nominal	Source IP address
sport	integer	Source port number
dstip	nominal	Destination IP address
dsport	integer	Destination port number
proto	nominal	Transaction protocol
state	nominal	Indicates to the state and its dependent protocol
dur	Float	Record total duration
sbytes	Integer	Source to destination transaction bytes
dbytes	Integer	Destination to source transaction bytes
sttl	Integer	Source to destination time to live value
dttl	Integer	Destination to source time to live value
sloss	Integer	Source packets retransmitted or dropped
dloss	Integer	Destination packets retransmitted or dropped
service	nominal	http, ftp, smtp, ssh, dns, ftp-data ,irc and (-) if not much used service
Sload	Float	Source bits per second
Dload	Float	Destination bits per second
Spkts	integer	Source to destination packet count
Dpkts	integer	Destination to source packet count
swin	integer	Source TCP window advertisement value
dwin	integer	Destination TCP window advertisement value
stcpb	integer	Source TCP base sequence number
dtcpb	integer	Destination TCP base sequence number
smeansz	integer	Mean of the how packet size transmitted by the src
dmeansz	integer	Mean of the how packet size transmitted by the dst
trans_depth	integer	Represents the pipelined depth into the connection of http request/response transaction
res_bdy_len	integer	Actual uncompressed content size of the data transferred from the server's http service.

Table 9 Number of system calls traces in different categories of ADFA-LD and ADFA-WD

ADFA-LD			ADFA-WD	
Dataset	Traces	System Calls	Traces	System Calls
Training data	833	308,077	355	13,504,419
Validation data	4372	2,122,085	1827	117,918,735
Attack data	746	317,388	5542	74,202,804
Total	5951	2,747,550	7724	205,625,958

Table 11 ADFA-WD Vectors and Effects

Vectors
TCP ports - Web-based vectors Browser attacks - Malware attachments
Effects
Effects - Bind Shell - Reverse shell - exploitation Remote operation - Staging - System manipulation Privilege escalation - Data exfiltration -Back door insertion

CICIDS 2017

CICIDS2017 dataset comprises both benign behaviour and also details of new malware attacks: such as Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet and DDoS (Sharafaldin et al., 2018). This dataset is labelled based on the timestamp, source and destination IPs, source and destination ports, protocols and attacks. A complete network topology was configured to collect this dataset which contains Modem, Firewall, Switches, Routers, and nodes with different operating systems (Microsoft Windows (like Windows 10, Windows 8, Windows 7, and Windows XP), Apple's macOS iOS, and open source operating system Linux). This dataset contains 80 network flow features from the captured network traffic.

Comparison of public IDS datasets

Since machine learning techniques are applied in AIDS, the datasets that are used for the machine learning techniques are very important to assess these techniques for realistic evaluation. Table 12 summarises popular public data sets, as well as some analysis techniques and results for each dataset from prior research. Table 13 summarizes the characteristics of the datasets.

Feature selection for IDS

Feature selection is helpful to decrease the computational difficulty, eliminate data redundancy, enhance the detection rate of the machine learning techniques, simplify data and reduce false alarms. In this line of research, some methods have been applied to develop a lightweight IDSs.

Feature selection techniques can be categorized into wrapper and filter methods. Wrapper methods estimate subgroups of variables to identify the feasible interactions between variables. There are two main drawbacks of these techniques: accumulative overfitting when the amount of data is insufficient and the important calculation time when the amount of variables is big.

Filter methods are normally applied as a pre-processing stage. The selection of features is separate of any machine learning techniques. As an alternative, features are nominated on the basis of their scores in several statistical tests for their correlation with the consequence variable.

As an example of the impact of feature selection on the performance of an IDS, consider the results in Table 14 which show the detection accuracy and time to build the IDS mode of the C4.5 classifier using the full dataset with 41 features of NSI-KDD dataset and with different features.

Types of computer attacks

Cyber-attacks can be categorized based on the activities and targets of the attacker. Each attack type can be classified into one of the following four classes (Sung & Mukkamala, 2003):

- Denial-of-Service (DoS) attacks have the objective of blocking or restricting services delivered by the network, computer to the users.
- Probing attacks have the objective of acquisition of information about the network or the computer system.
- User-to-Root (U2R) attacks have the objective of a non-privileged user acquiring root or admin-user access on a specific computer or a system on which the intruder had user level access.
- Remote-to-Local (R2L) attacks involve sending packets to the victim machine. The cybercriminal

Table 10 ADFA-LD attack class

Attack	Payload	Vector	Count
Hydra-FTP	Password brute force	FTP by Hydra	162
Hydra-SSH	Password brute force	SSH Hydra	176
Adduser	Add new super user	Client-side poisoned executable	91
Java-Meterpreter	Java based Meterpreter	TikiWiki vulnerability exploit	124
Meterpreter	Linux Meterpreter Payload	Client side poisoned executable	75
Webshell	C100 Webshell	PHP remote file inclusion vulnerability	118

Table 12 Comparison of results achieved by various methods on publically available IDS datasets

Dataset	Result	Observations	Reference
DARPA 98	Snort's detection, 69% of total generated alerts are considered to be false alarms.	SIDS is applied without AIDS	Hu, et al. (2009)
	ANN analysis system calls, 96% detection rate.	A classifier based on artificial neural network (ANN) has been executed for preparing and testing of framework.	McHugh (2000)
	SVM on subset of DARPA 98, 99.6% detection rate.	SVM isolates information into various classes by a hyperplane or hyperplanes since it can deal with multidimensional information, SVM usually demonstrate good performance for a binary class problem.	Chen, et al. (2005)
KDDCUP 99	Multivariate statistical analysis of audit data, 90% detection rate	Multivariate is used to reduce false alarm rates.	Ye, et al. (2002), Hotta, et al. (2008)
	The best results have been achieved by the C4.5 algorithm which attains the 95% true positive rate.	The decision trees created by C4.5 can be utilized for classification	Ferrari and Cribari-Neto (2004); Shafi and Abbass (2013); Laskov, et al. (2005)
	SMO classifier 97% detection rate.	This SVM based classifier with SMO implementation produces good detection accuracy. However, the accuracy reported is less than that in (Chen et al., 2005), because the KDDCUP 99 dataset is more complex and comprehensive than DARPA 98 dataset.	Shafi and Abbass (2013)
	The best model is an HNB model, where 95% confidence level is used to compare the models.	Hidden Naïve Bayes (HNB) techniques could be applied to IDS area that suffer from dimensionality, highly associated attributes and high network speed. HNB technique is better than the one based on the traditional NB method in terms of detection accuracy for IDS.	Koc, et al. (2012)
NSL-KDD	K-Nearest Neighbour (k-NN) algorithm, the detection rate of 94%.	The k-NN algorithm uses all labelled training instances as a model of the target function. During the classification phase, k-NN uses a similarity-based search strategy to determine a locally optimal hypothesis function.	Adebowale, et al. (2013)
	Naïve Bayes, the detection rate is 89%.	Bayesian classifiers provide moderate accuracy because the focus is on classifying the classes for the instances, not the exact probabilities.	Adebowale, et al. (2013)
	C4.5 gave the best detection rate of 99%.	C4.5 selects the feature of the data that most efficiently divides its set of samples into subsets, contributing to improved accuracy	Thaseen and Kumar (2013)
	SMO classifier, the detection rate is 97%.	The work also uses SVM based classifier and achieves detection rate similar to (Chen et al., 2005).	Adebowale, et al. (2013)
	Expectation Maximization (EM) clustering, the accuracy is 78%	EM forms a "soft" task of each row to various clusters in percentage to the probability of each cluster. The accuracy in this method is low as EM does not give a parameter covariance matrix for standard errors	Ahmed, et al. (2016)
ADFA-WD	Creech et al. have used Hidden Markov Model (HMM), Extreme Learning Machine (ELM) and SVM. They reported 74.3% accuracy for HMM, 98.57% accuracy for ELM and 99.64% accuracy for SVM.	The ADFA-WD is a much new data set and contains new attacks. This is why reported accuracy was not as good as for every machine learning technique when compared to the accuracy using legacy KDD98 data. SVM has been reported to produce the highest accuracy.	Creech and Hu (2014b)
ADFA-LD	100% accuracy for using ELM using original semantic feature	New semantic features are applied. Therefore, ELM, are capable to use the new semantic feature easily and quickly by including amounts of semantic phrases.	Creech and Hu (2014b)
CICIDS2017	94.5% accuracy obtained by using MLP solely, by using MLP and Payload Classifier together 95.2% accuracy rate is detected.	Feature selection is done by using Fisher Score algorithm.	Usteba, et al. (2018)
Bot-IoT	The highest accuracy from the SVM model. 98% detection rate	This SVM based method has produced good detection accuracy (Mitchell & Chen, 2015; Chen et al., 2005; Ferrari & Cribari-Neto, 2004)	Koroniotis, et al. (2018)

Table 13 Comparision of datasets (✓ = True, ✖ = False)

Dataset	Realistic Traffic	Label data	IoT traces	Zero-day attacks	Full packet captured	Year
DARPA 98	✓	✓	✖	✖	✓	1998
KDDCUP 99	✓	✓	✖	✖	✓	1999
CAIDA	✓	✖	✖	✖	✖	2007
NSL-KDD	✓	✓	✖	✖	✓	2009
ISCX 2012	✓	✓	✖	✖	✓	2012
ADFA-WD	✓	✓	✖	✓	✓	2014
ADFA-LD	✓	✓	✖	✓	✓	2014
CICIDS2017	✓	✓	✖	✓	✓	2017
Bot-IoT	✓	✓	✓	✓	✓	2018

learns the user's activities and obtains privileges which an end user could have on the computer system.

Within these broad categories, there are many different forms of computer attacks. A summary of these attacks with a brief explanation, characteristics, and examples are presented in Table 15.

IDS evasion techniques

This section discusses the techniques that a cybercriminal may use to avoid detection by IDS such as Fragmentation, Flooding, Obfuscation, and Encryption. These techniques pose a challenge for the current IDS as they circumvent existing detection methods.

Fragmentation

A packet is divided into smaller packets. The fragmented packets are then be reassembled by the recipient node at the IP layer before forwarding it to the Application layer. To examine fragmented traffic correctly, the network detector needs to assemble these fragments similarly as it was at fragmenting point. The restructuring of packets needs the detector to hold the data in memory and match the traffic against a signature database. Methods used by attackers to escape detection by hiding attacks as legitimate traffic are fragmentation overlap, overwrite, and timeouts (Ptacek & Newsham, 1998; Kolias et al., 2016). Fragmentation attack replaces information in the constituent fragmented packets with new information to generate a malicious packet.

Table 14 Detailed accuracy for C4.5 Decision tree classifier with different feature sets

Filter techniques	# of features	Accuracy	Time
Full set	41	99.55	2.76 Sec
Info Gain	13	99.64	0.84 Sec
Gain ratio	13	99.64	1.31 Sec
Chi-squared	13	99.65	0.92 Sec
Relief	13	99	0.93 Sec

Figure 8 shows the fragment overwrite. Packet Fragment 3 is generated by the attacker. The network intrusion detector must retain the state for all of the packets of the traffic which it is detecting.

The duration of time that the detector can maintain a state of traffic might be smaller than the period that the destination host can maintain a state of traffic (Xiong et al., 2017). The malware authors try to take advantage of any shortcoming in the detection method by delivering attack fragments over a long time.

Flooding

The attacker begins the attack to overwhelm the detector and this causes a failure of control mechanism. When the detector fails, all traffic would be allowed (Kolias et al., 2016). A popular method to create a flooding situation is spoofing the legitimate User Datagram Protocol (UDP) and Internet Control Message Protocol (ICMP). The traffic flooding is used to disguise the abnormal activities of the cybercriminal. Therefore, IDS would have extreme difficulty to find malicious packets in a huge amount of traffic.

Obfuscation

Obfuscation techniques can be used to evade detection, which are the techniques of concealing an attack by making the message difficult to understand (Kim et al., 2017). The terminology of obfuscation means changing the program code in a way that keeps it functionally identical with the aim to reduce detectability to any kind of static analysis or reverse engineering process and making it obscure and less readable. This obfuscation of malware enables it to evade current IDS.

Obfuscation attempts to utilize any limitations in the signature database and its capability to duplicate the way the computer host examines computer's data (Alazab & Khresiat, 2016). An effective IDS should be supporting the hexadecimal encoding format or having these hexadecimal strings in its set of attack signatures (Cova et al., 2010). Unicode/UTF-8 standard permits one character to be

Table 15 Classes of computer attacks

Types of Attack	Explanation	Example
Buffer Overflow	Attacks the buffer's boundaries and overwrites memory area.	Long URL strings are a common input. Cowan, et al. (1998)
Worm	Reproduces itself on the local host or through the network.	SQL Slammer, Mydoom, CodeRed Nimda.
Trojan	Programs appear attractive and genuine, but have malicious code embedded inside them.	Zeus, SpyEye Alazab, et al. (2013)
Denial of service (DoS)	A security event to disrupt the network services. It is started by forcing reset on the target computers. The users can no longer connect to the system because of unavailability of service.	Buffer overflow, Ping of death (PoD), TCP SYN, smurf, teardrop Zargar, et al. (2013)
Common Gateway Interface (CGI) Scripts	The attacker takes advantage of CGI scripts to create an attack by sending illegitimate inputs to the web server.	Phishing email; Aljawarneh (2016)
Traffic Flooding	Attacks the limited size of NIDS to handle huge traffic loads and to investigate for possible intrusions. If a cybercriminal can cause congestion in the networks, then NIDS will be busy in analyzing the traffic.	Denial of Service (Dos) or Distributed Denial of Service (DDoS) Zargar, et al. (2013)
Physical Attack	Aims to attack the physical mechanisms of the computer system.	Cold boot, evil maid (Pasqualetti et al., 2013).
Password Attack	Aims to break the password within a small time, and is noticed by a sequence of failures login.	A dictionary attack, Rainbow attack (Das et al., 2014).
Information Gathering	Gathers information or finds weaknesses in computers or networks by sniffing or searching.	System scan, port scan, (Bou-Harb et al., 2014).
User to Root (U2R) attack	The cybercriminal accesses as a normal user in the beginning and then upgrades to a super-user which may lead to exploitation of several vulnerabilities of the system.	Intercept packets, rainbow attack, social engineering Rootkit, load module, (Perl Ralyn, 2014).
Remote to Local (R2L) attack	The cybercriminal sends packets to a remote system by connecting to the network without having an account on the system	Warezclient, ftp write, multihop,phf, spy, warezmaster, imap (Ralyn, 2014).
Probe	Identifying the valid IP addresses by scanning the network to gather host data packets.	Sweep, portsweep (So-In et al., 2014)

symbolized in several various formats. Cybercriminals may also use double-encoded data, exponentially escalating the number of signatures required to detect the attack.

SIDS relies on signature matching to identify malware where the signatures are created by human experts by translating a malware from machine code into a symbolic language such as Unicode. However, the use of code obfuscation is very valuable for cybercriminals to avoid IDSs.

Encryption

Generally, encryption offers a number of security services, such as data confidentiality, integrity, and privacy. Malware authors employ these security attributes to escape detection and conceal attacks that may target a computer system. For example, attacks on encrypted protocols such as HyperText Transfer Protocol Secure (HTTPS) cannot be read by an IDS (Metke & Ekl, 2010). The IDS cannot match the encrypted traffic to the existing Database signatures if it doesn't interpret the encrypted traffic. Therefore, examining encrypted traffic makes it difficult for detectors to detect attacks (Butun et al., 2014). For example, packet content-based features have been applied extensively to identify malware from normal traffic, which cannot readily be applied if the packet is encrypted.

These challenges motivate investigators to use some statistical network flow features, which do not rely on packet content (Camacho et al., 2016). As a result of this, malware can potentially be identified from normal traffic.

Challenges of IDS

Although there has been a lot of research on IDSs, many essential matters remain. IDSs have to be more accurate, with the capability to detect a varied ranging of intrusions with fewer false alarms and other challenges.

Challenges of IDS for ICSs

Industrial Control Systems (ICSs) are commonly comprised of two components: Supervisory Control and Data Acquisition (SCADA) hardware which receives information from sensors and then controls the mechanical machines; and the software that enables human administrators to control the machines.

Cyber attacks on ICSs is a great challenge for the IDS due to unique architectures of ICSs as the attackers are currently focusing on ICSs. A standout amongst the recent attacks against ICSs is the Stuxnet attack, which is known as the first cyber-warfare weapon. Dissimilar to a typical attack, the primary target of Stuxnet was probably

Fig. 8 Fragment – Overwrite

the Iranian atomic program (Nourian & Madnick, 2018). Attacks that could target ICSs could be state-sponsored or they might be launched by the competitors, internals attackers with a malicious target, or even hacktivists.

The potential consequences of compromised ICS can be devastating to public health and safety, national security, and the economy. Compromised ICS systems have led to the extensive cascading power outages, dangerous toxic chemical releases, and explosions. It is therefore important to use secure ICSs for reliable, safe, and flexible performance.

It is critical to have IDS for ICSs that takes into account unique architecture, realtime operation and dynamic environment to protect the facilities from the attacks. Some critical attacks on ICSs are given below:

- In 2008, Conficker malware infected ICS systems, such as an aeroplane's internal systems. Conficker disables many security features and automatic backup settings, erases stored data and opens associations to get commands from a remote PC (Pretorius & van Niekerk, 2016).
- In 2009, a 14-year-old schoolboy hacked the city's tram system and used a homemade remote device to redirect a number of trams, injuring 12 passengers (Rege-Patwardhan, 2009).
- In 2017, WannaCry ransomware spread globally and seriously effected the National Health System, UK and prevented emergency clinic specialists from using health systems (Mohurle & Patil, 2017).

Since Microsoft no longer creates security patches for legacy systems, they can simply be attacked by new types of ransomware and zero-day malware.

Similiarly, it may not be possible to fix or update the operating systems of ICSs for legacy applications.

A robust IDS can help industries and protect them from the threat of cyber attacks. Unfortunately, current intrusion detection techniques proposed in the literature focus at the software level. A vital detection approach is needed to detect the zero-day and complex attacks at the software level as well as at hardware level without any previous knowledge. This can be done by integrating both hardware and software intrusion detection systems and extracting useful features of both HIDS and NIDS.

Challenge of IDS on intrusion evasion detection

Detecting attacks masked by evasion techniques is a challenge for both SIDS and AIDS. The ability of evasion techniques would be determined by the ability of IDS to bring back the original signature of the attacks or create new signatures to cover the modification of the attacks. Robustness of IDS to various evasion techniques still needs further investigation. For example, SIDS in regular expressions can detect the deviations from simple mutation such as manipulating space characters, but they are still useless against a number of encryption techniques.

Discussion and conclusion

Cybercriminals are targeting computer users by using sophisticated techniques as well as social engineering strategies. Some cybercriminals are becoming increasingly sophisticated and motivated. Cybercriminals have shown their capability to obscure their identities, hide their communication, distance their identities from illegal profits, and use infrastructure that is resistant to compromise. Therefore, it becomes increasingly important for computer systems to be protected using advanced intrusion detection systems which are capable of detecting modern malware. In order to design and build such IDS systems, it is necessary to have a complete overview of the strengths and limitations of contemporary IDS research.

In this paper, we have presented, in detail, a survey of intrusion detection system methodologies, types, and technologies with their advantages and limitations. Several machine learning techniques that have been proposed to detect zero-day attacks are reviewed. However, such approaches may have the problem of generating and updating the information about new attacks and yield high false alarms or poor accuracy. We summarized the results of recent research and explored the contemporary models on the performance improvement of AIDS as a solution to overcome on IDS issues.

In addition, the most popular public datasets used for IDS research have been explored and their data collection techniques, evaluation results and limitations have been discussed. As normal activities are frequently changing and may not remain effective over time, there exists a need for newer and more comprehensive datasets that contain wide-spectrum of malware activities. A new malware dataset is needed, as most of the existing machine

learning techniques are trained and evaluated on the knowledge provided by the old dataset such as DARPA/KDD99, which do not include newer malware activities. Therefore, testing is done using these dataset collected in 1999 only, because they are publicly available and no other alternative and acceptable datasets are available. While widely accepted as benchmarks, these datasets no longer represent contemporary zero-day attacks. Though ADFA dataset contains many new attacks, it is not adequate. For that reason, testing of AIDS using these datasets does not offer a real evaluation and could result in inaccurate claims for their effectiveness.

This study also examines four common evasion techniques to determine their ability to evade the recent IDSs. An effective IDS should be able to detect different kinds of attacks accurately including intrusions that incorporate evasion techniques. Developing IDSs capable of overcoming the evasion techniques remains a major challenge for this area of research.

Acknowledgments
The research is supported by the Internet Commerce Security Laboratory, Federation University Australia. The authors are grateful to the Centre for Informatics and Applied Optimization (CIAO) for their support.

Authors' contributions
AK has participated presented, in detail, a survey of intrusion detection system methodologies, types, and technologies with their advantages and limitations. Several machine learning techniques have been proposed to detect zero-day attacks are reviewed. IG, PV, and JK have gone through the article. All authors read and approved the final manuscript.

Funding
This work was carried out within the Internet Commerce Security Lab, which is funded by Westpac Banking Corporation.

Availability of data and materials
This manuscript has not been published and is not under consideration for publication elsewhere.

Competing interests
The authors declare that they have no competing interests.

Received: 29 October 2018 Accepted: 25 June 2019
Published online: 17 July 2019

References

A. Abbasi, J. Wetzels, W. Bokslag, E. Zambon, and S. Etalle, "On emulation-based network intrusion detection systems," in Research in attacks, intrusions and defenses: 17th international symposium, RAID 2014, Gothenburg, Sweden, September 17–19, 2014. Proceedings, A. Stavrou, H. Bos, and G. Portokalidis, Eds. Cham: Springer International Publishing, 2014, pp. 384–404

A. A. Aburomman and M. B. Ibne Reaz, "A novel SVM-kNN-PSO ensemble method for intrusion detection system," Appl Soft Comput, vol. 38, pp. 360–372, 2016/01/01/ 2016

Adebowale A, Idowu S, Amarachi AA (2013) Comparative study of selected data mining algorithms used for intrusion detection. International Journal of Soft Computing and Engineering (IJSCE) 3(3):237–241

Agrawal S, Agrawal J (2015) Survey on anomaly detection using data mining techniques. Procedia Computer Science 60:708–713

M. Ahmed, A. Naser Mahmood, and J. Hu, "A survey of network anomaly detection techniques," J Netw Comput Appl, vol. 60, pp. 19–31, 1// 2016

A. Alazab, J. Abawajy, M. Hobbs, R. Layton, and A. Khraisat, "Crime toolkits: the Productisation of cybercrime," in 2013 12th IEEE International conference on trust, security and privacy in computing and communications, 2013, pp. 1626–1632

A. Alazab, M. Hobbs, J. Abawajy, and M. Alazab, "Using feature selection for intrusion detection system," in 2012 International symposium on communications and information technologies (ISCIT), 2012, pp. 296–301

Alazab A, Hobbs M, Abawajy J, Khraisat A, Alazab M (2014) Using response action with intelligent intrusion detection and prevention system against web application malware. Information Management & Computer Security 22(5):431–449

Alazab A, Khresiat A (2016) New strategy for mitigating of SQL injection attack. Int J Comput Appl 154(11)

Alcaraz C (2018) Cloud-assisted dynamic resilience for cyber-physical control systems. IEEE Wirel Commun 25(1):76–82

S. A. Aljawarneh, "Emerging challenges, security issues, and Technologies in Online Banking Systems," Online Banking Security Measures and Data Protection, p. 90, 2016

C. Annachhatre, T. H. Austin, and M. Stamp, "Hidden Markov models for malware classification," Journal of Computer Virology and Hacking Techniques, vol. 11, no. 2, pp. 59–73, 2015/05/01 2015

Ara A, Louzada F, Diniz CAR (2017) Statistical monitoring of a web server for error rates: a bivariate time series copula-based modeling approach. J Appl Stat:1–14

Ashfaq RAR, Wang X-Z, Huang JZ, Abbas H, He Y-L (2017) Fuzziness based semi-supervised learning approach for intrusion detection system. Inf Sci 378:484–497

Australian. (2017, November). Australian cyber security center threat report 2017. Available: https://www.acsc.gov.au/publications/ACSC_Threat_Report_2017.pdf

S. Axelsson, "Intrusion detection systems: a survey and taxonomy," technical report 2000

Bajaj K, Arora A (2013) Dimension reduction in intrusion detection features using discriminative machine learning approach. IJCSI International Journal of Computer Science Issues 10(4):324–328

Bhuyan MH, Bhattacharyya DK, Kalita JK (2014) Network anomaly detection: methods, systems and tools. IEEE Communications Surveys & Tutorials 16(1):303–336

L. J. Blount, D. R. Tauritz, and S. A. Mulder, "Adaptive rule-based malware detection employing learning classifier systems: a proof of concept," in Computer software and applications conference workshops (COMPSACW), 2011 IEEE 35th annual, 2011, pp. 110–115: IEEE

Bou-Harb E, Debbabi M, Assi C (2014) Cyber scanning: a comprehensive survey. IEEE Communications Surveys & Tutorials 16(3):1496–1519

Breach_Level_Index. (2017, November). Data breach statistics. Available: http://breachlevelindex.com/

Breiman L (1996) Bagging predictors. Machine Learning, journal article 24(2):123–140

Buczak AL, Guven E (2016) A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Communications Surveys & Tutorials 18(2):1153–1176

Butun I, Morgera SD, Sankar R (2014) A survey of intrusion detection systems in wireless sensor networks. IEEE Communications Surveys & Tutorials 16(1):266–282

J. Camacho, A. Pérez-Villegas, P. García-Teodoro, and G. Maciá-Fernández, "PCA-based multivariate statistical network monitoring for anomaly detection," Computers & Security, vol. 59, pp. 118–137, 6// 2016

O. Can and O. K. Sahingoz, "A survey of intrusion detection systems in wireless sensor networks," in 2015 6th international conference on modeling, simulation, and applied optimization (ICMSAO), 2015, pp. 1–6

L. Chao, S. Wen, and C. Fong, "CANN: an intrusion detection system based on combining cluster centers and nearest neighbors," Knowl-Based Syst, vol. 78, pp. 13–21, 4// 2015

S. Chebrolu, A. Abraham, and J. P. Thomas, "Feature deduction and ensemble design of intrusion detection systems," Computers & Security, vol. 24, no. 4, pp. 295–307, 6// 2005

W.-H. Chen, S.-H. Hsu, and H.-P. Shen, "Application of SVM and ANN for intrusion detection," Comput Oper Res, vol. 32, no. 10, pp. 2617–2634, 2005/10/01/ 2005

M. Cova, C. Kruegel, and G. Vigna, "Detection and analysis of drive-by-download attacks and malicious JavaScript code," Presented at the Proceedings of the 19th international conference on world wide web, Raleigh, North Carolina, USA, 2010

C. Cowan *et al.*, "Stackguard: automatic adaptive detection and prevention of buffer-overflow attacks," in USENIX security symposium, 1998, vol. 98, pp. 63–78: San Antonio, TX

G. Creech, "Developing a high-accuracy cross platform host-based intrusion detection system capable of reliably detecting zero-day attacks," University of New South Wales, Canberra, Australia, 2014

Creech G, Hu J (2014a) A semantic approach to host-based intrusion detection systems using Contiguousand Discontiguous system call patterns. IEEE Trans Comput 63(4):807–819
Creech G, Hu J (2014b) A semantic approach to host-based intrusion detection systems using contiguous and Discontiguous system call patterns. IEEE Trans Comput 63(4):807–819
A. Das, J. Bonneau, M. Caesar, N. Borisov, and X. Wang, "The tangled web of password reuse," in NDSS, 2014, vol. 14, pp. 23–26
H. Debar, M. Dacier, and A. Wespi, "A revised taxonomy for intrusion-detection systems," in *Annales des télécommunications*, 2000, vol. 55, no. 7–8, pp. 361–378: Springer
Z. Du, K. Palem, A. Lingamneni, O. Temam, Y. Chen, and C. Wu, "Leveraging the error resilience of machine-learning applications for designing highly energy efficient accelerators," in 2014 19th Asia and South Pacific design automation conference (ASP-DAC), 2014, pp. 201–206
S. Dua and X. Du, Data mining and machine learning in cybersecurity. CRC press, 2016
S. Duque and M. N. b. Omar, "Using data mining algorithms for developing a model for intrusion detection system (IDS)," Procedia Computer Science vol. 61, no. Supplement C, pp. 46–51, 2015/01/01/ 2015
S. Elhag, A. Fernández, A. Bawakid, S. Alshomrani, and F. Herrera, "On the combination of genetic fuzzy systems and pairwise learning for improving detection rates on intrusion detection systems," Expert Syst Appl, vol. 42, no. 1, pp. 193–202, 1// 2015
D. M. Farid, N. Harbi, and M. Z. Rahman, "Combining naive bayes and decision tree for adaptive intrusion detection," arXiv preprint arXiv:1005.4496, 2010
S. L. P. Ferrari and F. Cribari-Neto, J Appl Stat, vol. 31, no. null, p. 799, 2004
M. Goldstein, "FastLOF: an expectation-maximization based local outlier detection algorithm," in Pattern recognition (ICPR), 2012 21st international conference on, 2012, pp. 2282–2285: IEEE
Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH (2009) The WEKA data mining software: an update. ACM SIGKDD explorations newsletter 11(1):10–18
Hendry G, Yang S (2008) Intrusion signature creation via clustering anomalies
P. Hick, E. Aben, K. Claffy, and J. Polterock, "the CAIDA DDoS attack 2007 dataset," ed, 2007
Hoque MAM, Bikas MAN (2012) An implementation of intrusion detection system using genetic algorithm. International Journal of Network Security & its Applications 4:2
L. R. Hotta, E. C. Lucas, and H. P. Palaro, Multinat Financ J, vol 12, no null, p. 205, 2008
Hu J, Yu X, Qiu D, Chen HH (2009) A simple and efficient hidden Markov model scheme for host-based anomaly intrusion detection. IEEE Netw 23(1):42–47
Hu W, Gao J, Wang Y, Wu O, Maybank S (2014) Online Adaboost-based parameterized methods for dynamic distributed network intrusion detection. IEEE Transactions on Cybernetics 44(1):66–82
N. Hubballi and V. Suryanarayanan, "False alarm minimization techniques in signature-based intrusion detection systems: a survey," Comput Commun, vol. 49, pp. 1–17, 8/1/ 2014
M. A. Ibabar, R. Alvalu, and S. S. Reddy S, "RFACDE: A Novel Ensemble Intrusion Detection System," *Procedia Computer Science*, vol. 115, pp. 226–234, 2017/01/01/ 2017
S.-Y. Ji, B.-K. Jeong, S. Choi, and D. H. Jeong, "A multi-level intrusion detection method for abnormal network behaviors," J Netw Comput Appl, vol. 62, no. Supplement C, pp. 9–17, 2016/02/01/ 2016
KDD. (1999, June). *The 1999 KDD intrusion detection*. Available: http://kdd.ics.uci.edu/databases/kddcup99/task.html
Kenkre PS, Pai A, Colaco L (2015a) Real time intrusion detection and prevention system. In: Satapathy SC, Biswal BN, Udgata SK, Mandal JK (eds) Proceedings of the 3rd International conference on Frontiers of intelligent computing: theory and applications (FICTA) 2014: volume 1. Springer International Publishing, Cham, pp 405–411
Kenkre PS, Pai A, Colaco L (2015b) Real Time Intrusion Detection and Prevention System. Springer International Publishing, Cham, pp 405–411
Khraisat A, Gondal I, Vamplew P (2018) An anomaly intrusion detection system using C5 decision tree classifier. In: Trends and applications in knowledge discovery and data mining. Springer International Publishing, Cham, pp 149–155
D. Kim *et al.*, "DynODet: detecting dynamic obfuscation in malware," in Detection of intrusions and malware, and vulnerability assessment: 14th international conference, DIMVA 2017, Bonn, Germany, July 6–7, 2017, Proceedings, M. Polychronakis and M. Meier, Eds. Cham: Springer International Publishing, 2017, pp. 97–118
G. Kim, S. Lee, and S. Kim, "A novel hybrid intrusion detection method integrating anomaly detection with misuse detection," Expert Syst Appl, vol. 41, no. 4, Part 2, pp. 1690–1700, 2014/03/01/ 2014
L. Koc, T. A. Mazzuchi, and S. Sarkani, "A network intrusion detection system based on a hidden Naive Bayes multiclass classifier," Expert Syst Appl, vol. 39, no. 18, pp. 13492–13500, 2012/12/15/ 2012
Kolias C, Kambourakis G, Stavrou A, Gritzalis S (2016) Intrusion detection in 802.11 networks: empirical evaluation of threats and a public dataset. IEEE Communications Surveys & Tutorials 18(1):184–208
N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of things for network forensic analytics: bot-IoT dataset," arXiv preprint arXiv:*1811.00701*, 2018
Kreibich C, Crowcroft J (2004) Honeycomb: creating intrusion detection signatures using honeypots. SIGCOMM Comput Commun Rev 34(1):51–56
Kshetri N, Voas J (2017) Hacking power grids: a current problem. Computer 50(12):91–95
P. Laskov, P. Düssel, C. Schäfer, and K. Rieck, "Learning intrusion detection: supervised or unsupervised?," in Image analysis and processing – ICIAP 2005: 13th international conference, Cagliari, Italy, September 6–8, 2005. Proceedings, F. Roli and S. Vitulano, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 50–57
Y. Li, J. Xia, S. Zhang, J. Yan, X. Ai, and K. Dai, "An efficient intrusion detection system based on support vector machines and gradually feature removal method," Expert Syst Appl, vol. 39, no. 1, pp. 424–430, 2012/01/01/ 2012
Liao H-J, Lin C-HR, Lin Y-C, Tung K-Y (2013b) Intrusion detection system: a comprehensive review. J Netw Comput Appl 36(1):16–24
H.-J. Liao, C.-H. Richard Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: a comprehensive review," J Netw Comput Appl, vol. 36, no. 1, pp. 16–24, 2013a/01/01/ 2013
Lin C, Lin Y-D, Lai Y-C (2011) A hybrid algorithm of backward hashing and automaton tracking for virus scanning. IEEE Trans Comput 60(4):594–601
W.-C. Lin, S.-W. Ke, and C.-F. Tsai, "CANN: an intrusion detection system based on combining cluster centers and nearest neighbors," Knowl-Based Syst, vol. 78, no. Supplement C, pp. 13–21, 2015/04/01/ 2015
Liu X, Zhu P, Zhang Y, Chen K (2015) A collaborative intrusion detection mechanism against false data injection attack in advanced metering infrastructure. IEEE Transactions on Smart Grid 6(5):2435–2443
T. F. Lunt, "Automated audit trail analysis and intrusion detection: a survey," in Proceedings of the 11th National Computer Security Conference, 1988, vol. 353. Baltimore, MD
J. Lyngdoh, M. I. Hussain, S. Majaw, and H. K. Kalita, "An intrusion detection method using artificial immune system approach," in International conference on advanced informatics for computing research, 2018, pp. 379–387: Springer
McHugh J (2000) Testing intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory. ACM Trans Inf Syst Secur 3(4):262–294
C. R. Meiners, J. Patel, E. Norige, E. Torng, and A. X. Liu, "Fast regular expression matching using small TCAMs for network intrusion detection and prevention systems," presented at the Proceedings of the 19th USENIX conference on security, Washington, DC, 2010
Meshram A, Haas C (2017) Anomaly detection in industrial networks using machine learning: a roadmap. In: Beyerer J, Niggemann O, Kühnert C (eds) Machine learning for cyber physical systems: selected papers from the international conference ML4CPS 2016. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 65–72
Metke AR, Ekl RL (2010) Security Technology for Smart Grid Networks. IEEE Transactions on Smart Grid 1(1):99–107
MIT Lincoln Laboratory. (1999, June). DARPA Intrusion Detection Data Sets. Available: https://www.ll.mit.edu/ideval/data/
Mitchell R, Chen IR (2015) Behavior rule specification-based intrusion detection for safety critical medical cyber physical systems. IEEE Transactions on Dependable and Secure Computing 12(1):16–30
C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in cloud," J Netw Comput Appl, vol. 36, no. 1, pp. 42–57, 2013/01/01/ 2013
Mohurle S, Patil M (2017) A brief study of wannacry threat: ransomware attack 2017. Int J Adv Res Comput Sci *8*(5)
S. N. Murray, B. P. Walsh, D. Kelliher, and D. T. J. O'Sullivan, "Multi-variable optimization of thermal energy efficiency retrofitting of buildings using static modelling and genetic algorithms – a case study," Build Environ, vol. 75, no. Supplement C, pp. 98–107, 2014/05/01/ 2014

Nourian A, Madnick S (2018) A systems theoretic approach to the security threats in cyber physical systems applied to Stuxnet. IEEE Transactions on Dependable and Secure Computing 15(1):2–13
Pasqualetti F, Dörfler F, Bullo F (2013) Attack detection and identification in cyber-physical systems. IEEE Trans Autom Control 58(11):2715–2729
A. Patel, M. Taghavi, K. Bakhtiyari, and J. Celestino Júnior, "An intrusion detection and prevention system in cloud computing: a systematic review," J Netw Comput Appl, vol. 36, no. 1, pp. 25–41, 2013/01/01/ 2013
Pretorius B, van Niekerk B (2016) Cyber-security for ICS/SCADA: a south African perspective. International Journal of Cyber Warfare and Terrorism (IJCWT) 6(3):1–16
T. H. Ptacek and T. N. Newsham, "Insertion, evasion, and denial of service: eluding network intrusion detection," DTIC Document 1998
W. Qingtao and S. Zhiqing, "Network anomaly detection using time series analysis," in Joint international conference on autonomic and autonomous systems and international conference on networking and services - (icas-isns'05), 2005, pp. 42–42
Quinlan JR (1986) Induction of decision trees. Mach Learn 1(1):81–106
J. R. Quinlan, C4. 5: programs for machine learning. Elsevier, 2014
Raiyn J (2014) A survey of cyber attack detection strategies. International Journal of Security and its Applications 8(1):247–256
Rath PS, Barpanda NK, Singh R, Panda S (2017) A prototype Multiview approach for reduction of false alarm rate in network intrusion detection system. Int J Comput Netw Commun Secur 5(3):49
Rege-Patwardhan A (2009) Cybercrimes against critical infrastructures: a study of online criminal organization and techniques. Crim Justice Stud 22(3):261–271
K. Riesen and H. Bunke, "IAM graph database repository for graph based pattern recognition and machine learning," in Structural, syntactic, and statistical pattern recognition: joint IAPR international workshop, SSPR & SPR 2008, Orlando, USA, December 4-6, 2008. Proceedings, N. da Vitoria Lobo et al., Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 287–297
Roesch M (1999) Snort-lightweight intrusion detection for networks. In: Proceedings of the 13th USENIX conference on system administration. Seattle, Washington, pp 229–238
Rutkowski L, Jaworski M, Pietruczuk L, Duda P (2014) Decision trees for mining data streams based on the Gaussian approximation. IEEE Trans Knowl Data Eng 26(1):108–119
Sadotra P, Sharma C (2016) A survey: intelligent intrusion detection system in computer security. Int J Comput Appl 151(3):18–22
Sadreazami H, Mohammadi A, Asif A, Plataniotis KN (2018) Distributed-graph-based statistical approach for intrusion detection in cyber-physical systems. IEEE Transactions on Signal and Information Processing over Networks 4(1):137–147
Shafi K, Abbass HA (2013) Evaluation of an adaptive genetic-based signature extraction system for network intrusion detection. Pattern Analysis and Applications, journal article 16(4):549–566
Shakshuki EM, Kang N, Sheltami TR (2013) A secure intrusion-detection system for MANETs. IEEE Trans Ind Electron 60(3):1089–1098
I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in ICISSP, 2018, pp. 108–116
Shen C, Liu C, Tan H, Wang Z, Xu D, Su X (2018) Hybrid-augmented device fingerprinting for intrusion detection in industrial control system networks. IEEE Wirel Commun 25(6):26–31
Shiravi A, Shiravi H, Tavallaee M, Ghorbani AA (2012) Toward developing a systematic approach to generate benchmark datasets for intrusion detection. computers & security 31(3):357–374
C. So-In, N. Mongkonchai, P. Aimtongkham, K. Wijitsopon, and K. Rujirakul, "An evaluation of data mining classification models for network intrusion detection," in 2014 fourth international conference on digital information and communication technology and its applications (DICTAP), 2014, pp. 90–94
P. Stavroulakis and M. Stamp, Handbook of information and communication security. Springer Science & Business Media, 2010
Studnia I, Alata E, Nicomette V, Kaâniche M, Laarouchi Y (2018) A language-based intrusion detection approach for automotive embedded networks. Int J Embed Syst 10(1):1–12
Subramanian S, Srinivasan VB, Ramasa C (2012) Study on classification algorithms for network intrusion systems. Journal of Communication and Computer 9(11):1242–1246
A. H. Sung and S. Mukkamala, "Identifying important features for intrusion detection using support vector machines and neural networks," in Symposium on Applications and the Internet, 2003, pp. 209–216
Symantec, "Internet security threat report 2017," April, 7017 2017, vol. 22 Available: https://www.symantec.com/content/dam/symantec/docs/reports/istr-22-2017-en.pdf
Tan Z, Jamdagni A, He X, Nanda P, Liu RP (2014) A system for denial-of-service attack detection based on multivariate correlation analysis. IEEE Transactions on Parallel and Distributed Systems 25(2):447–456
M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in 2009 IEEE symposium on computational intelligence for security and defense applications, 2009, pp. 1–6
S. Thaseen and C. A. Kumar, "An analysis of supervised tree based classifiers for intrusion detection system," in 2013 international conference on pattern recognition, informatics and Mobile engineering, 2013, pp. 294–299
S. Ustebay, Z. Turgut, and M. A. Aydin, "Intrusion detection system with recursive feature elimination by using random Forest and deep learning classifier," in 2018 International congress on big data, deep learning and fighting cyber terrorism (IBIGDELFT), 2018, pp. 71–76
Vigna G, Kemmerer RA (1999) NetSTAT: a network-based intrusion detection system. J Comput Secur 7:37–72
J. Viinikka, H. Debar, L. Mé, A. Lehikoinen, and M. Tarvainen, "Processing intrusion detection alert aggregates with time series modeling," Information Fusion, vol. 10, no. 4, pp. 312–324, 2009/10/01/ 2009
D. Wagner and P. Soto, "Mimicry attacks on host-based intrusion detection systems," presented at the Proceedings of the 9th ACM conference on computer and communications security, Washington, DC, USA, 2002
N. Walkinshaw, R. Taylor, and J. Derrick, "Inferring extended finite state machine models from software executions," *Empirical Software Engineering*, journal article vol. 21, no. 3, pp. 811–853, June 01 2016
G. Wang, J. Hao, J. Ma, and L. Huang, "A new approach to intrusion detection using artificial neural networks and fuzzy clustering," Expert Syst Appl, vol. 37, no. 9, pp. 6225–6232, 2010/09/01/ 2010
L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, "IoT security techniques based on machine learning," arXiv preprint arXiv:1801.06275, 2018
Xiong Q, Xu Y, Zhang B f, Wang F (2017) Overview of the evasion resilience testing Technology for Network Based Intrusion Protecting Devices. In: 2017 IEEE 18th International symposium on high assurance systems engineering (HASE), pp 146–152
X. Yang and Y. L. Tian, "EigenJoints-based action recognition using Naïve-Bayes-nearest-neighbor," in 2012 IEEE computer society conference on computer vision and pattern recognition workshops, 2012, pp. 14–19
Ye N, Emran SM, Chen Q, Vilbert S (2002) Multivariate statistical analysis of audit trails for host-based intrusion detection. IEEE Trans Comput 51(7):810–820
Y. Yuan, G. Kaklamanos, and D. Hogrefe, "A novel semi-supervised Adaboost technique for network anomaly detection," Presented at the Proceedings of the 19th ACM international conference on modeling, analysis and simulation of wireless and Mobile systems, Malta, Malta, 2016
Zargar J, Tipper (2013) A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks. IEEE Communications Surveys & Tutorials 15(4):2046–2069

Publisher's Note

Chapter 3 : Anomaly Intrusion Detection System using C5 Decision Tree Classifier

In this chapter, the accuracy of various classifiers has been compared with the C5 decision tree classifier. This chapter presents studies to decide which classifier should be used in the first stage of our proposed framework which will be discussed in detail in the next chapter. We have evaluated the performance of the C5 classifier with other classifiers in terms of accuracy, alarm rates time and memory usage.

C5's performance has been studied using the NSL-KDD benchmark dataset which contains various types of intrusions attacks. The characteristics, importance and suitability of this dataset are discussed in Chapter 2. Our results show that C5 has achieved high accuracy and low false alarms as an intrusion detection system as compared to other well-known classifiers

in the literature. Furthermore, our study in this chapter reveals that C5 consumes less time and low memory as compared to other techniques.

In this chapter, we have addressed question 1. The work presented in this chapter has been published as the following paper:

Khraisat, A., Gondal, I., & Vamplew, P.' "An Anomaly Intrusion Detection System Using C5 Decision Tree Classifier" Asia-Pacific Conference on Knowledge Discovery and Data Mining (pp. 149-155). Springer, 2018.

Table 4. Time Consuming for each classifier in seconds

Classification algorithm	Time
C5	70.6
C4.5	27.35
SVM	1423.92
Naive Bayes	1.02

5 Discussion and Conclusion

In this paper, an AIDS is proposed with the use of C5 classifier to detect both the normal and anomalous activities accurately. The aim of this approach is to identify attacks with enhanced detection accuracy and decreased false-alarm rates. We have established the robustness of our proposed techniques for intrusion detection by testing them on a NSL-KDD dataset that contains various types of intrusions. Our proposed method is evaluated on NSL-KDD dataset. Our experimental results indicate that our approach can detect malware traffic with a high detection rate of 99.82%. This demonstrates the significance of using C5 classifier in AIDS and makes the detection more effective. C5 are more powerful than C4.5, SVM and Naive Bayes because the memory usage is minimum, good speed and it also has excellent accuracy. In other words, C5 classifier provides high computational efficiency for classifier training and testing.

References

1. Bajaj, K., Arora, A.: Dimension reduction in intrusion detection features using discriminative machine learning approach. IJCSI Int. J. Comput. Sci. Issues **10**(4), 324–328 (2013)
2. Chebrolu, S., Abraham, A., Thomas, J.P.: Feature deduction and ensemble design of intrusion detection systems. Comput. Secur. **24**(4), 295–307 (2005)
3. Denning, D.E.: An intrusion-detection model. IEEE Trans. Softw. Eng. **2**, 222–232 (1987)
4. Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G., Vázquez, E.: Anomaly-based network intrusion detection: techniques, systems and challenges. Comput. Secur. **28**(1–2), 18–28 (2009)
5. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. ACM SIGKDD Explor. Newsl. **11**(1), 10–18 (2009)
6. Hearst, M.A., Dumais, S.T., Osuna, E., Platt, J., Scholkopf, B.: Support vector machines. IEEE Intell. Syst. Appl. **13**(4), 18–28 (1998)
7. Lee, W., Stolfo, S.J., Mok, K.W.: A data mining framework for building intrusion detection models. In: Proceedings of the 1999 IEEE Symposium on Security and Privacy, pp. 120–132. IEEE (1999)
8. McCallum, A., Nigam, K., et al.: A comparison of event models for Naive Bayes text classification. In: AAAI-98 Workshop on Learning for Text Categorization, vol. 752, pp. 41–48. Citeseer (1998)

In the paper, we have used k-fold cross validation technique for performance evaluation. In this technique, dataset is randomly divided into k different parts.

In the evaluation, we measured the effectiveness and efficiency of different classification algorithms that wrongly identify the percentage of the False Negative alarm Rate (FN rate) and False Positive (FP rate). Table 2 provide the overall results of our experiments, which indicate that C5 classifiers are best at classifying the intrusions; it has successfully distinguished between normal and anomalous activity with minimum number of false alarm.

Table 1. Confusion matrix for an anomaly detection system

	Predicted class		
Actual class		Normal	Attack
	Normal	True Negative(TN)	False Positive(FP)
	Attack	False Negative(FN)	True Positive(TP)

Table 2. Confusion matrix for different classification algorithms

Classification algorithm	C5		C4.5		SVM		Naive Bayes	
Classified as	a	b	a	b	a	b	a	b
a = normal	67249	94	67200	143	66370	973	63060	4283
b = anomaly	121	58509	132	58498	2296	56334	7832	50798

Table 3 showed the accuracy for all the classifiers and shows that C5 classifiers have outperformed other classifiers in the study. C5 classifier has the highest accuracy of 99.82% which is followed by C4.5, SVM and Naive Bayes respectively. The number of false alarms, accuracy and time of building IDSs should be considered for IDS evaluation. Although C5 decision tree classifier wasn't faster classifier as shown in Table 4 C5 is the best in term of the accuracy and low false alarm. Naive Bayes is the fastest, but has the lowest accuracy by a substantial margin. The time that takes for generating the ruleset in C5 is 2.06, while the time that takes for generating the ruleset in c4.5 is 29.98, which is slower than C5. The reasons for this, in C5 the rules are generated separately.

Table 3. Accuracy in detection by using different algorithms

Classification algorithm	Accuracy
C5	99.82%
C4.5	99.78%
SVM	97.40%
Naive Bayes	90.38%

stage, the classifier is trained to detect the attacks. In the detection phase, data mining techniques are used to generate rule sets that are considered as abnormal activities and used by the classification algorithm already learned to classify the item set as an attack. After testing stage, we compute the accuracy rate, and other performance statistics to distinguish which classifier has predicted successfully.

4 Experimental Analysis

WEKA platform is used [5] to study J48, Naive Bayes and SVM. A commercial system from RuleQuest Research is used for C5 algorithm's [10]. NSL-KDD dataset is used [12]. We compared four different classifiers: C4.5, SVM, Naive Bayes and C5 to evaluate the performance of classification techniques.

4.1 Dataset Description

NSL-KDD data set has been used to overcome KDD cup99 dataset problem. A statistical analysis have been done on KDD cup99 dataset and found issues which have affected the ability to evaluate anomaly detection approaches. It is revealed the main issue is that KDD cup99 dataset has a huge number of redundant records [17]. NSL-KDD is considered as benchmark dataset in evaluating the performance of intrusion detection techniques [12].

The amount of training and testing records in NSL-KDD dataset are significant so the performance of classifiers can be evaluated reliably. The dataset has 125,973 records, where 67,343 are normal cases and 58,630 are anomalies. The dataset contains 22 types of attack, and 41 features.

4.2 Model Evaluation and Results

Our model will be evaluated based on the following standard performance measures:

- True positive (TP): Number of cases correctly predicted as anomaly. True negative (TN): Number of cases correctly predicted as normal.
- False positive (FP): Number of wrongly predicted as anomalies, when the classifier labels normal user activity as an anomaly. False negative (FN): Number of wrongly predicted as normal cases, when a detector fails to identify the anomaly.

Table 1 shows the confusion matrix for a two-class classifier. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class.

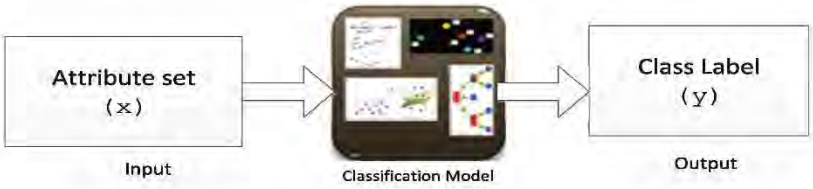


Fig. 1. Classification techniques

- Easy to understand the tree, as the large decision tree can be viewed as a set of rules. C5 can provide the knowledge for the noisy or missing data.
- Addresses over fitting and error pruning issues. Winnowing technique in C5 classifier can predict which attributes are relevant and which are not in the classification. It is useful while dealing with big datasets.

In machine learning, **Naive Bayes** classifiers are a family of least complex probabilistic classifiers based on using Bayes' theorem with robust (naive) independence assumptions between the attributes [8]. It is simple to build, with no complex iterative parameter estimation which makes it suitable for very large datasets. **SVM Model** is a demonstration of the examples as points in space, mapped so that the examples of the separate categories are split by a clear space that is as varied as possible. New examples are then matched into that similar space and predicted to belong to a group based on which side of the gap they belong to [6].

3.2 Framework of Intrusion Detection System

Our purpose is to examine different machine learning techniques that could minimize both the number of false negatives and false positives and to understand which techniques might provide the best accuracy for each category of attack patterns. Different classification algorithms have been applied and evaluated. Figure 2 shows a conceptual framework of our IDS.

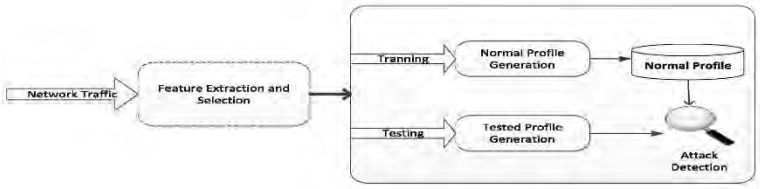


Fig. 2. Overall approach

Collected data is a network traffic, which is used to do feature extraction and selection. In the training phase, a normal profile is developed and in this

The rest of the paper is organized as follows. Related worked is discussed in Sect. 2. The IDS model with the dataset details is discussed in Sect. 3. Conceptual framework of our IDS model is proposed in Sect. 4. In Sect. 5, the experiment details are given and evaluation results are presented and discussed. Finally, we conclude the paper in Sect. 5.

2 Related Works

Some prior research has examined the use of different techniques to build AIDSs. Chebrolu et al. examined the performance of two feature selection algorithms involving Bayesian networks (BN) and Classification Regression Trees (CRC), and combined methods [2]. Karan et al. proposed a technique for feature selection using a combination of feature selection algorithms such as Information Gain (IG) and Correlation Attribute evaluation then they tested the performance of the selected feature by applying different classification algorithms such as C4.5, Naive Bayes, NB-Tree and Multi-Layer Perceptron [1]. Subramanian et al. propose classifying NSL-KDD dataset using decision tree algorithms to construct a model with respect to their metric data and studying the performance of decision tree algorithms [11].

C5 algorithm's performance is explored very well in a different domain such as modelling landslide susceptibility. Miner et al. used data mining techniques in the topic of landslide susceptibility mapping. They used C5 classifier to handle the complete dataset and address some limitations of WEKA, one of the best results were obtained from C5 applications [9].

3 IDS Model

A prediction model has two main components which are training phase and testing phase. In the training phase the normal profile is created, and in the testing phase the user actions are verified against the corresponding profile. We classify each of the collected data records obtained from the feature phase as normal or an anomaly. In the testing stage, we examine each model.

3.1 Classification

A classification technique is a systematic approach for building classification models from an input data set. Classification is the task of mapping a data item into one of a number of predefined classes [7]. Figure 1 shows a general approach for applying classification techniques.

Decision Trees. are considered one of the most popular classification techniques. Quinlan (1993) has advocated for the decision tree approach and the latest implementation of Quinlan's model is C5 [10]. In this paper we will apply C5 classifier, the algorithm has many advantages like:

An Anomaly Intrusion Detection System Using C5 Decision Tree Classifier

Ansam Khraisat(✉), Iqbal Gondal, and Peter Vamplew

Internet Commerce Security Laboratory (ICSL), Federation University Australia, Ballarat, Australia
{a.khraisa,iqbal.gondal,p.vamplew}@federation.edu.au

Abstract. Due to increase in intrusion activities over internet, many intrusion detection systems are proposed to detect abnormal activities, but most of these detection systems suffer a common problem which is producing a high number of alerts and a huge number of false positives. As a result, normal activities could be classified as intrusion activities. This paper examines different data mining techniques that could minimize both the number of false negatives and false positives. C5 classifier's effectiveness is examined and compared with other classifiers. Results should that false negatives are reduced and intrusion detection has been improved significantly. A consequence of minimizing the false positives has resulted in reduction in the amount of the false alerts as well. In this study, multiple classifiers have been compared with C5 decision tree classifier using NSL_KDD dataset and results have shown that C5 has achieved high accuracy and low false alarms as an intrusion detection system.

Keywords: Malware · Intrusion detection system · NSL_KDD
Anomaly detection

1 Introduction

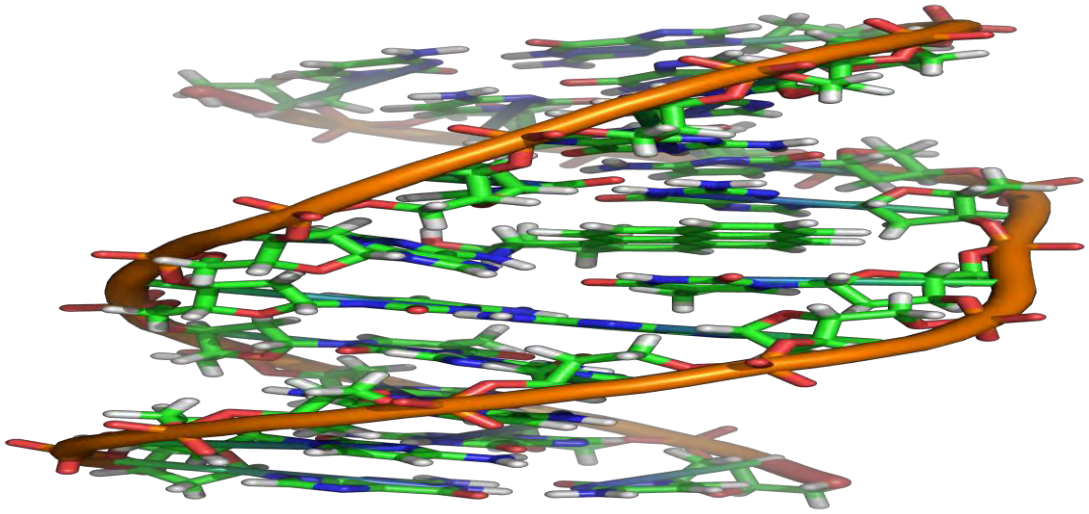
Anomaly Intrusion Detection Systems (AIDS) [3] have attracted the interest of many researchers due to their potential to detect a zero-day attack. AIDS recognizes abnormal user behavior on a computer system. The assumption for this technique is that attacker activity differs from normal user activity. AIDS [4] creates a behavior profile of normal user's activity by using selected features and machine learning approaches. It then examines the behaviors of new data with the predefined normal behavior profile and tries to identify abnormalities. Those behaviors of users which are unusual are identified as potential attacks.

In this research work, a range of data mining techniques including SVM, Naive Bayes, C4.5 implemented in the WEKA package (developed by the University of Waikato, New Zealand) as well as the C5 algorithm [10] were applied on the NSL-KDD dataset.

M. Ganji et al. (Eds.): PAKDD 2018, LNAI 11154, pp. 149–155, 2018.
https://doi.org/10.1007/978-3-030-04503-6_14

9. Miner, A., Vamplew, P., Windle, D., Flentje, P., Warner, P.: A comparative study of various data mining techniques as applied to the modeling of landslide susceptibility on the Bellarine Peninsula, Victoria, Australia (2010)
10. Quinlan, R.: Data mining tools See5 and C5. 0 (2004)
11. Subramanian, S., Srinivasan, V.B., Ramasa, C.: Study on classification algorithms for network intrusion systems. J. Commun. Comput. **9**(11), 1242–1246 (2012)
12. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the KDD CUP 99 data set. In: IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009, pp. 1–6. IEEE (2009)

Chapter 4 : Hybrid Intrusion Detection System Based on the Stacking Ensemble of C5 Decision Tree Classifier and One-Class Support Vector Machin



Traditional IDSs suffer from limitations such as they can be penetrated easily by attackers, incapability to differentiate new malicious requests, hard to update, low accuracy and high false alarms. AIDS has its limitations and SIDS can only identify intrusions which are known previously. Therefore, the intrusion systems must be updated with newly detected malware signatures. In this chapter, a Hybrid IDS (HIDS) is presented to overcome the individual drawbacks of SIDS and AIDS.

Machine learning based IDSs classify attacks based on the input data sources. In general, there are two types of IDS technologies, namely Host-based IDS (HIDS) and Network-based IDS (NIDS). NIDS monitors the network traffic to detect external malicious attacks that could be

initiated from the external threats. NIDS has a poor performance to inspect all data in a high bandwidth network because of high-speed communication networks (Bhuyan, Bhattacharyya, & Kalita, 2014). NIDS deployed at a number of positions within a particular network topology, together with HIDS and firewalls, can give a concrete, resilient, several phased protection against both external and insider attack.

In this chapter, an ensemble of C5 and One-Class Support Vector Machine classifiers is used in a novel way to build the proposed HIDS and it has been evaluated using know benchmark data sets. HIDS makes use of the advantages of both Signature intrusion detection systems and Anomaly-based Intrusion Detection System techniques. We have used two different datasets to asses our method and compared it with other frameworks. Our results showed that our proposed HIDS achieved the highest accuracy with low alarm rates. This chapter has addressed Research Question#2, outlined in Chapter 1. A Journal paper has been published: Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J.; Alazab, A. Hybrid Intrusion Detection System Based on the Stacking Ensemble of C5 Decision Tree Classifier and One-Class Support Vector Machine. Electronics, 9, 173, 2020.

classifiers to obtain the accuracy. Table 18 shows the accuracy rates for different machine learning techniques, specifically C4.5, Naïve Bayes, Random Forest, multi-layer perception, SVM, CART, and KNN on the NSL-KDD dataset. The results show that our proposed technique, which combined the two stages, achieved the best performance, reaching an accuracy of 83.24%.

Table 18. Performance comparison between different classifiers and proposed algorithm on KDDTest+.

Machine Learning Techniques	NSL-KDD Accuracy
C4.5 [26]	81%
Naïve Bayes	76.56%
Random Forest	80.67%
Multi-layer perception	77.41%
SVM	69.52%
CART	80.3%
KNN	79.4%
Proposed Technique	83.24%

5. Conclusions

To create attacks in high volume, cybercriminals began using new techniques, like polymorphism, to change the signature each time and to generate new attacks. Efficient IDSs should be able to detect known and zero-day attacks reliably. In this paper, a novel framework is developed to build an intelligent IDS that overcomes the weaknesses of current IDSs, which means including detection methods for both known and unknown threats. The main contribution of our framework is the integration of the signature and anomaly intrusion detection systems, which takes advantage of the respective strengths of SIDS and AIDS. In the proposed IDS, signature-based IDS is applied to identify previously known intrusions, while an anomaly-based IDS is applied to detect unknown zero-day intrusions. We have effectively created signatures from anomaly IDSs to identify different intrusions to add in signature databases. Additionally, the advantage of the proposed IDS is not only the higher detection rate, but also the enhanced scalability, such as when new intrusions are stored to the signature intrusion database. We used the C5 classifiers to create an intrusion signature, which is capable of generating a rule pattern more rapidly and can detect the intrusions with fewer numbers of signatures. We have shown that an ensemble of the C5 classifier (signature) and one-class SVM (anomaly) can result in a better detection rate when compared with other machine learning techniques in terms of the detection rate, false alarms, true negative, false positive, false negative, recall, precision, specificity, sensitivity, and F-Measure. Compared to the single algorithms, combining multiple algorithms has given much better results. Our Hybrid IDS has shown superior results, as compared to existing techniques. Our future research will be focused on the way in which to apply this technique in order to improve the accuracy of IDSs in detecting different attacks.

Author Contributions: A.K. is the main author of the current paper. A.K. contributed to the development of the ideas, design of the study, theory, result analysis, and article writing. A.K. also designed the experiments and then performed the experiments. I.G., P.V., J.K. and A.A. undertook the revision works of the paper. All authors have read and agreed to the published version of the manuscript.

Acknowledgments: This work was done in the Internet Commerce Security Lab (ICSL) FedUni. Westpac Bank, ACSC, Victorian Government and IBM are partners in the ICSL.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sun, X.; Dai, J.; Liu, P.; Singhal, A.; Yen, J. Using Bayesian Networks for Probabilistic Identification of Zero-Day Attack Paths. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 2506–2521. [CrossRef]
2. Alazab, M.; Tang, M. *Deep Learning Applications for Cyber Security*; Springer: Berlin/Heidelberg, Germany, 2019.

The accuracy of Stage 3 with the use of NSL-KDD Test+ and ADFA datasets is shown in Tables 15 and 16, respectively.

Table 15. Detailed accuracy at Stage 3 on the KDDTest+.

Class	TP Rate	FP Rate	F-Measure
Normal	0.972	0.273	0.833
Malware	0.727	0.028	0.832

Table 16. Detailed accuracy at Stage 3 on the ADFA dataset.

Class	TP Rate	FP Rate	F-Measure
Normal	0.976	0.029	0.971
Malware	0.971	0.024	0.976

As revealed in Figure 6, the detection accuracy of the intrusion is 81.5% with the use of NSL-KDD Test+ dataset and 97.3% for the ADFA dataset at Stage one. Meanwhile, the detection accuracy of the intrusion is 72.2% with NSL-KDD Test+ dataset and 76.4% for ADFA dataset at Stage two. At Stage 3, the accuracy rates are improved to 83.2% and 97.4%, respectively, for the both datasets. Results show that our suggested framework yields a superior detection rate and a lower false alarm rate, as compared with the single stage method.

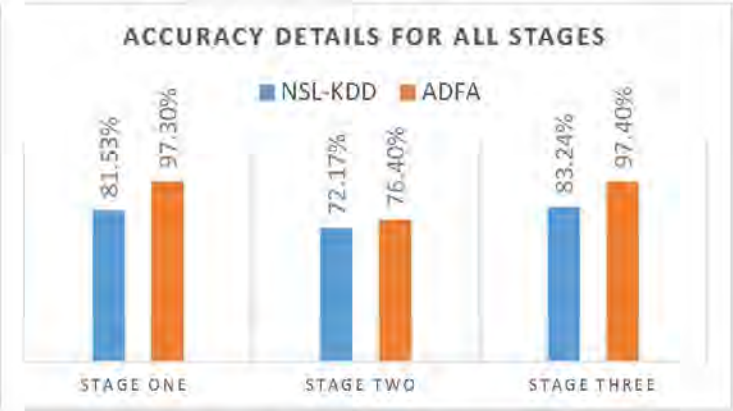


Figure 6. Accuracy details for all stages.

To further analyze performance, our approach is compared with different approaches reported in the literature in terms of the overall accuracy. Table 17 shows this comparison for the NSL-KDD dataset. Results show that our proposed model, which combines two stages, outperforms other approaches.

Table 17. Accuracy comparison of the proposed model on NSL-KDD (Test+ dataset).

Research	Accuracy Rate on KDDTest+
Abbasi, et al. [31]	77.38%
Panda, et al. [32]	81.47%
Abbasi, et al. [31]	79.66%
Proposed Technique	83.24%

According to the results shown in Figure 6, the accuracy obtained by our proposed algorithm on the KDDTest+ and ADFA datasets are supreme, as compared to the accuracy obtained by different

The detailed accuracy for C5 classifier with the use of NSL-KDD Test+ is shown in Table 9 and on ADFA dataset shown in Table 10.

Table 9. Detailed accuracy of C5 decision tree with KDDTest+.

Class	TP Rate	FP Rate	F-Measure
Normal	0.972	0.311	0.815
Malware	0.689	0.028	0.805

Table 10. Detailed accuracy of C5 decision tree on ADFA dataset.

Class	TP Rate	FP Rate	F-Measure
Normal	0.975	0.028	0.971
Malware	0.972	0.025	0.976

4.3.2. Stage Two: AIDs Results

One-class SVM with an RBF kernel was applied using LIBSVM. Confusion matrix results are shown in Table 11 for both datasets: NSL-KDD Test+ and ADFA.

Table 11. Performance of the One-Class Support Vector Machine.

Dataset	NSL-KDD		ADFA	
Class	Normal	Malware	Normal	Malware
Normal	9500	211	33,079	3921
Malware	6064	6769	15,554	29,778

The detailed analyses of the accuracy of the One-Class SVM classifier on NSL-KDD Test+ and ADFA datasets are highlighted in Tables 12 and 13, respectively. For AIDS, the detection accuracy is 72.17% with the use of NSL-KDD Test+ dataset and 76.4% for the ADFA dataset.

Table 12. Performance of the One-class SVM on the NSL-KDD Test+.

Class	TP Rate	FP Rate	F-Measure
Normal	0.978	0.473	0.752
Malware	0.527	0.022	0.683

Table 13. Performance of the One-class SVM on the ADFA dataset.

Class	TP Rate	FP Rate	F-Measure
Normal	0.894	0.343	0.773
Malware	0.657	0.106	0.754

4.3.3. Stage Three: Combination of the Two Stages

The Confusion matrix of the mixture of both classifiers in Stage three is shown in Table 14 for both NSL-KDD Test+ and ADFA.

Table 14. Confusion matrix results for Stage 3.

Dataset	NSL-KDD		ADFA	
Class	Normal	Malware	Normal	Malware
Normal	9500	211	36,114	886
Malware	6064	6769	1305	44,027

- F-measure (FM): The *FM* is the mean of the precision and recall. F-Measure is favored when only one accuracy metric is needed as an evaluation measurement:

$$\text{F} - \text{measure} = \frac{2 * Recall * Precision}{Recall + Precision} \quad (12)$$

4.3. Experimental Results

The effectiveness of our proposed model is evaluated with other machine learning techniques that use the same datasets mentioned earlier.

In the first instance, for selected classification techniques, the dataset is divided into training and testing subsets for assessment purposes. For the NSL-KDD dataset, the process of training and testing the different stages is outlined in Table 7.

Table 7. Steps followed for training and testing the NSL-KDD dataset.

Steps	Details
Step 1	For NSL-KDD, we divided the data for training and testing. KDDTrain+, which is generated from KDD train set, is used as training set and KDDTest+ is used as testing set. The generated datasets, KDDTrain+ and KDDTest+, contain 125,973 and 22,544 records, respectively.
Step 2	Trained the SIDS using KDDTrain+; and then tested it using KDDTest+.
Step 3	Finally, tested the SIDS using KDDTest+. If SIDS classifies a sample as malware, then labeled it as malware. But, if SIDS classifies as normal, then those samples are passed to AIDS for further analysis
Step 4	Trained AIDS with KDDTrain++ and tested the samples labelled as normal by SIDS

With the ADFA dataset, we used the widely adopted 10-fold cross-validation scheme for training and testing purpose. In a 10-fold cross-validation, the dataset is split into 10 approximately equal sized non-overlapping subsets. Nine subsets are used for building the classifier in the training stage, while the remaining one subset is used to test the model. The test set is employed to estimate the IDS accuracy. This process is repeated 10 times, each time using a separate fold for testing. In this way, the whole dataset goes through the testing phase in turns, with each sample being tested once. The overall accuracy estimate is the mean of 10 rounds.

The proposed IDS accuracy has been evaluated for all stages; four statistics evaluation measurement have been computed: True positive rate, F-measure, false positive rate, and accuracy.

4.3.1. Stage One: SIDS Results

This experiment was conducted by using NSL-KDD Test+ dataset and ADFA dataset. To evaluate the performance of the proposed technique, the Confusion matrix was used. The Confusion matrix results for the C5 classifier in stage one is shown in Table 8 for both NSL-KDD Test+ and ADFA.

Table 8. Confusion matrix results with the use C5 classifier on KDDTest+ and ADFA dataset.

Dataset	NSL_KDD		ADFA	
Class	**Normal**	**Malware**	**Normal**	**Malware**
Normal	9488	263	36,065	935
Malware	3900	8933	1250	44,082

4.2. Evaluation Metrics

Table 6 shows the confusion matrix for a two-class classifier that would commonly be used in an IDS. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class.

Table 6. Confusion matrix of an IDS for evaluation purpose.

Actual Class	Predicted Class		
	Normal	Intrusion	
Normal	True negative (TN)	False Positive (FP)	
Intrusion	False Negative (FN)	True positive (TP)	

Usually, the evaluation of the IDS is assessed based on the Confusion matrix measurement as follows:

- True Positive Rate (TPR): It measures the quantitative relation between the attacks and the overall attacks number. *TPR* is 1 when all intrusions are correctly identified, and that is extremely rare for an IDS. *TPR* is also called the Detection Rate (DR) and is defined as:

$$TPR = \frac{TP}{TP + FN} \tag{6}$$

- False Positive Rate (FPR): It measures the quantitative relation between the normal cases that are detected as attacks and the overall number of normal cases. *FPR* is calculated as:

$$FPR = \frac{FP}{FP + TN} \tag{7}$$

- False Negative Rate (FNR): FNR shows that the intrusion detection system could not classify the intrusion and has classified it as normal. The *FNR* is calculated as:

$$FNR = \frac{FN}{FN + TP} \tag{8}$$

- Classification rate (CR) or Accuracy: The *CR* is the total accuracy of the IDS in classifying both normal and intrusion attacks and is calculated as:

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

- Precision (P): *P* is the percentage of total true positives (*TP*) instances divided by total number of true positives (*TP*) and false positives (*FP*) instances:

$$Precision = \frac{TP}{TP + FP} \times 100\% \tag{10}$$

- Recall (R): Refers to the percentage of total relevant results correctly classified, true positives (*TP*), divided by the total true positives and false negatives (*FN*) instances:

$$Recall = \frac{TP}{TP + FN} \times 100\% \tag{11}$$

restricted flexibility to learn the true behavior of normal and abnormal activities from the dataset. They proposed a new data set, NSL-KDD, which contains marked records of the overall KDD data set and does not encounter the previously mentioned inadequacies. Table 5 shows the list of attacks presented in NSL-KDD dataset. This dataset has been widely used as a public dataset for the validation of IDS.

Table 5. List of attacks presented in NSL-KDD dataset.

Attack Name	Description
Denial of Service (DoS) attack	Make a computer service unavailable to its legitimate users by overwhelming the computer system. Attacker could send high volume of packet to target computer so normal traffic cannot be handled.
Buffer overflow	Occurs when more data is put into a fixed-length buffer than the buffer can handle.
FTP writes	Add files to ftp directory and eventually gain access to the computer system
Guessing password	Aims to find the correct password by trying systematic guessing of passwords techniques such as dictionary password or brute force attack
IMAP	Internet Message Access Protocol (IMPAP) permits cybercriminal to mount brute force attacks without being locked out or triggering an alert by intrusion detection system
IP Sweep	Occurs when cybercriminal sends Internet Control Message Protocol (ICMP) echo requests (pings) to several computer addresses. If a computer receiver host response, the response discloses the victim's IP address to the cybercriminal.
LAND (Local Area Network Denial)	Occurs when cybercriminal submits TCP SYN spoofed datagram where sender and receiver IPs and ports are be configured the same. When the victim computer attempts to reply, it enters into a loop, repetitively sending responses to itself which ultimately sources the victim machine to damage.
load module	Attacker aims to load two dynamically loadable kernel drivers into operating system to gain root access on the target computer.
Multi hop routing	Occurs when cybercriminal change routing messages to cause wrong routing updates which can ultimately lead to network failure.
Neptune (SYN flood)	Occurs when cybercriminal generates a SYN flood attack against a network host by sending session establishment packets via a fake source IP address.
Nmap (port scanning)	Occurs when cybercriminal sends packets to target computer to discover what services are running on the target computer
Perl scripting	Occurs when cybercriminal inject malicious code in a target computer
phf script	A script named "phf" which is installed by default in the web server directory could be used to send illegitimate packets to the web server.
Ping of Death (PoD)	Cybercriminal tries to freeze the victim's computer by sending abnormal or large packets using a simple ping command.
Port sweep	In Ports weep attack, various hosts are scanned on a particular listening port. For instance, if the cybercriminal would like to detect all the web servers which are using ports 80 and 443.
Rootkit	This attack is used to obtain root or administrator access
Satan	Satan is a tool designed to probe a target computer system
Smurf	It is a distributed denial-of-service attack in which huge amounts of packets with the intended computer target are tricked as source IP to broadcast to a victim computer network.
Spy	This attack enables to collect data about a target computer without their knowledge
Teardrop	A *teardrop attack* is DoS *attack* which sends fragmented packets to a target computer
Warezmaster	Cybercriminal access on a computer server using guest account. The cybercriminal creates a hidden file and uploads "warez" (malicious file) onto webserver. Victim user can then later download these files.
Warezclient	Warezclient attack could be executed by victim during an FTP connection after warezmaster attack.
Normal	Not attack
Unknown	Unknown attack

Table 2. ADFA-LD (ADFA Linux) dataset features.

Seq.	Name of the Category	Description
1	Flow features	It represents the v features communication between the machine to the computer server or server to-client such as source of IP address, source port number and destination IP address.
2	Basic features	It contains the features that describe the communications of protocols such as the connection duration, source to destination transaction bytes and packet count
3	Content features	based on data from packet contents.
4	Time features	It comprises the attribute of time such as the total time taken to send the first packet to the destination as well as the time taken to get the response packet.
5	Generated features	Related with protocols service.
6	Connection features	Built based on the chronological order of the last time feature
7	Labelled Features	It could be normal or intrusion

Table 3. Number of system calls traces in various categories of AFDA-LD and AFDA-WD datasets.

	ADFA-LD		ADFA-WD	
Dataset	Traces	System Calls	Traces	System Calls
Training data	833	308,077	355	13,504,419
Validation data	4372	2,122,085	1827	117,918,735
Attack data	746	317,388	5542	74,202,804
Total	5951	2,747,550	7724	205,625,958

Table 4. ADFA-LD attack classes.

Attack	Payload	Description
Hydra-FTP	Password brute force	This type of attack comprises of an attacker trying several passwords or passphrases with the hope of eventually guessing File Transfer Protocol (FTP) password correctly
Hydra-SSH	Password brute force	For guessing Secure Shell (SHH) password.
Add user	Add new super user	Add user command creates a new user
Java Meterpreter	Java based Meterpreter	This is an attack payload that offers a communication shell to the cybercriminal from which they can explore the target computer and execute malicious activities.
Meterpreter	Linux Meterpreter Payload	Client-side poisoned executable
Web shell	C100 Web shell	Web shell is a script running on a server that allows remote access and provides a set of functions to execute or a command-line interfaces on the system that hosts the Web server for the use of cybercriminal.

4.1.2. NSL-KDD Dataset

The KDD 1999 data set has been examined by Tavallaee et al. [30] and found a number of weaknesses. A few problems were noted, relating to synthesizing the network and attack data (after sampling the actual traffic) because of privacy issue, an unidentified packet loss caused by network traffic, and unclear attack definitions. Tavallaee et al. also completed statistical evaluations and revealed a high number of redundant records resulting in bias in the dataset. Hence, high bias can cause IDS to be inaccurate in terms of high false alarms. Therefore, machine learning techniques have

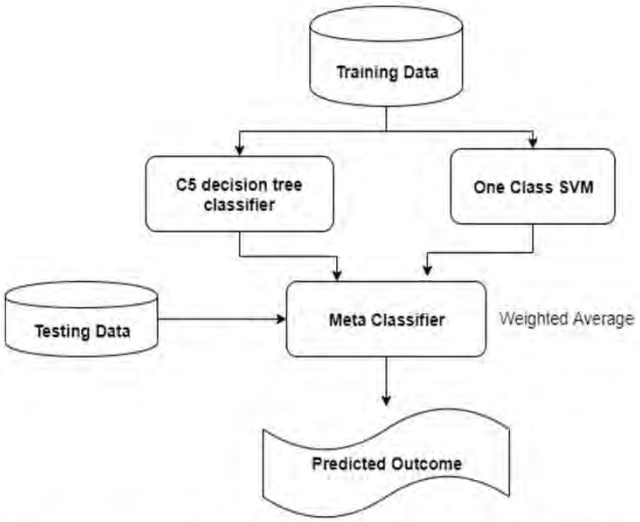


Figure 5. Hybrid Intrusion Detection System based on ensemble of C5 and One-Class SVM.

4. Model Evaluation

The NSL-KDD and ADFA datasets are used to evaluate the proposed hybrid IDS. The experiments have been performed using C5 and LIBSVM, library for Support Vector Machines, implementation of the support vector machine with default parameters. Details of the datasets are presented below.

4.1. Datasets

4.1.1. ADFA Dataset

Creech and Hu [29] built the ADFA Linux (ADFA-LD) cyber security benchmarks datasets for assessment of IDSs. Ubuntu Linux version 11.04 was used as the operating system to collect this dataset. Ubuntu Linux configuration offers several functions including the sharing of files, a database movement system, network settings, and a web server.

File transfer protocol, secure web server, secure shell protocol, and MySQL database are activated based on default ports. Personal Home Page (PHP) was used as a server scripting language and to make the Web pages dynamic and interactive. Apache was installed to enable web-based services. Apache acted as a middleman between the server and user computer.

The ADFA-LD is freely accessible on the Internet and can be found in Reference [29]. Table 2 shows ADFA-LD features and their types.

The ADFA Windows Dataset (ADFA-WD) offers a modern Windows dataset for HIDS evaluation. Table 3 presents various system calls in AFDA-LD and AFDA-WD. Table 4 defines each attack in details in the ADFA-LD dataset.

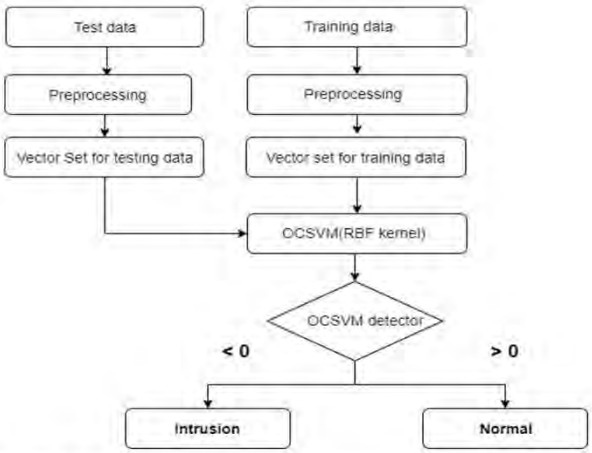


Figure 4. Anomaly-based Intrustion Detection System (AIDS) based one-class svm.

3.3. Stage 3: HIDS Based Stacking Ensemble of C5 and One-Class SVM

SIDS and AIDS have complementary strengths and weaknesses, so we developed a hybrid method using an ensemble of both techniques. In machine learning, ensemble techniques use many learning algorithms to accurately predict the outcome. In other words, different classifier models are trained on the same target and then their results are combined. In the first stage, a set of base level classifiers C1, C2, ... , Cn are created. In the second stage, a meta-level classifier is built by uniting the base level classifier.

While many ensemble methods have been proposed in the literature, it is a difficult task to find a suitable ensemble configuration to detect zero-day attacks. There are three popular ensemble methods: Bootstrap aggregating, boosting, and stacking. Bootstrap aggregating, known as bagging, employs the simplest way of combining predictions that belong to the same class. For example, if we had four bagged decision trees that made the following class predictions for an input sample: Malware, normal, malware and malware, we would take the most frequent class and predict malware. Boosting steadily creates an ensemble via preparing each new model, utilizing the misclassified training instance that past models misclassified. An example of boosting is the AdaBoost algorithm, which uses a boosting technique. Stacking, also known as stacked generalization, is a technique that combines the other models' predictions.

In stacking, predictions of base learners (stage one) are used as input for the meta-learner (stage two). Stacking is a parallel integration of classifiers in which all the classifiers are implemen-ed parallel to eatchother and learning takes place at the meta-level. In this paper, two models, namely C5 and OCSVM, are built and then the predictions of the primary models are combined, as shown in Figure 5.

The focus is on how to enhance IDS accuracy by employing the stacking approach. Meanwhile, the current conventional data mining approaches focus on how to enhance the performance of a single model. Our work focuses on how different classifiers can be combined to improve the overall performance of IDS. It was also observed that this approach yields better accuracy in the area of intrusion detection, as illustrated in the following section.

where w is the normal vector and b is a bias term. The OCSVM adjusts the hyperplane to find a linear classifier by optimising the rule f. This classification rule can be used to assign a label to a test example x. x is classified as an intrusion if the $f(x)$ result is less than zero, or else it is classified as normal. As presented in Figure 3, the result of $f(x)$ can clarify the classification condition: Positive is considered to in the normal class, negative is in the intrusion class.

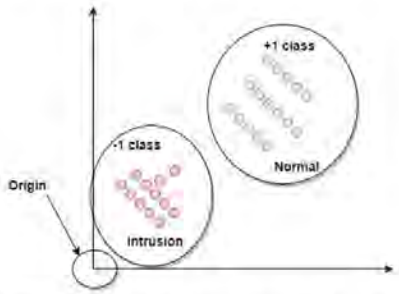


Figure 3. One-class SVM (Support Vector Machine) classifier based on relaxation parameters.

The one class SVM intrusion detection system can be expressed as mapping the data into a feature vector H using a suitable kernel function, and then attempts to isolate the mapped vectors from the origin with a determined margin (see Figure 3).

$$f(x) = \begin{cases} +1, & if\ x \in Normal \\ -1, & if\ x \in Intrusion \end{cases} \quad (2)$$

In stage 2 of the one class SVM, let $x1$, $x2$, …, xl be the training examples belonging to one class X, where X is a compact subset of R^N. Let Φ: X → H be a kernel map that transforms the training examples to another space. Then, to separate the data set from the origin, one needs to solve the following quadratic programming problem:

$$min\frac{1}{2}\|w\|^2 + \frac{1}{Vl}\sum_{i=1}^{l}\xi_i - p \quad (3)$$

which is subject to

$$(w \times \Phi(xi)) \geq \rho - \xi i\, i = 1, 2, \ldots, l\, \xi i \geq 0 \quad (4)$$

If w and ρ are solved in this problem, then the decision function

$$f(x) = sign((w \times \Phi(x)) - \rho) \quad (5)$$

will be normal for most instances xi comprised in the training data set.

The basic idea of OCSVM training is to build the OCSVM intrusion detection system that is able to detect the intrusions. Initially, both the training set and the test set are pre-processed to acquire the vector sets based on their unlikely data types. The training vector set at that point is employed to train the OCSVM intrusion detection system and the OCSVM detector is then used on the test vector set. If the return value result for the intrusion system OCSVM function $f(x)$ is less than zero, an intrusion is detected, otherwise it is normal. This entire stage technique is shown in a flowchart in Figure 4.

If no match is found, it will go to AIDS, which is the second stage of the framework, as shown in Figure 2.

Figure 2. Hybrid intrusion detection system flowchart.

The first stage of the proposed framework results in one of two possible outcomes: Known attacks and unknown samples. At the second stage, the unknown samples (branching out at 'Not Exists' in the figure) are then presented to AIDS for further training and analysis to overcome the shortcoming of SIDS.

3.2. *Stage 2: AIDS Based One-Class SVM*

To successfully identify new intrusions, the result of the SIDS phase should be used as input data in the AIDS stage to detect new intrusions. AIDS should be built based on the normal activity of a user, which could have been used during the training stage. Then, intrusions are detected based on the measured state of the user profile, which is compared to the normal profile (determined based on the model), if it varies more than the described threshold, then it is marked as a malicious. The one class SVM learning model is used to identify normal behavior, as the One-class SVM (OCSVM), which learns the attributes of benign samples without using any information from the other class. One-class SVM was suggested by Schölkopf et al. [28] to predict the support of a high-dimensional distribution by modifying the SVM method to the one-class problem. It involves the first feature processing through a kernel and then employs relaxation parameters to separate the test point of a class from the rest of the datasets or origin [29], as illustrated in Figure 3. Relaxation parameters techniques are iterative approaches for solving large sparse linear systems. It is also used to solve linear least-squares and nonlinear equations problems. Relaxation parameters help SVM to control the compromise between the reaching of a low detection rate on the training stage and a low detection rate on the testing stage, which is the capacity to identify and classify unknown malwares.

The OCSVM classifier transforms instances into a large dimensional attribute space (via a kernel) and locates the suitable location of the boundary hyperplane, which splits the training data. The OCSVM is a normal binary-class SVM where all the training data are based on the first class. Thus, we consider those profiles to be abnormal, which are close to the origin of coordinates in a feature space. The establishment of the hyperplane needs to follow the categorization rule:

$$f(x) = (w, x) + b \tag{1}$$

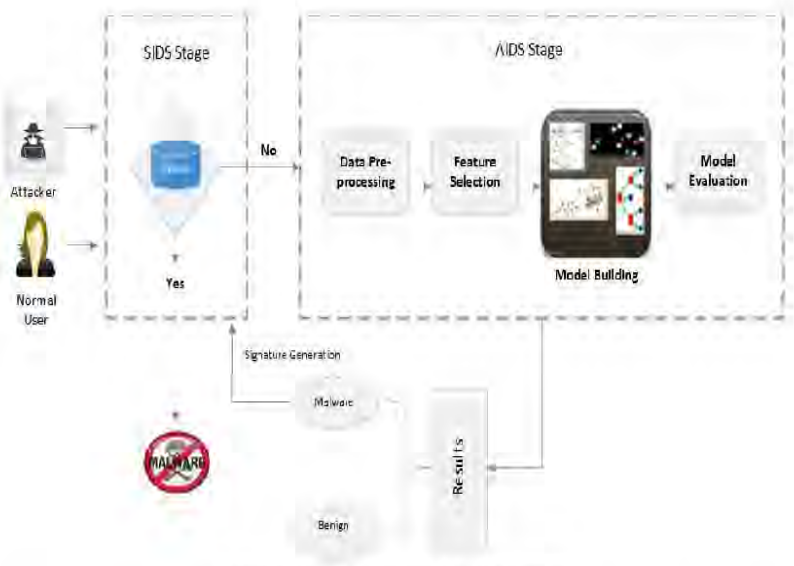


Figure 1. Hybrid intrusion detection system.

AIDS profiles normal user activities and raises a malicious alarm when the difference between a given observation at an instant and the recorded value of the profile exceeds a predefined threshold. User profiles are created from the records generated from user activities and are marked as benign. AIDS feedbacks malicious records to SIDS to be saved in the signature database. The principle reason for storing the intrusion data in the signature database is to mitigate against known intrusions. The performance of the proposed HIDS and its components (SIDS and AIDS) are evaluated by conducting experiments separately. In the following, the two phases of the proposed detection system are elaborated.

3.1. Stage 1: SIDS Based C5

SIDS is used in the first stage as it provides high accuracy in detecting well known intrusions and generates low false alarms. As a result, false positives are low as all known signatures for malicious samples are kept in the database. Therefore, attacks can be detected with high accuracy while reducing false positives.

In SIDS, the C5 classifier is used in this stage to detect well-known intrusions. In our previous work, C5 was analyzed and contrasted with other machine learning techniques [26]. The results revealed that C5 performs very well in terms of the detection rate and false alarm rate. The C5 algorithm is an improved version of the commonly used C4.5 classifier which was developed by Quinlan [27], based on decision tree [26]. It incorporates variable misclassification costs, handles missing data, can handle large numbers of input fields, and builds the model very fast. It takes a set of known data as the input and builds a decision tree from that data. In C5, the decision tree is built in a top-down fashion. The first attribute and its values are at the top of the tree and the next branch leads to either an attribute or outcome. C5 decision trees are created in view of a number of features and a set of training stages, and then the tree could be categorized by using a subsequent set to distinguish other samples.

We have used the C5 classifier for SIDS, as shown in Figure 2. Unknown samples are handled through pattern matching in order to determine whether they represent normal or abnormal activities. If the unknown sample is found in the signature database, then it triggers an alarm that it is a malware.

Table 1. Comparison of intrusion detection systems (IDS) techniques and datasets issue covered (✓: Topic is covered, ✕ the topic is not covered).

IDS Methods	Intrusion Detection System Techniques						Dataset Issue
	SIDS	AIDS				Hybrid IDS	
		Supervised Learning	Unsupervised	Semi-Supervised Learning	Ensemble Methods		
Lunt [19]	✓	✕	✕	✕	✕	✕	✕
Axelsson [20]	✓	✓	✕	✕	✕	✕	✕
Liao, et al. [21]	✓	✓	✓	✕	✕	✓	✕
Agrawal and Agrawal [22]	✓	✓	✓	✓	✓	✓	✕
Buczak and Guven [23]	✓	✓	✓	✕	✓	✓	✓
Ahmed, et al. [24]	✕	✓	✓	✕	✕	✕	✓
Khraisat, et al. [4]	✓	✓	✓	✓	✓	✓	✓

Ghanem et al. proposed a hybrid detection approach for large datasets using detectors generated based on different machine learning techniques. Anomaly detectors were developed based on self and non-self-training data to obtain self-detectors [25]. K-means that clustering is used to decrease the volume of the training dataset by removing the redundant detector. The key role of AIDS is to create normal profiles of attacks. If the patterns are general in nature, then it is unable to identify several intrusions, which results in a poor detection rate. If the profiles are very specific, then it can identify different intrusions, but several normal behaviors could be classified as attacks. However, none of the previous works have attempted to strike a balance between accuracy and false positives. Recent studies focus on decreasing the false positive rate of AIDS by proposing a hybrid IDS. While earlier studies only integrate the results of both detection models, but in the proposed technique, the intrusion detection systems are hierarchically combined to improve accuracy. This allows the AIDS to improve its normal profiling capability with the use of the signature detection model. The details of the proposed Hybrid IDS are described in Section 3.

3. Hybrid Intrusion Detection System

Hybrid IDS is developed to overcome the disadvantages of SIDS and AIDS as it integrates SIDS and AIDS to detect both unknown and known attacks. In our approach, we used AIDS to identify unseen intrusions, while SIDS is used to identify well-known attacks. Our system is based on three stages process:

- Stage 1: SIDS based C5
- Stage 2: AIDS based One-Class SVM
- Stage 3: HIDS based Stacking Ensemble of C5 and One-Class SVM

Our hybrid IDS (HIDS) ultimately combines the C5 classifier (in the first stage) and One Class Support Vector Machine (in the second stage). The central idea of this novel approach was to combine the advantages of both SIDS and AIDS to build an efficient IDS. The Hybrid IDS has two phases; the SIDS phase and AIDS phase are shown in Figure 1.

Our intrusion detection techniques included online and offline, which means detection intrusion later and online stages, which means detection intrusions in real-time. In the offline stage, the C5 classifier learning method was used to update the signature database. This stage deals with the stored signature and passes it through some processes to decide if it is an attack or not. In the online stage, the initial detection model was created using a one-class SVM. The online IDS deals with the network in real-time. This stage analyses the network traffic to decide if it is an intrusion or not.

- Proposes a novel intelligent framework to identify well-known intrusions and zero-day attacks with high detection accuracy and low false-alarm rates.
- Develops a stacked hybrid IDS based on SIDS and AIDS to harness their respective strengths for better detection.
- Evaluates Hybrid Intrsuion Detection System (HIDS) on the Network Security Laboratory-Knowledge Discovery in Databases (NSL-KDD) and Australian Defence Force Academy (ADFA) datasets benchmark datasets in terms of accuracy and F-measure, and studies and validates its superior performance.

The rest of the paper is organized as follows: Section 2 reviews a few related works. Section 3 describes the detailed description of the proposed hybrid intrusion detection. Experimental results are presented in Section 4. Section 5 concludes the paper with a brief discussion and summary.

2. Related Work

There are many IDSs in the literature to identify abnormal activities, but most of these IDSs produce a large number of false positives and low detection accuracy. However, many hybrid IDSs have been proposed to overcome the drawbacks of SIDS and AIDS. Sumaiya et al. proposed an intrusion detection model by applying the chi square attribute extraction and multiclass support vector machine [11]. Syarif et al. used boosting, stacking, and bagging ensemble techniques for IDS to enhance the intrusion detection rate and to reduce false alarms [12]. They used four diverse machine learning techniques: Naïve Bayes, artificial neural networks, decision tree, genetic algorithms, rule induction, and k-nearest neighbor as the foundation classifiers for the ensemble methods. They highlighted that their proposed method has an accuracy of 99% in finding known attacks, but could only identify zero-day attacks at around 60% accuracy rates.

Kim et al. [13] presented a HIDS technique that hierarchically combines SIDS and AIDS models. Signature detection is developed based on the J48 decision tree classifier. Then, various one-class support vector machines models are built for the split subsets, which can reduce the profiling capability.

Muniyandi et al. [14] proposed an anomaly detection method that employs "K-Means + C4.5", for classifying anomalous and normal activities in a computer system. The K-Means clustering method is employed to divide the training data into k number of clusters by using the Euclidean distance similarity. The Hybrid method beats the individual method in terms of accuracy, but it causes high false alarms.

Al-Yaseena et al. [15] proposed a multi-level hybrid IDS which employs SVM and extreme learning machine to enhance the detection efficiency for known and unknown attacks. The system performed well on the KDD Cup 1999 dataset in terms of the false alarm rate, which was 1.87%.

Koc et al. proposed the Hidden Naïve Bayes (HNB) model for IDS issues such as dimensionality, correlated features, and big data stream [16]. The results showed that this method achieved an overall performance that was comparable to the Naïve Bayes model. Sivatha et al. proposed a lightweight IDS to classify abnormal activities in the network using wrapper based feature selection techniques that create good intrusion detection rates by adding the neural ensemble decision tree classifier [17].

M. Alazab proposed a methodology to extract features statically and dynamically from malware such as the Windows Application Programming Interface (API) calls and create a malware behavior profile by extracting malware API calls throughout execution [18].

Table 1 shows the IDS techniques and datasets covered by different intrusion detection system survey papers. Khraisat et al. presented a survey of current intrusion detection systems, which was a wide-ranging review of ID technqiues, and the datasets usually employed for evaluation purposes [4]. Several intrusion detection system techniques that have been developed to improve the detection rate. However, such technques may have difficulty increating and updating the signarture of new malware and in yielding high false alarms or poor detection rates.

about a known malware so that malware can be detected in the future [5]. If that specific signature is identified again, the traffic can be identified as being malware. SIDS usually gives an excellent detection accuracy, particularly for previously known intrusions.

Since SIDS can be as effective as the update of the signature database, three issues arise. Firstly, it is easy to trick signature-based systems through the polymorphic behavior of malware. This method fails the similarity test as it does not match with any signature stored in the IDS database, giving the attacker a chance to gain access to the computer system. Secondly, the higher the number of signatures in the database, the longer it takes to analyses and process the huge volume of data. Thirdly and most importantly, SIDS has difficulty in detecting zero-day malware as the signature is not stored in the database [6].

AIDS systems have overcome the limitation of SIDS and are being used to identify malicious attacks on computer systems. The assumption for this technique is that the profile of a malicious activity differs from typical user behavior activities [7]. AIDS creates a statistical model describing the normal user activity and any abnormal activity that deviates from the normal model is detected. The design idea behind AIDS is to profile and represent the normal and expected standard behavior profile through monitoring activities and then defininining anomalous activities by their degree of deviation from the normal profile. AIDS uses features such as the number of emails sent by a user, the number of failed logins tries for a user, and the degree of processor use for a host in a given timeframe in learning the normal behaviors. Anomaly detection techniques have the ability of strong generalizability and to detect new attacks, while its drawbacks could be in the form of large false alarm rates due to the changing cyber-attack landscape.

The behaviors of alien users are deemed different to the standard activities and are categorized as intrusions. AIDS includes two phases: Thee training phase and testing phase. In the training phase, the normal traffic profile is learned from the data that represent normal behavior, and then the testing is done on a data set that is not seen by the model during the training phase.

AIDS could be classified into several sub-classes based on the learning methods, for instance, statistical based, knowledge based, and machine learning based [8].

The key advantage of AIDS is its ability to identify zero-day attacks as it does not have to rely on the signature database to detect an attack. AIDS triggers an alert signal when the examined behavior differs from the usual activity [9]. Furthermore, AIDS has various benefits: Firstly, it has the capability to discover internal malicious activities. If an intruder starts transacting on a stolen account, which could be misidentified as the normal user's activity, then it generates an alarm. Secondly, it is very difficult for a cybercriminal to learn what a normal user's behavior is without producing alerts, as the system is constructed from customized profiles [4].

Traditional IDSs have limitations: Inability to differentiate new malicious attacks, the need to be updated, low accuracy, and high false alarms. AIDS also has shortcomings such as a high number of false alarms [10]. To overcome those limitations, an innovative IDS model is proposed with the integration of SIDS and AIDS in order to achieve accuracy and to reduce the false alarm. Well known intrusions could be detected by SIDS and new attacks could be detected by AIDS.

In this paper, the signature intrusion detection system is built, based on the C5.0 Decision tree classifier, and the Anomaly intrusion detection system is built, based on one-class Support Vector Machine (SVM). The target is to understand how the intrusion detection system accuracy can be enhanced by utilising the ensemble stacking technique. While the traditional intrusion detection system concentrates on how the performance of a one classifier can be enhanced, this research studies how diverse machine learning techniques can be integrated to enhance intrusion detection accuracy. In this research, two machine learning techniques, namely the decision tree c5 and one class SVM classifier, were chosen to build the intrusion detection system. The decision tree c5 and one class SVM classifier have been evaluated independently and in combination by using stacking ensemble, which is tested as well.

Our paper makes the following contributions:

electronics

MDPI

Article

Hybrid Intrusion Detection System Based on the Stacking Ensemble of C5 Decision Tree Classifier and One Class Support Vector Machine

Ansam Khraisat *, Iqbal Gondal, Peter Vamplew, Joarder Kamruzzaman and Ammar Alazab

Internet Commerce Security Laboratory, Federation University Australia, Mount Helen 3350, Australia; iqbal.gondal@federation.edu.au (I.G.); p.vamplew@federation.edu.au (P.V.); joarder.kamruzzaman@federation.edu.au (J.K.); aalazab@mit.edu.au (A.A.)

* Correspondence: a.khraisat@federation.edu.au

Received: 10 January 2020; Accepted: 14 January 2020; Published: 17 January 2020

check for updates

Abstract: Cyberttacks are becoming increasingly sophisticated, necessitating the efficient intrusion detection mechanisms to monitor computer resources and generate reports on anomalous or suspicious activities. Many Intrusion Detection Systems (IDSs) use a single classifier for identifying intrusions. Single classifier IDSs are unable to achieve high accuracy and low false alarm rates due to polymorphic, metamorphic, and zero-day behaviors of malware. In this paper, a Hybrid IDS (HIDS) is proposed by combining the C5 decision tree classifier and One Class Support Vector Machine (OC-SVM). HIDS combines the strengths of SIDS) and Anomaly-based Intrusion Detection System (AIDS). The SIDS was developed based on the C5.0 Decision tree classifier and AIDS was developed based on the one-class Support Vector Machine (SVM). This framework aims to identify both the well-known intrusions and zero-day attacks with high detection accuracy and low false-alarm rates. The proposed HIDS is evaluated using the benchmark datasets, namely, Network Security Laboratory-Knowledge Discovery in Databases (NSL-KDD) and Australian Defence Force Academy (ADFA) datasets. Studies show that the performance of HIDS is enhanced, compared to SIDS and AIDS in terms of detection rate and low false-alarm rates.

Keywords: anomaly detection; hybrid approach; C5.0 Decision tree; Cyber analytics; data mining; machine learning; Zero-day malware; Intrusion; Intrusion Detection System

1. Introduction

Zero-day intrusion detection is a serious challenge as hundreds of thousands of new intrusions are detected every day and the damage caused by these intrusions is becoming increasingly harmful [1,2] and could result in compromising business continuity. Computer attacks are becoming more complicated and lead to challenges in detecting the intrusion correctly [3].

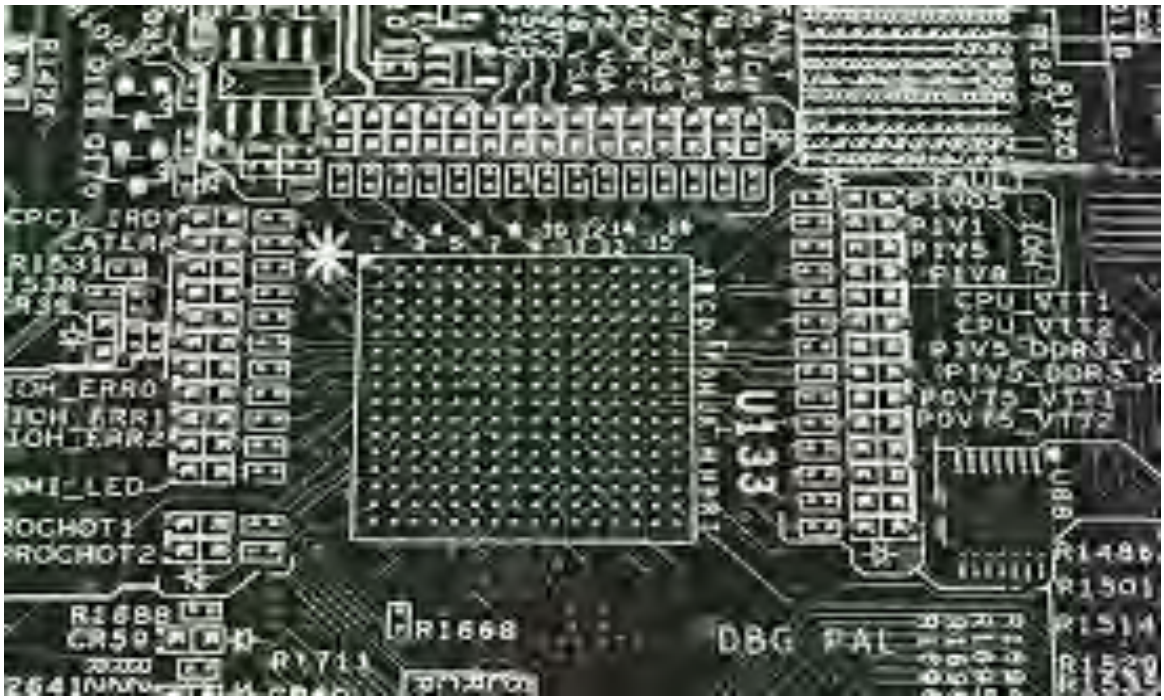
Intrusion detection systems (IDS) detect suspicious activities and known threats and generate alerts. Intrusions could be identified as any activity that causes damage to an information system [4]. IDS could be software or hardware systems capable of identifying any such malicious activities in computer systems. The goal of intrusion detection systems is to monitor the computer system to detect abnormal behavior, which could not be detected by a conventional packet filter. It is very vital to achieve a high degree of cyber resilience against the malicious activities and to identify unauthorised access to a computer system by analysing the network packets for signs of malicious activity.

IDSs use two broad methodologies for intrusion detection: The Signature-based Intrusion Detection System (SIDS) and Anomaly-based Intrusion Detection System (AIDS). SIDS, also called Knowledge-Based Detection or Misuse Detection, is a process where a signature identifier is determined

27. Quinlan, J.R. *C4. 5: Programs for Machine Learning*; Elsevier: Amsterdam, The Netherlands, 2014.
28. Schölkopf, B.; Platt, J.C.; Shawe-Taylor, J.; Smola, A.J.; Williamson, R.C. Estimating the support of a high-dimensional distribution. *Neural Comput.* **2001**, *13*, 1443–1471. [CrossRef] [PubMed]
29. Creech, G.; Hu, J. A Semantic Approach to Host-Based Intrusion Detection Systems Using Contiguousand Discontiguous System Call Patterns. *IEEE Trans. Comput.* **2014**, *63*, 807–819. [CrossRef]
30. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6.
31. Abbasi, A.; Wetzels, J.; Bokslag, W.; Zambon, E.; Etalle, S. On Emulation-Based Network Intrusion Detection Systems. In Proceedings of the Research in Attacks, Intrusions and Defenses: 17th International Symposium, RAID 2014, Gothenburg, Sweden, 17–19 September 2014; Stavrou, A., Bos, H., Portokalidis, G., Eds.; Springer: Cham, Switzerland, 2014; pp. 384–404. [CrossRef]
32. Panda, M.; Abraham, A.; Patra, M.R. Discriminative multinomial Naïve Bayes for network intrusion detection. In Proceedings of the 2010 Sixth International Conference on Information Assurance and Security, Baltimore, MD, USA, 23–25 August 2010; pp. 5–10.

3. Alazab, A.; Hobbs, M.; Abawajy, J.; Khraisat, A. Malware detection and prevention system based on multi-stage rules. *Int. J. Inf. Secur. Priv.* **2013**, *7*, 29–43. [CrossRef]
4. Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J. Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity* **2019**, *2*, 20. [CrossRef]
5. Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J.; Alazab, A. A Novel Ensemble of Hybrid Intrusion Detection System for Detecting Internet of Things Attacks. *Electronics* **2019**, *8*, 1210. [CrossRef]
6. Alazab, A.; Hobbs, M.; Abawajy, J.; Khraisat, A.; Alazab, M. Using response action with intelligent intrusion detection and prevention system against web application malware. *Inf. Manag. Comput. Secur.* **2014**, *22*, 431–449. [CrossRef]
7. Alazab, A.; Hobbs, M.; Abawajy, J.; Alazab, M. Using feature selection for intrusion detection system. In Proceedings of the 2012 International Symposium on Communications and Information Technologies (ISCIT), Gold Cost, Australia, 2–5 October 2012; pp. 296–301.
8. García-Teodoro, P.; Díaz-Verdejo, J.; Maciá-Fernández, G.; Vázquez, E. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Comput. Secur.* **2009**, *28*, 18–28. [CrossRef]
9. Huda, S.; Abawajy, J.; Alazab, M.; Abdollalihian, M.; Islam, R.; Yearwood, J. Hybrids of support vector machine wrapper and filter based framework for malware detection. *Future Gener. Comput. Syst.* **2016**, *55*, 376–390. [CrossRef]
10. Alazab, A.; Abawajy, J.; Hobbs, M.; Khraisat, A. Crime toolkits: The current threats to web applications. *J. Inf. Priv. Secur.* **2013**, *9*, 21–39. [CrossRef]
11. Sumaiya Thaseen, I.; Aswani Kumar, C. Intrusion detection model using fusion of chi-square feature selection and multi class SVM. *J. King Saud Univ. Comput. Inf. Sci.* **2017**, *29*, 462–472. [CrossRef]
12. Syarif, I.; Zaluska, E.; Prugel-Bennett, A.; Wills, G. *Application of Bagging, Boosting and Stacking to Intrusion Detection*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 593–602.
13. Kim, G.; Lee, S.; Kim, S. A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. *Expert Syst. Appl.* **2014**, *41*, 1690–1700. [CrossRef]
14. Muniyandi, A.P.; Rajeswari, R.; Rajaram, R. Network anomaly detection by cascading k-Means clustering and C4. 5 decision tree algorithm. *Procedia Eng.* **2012**, *30*, 174–182. [CrossRef]
15. Al-Yaseen, W.L.; Othman, Z.A.; Nazri, M.Z.A. Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system. *Expert Syst. Appl.* **2017**, *67*, 296–303. [CrossRef]
16. Koc, L.; Mazzuchi, T.A.; Sarkani, S. A network intrusion detection system based on a Hidden Naïve Bayes multiclass classifier. *Expert Syst. Appl.* **2012**, *39*, 13492–13500. [CrossRef]
17. Sivatha Sindhu, S.S.; Geetha, S.; Kannan, A. Decision tree based light weight intrusion detection using a wrapper approach. *Expert Syst. Appl.* **2012**, *39*, 129–141. [CrossRef]
18. Alazab, M. Profiling and classifying the behavior of malicious codes. *J. Syst. Softw.* **2015**, *100*, 91–102. [CrossRef]
19. Lunt, T.F. Automated audit trail analysis and intrusion detection: A survey. In Proceedings of the 11th National Computer Security Conference, Baltimore, MD, USA, 17–20 October 1988.
20. Axelsson, S. *Intrusion Detection Systems: A Survey and Taxonomy*; Technical Report; Department of Computer Engineering, Chalmers University of Technology: Gothenburg, Sweden, 2000.
21. Liao, H.-J.; Lin, C.-H.R.; Lin, Y.-C.; Tung, K.-Y. Intrusion detection system: A comprehensive review. *J. Netw. Comput. Appl.* **2013**, *36*, 16–24. [CrossRef]
22. Agrawal, S.; Agrawal, J. Survey on anomaly detection using data mining techniques. *Procedia Comput. Sci.* **2015**, *60*, 708–713. [CrossRef]
23. Buczak, A.L.; Guven, E. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 1153–1176. [CrossRef]
24. Ahmed, M.; Naser Mahmood, A.; Hu, J. A survey of network anomaly detection techniques. *J. Netw. Comput. Appl.* **2016**, *60*, 19–31. [CrossRef]
25. Ghanem, T.F.; Elkilani, W.S.; Abdul-kader, H.M. A hybrid approach for efficient anomaly detection using metaheuristic methods. *J. Adv. Res.* **2015**, *6*, 609–619. [CrossRef]
26. Khraisat, A.; Gondal, I.; Vamplew, P. An Anomaly Intrusion Detection System Using C5 Decision Tree Classifier. In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, Melbourne, Australia, 3–6 June 2018; pp. 149–155.

Chapter 5 : A Low-Level Intrusion Detection System Based on Hardware Performance Counters



In the previous chapter, we have applied our IDS framework on both HIDS on NIDS. In this chapter, we propose an intrusion detection system using low-level Hardware performance counters as features.

HIDS inspect data that originates from the host system, audits sources, such as operating system, window server logs, firewalls logs, application system audits, or database logs. HIDS can detect insider attacks that do not involve network traffic (Garfinkel & Rosenblum, 2003). To reduce performance overhead HPCs can be used to detect attacks. Also, it is very hard for the attackers to bypass intrusion detection based on HPCs as these features are reliable in

exhibiting the behaviours of the applications. Therefore, HPC features are strong which can be used to identify whether an application is normal or malware.

Studies presented in this chapter have shown that HPC features based IDS provide higher detection rate as it is hard for a malware authors to avoid detection as attackers require high access privileges. We collect normal and malware applications and extract their HPC features, and three different attack techniques were included. In this chapter, we have applied our novel methodology on the lower layers of the computer systems, to identify intrusion by examining the utilization of Hardware Performance Counters. This chapter has been ACCEPTED to be included in the upcoming book entitled “Malware Analysis using Artificial Intelligence and Deep Learning”, which will be published by Springer on 5/1/2020.

A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, " A Low-Level Intrusion Detection System Based on Hardware Performance Counters," Malware Analysis using Artificial Intelligence and Deep Learning, Springer, 2020 (Accepted)

5.1 Abstract

Traditionally, Intrusion Detection Systems (IDSs) rely on computer program behaviors at operating systems' level to detect malware. Most of these techniques use high semantic features, such as functions and system calls. These high semantic features are susceptible to malicious attacks at higher privilege levels. In particular, a malicious malware rootkit may bypass intrusion detection by manipulating system data or operating system code. In this paper, a framework for profiling normal and malicious activities is proposed. This framework is based on Hardware Performance Counters (HPCs) and hybrid IDS to detect malware. Extensive experiments have been conducted to study the effectiveness of the HPCs that could distinguish between malware and normal applications. The performance of the proposed approach has been tested on Windows-based malware families and demonstrated a detection rate of 99%.

5.2 Introduction

The number of malware has been increasing and causing severe damage to Internet applications over the past few years (Zhang, Leach, Stavrou, Wang, & Sun, 2015). McAfee threats report has revealed that during the first quarter of 2017 more than 40 million different malware was in the wild (McAfee, 2016). Furthermore, malware authors are gaining complete control over the computer's system by shifting their efforts to the lower layers by designing more sophisticated malware. These malware pose a great threat by disabling the installed anti-malware system (A. Alazab, Abawajy, Hobbs, & Khraisat, 2013). Recently, there have been researching efforts to identify intrusion by examining the utilization of Hardware Performance Counters (HPCs). The aim of these works is to distinguish a malware on the basis of its profile

that characterizes the manner in which it impacts execution counters that are incorporated into the processor of the machine. Existing research has concentrated on running malware pairs while gathering HPC data and utilizing that information to create Intrusion Detection System (IDS) based on different machine learning techniques. Initial outcomes have been promising, showing detection rates of over 91% (Sayadi et al., 2018). However, rootkits intrusion identification is not promising. Rootkits are an uncommon kind of malware that modifies parts of the running operating system bit so as to conceal the existence of malicious activity on a computer. Modern rootkits are extremely obfuscated to evade detection and analysis, feature encrypted files, encrypted connections, and a flexible design that permits various kinds of malware to collaborate to exploit a computer system. Advanced rootkits perform malicious activities, such as concealing malware activity, payload functionality, sending spam emails, stealing user information and easy installation. Rootkit detection is difficult as rootkit might subvert the IDS. Though Demme et al. achieved some success in detecting rootkits with some initial experiments, the detection accuracy was low (Demme et al., 2013).

Conventional malware analysis employs virtualization (Dinaburg, Royal, Sharif, & Lee, 2008) (Deng, Zhang, & Xu, 2013) and emulation (Yan, Jayachandra, Zhang, & Yin, 2012) (Egele, Scholte, Kirda, & Kruegel, 2012) methodologies to explore malware behavior at runtime. These methods execute the malware in a Virtual Machine (VM) or use emulators to analyze the software to examine the presence of malware. Unfortunately, the malware author can simply evade this analysis technique by applying different anti-debugging, anti-virtualization, and anti-emulation techniques (Branco, Barbosa, & Neto, 2012) (Pavlyushchik, 2014). Malware can simply detect the presence of a VM or emulator and can change their behaviors.

In addition to this, malware authors regularly use advanced techniques such as obfuscation to avoid detection. This can lead to zero-day attacks being undetected by the existing intrusion detection systems (M. Alazab, Venkatraman, Watters, Alazab, & Alazab, 2012). Polymorphism and metamorphism are two common obfuscation methods used by malware authors. For example, to evade detection, malware can hide their persistence by encrypting malicious payload and decrypting it during runtime. A polymorphic malware obfuscates using several transformations, such as adding some extra instructions to malware to change its appearance, but keeps its functionality intact, or changing the sequence of instructions and adding jump instructions to keep the original malware functionality (M. Alazab et al., 2012). It is quite simple for malware authors to write different malware variants that are functionally the same but can evade signature methods (A. Alazab et al., 2013).

It has been argued that hardware-level detection is more suitable than other kinds of detections (Demme et al., 2013; Singh, Evtyushkin, Elwell, Riley, & Cervesato, 2017; Tang, Sethumadhavan, & Stolfo, 2014; Wang & Karri, 2016). This is because it is harder for a malicious program to defeat an intrusion system that uses hardware events, due to the difficulties faced by the malware author to control low-level hardware events (Demme et al., 2013; Singh et al., 2017). For instance, an attacker can more easily change high semantic information than low semantic information. Therefore, building IDS directly on the top of hardware gives significant advantages, such as a reduction in overhead introduced, as only the virtual machine needs to be changed. Furthermore, hardware-based IDSs are easy to deploy as they are not tied to any specific operating system and implementation can happen transparently underneath various operating systems (Demme et al., 2013).

This paper proposes a Hybrid IDS to improve the performance of hardware-based malware detectors utilizing a few very low features of microarchitectural events HPCs that are extracted at run-time by current HPCs. In this paper, an analysis of the effectiveness of HPCs to distinguish between malware and normal applications is conducted. HPCs are counters built in the modern computes to store the counts of hardware-related activities within computer systems such as cache hits, clock cycles, and integer instructions. We experimentally show how several types of intrusion and attack techniques that affect the HPCs and reveal which significant HPCs can be used in distinguishing malware. Our results show that the HPCs are impacted severely by system call hooking attacks.

The main contributions of this paper are as follows:

- We investigate which malware and attack techniques impact HPCs and how these HPCs can be used for successful malware detection.
- Apply information gain principle to select the top HPC features which show maximum improvement in detection.
- Design of a low-level intrusion detection system (IDS) based on HPCs. We adopt a hybrid approach that exploits the strength of signature and anomaly-based methods utilizing features extracted from HPC events. The proposed IDS demonstrates promising performance.

5.3 Background

For the development of the HPC based IDS extensive literature review on IDSs, HPCs and threat models are presented to develop the motivation of our work

5.3.1 Intrusion Detection System

IDSs are typically classified into Signature-based Intrusion Detection System (SIDS) and Anomaly-based Intrusion Detection System (AIDS). SIDS relies on pattern matching techniques to decide whether a given pattern is a known attack. SIDS usually attains high accuracy in detecting known malware [16]. However, it has difficulty in detecting zero-day attacks as the zero-day pattern does not exist in the training databases.

On the other hand, AIDS relies on events related to normal system behaviors [16, 17], so abnormal behavior can be taken as attacks. In general, IDSs can be used to monitor the system activities at different system levels.Table 5. 1 shows such IDSs at different levels along with their capabilities(M. Alazab, 2015).

Level characteristics	**IDS levels**			
	Application	**Operating System (OS)**	**Virtual Machine Monitor (VMM)**	**Network Level**
Description	IDSs rely on the information that is available from software applications.	The Operating System IDS observe resource usage of the OS and the network resource	VMM-level architectural events	An IDS is placed at points in a network so that it can monitor the traffic between various devices
Example of Features	Files, Downloads, File Systems, API calls	System Calls, Memory, Dynamic Instructions, System Calls, Registry Entries	uArch Events	Packet size, Source port, Destination port, Timestamp
Semantics	High	Medium	Low	Medium
Robustness	Low	Medium	High	Medium
Ease of deployment	Low	Medium	High	High

Applicability	High	High	Medium	High

Table 5. 1 Intrusion detection systems level

Application-level IDSs have more ability to detect and classify attacks as compared to IDSs at lowers level. Many features may have extra overhead during feature extraction. IDS at OS level and function by monitoring the system calls being executed by programs. VMM IDS only needs features that are obtained from the VMM layer, and this helps to reduce overhead. The VMM-level IDS have a great ability not to be affected by malicious attacks, whereas application-level and operating system semantics are still susceptible to malicious attacks due to higher privilege levels exploitation. For instance, a rootkit may bypass both application level and Operating System-level security check but may be detected at VM level. Network intrusion detection systems are implemented at dedicated devices within the network, where it can inspect inbound and outbound network packets from all the computer systems.

5.3.2 Hardware Performance Counters (HPCs)

HPCs are a collection of special-purpose registers in modern microprocessors to collect the counts of hardware-related activities, such as cache misses and cache hits for various cache levels, pipeline stalls, processor cycles, instruction issues, and branch misprediction (Tang et al., 2014). Initially developed by computer hardware engineers for debugging CPUs, HPCs can also be programmed to calculate CPU events such as instruction types and executed code paths.

However, the performance of the CPU is significantly affected by how a program uses its resources, e.g. memory access patterns effecting cache performance, instruction control flow

effecting cache performance and branch prediction, the amount of inherent parallelism in the instructions affecting the utilization of functional units, data cache misses indicating important signs about slow/fast program code. Therefore, there are a lot of events happening inside a CPU which indicate performance.

Cybersecurity researchers have proposed HPCs based techniques for identifying intrusions. Intrusion detection system based on HPCs creates a profile of software by examining the counter values standard regularly. Next, machine learning could be used to differentiate malware behaviors among normal application. There are many advantages for building intrusion Detection system based on HPCS. (1) Reduction in performance overhead in detecting malware. (2) It is very hard for the attackers to bypass intrusion detection based on HPCs as the HPCs features are reliable in showing the nature of the application, whether it is normal or malware.

5.3.3 Threat Model

In this chapter, we focus on malware attacks that have a high level of privilege within the guest VM. These malware can pose dangerous threats to the guest VM's kernel space on both hypervisor and operating system level, as it has sufficient privileges so that they may successively change the boot process to run their malicious code, e.g. rootkit. Rootkits are used by malware authors to avoid detection and maximize their probability of a successful attack on the target computer systems.

The rootkit normally hides their attack activity, by changing the configuration of the operating system to obscure specific processes, network ports, and directories; and installs Kernel Hooks in an attempt to remain stealthy. Once the rootkit is set up, it controls the victim operating

system and is able to hide its persistence from traditional IDSs (Wang & Karri, 2016). Detection systems attempt to identify these malicious activities by deploying IDS at higher privilege levels, resulting in competition between security vendors an attacker to control the lower levels of the system. Therefore, traditional detection mechanisms operate at the operating system levels, such as memory-based integrity scanners or heuristic-based analysis, which are not adequate to identify rootkits on lower system levels (Vinayakumar et al., 2019).

5.3.4 Motivation

Before we present our IDS, we will give an intuitive explanation of the underlying ideas. For example, consider the two matrix operation code samples in Table 5. 2 The only difference between the two is the order of the array indices on the inner loop. However, their performance varies significantly because of cache performance.

Example 1	*Example 2*

int main (void)
{
int i, j;
for (i = 0; i < 1000; i++)
for (j = 0; j < 1000; j++)
array[j][i]++;
return 0;
}
	int main (void)
	{
	int i, j;
	for (i = 0; i < 1000; i++)
	for (j = 0; j < 1000; j++)
	array[i][j]++;
	return 0;
	}

Table 5. 2 Two Examples of matrix manipulation

Table 5. 3 shows the results of the execution of both pieces of code while capturing four HPCs related to the L1 cache. We execute those examples on Intel processors, running 64-bit Windows 7 operating system (OS). As can be seen, Example 1 has a significant number of cache misses.

Name	Instruction Executed	L1-I cache load misses	Li-d cache load-misses	L1-d cache store-Misses	L1-d cache pre-fetch misses
Example 1	21,270,318	1,991,540	999,121	1,092,899	993,746
Example 2	21,265,465	20,412	24,976	20,485	3,761

Table 5. 3 Values of two examples measured by HPCs

The difference in profiles between the two examples listed above raises a question – Can an Intrusion detection system based on HPCs distinguish between malware and normal applications? In this paper, we assess the effectiveness of HPCs in capturing the behavioral semantics of a program. To this end, we profile normal applications and malicious applications using HPCs and report the performance of building an intelligent intrusion detection system.

5.4 Related Work

Current IDSs can be categorized based on two main characteristics: the detection approach and the malware features used for their detection. Various features are used to build IDSs, e.g., memory, dynamic instructions, system calls, registry entries, and others (Liao, Lin, Lin, & Tung, 2013; Shakshuki, Kang, & Sheltami, 2013). The usage of low-level hardware events for building intrusion detection system is a recent research direction. Consequently, there have been very few related works focusing on how to use HPCs as features for building IDS, but recently the feasibility of designing malware detection with performance counters has been studied. Demme et al. found that existing performance counters can be used to identify malware (Demme et al., 2013). They investigated a small set of program behavior within a family of malware on Android and Intel Linux platforms. The work demonstrated that offline machine learning techniques could be used to classify malware based on performance counters. They achieved prediction accuracies ranging from 30% to 98% across different Android malicious

software. However, their methodology lacks any information about which HPC events could be used for detection purpose (Huda et al., 2016).

The work of de Melo et al. (de Melo, 2009) used unsupervised machine learning to build profiles of the HPC patterns of benignwares and then identify abnormalities. de Melo et al. have applied F-Score as their feature selection and have provided a comparison of performance while using different sampling frequencies for the HPCs. They used only two applications, namely, Internet Explorer and Adobe Acrobat, in their proof of concept and Metasploit to create the exploits for the applications. We avoid using F1-score feature selection as it does not reflect any mutual dependence between the collected events. F-score discloses the discriminative power of each feature individually as one score is calculated for the first feature, and another score is calculated for the second feature. But it does not consider the utility of everything on the combination of both features.

In (Ozsoy, Donovick, Gorelik, Abu-Ghazaleh, & Ponomarev, 2015), Ousoy et al. proposed a Malware-Aware Processor (MAP), which is a sub-semantic hardware malware detector. This work investigated several sub-semantic feature vectors, and built an online hardware-supported malware detector while using various architectural features in distinguishing malware from normal programs online.

5.5 Experimental Setup

To investigate if HPC deviation could be used as features to identify malware, we conducted numerous feasibility experiments by building IDS using existing machine learning techniques

and validating their detection effectiveness based on the standard performance measures. In the following sections, we describe our experimental setup and explain how we collect normal and malware applications and extract their HPC features.

5.5.1 Measurement Infrastructure

All the experiments and measurements were executed on the VMWare Virtual Machine environment, installed with Windows 7, 64-bit and running a single-core with 512 MB of RAM. VTune Performance Analyzer (VirusTotal) was used to collect Hardware Performance Counters (HPCs) from both malware and benignware. Anti-security was disabled and connected to the network, while HPC features were extracted from malware. Every time a programmed event occurred, the count register was incremented. The quantity of hardware events that can be gathered simultaneously is limited by CPU abilities (Malladi, 2009).

5.5.2 Collection of Clean and Infected Measurements

In order to collect clean normal HPCs for windows, normal window's command was executed, such as nslookup, ping, tasklist, netstat, and ipconfig command were used as normal.

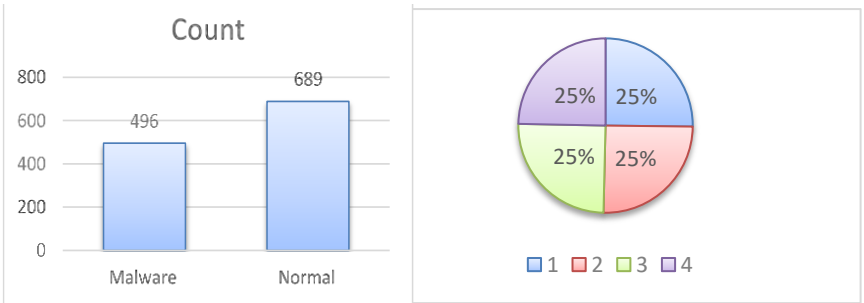


Figure 5.1 Distribution of malware and normal

Each command was executed individually and HPCs were extracted. For HPC data collection from malware, different variants of malware were used such as ZeroAcces, ZeusBoot, and Turla. The malware was chosen that had the following capabilities: hide processes, hidden services, hide listening TCP/UDP ports, hide kernel modules, hidden drivers, hide files and hide registry entirely. After execution of a single command, VM is restored to the normal state. This guarantees the HPC events are collected independently across normal and exploit execution. The dataset used in this experiment contains 1,185 executable application in total, as shown in Figure 2. Among them, 496 malicious software have been collected from the honeynet project, VX Heavens (VX Heavens, 2011), Virus Total (Total, 2012) and other sources. Three different attack techniques were included. Table 5. 4 shows a summary of the three-malware functionalities used in this study.

Attack techniques	**Functionality**
Direct kernel object manipulation (DKOM)	DKOM is a common malware method to hide possibly harmful processes, drivers, files, and intermediate connections from the task manager and event scheduler. Malware escalates the privileges of processes to gain control.
I/O Request Packet (IRP) hooking	Malware make specific files or processes disappears by avoiding them from showing in any file entries or task managers e.g. hiding file using IRP hooking or hiding process using IRP hooking
Process Hiding using System Service Dispatch Table (SSDT) hooking	Malware can change the original pointer value of an entry by the address of a function with the same prototype in kernel mode. For example, process Hiding SSDT hooking or file hiding using SSDT Hooking

Table 5. 4 Malware attack techniques used in this study

In our study, 237 HPC features were extracted into vectors as presented in Table 5. 5. An additional column of “class” has been added to each row in which extracted data is labeled as “Malware” or “Normal” depending on the application type.

ICACHE.IFETCH_STALL	ICACHE.MISSES	IDQ.ALL_DSB_CYCLES_4_UOPS	IDQ.ALL_DSB_CYCLES_ANY_UOPS	…… …… ……	Class
4995	4652	3985	6725	……	Malware
5249	4554	4215	6474	……	Malware
. . .	. . .	. . .	. . .	. . .	. . .
4846	4662	4251	6547	……	Malware
4149	3055	1641	2633	……	Normal
4150	2772	2075	2698	……	Normal
4177	2474	1831	2602	……	Normal

Table 5. 5 HPC features were extracted into vectors Sample

5.6 Proposed Hybrid Model for IDS

We rely on machine learning techniques and use HPC features to build intrusion detection system. Hybrid IDS is created to conquer the drawback of SIDS and AIDS as it coordinates SIDS and AIDS to identify both zero-day and known attacks. In our approach, AIDS is utilized to distinguish zero-day attacks, while SIDS is utilized to recognize understood attacks. The key idea of our approach is to consolidate the benefits of both SIDS and AIDS to create robust IDS. The techniques for creating and joining a few classifiers to achieve high accuracy is called boosting. Our model contains the following components as shown in Figure 3.

- Hardware Feature Extraction - Extracting the HPC events from both malware and benign ware.
- Feature selection – Selecting suitable features from 237 HPC features, which can efficiently decrease the redundant and inappropriate features in the original large HPC features. Feature selection often leads to increased detection accuracy and reduced false alarm rate.

- SIDS: Normal Model Creation - Model is created using the HPC features selected by the previous stage. Once the suitable subset of HPC features is nominated, this subset is then used in the classifier training phase where machine learning techniques are applied.
- AIDS: Anomaly Detection - Identifying abnormal behavior that differentiates from normal behavior shown in Figure 3.

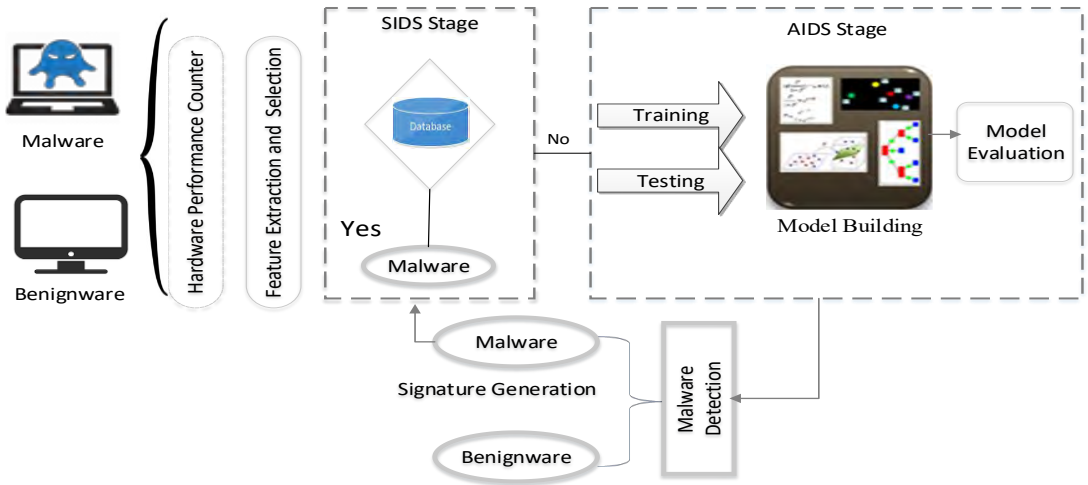


Figure 5.2 The high-level design of the HPC-based on Intrusion Detection System

Figure 5.2 shows Hybrid IDS that could overcome the disadvantage of SIDS and AIDS. Our system integrates SIDS and AIDS to create an efficient hybrid IDS to detect both unknown and known attacks. In our approach, AIDS is used to identify zero-day attacks, while SIDS is used to identify well-known attacks. The central idea of the novel approach is to combine the strengths and advantages of SIDS and AIDS to build further efficient IDS.

AIDS aims to profile the normal applications activity that would be accepted and would raise a malicious alarm when the difference between a given observation at an instant and the normal request exceeds a predefined feature selection threshold.

Applications profile is created by employing records that are recognised as benign actions. Next, it observes the behaviour of the traffic and matches the new records with the built profiles and attempts to identify abnormalities. If a malicious request is identified, the system will save it in the signature database. The primary purpose of storing the malicious pattern in the database is to achieving protection against a similar attack in the future.

5.6.1 Feature Extraction

Feature extraction should result in a set of non-redundant and low-dimensional features from the HPCs. The HPC features applied in this study describe the typical behavior of an application. They include various metrics, such as CPU Cycle, branch-instruction, and branch misses. Table 5. 6 shows a sample of HPC features which are extracted at the program execution stage. A kernel driver that configures the Hardware Performance Counters was developed and written for collecting the microarchitectural events and storing them in a database, which contains an execution of a million instructions to form the selected feature.

Event Name	Description
ALL_BRANCHES	All (macro) branch instructions retired.
CONDITIONAL	Conditional branch instructions retired.
CONDITIONAL_PS	Conditional branch instructions retired.
NEAR_CALL	Direct and indirect near call instructions retired.
NEAR_CALL_PS	Direct and indirect near call instructions retired.
NEAR_CALL_R3	Direct and indirect macro near call instructions retired (captured in ring 3).
NEAR_CALL_R3_PS	Direct and indirect macro near call instructions retired.
ALL_BRANCHES_PEBS	All (macro) branch instructions retired.

NEAR_RETURN	Return instructions retired.
NEAR_RETURN_PS	Return instructions retired.
NOT_TAKEN	Not taken branch instructions retired.
NEAR_TAKEN	Taken branch instructions retired.
NEAR_TAKEN_PS	Taken branch instructions retired.
FAR_BRANCH	Far branch instructions retired.

Table 5. 6 HPC Features

Figure 5.3 shows a comparison of the distribution of events from normal runs versus different malware. Variations are cleared from baseline features because of the changing phases of malicious execution. In a box plot, we draw a box from the first quartile to the third quartile. In malicious activities, it has two malicious activities, namely, the application that hid the process and the application that used TCP hooking to hide the TCP ports from netstat.exe. In normal applications, normal applications were executed without any malicious activity.

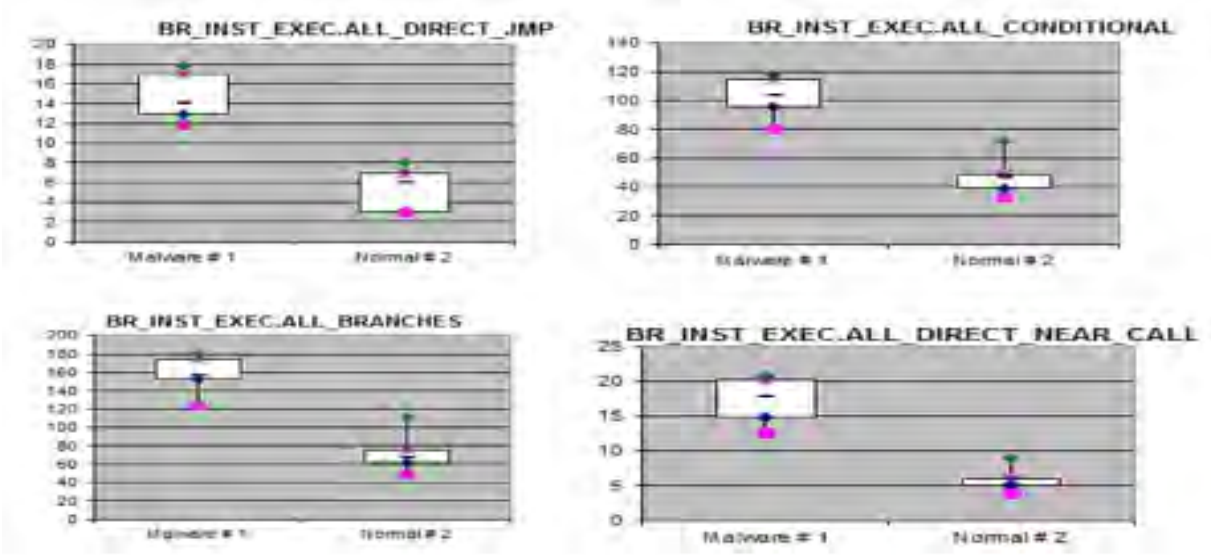


Figure 5.3 Comparison of the distribution of events from normal runs versus different malware

A lot of extracted features can cause many problems, for example, misleading the classification algorithm, over-fitting, decreasing generality, rising model complexity and the execution time.

These aspects can impact negatively when applying machine learning techniques on IDSs. Every time the guest operating system needs to interact with the hardware, the Virtual Machine Monitor (VMM) must intervene. Therefore, HPC features are extracted from this intervention. Hence, any abnormality in performance traces can be used as signs of potential attacks. With the extracted labeled collection, the relevance features are analyzed for both normal and attack and to determine the most relevant feature.

HPC features are extracted from both normal and malicious application by using VTune, a tool primarily designed to help programmers in optimizing their applications [19]. For malicious activities, Zeus as malware is executed. In normal applications, normal applications are executed without any malicious activity.

5.6.2 Feature selection

One of the main issues with IDSs is dealing with many irrelevant features, which can cause overhead on the system. Therefore, most IDS apply feature selection. As some of the features might be unimportant, which leads to the false detecting of malware. Therefore, the purpose of the feature selection is to identify significant features that can be used in the IDS to detect various malware efficiently.

We applied an information gain method for feature selection, due to its fast execution time to select the best performing feature set for that particular type of model. Information gain is regularly used to reveal how well each distinct attribute separates the given data set. The overall entropy “I” of a given dataset “S” is described as (Gray, 2010).

$$I(S) = -\sum_{i=1}^{c} p_i \, log_2 \, p_i$$

Where C is the total number of classes and p_i is the portion of instances that belong to class i. The decrease in entropy or the information gain is calculated for each feature:

$$IG(S, A) = I(S) - \sum_{v \epsilon A} \frac{|S_{A,v}|}{|S|} I(S_v)$$

For both normal and attack classes, information gain is calculated, and we removed the feature set whose information gain was less than predetermined thresholds of 0.5. This calculation involved the estimation of the conditional probabilities of a class for a given a term and entropy computations. The feature with good information gain is considered the most discriminative feature. Table 6 presents the information gain of HPC features. In total, 237 features are examined. Among them, 22 HPCs are the most significant concerning malware detection. A higher rank points out that the feature distinguishes well between normal and malware applications. The features in Table 5. 7 are presented in the descending order of their contribution in identifying malware.

Ranked	Features
0.7814	BR_INST_RETIRED.ALL_BRANCHES_PS
0.6482	ITLB_MISSES.WALK_DURATION
0.6187	BR_INST_RETIRED.NEAR_CALL_R3_PS
0.6168	L2_RQSTS.DEMAND_DATA_RD_HIT
0.6083	BR_INST_EXEC.TAKEN_CONDITIONAL
0.6008	BR_INST_RETIRED.NEAR_RETURN_PS
0.5977	L2_RQSTS.REFERENCES

0.5965	BR_INST_RETIRED.ALL_BRANCHES
0.5902	L2_RQSTS.CODE_RD_HIT
0.583	BR_INST_EXEC.TAKEN_DIRECT_JUMP
0.5821	L2_LINES_IN.S
0.5794	BR_INST_RETIRED.NOT_TAKEN
0.5639	L2_RQSTS.CODE_RD_MISS
0.5612	L2_LINES_IN.E
0.5504	BR_MISP_EXEC.ALL_BRANCHES
0.5457	L2_RQSTS.ALL_DEMAND_REFERENCES
0.5327	L2_RQSTS.MISS
0.53	BR_INST_RETIRED.NEAR_CALL_R3
0.5252	OFFCORE_REQUESTS_OUTSTANDING.DEMAND_RFO
0.5183	MEM_LOAD_UOPS_L3_HIT_RETIRED.XSNP_NONE
0.5169	BR_INST_EXEC.TAKEN_DIRECT_NEAR_CALL
0.5098	BR_INST_RETIRED.NEAR_CALL

Table 5. 7 Information gain for different HPC features

5.6.3 Building Classification Models

Once HPC features are selected, we ran experiments using Hybrid IDS to evaluate their capability to distinguish between malicious and normal execution. In order to achieve high accuracy and low false alarm rates, we combined two different classifiers execution (Quinlan, 1996). The proposed model involves two phases: the SIDS phase uses C5 classifier, and AIDS stage one-class SVM is used. Each phase has two stages, training and testing. In the training stage, the normal activities profile is labeled by using standard programs that are accepted as normal behaviour; the testing stage new data set is used. The hybrid model can classify activities of new users either normal profile (no attack) or deviation from the normal profile (attack).

5.7 Performance Evaluation of the Proposed IDS

In this section, we provide the detailed results of the experiments achieved using the proposed framework with the selected HPC features.

5.7.1 Evaluation Metrics for Models

In this experiment, we have evaluated the effectiveness of our IDS using Confusion Matrix as shown in Table 5. 8. Various elements are:

- True positive (TP): Number of correctly identified malicious code.
- True negative (TN): Number of correctly identified benign code.
- False positive (FP): Number of wrongly identified benign code, when a detector identifies the benign file as a Malware.
- False negative (FN): Number of wrongly identified malicious code, when a detector fails to detect the Malware because the virus is new and no signature is yet available.
- Total Accuracy: Proportion of completely accurately classified instances, either positive or negative.
- The Detection Rate (DR) is calculated according to the following equation: $DR\ (i) = \frac{\mathrm{Xii}}{\sum_{i=1}^{6} \mathrm{Xij}}$ False alarm rate of IDSs can be computed by $F\ (i) = 1 - \frac{\mathrm{Xii}}{\sum_{i=1}^{6} \mathrm{Xij}}$

The accuracy is calculated according to the following equation:

$$Accuracy = (TP + TN)/(TP + FN + FP + TN)$$

Class	Predicted attack	Predicated normal
Actual attack	True positive (TP)	False negative (FN)
Actual normal	False positive (FP)	True negative (TN)

Table 5. 8 Confusion Matrix

Table 5. 9 shows the confusion matrix, the element $X(1 <= i <= 6; 1 <= j <= 6)$ denotes the number of records that belong to class i and were classified as class j by IDSs. Therefore, based on the confusion matrix, we can easily compute other performance criteria.

Classified as	a	b	C	d	e	f
a=DKOM	X11	X12	X13	X14	X15	X16
b=File Hiding IRP hooking	X21	X22	X23	X24	X25	X26
c=File Hiding using SSDT Hooking	X31	X32	X33	X34	X35	X36
d=port filtering using IRP Hooking	X41	X42	X43	X44	X45	X46
e=Process Hiding using SSDT hooking	X51	X52	X53	X54	X55	X56
f=Normal	X61	X61	X63	X64	X65	X66

Table 5. 9 Confusion matrix of our proposed IDSS

5.7.2 Experiment Results

As discussed in the feature selection section, we determine the specific feature for each of rootkit functionality, as well as the normal characteristics.

5.7.2.1 Stage one: SIDS Results

Confusion matrix results for C5 classifier in stage one is shown in Table 5. 11 malware signature, we applied C5 , a rule learning program, to our training data (Cohen, 1995). C5 is fast and creates accurate rule sets. Table 5. 10 presents generation c5 rules from the SIDS stage for classifying all diversity of malware.

……

……

IF (BR_INST_RETIRED.NEAR_RETURN_PS <= 8455) and (BR_INST_EXEC.ALL_INDIRECT_NEAR_RETURN >= 682) **THEN** Class=Malware

IF (BR_INST_EXEC.TAKEN_INDIRECT_JUMP_NON_CALL_RET >= 5546) and (INST_RETIRED.PREC_DIST >= 19694) THEN Class=Malware

IF (RESOURCE_STALLS.ROB <= 862) and (IDQ.ALL_DSB_CYCLES_4_UOPS <= 1878) **THEN** class2=Netfilter (68.0/0.0)

IF (BR_MISP_EXEC.ALL_BRANCHES <= 1280) and (BR_INST_RETIRED.NEAR_CALL_R3_PS >= 4825) and (ITLB_MISSES.WALK_COMPLETED <= 484) **THEN** class2=Netfilter

IF (MEM_LOAD_UOPS_RETIRED.L2_HIT_PS <= 958) and (MEM_UOPS_RETIRED.ALL_LOADS_PS <= 8539) **THEN** class2=file_ssdt

IF (BR_INST_EXEC.ALL_CONDITIONAL <= 17987) and (BR_INST_RETIRED.ALL_BRANCHES_PS >= 12354) **THEN** class2=file_ssdt

IF (ICACHE.MISSES <= 1627) **THEN** class2=Process_ssdt

IF (BR_INST_RETIRED.NEAR_TAKEN_PS <= 12410) **THEN** class2=file _irp (99.0/0.0)

IF (DTLB_STORE_MISSES.STLB_HIT_2M >= 467) and (BR_MISP_RETIRED.NEAR_TAKEN >= 708) **THEN** class2=Dkom (100.0/0.0)

……

……

Table 5. 10 Malware signature generation samples

Classified as	**a**	**b**	**c**	**d**	**E**	**f**
a=DKOM	99	0	0	0	0	1
b=File Hiding IRP hooking	7	93	0	0	0	0
c=File Hiding using SSDT Hooking	1	0	91	5	2	0
d=port filtering using IRP Hooking	3	0	0	93	0	2
e=Process Hiding using SSDT hooking	2	0	5	0	92	0
f=Normal	4	0	0	0	0	685

Table 5. 11 Confusion Matrix results of using C5

The detailed analysis of the accuracy of C5 decision tree classification is shown in Table 5. 12.

Classified as	TP Rate	FP Rate	F-Measure
a=DKOM	0.49	0	0.658
b=File Hiding IRP hooking	1	0	1
c=File Hiding using SSDT Hooking	1	0.006	0.971
d=port filtering using IRP Hooking	1	0	1
e=Process Hiding using SSDT hooking	0.939	0	0.969
f=Normal	1	0.103	0.964
Weighted Avg.	0.952	0.06	0.945

Table 5. 12 Detailed analysis of accuracy by applying C5

Stage two: AIDs Results

One-class SVM with RBF kernel was implemented using LIBSVM. Results in the form of a confusion matrix of stage two are shown in Table 5. 13.

Classified as	a	b	c	d	E	f
a = Dkom	100	0	0	0	0	0
b=File Hiding IRP hooking	0	100	0	0	0	0
c=File Hiding using SSDT Hooking	0	0	99	0	0	0
d=port filtering using IRP Hooking	0	0	0	98	0	0
e=Process Hiding using SSDT hooking	0	0	0	0	99	0
f = Normal	100	0	0	0	0	589

Table 5. 13 Confusion Matrix results of using one-class support vector machine

The detailed analysis of the accuracy of One-Class SVM classifier are shown inTable 5. 14.

Class	TP Rate	FP Rate	F-Measure
a=DKOM	1	0.092	0.667
b=File Hiding IRP hooking	1	0	1
c=File Hiding using SSDT Hooking	1	0	1
d=port filtering using IRP Hooking	1	0	1
e=Process Hiding using SSDT hooking	1	0	1

f=Normal	0.855	0	0.922
Weighted Avg.	0.916	0.008	0.926

Table 5. 14 Detailed analysis of accuracy by applying one-class support vector machine

5.7.2.2 Stage Three: The Combination of the two stages

In Hybrid IDS, the C5 classifier is applied as a first stage, and one class SVM is employed as the second stage to create hybrid IDS. Stacking ensemble methods is used to combine those two stages. K-fold cross-validation is applied for assessing the results of a statistical analysis generating an independent dataset using 10 folds. For k=10 folds, 90% of full data is used for training and 10% for testing. Training and testing were performed using 10-fold cross-validation on all malware and benignware dataset. Confusion matrix of the combination of the classifiers in stage three is shown in Table 5. 15.

Classified as	**A**	**b**	**c**	**d**	**E**	**f**
a=DKOM	100	0	0	0	0	0
b=File Hiding IRP hooking	0	100	0	0	0	0
c=File Hiding using SSDT Hooking	0	0	95	1	2	1
d=port filtering using IRP Hooking	0	0	0	98	0	0
e=Process Hiding using SSDT hooking	0	0	3	0	96	0
f=Normal	3	0	0	0	0	686

Table 5. 15 Confusion matrix by using Hybrid classification

The accuracy of stage 3 is shown in Table 5. 16.

Class	**TP Rate**	**FP Rate**	**F-Measure**	**ROC Area**
DKOM	1	0.003	0.985	0.998
File Hiding IRP hooking	1	0	1	1

File Hiding using SSDT Hooking	0.96	0.003	0.964	0.978
port filtering using IRP Hooking	1	0.001	0.995	1
Process Hiding using SSDT hooking	0.97	0.002	0.975	0.984
Normal	0.996	0.002	0.997	0.997
Weighted Avg.	0.992	0.002	0.992	0.995

Table 5. 16 Detailed analysis of accuracy by applying the Hybrid classifier

As shown in Figure 5.4, the accuracy of detection of malware is 94.6% in stage one. While the accuracy of detection of malware is 91.56% in stage two. In stage 3, the accuracy result is 99.16%, which has improved significantly as compared to stage 1 and stage 2. Therefore, the general performance of the suggested framework is enhanced in terms of the detection rate and low false alarms rate in contrast to the current methods.

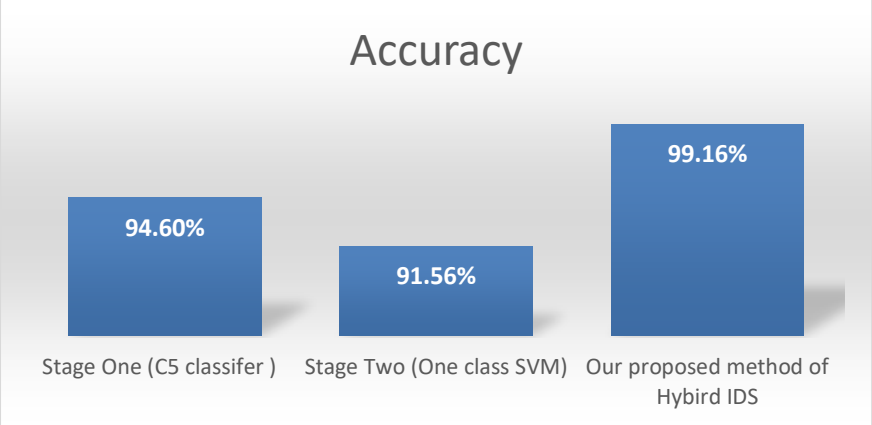


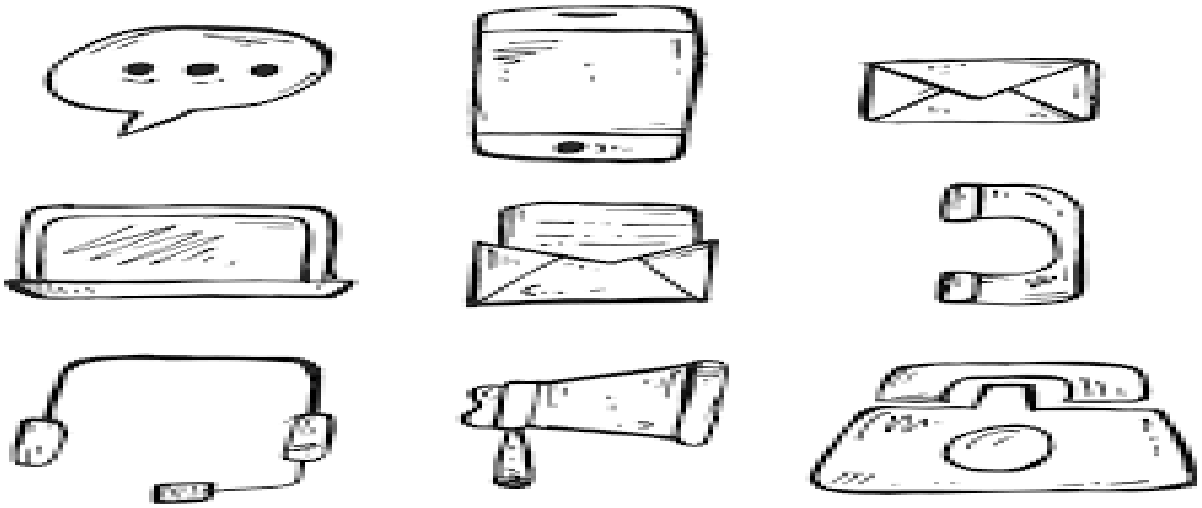
Figure 5.4 Accuracy details results of all stages

5.8 Conclusion

Hybrid intrusion detection system using low-level information such as HPCs would be highly reliable against suspicious behavior that might be a zero-day attack. In addition, using HPC as a feature provides robustness for IDS as it is difficult for a malware author to evade detection

as high access privileges are required to attack at a lower level. In this chapter, we presented a Low-Level Intrusion Detection System based on HPCs, and the central idea is to use HPCs to build IDSs to detect malware. We have shown different kinds of HPCs feature that we were able to extract from the computer system and used them to build IDS. We also investigated and examined the significant HPCs features to differentiate between normal and abnormal activity. Our results show that our hybrid IDS improved intrusion detection as it can also detect known and unknown attacks as well. Paper has demonstrated with the use of these features to build hybrid IDS are more reliable because it is difficult for a malware author to change hardware information. Our proposed method has been evaluated with the use of recent windows rootkit. Our experimental results indicated that our approach could detect malware with a high detection accuracy of 99%. This demonstrates the value of using HPC and hybrid IDSs to detect malware attacks successfully.

Chapter 6 : On the Detection of Internet of Things Intrusion using Hybrid Intrusion Detection System



This chapter presents a Hybrid Intrusion Detection System which is novel in its ability to protect IoT infrastructure and detect any abnormal activities. This chapter presents the design, implementation, and evaluation of the proposed system. Results evaluating the performance of the system show that combining two stages by applying stacking ensemble methods improved the detection accuracy. Consequently, a combined model is advised to build IDS for IoT systems. This chapter has addressed Research Question#4. This work in this chapter has been published as the following paper:

A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, and A. Alazab, "A Novel Ensemble of Hybrid Intrusion Detection System for Detecting Internet of Things Attacks," vol. 8, no. 11, p. 1210, 2019.

TnP_Per_Dport	Total Number of packets per dport
AR_P_Proto_P_SrcIP	Average rate per protocol per Source IP (calculated by pkts/dur)
AR_P_Proto_P_DstIP	Average rate per protocol per Destination IP
N_IN_Conn_P_SrcIP	A number of inbound connections per source IP
N_IN_Conn_P_DstIP	Number of inbound connections per destination IP
AR_P_Proto_P_Sport	Average rate per protocol per sport
AR_P_Proto_P_Dport	Average rate per protocol per sport
Pkts_P_State_P_Protocol_P_DestIP	A number of packets grouped by the state of flows and protocols per destination IP
Pkts_P_State_P_Protocol_P_SrcIP	A number of packets grouped by the state of flows and protocols per source IP
Attack	Class label: 0 for Normal traffic, 1 for Attack Traffic
Category	Traffic category
Subcategory	Traffic subcategory

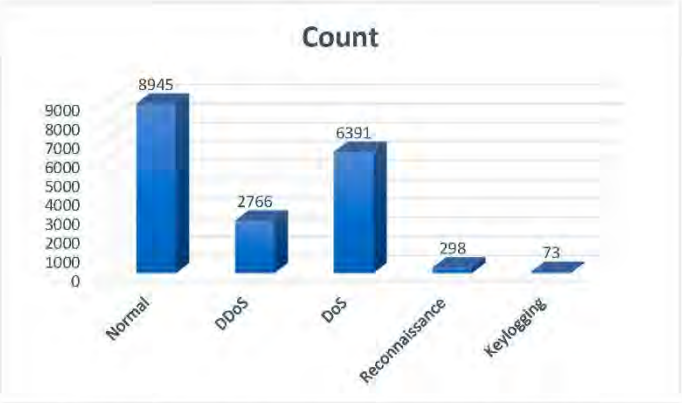


Figure 4. Statistics of attacks and normal behavior in the Bot-IoT dataset.

5.2. Evaluation Metrics for Models

In this experiment, we have evaluated the effectiveness of our IDS with the use of the Confusion Matrix as shown in Table 5. The following metrics were calculated:

True positive (TP): the number of rightly recognized malicious code.

True negative (TN): the number of rightly recognized benign code.

False positive (FP): the number of incorrectly identified benign code, when a detector recognizes a benign code as a Malware.

False negative (FN): the number of incorrectly recognized malicious code, when a detector recognizes a Malware as a benign code.

Total Accuracy: proportion of accurately classified instances, either positive or negative. The accuracy is calculated according to the following equation:

Table 4. Features for IoT networks.

Feature	Description
pkSeqID	Row Identifier
Stime	Record start time
Flags	Flow state flags were seen in transactions
flgs_number	Numerical representation of feature flags
Proto	A textual representation of transaction protocols presents in network flow
proto_number	Numerical representation of feature proto
Saddr	Source IP address
Sport	Source port number
Daddr	Destination IP address
Dport	Destination port number
Pkts	Total count of packets in a transaction
Bytes	Total number of bytes in the transaction
State	Transaction state
state_number	Numerical representation of feature state
Ltime	Record last time
Seq	Argus sequence number
Dur	Record total duration
Mean	The average duration of aggregated records
Stddev	The standard deviation of aggregated records
Sum	The total duration of aggregated records
Min	The minimum duration of aggregated records
Max	The maximum duration of aggregated records
Spkts	Source-to-destination packet count
Dpkts	Destination-to-source packet count
Sbytes	Source-to-destination byte count
Dbytes	Destination-to-source byte count
Rate	Total packets per second in transaction
Srate	Source-to-destination packets per second
Drate	Destination-to-source packets per second
TnBPSrcIP	Total Number of bytes per source IP
TnBPDstIP	Total Number of bytes per Destination IP
TnP_PSrcIP	Total Number of packets per source IP
TnP_PDstIP	Total Number of packets per Destination IP
TnP_PerProto	Total Number of packets per protocol

how well it models the dataset. The tree can then be applied as a rule set for detecting whether a test sample is malware or benign software.

Unknown traffic was handled through pattern matching to determine whether it represents normal or abnormal activities. If the request matches with an attack signature from the database, it raises an alarm that it is a malicious sample. If it did not match, it will go to AIDS, which is the next stage of the framework as shown in Figure 3, as AIDS is designed to detect unknown attacks, such as a zero-day attack.

4.4. Stage Two: AIDS Stage

In order to effectively recognize unknown attacks, the output of SIDS-stage is used to train AIDS to recognize abnormal activities. AIDS, being trained using benign samples, should be able to separate activities which do not appear to be normal, i.e., unusual behaviors exhibited by malware type software. To train AIDS, One-Class SVM is used, which learns the attributes of benign samples without using any information from the other class. Such a classifier can identify normal activities with far more success as normal class training data are easily available. In contrast, zero-day attacks are rare. Hence, we may have few instances of training datasets for zero-day attacks or even none.

Therefore, in the second stage, normal behavior is identified, and anything outside the normal behavior is classified as a zero-day attack. One-class classification techniques aim to build classification models when the malware class is unavailable, poorly sampled, or not well identified. The unique circumstances constrain the learning of efficient classifiers by describing class boundary just with the information of the normal class. In contrast to the traditional multi-class classification paradigm, in one-class classification, normal behavior is well described by examples in the training data, while the unknown malware has no example.

4.5. Stage Three: Stacking of the Two Stages

SIDS and AIDS have correlative qualities and shortcomings; thus, we propose to build up a hybrid method utilizing an ensemble of both approaches. In machine learning, ensemble techniques are used to enhance prediction accuracy. Although many ensemble methods have been proposed, this is a difficult task to find an appropriate ensemble configuration for detecting the zero-day attack. A C5 classifier was used as a first stage and a one class SVM was used as the second stage to create an ensemble of classifiers to improve accuracy for IDS.

5. Experimental Setup

The Bot-IoT dataset is used to evaluate the proposed hybrid IDS. The experiments have been performed using C5 and LIBSVM with default parameters. Details of the datasets are presented below.

5.1. Dataset

The Bot-IoT dataset, which includes normal IoT network traffic along with a variety of attacks, was used to evaluate our proposed framework. This dataset was selected because it represents a realistic IoT ecosystem environment. The dataset contains DDoS, DoS, OS and Service Scan, Keylogging, and Data exfiltration attacks. All these data were pre-processed to identify network-level patterns for diverse kinds of traffic that devices create; and use these patterns to detect any intrusion behaviors in the IoT Infrastructure [20]. The features and their descriptions are presented in Table 4 while the number of benign and attack samples in the dataset is shown in Figure 4.

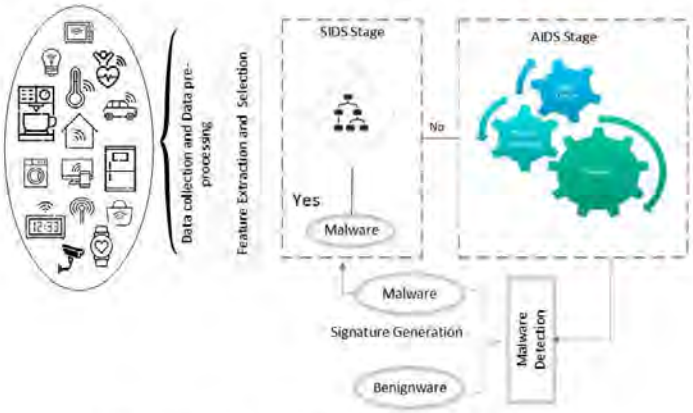


Figure 3. Hybrid Intrusion Detection System for the IoT ecosystem.

4.1. Feature Selection

The IoT ecosystem is made up of smart devices with limited processing power, memory, energy, and communication range. One main issue among many others with IDSs is dealing with many irrelevant features, which can cause overhead on the system. It is well known that redundant, irrelevant features often lead to low detection rate. Therefore, the purpose of the feature selection is to identify significant features which can be used in the IDS to detect various attacks efficiently [38].

With the extracted labels, the features are analyzed for both normal and abnormal behaviors to determine the most relevant features. We applied an information gain method for feature selection. The information gain methods had a fast execution time and this technique extracted the best performing feature set for the particular type of model. In literature, information gain was regularly applied to assess how well each distinct attribute separates the given data set. The overall entropy "I" of a given dataset "S" is described as [39]:

$$I(S) = -\sum_{i=1}^{c} p_i \log_2 p_i \tag{1}$$

where, "C" refers to the total number of classes and p_i denotes the portion of instances that belongs to class i. The decrease in entropy or information gain is calculated for every feature according to:

$$IG(S.A) = I(S) - \sum_{v \in A} \frac{|S_{A,v}|}{|S|} I(S_v) \tag{2}$$

where v values of A and $S_{A,v}$ are the instances of a set.

4.2. Building Classification Models

Once the selected features are identified, we ran experiments using Hybrid IDS to evaluate their capability to distinguish malicious activities from normal activities. Our Hybrid IDS model involved two phases, namely SIDS and AIDS.

4.3. Stage One: SIDS Stage

In the SIDS phase, C5 decision tree classifier was used to create a decision tree [40]. Once a decision tree is created, it can be applied to detect other samples with varying success depending on

Patil and Modi [33] designed a virtual environment monitoring system to prevent intrusions in IoT. This system used predefined signature database for known attacks and it applies anomaly-based detection for unknown attacks.

Table 3 shows the IDS techniques and datasets covered by this paper and previous research papers.

Table 3. Comparison of this research and similar researches: (✔ : Topic is covered, ✘ the topic is not covered).

Related researches	Intrusion Detection System Techniques						IoT Dataset
	SIDS	AIDS				Hybrid IDS	
		Supervised learning	Unsupervised	Semi-supervised learning	Ensemble methods		
Kenkre et al. [34]	✔	✘	✘	✘	✘	✘	✘
Hodo et al. [25]	✘	✔	✘	✘	✘	✘	✔
Liao et al. [35]	✔	✔	✔	✘	✘	✔	✘
Ashfaq et al. [36]	✘	✘	✘	✔	✘	✘	✘
Al-Yaseen et al. [37]	✘	✘	✘	✘	✘	✔	✘
This paper	✔	✔	✘	✘	✔	✔	✔

4. Proposed Hybrid Model for IDS

Hybrid IDS has been proposed to overcome the shortcomings of SIDS and AIDS, as it brings together SIDS and AIDS to identify both unknown and known attacks. Novel techniques were used to combine the results of SIDS and AIDS. In our methodology, AIDS was utilized to recognize zero-day attacks, while SIDS was utilized to distinguish well-known attacks. Boosting method was used to combine the classifiers and to decrease the bias of the combined model. The Hybrid IDS has two stages; the SIDS stage and AIDS stage, as shown in Figure 3. AIDS aims to profile the normal nodes activity and would raise a malicious alarm when the difference between normal requests exceeds the predefined threshold for a given observation. Nodes' profiles were created by employing records that were recognized as benign actions. Next, it observed the behavior of the traffic and matches the new records with the built profiles and attempts to identify abnormalities. If any malicious request was identified, the system will save it in the signature database. The main purpose of storing the malicious pattern in the database was to achieve protection against the similar attacks in upcoming malicious activity. In other words, the SIDS will have an appropriate history of previously known attacks.

the IoT ecosystem should operate under rigorous settings of low processing ability, high speed connection, and big capacity data processing.

Rathore et al. proposed semi-supervised Fuzzy learning based distributed attack detection framework for IoT [28]. The evaluation was done on the Network Security Laboratory - Knowledge Discovery in Databases (NSL-KDD) dataset and consequently suffers from the same dataset limitations as mentioned above.

Cho et al. proposed a methodology for checking packets that are passing through the border router for communication between physical and the network devices. Their methodology was based on the botnet attacks by checking the packet length [29]. However, no information is presented about the technique can be employed to create normal behavior profiles. It is also not clear how the proposed IDS techniques would work on resource constraints nodes in the IoT.

Table 2. Summary of the proposed research to IDSs for IoT.

Key references	Placement Strategy	Detection Techniques	Security Threat	Validation Strategy
Cho et al. [29]	Centralized	AIDS	Botnet	Simulation
Raza et al. [23]	Hybrid, centralized and distributed	Hybrid	Routing attacks	Simulation
Rathore and Park [28]	Distributed	AIDS	Network attack	Empirical (NSL-KDD Dataset)
Hodo et al. [25]	Centralized	AIDS	DoS attack	Simulation
Diro and Chilamkurti [27]	Distributed	AIDS	Network attack	Empirical (NSL-KDD Dataset)
Moustafa et al. [30]	Distributed	Hybrid	The botnet, Man in the Middle	Empirical
Cervantes et al. [31]	Distributed	Hybrid	Sinkhole attacks	Simulation
Venkatraman and Surendiran [32]	Distributed	Hybrid	DoS, control hijacking and replay	Simulation

Moustafa et al. proposed an ensemble of IDSs to detect abnormal activities, in specific botnet attacks against Domain Name System (DNS), Hyper Text Transfer Protocol (HTTP), and Message Queue Telemetry Transport (MQTT) [30]. Their ensemble methods are based on the AdaBoost learning method and they used three machine learning techniques: Artificial Neural Networks (ANN), Decision Tree (DT), and Naive Bayes (NB) to evaluate their methodology [30]. The proposed IDS results in significant overhead, which degrades its performance.

Hodo et al. used an Artificial Neural Network (ANN) to detect DDoS and DoS attacks against legitimate IoT network traffic. The proposed ANN model was tested with the use of a simulated IoT network. Hoda et al. proposed a threat analysis of IoT using ANN to detect DDoS/DoS attacks. A multi-level perceptron, a type of supervised ANN, was trained using internet packet traces, and then, the model was assessed on its ability to thwart (DDoS/DoS) attacks [25]. Hoda et al. did not consider the effectiveness after the deployment of the proposed IDS in the IoT ecosystem on low capacity devices. According to their experimentation, the system achieved an accuracy of 99.4% for DDoS/DoS. However, no details of the dataset are provided.

Cervantes et al. proposed IDS for detecting sinkhole attacks on 6LoWPAN for the IoT. Their IDS approach applies a combination of anomaly detection and support vector machine (SVM). Each IDS agent trains the SVM, and executes a majority voting decision to mark the infected nodes [31]. Their simulation results showed that their IDS achieve a sinkhole detection rate up to 92% on the fixed scenario and 75% in a mobile scenario. However, their approach has not been evaluated for other types of attacks in the IoT.

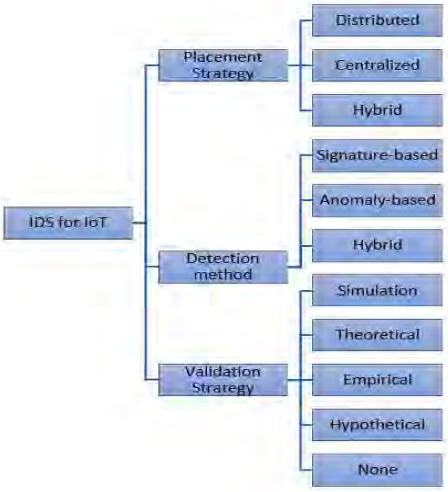


Figure 2. Classification of Intrusion Detection Systems for IoT.

Raza et al. used a hybrid, centralized, and distributed approach and placed IDS modules both in the border router and in the nominated nodes [23]. They applied signature- and anomaly-based techniques to detect routing attacks, where an attacker provides nearby nodes with false routing data and then modifies the data that transmit through it.

Current works on IDS for IoT have three primary classes: Anomaly-based Intrusion Detection System (AIDS), Signature-based Intrusion Detection Systems (SIDS), and hybrid. In short, SIDS relies on pattern matching techniques for finding known attacks; these are also known as Knowledge-based Detection or Misuse Detection [24]. In SIDS, matching methods are used to find a previous intrusion. In AIDS, a normal model of the behavior of a computer system is determined using machine learning, statistical-based or knowledge-based methods. Any significant deviation between the observed behavior and the model is regarded as an anomaly, which can be interpreted as an intrusion. The assumption for this group of techniques is that malicious behavior differs from typical user behavior, while the Hybrid IDS methodology combines SIDS with AIDS to improve the detection rate and decrease false alarms.

To validate the effectiveness of IDSs, researchers have used different techniques, such as theoretical, empirical, and hypothetical strategies, for validating their techniques.

Hoda et al. used AIDS based on a neural network for detecting Denial of Service attacks over the IoT networks. Their IDS approach was based on categorizing normal and abnormal patterns. The AIDS model was tested against a simulated IoT network [25].

Diro et al. developed an IoT network attack detection system on the basis of distributed deep learning. Their work showed that distributed attack detection could identify IoT attacks better than a centralized strategy with 96% detection rate. Their approach was evaluated using NLS-KDD dataset. Even though this dataset is another version of the KDD data set, it still suffers from various issues reviewed by McHugh [26]. We believe this dataset should not be used as an effective benchmark dataset in the IoT, as this data was collected from the traditional network [27]. This leads us to develop IDSs that take in consideration the specific requirement of IoT protocol such as Low-power Wireless Personal Area Networks (6LowPAN). Hence, intrusion detection system that is created for

Privilege escalation	The attacker takes advantage of programming errors or software flaws to permit cybercriminals to elevate access to IoT infrastructure.	Grant the cybercriminal elevated access to the IoT ecosystem and its associated data and applications.	[16]

An IoT botnet consisting of exposed IoT devices, such as electronic appliances, security systems, cars, thermostats and lights in private or commercial environments, speaker systems, alarm clocks, vending machines, and many other can be affected by the intrusion attacks. These intrusions permit a cybercriminal to control the sensors. Dissimilar to conventional botnets, affected IoT devices search to spread their malicious activity to an ever-increasing number of devices. A conventional botnet may comprise thousands of bots, but IoT botnet is bigger in scale, with a large number of attached devices [20]. For example, a large DNS server company called Dyn was targeted by cyber attackers on October 21, 2016. This attack was actually launched by an extraordinarily large number of DNS lookup requests from tens of millions of IP addresses [21]. The requests from the Botnet infected a large number of internet connected devices like printers, digital cameras, and other devices. This IoT botnet attack was caused by malicious software named Mirai. Due to Mirai infection, computers persistently browse the internet for devices that are vulnerable and use default username and password to access the system, infecting them with malicious software.

At Black Hat 2015, security researchers revealed how they attacked Chrysler Jeep Cherokee. While attacking the Jeep's system of IoT devices and sensors, one could remotely control a Jeep as it drives down the highway [22].

3. Related Work

In this section, a review of the existing IDS research for IoT is presented. Each research was categorized by considering the following characteristics: IDS placement strategy, detection method, and validation strategy. Figure 2 shows the classification of IDS for IoT networks, while Table 2 provides some recent related research.

In IDS placement strategies, IDS can be classified as distributed, centralized, or hybrid. In distributed placement, the IoT devices could be responsible for checking other IoT devices.

In the centralized IDS location, the IDS is placed in central devices, for instance, in the boundary switch or a nominated device. All the information that the IoT devices collect and then send to the network boundary switch passes through the boundary switch. Consequently, the IDS positioned in a boundary switch can check the packets switched between the IoT devices and the network. In spite of this, checking the net-work packets that pass through the boundary switch is not adequate to identify anomalies that affect the IoT devices.

A Man-in-the-Middle attack (MitM)	Man-in-the-middle is a type of eavesdropping attack. This attack could permit the attacker secretly relays and possibly alters the communications between two IOT devices.	Attackers have used a network packet analyzer, i.e., Wireshark for analyzing network traffic. IoT device communicates with other IoT devices. This connectivity is not encrypted or even authenticated. That is why it is very easy for an attacker to target access to the network, thereby allowing them mount attack such as Address Resolution Protocol (ARP) poisoning.	[13]
Reconnaissance	The aim to find data about an IoT infrastructure, including the network services and devices that are running	This can be achieved by scanning network ports and packet sniffers	[15]
Connected Device—Denial of Service (DoS)	Electronic devices and its connected devices are deactivated or changed by a cybercriminal, via physical or remote access to the IoT sensors.	An attacker can deny the sensors the capability to send and receive communication. Another example could be battery abuse, device disabling, or device bricking.	[16]
Server-side Denial of Service (DoS)	Server-side functionality, set to assist smart-devices, is affected and blocked by an attacker, attacking the sensor from his own smart-device.	DoS can flood devices with overwhelming traffic.	[17]
IoT Botnet	Group of hacked computers, smart devices, and appliances connected to the Internet are known as IoT botnet, these devices are the one chosen for attacks. They mainly attack online clients and devices such as IP cameras and home routers.	The Mirai malware is seen as a milestone in the threat landscape and exploits security holes in IoT devices and launches attacks [18].	[19]

- Gateway to data systems: data sent from a gateway to a suitable data system.
- Between data systems: information transmission within data centers or clouds.

2.1. IoT Threat Model

The rapid increase of the IoT adoption also increases the number of security threats that cybersecurity researchers must consider in order to devise a robust IDS. Several types of malicious activities try to attack the security and privacy of the IoT devices and potentially all smart devices on the publicly accessible Internet can be a target. The IoT is vulnerable against attacks for a number of reasons. Firstly, IoT devices are often unattended (e.g., sensors positioned in remote locations) and in this way, this makes it very easy for an attacker to gain access to them physically. Secondly, the greater part of the data communication is wireless, which makes it easier to eavesdrop. Lastly, the majority of the IoT devices have low storage capacities and limited processing capability. For example, additional security software could not be installed in the IoT devices.

Cybercriminals can interrupt or modify the behavior of smart devices using various hacking techniques [12]. Some of the hacking techniques need physical access to smart devices, making an attack harder to achieve, although not impossible given the physically unsecured nature of many IoT devices. Other attacks could be completed over the network from a remote site. Table 1 shows common attack types on attack smart devices.

Table 1. Common attack types which are used to attack smart devices.

IoT attack types	Description	Examples	References
Device attack	Defined as an attack in which someone takes advantage of any bug or vulnerability to gain access to the IoT infrastructure.	For smart IoT devices, such as security surveillance cameras, a cybercriminal could basically get physical access to the device and this will permit a cybercriminal to modify the design settings.	[13]
Attacks on Wi-Fi/Ethernet	Numerous malicious activities can be performed on smart IoT devices if an attacker gains physical access to the local network wirelessly.	In the network level attacks, cybercriminals are able to redirect network traffic, for example, Address Resolution Protocol poisoning (ARP) or by changing the Domain Name System (DNS) settings.	[13]
Cloud infrastructure attacks	IoT device interconnects with back-end cloud services. IoT cloud services might permit the client to select simple passwords.	A lot of cloud services have a logical weakness, which is actually the permission of cloud to a cybercriminal of obtaining sensitive information of the customer and also the access to the device without any authentication. Common vulnerabilities of management console are also contained in these services.	[14]

Smart devices can be connected via a wired or wireless connection. The wireless connections pose security challenges, as many diverse wireless communication methods and protocols could be applied to interconnect IoT devices. These technologies include Low power Wireless Personal Area Networks (6LoWPAN), ZigBee, Bluetooth Low Energy (BLE), Z-Wave, and Near Field Communication (NFC) [10].

Figure 1 shows the IoT system architecture with layers where attacks can occur. An IoT system can comprise three fundamental layers which are the perception layer, network layer, and application layer [11]. The perception layer is the lowest layer of the conventional architecture of IoT. This layer consists of devices, sensors, actuators, and controllers. This layer's fundamental task is to gather valuable information from IoT sensors systems. Network layer ensures the successful transmission of data while application layer is the highest layer that processes the data for visualization. This layer consists of various applications that essentially use the data provided by the underlying layers.

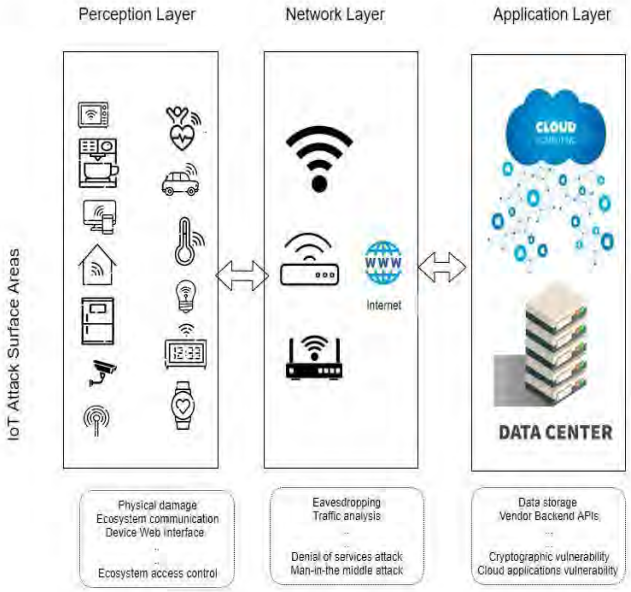


Figure 1. Internet of Things (IoT) architecture and layer attacks.

The data transfers among these levels takes place via following transmission channels:

- Device to device (D2D): peer to peer communication between two devices while using communication technologies such as Bluetooth, ZigBee, and Wi-Fi are common in the IoT system.
- Device to gateway: the gateway acts as a connection between the cloud and another node in IoT (e.g., controllers, sensors, and intelligent devices). All information to the data system is routed through the interconnected gateways. They have two main tasks: (i) to combine data from sensors and route it to the relevant data system; and (ii) to analyze data and, if a fault is detected, to initiate the recovery mechanism as per application's security requirements.

source IP addresses to hide attacks, so it becomes undetectable by the traditional IDS. Second, IoT specific features present a challenge for creating IDS. IoT devices are huge in number and need to host IDS agents; furthermore, low storage and computational capacity of IoT devices impose constraints on how IDS systems can be implemented. Third, another important issue is the characteristic associated with the IoT network design. In the traditional networks, the computer system is completely connected to specific computer nodes that are responsible for sending packets to the endpoints. In contrast, the IoT ecosystem communicates with numerous sensors and actuators to accomplish several monitoring and control tasks. IoT devices have significantly more varieties and type of networks than traditional networks. Therefore, applying traditional IDS detection system to IoT ecosystem is hard because of its specific features, such as limited resource, particular protocol stacks, and network requirements. For these reasons, an innovative hybrid IDS model has been proposed in this paper integrating SIDS and AIDS that can provide robust intrusion detection. Hybrid IDS is developed to counter the drawback of SIDS and AIDS, as it uses SIDS and AIDS to identify both zero-day and known attacks. The objective of the hybrid IDS is to overcome the limitations of the SIDS techniques and take advantage of the processing cost of the AIDS techniques. Therefore, HIDS has no negative impact on the node's energy consumption. However, current IDSs are not adequate to detect various attacks against the IoT systems, and they require high consumption of memory and processing. In our approach, AIDS is utilized to distinguish zero-day attacks, while SIDS is utilized to recognize known attacks. The key idea of our approach is to consolidate the benefits of both SIDS and AIDS to create robust IDS. The technique for creating and joining a few classifiers to achieve high accuracy is called boosting. SIDS is developed based on the C5.0 Decision tree classifier. Decision Trees are considered one of the most popular classification techniques. The decision tree is made up of nodes that shape a rooted tree, meaning it is a directed tree with a node called a "root" that has no incoming edges. The C5.0 decision trees provide outputs, using one attribute at a time to distinguish the data. New data can be categorized by sets of criteria defined at the nodes [8].

AIDS is developed based on a one-class Support Vector Machine (SVM). AIDS uses the known attack information and builds the profiles of normal behaviors of operations correctly. Our model contains the feature selection component for selecting suitable features, which can efficiently decrease the redundant and inappropriate features. Feature selection often leads to increased detection accuracy, reduced false alarm rate and reduced storage and computational capacity of IoT.

The main contributions of this paper are as follows:

- Development of feature selection technique based on information gain principle to select IoT features that result in maximum difference of features amongst all the applications profiled.
- Development of Hybrid Intrusion Detection System (HIDS) for IoT devices and gateways that uses a C5 classifier in the first stage and one class SVM in the second stage to create an effective ensemble architecture for improved accuracy. The experimental results show that the HIDS attains 99.97% accuracy of detection.

This paper is structured as follows. The background is provided in section 2. Related work is discussed in section 3. We present our approach to building models for the study in section 4. The experimental setup is presented in section 5. Lastly, the conclusion is presented in section 6.

2. Background

IoT is made up of smart devices that interconnect with one another. It permits the smart devices to gather and share information. IoT devices use a back-end cloud services for intensive processing to maintain remote control [9]. Clients are able to gain access to this data and control their devices through a mobile application or web-based interface. With a large number of sensors and actuators connected to the Internet, it is important to gather raw data and apply data mining techniques to extract more interesting information about the devices to develop efficient IDSs.

electronics

MDPI

Article

A novel Ensemble of Hybrid Intrusion Detection System for Detecting Internet of Things Attacks

Ansam Khraisat *, Iqbal Gondal, Peter Vamplew, Joarder Kamruzzaman and Ammar Alazab

Internet Commerce Security Laboratory, Federation University Australia, Mount Helen, Australia; iqbal.gondal@federation.edu.au (I.G.); p.vamplew@federation.edu.au (P.V.); joarder.kamruzzaman@federation.edu.au (J.K.); aalazab@mit.edu.au (A.A.)

* Correspondence: a.khraisat@federation.edu.au

Received: 5 September 2019; Accepted: 18 October 2019; Published: 23 October 2019

Abstract: The Internet of Things (IoT) has been rapidly evolving towards making a greater impact on everyday life to large industrial systems. Unfortunately, this has attracted the attention of cybercriminals who made IoT a target of malicious activities, opening the door to a possible attack to the end nodes. Due to the large number and diverse types of IoT devices, it is a challenging task to protect the IoT infrastructure using a traditional intrusion detection system. To protect IoT devices, a novel ensemble Hybrid Intrusion Detection System (HIDS) is proposed by combining a C5 classifier and One Class Support Vector Machine classifier. HIDS combines the advantages of Signature Intrusion Detection System (SIDS) and Anomaly-based Intrusion Detection System (AIDS). The aim of this framework is to detect both the well-known intrusions and zero-day attacks with high detection accuracy and low false-alarm rates. The proposed HIDS is evaluated using the Bot-IoT dataset, which includes legitimate IoT network traffic and several types of attacks. Experiments show that the proposed hybrid IDS provide higher detection rate and lower false positive rate compared to the SIDS and AIDS techniques.

Keywords: IoT; network; security; anomaly detection; zero-day malware; intrusion; intrusion detection system

1. Introduction

Internet of Things (IoT) is an interconnected system of devices that facilitate seamless information exchange between physical devices. These devices could be medical and healthcare devices, driverless vehicles, industrial robots, smart TVs, wearables and smart city infrastructures, and they can be remotely monitored and regulated [1,2]. IoT devices are expected to become more prevalent than mobile devices and will have access to the most sensitive information such as personal information [3]. This will result in increasing attack surface area and probabilities of attacks will increase. For instance, 'Mirai' is a botnet that mounted a Distributed Denial of Service (DDoS) attack that left much of the network unapproachable [4].

Due to the significance of IoT devices in our daily lives, it is crucial to develop IoT intelligent IDS capable of detecting both pre-known and zero-day attacks. As IoT devices are part of infrastructure, it makes them a target of cyber-attacks. Symantec reported a 600% increase in attacks against the IoT platforms in 2018 [5], which means that attackers are aiming to exploit the connected nature of these devices.

Intrusion Detection System (IDS) technology has originally been developed for traditional networks, and therefore, the current techniques IDSs for IoT are insufficient to detect different types of attacks for the following reasons [6]. First, the current IDS protect against known security threats, which means they are easily defeated by the new kinds of intrusions by attackers, as they can evade the traditional IDS [7]. For instance, the increased volume of DDoS attacks uses techniques that spoof

35. Liao, H.J.; Lin, C.H.R.; Lin, Y.C.; Tung, K.Y. Intrusion detection system: A comprehensive review. *J. Netw. Comput. Appl.* **2013**, *36*, 16–24.
36. Ashfaq, R.A.R.; Wang, X.Z.; Huang, J.Z.; Abbas, H.; He, Y.L. Fuzziness based semi-supervised learning approach for intrusion detection system. *Inf. Sci.* **2017**, *378*, 484–497.
37. Al-Yaseen, W.L.; Othman, Z.A.; Nazri, M.Z.A. Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system. *Expert Syst. Appl.* **2017**, *67*, 296–303.
38. Alazab, A.; Hobbs, M.; Abawajy, J.; Alazab, M. Using feature selection for intrusion detection system. In Proceedings of the 2012 International Symposium on Communications and Information Technologies (ISCIT), Gold Coast, Australia, 2–5 October 2012; pp. 296–301.
39. Gray, R.M. *Entropy and Information Theory*; Springer: Berlin/Heidelberg, Germany, 2010.
40. Khraisat, A.; Gondal, I.; Vamplew, P. An Anomaly Intrusion Detection System Using C5 Decision Tree Classifier. In *Trends and Applications in Knowledge Discovery and Data Mining*; Springer International Publishing: Cham, Switzerland, 2018; pp. 149–155.

12. Alazab, A.; Abawajy, J.; Hobbs, M.; Layton, R.; Khraisat, A. Crime toolkits: the productisation of cybercrime. In Proceedings of the 2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, Melbourne, Australia, 16–18 July 2013; pp. 1626–1632.
13. Barcena, M.B.; Wueest, C. *Insecurity in the Internet of Things*. Symantec: Mountain View, CA, USA, 2015.
14. Singh, J.; Pasquier, T.; Bacon, J.; Ko, H.; Eyers, D. Twenty Security Considerations for Cloud-Supported Internet of Things. *IEEE Internet Things J.* **2016**, *3*, 269–284.
15. Abomhara, M. Cyber security and the internet of things: vulnerabilities, threats, intruders and attacks. *J. Cyber Secur. Mobil.* **2015**, *4*, 65–88.
16. Bertino, E.; Islam, N. Botnets and internet of things security. *Computer* **2017**, *2*, 76–79.
17. Lu, Y.; da Xu, L. Internet of Things (IoT) cybersecurity research: a review of current research topics. *IEEE Internet Things J.* **2018**, *6*, 2103–2115.
18. Kolias, C.; Kambourakis, G.; Stavrou, A.; Voas, J. DDoS in the IoT: Mirai and other botnets. *Computer* **2017**, *50*, 80–84.
19. Sarang, R. Trending: IoT Malware Attacks of 2018. Available online: https://securingtomorrow.mcafee.com/consumer/mobile-and-iot-security/top-trending-iot-malware-attacks-of-2018/ (accessed on 3 March 2018).
20. Koroniotis, N.; Moustafa, N.; Sitnikova, E.; Turnbull, B. Towards the Development of Realistic Botnet Dataset in the Internet of Things for Network Forensic Analytics: Bot-IoT Dataset. *arXiv* **2018**, arXiv:1811.00701.
21. Mansfield-Devine, S. DDoS goes mainstream: how headline-grabbing attacks could make this threat an organisation's biggest nightmare. *Netw. Secur.* **2016**, *2016*, 7–13.
22. Greenberg, A. Hackers remotely kill a jeep on the highway—with me in it. *Wired* **2015**, *7*, 21.
23. Raza, S.; Wallgren, L.; Voigt, T. SVELTE: Real-time intrusion detection in the Internet of Things. *Ad Hoc Netw.* **2013**, *11*, 2661–2674.
24. Alazab, A., Hobbs, M., Abawajy, J., Khraisat, A., & Alazab, M. Using response action with intelligent intrusion detection and prevention system against web application malware. *Information Management & Computer Security*. 2014, *22*, 431–449.
25. Hodo, E.; Bellekens, X.; Hamilton, A.; Dubouilh, P.L.; Iorkyase, E.; Tachtatzis, C.; Atkinson, R. Threat analysis of IoT networks using artificial neural network intrusion detection system. In Proceedings of the 2016 International Symposium on Networks, Computers and Communications (ISNCC), Yasmine Hammamet, Tunisia, 11–13 May 2016, pp. 1–6.
26. McHugh, J. Testing Intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory. *ACM Trans. Inf. Syst. Secur.* **2000**, *3*, 262–294.
27. Diro, A.A.; Chilamkurti, N. Distributed attack detection scheme using deep learning approach for Internet of Things. *Future Gener. Comput. Syst.* **2018**, *82*, 761–768.
28. Rathore, S.; Park, J.H. Semi-supervised learning based distributed attack detection framework for IoT. *Appl. Soft Comput.* **2018**, *72*, 79–89.
29. Cho, E.J.; Kim, J.H.; Hong, C.S. *Attack Model and Detection Scheme for Botnet on 6LoWPAN*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 515–518.
30. Moustafa, N.; Turnbull, B.; Choo, K.R. An Ensemble Intrusion Detection Technique based on proposed Statistical Flow Features for Protecting Network Traffic of Internet of Things. *IEEE Internet Things J.* **2019**, *6*, 4815–4830.
31. Cervantes, C.; Poplade, D.; Nogueira, M.; Santos, A. Detection of sinkhole attacks for supporting secure routing on 6LoWPAN for Internet of Things. In Proceedings of the 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM), Ottawa, ON, Canada, 11–15 May 2015; pp. 606–611.
32. Venkatraman, S.; Surendiran, B. Adaptive hybrid intrusion detection system for crowd sourced multimedia internet of things systems. *Multimed. Tools Appl.* **2019**, 1–8.
33. Patil, R.; Modi, C. *Designing a Virtual Environment Monitoring System to Prevent Intrusions in Future Internet of Things*; Springer: Singapore, 2019; pp. 345–351.
34. Kenkre, P.S.; Pai, A.; Colaco, L. Real Time Intrusion Detection and Prevention System. In Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014, Bhubaneswar, India, 14–15 November 2014, Satapathy, S.C., Biswal, B.N., Udgata, S.K., Mandal, J.K., Eds.; Springer International Publishing: Cham, Switzerland, 2015; Volume 1, pp. 405–411.

Author Contributions: A.K. is the main author of the current paper. A.K. contributed to the development of the ideas, design of the study, theory, result analysis, and article writing. A.K. also designed the experiments and then performed the experiments. I.G., P.V., J.K., and AA undertook the revision works of the paper. All authors read and approved the final manuscript.

Funding: This work is done in Internet Commerce Security Lab, which is funded by Westpac Banking Corporation.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

IDSs	Intrusion Detection Systems
SIDS	Signature Intrusion Detection
AIDS	Anomaly Intrusion Detection System
AI	Artificial Intelligence
CPU	Central Process Unit
FN	False Negative
FP	False Positive
HIDS	Host-based Intrusion Detection System
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
HIDS	Hybrid Intrusion Detection System
IoT	Internet of Things

References

1. Sarkar, S.; Chatterjee, S.; Misra, S. Assessment of the Suitability of Fog Computing in the Context of Internet of Things. *IEEE Trans. Cloud Comput.* **2018**, *6*, 46–59.
2. Chowdhury, A.; Karmakar, G.; Kamruzzaman, J. The Co-Evolution of Cloud and IoT Applications: Recent and Future Trends. In *Handbook of Research on the IoT, Cloud Computing, and Wireless Network Optimization*; IGI Global: Hershey, PA, USA, 2019; pp. 213–234.
3. Saha, H.N.; Mandal, A.; Sinha, A. Recent trends in the Internet of Things. In Proceedings of the 2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA 9-11 January 2017; pp. 1–4.
4. Yar, M.; Steinmetz, K.F. *Cybercrime and Society*; SAGE Publications Limited: Thousand Oaks, CA, USA, 2019.
5. Symantec. Internet Security Threat Report. Available online: https://www.symantec.com/security-center/threat-report (accessed on 23 Feburary 2018).
6. Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J. Survey of intrusion detection systems: techniques, datasets and challenges. *Cybersecurity* **2019**, *2*, 20.
7. Ammar, A.; Michael, H.; Jemal, A.; Ansam, K.; Mamoun, A. Using response action with intelligent intrusion detection and prevention system against web application malware. *Inf. Manag. Comput. Secur.* **2014**, *22*, 431–449.
8. I. D. Mienye, Y. Sun, and Z. Wang. Prediction performance of improved decision tree-based algorithms: a review. *Procedia Manuf.* **2019**, *35*, 698–703.
9. Zarpelao, B.B.; Miani, R.S.; Kawakani, C.T.; de Alvarenga, S.C. A survey of intrusion detection in Internet of Things. *J. Netw. Comput. Appl.* **2017**, *84*, 25–37.
10. Al-Sarawi, S.; Anbar, M.; Alieyan, K.; Alzubaidi, M. Internet of Things (IoT) communication protocols: Review. In Proceedings of the 2017 8th International Conference on Information Technology (ICIT), Amman, Jordan, 17 May 2017, pp. 685–690.
11. Sonar, K.; Upadhyay, H. A survey: DDOS attack on Internet of Things. *Int. J. Eng. Res. Dev.* **2014**, *10*, 58–63.

Weighted Avg	0.94	0.002	0.957

As shown in Figure 5, the accuracy of detection of malware is 94% on the IoT intrusion dataset in stage one, while it is 92.5 % in stage two. In stage 3, the accuracy results have been improved to 99.97%. Therefore, the proposed framework yields higher detection accuracy and lower false alarm rate in contrast to the standalone single stage.

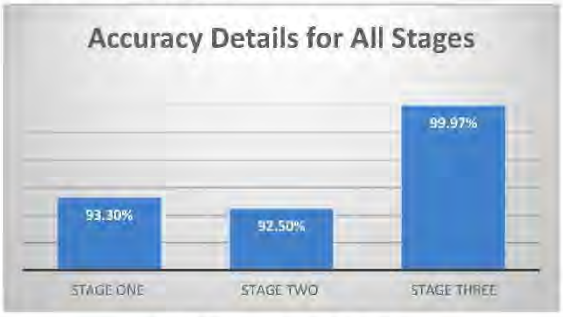


Figure 5. Accuracy details for all stages.

Table 14 shows the performance of different machine learning techniques, namely C4.5, Naïve Bayes, Random Forest, multi-layer perception, Support Vector Machine (SVM), Classification And Regression Tree (CART), and K Nearest Neighbor (KNN) on the Bot-IoT dataset. The results show that the combination of the two stages provides the best performance attaining an accuracy of 99.97%.

Table 14. The performance of different machine learning techniques.

Machine Learning Techniques	Accuracy
C4.5 [8]	92%
Naïve Bayes	87.56%
Random Forest	92.67%
Multi-layer perception	87.41%
SVM	89.52%
CART	80.3%
KNN	88.4%
Proposed Technique	99.97%

6. Conclusion

This paper presents the design, implementation, and evaluation of proposed novel IDS for intrusion detecting for IoT infrastructure. The proposed system relies on the feature set extracted from IoT ecosystem to effectively detect various types of IoT attacks. A set of features is used to create an effective Hybrid IDS for detecting IoT attacks. Experimental results show that combining two stages of the proposed framework through stacking ensemble method improves the detection accuracy. We have shown that an ensemble of C5 and one-class SVM in two cascaded stages is superior to individual techniques. Our experimental results show that our suggested hybrid IDS has superior performance overall in terms of accuracy and false alarm rate compared with the other machine learning techniques and approaches reported in previous studies. This suggests that our proposed technique will be very useful in designing modern IDSs. Future work includes extending the proposed IDS to detect other types of attacks against IoT systems.

Class	TP Rate	FP Rate	F-Measure
Normal	0.864	0	0.927
DDoS	0.996	0	0.998
DoS	0.999	0.001	0.998
Reconnaissance	0.997	0.067	0.327
Keylogging	1	0	1
Weighted Avg	0.933	0.001	0.953

Stage two: AIDs Results:

One-class SVM with RBF kernel was implemented using LIBSVM. Results in the form of a confusion matrix of stage two are shown in Table 10. Confusion matrix of using One-Class Support Vector Machine.

Table 10. Confusion matrix of using One-Class Support Vector Machine.

Classified as	a	b
a = Normal	7618	1327
b = Intrusion	4	9524

The detailed analysis of the accuracy of One-Class SVM classifier result is shown in Table 11.

Table 11. Detailed accuracy of using one class SVM.

Class	TP Rate	FP Rate	F-Measure
Normal	0.852	0	0.920
Intrusion	1	0.148	0.935
Weighted Avg	0.928	0.077	0.927

Stage Three: The Combination of the two stages:

In Hybrid IDS, the C5 classifier is applied as a first stage, and one class SVM is employed in the second stage to develop hybrid IDS. Stacking ensemble method is used to combine the two stages. Confusion matrices of the combination of the classifiers in stage three is shown in Table 12. The details accuracy of stage 3 is shown in Table 13.

Table 12. Confusion matrix with the use of Hybrid classification.

Classified As	a	B	c	D	e
a = Normal	7869	0	0	1076	0
b = DDoS	0	2737	29	0	0
c = DoS	0	0	6384	7	1
d = Reconnaissance	0	0	2	296	0
e = Keylogging	0	0	0	0	73

Table 13. Detailed accuracy of using stage 3.

Class	TP Rate	FP Rate	F-Measure
Normal	0.88	0	0.936
DDoS	0.99	0	0.995
DoS	0.999	0.003	0.997
Reconnaissance	0.993	0.06	0.353
Keylogging	1	0	1

- m is the number of classes.

Table 7 presents the information gain of IoT features. In total, 43 features are examined. Among them, 13 features are the most significant concerning malware detection. A higher rank of a feature makes it suitable to distinguish well between normal and malware applications. The features in Table 7 are presented in the descending order of their contribution in identifying malware.

Table 7. Information gain for different features.

Ratio	Feature Name
0.7579	dport
0.6433	seq
0.62	dur
0.4393	flgs_number
0.4381	flgs
0.3547	sport
0.3346	N_IN_Conn_P_DstIP
0.3023	srate
0.2873	AR_P_Proto_P_Sport
0.2817	daddr
0.2788	TnBPDstIP
0.2772	rate
0.274	AR_P_Proto_P_SrcIP

Stage one: SIDS Results:

We have used the k-fold cross-validation technique for performance evaluation. In this technique, the dataset is randomly divided into k different parts. For each iteration, one-fold was selected for testing and all other (k-1) folds were treated as the training dataset. For all experiments, the value of k was taken as 10 because of low bias, low variance, low overfitting, and good error estimation. The folds were stratified so that the class was characterized in approximately the same proportions as in the full dataset. Each fold was held out one by one and the learning scheme was trained on the remaining nine folds; then its error rate was calculated for the holdout set. The learning procedure was performed 10 times on different training sets, and finally, the 10 error rates were averaged to yield an overall error estimate.

To assess the performance of the proposed technique, the confusion matrix is used. Confusion matrix results for C5 classifier in stage one is shown in Table 8. The detailed analysis of the accuracy of C5 decision tree classification is shown in Table 9. Detailed accuracy of C5 decision tree.

Table 8. Confusion Matrix results of using C5 classifier.

Classified as	a	b	c	d	E
a = Normal	7728	0	0	1217	0
b = DDoS	0	2754	12	0	0
c = DoS	0	0	6384	7	0
d = Reconnaissance	0	0	1	297	0
e = Keylogging	0	0	0	0	73

Table 9. Detailed accuracy of C5 decision tree.

$$Accuracy = (TP + TN)/(TP + FN + FP + TN) \quad (3)$$

False alarm rate of IDSs can be computed by $F(i) = 1 - \frac{Xii}{\sum_{j=1}^{6} Xij}$

Table 5. Confusion matrix.

	Predicted Attack	Predicated Normal
Actual Attack	True positive (TP)	False Negative
Actual Normal	False Positive	True Negative

Table 5 shows the confusion matrix for two classes and Table 6 shows the confusion matrix for six classes, one normal and five attacks, the element $X(1 <= i <= 6; 1 <= j <= 6)$ denotes the number of records that belong to class i and were classified as class j by the IDS. On the basis of the confusion matrix, one can easily compute performance criteria, for example, the detection rate of class i: $DR(i) = \frac{Xii}{\sum_{j=1}^{6} Xij}$.

Table 6. Confusion matrix to evaluate the performance of our proposed IDS.

classified as	a	b	c	d	E	f
a = Normal	X11	X12	X13	X14	X15	X16
b = DDoS	X21	X22	X23	X24	X25	X26
c = DoS	X31	X32	X33	X34	X35	X36
d = Reconnaissance	X41	X42	X43	X44	X45	X46
e = Keylogging	X51	X52	X53	X54	X55	X56

5.3. Experimental Results

In this section, we provide the detailed results of the experiments using the proposed framework. The proposed model is applied to the IoT ecosystem, and its performance is evaluated against the other state-of-the-art machine learning techniques using the BoT-IoT intrusion dataset. To evaluate the system's accuracy for all stages, four measures were computed: true positive rate, F-measure, false positive rate, and accuracy.

5.3.1. Feature Selection Results

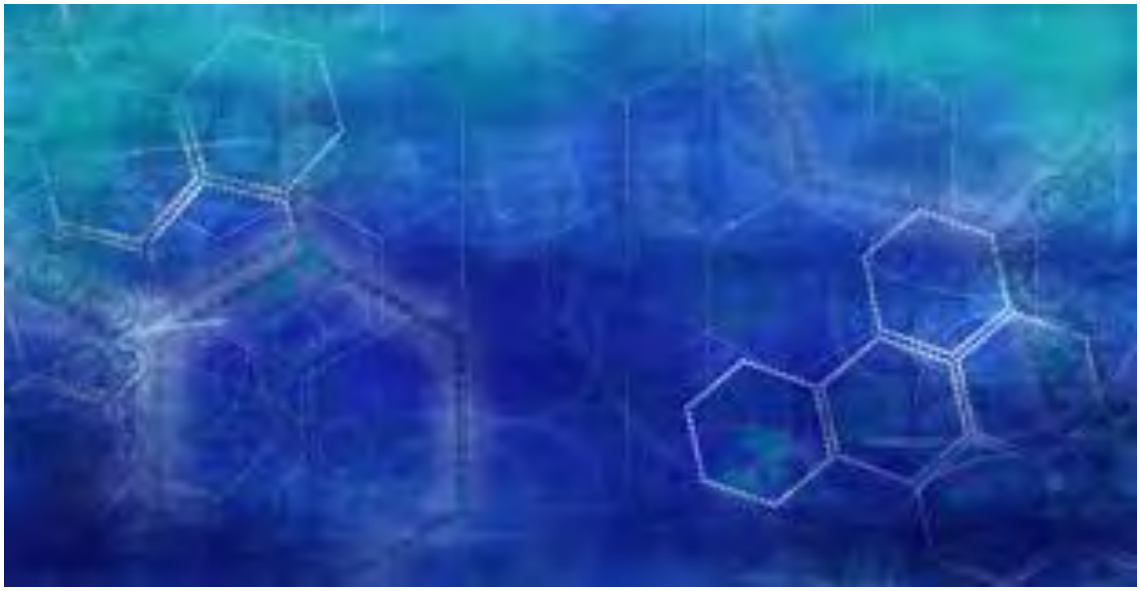
For both benign and malicious classes, information gain is calculated, and we removed the feature sets whose information gain was less than predetermined thresholds (set arbitrarily to 0.2). This calculation involves the estimation of the conditional probabilities of a class for a given feature, and entropy computations. The feature with good information gain is considered the most discriminative feature. For example, if the ranked value is higher than threshold then it indicates that a feature is useful for distinguishing this class, Otherwise, it will be eliminated from the feature sets. To get a better threshold value, the distribution of the Information Gain (IG) values is computed and verified with diverse threshold numbers on the training dataset.

$$IG(t) = -\sum_{i=1}^{m} P(c_i) \log_{P(c_i)} + p(t) \sum_{i=1}^{m} p(c_i|t) \log p(c_i|t) + P(\dot{t}) \sum_{i=1}^{m} P(c_i|t) \log \dot{P}(c_i|\dot{t}) \quad (4)$$

where:

- c_i represents (i) category.
- $P(c_i)$: probability that random instance belongs to class c_i
- $P(t)$ and $\overrightarrow{P(t)}$: probability of the occurrence of the feature w in a randomly selected document.
- $P(c_i|t)$: probability that a randomly selected instance belongs to class c_i if instance has the feature w.

Chapter 7 : Conclusions and Future Directions



7.1 Overview

The research presented in the previous chapters highlights that identifying both the well-known and the zero-day attacks with a high level of detection accuracy and low rate of false-alarm is achievable by developing Hybrid Intrusion Detection System. This chapter concludes the works presented in previous chapters and summarizes the contributions made in this thesis and highlights the possible future research directions.

7.2 Discussion

The future of cybersecurity of information systems is firmly associated with business continuity. The cyber-attacks will continue to increase as the cyber attackers are readily developing new malicious strategies to attack. This will become worse with the advancement

of IT technologies usable in our everyday life. The rise in attack sophistication means the cyber risks will keep on increasing and attacks will be damaging than ever before. Sophisticated cyber-attacks also are motivating factors for defenders to safeguard data and the resources of their organizations more effectively.

The current cybersecurity environment dictates that SIDS is not sufficient to prevent sophisticated and emerging intrusions to the systems. On the other hand, network and behavior analyses are vital to understanding attack techniques and can provide additional insight into network vulnerability. Achieving comprehensive protection from malicious attacks is becoming challenging with the help of existing detection techniques. The zero-day attacks get realized only after the attacks have occurred as there is no information available on these attacks. Therefore, it is vital to design attack detection systems capable of dealing with zero-day attacks. This thesis presents research that provides a solution to tackle zero-day attacks.

In Chapter 2 of this thesis, a survey of a comprehensive evaluation of recent works, and the datasets commonly employed for assessment purposes in this field are presented. We examined a few research issues that have been considered in regard to intrusion detection. This chapter also studies different well-known intrusion avoidance methods. An effective IDS should be able to detect different kinds of attacks accurately including intrusions that incorporate evasion techniques. Developing IDSs which are capable of overcoming the evasion techniques remains a major challenge for this area of research. Overall thesis contribution is the development of hybrid IDS that contains AIDS and SIDS as its components.

In chapter 3, we experimentally tested multiple data mining techniques to minimize both the number of false negatives and false positives. The objective of this chapter was to find out the best available classification technique for building AIDS. This study was implemented by analysing the NSL-KDD dataset and then observing the performance of classification algorithms. AIDS was able to identify and classify the attacks with low false alarms. It has been shown that AIDS can be very effective if it achieves a high detection rate and low false alarm. In this chapter, different machine learning techniques were applied to build efficient AIDS. The efficiency of C5 classifier was examined along with other classifiers. Our results showed a low alarm rate and IDS accuracy have improved.

Consequently, in Chapter 4 we proposed an intrusion detection framework to overcome the weaknesses of existing frameworks and to enhance detection accuracy, decrease the false-alarms rate with the ability to discover and identify the zero-day attacks intelligently. The main contribution of our framework was the integration of the Signature Intrusion Detection System with Anomaly Intrusion Detection System. In the proposed IDS, SIDS was applied to identify previously known intrusions and AIDS was applied to detect unknown zero-day intrusions. The goal of this framework was to exploit the individual strengths of each technique towards building a hybrid framework for better detection. A number of widely adopted metrics including accuracy and F-measure were applied to evaluate the efficacy of our proposed models and to compare with existing approaches. We showed that an ensemble of C5 and one-class SVM in two cascaded stages is superior to individual techniques. Our experimental results showed that our suggested hybrid IDS attained superior overall performance in terms of accuracy and false alarm rate compared with the other machine learning techniques and

approaches reported in previous studies. This suggests that our proposed technique will be very useful for modern IDSs.

In Chapter 5, it was demonstrated that the proposed hybrid IDS framework can work at hardware level attack detection as well. We experimentally showed how several types of intrusions and attack techniques manifest in Hardware Performance Counters signal variations. Studies have revealed the most significant HPCs which can be used in distinguishing malware. Our results show that the HPCs are most impacted by the techniques of attack such as system call hooking.

In chapter 6, the proposed framework was evaluated using IoT Botnet dataset, which includes legitimate IoT network traffic and several types of attacks. The result showed that our methods achieved high detection accuracy and a low false alarm rate.

7.3 Accomplishments

In this thesis, we have initially conducted a comparative study of various classifiers to act as an IDS and then we have developed a hybrid IDS framework which can detect known and zero-day attacks. Further, we have implemented our proposed framework on HPC to investigate the performance of the proposed framework at hardware level intrusions. Finally, we also investigated the performance of the hybrid IDs framework on IoT networks. This has shown the robustness of our proposed framework in detecting known and unknown attacks in various operating environments.

Chapter 2 presented an extensive literature review of IDS techniques and datasets used, and this chapter's work has been published as a journal paper.

It was shown in Chapter 3 that the C5 classifier is superior to other well-known classifiers as an IDS system. Experimental results demonstrated C5's superior performance overall in terms of accuracy and false alarm rate as compared with the other machine learning techniques and approaches reported in previous studies. This study established the strength of the C5 classifier in designing modern IDSs.

Chapter 4 proposed a hybrid IDS by combining C5 and One-Class Support Vector Machine classifiers. HIDS fuses the advantages of Signature intrusion detection systems and Anomaly-based Intrusion Detection System. The aim of this framework was to identify both the well-known and zero-day attacks with high detection accuracy and low false-alarm rates. The proposed HIDS was evaluated using the NSL-KDD and the Australian Defense Force Academy (ADFA) datasets. Our Studies showed improved performance of HIDS as compared to SIDS in terms of detection rate and low false-alarm rates. In the proposed IDS, SIDS was applied to identify previously known intrusions and AIDS was applied to detect unknown zero-day intrusions. The goal of this framework was to exploit the individual strengths of each technique toward building a hybrid framework for better detection. A number of widely adopted metrics including accuracy and F-measure were used to evaluate the performance of our proposed models and to compare with existing approaches. Our experimental results showed that the proposed hybrid IDS outperformed other machine learning techniques and approaches reported in previous studies in terms of accuracy and false alarm rate. For example, the results from Chapter 4 showed the accuracy of 81.5 % in detecting malware with the use of NSL-KDD Test+ dataset and 97.3% for ADFA dataset for SIDS stage. While the accuracy of detection of malware was 72.2 % with the use of NSL-KDD Test+ dataset and 76.4% for ADFA dataset for AIDS stage. For hybrid stage, the accuracy results improved to 83.2% and

97.4% respectively on the two datasets. Therefore, the proposed framework yields a higher detection rate and lower false alarms rate in contrast to the standalone single stage, concluding that HIDS enabled IDS can detect attacks more efficiently.

In Chapter 5, studies were presented on a hybrid framework for profiling normal and malicious activities based on HPCs. Extensive experiments were conducted to study the effectiveness of the HPCs that could distinguish between malware and normal applications. The performance of the suggested approach was tested on windows-based malware families. An Intrusion detection system using low-level information such as HPCs would be highly reliable against suspicious behavior that might be a zero-day attack. Using HPC as a feature provides robustness for IDS as it is difficult for a malware author to evade detection as high access privileges required at the hardware level. We also showed different kinds of HPC features as we were able to extract from the computer system and used them to build an IDS. We also investigated and examined the significant HPC features in differentiating between normal and abnormal activity. HPC based IDS are reliable because it is difficult for a malware author to change hardware information. Our proposed method was evaluated on recent windows rootkit. Experimental results indicated that our approach could detect malware with a high detection rate of number 99%. This demonstrates the value of using HPCs in an IDS and makes the detection more accurate.

Next, the proposed HIDS framework was evaluated using the Bot-IoT dataset, which includes IoT network traffic originating from several types of attacks. These experiments showed that the proposed hybrid IDS provided higher detection rate and lower false positive percentage as compared to the SIDS and AIDS techniques alone. Results also showed that combining two

stages by applying stacking ensemble methods improved the detection accuracy. Consequently, a hybrid model is advised to detect intrusions on IoT systems.

7.4 Future Work and Final Thoughts

The IDS systems proposed in the thesis can be used and applied to other domains and can be a base for future research. This section discusses several areas of research that can build on the work proposed work in this thesis. The potential areas for future research are summarized below:

More features: Future research can explore a wider variety of features of the network system, host system and IoT ecosystem for differentiating between normal and abnormal activities. Using features from the hardware level can result in more reliable IDSs because it is difficult for a malware author to change hardware information. The features need to be processed in real-time to ensure that an IDS can cope with high-speed network traffic. An IDS approach needs to detect the zero-day and complex attacks at the software level as well as at the hardware level without any previous knowledge. This can be done by exploring both hardware and software intrusion detection systems and extracting useful features of both HIDS and NIDS.

More Classifiers: The machine learning framework proposed in this research could include more classification techniques. Several other classifiers could be added to this list, including Logistic Regression and Ordinary Least Squares in the future research. Several Ensemble Methods can be investigated in future work to achieve the best possible performance.

Different System: Our proposed Hybrid IDS, which is based on C5 Classifier and One-Class Support Vector Machine, is applied on a software level, hardware level and IoT systems but it

has the potential in securing cloud computing, IoT ecosystem network, wireless network and other systems. Future research can explore those areas.

Time and Space Complexity: The major limitation for real-time implementation of such machine learning in malware detection is the time and space complexity. The disassembly, parsing and feature extraction are time-consuming processes and need a lot of memory space and CPU speed as well. In an automated Intrusion Detection system, these processes need to be streamlined for better efficiency.

Zero-day attacks: Machine learning techniques can handle tasks when normal and intrusion samples are available in training set in high numbers. But some intrusions are so infrequent that we may have only one instance of intrusion for training. This is typical for zero-day attacks. In this case, machine learning schemes that can support unbalanced datasets can be investigated with our framework.

Another possible future work could be to identify all kinds of intrusion that target IoT and cloud computing systems. Further, deep machine learning techniques can be applied for intrusion detection for zero-day malware. The future intrusion detection systems could identify intrusions while considering the behaviours of the malware for better accuracy.

Bibliography

1. Alazab, A., Abawajy, J., Hobbs, M., & Khraisat, A. (2013). Crime toolkits: The current threats to web applications. Journal of Information Privacy and Security, 9(2), 21-39.
2. Alazab, M. (2015). Profiling and classifying the behavior of malicious codes. Journal of Systems and Software, 100, 91-102. doi:https://doi.org/10.1016/j.jss.2014.10.031
3. Alazab, M., Venkatraman, S., Watters, P., Alazab, M., & Alazab, A. (2012). Cybercrime: the case of obfuscated malware. In Global Security, Safety and Sustainability & e-Democracy (pp. 204-211): Springer.
4. Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K. (2014). Network Anomaly Detection: Methods, Systems and Tools. IEEE Communications Surveys & Tutorials, 16(1), 303-336. doi:10.1109/SURV.2013.052213.00046
5. Branco, R. R., Barbosa, G. N., & Neto, P. D. (2012). Scientific but not academical overview of malware anti-debugging, anti-disassembly and anti-vm technologies. Black Hat.
6. Buczak, A. L., & Guven, E. (2016). A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. IEEE Communications Surveys & Tutorials, 18(2), 1153-1176. doi:10.1109/COMST.2015.2494502
7. Burguera, I., Zurutuza, U., & Nadjm-Tehrani, S. (2011). Crowdroid: behavior-based malware detection system for Android. Paper presented at the Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices, Chicago, Illinois, USA.
8. Cohen, W. W. (1995). Fast effective rule induction. In Machine Learning Proceedings 1995 (pp. 115-123): Elsevier.
9. de Melo, A. C. (2009). Performance counters on Linux. Paper presented at the Linux Plumbers Conference.

10. Demme, J., Maycock, M., Schmitz, J., Tang, A., Waksman, A., Sethumadhavan, S., & Stolfo, S. (2013). On the feasibility of online malware detection with performance counters. Paper presented at the ACM SIGARCH Computer Architecture News.
11. Deng, Z., Zhang, X., & Xu, D. (2013). Spider: Stealthy binary program instrumentation and debugging via hardware virtualization. Paper presented at the Proceedings of the 29th Annual Computer Security Applications Conference.
12. Dinaburg, A., Royal, P., Sharif, M., & Lee, W. (2008). Ether: malware analysis via hardware virtualization extensions. Paper presented at the Proceedings of the 15th ACM conference on Computer and communications security.
13. Egele, M., Scholte, T., Kirda, E., & Kruegel, C. (2012). A survey on automated dynamic malware-analysis techniques and tools. ACM Computing Surveys (CSUR), 44(2), 6.
14. Ehrenfeld, J. M. J. J. o. m. s. (2017). Wannacry, cybersecurity and health information technology: A time to act. 41(7), 104.
15. Fan, Y., Ye, Y., & Chen, L. (2016). Malicious sequential pattern mining for automatic malware detection. Expert Systems with Applications, 52, 16-25.
16. Garfinkel, T., & Rosenblum, M. (2003). A Virtual Machine Introspection Based Architecture for Intrusion Detection. Paper presented at the Ndss.
17. Gray, R. M. (2010). Entropy and information theory: Springer Verlag.
18. Huda, S., Abawajy, J., Alazab, M., Abdollalihian, M., Islam, R., & Yearwood, J. (2016). Hybrids of support vector machine wrapper and filter based framework for malware detection. Future Generation Computer Systems, 55, 376-390. doi:https://doi.org/10.1016/j.future.2014.06.001
19. Idika, N., & Mathur, A. P. J. P. U. (2007). A survey of malware detection techniques. 48, 2007-2002.
20. Khraisat, A., Gondal, I., & Vamplew, P. (2018). An Anomaly Intrusion Detection System Using C5 Decision Tree Classifier, Cham.
21. Khraisat, A., Gondal, I., Vamplew, P., & Kamruzzaman, J. (2019). Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecurity, 2(1), 20. doi:10.1186/s42400-019-0038-7

22. Khraisat, A., Gondal, I., Vamplew, P., Kamruzzaman, J., & Alazab, A. (2019). A Novel Ensemble of Hybrid Intrusion Detection System for Detecting Internet of Things Attacks. Electronics, 8(11), 1210.
23. Khraisat, A., Gondal, I., Vamplew, P., Kamruzzaman, J., & Alazab, A. J. E. (2020). Hybrid Intrusion Detection System Based on the Stacking Ensemble of C5 Decision Tree Classifier and One Class Support Vector Machine. 9(1), 173.
24. Kramer, M., Braverman, M., Seinfeld, M. E., Garms, J., Marinescu, A. M., Chicioreanu, G. C., & Field, S. A. (2010). System and method of efficiently identifying and removing active malware from a computer. In: Google Patents.
25. Lantz, B. (2019). Brett Lantz on implementing a decision tree using C5.0 algorithm. Retrieved from https://hub.packtpub.com/brett-lantz-on-implementing-a-decision-tree-using-c5-0-algorithm-in-r/
26. Lazarevic, A., Kumar, V., & Srivastava, J. (2005). Intrusion Detection: A Survey. In V. Kumar, J. Srivastava, & A. Lazarevic (Eds.), Managing Cyber Threats: Issues, Approaches, and Challenges (pp. 19-78). Boston, MA: Springer US.
27. Liao, H.-J., Lin, C.-H. R., Lin, Y.-C., & Tung, K.-Y. (2013). Intrusion detection system: A comprehensive review. Journal of Network and Computer Applications, 36(1), 16-24.
28. Malladi, R. K. (2009). Using Intel® VTune™ Performance Analyzer Events/Ratios & Optimizing Applications. http:/software. intel. com.
29. McAfee. (2016). McAfee Labs Threats Report: March 2016. Retrieved from http://www.mcafee.com/au/resources/reports/rp-quarterly-threats-mar-2016.pdf
30. Ozsoy, M., Donovick, C., Gorelik, I., Abu-Ghazaleh, N., & Ponomarev, D. (2015). Malware-aware processors: A framework for efficient online malware detection. Paper presented at the 2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA).
31. Pavlyushchik, M. A. (2014). System and method for detecting malicious code executed by virtual machine. In: Google Patents.
32. Quinlan, J. R. (1996). Learning decision tree classifiers. ACM Comput. Surv., 28(1), 71-72. doi:10.1145/234313.234346

33. Sayadi, H., Patel, N., S. M, P. D., Sasan, A., Rafatirad, S., & Homayoun, H. (2018, 24-28 June 2018). Ensemble Learning for Effective Run-Time Hardware-Based Malware Detection: A Comprehensive Analysis and Classification. Paper presented at the 2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC).
34. Shakshuki, E. M., Kang, N., & Sheltami, T. R. (2013). EAACK—a secure intrusion-detection system for MANETs. IEEE Transactions on Industrial Electronics, 60(3), 1089-1098.
35. Singh, B., Evtyushkin, D., Elwell, J., Riley, R., & Cervesato, I. (2017). On the detection of kernel-level rootkits using hardware performance counters. Paper presented at the Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security.
36. Stakhanova, N., Basu, S., & Wong, J. (2007). A taxonomy of intrusion response systems. International Journal of Information and Computer Security, 1(1), 169-184.
37. Stavroulakis, P., & Stamp, M. (2010). Handbook of information and communication security: Springer Science & Business Media.
38. Tang, A., Sethumadhavan, S., & Stolfo, S. J. (2014). Unsupervised anomaly-based malware detection using hardware features. Paper presented at the International Workshop on Recent Advances in Intrusion Detection.
39. Total, V. (2012). VirusTotal-Free online virus, malware and URL scanner. Online: https://www. virustotal. com/en.
40. Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep Learning Approach for Intelligent Intrusion Detection System. IEEE Access, 7, 41525-41550. doi:10.1109/ACCESS.2019.2895334
41. VirusTotal. Analyze suspicious files and URLs to detect types of malware Retrieved from https://www.virustotal.com/#/home/upload
42. VX Heavens. (2011). VX Heavens Site. Retrieved from http://vx.netlux.org/
43. Wang, X., & Karri, R. (2016). Reusing hardware performance counters to detect and identify kernel control-flow modifying rootkits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 35(3), 485-498.

44. Yan, L.-K., Jayachandra, M., Zhang, M., & Yin, H. (2012). V2E: combining hardware virtualization and softwareemulation for transparent and extensible malware analysis. ACM Sigplan Notices, 47(7), 227-238.
45. Zhang, F., Leach, K., Stavrou, A., Wang, H., & Sun, K. (2015). Using hardware features for increased debugging transparency. Paper presented at the 2015 IEEE Symposium on Security and Privacy.

These references are for the “Introduction, Chapter 5 and the Conclusions” sections and each individual chapter has its own bibliography in its associated paper.

Appendix

Definitions

Malware: It is a term used for any software or program that caused harm for the computer systems and affect its performance. Also described as any pace of code that has changed removed from the software system which aims to affect and damage the system functionality. There are many types of malware such as Viruses, Trojan, Worm, Ransomware and many more.

Zero-day attacks: it is referred to the new attacks or unknown attack, which is just created by the cybercriminals. Any new attack does not have any history of the detection system is a zero-day attack. In the cybersecurity, the intrusion detection system can't detect this attack as it doesn’t have a signature.

Cybercrimes: the type of crimes that take please on the computer system, includes any crime from accessing others information illegally, phishing email, theft, distribute viruses to hacking and spamming.

Zuse : it is one of the trojan horse packages that run on the Microsoft Windows .it is mainly used to steal the bank accounts

Buffer Overflow: it is an attack that occurs when cybercriminals overload systems. The attackers send code designed to cause a buffer overflow and hold original code and swap it with malicious code.

Denials-of Service (DoS): it is an attack that makes the network services not available. It is produced by forcing a reset on the machine or network resource and make them unavailable, then the users can't connect sufficiently because of the service unavailability.

User to Root (U2R) attack: The attackers gain access as a normal host first and then upgrades to the root access, which could lead to the exploitation of several vulnerabilities on the computer system.

Remote to Local (R2L) attack: The cybercriminal sends packets to a remote system by connecting with the network without having an account on the system.

Ensemble methods: it is the methods used to combine multiple machine learning algorithms and to achieve better prediction results. By combining the decisions from various machine learning models the overall performs will improve.

Dataset

The flowing section list sample of the dataset that we have collected and extracted from HPC features which was used on chapter 5:

Class CPU_CLK_UNHALTED.REF_TSC L2_RQSTS.ALL_PF
ICACHE.IFETCH_STALL BR_INST_RETIRED.NEAR_TAKEN
BR_INST_RETIRED.NEAR_TAKEN_PS BR_INST_RETIRED.NOT_TAKEN
BR_MISP_EXEC.ALL_BRANCHES
L1D_PEND_MISS.PENDING_CYCLES
L2_RQSTS.DEMAND_DATA_RD_HIT
L2_RQSTS.DEMAND_DATA_RD_MISS BR_INST_RETIRED.NEAR_CALL_R3
BR_INST_EXEC.TAKEN_INDIRECT_JUMP_NON_CALL_RET
MEM_LOAD_UOPS_L3_HIT_RETIRED.XSNP_NONE_PS
L2_RQSTS.ALL_RFO L2_LINES_OUT.DEMAND_DIRTY
L2_TRANS.L2_WB

0	5421706	1656 1875	2564 3146	7614	5923	8389	4182	822	0	4510	857
415	5804807	2414 1691	3084 2271	6874	8175	8980	4723	1795	833	3490	448
0	6027 1053	1004 862	4197 2364	9120	9070	12352	4272	2627	1237	3537	78
0	6792 554	979 1571	2420 2791	5546	6119	7769	1842	1097	307	3430	722
2103	6627 0	1162 0	3797 2027	8583 2277	8405	10412	3326	1029	372	1674	
0	6842 2117	1295 2654	3669 4207	7132	5314	9804	5002	1226	306	3324	803
0	5035 1488	614 2184	2908 3775	7295	7716	8143	2818	1527	0	2718	971
1117	5688 0	889 804	4538 2326	10222 5051	9594	13071	2428	2947	524	3965	
1767	6986 0	1448 459	3665 1387	6721 4317	6260	10889	3761	1026	409	3854	
0	7961 1617	1667 3442	6400 2615	9272	11227	13617	5867	1833	787	5164	417

1388	48960	7840	25362055	72733683	6758	7882	1852	2322	425	5785	
11924621	8461667	44423144	5820	6893	13044	4801	4354	0	4297	1694	
2199	4082776	1440844	33412250	83513500	8215	9537	2171	1872	0	2953	
1280	60250	13441951	41871616	72121928	8299	13239	6045	2051	0	4332	
0	57641259	13458864	48682869	8894	8696	12602	3666	2714	0	3085	999
0	5422964	13831290	41733775	8370	8196	10769	5168	1331	0	3085	461
0	46548232	12932126	41384402	10346	8884	12286	4093	2237	923	2303	868
0	3422428	4451685	23853886	6124	6247	8484	3469	1906	500	3072	822
0	53041214	13022619	26403149	6784	6124	9556	6776	1413	824	11340	623
1866	649800	529943	33702663	75592396	7080	9402	3090	1002	504	5024	
2434	585500	1207483	74342231	128503672	11347	13881	3488	2992	487	3247	
0	4069836	9471621	24492815	6657	7192	8801	6706	2180	386	3629	76
1873	516400	647755	41091704	81462135	7883	10465	3340	1027	0	4489	
1236	593600	49200	20093192	51321700	4899	7745	4584	2528	0	2923	
0	6203925	4591396	29093583	5102	5387	6985	1698	896	399	2843	531
1248	526200	1203783	52632139	109333175	9467	15563	4048	3023	544	3813	
1144	518600	1615470	32061152	79293053	6782	10789	3611	1790	914	2997	

1165	54950	976 1453	7865 2183	9413 4005	9504	13493	5332	2627	0	2500	
0	5815875	1367 2416	2816 3425	5179	5028	9939	4676	1842	377	11354	906
1610	43320	1473 754	3133 2084	6136 2891	6075	11185	4000	2914	859	3489	
1614	51300	1103 719	3052 1667	7204 3517	6904	9492	5875	1437	386	3228	
2447	57800	879 1299	2720 1950	8025 3547	8958	10990	2065	2187	394	3456	
461	52191091	1327 2225	4810 3561	9177	8962	15768	2699	2745	481	4876	398
0	7472456	475 2023	3475 2586	6631	6673	10200	3182	1978	0	5194	743
1260	67580	743 1021	3679 2168	7699 4595	6652	10274	3444	1805	471	11459	
0	6737451	556 2267	2540 2985	7099	7313	9981	5800	1721	0	2929	540
0	64021784	1940 1591	4266 2174	5206	7676	9548	5186	2159	433	1991	602
0	5135871	628 2922	4516 3716	11251	11126	13796	2234	2806	0	3364	906
1313	7269482	1198 1962	3477 1842	6245 3249	5798	10292	5217	1843	0	11279	
0	7414518	1570 3903	2826 2719	6790	6458	8323	3964	3619	738	4187	897
0	5751487	831 1908	2770 3759	6237	7029	11062	2326	2194	829	2080	846
1399	52380	534 1922	6436 1687	8779 3030	8857	12805	5769	3546	858	3850	
2239	4887682	1846 792	3054 794	6917 2850	6188	10050	5560	1430	404	3365	
2112	62200	1505 794	2853 1253	3944 3616	4194	6462	1858	2523	451	4729	

0	7760990	1126 1337	1898 2738	6542	7085	11759	4642	2592	0	2986	828
0	5538723	1791 1570	2701 3422	7569	5888	8500	4197	821	0	4598	883
445	5727803	2388 1974	3146 2214	6955	8376	9126	4591	1864	813	3213	458
0	5909854	1047 1170	4247 2323	9054	9005	12500	4292	2790	1357	3602	783
0	6709534	964 1436	2269 2942	5622	6340	7976	2154	993	300	3631	708
2117	67090	10280	3883 2494	8605 2225	8308	10522	3592	1099	387	1640	
0	6868 2209	1376 2543	3715 4152	7232	5393	9871	4870	1121	324	3380	799
1353	52430	613 1538	2803 1995	7151 3969	7698	8296	2565	1440	0	2750	
0	5828804	864 2457	4733 5122	9940	9655	12646	2256	3080	542	4123	949
1811	69890	1221 444	3414 1186	6609 4267	6014	10996	3532	1120	383	3574	
0	7995 1664	1728 3491	6300 2428	9124	11369	13554	5880	1727	785	5291	430
1639	47720	7820	2374 1983	7107 3974	6722	7813	2156	2411	421	5882	
1746	46570	848 1407	4294 1955	5576 3178	6909	13034	4819	4205	0	4591	
1919	3549778	823 869	2542 2138	7795 2875	8011	9045	1509	1258	0	2693	
0	5398 1560	957 980	3651 1259	6828	7743	12821	5206	1494	0	3938	991
0	5200983	804 857	4082 1947	8293	8446	12309	3446	2456	0	2586	996
0	5151943	968 700	3703 3070	7740	7636	10171	4423	1106	0	2856	470

0	4289811	837 1940	3650 3850	9891	8167	11628	3595	1906	950	1838	882
0	2874430	458 1071	1837 3406	5713	5558	8305	3044	1580	532	2412	825
0	4737927	861 1743	2263 2532	6028	5536	8917	6438	995	807	10619	647
1540	58270	517 912	2949 2131	7173 1970	6531	8984	2845	600	499	4565	
1969	56290	1006 471	6735 1767	12199 3355	10732	13278	3133	2490	499	2573	
0	3396846	953 735	1781 2339	6106	6753	8569	6026	1738	391	3119	767
1387	45490	664 736	3689 1269	7791 1517	7486	10017	2854	775	0	3858	
0	51500	492 2461	1688 1287	4425	4058	7064	3719	2073	0	2380	936
0	5881936	435 776	2428 3326	4596	5057	6575	1325	831	427	2270	527
0	4880772	744 1612	4555 2614	10611	8974	14880	3511	2614	541	3561	703
0	4787493	1314 875	2460 2562	7379	6166	10179	3067	1166	901	2690	893
0	5114854	988 1593	7483 3589	8978	8998	13049	4543	2014	0	1871	630
0	5434897	800 2015	2337 2780	4502	4736	9659	4085	1232	377	10788	881
1209	40520	822 756	2376 1510	5430 2342	5629	10530	3615	2256	841	2883	
1197	46220	799 688	2618 1254	7018 3043	6455	8730	5419	1019	403	2904	
2203	52460	886 937	2506 1257	7824 3288	8398	10538	1493	1686	373	2850	
469	4682949	636 2063	4394 3195	8499	8538	15427	2249	2371	491	4175	392

0	6971470	4631147	30322160	5931	6188	9374	2766	1649	0	4421	753
0	6590923	7431716	33013897	7258	6108	10053	2648	1490	491	10854	888
0	6051453	5861605	22682758	6613	6924	8997	4997	1390	0	2569	537
0	6236 1196	1414 1170	4117 1595	4777	7177	8955	4680	1641	458	1554	623
0	4610861	6212357	39303001	10608	10745	13368	1539	2299	0	2723	914
486	6632 1836	980 1412	2818 2576	5858	5208	9994	4609	1570	0	10971	848
0	6970507	9053415	20472230	6353	6157	7749	3134	3356	746	3725	892
0	5086515	8451610	23763533	5695	6458	10556	1847	1539	829	1450	840
1052	46590	5281289	57341308	8389 2350	8618	12255	5264	2674	879	3344	
1674	4347707	1087777	2381761	6266 2277	5419	9644	5048	1077	413	2974	
1545	56420	933805	2590880	3488 3210	3489	6026	1249	1873	456	4673	
0	7186996	826917	16482206	5782	6531	11037	4439	2177	0	2357	826
0	4846719	12471209	20942617	7220	5538	8186	3743	810	0	4149	881
444	5430781	19351171	25671894	6278	7838	8384	4326	1453	824	2830	453
0	5675783	875835	34841828	8444	8520	12003	3814	2433	830	3193	800
0	6145545	9571232	20342423	5024	5821	7313	1170	751	299	3003	721
1538	63550	7080	33501914	8100 1678	7633	9970	3300	957	396	1101	

0	6238 1848	803 2048	3031 3697	6369	4949	9373	4348	919	316	2830	793
0	4860 1176	611 1336	2289 3301	6647	7205	7980	2278	1143	0	1941	845
0	5271 807	894 2109	3888 4574	9574	9069	12525	2027	2430	545	3581	782
0	6781 531	956 1748	2779 2974	6000	6525	7987	2047	1415	297	3701	698
2434	6857 0	1448 0	4083 2332	8957 2255	8386	10629	3788	1422	390	1911	
0	7058 2218	1674 2791	3728 4432	7176	5790	10173	5054	1526	296	3667	781
1447	5448 0	601 1871	3125 2160	7595 4008	7745	8570	2769	1606	0	2777	
1186	6185 0	862 788	4838 2547	10299 5412	9851	13240	2554	3194	513	4361	
2012	7272 0	1366 435	3784 1546	6904 4524	6462	11154	3728	1505	389	3901	
0	8178 2106	2040 3606	6597 2845	9572	11553	13791	6105	2318	781	5731	409
1765	5261 0	771 0	2729 2392	7365 3984	7138	8259	2341	2637	409	607	
1732	4969 0	832 1535	4663 2003	5846 3220	7198	13270	4974	4469	0	4581	
2697	4556 783	1566 839	3391 2844	8795 3674	8542	9749	2211	2104	0	3377	
1628	6358 0	1506 2256	4714 1920	7576 1955	8417	13565	6191	2395	0	4449	
0	6076 1388	1421 841	4765 3092	8984	8948	12929	4134	3006	0	3260	985
0	5817 943	1543 1516	4353 4136	8593	8423	10941	5064	1593	0	3611	4
0	4943 809	1370 2566	4387 4698	10378	8900	12583	4448	2646	914	2663	851

0	3605428	4411792	25994210	6360	6519	8723	3701	2289	508	3047	810
0	56071586	15082505	29463429	6689	6409	9728	6928	1471	788	11476	620
2191	67580	517931	38622995	77672698	7391	9575	3298	989	499	5363	
2581	64430	16600442	75992437	128104056	11584	14288	3645	3052	496	3414	
0	4289829	9481684	24422887	6749	7463	9223	6645	2325	375	4019	760
2028	53180	635728	43122048	87482129	8288	10772	3550	1417	0	4812	
1528	60880	4640	24233111	51882225	4910	7754	4559	2706	0	3073	
0	6469929	4311616	29964050	5245	5515	7447	2181	1143	409	3183	521
1385	52860	1504750	55052505	112183343	9945	15732	4067	3448	537	4119	
1537	57280	20914651	32981661	82243369	6787	10929	3630	2030	890	3433	
1254	58250	9891548	83912409	98764219	9748	13989	5355	2702	0	2653	
0	6234863	14952839	30583726	5417	5481	10474	4823	2109	365	11426	885
1759	44980	1453733	32032436	63413152	6263	11215	4136	3072	836	3862	
2026	53080	1443700	34481909	75573985	7241	9528	6220	1732	378	3525	
2629	61530	8721576	31812176	83983908	9052	11296	2228	2479	383	3442	
463	56211365	13162690	50414122	9320	9396	15883	2930	3034	484	5000	379
449	77021891	4522835	3640	6991	7022	10129	3294	2268	0	5422	733

7264	736	3973	7894	6892	10746	3472	2037	478	11763	1421	
1396	2485	4723									
6879	550	2976	7278	7682	10050	5877	1985	0	3190	531	0
462	2448	3417									
5883	0	2788	7012	7355	11404	1358	2237	800	5955	1413	0
841	2656	2513									
	5581	1224	4118	8235	8235	10588	4959	1395	0	3256	444
930	1395	3750									
	4706	1000	4103	10256	8718	12308	4167	2273	909	2353	833
784	2353	4490									
	3265	417	2381	6190	6190	8571	3600	1951	488	2917	800
408	1633	3871									
	6364	496	3529	7647	7059	9412	3158	976	488	5000	
1860	0	909	2727	2449							
	6087	1379	7333	12667	11333	14000	3462	2857	476	3182	
2400	0	435	2174	3846							
0	3994	930	2264	6415	7170	9057	6531	2222	370	3810	741
0	808	1418	2745								
	5091	625	4211	8421	7895	10526	3390	1081	0	4500	
1852	0	727	1818	2000							
	6364	426	2857	5000	5357	7143	1961	1038	389	2927	513
0	909	1364	3810								
	5185	1111	5161	10968	9677	15484	3922	3077	513	3830	
1081	0	741	2222	3200							
	4906	1739	2857	6939	6122	10204	5417	1538	385	3404	
2326	667	755	755	2807							
	6154	1429	3014	4110	4110	6575	1739	2500	417	5000	
2083	0	769	1154	3600							
	7805	1111	2000	6500	7000	11500	4706	2500	0	2963	800
0	976	1463	2692								
	5424	1887	2727	7727	5909	8636	4211	784	0	4545	851
0	678	1695	3256								
	5769	2326	3182	6818	8182	9091	4681	2000	800	3404	426
408	769	1923	2347								

0	60001000	10911000	40002449	9000	9000	12500	4444	2857	1429	3636	7550
0	6667513	9521538	24562807	5614	6316	7719	1905	1111	278	3500	6900
2128	66670	113200	39132222	86962128	8261	10435	3600	1111	370	1613	
0	68422105	12772632	35004286	7000	5500	10000	4878	1127	282	3333	7690
1071	52000	5881600	28002000	72003784	7600	8400	2553	1429	0	2593	
1053	58820	851784	45832353	100005098	9583	12917	2407	3000	500	4000	
1818	70830	1200417	35901250	66674211	6154	10769	3571	1111	370	363	
0	80001714	18523429	64292667	9286	11429	13571	5778	1923	769	5385	4080
1404	50000	7690	25202000	72443784	6772	7874	2041	2400	400	5854	
1538	46810	8161277	43751702	56253111	6875	13125	4800	4242	0	4364	
2552	4513791	1599858	33432827	86283788	8669	9815	2268	2143	0	3508	
1627	62900	15152288	45541968	77162048	8650	13386	6074	2412	0	4714	
0 5121	6208 1504 1637	1565 836 0	4812 3001 3560	9048 464	9071 5831 0	12969 1480 955	4270 4351 1571	3172 8387 3991	0 8606	3329 10975	984
0	4981786	11552718	42914743	10613	8956	12671	4533	2412	919	2567	8460
0	3367424	4331840	25874154	6388	6442	8813	3774	2194	506	3304	8130
0	54691747	16222494	29543408	6851	6477	10004	7131	1401	786	11620	6170

2026	66630	509919	38993116	79652825	7334	9776	3407	984	510	5335	
2770	63200	1629443	76722534	129724042	11573	142600	3802	3226	489	3291	
1154	53330	1702444	31111333	8000 31820	6667	10667 55320	3529968	1778 8000	889 9600	3043 9600	
13600	5238	2581	0	2500	1143	0	1277	2128	4082		
0	59570	1200851	27452553	50983396	5098	10196	4516	1724	345	11315	870
1500	43640	1333727	29792182	59572963	5957	11064	4000	2857	816	3556	
1633	50850	1250678	32651695	73473673	6939	9388	5946	1481	370	3404	
2449	59090	8511364	29411818	82353721	8824	11176	2000	2182	364	3333	
444	53061224	11762449	48483774	9091	9091	15758	2553	2791	465	4828	377
1200	70370	7231111	38102222	76194444	6667	10476	3158	1818	455	11407	
0	6667444	5412222	27913077	6977	7442	9767	5714	1818	0	3043	513
0	65311633	17021633	44002222	5200	7600	9600	5128	2128	426	2000	588
0	5106851	5882979	44443636	11111	11111	13889	2105	2979	0	3111	889
1304	7111455	13332222	33331778	62503265	5833	10417	5128	2000	0	11385	
0	7500500	14714000	25452800	6909	6545	8364	3922	3636	727	4242	870
0	5714476	8161905	27453913	6275	7059	10980	2400	2041	816	2174	816
1429	53060	5002041	63161633	89473077	8947	12632	5714	3333	833	3784	

0	4180	939	2437	6586	7570	9306	6674	2441	390	4152	759
	809	1670	3041								
2082	5363	648	4464	8749	8140	10646	3651	1408	0	4607	
	0	738	2097	2293							

C5 algorithm

The fowling article provides more details of C5 algorithm and why it is been used in our research:(Lantz, 2019)

The C5.0 decision tree algorithm

The C5.0 algorithm has become the industry standard for producing decision trees because it does well for most types of problems directly out of the box. Compared to other advanced machine learning models, the decision trees built by C5.0 generally perform nearly as well but are much easier to understand and deploy. Additionally, as shown in the following table, the algorithm's weaknesses are relatively minor and can be largely avoided.

Strengths

- An all-purpose classifier that does well on many types of problems.
- Highly automatic learning process, which can handle numeric or nominal features, as well as missing data.
- Excludes unimportant features.
- Can be used on both small and large datasets.
- Results in a model that can be interpreted without a mathematical background (for relatively small trees).
- More efficient than other complex models.

Weaknesses

- Decision tree models are often biased toward splits on features having a large number of levels.
- It is easy to overfit or underfit the model.
- Can have trouble modeling some relationships due to reliance on axis-parallel splits.
- Small changes in training data can result in large changes to decision logic.

- Large trees can be difficult to interpret and the decisions they make may seem counterintuitive.

To keep things simple, our earlier decision tree example ignored the mathematics involved with how a machine would employ a divide and conquer strategy. Let's explore this in more detail to examine how this heuristic works in practice.

Choosing the best split

The first challenge that a decision tree will face is to identify which feature to split upon. In the previous example, we looked for a way to split the data such that the resulting partitions contained examples primarily of a single class. The degree to which a subset of examples contains only a single class is known as purity, and any subset composed of only a single class is called pure.

There are various measurements of purity that can be used to identify the best decision tree splitting candidate. C5.0 uses entropy, a concept borrowed from information theory that quantifies the randomness, or disorder, within a set of class values. Sets with high entropy are very diverse and provide little information about other items that may also belong in the set, as there is no apparent commonality. The decision tree hopes to find splits that reduce entropy, ultimately increasing homogeneity within the groups.

Typically, entropy is measured in bits. If there are only two possible classes, entropy values can range from 0 to 1. For n classes, entropy ranges from 0 to log2(n). In each case, the minimum value indicates that the sample is completely homogenous, while the maximum value indicates that the data are as diverse as possible, and no group has even a small plurality.

In mathematical notion, entropy is specified as:

$$\text{Entropy}(S) = \sum_{i=1}^{c} -p_i \ log_2(p_i)$$

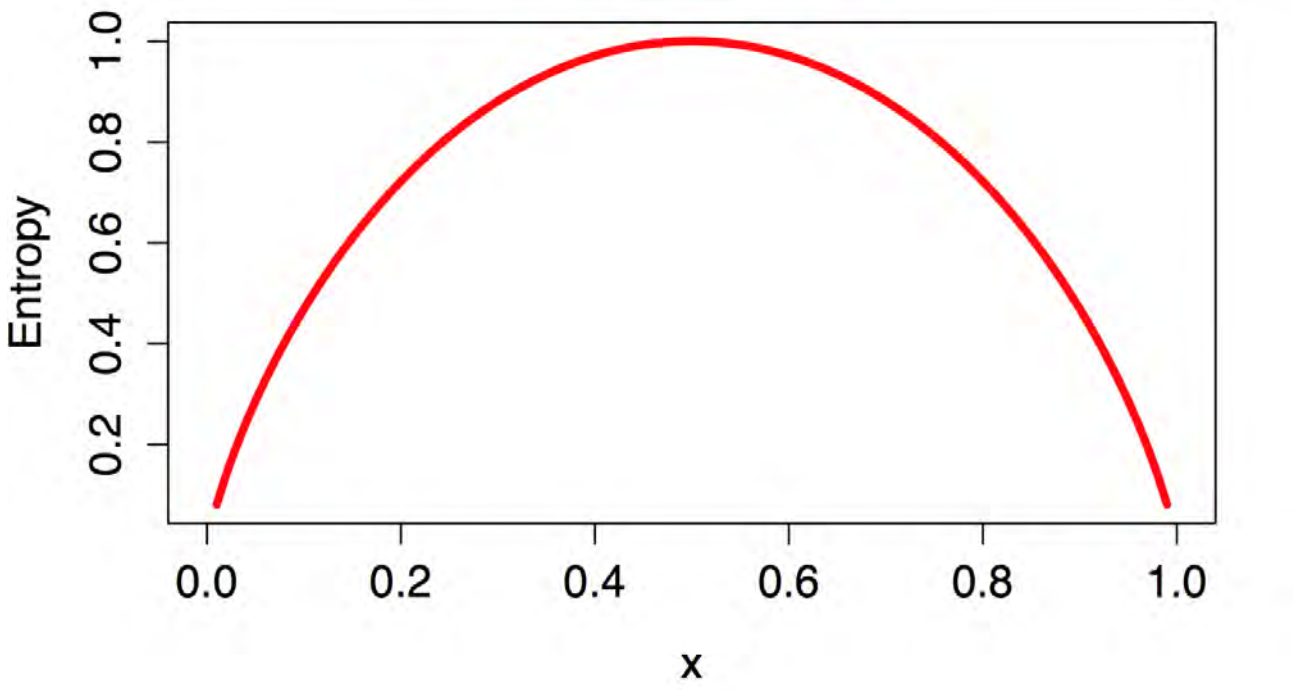
In this formula, for a given segment of data (S), the term c refers to the number of class levels, and pi refers to the proportion of values falling into class level i. For example, suppose we have a partition of data with two classes: red (60 percent) and white (40 percent). We can calculate the entropy as:

```
> -0.60 * log2(0.60) - 0.40 * log2(0.40)


[1] 0.9709506
Copy
```

We can visualize the entropy for all possible two-class arrangements. If we know the proportion of examples in one class is x, then the proportion in the other class is (1 – x). Using the curve() function, we can then plot the entropy for all possible values of x:

```
> curve(-x * log2(x) - (1 - x) * log2(1 - x),


    col = "red", xlab = "x", ylab = "Entropy", lwd = 4)
Copy
```

This results in the following figure:



The total entropy as the proportion of one class varies in a two-class outcome

As illustrated by the peak in entropy at x = 0.50, a 50-50 split results in the maximum entropy. As one class increasingly dominates the other, the entropy reduces to zero.

To use entropy to determine the optimal feature to split upon, the algorithm calculates the change in homogeneity that would result from a split on each possible feature, a measure known as information gain. The information gain for a feature F is calculated as the difference between the entropy in the segment before the split (S1) and the partitions resulting from the split (S2):

$$\text{InfoGain}(F) = \text{Entropy}(S_1) - \text{Entropy}(S_2)$$

One complication is that after a split, the data is divided into more than one partition. Therefore, the function to calculate Entropy(S2) needs to consider the total entropy across all

of the partitions. It does this by weighting each partition's entropy according to the proportion of all records falling into that partition. This can be stated in a formula as:

$$\text{Entropy}(S) = \sum_{i=1}^{n} w_i \ \text{Entropy}(P_i)$$

In simple terms, the total entropy resulting from a split is the sum of entropy of each of the n partitions weighted by the proportion of examples falling in the partition (wi).

The higher the information gain, the better a feature is at creating homogeneous groups after a split on that feature. If the information gain is zero, there is no reduction in entropy for splitting on this feature. On the other hand, the maximum information gain is equal to the entropy prior to the split. This would imply the entropy after the split is zero, which means that the split results in completely homogeneous groups.

The previous formulas assume nominal features, but decision trees use information gain for splitting on numeric features as well. To do so, a common practice is to test various splits that divide the values into groups greater than or less than a threshold. This reduces the numeric feature into a two-level categorical feature that allows information gain to be calculated as usual. The numeric cut point yielding the largest information gain is chosen for the split.

Note: Though it is used by C5.0, information gain is not the only splitting criterion that can be used to build decision trees. Other commonly used criteria are Gini index, chi-squared statistic, and gain ratio. For a review of these (and many more) criteria, refer to An Empirical Comparison of Selection Measures for Decision-Tree Induction, Mingers, J, Machine Learning, 1989, Vol. 3, pp. 319-342.

Pruning the decision tree

As mentioned earlier, a decision tree can continue to grow indefinitely, choosing splitting features and dividing into smaller and smaller partitions until each example is perfectly classified or the algorithm runs out of features to split on. However, if the tree grows overly large, many of the decisions it makes will be overly specific and the model will be overfitted to the training data. The process of pruning a decision tree involves reducing its size such that it generalizes better to unseen data.

One solution to this problem is to stop the tree from growing once it reaches a certain number of decisions or when the decision nodes contain only a small number of examples. This is called early stopping or prepruning the decision tree. As the tree avoids doing needless work, this is an appealing strategy. However, one downside to this approach is that there is no way to know whether the tree will miss subtle but important patterns that it would have learned had it grown to a larger size.

An alternative, called post-pruning, involves growing a tree that is intentionally too large and pruning leaf nodes to reduce the size of the tree to a more appropriate level. This is often a more effective approach than prepruning because it is quite difficult to determine the optimal depth of a decision tree without growing it first. Pruning the tree later on allows the algorithm to be certain that all of the important data structures were discovered.

Note: The implementation details of pruning operations are very technical and beyond the scope of this book. For a comparison of some of the available methods, see A Comparative Analysis of Methods for Pruning Decision Trees, Esposito, F, Malerba, D, Semeraro, G, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1997, Vol. 19, pp. 476-491.

One of the benefits of the C5.0 algorithm is that it is opinionated about pruning—it takes care of many of the decisions automatically using fairly reasonable defaults. Its overall strategy is to post-prune the tree. It first grows a large tree that overfits the training data. Later, the nodes and branches that have little effect on the classification errors are removed. In some cases, entire branches are moved further up the tree or replaced by simpler decisions. These processes of grafting branches are known as subtree raising and subtree replacement, respectively.

Getting the right balance of overfitting and underfitting is a bit of an art, but if model accuracy is vital, it may be worth investing some time with various pruning options to see if it improves the test dataset performance.

To summarize, decision trees are widely used due to their high accuracy and ability to formulate a statistical model in plain language. Here, we looked at a highly popular and easily configurable decision tree algorithm C5.0. The major strength of the C5.0 algorithm over other decision tree implementations is that it is very easy to adjust the training options.