Doctoral Thesis

Zero-watermarking for Data Integrity, Secure Provenance and Intrusion Detection in IoT Networks

Author:
Omair Mohamad Khair Faraj

Supervisors:
Prof. David Megías Jiménez
Prof. Joaquin Garcia-Alfaro

*A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Philosophy*

in the

K-ryptography and Information Security for Open Networks (KISON) Group
and Security and Confidence oN digital systems (SCN) Group
Network and Information Technologies

February 3, 2025

Declaration of Authorship

I, Omair Mohamad Khair Faraj, declare that this thesis titled, “Zero-watermarking for Data Integrity, Secure Provenance and Intrusion Detection in IoT Networks” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:
Omair Mohamd Khair Faraj [illegible]
Date:
03/02/2025

“*Don’t Just Aspire to Make a Living, Aspire to Make a Difference...*”

Abstract

This thesis investigates the integration of advanced security techniques into Intrusion Detection System (IDS) for Internet of Things (IoT) networks, which are increasingly susceptible to various cyber threats due to their interconnected nature and constrained resources. Traditional security techniques, like signature-based detection, can only identify known attacks, while anomaly detection can identify unknown attacks but often generates high false alarm rates. This makes advanced IDS crucial. Additionally, the challenge of efficiently and securely transmitting rapidly growing provenance information in IoT networks underscores the need for innovative solutions to ensure the integrity and reliability of data within IoT environments. We propose a robust and lightweight system to address data integrity issues, secure provenance information, and enhance the effectiveness of anomaly-based IDS. The proposed solution introduces a novel zero-watermarking approach, utilizing data provenance information as a lightweight methodology.

Firstly, we conduct a systematic review of recent Machine Learning (ML)-based IDS for IoT networks, identifying key challenges such as detection rates, false positives, real-time detection, computational overhead, and energy consumption. We highlight the necessity for extensive research covering all attack types and recent IoT technologies. Additionally, we emphasize the potential of watermarking algorithms as a resource-efficient solution for IDS implementation. Secondly, we examine the integration of IoT and data provenance, addressing vulnerabilities and the need for data trustworthiness, quality, traceability, and security. We identify significant research gaps and emphasize the need for further exploration to enhance network security, providing research directions to improve existing provenance security techniques in IoT.

Also, we propose a zero-watermarking approach ensure data integrity and secure transmission of provenance information in IoT networks. We present algorithms and modeling for various scenarios, validating the security capabilities of our approach through formal security analysis and numerical simulations. Our findings demonstrate that the proposed scheme is lightweight, computationally efficient, and consumes less energy compared to existing solutions. Additionally, we propose a novel approach to enhance the performance of anomaly-based Network Intrusion Detection Systems (NIDS) by integrating zero-watermarking with ML-based techniques. Using a two-layer approach, combining ML-based model for initial classification and data provenance-based zero-watermarking for secondary classification, we achieve high accuracy and significantly reduce false alarms. Evaluation using different datasets confirms the effectiveness of our approach in terms of classification performance and computational efficiency. Finally, we identified several possibilities for future work to further extend and improve the existing field knowledge, highlighting promising research directions for advancing watermarking techniques, data provenance and IDS in IoT networks.

Resumen

Esta tesis investiga la integración de técnicas de seguridad avanzadas en Sistemas de Detección de Intrusiones (IDS) para redes de Internet de las Cosas (IoT), que son cada vez más susceptibles a diversas amenazas cibernéticas debido a su naturaleza interconectada y recursos limitados. Las técnicas de seguridad tradicionales, como la detección basada en la firma, solo pueden identificar los ataques conocidos, mientras que la detectar anomalías puede identificar ataques desconocidos pero a menudo genera altas tasas de alarma falsa. Esto hace que el IDS avanzado sea crucial. Además, el reto de transmitir de manera eficiente y segura la información de procedencia que crece rápidamente en las redes de IoT subraya la necesidad de soluciones innovadoras para garantizar la integridad y fiabilidad de los datos dentro de los entornos IoT. Proponemos un sistema robusto y ligero para abordar los problemas de integridad de los datos, proteger la información de procedencia y aumentar la eficacia de los IDS basados en anomalías. La solución propuesta introduce un nuevo enfoque de marcación de agua cero, utilizando la información de procedencia de los datos como una metodología ligera.

En primer lugar, realizamos una revisión sistemática de los últimos IDS basados en el aprendizaje automático (ML) para las redes IoT, identificando desafíos clave como las tasas de detección, los falsos positivos, la detección en tiempo real, las superficies computacionales y el consumo de energía. Destacamos la necesidad de una extensa investigación que abarque todos los tipos de ataques y las últimas tecnologías de IoT. Además, enfatizamos el potencial de los algoritmos de marcado hídrico como una solución eficiente en términos de recursos para la implementación de IDS. En segundo lugar, examinamos la integración de IoT y la procedencia de datos, abordando las vulnerabilidades y la necesidad de fiabilidad, calidad, trazabilidad y seguridad de los datos. Identificamos lagunas significativas en la investigación y enfatizamos la necesidad de seguir explorando para mejorar la seguridad de la red, proporcionando direcciones de investigación para mejorar las técnicas de seguridad de procedencia existentes en IoT.

Proponemos un enfoque de cero-marcado para asegurar la integridad de los datos y la transmisión segura de la información de procedencia en redes IoT. Presentamos algoritmos y modelado para diversos escenarios, validando las capacidades de seguridad de nuestro enfoque mediante análisis de seguridad formal y simulaciones numéricas. Nuestros hallazgos demuestran que el esquema propuesto es ligero, computacionalmente eficiente y consume menos energía en comparación con las soluciones existentes. Además, proponemos un enfoque novedoso para mejorar el rendimiento de los Sistemas de Detección de Intrusiones de Red (NIDS) basados en anomalías, integrando el cero-marcado con técnicas basadas en ML. Utilizando un enfoque de dos capas, que combina un modelo basado en ML para la clasificación inicial y el cero-marcado basado en la procedencia de datos para una clasificación secundaria, logramos alta precisión y reducimos significativamente las falsas alarmas. La evaluación con diferentes conjuntos de datos confirma la efectividad de nuestro enfoque en términos de rendimiento de clasificación y eficiencia computacional. Finalmente, identificamos varias posibilidades para trabajos futuros, resaltando direcciones de investigación prometedoras para avanzar en las técnicas de marcado, la procedencia de datos y los IDS en redes IoT.

Resum

Aquesta tesi investiga la integració de tècniques de seguretat avançades al sistema de detecció d'intrusions (IDS) per a xarxes d'Internet de les coses (IoT), que són cada cop més susceptibles a diverses amenaces cibernètiques a causa de la seva naturalesa interconnectada i recursos limitats. Les tècniques de seguretat tradicionals, com la detecció basada en signatura, només poden identificar atacs coneguts, mentre que la detecció d'anomalies pot identificar atacs desconeguts, però sovint genera altes taxes de falses alarmes. Això fa que l'IDS avançat sigui crucial. A més, el repte de transmetre de manera eficient i segura la informació de procedència que creix ràpidament a les xarxes IoT subratlla la necessitat de solucions innovadores per garantir la integritat i la fiabilitat de les dades als entorns IoT. Proposem un sistema robust i lleuger per abordar els problemes d'integritat de les dades, assegurar la informació de procedència i millorar l'eficàcia de l'IDS basat en anomalies. La solució proposada introdueix un nou enfocament de marca d'aigua zero, utilitzant informació de procedència de dades com a metodologia lleugera.

En primer lloc, realitzem una revisió sistemàtica dels IDS recents basats en Machine Learning (ML) per a xarxes IoT, identificant reptes clau com ara taxes de detecció, falsos positius, detecció en temps real, sobrecàrrega computacional i consum d'energia. Destaquem la necessitat d'una investigació exhaustiva que cobreixi tots els tipus d'atac i les tecnologies IoT recents. A més, posem èmfasi en el potencial dels algorismes de marca d'aigua com a solució eficient dels recursos per a la implementació d'IDS. En segon lloc, examinem la integració de l'IoT i la procedència de les dades, abordant les vulnerabilitats i la necessitat de fiabilitat, qualitat, traçabilitat i seguretat de les dades. Identifiquem llacunes importants en la recerca i emfatitzem la necessitat d'explorar més per millorar la seguretat de la xarxa, proporcionant direccions de recerca per millorar les tècniques de seguretat de procedència existents a IoT.

Proposem un enfocament de zero-marcatge per assegurar la integritat de les dades i la transmissió segura de la informació de procedència en xarxes IoT. Presentem algoritmes i models per a diversos escenaris, validant les capacitats de seguretat del nostre enfocament mitjançant anàlisi de seguretat formal i simulacions numèriques. Els nostres resultats demostren que l'esquema proposat és lleuger, eficient computacionalment i consumeix menys energia en comparació amb les solucions existents. A més, proposem un enfocament innovador per millorar el rendiment dels Sistemes de Detecció d'Intrusions de Xarxa (NIDS) basats en anomalies, integrant el zero-marcatge amb tècniques basades en aprenentatge automàtic (ML). Utilitzant un enfocament de dues capes, que combina un model basat en ML per a la classificació inicial i el zero-marcatge basat en la procedència de dades per a una classificació secundària, aconseguim alta precisió i reduïm significativament les falses alarmes. L'avaluació amb diferents conjunts de dades confirma l'efectivitat del nostre enfocament en termes de rendiment de classificació i eficiència computacional. Finalment, identifiquem diverses possibilitats per a futurs treballs, destacant direccions de recerca prometedores per avançar en les tècniques de marcatge, la procedència de dades i els IDS en xarxes IoT.

Résumé

Cette thèse examine l'intégration de techniques de sécurité avancées dans le Système de détection d'intrusion (IDS) pour les réseaux Internet des objets (IoT), qui sont de plus en plus susceptibles de diverses menaces informatiques en raison de leur nature interconnectée et des ressources limitées. Les techniques de sécurité traditionnelles, comme la détection basée sur la signature, ne peuvent identifier que les attaques connues, tandis que la détection d'anomalies peut identifier des attaques inconnues mais génère souvent des taux élevés d'alarmes fausses. Cela rend l'IDS avancé crucial. En outre, le défi de transmettre efficacement et en toute sécurité des informations de provenance en croissance rapide dans les réseaux IoT souligne la nécessité de solutions novatrices pour assurer l'intégrité et la fiabilité des données dans les environnements IoT. Nous proposons un système robuste et léger pour résoudre les problèmes d'intégrité des données, sécuriser les informations de provenance et améliorer l'efficacité de l'IDS basé sur les anomalies. La solution proposée introduit une nouvelle approche de zero-watermarking, utilisant l'information de provenance des données comme une méthodologie légère.

Tout d'abord, nous effectuons un examen systématique des récents IDS basés sur l'apprentissage automatique (ML) pour les réseaux IoT, en identifiant les principaux défis tels que les taux de détection, les faux positifs, les détections en temps réel, l'excédent informatique et la consommation d'énergie. Nous soulignons la nécessité d'une recherche approfondie couvrant tous les types d'attaques et les dernières technologies IoT. En outre, nous soulignons le potentiel des algorithmes de marquage hydraulique en tant que solution efficace en termes de ressources pour la mise en œuvre de l'IDS. Deuxièmement, nous examinons l'intégration de l'IoT et de la provenance des données, en abordant les vulnérabilités et la nécessité de fiabilité, de qualité, de traçabilité et de sécurité des données. Nous identifions des lacunes importantes dans la recherche et soulignons la nécessité de poursuivre l'exploration afin d'améliorer la sécurité du réseau, en fournissant des directives de recherche pour améliorer les techniques de sécurité de provenance existantes dans l'Internet des objets.

Nous proposons une approche de zéro-watermarking pour assurer l'intégrité des données et la transmission sécurisée des informations de provenance dans les réseaux IoT. Nous présentons des algorithmes et des modèles pour divers scénarios, validant les capacités de sécurité de notre approche par des analyses de sécurité formelles et des simulations numériques. Nos résultats montrent que le schéma proposé est léger, efficace sur le plan computationnel et consomme moins d'énergie par rapport aux solutions existantes. De plus, nous proposons une approche innovante pour améliorer la performance des Systèmes de Détection d'Intrusions Réseau (NIDS) basés sur des anomalies en intégrant le zéro-watermarking avec des techniques basées sur l'apprentissage automatique (ML). En utilisant une approche à deux couches, combinant un modèle ML pour la classification initiale et le zéro-watermarking basé sur la provenance des données pour une classification secondaire, nous obtenons une précision élevée et réduisons de manière significative les fausses alertes. L'évaluation avec différents ensembles de données confirme l'efficacité de notre approche en termes de performance de classification et d'efficacité computationnelle. Enfin, nous identifions plusieurs possibilités de travaux futurs, en mettant en avant des axes de recherche prometteurs pour faire progresser les techniques de watermarking, la provenance des données et les IDS dans les réseaux IoT.

Acknowledgements

Reaching this milestone has been a journey filled with countless people whose encouragement, guidance, and support were invaluable. Although I cannot list everyone here, I am deeply grateful to each and every one who played a role in my success.

First and foremost, I thank God for granting me the strength, wisdom, and perseverance to reach this point. Without His guidance, this achievement would not have been possible, and I am forever indebted for the blessings that have guided me on this path.

I would also like to express my deepest gratitude to my parents, my father, Prof. Mohamad Khair Faraj, and my mother, Aysha Abdulkhalek, for their unconditional love, support, and sacrifices. Their encouragement and unwavering belief in me have been the foundation of my journey, inspiring me every step of the way.

To my brother, Dr. Khaireldin Faraj, and my sisters, Bayan and Rawan, thank you for your constant encouragement, understanding, and support. Your faith in me and your pride have been a source of strength during challenging times.

To my extended family, Faraj Family and Abdulkhalek Family, as well as my dear friends and loved ones, thank you for the joy, comfort, and companionship you've provided throughout this journey. You have been a crucial part of my life, and I am truly blessed to have you all.

I am profoundly grateful to my supervisors, Prof. David Megías Jiménez and Prof. Joaquin Garcia-Alfaro, for their exceptional guidance, knowledge, and encouragement. Their mentorship has been instrumental in shaping my research and my academic path, and their insights and opportunities have broadened my horizons in ways I could not have imagined.

A special thanks goes to Prof. Ahmad Muhieddine, Dr. Abdelmehsen Ahmad, and Dr. Ahmad Faraj, whose encouragement and motivation gave me the courage to embark on this PhD journey. Their early belief in my potential has left a lasting impact on my career.

To my friends who welcomed me when I arrived in Barcelona and Paris, thank you for your warmth, companionship, and support, which helped me adjust to new environments and embrace this journey with confidence.

I am deeply grateful to my thesis reviewers, Prof. Michal Choras and Prof. Mohamed Mosbah, for their thoughtful feedback and for serving on my jury. To my examiners, Dr. Alexandre Viejo and Dr. Carlos Borrego Iglesias, thank you for being part of this important milestone. It has been an honor to have such distinguished scholars review my work.

I extend my appreciation to the administrative staff at both Universitat Oberta de Catalunya (UOC) and Institut Polytechnique de Paris (IPParis) for their unwavering support in managing the paperwork and logistics. Their kindness and assistance ensured a smooth process, allowing me to focus on my research.

Finally, to everyone who helped me in any way along this journey, no matter how big or small, thank you. Your support has not gone unnoticed, and I am profoundly grateful to each of you. This thesis is a result of the collective contributions of all who believed in me, and I carry your encouragement with me as I move forward.

Contents

List of Figures

List of Tables

List of Abbreviations

Acronym	**Description**
CPS	**C**yber-**P**hysical **S**ystem
IDS	**I**ntrusion **D**etection **S**ystem
IDSs	**I**ntrusion **D**etection **S**ystem**s**
IoT	**I**nternet **o**f **T**hings
WSN	**W**ireless **S**ensor **N**etwork
ECC	**E**lliptic **C**urve **C**ryptography
SGW	**S**in**G**le chaining **W**atermark
LSB	**L**east **S**ignificant **B**it
LWC	**L**ight**W**eight **C**hained Watermarking
FWC-D	Lightweight **F**ragile **W**atermarking S**C**heme
RWFS	**R**andomized **W**atermarking **F**iltering **S**cheme
MAC	**M**edium **A**ccess **C**ontrol
PRW	**P**osition **R**andom **W**atermark
PUFs	**P**hysical **U**nclonable **F**unction**s**
RSSI	**R**eceived **S**ignal **S**trength **I**ndicator
ACT	**A**symmetric **C**ryptography **T**echnique
DDoS	**D**istributed **D**enial **o**f **S**ervice
ZWT	**Z**ero-**W**atermarking **S**cheme
AES	**A**dvanced **E**ncryption **S**tandard
DES	**D**ata **E**ncryption **S**tandard
NIDS	**N**etwork **I**ntrusion **D**etection **S**ystem
ML	**M**achine **L**earning
SVM	**S**upport **V**ector **M**achine
HIDS	**H**ost-based **I**ntrusion **D**etection **S**ystem
CIDS	**C**ollaborative **I**ntrusion **D**etection **S**ystem
DoS	**D**enial-**o**f-**S**ervice
GA	**G**enetic **A**lgorithm
ANN	**A**rtificial **N**eural **N**etworks
DT	**D**ecision **T**rees

KNN	**K**-**N**earest **N**eighbors
RNN	**R**ecurrent **N**eural **N**etwork
LSTM	**L**ong **S**hort-**T**erm **M**emory
GRU	**G**ated **R**ecurrent **U**nit
KPCA	**K**ernel **P**rincipal **C**omponent **A**nalysis
NODS	**N**etwork **O**utlier **D**etection **S**ystem
GNB	**G**aussian **N**aive **B**ayes
RBF	**R**adial **B**asis **F**unction
IGAN	**I**mbalanced **G**enerative **A**dversarial **N**etwork
ROC	**R**eceiver **O**perating **C**haracteristic
AUC	**A**rea **U**nder **C**urve
PCA	**P**rincipal **C**omponent **A**nalysis
DL	**D**eep **L**earning
PPF	**P**robabilistic **P**acket **F**low
MAC	**M**essage **A**uthentication **C**ode
U2R	**U**ser to **R**oot
R2L	**R**emote to **L**ocal
LAN	**L**ocal **A**rea **N**etwork
FNR	**F**alse **N**egative **R**ate
FPR	**F**alse **P**ositive **R**ate
TP	**T**rue **P**ositive
FN	**F**alse **N**egative
FP	**F**alse **P**ositive
TN	**T**rue **N**egative
MITM	**M**an-**i**n-**t**he-**M**iddle

List of Symbols

Symbol	**Description**
$d_{n,k}$	Captured data packet k from node n
N	Number of nodes in the network
H	Number of hops the data packet $d_{n,k}$ routed through
w_{ip}	IoT Device n IP Address
w_t	Sensed data (d_n) capturing time
w_{sq}	Sensed data (d_n) generated sequence number in single-hop scenario
w_{sq_i}	Generated by hop sequence number in multi-hop scenario
$sw_{f_{n,k,i}}$	Sub-watermark i generated using data features of $d_{n,k}$
$sw_{h_{n,k,i}}$	Sub-watermark i generated using hash function for $d_{n,k}$
$E(sw_{f_{n,k,i}})$	Encrypted sub-watermark $sw_{f_{n,k,i}}$
$W_{F_{n,k,i}}$	Final generated watermark i of data packet k from node n
$d_{(n,k)W_{F_{n,k,i}}}$	Watermarked data
$R(W_{F_{n,k,i}})$	Re-generated watermark
$R(sw_{f_{n,k}})$	Re-generated sub-watermark from data features
$R(sw_{h_{n,k,i}})$	Re-generated sub-watermark from hash function
$p_{n,k,i}$	Provenance record i of data packet k from node n
$P_{n,k}$	Set of provenance records of data packet k from node n
$\|\|$	Concatenation
$ENC()$	Encryption function
$DEC()$	Decryption function
$H()$	One-way hash function
K_j	j^{th} Generated and distributed secret key for encryption/decryption
$QRY()$	Query function from network database
$STR()$	Store function to network database
I_l	Intermediate node l
S_n	Source node n
G	Base station or Gateway
q	Number of bits deleted by an attacker

This thesis is dedicated to my family: my father, my mother, my brother and my two sisters.

Chapter 1

Introduction

1.1 Security Issues

The IoT is an intelligent system composed of physical objects interconnected in a dynamic network infrastructure, which allows it to collect and exchange data between different sources and destinations. These objects route data being captured from the environment to the unsafe Internet to be managed, processed and analyzed using different technologies. This makes it easier for an attacker to access the network and, thus, the system becomes vulnerable to intrusions. Such intelligent systems are being used in various applications, such as healthcare systems, home appliances, car automation, industrial control, or environmental monitoring, among others. For these reasons, and for more than two decades, protecting and securing networks and information systems have been delivered through Intrusion Detection Systems (IDSs). It is difficult to apply traditional protection techniques to IoT networks due to some characteristics such as specific protocol stacks, constrained-resource devices, computational and power capabilities, storage limitations and network standards [1].

At first, processing capabilities and storage limitations of network devices that host Intrusion Detection System (IDS) algorithms is a critical issue. In conventional networks, IDS agents are deployed by network administrator in intermediate nodes that have high computing capacity [2]. On the other hand, network nodes in IoT environments are resource-constrained. Therefore, finding nodes with the ability to support IDS agents is difficult in such systems. Another major issue is the network architecture. Specific nodes, such as routers and switches, are responsible for forwarding packets to final destinations that are directly connected to end systems in traditional networks. IoT networks are generally multi-hop. In this case, network nodes forward packets simultaneously and act as end systems. Hence, for the sensed data packets to reach the final destination (e. g., gateway, central processing unit, etc.) it will be forwarded through a path of sensor nodes placed on different light poles [2]. Sharing these data packets through the shared wireless medium expose the network to be vulnerable to several types of security attacks, such as data forgery, packet replay, data modification, data insertion, or packet drop attack [3, 4].

Furthermore, as the digital age advances, the importance of network security has become a major issue in the cybersecurity community. The rise of information and network technologies has led to an accumulation of a huge amount of data related to organizational operations and individual activities [5, 6]. If compromised, this data could lead to significant losses and security breaches. The increase reliance on network infrastructure introduce the importance of securing these networks against malicious intrusions and cyber threats [7, 8]. To protect networks from intrusions and attacks, various approaches have been proposed and implemented, including firewalls, digital signatures, and IDSs. IDS in particular help the network detect threats to take actions based on the detection procedure. It is capable of identifying various forms of malicious network activity and irregular computer system usage, a task conventional firewalls are unable to perform. IDS relies on the premise that the actions of intruders deviate from those of legitimate users [9].

IDS play an important role in detecting different types of attacks, serving as a valuable tool for the in-depth defense of computer and sensor networks. They monitor network traffic for known or potential malicious activities and trigger an alarm when malicious activity is detected. IDS are generally categorized into two types: misuse and anomaly intrusion detection systems. Misuse IDS identify intrusions based on system weaknesses and known attack signatures, but they fail to recognize new or unfamiliar attacks. In contrast, anomaly IDS are based on normal behavior parameters and use them to identify any action that significantly deviates from normal behavior [10–12]. With the development of Machine Learning (ML), these methods have been widely applied in intrusion detection. ML-based IDS provide a learning-based system to discover classes of attacks based on learned normal and attack behavior. The goal is to generate a general representation of known attacks. Misuse detection techniques fail to detect unknown attacks, although they provide good detection accuracy for detecting well-known attacks. Various ML techniques have been explored and implemented to build an anomaly-based IDS [13–16].

Supervised learning, which creates a mapping function based on pre-defined input-output pairs, is the most widely used technique in IDS. Unsupervised learning, which allows a model to discover internal relationships by itself, is also used. Another approach that has been widely adopted in the IDS research community is the hybrid approach. This approach combines two or more learning techniques to exploit the advantages of each of them and improve the overall detection rate. It is also an effective technique used to reduce bias towards more frequent attacks as a result of data set imbalance. However, this method increases complexity and computational time of the learning model [17, 18]. We believe that there is a need to introduce new security methods to overcome the challenges of the existing ML-based IDSs. In response to the aforementioned security concerns, it is essential to develop a robust and lightweight system capable of addressing data integrity issues, securely transmitting provenance information, and enhancing the efficacy of current anomaly-based IDSs. The objective of this thesis is to propose such a solution, introducing a new approach based on zero-watermarking as a lightweight methodology using data provenance information.

1.2 Motivation

The development of security solutions for computer and IoT networks has unveiled several unresolved challenges. These challenges raise significant concerns about network security and the effectiveness of anomaly-based IDSs. Here, we summarize the key issues:

- There is a need for a lightweight IDS scheme that maintains data integrity, trustworthiness and the ability to secure provenance to ensure that data is forwarded safely in IoT networks. Data provenance provides history of the data origin and how it is routed over time, which makes it an important tool for the assurance of data trustworthiness [19, 20]. Most of the previous research on provenance considered studying modeling, collecting and querying data provenance without focusing on its security. Moreover, very few approaches considered provenance in sensor networks. In such networks, there is a set of challenges to deploy provenance solutions. These challenges are (i) manage processing overhead of each individual node, (ii) efficiently transmit provenance while minimizing the additional bandwidth consumption, and (iii) transmit provenance securely from source to destination with the prompt react to any attack [19]. There is a need to design and implement a complete framework that deals with the above mentioned provenance challenges while ensuring data integrity in an IoT network. These requirements can be achieved through watermarking techniques that are lightweight and require less computational and storage capabilities. Scalability is also an important requirement, since the size of provenance increases proportionally as the number of nodes engaged in the forwarding process increases.

- The integration of ML-based IDS has significantly enhanced security by enabling the capture of network attacks in both traditional computer networks and emerging IoT environments. One of the most known supervised ML techniques, Support Vector Machine (SVM), have been successfully applied to handle complex patterns, which are nonlinear and high dimensional. SVM has proved to perform better than traditional learning approaches in terms of classification and detection of attacks in a binary and multi-class classification scenarios in network security applications. SVM provide several advantages over other ML and Deep Learning (DL). SVM provide interpretable decision boundaries, making it easier to understand and trust the model's predictions. They perform well with network traffic datasets, have faster training times, and are robust to noisy data. Additionally, SVM offer feature importance scores, help avoid overfitting, and are more resource-efficient compared to DL models, making them an effective solution for IDSs [21–24]. However, IDS often deal with large volumes of data, which may contain irrelevant and redundant features. This can slow down the training and testing process, consume more resources, and result in a poor detection rate [25]. Moreover, the misclassification of packets in ML-based IDSs remains a significant concern. This includes two types of errors: Type I error (false positives), where the IDS mistakenly classifies normal traffic as malicious, leading to unnecessary alerts that can cause alert overload and waste resources; and Type II error (false negatives), where the system fails to detect an actual intrusion, incorrectly classifying malicious activity as normal. The latter is particularly concerning as it allows threats to go undetected, potentially resulting in severe security breaches. These errors can result in security breaches and vulnerabilities going unnoticed, posing significant risks to the system's integrity and safety. To address this issue, ML approaches need to be assisted with other security techniques to minimize the number of misclassified packets and increase detection rate.

1.3 Objectives

In this dissertation, we investigate security issues in IoT networks. We study watermarking, data provenance and ML-based IDS in IoT. We establish the following objectives:

- **Objective 1.1:** Analyze the security approaches using ML-based IDS in IoT networks. Identify the existing limitations in IDS solutions and future research perspectives.

- **Objective 1.2:** Study the existing security techniques for data provenance in IoT networks. Understand the provenance storage mechanisms and encoding methods. Assess the current shortcomings of existing solutions and consider future research directions.
- **Objective 1.3:** Develop a new approach to ensure data integrity and securely transmit provenance information in IoT networks. We consider the two scenarios of IoT: single-hop and multi-hop. Additionally, our objective is to investigate watermarking as a promising technique that provides a lightweight and robust model for resource-constrained IoT networks.
- **Objective 1.4:** Propose a new approach to enhance classification performance of anomaly-based Network Intrusion Detection System (NIDS). Also, our objective is to reduce misclassification errors and computational overhead of these systems.
- **Objective 1.5:** Evaluate the robustness of the proposed approaches via security analysis.
- **Objective 1.6:** Validate the proposed approaches via numerical simulation and experiments.

1.4 Contributions

The main contributions of this thesis are as follows:

- We conduct a survey on the state of the art of IDS in IoT networks. We review approaches addressing this problem. We focus on ML-based solutions as a representative trend in the related literature. We survey and classify ML-based techniques that are suitable for the construction of IDS for IoT networks. We contribute with a detailed classification of each approach based on our own taxonomy. Open issues and research challenges are also discussed and provided (Objective 1.1).
- We present a systematic literature review of data provenance in IoT networks, exploring existing techniques, practical implementations, security requirements, and performance metrics to evaluate such approaches and compare respective contributions and shortcomings. We propose a taxonomy that categorizes attributes related to the development of data provenance in IoT. Additionally, we identify open issues and present future research directions, providing useful insights for the evolution of data provenance research in the context of the IoT (Objective 1.2).
- We propose a complete framework called Zero-watermarkIng based data pRovenanCe for iOt Networks (ZIRCON) that deals with the previously mentioned provenance challenges while ensuring data integrity in an IoT network. These requirements can be achieved through watermarking techniques that are lightweight and require less computational and storage capabilities. Scalability is also an important requirement, since the size of provenance increases proportionally as the number of nodes engaged in the forwarding process increases. To address this issue, we introduce a tamper-proof network database, connected to all nodes and gateway, that stores the provenance information at each hop. In this framework, we introduce a zero-watermarking approach that securely

communicates provenance information through a tamper-proof database and provides data integrity for real-time systems (Objectives 1.3, 1.5, and 1.6).

- We propose ZW-IDS, a novel approach which integrates an anomaly-based NIDS with a zero-watermarking-based approach for data provenance. Data provenance provides the capability to ensure data trustworthiness by summarizing the history of ownership and actions performed on collected data from the source device to the final destination. While previous studies focused on modeling, collecting, and querying provenance, IDS have been overlooked. The main goal of this approach is to minimize the false positive rate and false negative rate, improve the computational complexity and enhance the classification performance of NIDS. Firstly, we introduce a first layer of classification using SVM by adopting a feature selection method based on Extremely Randomized Trees. Secondly, we propose a novel zero-watermarking approach using data provenance as a second layer of classification, where we use provenance information as extracted features to generate a zero-watermark for each captured data packet. Moreover, we apply these two layers of classification to build an effective anomaly-based NIDS and evaluate the effectiveness and feasibility of our approach by conducting experiments on NSL-KDD [26] and CICIDS2017 [27] network traffic intrusion detection datasets (Objectives 1.4 and 1.6).

It is worth noting that for a thorough understanding of the proposed system, individual contributions are not delineated in separate chapters. Instead, subsequent chapters are organized based on methodological subjects and objectives.

1.5 Publications

Part of the work covered in this thesis has already been published in different international peer-reviewed conferences and journals. We list the scientific publications that are directly related to the work in this dissertation.

Conference Papers

C. 1. **O. Faraj**, D. Megías, A-M. Ahmad, J. Garcia-Alfaro. 2020. Taxonomy and challenges in machine learning-based approaches to detect attacks in the internet of things. *In Proceedings of the 15th International Conference on Availability, Reliability and Security (ARES '20).* Association for Computing Machinery, New York, NY, USA, Article 79, 1–10. https://doi.org/10.1145/3407023.3407048

C. 2. **O. Faraj**, D. Megías, J. Garcia-Alfaro. 2024. ZW-IDS: Zero-Watermarking-based network Intrusion Detection System using data provenance. *In The 19th International Conference on Availability, Reliability and Security (ARES 2024), July 30–August 02, 2024, Vienna, Austria.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3664476.3670933.

Journal Papers

J. 1. **O. Faraj**, D. Megías, J. Garcia-Alfaro. (2024). ZIRCON: Zero-watermarking-based approach for data integrity and secure provenance in IoT networks. *Journal of Information Security and Applications*, 85, 103840. https://doi.org/10.1016/j.jisa.2024.103840.

J. 2. **O. Faraj**, D. Megías, J. Garcia-Alfaro. Security Approaches for Data Provenance in the Internet of Things: A Systematic Literature Review. *Computing Surveys*, *[Accepted]*. https://doi.org/10.48550/arXiv.2407.03466.

1.6 Organization

The rest of this thesis is organized as follows:

- **Chapter 2, Concepts and Background.** This chapter provides essential technical concepts and background on zero-watermarking, data provenance and IDSs. It presents several proposed taxonomies, definitions, attacks and methods investigated in this thesis.
- **Chapter 3, State of The Art.** This chapter surveys existing works regarding the concerned topics of the thesis such as zero-watermarking approaches in IoT networks, data provenance methods, and ML-based solutions for IDSs. It includes an analysis of the literature. This chapter also presents open issues and research challenges as findings from the analyzed related works.
- **Chapter 4, Zero-watermarking Scheme for Data Integrity and Secure Provenance.** This chapter develops the high-level architecture of the proposed ZIRCON framework with theoretical modeling and protocols.
- **Chapter 5, Security Analysis and Numerical Simulation.** This chapter provides the security analysis and numerical simulation for validating the proposed model.
- **Chapter 6, Zero-watermarking and Intrusion Detection System.** This chapter presents the proposed integration of zero-watermarking approach with ML-based IDS along with the experiments conducted on different network traffic datasets.
- **Chapter 7, Conclusion and Future Research.** This chapter concludes the thesis by summarizing the main points of the dissertation. The relevance of zero-watermarking for data integrity, secure provenance and intrusion detection and possible future works are also discussed.
- **Appendix A, Data Provenance Comparison Tables.** This appendix presents additional tables for further analysis of the different security approaches in data provenance for IoT networks. The comparison tables include system features, security requirements, studied attacks, and performance metrics.

Chapter 2

Concepts and Background

2.1 Introduction

An intelligent system known as the IoT comprises a network infrastructure of physical objects that are interconnected, enabling the exchange and collection of data from diverse sources and destinations. The objects are designed to transfer data captured from the environment over an insecure internet connection, where it is processed, managed, and analyzed through various technologies [28, 29]. IoT increases the level of network vulnerabilities as it becomes more accessible to attackers. Despite this, the intelligent system is widely applied in different fields such as industrial control, healthcare systems, home appliances, car automation, and environmental monitoring [30, 31]. These systems rely on various sensors for data acquisition and transmission, which are resource constrained due to deployment in an unprotected physical environment. IoT networks operate via single-hop and multi-hop structures, with data transmission vulnerable to tampering [32]. Furthermore, the data generated from a huge number of sensor node sources is subjected to in-network processing by intermediate nodes during transmission to a base station, and this processed data is used in the decision making processes, making it critical to evaluate data trustworthiness, detect attacks during data generation and processing, and identify nodes and actions responsible for those attacks [33].

IoT systems lack the capacity to perform identity management, ensure data trustworthiness, detect abnormal behavior, and control unauthorized data access, becoming vulnerable to cyber attacks [34]. Data provenance solutions have been proposed aiming to contribute to overcome the aforementioned security issues in IoT systems. Initially, data provenance was used for heterogeneous database systems [35]. It was originally introduced to document the origin, history, derivation, creation and usage of some entity, which is also known as lineage [36]. It explains how data evolves from process to another. Many researchers used provenance in the field of information technology, which is defined as the process of recording the history of origin, evolution, process activities and manipulation of data over time [37–39].

Researchers introduce data provenance as a solution to provide a number of required features to ensure trustworthiness, integrity, data quality, confidentiality, availability and other security requirements. In this regard, data provenance is presented in many domains such as file systems, databases, cloud computing, IoT, distributed networks and many more domains. Moreover, watermarking is emerging as a technique to enhance security in IoT and computer networks. However, the integration of watermarking with data provenance in these networks remains largely not explored and implemented. In our work, we focus on data provenance within the IoT and Wireless Sensor Network (WSN) domains, as shown in Figure 2.1. In this section, we introduce the key concepts and technologies used in our thesis, including IoT, watermarking, data provenance, and IDS.

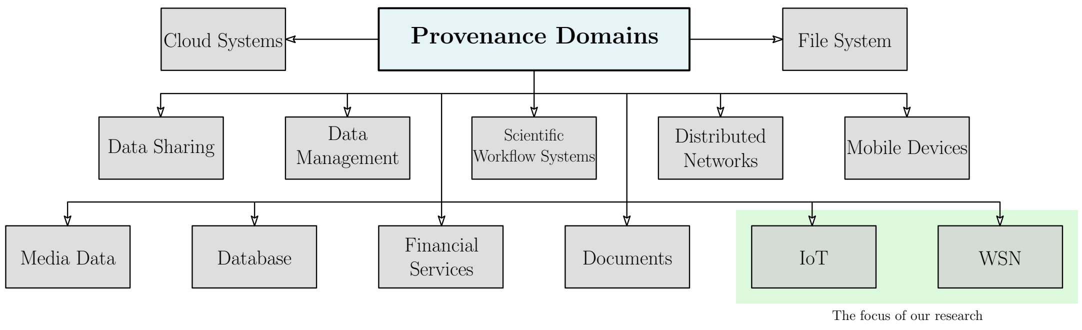


FIGURE 2.1: Provenance domains and the scope of our research.

2.2 Overview of IoT Networks

The realization of the IoT concept in practical terms can be achieved by combining various enabling technologies. We direct our attention towards sensor networks, which stand out as a highly used technology across a wide range of applications. Sensor networks have been proposed for numerous scenarios, including environmental monitoring, e-health, intelligent transportation systems, military applications, and industrial monitoring , among others. These networks consist of a number of sensing nodes that communicate in a wireless multi-hop model as shown in Figure 2.2. Typically, nodes transmit captured data to a limited number of intermediate forwarding nodes [40].

In recent years, scientists have conducted extensive research on sensor networks, examining various challenges across different layers of the network protocol stack [41]. It faces significant vulnerability to attacks due to several factors. Firstly, many of its components are often in stand-by mode for extended periods, making them easy targets for physical attacks. Secondly, the widespread use of wireless communication in the IoT makes eavesdropping a simple task. Lastly, most IoT components possess limited capabilities in terms of energy and computing resources, which prevents the implementation of complex security schemes. Specifically, trustworthiness and data integrity present major security concerns [40]. The presence of diverse data sources requires the establishment of trustworthiness for the data, ensuring that only reliable information is taken into account during the decision-making process. In addition, data integrity solutions play an important role in ensuring that any unauthorized modifications to data being transmitted by source nodes or intermediate nodes are detected by the system, preventing adversaries from tampering with the information.

For these concerns, new solutions were proposed to ensure the trustworthiness of data, such that any decision process is based on trustworthy data. Data provenance is introduced as one of these effective solutions that can be used in constrained networks to ensure trustworthiness of sensed data. Data provenance techniques must consider additional methods to tackle the problem of provenance integrity and data packet integrity.

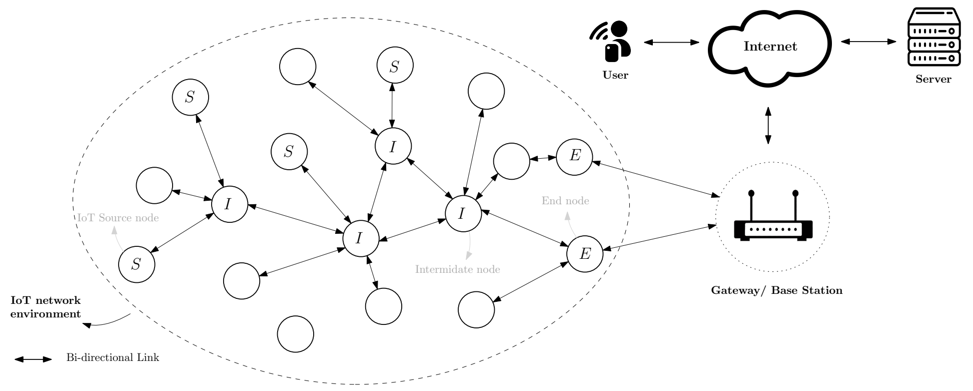


FIGURE 2.2: IoT network model.

2.3 Watermarking

Digital watermarking is one of the well known advances in IoT security. It can detect if sensory data have been modified in a precise way, and prevents the interception of this data effectively. Additionally, it can be used for protecting content integrity of multimedia digital works as images, audio and video, and copyright information [42]. Digital watermarking has many advantages over other security techniques such as:

1. The three watermarking processes (generation, embedding and extraction) requires less energy than traditional encryption due to its lightweight calculations.

2. Carrier data directly holds watermark information without adding any network communication overhead [43].

3. Digital-watermarking reduces end-to-end delay (due to lightweight watermark generation process) in a significant way compared to traditional security techniques with high complexity.

There are two main types of digital watermarking: fragile watermarking and robust watermarking (based on anti-attack properties: fragile watermark becomes undetectable after data being modified, while robust watermarks can survive many forms of distortion) [44, 45]. Robust watermarking is mainly used for copyright protection and is not sensitive to tampering. On the other hand, fragile watermarking is greatly sensitive to altering, and any change in the carrier leads to a failure in the extraction of watermark, that is used in verification of data integrity [46].

Watermarking algorithms consist mainly from three processes: watermark generation, watermark embedding and watermark extraction and verification. The general concept of watermarking is presented in Figure 2.3. Based on fragile watermarking algorithms, the source nodes collect data and generate a watermark. Then, this watermark is embedded in the original data using a predefined rule to construct a watermarked data packet that will be transferred to the destination node through the network. Through the transmission channel, the packet may be subject to many types of attacks and unauthorized access. The destination node receives the packet to extract the digital watermark and separates the original data based on the defined rule used at the source node. The restored data is then used to re-generate a watermark using the generation algorithm applied at the sensing node. Finally, the extracted watermark and the re-generated watermark will be compared to verify data integrity [47]. In this thesis, we focus on securely transmitting captured data and provenance information using zero-watermarking approach based on fragile watermarking. The works that are related to the proposed scheme fall into two classes: data integrity using watermarking and provenance security.

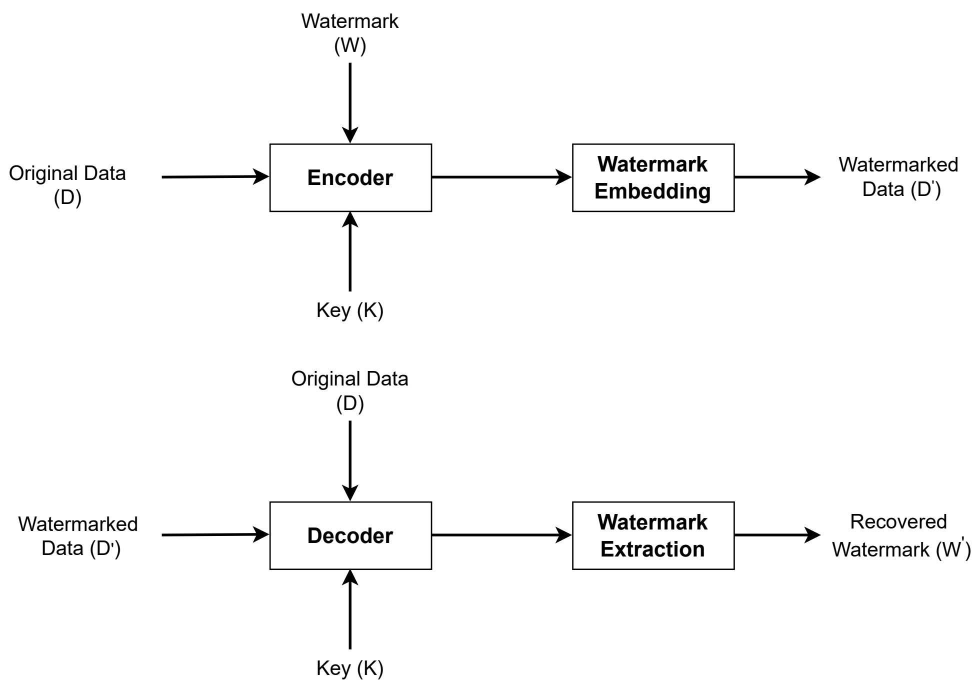


FIGURE 2.3: Watermarking concept.

2.3.1 Zero-Watermarking

A relatively recent technique for digital watermarking is zero-watermarking. Watermark generation, embedding, and extraction processes vary depending on the type of watermarking technique. Examples include hash functions (cryptographic schemes), unique codes inserted in information-hiding schemes, and bit position modifications [48]. Zero-watermarking approaches involve the generation of watermarks by the source node by the extraction of significant features from the original data without modifying the data related to these features. Zero-watermarking allows for the application of various functions for the generation process. The generated watermarks in zero-watermarking are not embedded in the data payload, but it is invisibly added to the data packet and without any modification to the data payload. Although several zero-watermarking techniques exist in the literature, very few methods are proposed to ensure data integrity and secure provenance in network security. Furthermore, to the best of our knowledge, there are no proposed techniques that use zero-watermarking in an NIDS with data provenance.

2.4 Data Provenance

The definition of provenance has undergone changes over time, adapting to different application contexts. In database systems, provenance was traditionally referred to as lineage and focused on identifying the source of data resulting from query processing, commonly referred to as data provenance [49, 50]. Data lineage was initially formalized by Cui, Widom, and Wiener [51], specifically in the field of relational databases. It aimed to trace each tuple "t" in the input tables that played a role in generating the output of a query from the database [52].

Provenance, also referred to as pedigree, or genealogy, is a form of metadata that documents the origin and utilization of a given entity [38, 53, 54]. In the field of information technology, provenance considers data as the counterpart or reflection of an art object. In this field, researchers also define provenance as the complete information about the entire process of data generation and evolution over time. This includes capturing the static origins of data as well as tracking their dynamic changes throughout their transmission process [55]. As a summary, provenance holds the capability to provide insights into the what, where, when, how, and why aspects of a given data object.

2.4.1 Data Provenance Integration with IoT

The interconnectivity of the IoT has brought significant improvements to a wide range of different application scenarios. It has also introduced security and privacy concerns due to data transmission over open networks. Attackers can easily inject malicious data into the transmission path and maliciously manipulate data without any notice from the entities of the networks, thereby compromising the integrity and trustworthiness of the data. Additionally, the rapid spread of malicious data can cause catastrophic failures. Moreover, the sharing, transmission, and processing of data leads to the risk of user privacy breaches.

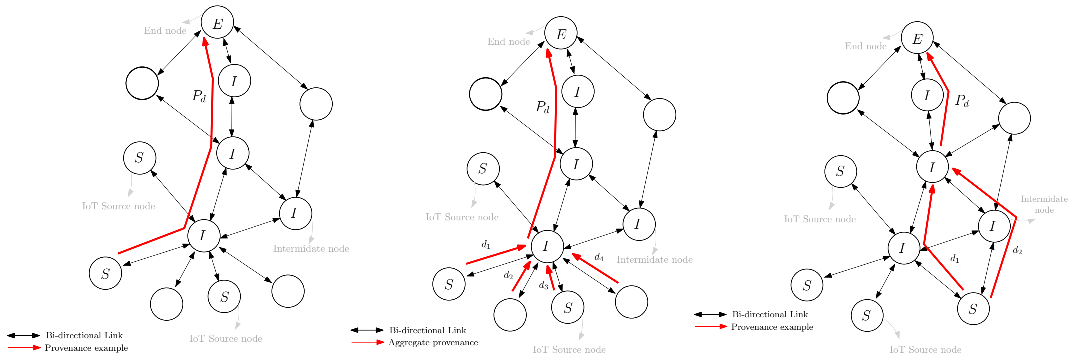
Data provenance, which records the origin and processing history of data, presents a potential solution to address these aforementioned issues. Provenance information can record the original source node which produced or forwarded data packet and the operations which the data went through. In many cases, provenance can provide the time and location of the entities that engaged in any operational procedure on data. We propose a taxonomy for the different aspects of data provenance in IoT as shown in Figure 2.6.

The provenance of a data item d, denoted as P_d, is outlined in Definition 1. P_d records information about the origin and the series of data actions a data item undergoes during its transmission from source to its final destination. There are different types of provenance according to the position of nodes in the network such as simple provenance, as shown in Figure 2.4a; aggregate provenance, as shown in Figure 2.4b; and different data path from the same source, as shown in Figure 2.4c.

Definition 1. *Given a data packet d, the provenance P_d is a graph $G(V, E)$ satisfying the following properties: 1) P_d is a subgraph of the sensor network $G(V, L)$; 2) for $v_i, v_j \in V$, v_i is a child of v_j if and only if* $\mathrm{HOST}(v_i) = n_i$ *participated in the distributed calculation of d and/or forwarded the data to* $\mathrm{HOST}(v_j) = n_j$*, whereby:*

- *$N = \{n_i/n_j \,|\, n_i/n_j$ is a network node of identifier is $i, j\}$: a set of network nodes.*
- *$E = \{e_{i,j} \,|\, e_{i,j}$ is an edge connecting nodes n_i and $n_j\}$: a set of edges connecting nodes.*

- HOST$(\cdot)$*: The function that assigns a host node to each network node in* V.HOST(v_i) *denotes the host node of the network node* v_i*.*



(a) (b) (c)

FIGURE 2.4: Data provenance types in an IoT network. (a) Simple provenance. (b) Aggregate provenance. (c) Provenance with different data path from one source.

Data Provenance Storage Mechanisms

Efficiently managing the increasing granularity of captured information in provenance, as the number of sensor nodes in a network grows, is a crucial demand. In IoT, the provenance expands rapidly due to the increase participation of forwarding nodes. The elements of provenance information vary across applications and methods, depending on the specific requirements fulfilled by the proposed solution. Furthermore, gathering detailed information about data generation, processing, and forwarding enables the system to ensure diverse security requirements. Therefore, it is essential to store this data efficiently to minimize bandwidth, storage, and energy overhead. Provenance information can be stored using four main storage techniques: *local database*, *blockchain*, *in-packet* storage, and *cloud-based*.

1. **Local Database**: In the *local database* system, provenance information is stored either in a distributed or centralized database, depending on the deployed solution's application. This storage system has challenges regarding storing and querying flexibility, as well as how IoT nodes store data information at each forwarding node. Figure 2.5a shows an example of the use of a local database in an IoT network.

2. **In-packet**: The second storage technique involves embedding the provenance records within the data packet and maintaining the provenance chain throughout its data path to the final destination as shown in Figure 2.5b. As previously mentioned, when data items are processed and transmitted across large-scale systems, the size of the provenance can significantly exceed the size of the data itself. For example, according to the findings of Jayapandian et al. [56], in the proposed MiMI system, the provenance associated with data of size 270 MB amounts to approximately 6 GB in size [56]. This limits the inclusion of numerous information elements in the provenance record, and different provenance systems must selectively retrieve certain provenance information. In many applications, it may not be feasible to ensure security requirements by limiting

the provenance information to a very small size which makes it challenging to achieve the required security conditions.

3. **Blockchain-based**: *Blockchain*, a peer-to-peer network's decentralized ledger maintained by all peers, which offers distributed data storage and has been applied to establish data provenance by recording data processes as blockchain transactions [57]. The large number of data packets generated from different source nodes in the IoT network complicates the use of blockchain for storing provenance records, as each record becomes a transaction. This complexity introduces challenges and limitations when applying blockchain for provenance storage. However, in addition to traditional internet-based blockchain solutions, edge-deployed blockchain frameworks, such as Hyperledger Fabric, can also be used to store provenance information closer to the data sources. An example of connecting a blockchain to an IoT network for data provenance is shown in Figure 2.5c.

4. **Cloud-based**: In the context of IoT applications, cloud computing is often used to store provenance records as shown in Figure 2.5d. Cloud storage provides several advantages for managing and storing provenance records in IoT. Cloud storage solutions can easily scale to accommodate the massive volumes of data generated by IoT devices. As the number of connected devices increases, cloud platforms can dynamically allocate resources to handle the growing data storage requirements. It is also designed to be compatible with various IoT devices and platforms. This ensures that provenance records from diverse sources can be efficiently stored, managed, and retrieved. Integrating cloud storage with existing IoT architectures and applications can be complex. Compatibility issues, API variations, and the need for uninterrupted data flow between devices and the cloud may introduce integration challenges. Moreover, IoT devices generate huge amount of data, and transferring this data to and from the cloud can introduce latency issues. The efficiency of cloud storage solutions relies on the speed and reliability of data transfer, which can be a challenge, basically in real-time applications [58–61]. An example of connecting a cloud database system to an IoT network for data provenance is shown in Figure 2.5d.

Data Provenance Categories

The main objective when implementing a new data provenance method is to securely store and transmit provenance information. To do so, many challenges in term of security and privacy need to be addressed, which is related to provenance manipulation in collection, storing, and transmission. It is also essential for the designed system to consider the security requirements to overcome such challenging issues [49, 62–66]. In this context, secure provenance solutions can be divided into the following categories: Cryptography-based, Digital Watermarking, Bloom filters, Physical Unclonable Functions, Fingerprints, Blockchain-based, Frameworks with storing methods, Data Santization, Lexical chaining, Graph-based, Path difference and Logging-based. These categories includes different security techniques that may be combined in some applications to assure data provenance in different IoT scenarios. These methods rely on many factors to be developed such as network resources, IoT application, needed security requirements, provenance storage approach, energy utilization, storage overhead, attack types, and network architecture. The categorization of security techniques for data provenance is shown in the proposed taxonomy in Figure 2.6.

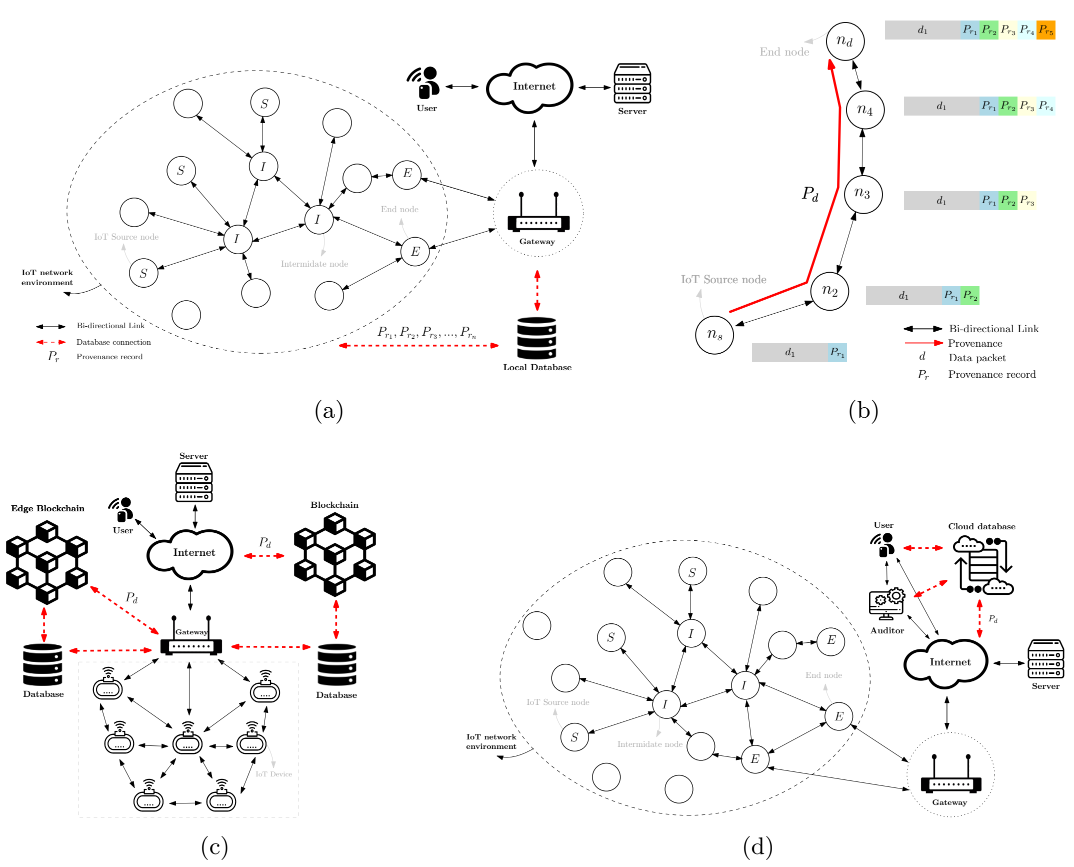


FIGURE 2.5: Provenance storage mechanisms. (a) Local database. (b) In-packet storage. (c) Blockchain-based. (d) Cloud-based.

2.4.2 Security Requirements

Introducing the challenges of ensuring data trustworthiness in such limited and constrained networks have set a number of security requirements that should be satisfied to have a robust system against different types of attacks. These requirements differ from one scenario to another and are based on the application of the IoT system. The requirements are described as follows:

- *Data Integrity:* If the provenance information is altered, it becomes impossible to perform accurate identity management, leading to delayed detection of faulty data propagation. To prevent attackers from manipulating provenance information by selectively adding or removing information, it is important to maintain the integrity of the provenance information [34].

- *Confidentiality:* It means that any potential adversary is unable to extract any details about the origin or content of a data packet just by examining the packet's data information and metadata [34].

- *Availability:* In general, availability refers to the ability to access data and is often ensured through fault-tolerance mechanisms like data replication across multiple locations. In the context of provenance, availability has been considered a part of integrity. This means that integrity verification is required to confirm, for instance, that provenance data has not been altered [67, 68] or intentionally deleted in a selective manner [69, 70].

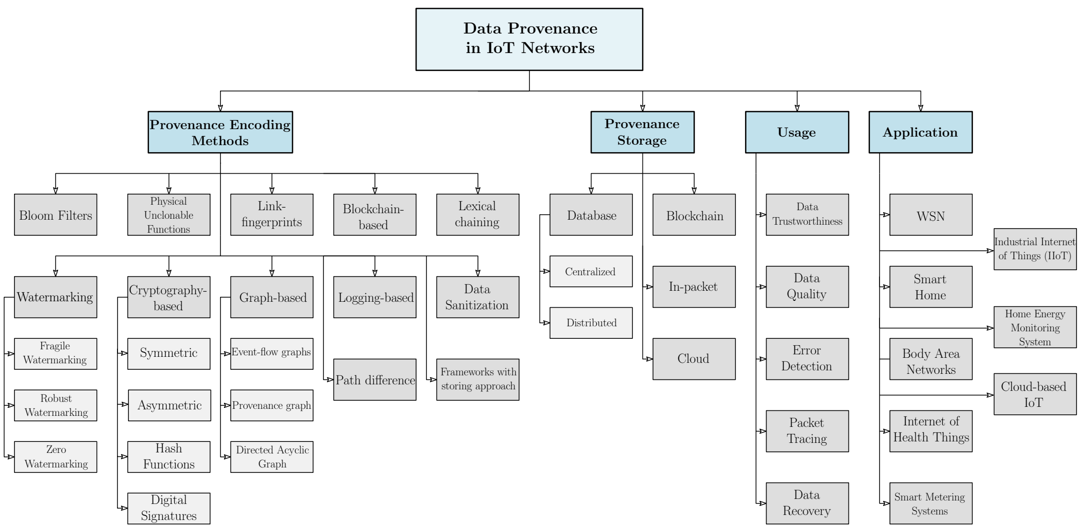


FIGURE 2.6: Taxonomy of Data Provenance in IoT networks.

- *Privacy:* Provide extra guarantees that the sources of this provenance data are not tracked by unauthorized parties. In many cases, the provenance information is more important than the data itself. This requires the protection of the privacy of such data.
- *Freshness:* Data freshness refers to the condition where the data is in its most recent and up to date form. This guarantees that no adversary can replay previous information to deceive a gateway or base station. Sensor data measurements are consistently updated, enabling the gateway to rely on the current state of the environmental data, thus preventing reliance on replayed data packets.
- *Non-repudiation:* Non-repudiation ensures that a user cannot deny their participation in any activity once the data provenance is recorded. In some cases, both parties need to follow a provenance commitment protocol to ensure mutual non-repudiation. This protocol stops either party from denying their actions or participation in the recorded activities [71, 72].
- *Unforgeability:* The most critical aspect of a secure provenance system is unforgeability. Unforgeability means that any adversary attempting to alter an existing provenance record or introduce a new forged record will be unable to do so undetected. In other words, unforgeability provides tamper evidence, ensuring the integrity and authenticity of the provenance records [71, 73, 74].

2.4.3 Attacks

IoT networks, as mentioned above are vulnerable to many active and passive attacks. The proposed security techniques takes into account these attacks and try to prove their robustness against it through many proposed solutions using different technologies. Provenance information may be more sensitive than the data itself, which makes it a priority for attackers. Attacks on such systems are of two types: *attacks on data* and *attacks on provenance.* In this section, we define the different type of attacks that may be launched in an IoT environment against data and provenance. We also provide a taxonomy for these attacks as shown in Figure 2.7.

Data Attacks

- *Packet drop attack:* Is a security threat where an adversary intentionally discards or drops specific data packets within the network. This malicious act disrupts communication between IoT devices and can lead to various issues, such as data loss, delayed information delivery, or service unavailability. As IoT devices often rely on wireless communication and may operate in resource-constrained environments, packet drop attacks can significantly impact the overall system's functionality and reliability.

- *Packet replay attack:* Is a type of security attack where an adversary intercepts and records data packets as they travel through the network. The attacker then replays or re-sends these recorded packets at a later time, attempting to deceive the system into thinking the information is fresh and legitimate. This malicious action can lead to various security issues, such as unauthorized access, data manipulation, or false triggering of actions based on the replayed data.

- *Data forgery:* Is a form of security attack where an adversary manipulates or alters the data transmitted between IoT devices. The attacker may forge false sensor readings, control commands, or other critical information to deceive the system or cause malicious actions. This type of attack can lead to incorrect decisions, unauthorized access, or compromised system integrity.

- *Data modification attack:* Is a type of security attack where an unauthorized entity alters or modifies the data being transmitted between IoT devices. The attacker may tamper with sensor readings, control commands, or other critical information, leading to incorrect decisions or actions within the system. This can result in operational disruptions, potential risks, or unauthorized access to sensitive data.

- *Eavesdrop:* Refers to the unauthorized interception and monitoring of communication between IoT devices. A malicious attacker, also known as an eavesdropper, captures and listens to data packets as they traverse the network. By doing so, the eavesdropper can access sensitive information, such as sensor readings, control commands, or personal data, which can lead to privacy breaches, security vulnerabilities, and potential misuse of the intercepted data.

Provenance Attacks

- *Provenance record drop attack:* An individual with malicious intent or a group of colluding users can intentionally drop specific provenance records or even the entire set of captured provenance information from a system.

- *Provenance replay attack:* An attacker could attempt to deceive the system by replaying a previously recorded provenance record at a later point, thereby manipulating the integrity of the provenance chain.

- *Forging provenance attack:* In the context of provenance records, a malicious user or a group of colluding users can engage in forgery by creating false data entries. These forged records can be inserted between legitimate provenance records or added at the end of the existing chain of provenance. The attacks that involve adding false records at the end are commonly known as append attacks. When multiple consecutive adversaries are involved, the process of

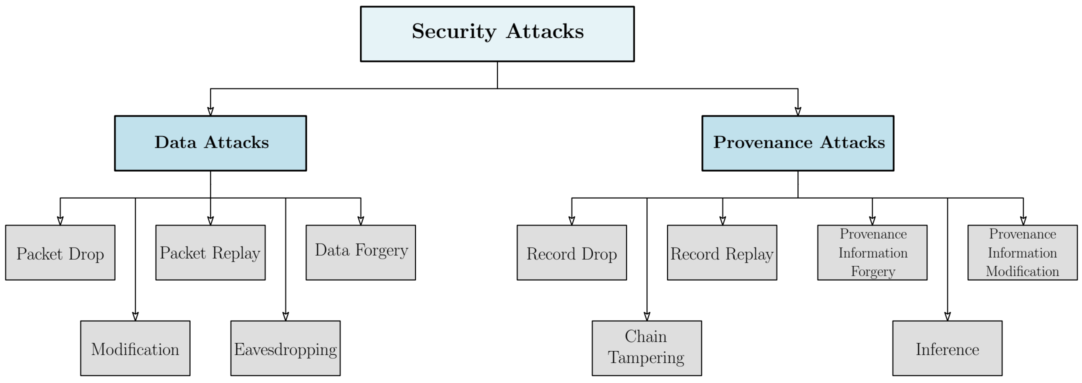


FIGURE 2.7: Taxonomy of security attacks in IoT networks with data provenance approach.

forging and adding records can be further simplified, as they can only insert forged provenance between themselves.

- *Provenance modification attack:* The attacker's objective is to manipulate the provenance record by removing specific information, or modifying existing information to deceive the system.
- *Provenance chain tampering:* Provenance records are linked together in a chain to create a complete provenance information of a specific data packet. An attacker's goal is to alter the order of this chain and modify its contents, thereby attempting to manipulate the integrity and reliability of the provenance information.
- *Inference attack:* An inference attack compromises the privacy of provenance data, enabling an adversary to deduce sensitive information about the sources and methods used to collect the provenance information.

2.5 Overview of Intrusion Detection Systems

An IDS acts like a network security guard. It constantly monitors and scans traffic for any unusual behavior that deviates from normal network activity and warns network administrators if it detect any suspicious network behavior [75]. IDS uses different techniques for intrusion detection such as signature-based or anomaly-based detection techniques [76]. IDS can be deployed in three main placement strategies as NIDS, Host-based Intrusion Detection System (HIDS), and Collaborative Intrusion Detection System (CIDS) which combines both NIDS and HIDS.

A NIDS monitors network traffic across multiple devices, providing broad visibility but struggling with encrypted data and internal threats. To effectively scan network traffic for suspicious activity, it is positioned at network key points like gateways and routers [77, 78]. A HIDS, installed on individual devices, offers deeper insights through detailed logs, but requires wider deployment and generates a huge amount of data [79]. Recognizing these trade-offs, many researchers implement both NIDS and HIDS to achieve better intrusion detection, combining network-level detection with detailed host-level monitoring. This combined approach maximizes security coverage while acknowledging the specific limitations of each system [80]. However, a CIDS typically require more resources (such as CPU, memory, and storage) compared to an

NIDS or an HIDS due to the need to monitor both network traffic and host activities simultaneously. This increased resource consumption can impact system performance and scalability.

In this thesis, we use an anomaly-based NIDS where the IDS is deployed on the edge router, since our model is proposed for detecting attacks in the network traffic. Consequently, in the remainder of this section, we provide a brief background on NIDS detection methods and ML in IDS. Figure 2.8 shows an overview of IDS in IoT networks with the different aspects that are needed to be developed when using ML approaches.

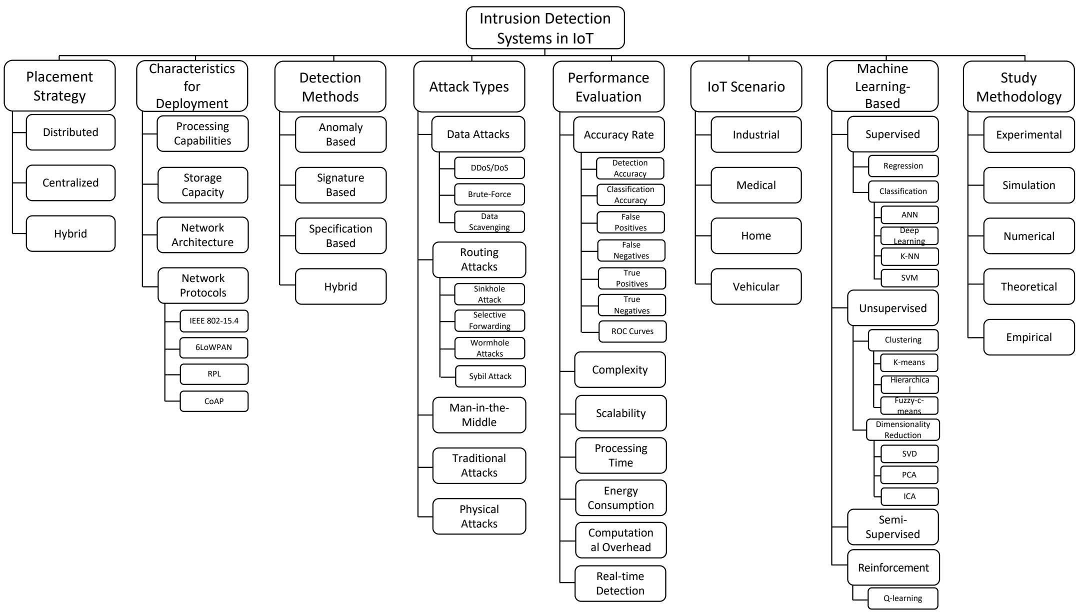


FIGURE 2.8: Taxonomy of Intrusion Detection Systems for IoT.

2.5.1 Network Intrusion Detection Systems (NIDS)

An NIDS, in communication networks, is a security tool designed to monitor and analyze network traffic to detect malicious activities or unauthorized access. An NIDS enhances the security of network devices by continuously monitoring and analyzing network communications, helping to detect and mitigate potential security threats to ensure the integrity and functionality of computer networks. An NIDS is implemented to detect network-based attacks on network devices. These attacks, often targeting protocol vulnerabilities, cause a significant threat. One example is Denial-of-Service (DoS) attacks, which aim to impact the availability of devices or networks [78, 81, 82].

The huge volume of data can become a real concern, with recent research exploring dimensionality reduction and smart processing for efficient alert handling [83]. Additionally, trust-based schemes are being proposed to ensure data quality while reporting critical information [81, 82, 84, 85]. The emerging infrastructure, protocols, new attacks, and systems in computer networks demand careful consideration when designing an NIDS. Addressing complex data handling, huge volume of network traffic, misclassification issues and trust concerns is essential for effective intrusion detection in such networks.

2.5.2 Detection Techniques

There are two types of detection techniques in IDS: signature-based and anomaly-based intrusion detection. Signature-based detection techniques focus on identifying predefined patterns associated with known malicious activities, utilizing knowledge derived from past cybersecurity incidents. This technique is effective for known or recognized attacks. However, it is not capable of detecting new attacks and relies heavily on an updated signature database for optimal performance.

Anomaly-based detection methods aim to identify deviations from normal behavior, distinguishing unknown attacks by learning previous benign patterns and classifying new behavior based on deviations from this learned norm [86]. Anomaly detection often faces challenges in distinguishing between unknown benign behavior, unknown malicious behavior, and system errors, leading to a higher misclassification rate. Additionally, the evolving nature of systems leads to several difficulties in characterizing normal behavior, potentially decreasing detection performance [87].

ML is a powerful tool for building an IDS that can recognize attacker patterns and deviations from normal behavior. Researchers have explored and successfully implemented various ML techniques to create anomaly-based NIDS. Many of these implementations still face the issue of misclassifying certain attack packets as legitimate ones, often resulting in critical consequences. This is particularly problematic in decision-making applications, where such errors can have severe implications on network security. To address this, we propose augmenting anomaly-based NIDS with a secondary layer of classification using ML techniques, incorporating zero-watermarking and data provenance.

2.5.3 Machine Learning in IDS

ML is a process that acts after learning from study or experience without being explicitly programmed, and thus it can improve automatically. The efficiency, reliability and cost-effectiveness that ML has brought to the computing processes have made it a major technique to be used in various fields including medical, engineering and computing [88–90]. ML relies on learning data sets taken as inputs. For this reason, it is used to enhance IoT and computer networks security. ML is often applied in anomaly-based, signature-based and specification-based attack detection methods. ML-based IDS enhances cybersecurity by analyzing network traffic and system behaviors to detect potential threats. ML algorithms learn from historical data to identify patterns and anomalies that indicate malicious activities. These systems continuously adapt to new threats by updating their models based on recent data, allowing for real-time detection and response to cyber attacks. By automating the identification of complex attack patterns and reducing false positives, ML-based IDS significantly improves the efficiency and effectiveness of network security measures [78, 91]. ML techniques can be classified in four main categories [92]:

- *Supervised learning*, where all training data are labeled. This category includes techniques such as:

    1. Regression: Used to predict continuous values. Examples include linear regression and polynomial regression.
    2. Classification: Used to predict discrete labels or categories. Examples include:

  - Decision Trees: Tree-like models of decisions.
  - Random Forest: An ensemble method using multiple decision trees.
  - Deep Learning: Neural networks with many layers, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).
  - Bayesian Methods: Probabilistic models that apply Bayes' theorem.
  - Support Vector Machines (SVM): Classifiers that find the hyperplane which best separates different classes.
  - k-Nearest Neighbor (k-NN): Classifiers based on the closest training examples in the feature space.
  - Artificial Neural Networks (ANN): Computational models inspired by the human brain, used for both regression and classification tasks.

- *Semi-supervised learning*, which uses a small amount of labeled training data with a larger set of unlabeled data to improve learning accuracy. This approach is particularly useful when labeling data is expensive or time-consuming. Algorithms in this category involve combining supervised learning methods with techniques that can infer labels for the unlabeled data.

- *Unsupervised learning*, which deals with unlabeled training data. This method includes:

  1. Clustering: Grouping data into clusters of similar items. Examples include:
     - k-Means: Partitions data into k clusters.
     - Hierarchical Clustering: Builds a tree of clusters.
     - Fuzzy-c-Means: Each data point can belong to multiple clusters with varying degrees of membership.
  2. Dimensionality Reduction: Reducing the number of random variables under consideration. Examples include:
     - Singular Value Decomposition (SVD): Factorizes a matrix into singular vectors and singular values.
     - Principal Component Analysis (PCA): Transforms data to new feature size where the greatest variances by any projection of the data come to lie on the new dimensions.
     - Independent Component Analysis (ICA): A computational method for separating a multivariate signal into additive, independent components.

- *Reinforcement learning*, which involves an agent interacting with an environment to learn optimal behaviors. The agent learns to make decisions by taking actions in different states to maximize cumulative rewards over time. This method includes Q-Learning, a model-free reinforcement learning algorithm where the agent learns a value function, which gives the expected utility of taking a given action in a given state and following the optimal policy thereafter.

Chapter 3

State of The Art

3.1 Data Integrity using Digital Watermarking in IoT Networks

The literature includes relevant watermarking algorithms used for data integrity and secure transmission in WSN and IoT environments. Existing watermarking schemes can be classified into two main methods: regular watermarking schemes and zero-watermarking schemes.

3.1.1 Regular Watermarking Schemes

In [93], Guo et al. propose a fragile digital watermarking algorithm called SinGle chaining Watermark (SGW), which verifies data integrity from the application layer of the data stream. The algorithm groups data based on the key and, then, the hash value of the data from the groups is calculated to be used as a watermark. To save bandwidth, the watermark is then embedded into the Least Significant Bit (LSB) of the data found in all groups. In this method, the watermark is used to link all groups and thus detect any deletion of data. Kamel et al. in [94] optimized Guo et al.'s method by proposing a Lightweight Chained Watermarking (LWC) scheme. In LWC, a dynamic group size is used and the watermark is generated by calculating the hash value of two consecutive groups of data. This leads to less computational overhead, which is an improvement from SGW that calculates the hash value of each data element in the group. The security vulnerabilities of SGW and LWC are addressed in [95], by proposing a Lightweight Fragile Watermarking Scheme (FWC-D). In this scheme, the algorithm uses a serial number (SN) that is attached to each group to detect how many group insertions or deletions have occurred in case of any insertion or deletion attack. The watermark is generated using a hash function and a group serial number. All groups are chained with a digital watermark after embedding the watermark of the current group into the previous group in order to bypass any replay attack. To solve the issue of synchronization between sender and receiver nodes in single chaining techniques such as SGW [93], LWC [94] and FWC-D [95], a dual-chaining technique is proposed in [96]. It generates and embeds fragile watermarks into data using dynamic groups.

A reversible watermarking-based algorithm for data integrity authentication is proposed in [97]. It applies prediction-error expansion for avoiding any loss in sensory data. Every two adjacent data items are grouped together, and the algorithm uses the first one to generate the watermark, and the other as a carrier for the watermark. Sun et al. [98] propose a lossless digital watermarking approach which embeds the generated watermark in the redundant space of data fields. The method does not increase data storage space due to the fixed size of redundant space. However, data integrity is only checked at the base station side. Guoyin et al. [47] propose a watermarking scheme for data integrity based on fragile watermarking in order to solve the problem of resource restrictions in the perception layer of an IoT network. They design a Position Random Watermark (PRW) that calculates the embedding positions for watermarks. The watermarks are generated using the SHA-1 one-way hash function, which is then embedded to the dynamic computed position. This scheme ensures the data integrity at the sink node. It cannot verify integrity along the route of the communication. The drawback of these solutions is that they only provide an end-to-end verification, since watermark generation and verification is based on a group of data packets. Hence, we consider our approach as a better alternative to serve hop by hop data integrity verification between source and final destination of data packets.

Zhou et al. [99] propose a secure data transmission scheme using digital watermarking technique in a WSN. In this scheme, the hash value of two time-adjacent sensitive data is calculated at the source node using a one-way hash function. Then, according to a digital watermarking algorithm, the sensitive data will be embedded into part of the hash sequence as watermark information. The scheme lacks any proof of concept and security analysis to check its robustness against different type of attacks. Another solution for attack detection presented in [100]. The method develops a Randomized Watermarking Filtering Scheme (RWFS) for IoT applications by deploying an en-route filtering that removes injected data at early stage communication based on randomly embedding a watermark in the payload of the packet. The scheme encrypts the data packet before transmission, which encounters additional computation overhead for sensor nodes.

To minimize energy consumption, Lalem et al. [101] propose a distributed watermarking technique for data integrity in a WSN using linear interpolation for watermark embedding. The technique allows each node to check the integrity of the received data locally by extracting the watermarked data and generating a new watermark for verification. This method is based on a fixed watermark parameter for all sensor nodes in the network that can be vulnerable to many attacks. Soderi et al. [102] propose WBPLSec protocol a watermarked-based approach to enhance the security of communication between nodes. This protocol utilizes a blind watermark algorithm along with a jam receiver operating over an acoustic channel to exchange a 128-bit key with neighboring devices. The process involves modulating the message using the Direct Sequence Spread Spectrum (DSSS) technique and embedding it with a shifting key to create a watermarked segment. This segment is then encoded into a waveform audio file format and transmitted via an amplifier. The receiver can extract the clean code by utilizing the information embedded in the watermark. However, this proposal necessitates the existence of a private and covert channel among nodes, which can lead to a reduction in wireless bandwidth.

3.1.2 Zero-watermarking Schemes

Zero-watermarking is a relatively new digital watermarking method. Each watermarking scheme has a different watermark generation, embedding and extraction process such as unique code (embedded in information hiding schemes), changing position of bits and hash functions (cryptographic schemes) [48]. In zero-watermarking schemes, watermarks are generated by source node from the extraction of important data features of original data without amendment to the data of these features. Different generation functions can be applied in zero-watermarking. The generated watermarks are not embedded in the data payload, but it is invisibly integrated in the data packet and the data remain unmodified.

Although several zero-watermarking techniques exist in the related literature, few methods are proposed to protect data integrity in IoT environments. In [103], a secure data aggregation watermarking-based scheme in homogeneous WSN (SDAW) is presented as a new security technique to protect data aggregation. In this mechanism, watermarks are generated using the Medium Access Control (MAC) address of sensor nodes and collected data by a one way hash function (SHA-1). The proposed scheme guarantees secure communication between the aggregation nodes and the base station. However, the authors do not provide any security analysis to check the resistance of this scheme against different type of attacks.

To ensure trustworthiness and data integrity in an IoT network, Hameed et al. [48] propose a zero-watermarking technique which generates and constructs a watermark in the original data before being transmitted. The generation process of the watermark is based on the original data features (data length, data occurrence frequency and data capturing time). This scheme is shown to be more computationally efficient and requires less energy compared to cryptographic techniques or reversible watermarking schemes. The proposed approach is vulnerable to modification attack, since it only uses data length, data occurrence frequency and data capturing time to generate the watermark. Hence, if data is modified by changing position of data values, the attack will not be detected.

3.2 Security Techniques for Data Provenance

Data provenance is a concept that is applied in many fields of study. It is uniquely defined by each application domain [104]. Data provenance in IoT networks serves to guarantee data trustworthiness by collecting the lineage of ownership and actions executed on collected data from the source node to its final destination. It is essential to record provenance for every data packet generated from source nodes and trace the involvement of forwarding nodes throughout the data transmission process, but deploying such a solution presents numerous challenges. One significant challenge is the rapid increase in provenance data during the transmission phase in IoT networks. Additionally, limitations arise from the constraints imposed by data storage capabilities, bandwidth, and energy consumption of nodes [105]. Data provenance methods establish user trust in the received information by validating its origin, ensuring that data packets were generated by the designated and authorized IoT node at the specified time and location [106]. Provenance can be described as a chain of nodes which traverses from source to destination.

Several surveys address the issue of data provenance in IoT networks. Most of these surveys focus on the security requirements for data provenance in IoT. Table 3.1 provides a comparison between the different review articles based on a number of aspects, including time range (represent the starting and ending years of the conduct review), review methodology (Does the article follows a specified review methodology to conduct the review?), taxonomy (Does the authors provide researchers with a taxonomy that summaries the different aspects used in the review?), security requirements (Does the review use and elaborate the needed security requirements in data provenance approaches?), attacks (Does the review analyze the different approaches based on the security attacks?), performance metrics (Does the review analyze the selected articles based on performance metrics used for the evaluation of any data provenance technique?) and the application domain that the review article covers.

In their study, Wang, Zheng, and Bertino [36] provide a deep understanding of provenance schemes, categorizing them into five main categories: distributed schemes, elementary schemes, lossy compression schemes, lossless compression schemes, and block schemes. The survey primarily focuses on WSN networks, examining the techniques used and discussing their respective advantages and disadvantages. One notable limitation of the study is the absence of considerations regarding attacks and performance metrics in relation to the reviewed works. Another survey that categorizes data provenance techniques is presented by Hu et al. [34]. The categorization is based on three main technologies logging-based, cryptography-based and blockchain-based technologies. The authors presented the provenance techniques based on the technology used and the requirements to achieve a secure provenance framework.

Zafar et al. [39] conducted a comparative analysis in their work, presenting a detailed taxonomy of secure provenance schemes. They further perform a discussion of existing schemes, focusing on their strengths and weaknesses. The survey provides a deep understanding of the concept of provenance and its lifecycle, while addressing the necessary security requirements for such systems. The survey does not mention the performance metrics used in the studied methods. Alam and Wang [107] propose a comprehensive survey that present and analyze different research methods on three aspects that are provenance management, capture, and analysis. They additionally address the problems of maintaining data integrity, identifying attack chains in trigger-action platforms, and policy compliance in provenance systems of IoT environments.

A comprehensive review on data provenance collection and security in distributed environments is presented by Ametepe et al. [108]. The work classifies provenance schemes based on provenance collection schemes, general provenance schemes, and basic provenance schemes. The paper discusses provenance security elements based on integrity, confidentiality and availability, but does not discuss provenance in IoT environemnt. Gultekin and Aktas [109] provide an SLR-based methodology on data provenance in IoT. The work selects 16 papers for the study. These papers are discussed based on the main subject that the study focus on. The survey lacks comparative analysis based on security requirements and other metrics that are needed to evaluate a provenance system.

The most recent survey on data provenance is provided by Pan, Stakhanova, and Ray [49]. The study explores the significance of data provenance in the domains of security and privacy. Additionally, the authors outline the foundational principles and models of data provenance, while investigating the mechanisms proposed by previous studies to achieve security objectives. Furthermore, they conduct a comprehensive review of existing schemes aimed at securing the collection and manipulation of data provenance, commonly referred to as secure provenance and the role of data provenance, specifically in terms of threat provenance. The study covers various application domains beyond IoT, including file systems, databases, generic systems, cloud computing, data management, distributed networks, and media data. While the survey examines many of the requirements for provenance systems, it does not specifically address the performance evaluation, deployment technologies and storage methods of provenance techniques in IoT environments. Furthermore, it lacks a comparative analysis in this regard.

The only survey [109] using a Systematic Literature Review (SLR) methodology for article selection does not address important aspects considered in our work, such as security requirements, security attacks, performance metrics, provenance storage methods, and taxonomy representation. In their review, the authors selected 16 papers, categorized into five main categories: blockchain, security, data flow, data trustworthiness, and privacy. However, the review lacks specific details on data provenance in IoT and does not provide discussions on open issues, research challenges, and future directions. Furthermore, a comparative analysis between the selected articles is not provided. All of the other review articles do not consider all the necessary requirements and aspects essential for analyzing security techniques proposed for data provenance in IoT.

In this thesis, we focus on IoT environments and provide an extensive comparative SLR review taking into account all the missing aspects from the discussed literature. For that, we used keyword search to make the first selection of potentially relevant scientific publications. We considered databases such as IEEE Xplore,[1] Science Direct,[2] Scopus,[3] Web of Science,[4] ACM Digital Library[5] and Springer Link[6] to collect the publications. Articles were filtered with the keywords *Data Provenance, Secure Provenance, Provenance, Internet of Things, IoT, Wireless Sensor Network,* and *WSN*. The most relevant literature was filtered according to their titles and abstracts. The collected data was later processed based on inclusion and exclusion criteria.

We classify and study the provided solutions based on the following main technologies: Cryptography-based, Bloom Filters, Link Fingerprints, Blockchain, Watermarking, Physical Unclonable Functions, Graph-based Provenance, Data Sanitization, Lexical Chaining, Path Difference, Logging-based and Frameworks that uses different Storing Methods.

3.2.1 Cryptography-based Methods

Various cryptographic techniques, including symmetric and asymmetric cryptography methods, hash functions, and digital signatures, are applied to design and implement systems for identifying the source of data and ensuring the integrity of both data and provenance. These concepts are used to propose solutions for data provenance in many applications. The literature includes the largest number of these solutions when dealing with cryptographic approaches. The trustworthiness of the data items in a sensor network is evaluated by Lim, Moon, and Bertino [105]. They compute trust

[1] `http://ieeexplore.ieee.org`
[2] `http://www.sciencedirect.com`
[3] `http://www.scopus.com`
[4] `http://www.webofknowledge.com`
[5] `https://dl.acm.org/`
[6] `https://link.springer.com/`

TABLE 3.1: Surveys related to data provenance in IoT.

Reference	Year	Time Range	Review Methodology	Taxonomy	Security Requirements	Attacks	Performance Metrics	Application Domain
Wang, Zheng, and Bertino [36]	2016	N/A	✗	✗	✓	✗	✗	WSN
Zafar et al. [39]	2017	N/A	✗	✓	✓	✓	✗	Cloud computing, WSN, IoT, smartphones
Hu et al. [34]	2020	N/A	✗	✗	✓	✗	✗	IoT
Alam and Wang [107]	2021	N/A	✗	✗	✓	✗	✗	IoT
Ametepe et al. [108]	2021	2010-2018	✗	✓	✓	✗	✗	Distributed Environments
Gultekin and Aktas [109]	2022	2012-2022	✓	✗	✗	✗	✗	IoT
Pan, Stakhanova, and Ray [49]	2023	2009-2022	✗	✗	✓	✓	✗	General

scores for sensor nodes and data packets by using data provenance. These scores provide a method of indicating the level of trustworthiness of both nodes and data items. This method provide security and trustworthiness for sensor networks, yet they fail to address the challenge of retrieving data provenance against various attacks.

Moreover, a dictionary-based secure provenance approach for WSN is proposed by Wang, Hussain, and Bertino [110]. They embed path indexes in the provenance instead of the actual data path by using packet path dictionaries. Therefore, compared to the current lossy provenance systems, the compressed provenance size in the proposed lossless approach is smaller. To achieve security requirements such as integrity, availability and authenticity of provenance, the AM-FM sketch method and a robust packet sequence number generating technique are used in the system. Provenance records are encoded at nodes that engage in every stage of data processing and transmission using the suggested dictionary-based system. A secure message authentication code integrates the packet and its provenance together to provide security against any unauthorized change. After verifying the Message Authentication Code (MAC) during decoding, the BS retrieves the packet's provenance graph.

While the majority of previous studies have concentrated on how to protect against data manipulation in Home Area Networks (HAN) networks, Chia, Keoh, and Tang [111] introduce a security topic that has received less attention in such networks, namely data provenance. To ensure that the stated energy usage is actually consumed and is gathered from the specified node at the exact location, they provide a unique technique based on threshold cryptography and Shamir's secret sharing [112]. The authors describe a unique use of secret sharing schemes for home energy monitoring networks, in which a secret, or a single private key, is distributed among all the members who provide data on energy consumption. This system also addresses the issue of location verification. To achieve this, authors incorporate a location generator as an additional component. The location generator employs trilateration techniques based on Received Signal Strength Indicator (RSSI) values and RSSI filtering methods to verify the location of the device. This not only ensures that the right power source is under observation but also detects any potential device relocation.

Suhail et al. [113] provide a solution to the challenge of integrating IoT with a system that is aware of data provenance, enabling the tracking of data flow across nodes and monitoring data transformations applied by nodes in the network. They introduce a lightweight method for transmitting provenance information for IoT sensor data. This method encodes data provenance information using a hash chain scheme as it traverses each participating node, with the final provenance verification taking place at the destination node. The technique is implemented within the context of the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [114] in Contiki OS [115]. Each generated data packet includes three fields: packet sequence number, data payload and provenance information (i.e. hash value of node ID).

Furthermore, Xu, Zhang, and Wang [116] present a provenance approach for WSN based on path index differences, where a new packet path is encoded using the index of a highly similar backbone path and the differences between them. Even if the topology of the WSN is unstable, this technique may reach an increased provenance compression rate compared to the dictionary based provenance scheme [110]. In the presented backbone path selection approach for WSN, the gradient of the node on the data packet path is also determined. In order to cover the commonly used packet transmission channels, the authors also develop a comparable backbone path elimination technique for WSN. To find the packet pathways with the highest similarity in the dictionaries, they develop a locality-sensitive hashing (LSH), SimHash [117, 118], based similarity comparison algorithm. The proposed approach is evaluated according to Total Energy Consumption (TEC) and Average Provenance Size (APS).

Moreover, Suhail et al. [119] present an approach called Provenance-enabled Packet Path Tracing for IoT devices connected through the RPL protocol. Their scheme involves including sequence numbers into the routing entries of the forwarding nodes' routing table, establishing a node-level provenance. Additionally, they introduced a system-level provenance that encompassed destination and source node IDs, enabling complete packet trace capture. To retrieve the entire data provenance using this approach, it is essential to sequentially access the storage space of each node along the routing path. Hence, the base station cannot independently decompose the complete provenance of each data packet.

Liu and Wu [65] introduced an algorithm for compressing provenance called index-based provenance compression. To reduce the overall size of the provenance data, their approach combines the concept of typical substring matching with path identifier and path index to represent path information within data provenance. Additionally, they expand the data provenance scheme to include attack detection and present a method for identifying malicious nodes based on this expanded scheme. The proposed scheme falls short in terms of ensuring data integrity, lacks a thorough security analysis within a defined threat model and results in computationally intensive operations.

Additionally, Tang and Keoh [120] present a methodology to ensure the unaltered reporting of energy usage data of home appliances. The proposed approach guarantees the collection of data from the correct and trusted source at the specified location. This framework is specifically designed for Home Area Networks (HAN) within a smart metering infrastructure, with the primary objective of confirming the authenticity of data, source identity, and location. The MAC is used to verify data authenticity and integrity. It should be noted that the authors assume the trustworthiness of the receiver (i.e. smart plug), while acknowledging the possible vulnerability of one of the two senders (i.e. smart plug or magnetic sensor) to attacks. To achieve location authenticity verification, the system integrates a third sender (i.e. a Bluetooth device), which operates independently and does not collude with other system components.

In their work, Porkodi and Kesavaraja [121] present a framework to secure the data provenance by introducing blockchain and access control policies in IoT systems. They use hybrid attribute-based encryption to securely transmit provenance information. The proposed method is evaluated using different performance metrics such as computational cost and the throughput of encryption/decryption, and the key strength is calculated using the effect of avalanche. Additionally, the authors conducted experiments to prove that the proposed approach reduces computational cost and achieves high throughput.

In their research, Xu, Bertino, and Wang [122] introduce two provenance schemes for Wireless Sensor Networks. The first is the Path Index Differences-based Provenance (PIDP) scheme, where provenance information is encoded using differences in packet path indices. Specifically, it links an index representing the packet path's similarity to the main backbone path and the variation between the actual packet path and this backbone path. Additionally, the authors propose a second scheme known as the Path Hash Value-based Provenance (PHP) scheme. In this approach, provenance data is encoded as a combination of the data source node's ID and a part of the hash value taken from the packet's path. Both schemes are evaluated based on their provenance compression rate and energy conservation rate, and these metrics are then compared to those of the DP scheme.

Xu and Wang [123] introduce a provenance scheme called Multi granularity Graphs-based Stepwise Refinement Provenance (MSRP). In this scheme, they utilize mutual information between pairs of nodes as a similarity index to categorize node IDs. This categorization forms the basis for generating multi granularity topology graphs. Additionally, they apply the Dictionary-based Provenance (DP) scheme for stepwise encoding of the provenance information. The Base Station (BS) follows the same stepwise approach for provenance recovery and simultaneously conducts data trustworthiness evaluation during decoding. The performance of the MSRP scheme is thoroughly evaluated through a combination of simulations and testbed experiments. Results show that the scheme achieve high provenance compression rate, energy utilization and efficiency of data trustworthiness assessment.

3.2.2 Bloom Filters

A compact data structure, using hashing, enables rapid verification of item presence within the structure. Provenance information is embedded within the generated structure and the bloom filter is transmitted along with the data. By employing this method, the original information remains inaccessible to potential adversaries. Due to the variability of the bloom filter value from one packet to another, establishing a connection or association between previous bloom filters and recent data becomes challenging.

In their work, Sultana et al. [33] present a secure provenance scheme for WSN. Their approach involves embedding provenance information into a Bloom filter (BF), which is transmitted alongside the data. This scheme effectively addresses the challenges posed by resource constraints in WSN. It requires a single channel for data and provenance transmission. The scheme overlooks data integrity and only focuses on studying the packet drop attack. Additionally, Siddiqui, Rahman, and Nadeem [124] present a data provenance technique for IoT devices that employs Bloom Filter and attribute-based encryption. This approach presents challenges as IoT devices typically have limited memory capacity, making it impractical to store extensive provenance information. Furthermore, the technique is vulnerable to physical attacks, whereby an attacker can easily manipulate the stored provenance information within an IoT device.

Harshan et al. [125] introduce a method for embedding provenance information known as the Deterministic Double-Edge (DDE) embedding mechanism, which is based on Bloom filters [126, 127]. In this approach, a relay node embeds information about both edges connected to it: the edge through which it receives a packet and the one through which it sends the packet. This means that a node can effectively cover two edges in the path with a single action, reducing the need for the next node in the path to update the provenance. This results in decreased packet delay. Authors show that the hop-counter within the provenance segment can also be utilized to coordinate the skipping strategy among nodes. Additionally, they propose upper bounds on the error rates of the DDE embedding technique, which are expressed as functions of the network's node count, the number of hops, the size of the Bloom filter, and the number of hash functions computed by each node.

3.2.3 Fingerprints

Fingerprints refers to unique identifiers or signatures assigned to the communication links between IoT devices. These fingerprints are typically generated based on various characteristics or attributes of the link, such as received signal strength, latency, packet loss, or other network metrics. The homomorphic feature of public-key cryptography is exploited by numerous anonymous fingerprinting systems. These approaches enable the user's fingerprint to be embedded in an encrypted domain (using public key) so that only the user may access the decrypted fingerprinted material by using the private key [128]. Data provenance solutions have been developed by analyzing and comparing link fingerprints generated from different attributes.

In their work, Ali et al. [129] present a method for enhancing the security of data provenance in bodyworn medical sensor devices. They achieve this by using the spatio-temporal characteristics of the wireless channels used for communication by these devices. This solution allows two parties to create highly similar link fingerprints, which uniquely link a data session to a specific wireless connection. This link enables a third party to later verify transaction details, especially the specific wireless link through which the data was transmitted. To validate this approach, experimental testing was conducted using MicaZ motes running TinyOS, and the results show an improvement in energy efficiency. Additionally, the proposed technique generates provenance information on each session, reducing the use of cryptographic methods.

Using different fingerprint method, Alam and Fahmy [130] propose a provenance encoding and construction method that adapts three encoding schemes: juxtaposition of ranks, prime scheme, and Rabin fingerprints. The method is referred to as Probabilistic Packet Flow (PPF). In the first scheme, provenance is constructed based on the rank of the node instead of the node ID, which requires fewer bits for encoding. Assuming that the packet contains a specific space to hold the identities of m nodes, a counter of $\log(m)$ bits is used to track the embedded ranks in the packet. The second scheme is based on prime multiplication, which embeds more node IDs using the same number of bits compared to the previous one. This method uses prime numbers as node IDs, and their multiplication results in encoding a set of IDs that can be uniquely factorized. In their third method, a data path traversed by a packet is a sequence of bits that represents the IDs of nodes on that path. The fingerprint of this sequence of bits is transmitted instead of the actual sequence.

Kamal and Tariq [131] introduce a lightweight protocol for a multi-hop IoT network, aiming to ensure both data security and the establishment of data provenance. The protocol utilizes link fingerprints derived from the RSSI of IoT nodes within the network. The protocol achieves data provenance by appending the encoded link fingerprint to the data packet header as it traverses each node. After receiving the packet, the server decode the packet header in a sequential process. However, provenance data expands rapidly, which requires transmitting a large amount of provenance information to data packets, thereby increasing the bandwidth overheads.

3.2.4 Blockchain-based Methods

Distributed ledger technologies, like blockchain, have gained visibility for maintaining the security and privacy of provenance data. These blockchain platforms serve as the primary means to ensure data integrity by creating transactions as provenance records that are chained in a secure manner. Once recorded on the blockchain, these records cannot be altered or removed, thereby establishing a trustworthy system for verifying the integrity of the provenance information in a network environment that is vulnerable to different attacks.

Baracaldo et al. [132] propose a framework in IoT environments to maintain and secure IoT provenance data. The proposed framework offers the following features:

1. Use of a completely distributed lightweight keyless blockchain element (i.e. Keyless Signature Infrastructure Module (KSI) [133, 134]) to guarantee that the integrity of the provenance data is protected.

2. Maintain provenance information confidentiality by granting specific access controls to various parties as required. With this method, provenance data can be subject to restricted access policies. The integration of cryptographic algorithms in the solution ensures confidentiality since it is difficult to manage the flow of data and its provenance in IoT contexts.

3. A high level of availability of provenance data is another feature of this system.

Zeng et al. [135] propose a blockchain-based data provenance scheme (BCP) which deploys a distributed blockchain database that is connected to the sensor network through edge computing. In this scheme, each node updates the provenance records along the packet data path. These records are obtained by high performance nodes (H-nodes), which are edge computing nodes placed either above or close to the WSN, through the process of packet sniffing [136]. Then, the base station retrieves the provenance records by querying the H-nodes. Moreover, provenance records are stored in a blockchain-chain database after encryption through invoking a smart contract running on Ethereum Virtual Machine (EVM) [137]. In this model, each provenance record contains node ID, sequence number, hop count, sequence numbers of the aggregated packets, and the number of times a packet is aggregated.

In their work, Javaid, Aman, and Sikdar [138] propose a solution for data integrity and data provenance in IoT networks by using Physical Unclonable Functions (PUFs) and a blockchain variant with smart contracts that is Ethereum. This method is called BlockPro. PUFs are used to maintain data provenance by providing unique hardware fingerprints. To overcome data tampering attacks and block unregistered devices, Ethereum is used as a decentralized digital ledger. With the presence of expanding sequence of records, data undergoes initial validation before being permanently stored on the blockchain. Once stored, it can not be tampered with or modified, thus ensuring data integrity.

Sigwart et al. [139] implement an IoT data provenance framework based on smart contracts using a generic data model to provide data provenance capturing, storing, and querying functionalities for different IoT use cases. This work extends their approach proposed in [140]. The framework uses the data provenance model by Olufowobi et al. [141]. The authors conduct an assessment of the proposed framework with specified requirements through the implementation of a proof-of-concept using Ethereum smart contracts. In this approach, a provenance record ($prov(dp)$) for a data point (dp) is composed of a 3-part structure. It links the address ($addr(dp)$) of dp (i.e. essentially an identifier ID) with the set of provenance records associated with the data points that are used to create a dp (referred to as $inputs(dp)$), and includes a context element ($context(dp)$). This element includes information for provenance purposes, such as details about the agents involved in the computation of the data point, timestamp, location, or the specific execution context within the IoT system.

According to Liu et al. [142], a multilayer provenance query index and a blockchain-based architecture are proposed for the network provenance in the IoT: blockchain-based secure and efficient distributed network provenance (SEDNP). For effective representations in the Verifiable Computation (VC) framework, the design integrates range, keyword, and K-hop ancestor queries into a single model. The authors also create a digest hashing method that verifies the provenance log and index. With constant-size digests, they decrease the storage and processing costs associated with on-chain transactions regardless of the volume of data. Along with extensive security research, they formalize preserving security to record the requirements for the validity and integrity of the query results. Finally, using a proof-of-concept approach that combines the Pinocchio VC framework and a testing blockchain network, the authors analyze the implementation issues.

For cloud-based IoT networks, Siddiqui et al. [143] provide an application layer data provenance system that relies on an execute-order architecture. By using outsourced encryption on Edge nodes using Ciphertext-Policy Attribute-based Encryption (CP-ABE), it allows fast transaction throughput on the blockchain network with minimal security overhead. The workload on the IoT nodes is reduced since every communication between IoT devices is connected to a blockchain network and recorded on authorized blockchain peers. Smart contracts are used to store the provenance data on the blockchain. The IoT Device Registration smart contract begins when an IoT node connects to the network. The Data Transfer smart contract is run whenever a message is transmitted from one IoT node to another. This is responsible for preserving the provenance information on the blockchain. The Provenance Verification smart contract, which is in charge of confirming the provenance data, is then run when it has been confirmed that the message was transmitted by the appropriate IoT node.

Another solution is presented by Yin and Fu [144], which use blockchain and smart contract to develop a data provenance scheme for IoT applications. The scheme includes a smart contract on the blockchain with reasonable access control policy. The access authorization for users and data sources is limited to maintain security of generated data. For data security and data integrity, the suggested approach implements two data structures: provenance record and provenance record set. To achieve data provenance, blockchain's non-repudiation is used. Access to the data is made secure by this structure. The provenance system can continue to function and the smart contract may impose the data owner's policy. To evaluate the feasibility of the proposed scheme, Ethereum network test is used for verification.

Additionally, Sun, Tang, and Du [145] construct a blockchain-based IoT data provenance model through adapting the PROV data model (PROV-DM) [146] to address the issue of recording provenance data generated by a multi-layer IoT system. The provenance elements are defined by PROV-DM along with their connections. The W3C provenance family of standards has this conceptual data model as its basis. To create an IoT data provenance model, it is therefore required to improve and expand the PROV-DM. In this work, authors present the needed requirements to design and build a robust data provenance model. Then, they propose a model based on PROV-DM using a blockchain network to secure provenance records from being tampered. To demonstrate how the suggested approach may be used to build a provenance graph and identify which agent is operating IoT data abnormally, they provide an application scenario in the IoT trustworthy data sharing system.

ProvNet, a distributed data sharing system that may ensure data ownership and accurately record and preserve data sharing provenance, is proposed by Chenli et al. [147]. Users can share data in two ways: through the same service provider, where provenance verification and storage will be handled by the service provider and certain users; or through other service providers, who will work together to maintain a provenance graph and authenticate sharing records. They suggest a blockchain variant structure that can provide both forward and backward tracking in order to store the provenance information during the sharing process. A directed graph is appropriate for keeping the provenance records because of the nature of data sharing, which allows senders to re-share the datasets they receive and allows datasets to be shared among many recipients simultaneously. ProvNet suggests storing the provenance records in a networked blockchain, or blocknet, as an alternative to a single blockchain. In order to perform forward tracking, ProvNet selects redactable blocks [148, 149] and Chameleon Hash [150, 151], allowing a block to record the hash value of its subsequent block while maintaining its own hash value.

3.2.5 Watermarking

Watermarking is a widely known advancement in the field of WSN security. It serves the purpose of identifying any alterations made to sensory data, effectively preventing unauthorized interception [152, 153]. The two main categories of digital watermarking are fragile watermarking and robust watermarking, each offering different anti-attack properties. Fragile watermarks become undetectable when data is modified, whereas robust watermarks can withstand various forms of distortion [44, 45]. Watermarking is being used in many security applications and is introduced in some proposed data provenance works. Sultana, Bertino, and Shehab [154] develop a technique for data provenance aimed at identifying malicious packet dropping attacks. The technique depends on the timing characteristics between packets after the process of embedding provenance information. Based on the distribution of inter-packet delays it detects the packet loss. Then, it identifies the presence of an attack and localizes the malicious node or link.

3.2.6 Physical Unclonable Functions

PUFs provide a hardware-oriented system that generates a response for a specific challenge, ensuring uniqueness for each device in the system. PUFs facilitate the authentication of IoT devices that generate sensitive data while maintaining device anonymity. Through the presence of a trusted third-party verifier, the identities of these devices can be verified without compromising their anonymity [155].

In their work, Aman, Basheer, and Sikdar [156] developed two secure protocols for data provenance in IoT networks, aiming to achieve authentication and privacy preservation. These protocols address two different scenarios: the first scenario involves a direct connection between an IoT device and a wireless gateway, while the second scenario deals with IoT devices indirectly connected to the wireless gateway through multiple hops of other IoT devices. Both protocols use PUFs in addition to wireless link fingerprints generated from the RSSI between communicating nodes. Through experimental analysis, the authors show that these protocols achieve high efficiency in terms of computational complexity and energy utilization, while also obtaining robustness against various physical and cloning attacks.

Aman, Basheer, and Sikdar [157] propose an analytical model to create a mechanism that enables the establishment of data provenance in IoT systems. Their approach incorporates PUFs and the extraction of fingerprints from the wireless channel, along with the implementation of mutual authentication and anonymity measures, all aimed at achieving robust data provenance. The approach lacks consideration for the multi-hop scenario and fails to adequately address tracking of data packet provenance.

Hamadeh and Tyagi [158] propose an approach which combine two solutions: data provenance and privacy prserving in IoT networks. They provide trustworthiness and dependable IoT networks by using PUFs with non-interactive zero knowledge proof. In this method, an IoT device has the capability to transmit data to its respective server without revealing its identity, as it provides proof of ownership. In particular, the method under consideration is designed to validate that the authorized device executed an authorized program for creating or modifying data. Authors introduce a privacy-centric data provenance protocol. To validate practicality and effieciency, they developed the protocol using Altera Quartus and subsequently implemented it on an Altera Cyclone IV FPGA.

3.2.7 Graph-based Provenance

Graph-based provenance is a method used in data provenance encoding to represent the origin or history of data. These graphs capture the relationships and dependencies among data elements, processes, and transformations, providing a visual or structural representation of how data has evolved over time. There are several types of graph-based provenance representations as shown below:

- *Event-flow graphs* are a type of provenance graph that represents data provenance by capturing events and the flow of data between them. Each event corresponds to a data operation or transformation, and edges in the graph indicate the data flow from one event to another. Chang et al. [159] propose a data provenance approach for provenance systems in IoT applications based on event-action flows instead of provenance graphs. Even-action flow is defined as a sequence of events and actions including a time-stamp in an execution trace. They are less complex than provenance graphs due their simple structure form which allows users to understand it easier. In their work, the authors present an event-flow graph to regular users as a static abstraction of every possible provenance graph for IoT applications. They dynamically link time-stamped events and actions to statically create event and action nodes. Then, users can query provenance information from the event-flow graph by selecting an event or action node to choose the associated time-stamped actions or events. After users create a query by picking a timestamped event or action, the system will respond with one of two types of information: "What provenance" answers questions about which events or actions are triggered by the user's specified action or event, and "Why provenance" provides insights into the events or actions responsible for causing the user-specified action or event. The system is developed as a Graphical User Interface (GUI) forming a user-friendly graphical provenance system.

- *Provenance graphs* include a variety of graph structures that capture provenance information. These graphs may use different notations and structures to represent data history, depending on the specific needs of the application or research. In a provenance graph, nodes represent data entities, processes, or events, and edges denote relationships, transformations, or dependencies between these nodes. Jaigirdar et al. [160] extend their work, Prov-IoT [161], to provide security information for provenance graphs. Prov-IoT is a security-aware IoT provenance model. In this work, security metadata is integrated with specified security policies within the provenance graphs. They propose an IoT-Health scenario with a number of potential threats: fault packet injection, node cloning, unauthorized access, malicious code injection and denial of service. Three major node types are used to describe the scenario: agent, entity, and activity [162]. The terms Was Associated With (WAsW), Was Informed By (WInB), Was Derived From (WDeF), and Was Generated By (WGeB) are used to represent the relationships between these three nodes. A general provenance graph for the IoT-Health scenario is generated using the W3C-standardized PROV-DM concept. To evaluate the system and check for potential risks, the approach is evaluated based on six cases: permission violation, missing Web Application Firewall (WAF), intrusion detection, unauthorized access, denial of service, and identifying failed signature verification.

- *Directed Acyclic Graph (DAG):* In the context of data provenance, a DAG is a specific type of provenance graph that has a directed structure with no cycles. This means that data transformations and dependencies are represented as a directed flow without loops or feedback. DAGs are used to represent provenance information because they are well-structured and provide efficient querying and tracking of provenance records. ProChain framework is a provenance-aware approach of traceability proposed by Al-Rakhami and Al-Mashari [163] for IoT-based supply chain systems. The IOTA protocol, a third-generation DLT, is utilized by the ProChain framework. To overcome limitations and provide a scalable, quantum-resistant, and attack-proof solution for the systems built around IoT, it makes use of the DAG information structure in contrast to the linear structure used by the blockchain [164]. ProChain enables food item traceability from manufacturing to retailer with the use of several IoT sensors and provenance data at each engaged supply chain phase. ProChain strengthens and improves the management and optimization of all operations while also serving as a guarantee for the quality and safety of food. On the Raspberry Pi 3B platform, the ProChain idea is evaluated by simulating an IoT-deployed supply chain. The average measured time and energy consumption were then evaluated to check on the usability of the framework. The authors review and implement the framework to show how it may be used in supply chain systems to couple supply chain data with the IOTA Tangle and generate provenance information by adding data for various payload sizes.

3.2.8 Data Sanitization

There are situations where privacy concerns prevent some information from being shared inside the provenance chain. This is where the method of data sanitization is presented. Data sanitization assists in hiding some important information which could compromise privacy. Data sanitization is mainly used to make sure that data is consistent, accurate, and dependable, such that it can be used for reporting, analysis, and data-hiding for privacy preservation. Lomotey et al. [165] present two approaches for device and data verification in an IoT network to achieve trust and privacy preservation. The first approach enables devices to subscribe and reveal their metadata to allow for data packet tracing without compromising privacy. To achieve this, a data sanitization method for hiding sensitive information is applied. Users and devices have the option to label attributes within the provenance data as non-shareable with other devices in the chain. In the second approach, the authors modeled the entire peer-to-peer IoT network as a graph network. Data origins are verified using Floyd's algorithm between different interconnected nodes. This work ensures transparency, traceability, and privacy preservation through both proposed approaches.

3.2.9 Lexical Chaining

Semantically similar words or phrases are linked together by lexical cohesiveness. The related elements can be linked together to create Lexical Chains after all cohesive relationships have been determined forming a conceptually accurate building blocks in a variety of natural language processing systems [166]. This is how a typical chain may be written:

$$\text{Device} \rightarrow \text{Type} \rightarrow \text{Owner} \rightarrow \text{Communicated}$$

would be feasible to track and categorize the communication channels between different IoT devices as well as their previous interactions using this method, which may be used for machine-to-machine communication. In their work, Lomotey, Pry, and Chai [167] first emphasize the use of provenance through the design of algorithms to confirm the origin of IoT-based data, and then they propose developing completeness techniques through visual analytics to trace data packets through the complete route in an IoT network. They present an improved system provenance mechanism in their study to achieve traceability. The technique is based on an associative logic to decide all connections established in a machine-to-machine communication between IoT nodes. To create links between device communication and data propagation to the n-th degree, a statistical lexical chaining based on the Adjusted Rand Index (ARI) is suggested as an alternative to knowledge-based methodology. Because data propagation pathways and object-to-object communications can be identified, the proposed IoT architecture makes traceability easier. Based on their findings, the suggested system demonstrates that, in terms of identifying linkability, unlinkability, and availability, the ARI is more accurate than the knowledge-based methodology. Additionally, visual analytics is offered to give a clearer understanding of interconnection in IoT nodes using visualization graphs such as HyperTree Graph, the Weighted Graph, and the combination of SpaceTree and RGraphs.

3.2.10 Path Difference

Path difference is the sum of an indicator variable which represent whether the actual packet path for the next hop is the same as the parent node of this packet along the routing path. Parent information can be used in reference packets for path reconstruction if the path difference value is equal to zero. If not, more time will be required to record the real forwarders. In order to minimize the message overhead, it is better to have a minimal path difference value.

In big-scale sensor networks having lossy links and complex routing dynamics, Gao et al. [168] mention the ineffectiveness of current path reconstruction techniques. The authors present Pathfinder, a cutting-edge path reconstruction method. Pathfinder takes advantage of temporal correlation between a group of packet paths at the node side and effectively compresses the route information with path variation; at the PC side, Pathfinder determines packet routes from the compressed data and uses smart path speculation to reassemble the data packet paths with a high reconstruction ratio. With the use of comprehensive simulations and traces from a large real-world sensor network, they construct Pathfinder and evaluate its performance against the two most similar approaches. Results indicate that Pathfinder performs noticeably better than MNT [169] and PathZip [126] in a number of network setups. Their findings show that Pathfinder achieve both low transmission overhead and high reconstruction ratio.

3.2.11 Logging-based

Concerns over creating appropriate forensic investigation models were highlighted by cybercrime occurrences seen in IoT networks. Every attack on an IoT network leaves behind some evidence of it, but the primary difficulty is locating, gathering, and correlating that evidence for accurate forensic analysis. In order to provide answers to specific investigation queries, it can sometimes be rather challenging to determine the linkages between discrete information gathered from IoT devices and network level activity. To tackle this issue, forensic investigators can benefit from provenance logging. Over time, network provenance generates a vast amount of information. IoT devices reduce this cost by using template systems, which require them to log just basic network characteristics as the traditional device logging.

Sadineni, Pilli, and Battula [170] propose a template-based provenance approach, ProvLink-IoT, to provide reliable forensic analysis in IoT networks. ProvLink-IoT is developed to analyze link-layer attacks. Many open source tools are used to implement the system in a simulated environment to approve its robustness in correlating evidence that is found in link-layer provenance. Based on provenance logs and network data gathered from the network, traceability graphs are generated in both normal and attack scenarios. The approach is studied using 6TiSCH network stack. To analyze the effects and conduct forensic analysis, the authors applied three link-layer attacks to the Time Slotted Channel Hopping (TSCH) and 6top (i.e. operational sublayer) layers of the 6TiSCH network. For performance evaluation, they used storage overhead and provenance growth rate.

3.2.12 Frameworks using Storing Methods

Researchers have used various storage methods, using database placement strategies, to develop frameworks designed for diverse IoT applications. These methods include a number of techniques for storing and querying provenance information within the database. They offer management solutions to ensure the secure storage of provenance data while meeting security requirements. These frameworks offer a complete cycle for tracking the history of provenance records, covering everything from initial capture to storage and analysis. In this section, we present selected works that have designed such frameworks for managing data provenance.

1. *Think-and-share optimization:* Alkhalil, Ramadan, and Ahmad [171] introduce a bio-inspired approach that uses the processes of human thinking to enhance data provenance in WSN. Their proposed Think-and-Share Optimization (TaSO) algorithm modularizes and automates data provenance management in enterprise-deployed WSN. The TaSO algorithm is designed of four phases: Think, Pair, Share, and Evaluate. The authors assess the effectiveness of their TaSO algorithm by evaluating key metrics such as connectivity percentage, closeness to the sink node, coverage, and running time.

2. *Selective instrumentation:* In their work, Wang et al. [172] introduce the ProvThings framework, designed for the capture, administration, and analysis of data provenance within IoT platforms. Their approach introduces a selective instrumentation algorithm that reduces the collection of provenance data by identifying security-sensitive sources and sinks. This method provides a means to trace complex chains of inter dependencies among IoT components. Additionally, the authors created a prototype of ProvThings for the Samsung SmartThings platform and evaluated its effectiveness against 26 IoT known attacks. The IoT provenance model is based on the W3C PROV-DM [173]. The results indicate that ProvThings imposes only a 5% overhead on physical IoT devices while enabling real-time querying of system behaviors.

3. *Trust management:* Elkhodr and Alsinglawi [174] extends a previous work that introduced a provenance-based trust management solution to assure data provenance [175]. Their Internet of Things Management Platform (IoT-MP) [175] establishes trust relationships between communicating IoT devices. This work uses the existing capabilities of IoT-MP to enhance privacy protection in IoT networks. Furthermore, it introduces a Data Provenance module aimed at enabling the retrieval of data origins and access to device's history, including the

networks it has interacted with. The method propose three states for IoT devices registered in the IoT-MP platform: New Resident, Visitor, and Returning Resident. Additionally, it includes a database module that is reconstructed to adapt to the changes in the architecture.

Further analysis of these techniques and their analysis is provided Appendix A in Tables A.1 to A.5. In Table A.1, each technique is analyzed based on the provenance encoding method, provenance storage method, application, security analysis, advantages and shortcomings. We present the main performance metrics that are needed to evaluate any data provenance approach in IoT networks that are provenance length, energy comparison, data packet size, link-loss rate, detection rate, false positive rate, false negative rate and computation time. We take into consideration for the analysis of each selected approach these performance metrics which are presented in Table A.3. Moreover, in Table A.2, we provide details on the provenance category, implementation, and simulation software. This resource is presented to help in exploring the different implementation tools and simulation software associated with each provenance category. In Table A.4, we provide the security requirements for data provenance integration in IoT networks. The security attacks, as illustrated in Figure 2.7, undergo a comparative analysis presented in Table A.5. This table shows the attacks studied in each of the studied techniques.

3.2.13 Discussion

Data provenance can be used to detect errors in the different stages of data generation and processing enabling the system to detect the nodes that produced those errors [176–178]. In addition, storing detailed information about data in the provenance record allows for data recovery when data is no more usable to ensure availability and achieve normal data communication within different network entities [179]. Data provenance enhances data readability when it includes detailed information about the data's origin and processing [180]. Furthermore, data provenance enhances data clarity, ensure data reliability, and facilitates data reuse [181]. One of the most important features of data provenance is providing the system with the ability to asses the trustworthiness of generated data through different secure provenance techniques [182]. However, the size of provenance records depends on the number of nodes involved in generating the provenance information and the quantity of attributes to be included in each record. To obtain a provenance chain that satisfies security requirements, it is essential to include many attributes that describes the origin, transformation, data path, data quality and any modifications the data has undergone. In large-scale networks with an increasing number of nodes, the size of provenance information grows rapidly, posing significant challenges in terms of storing and querying these records. This can limit the efficiency of provenance analysis. There should be a trade-off between the number of attributes included in the records and the limitations in the computational capabilities of the system. This requires to determine the most important security requirements of the system and the needed attributes that satisfies it with minimum storage and querying overhead.

The presence of constrained network components and limited resources in IoT environments present several challenges for data provenance schemes. Based on the analyzed literature, any data provenance scheme designed for such environments must address the following challenges [183–185]:

- *Minimizing bandwidth consumption and ensuring high data processing and throughput* in the provenance infrastructure.
- *Properly indexing the provenance records.* The complete provenance is often extensive. Queries typically involve more complex operations than simple name based databases. Users need to search for specific data sets based on subsets of attributes and values within the provenance chain. As different users may query different attributes depending on their objectives, sensor data storage systems must provide efficient indexing structures for databases across different dimensions.
- *Efficiently managing the size of the provenance data.* Typically, in large-scale systems, the size of provenance records tends to exceed the size of the actual data when they are processed and transmitted.
- *Establishing secure transmission of provenance information and enabling detection of malicious attacks and fast response from provenance management systems.* Moreover, efficiently storing provenance data. As original data undergoes multiple hops and accumulates complex processing history, the size of the resulting provenance information can become very large.
- *Querying provenance data.* There should be a flexibility in the reconstruction of the provenance when a certain authorized entity queries it from the provenance storage entity.
- *Collecting provenance information.* Different data features may be collected and stored based on the service and requirements of the IoT system. In many cases, the system requires to collect different operations or information about forwarding entities. Hence, it is important to deal with the challenging collection of these records from the generation of data to its final receiving node.

3.3 Machine Learning-based IDS

There have been previous works implementing IDS in networks using different ML and DL techniques, as we discuss in this section. With the advancement of computer networks, securing its infrastructure has made intrusion detection a very important issue to implement. Various ML methods, such as Fuzzy Logic, Decision Trees (DT), K-Nearest Neighbors (KNN), SVM, Random Forest, Naive Bayes, Logistic Regression and Artificial Neural Networks (ANN) approaches, are used in IDS to distinguish between normal network activity and malicious intrusions. SVM, in particular, has shown better performance compared to standard classification methods, allowing several researchers to propose several SVM-based IDS solutions. Despite the advantages of SVM-based IDS in terms of detection accuracy and learning speed over traditional algorithms, the issue of misclassification of attack packets still needs improvement. Furthermore, a number of different techniques have been proposed in the literature aimed at enhancing traditional ML algorithms.

In a study by Tao, Sun, and Sun [23], a new IDS is introduced to improve detection rate, false positive and false negative rates. This system, called FWP-SVM-GA, uses a Genetic Algorithm (GA) to improve the performance of an SVM algorithm. The GA first selects the most relevant features from the data. Then, SVM parameters are optimized to achieve the highest accuracy. After training the model, the FWP-SVM-GA can effectively identify and categorize unusual network activity. Focusing on a single attribute, packet arrival rate, instead of the complex features often found in online datasets, Jan et al. [186] propose an SVM-based classifier for a lightweight IDS evaluated using CICIDS2017 network traffic dataset and a generated dataset using MATLAB™ according to Poisson distribution. The classifier's performance using linear, polynomial, and radial-basis kernel functions is analyzed and compared to other ML techniques like neural networks, KNN, and DT.

Ravi, Chaganti, and Alazab [187] propose a thorough method for network intrusion detection, focusing on merging features from hidden layers of recurrent models. They explore traditional ML algorithms like Naive Bayes, Logistic Regression, KNN, DT, and Random Forest, as well as recurrent DL models –such as Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRUs)–. They also investigate reducing the complexity using Kernel Principal Component Analysis (KPCA) and assess performance across various intrusion datasets, leading to ensemble meta-classifiers and feature combination for improved detection.

A DL model based on LSTM for multiclass attack detection classification is proposed by Rao and Suresh Babu [188]. Enhancing the classifier's performance involves hybridizing a convolutional neural network called LeNet 5 and LSTM, and implementing Imbalanced Generative Adversarial Network (IGAN)-based class imbalance. This process can prevent the unnecessary time and space losses involved with oversampling as well as the loss of important samples due to random undersampling. Alghushairy et al. [189] propose a Network Outlier Detection System (NODS) for classifying normal and attack network traffic. The system uses SVM and Gaussian Naive Bayes (GNB) classification algorithms to categorize the behavior of incoming network connections impacting a computer network. Both algorithms were built and assessed using network traffic datasets. Data mining preprocessing steps for network flow data, alongside optimizing Radial Basis Function (RBF) control parameters and the GNB smoothing parameter, prove to enhance the overall effectiveness of the proposed NODS.

3.3.1 Discussion

While many techniques that use SVM or other ML and DL approaches for intrusion detection have been proposed lately, there are still some drawbacks with these approaches, such as: (1) Some algorithms might struggle with complex attacks that deviate from established patterns. Novel attacks or zero-day exploits, for instance, might bypass the detection capabilities of these models; (2) the effectiveness of ML models heavily relies on the training data (e.g., if the training data is limited or does not encompass a wide range of attack types, the model might not be able to generalize well to unseen attacks and, later on, this can lead to false positives when the system encounters attacks not included in its training set); and (3) some methods might establish static thresholds for identifying anomalies. While our third observation does not exclude detection of known attack patterns, attackers can adapt their methods to remain below these thresholds. This can trick an IDS to misclassify attacks as legitimate traffic. To handle the problem, we propose an augmented two-layer IDS approach using SVM for the first layer of classification and a zero-watermarking approach using provenance information as a second layer of detection. Even if the IDS misclassifies an attack in the first process, the other layer might still be able to prevent it from compromising the system.

3.4 Open Issues and Research Challenges

3.4.1 Open Issues

We identify first some representative open issues and potential research directions which can help in exploring different aspects of data provenance integration in IoT networks and ML-based IDS.

Data Provenance Techniques:

- ***Privacy and security of data stored on the blockchain:*** Integrating a blockchain with the IoT network provides a robust database that provides integrity to the provenance records stored as transactions. These records include sensitive information about the creation, processing and transmission of the data packets. Hence, it is essential to preserve privacy while maintaining the integrity of the complete provenance chain. Attackers may try to obtain secret information by analyzing the provenance chain which the blockchain holds. It is important to secure the records stored on the blockchain.

- ***Range of studied attacks:*** Most of the proposed approaches focus on a very few and limited number of attacks. In many cases, researchers take into account either data attacks or provenance attacks. Attackers aim to deceive the system by targeting both data and provenance records. The majority of existing works focus on the forgery and modification attacks, which are the most common threats. It is clear that almost all the proposed methods do not consider chain tampering. As mentioned above, a complete provenance information of a particular data packet is created by connecting provenance records in a chain. The objective of an attacker is to change this chain's sequence and contents to affect the accuracy and dependability of the provenance data. Hence, to develop any solution for provenance in IoT, the robustness of the system against data and provenance attacks must be taken into account.

- ***Watermarking as a solution:*** Maintaining data integrity through procedural requirements is the major goal of watermarking system design. These requirements cover node ownership, data integrity, and bandwidth establishment. Embedding a watermark with data is the main technique to achieve the requirements in watermarking schemes [33, 63, 183, 190]. The criteria used in the generation of watermarking schemes is not restricted to WSN; it also includes the security of multimedia programs [154, 191–193] and database formation [194–196]. This wide range of domains allows watermarking to be one of the lightweight solutions in limited networks such as IoT. Hence, the features of data hiding that watermarking provides a new provenance encoding technique to store provenance records in an efficient way in-terms of storage and transmission overhead. Only one relevant work [154] considered watermarking for data provenance. They consider inter-packet timing characteristics for embedding provenance information and does not use the data features and watermarking embedding with the data packets. This technique can be used to develop lightweight solutions for the limitations of IoT networks in terms of storage and computation.

- ***Lack of complete provenance management system and narrow focus:*** The literature is still missing a robust provenance management system that takes into account all required procedures from collecting, storing and analyzing provenance information. The system should satisfy the security requirements while maintaining storage and data management issues. To achieve this, these systems should be designed based on the architecture of IoT networks, taking into account various factors, such as communication protocols, computational capabilities, storage limitations, node coordination, database integration, and potential security attacks against data and provenance records. Additionally, it is essential to thoroughly test the robustness of the proposed provenance system, assessing its performance across all relevant metrics. This approach ensures that the system complies with all the necessary objectives, rather than evaluating its performance based on only a subset of criteria. Many of the existing approaches fail to adequately address these important issues while maintaining the security of provenance data. Furthermore, the proposed provenance approaches focus on a specific application for a specific domain. The fast development of interconnected applications of different domains require adapting these techniques to be applied to multiple domains that include different structure of provenance records.

- ***Lack of efficient query and provenance support:*** In IoT networks, which involve numerous interconnected devices generating huge amounts of data, efficiently querying and managing the history and transformations of this data is a complex task. The absence of effective mechanisms for querying and handling provenance can hinder the ability to trace the origin and processing of data, resulting in a limitation in providing security, freshness, and scalability in IoT applications. Addressing this issue involves developing more efficient and scalable methods for querying and managing provenance data in the unique and dynamic environment of IoT networks.

- ***The storage of blockchain is expensive:*** Blockchain relies on a decentralized and distributed network of nodes, each maintaining a copy of the entire blockchain. While this redundancy enhances security and fault tolerance, it significantly increases the storage requirements. In IoT networks, with a large number of devices generating data, the continuous increase in the size of the blockchain can grow rapidly. Additionally, achieving scalability in blockchain networks, especially in public networks, is a known challenge. As the number of transactions and participants increases, maintaining the performance and efficiency of the blockchain becomes a complex task, hence requiring more robust infrastructure and resulting in an increased operational costs. In order to overcome the issue of expensive storage in blockchain-based data provenance for IoT networks, researchers should develop a more scalable and efficient validation mechanisms, optimization of data storage techniques, and alternative blockchain architectures in IoT that balance the advantages of security with the need for cost-effective scalability in the context of IoT data provenance.

- ***Integrating data provenance with IDS:*** Adding data provenance to system logs transforms them from simple records into rich sources of security insights which are used in traditional Host-based Intrusion Detection Systems (HIDS). This helps investigators to detect security threats with greater accuracy, minimizing computational time and resources on false alarms, as extensive research has documented [197–210]. Data provenance can be integrated with HIDS to improve intrusion detection. IDS based on provenance, known as Provenance-based Intrusion Detection Systems (PIDS), leverage data provenance to identify intrusions. This involves examining not only the properties of system entities but also dealing with the causal relationships and information flow within a provenance graph [211]. However, the integration of data provenance and IDS in IoT networks has not been widely studied. This opens a crucial field for researchers to study, addressing the significance of integrating data provenance in IoT networks and exploring its various applications for enhancing intrusion detection.

ML-based IDS:

- ***Attack detection:*** More attack types must be covered, and IDSs must be proposed for a wide range of attack types rather than focusing on known ones. In the evolution of IoT networks and their applications, many new types of attacks are being deployed by malicious parties and eavesdroppers to have unauthorized access to the network. Such attacks need to be introduced to the research efforts in designing and implementing security solutions rather than just concentrating on the traditional and known attacks. In the process of detecting attacks, it is very important to reach the minimum processing time for the procedure to be completed. In real-time detection and delay-sensitive services, researchers need to establish algorithms and procedures that require very small processing time. This will also help to reduce the consumed energy and power in the network.

- ***Emerging technologies:*** IDSs must be proposed for different IoT technologies. Many researchers and organizations have proposed communication technologies and standards for IoT applications [212]. These technologies are used in routing, communication and integration between the internet and the network. The most popular technologies used in IoT networks are: IEEE802.15.4 for physical and medium access control layers, Bluetooth Low Energy (BLE) —an evolution of Bluetooth technology for low power devices—, WirelessHART for industrial process control, Z-wave for automation of small businesses and homes, the RPL routing protocol, 6LoWPAN standard to adapt the IPv6 packet to IEEE802.15.4, and CoAP and MQTT protocols for the application layer. These technologies are one of the main characteristics to be explored by IDSs to develop a system for the detection of security threats. New technologies, such as CoAP, BLE, Z-Wave and WirelessHART, which are commonly found in the market, need to be addressed and studied by IDS solutions rather than concentrating efforts only on previous technologies in IoT networks.

- ***Performance analysis:*** Energy and power of network nodes must be studied along with detection accuracy and processing time. These two attributes are very important in IoT networks, since IoT devices may have low energy and power capabilities and most of the security solutions require more energy and power than the devices can hold. This opens the challenge of lightweight solutions to be adopted by researchers. Moreover, researchers should take into account the challenges in security for IoT networks, such as scalability, hardware limitations of nodes, services that are delay-sensitive and the interaction between all layers of the IoT network architecture. Delay-sensitive services are critical and should be extensively studied. Any security problem in such services leads to dangerous consequences, especially when these services are found in medical, military and home appliance applications. Finally, one of the important tools to be used in decision making and especially in detecting attacks is the receiver operating characteristic curve (ROC). Most of reviewed systems selected in our work from recent published papers used ROC curves in their study. ROC curves compare the two operating characteristics: True Positive Rate (TPR) and False Positive Rate (FPR). This is used in the process of classifying whether we have attack or not.

3.4.2 Research Challenges

The integration of data provenance with IoT and the existing ML-based IDS raises critical security concerns. In this section we summarize some of the important challenges observed.

A first challenge deals with *provenance records processing and storage*. Indeed, provenance information may be larger than the data itself, since provenance records gets larger as the number of forwarding nodes increase. While some applications may involve a small-scale network, there is a need for a complete provenance representation in the provenance chain. This representation requires a number of attributes that need to be stored in the provenance records which is also an issue with storage.

Additionally, the diversity of IoT devices and sensors leads to a variety of data types and formats. Provenance records must also accommodate to the different types, leading to a larger and more complex datasets. Also, to provide useful information from the history and processing of data, provenance records need to be detailed and granular. This granularity adds to the size of the provenance dataset, especially when capturing detailed information about each data transformation.

In large-scale IoT deployments, devices are often interconnected in complex networks. Provenance records need to traverse these networks, leading to additional metadata and size considerations as data moves across multiple devices and systems. Moreover, IoT devices may have limited storage and bandwidth capacities. Transmitting, storing, and managing large provenance datasets can load these resources, impacting the overall performance and efficiency of IoT networks. There exist some solutions to overcome this problem such as compression techniques. These techniques have high loss rate and increase the computational complexity of the system with limited resources. It is challenging to take into account the required information to be stored in provenance records and maintain, at the same time, processing and storage overheads. Addressing the challenge of provenance size in IoT networks requires careful consideration of storage solutions, data compression techniques, and protocols for optimizing data transfer and processing. Balancing the need for detailed provenance information with the constraints of IoT environments is very important for effective and efficient provenance management in IoT networks.

A second challenge deals with *provenance attachment to data packets*. Making sure provenance flows with data is a challenging task. Provenance is a type of metadata which increases in size as the number of forwarding nodes increases. In an IoT network, data often traverses diverse and resource-constrained devices, making it essential to track the origin, transformations, and actions performed on the data. Embedding provenance information directly into data packets can be challenging due to constraints such as limited bandwidth, energy, and processing capabilities of IoT devices. Maintaining a balance between the requirements of data provenance and the limitations of IoT networks is essential. Addressing this issue involves developing efficient and lightweight methods for attaching and transmitting provenance records with data packets and ensuring that the provenance information is captured throughout the data path across the IoT network without causing significant overhead or affecting the functionality of the devices.

Two additional challenges are related to *provenance collection* and *provenance privacy*. The former, provenance collection, deals with the large amounts of data generated, usually in real time, in IoT network. This requires the need for tracking its origin, transformations, and actions resulting in an overhead for collecting provenance records. Additionally, the heterogeneity of IoT devices introduces complexities in standardizing provenance formats, as different devices may generate diverse data types and use different communication protocols. The resource-constrained nature of many IoT devices makes provenance collection more difficult, as it requires energy consumption, storage limitations, and processing capabilities. Furthermore, extracting the provenance information from the networks that are designed without considering the possibility of need for querying provenance information is a challenging task.

Provenance privacy emerges as a challenge in IoT networks due to the sensitive nature of data and the wide number of different interconnected devices. Provenance records, which trace the origin and transformations of data, provide details about the context and usage history of information. In an IoT system, where diverse and personal data is continuously generated, maintaining the privacy of individuals becomes essential. The challenge is to balance the need for important provenance information while protecting user privacy. Dealing with these issues requires careful consideration of secure transmission, encryption protocols, secure storage and access controls to ensure that while provenance records remain effective for achieving security requirements, they do not compromise the privacy rights of individuals and entities whose data contributes to the IoT network. Developing robust privacy-preserving mechanisms becomes essential to address these concerns and achieve trust in the secure development of IoT networks.

Addressing ML-based IDS necessitates additional research and presents numerous challenges that must be addressed. In signature detection, predefined attack patterns need to be matched with the current behavior of the network. These signatures are stored on the device and each signature matches a specific attack. Generally, signature detection methods are simple to use. However, they need a signature for each attack and should store this signature on devices. This requires storage capabilities and knowledge of each attack. These requirements grow as the number of attacks increases.

Anomaly detection techniques determines the ordinary behavior and uses it as a baseline to detect anomalies in the network. Deviations from this behavior is considered as an attack. These techniques are able to detect any attack and can be deployed in different environments, but it has high false positive rates and high false negative rates. Since some deviations from the normal behavior may be ordinary and a number of attacks may have a small deviation which is considered within the baseline [213]. Moreover, the enormous quantity of data generated in IoT networks lead to the need of intelligent tools to assist IDSs. These tools are established using Machine Learning techniques. Nowadays, IDSs are developing rapidly with the presence of these techniques. On the other hand, the robustness of these systems becomes questionable in the presence of adversarial attacks. A new arising framework is being developed known as Generative Adversarial Network (GAN) used to evade and deceive any IDS. It is used to fool machine learning algorithms. For this reason, the development of GAN should be used by researchers to improve recent IDSs and establish new ones to reach a system that can not be broken by robust attacks.

A final solution to handle attack detection is the use of challenge-response mechanisms, e.g., via watermarking authentication techniques. Such techniques are popular for the protection of Wireless Sensor Networks, specially in the context of IoT environments [214, 215]. From all the limitations and restrictions mentioned before, watermarking can be used to implement anomaly detection [216]. This is being tested in a number of new projects that consider watermarking as a solution for secure transmission, data integrity, authentication and confidentiality in cyber-physical systems. Watermarking techniques are attractive since they can be deployed with lightweight calculations and less energy consumption. Hence these solutions are power efficient, can be applied for large networks with no additional overhead on network communication and storage capacity of nodes, and reduce end-to-end delay [217–223].

Chapter 4

Zero-watermarking for Data Integrity and Secure Provenance

4.1 Introduction

The IoT is integrating the Internet and smart devices in almost every domain, such as home automation, e-healthcare systems, vehicular networks, industrial control and military applications. In these areas, sensory data, which is collected from multiple sources and managed through intermediate processing by multiple nodes, is used for decision-making processes. Ensuring data integrity and keeping track of data provenance is a core requirement in such a highly dynamic context, since data provenance is an important tool for the assurance of data trustworthiness. Dealing with such requirements is challenging due to the limited computational and energy resources in IoT networks. This requires addressing several challenges such as processing overhead, secure provenance, bandwidth consumption and storage efficiency.

This chapter introduces the theoretical formulation of our zero-watermarking approach for data integrity and secure provenance in IoT networks, including the system model, security assumptions, threat model and algorithms. We propose a zero-watermarking approach to verify data integrity at each hop of an IoT environment (i. e., from source node to gateway). Additionally, we ensure secure provenance of sensory data using a tamper-proof database connected to the gateway and to each node in the network. Data provenance information is used in the watermarking generation process and embedded with captured data packets to be used in the data integrity and secure provenance verification. The system is composed of the following entities:

- **IoT Source Node:** a small sensing device that collects data from surrounding environment. At each node, watermarking generation and embedding processes are applied to each data packet captured. The device performs some operations and communication procedures in the network.
- **IoT Intermediate Node:** an advanced sensing device with more power and computation capabilities, responsible for forwarding data packets from source nodes to the base station. It also performs watermark generation, embedding and storing on the received data packets. Data integrity will be checked for each data packet forwarded at this stage.
- **Base Station or gateway:** receive the forwarded data packets for data processing. Checks data integrity and provenance recovery, and applies the attack detection procedure.

- **Network Database:** a secure network database which is connected to the network nodes and gateway. It stores provenance information at each hop of the data path.

Notice that the suggested coordination approach is consistent with other strategies already in place on IoT-like deployments, such as tasks for uplink interference management, frequency hopping, or frequency allocation, among others, which combine distributed coordination tasks under the control of central entities [224].

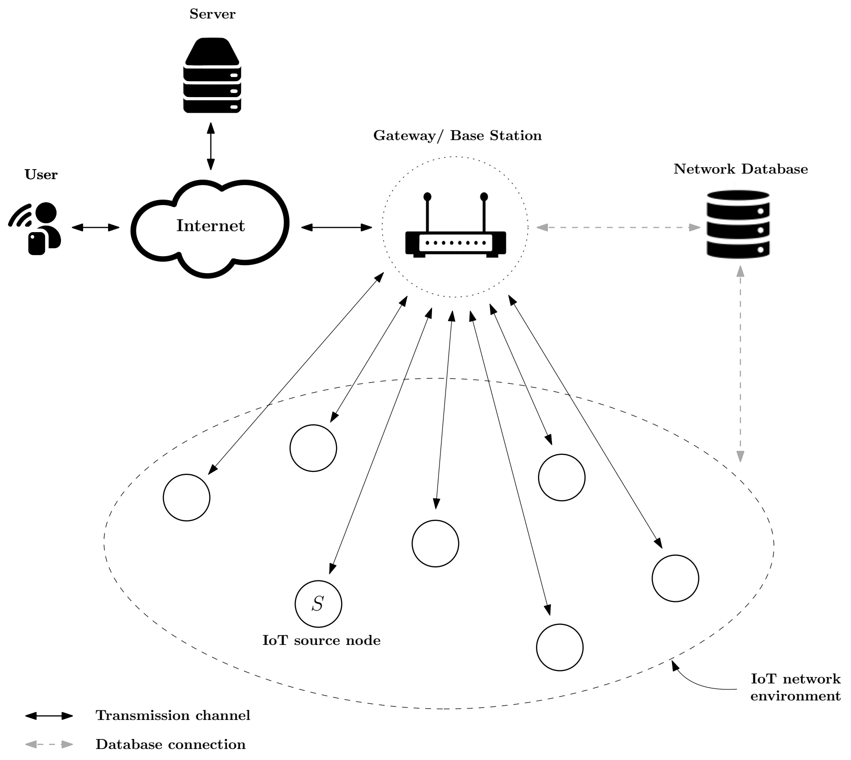


FIGURE 4.1: Single hop network model. The source nodes are directly connected to the base station and the tamper-proof network database through a transmission channel. The gray dotted two-way arrow represents the store-query connection between the network entities and the database.

4.2 Network Model

In the proposed network model, the network is assumed to consist of N IoT devices that are distributed in an IoT network. The network is deployed in an $L \times W$ area. Devices are connected to a gateway or base-station that is the management and controller unit. Nodes are of two types: normal sensor nodes and intermediate sensor nodes. Sensory data is routed from normal source nodes to the gateway through intermediate nodes. This implies that intermediate nodes have m times more energy than normal nodes (i. e., energy of a normal node $= E_0$, energy of intermediate node $= E_0 + m \times E_0$). Furthermore, a tamper-proof database is connected to the gateway and to each node. It is assumed that the database cannot be compromised by the attacker. The model consists of two main scenarios: single-hop and multi-hop. In the single-hop scenario, IoT devices transmits sensory data directly to the gateway through the transmission channel, as shown in Figure 4.1. However, in the multi-hop scenario, sensory data is routed from the source node to the gateway through intermediate nodes as shown in Figure 4.2.

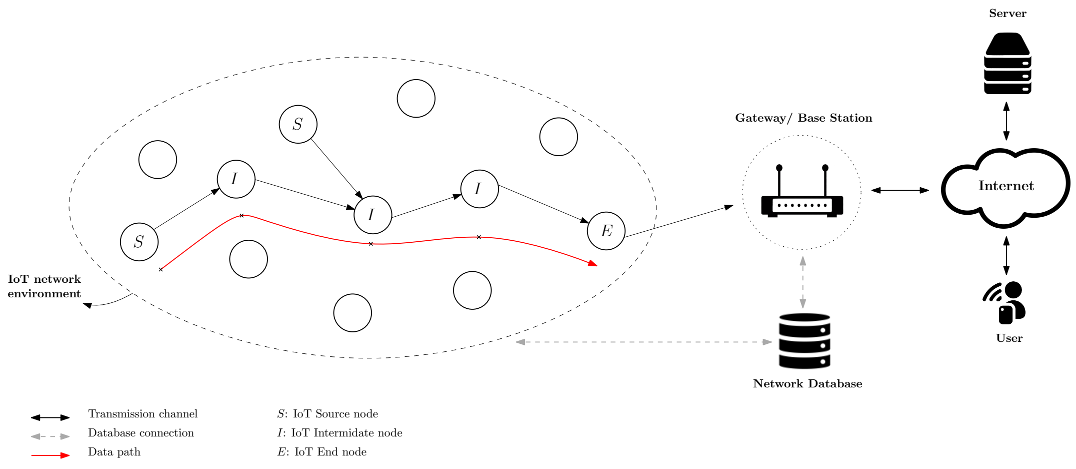


FIGURE 4.2: Multi hop network model. The source nodes are connected to the base station through multiple intermediate nodes. The gray dotted two-way arrow represents the store-query connection between the network entities and the database. The red arrow from the source to the end node represents the complete data path of a particular data packet.

4.2.1 Single Hop Model

In this scenario, the process of data integrity and secure provenance is composed of different units that form the overall system model as described in Figure 4.3. Sensor nodes capture data from the surrounding environment and send it to the feature extraction unit. The IP address of the IoT device, data capturing time and a generated unique packet sequence number are extracted and encrypted to generate a sub-watermark in the first sub-watermark generation unit. This sub-watermark forms the provenance record of a particular data packet. A hash function is used to generate another sub-watermark that is concatenated with the first one to generate a final watermark. The generated final watermark is then stored in a tamper-proof database. The data packet is then sent to the gateway through the transmission channel. At the gateway, data is received and forwarded to the zero-watermark re-generation unit. After the re-generation process, the stored watermark is queried from the database to be compared with the re-generated watermark in the watermark verification unit for provenance integrity check. A double verification procedure is applied for both integrity and provenance. At this stage, the gateway detects whether data and provenance is altered or not and performs either attack procedure or validates the origin of data received.

4.2.2 Multi Hop Model

The watermark generation, embedding, extraction and verification processes of the multi-hop scenario are shown in Figure 4.4. In this model, the data capturing time and IP address are extracted from sensed data packets and a generated packet sequence number are used to generate a sub-watermark, which is then encrypted using

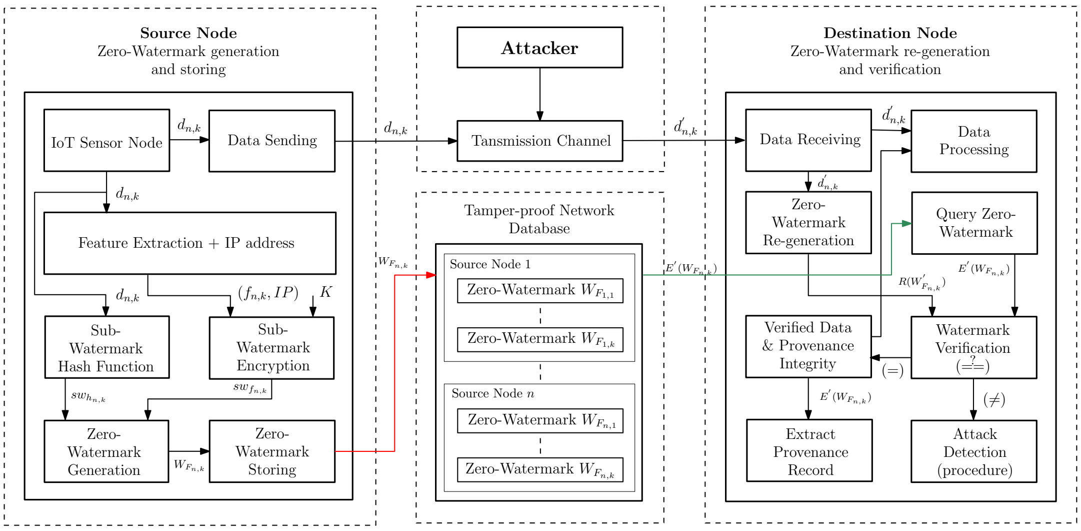

Source Node
Zero-Watermark generation and storing
IoT Sensor Node
Data Sending
Feature Extraction + IP address
Sub-Watermark Hash Function
Sub-Watermark Encryption
Zero-Watermark Generation
Zero-Watermark Storing
Attacker
Tansmission Channel
Tamper-proof Network Database
Source Node 1
Source Node n
Destination Node
Zero-Watermark re-generation and verification
Data Receiving
Data Processing
Zero-Watermark Re-generation
Query Zero-Watermark
Verified Data & Provenance Integrity
Watermark Verification
Extract Provenance Record
Attack Detection (procedure)

FIGURE 4.3: Zero-watermark generation, storing and verification block diagram in single hop scenario. The dotted blocks represent the entities involved in the network. The red arrow represents the *store* function from a source node to the database. The green arrow represents the *query* function from the database to the base station.

a secret key and concatenated with a generated hash value of data payload to construct the final watermark. The first sub-watermark or provenance record is stored in the network database and the final watermark is concatenated with the sensed data $d_{n,k}$ to be forwarded to the next intermediate node through the transmission channel. At the next hop node, the watermarked data is received. The watermark is then extracted from the received data packet. The received data is then used to re-generate a new sub-watermark that will be forwarded to the verification unit along with the extracted watermark and a queried provenance record. The intermediate node takes a decision whether an attack is detected or not. Based on this decision, it performs an attack detection procedure or generates the next-hop watermark that undergoes the same procedure of the source node (generation, embedding and storing); it uses new extracted features and provenance information. The watermarked data reaches the final destination (i. e., gateway) through transmission channel. The last embedded watermark is separated from the watermarked data and a final sub-watermark is re-generated. The data integrity unit accepts extracted watermark, re-generated sub-watermark and queried provenance record as input values to check whether data or provenance is modified or not. After that, the gateway performs two procedures based on the verification result: attack detection procedure or provenance validation.

4.3 Security Assumptions

We consider a set of security assumptions for the proposed system as follows:

- Nodes in the network are not trusted entities, i. e., , these nodes may be malicious. The protocols provided to guarantee the secure transfer of provenance information that is applied in intermediate nodes and gateway are proven to work properly in the presence of malicious nodes.
- The network database is a trusted and secure entity, it cannot be compromised by an external attacker to access or use its content.

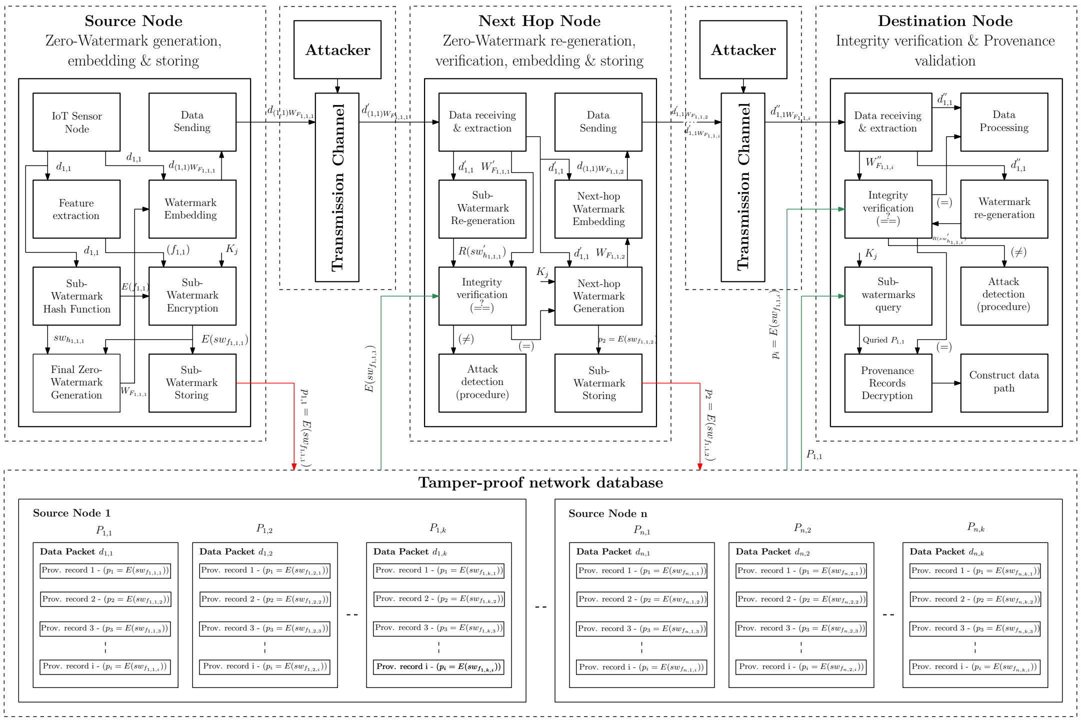

Source Node
Zero-Watermark generation, embedding & storing
IoT Sensor Node
Data Sending
Feature extraction
Watermark Embedding
Sub-Watermark Hash Function
Sub-Watermark Encryption
Final Zero-Watermark Generation
Sub-Watermark Storing
Attacker
Transmission Channel
Next Hop Node
Zero-Watermark re-generation, verification, embedding & storing
Data receiving & extraction
Sub-Watermark Re-generation
Next-hop Watermark Embedding
Integrity verification
Next-hop Watermark Generation
Attack detection (procedure)
Sub-Watermark Storing
Destination Node
Integrity verification & Provenance validation
Data Processing
Watermark re-generation
Sub-watermarks query
Attack detection (procedure)
Provenance Records Decryption
Construct data path
Tamper-proof network database
Source Node 1
Source Node n
Data Packet
Prov. record 1
Prov. record 2
Prov. record 3
Prov. record i

Figure 4.4: Zero-watermark generation, storing and verification block diagram in multi-hop scenario. The dotted blocks represent the entities involved in the network. The red arrows represent the *store* function from a source node to the database. The green arrows represent the *query* function from the database to the base station.

- To allow only authorized gateways to access the network database and query provenance information, only registered gateways are authorized and the query is applied after checking the integrity of the data received from the last hop of the data path.
- The database temporarily stores provenance information of a data packet at each hop from source to destination, only after being proved as trusted data. This linage can be retrieved once (by authorized gateway) and then it is removed from the database.
- The hashing functions used in the system are secure and cannot be inverted.
- The communication of extracted data features and provenance information (sub-watermarks) between source nodes and intermediate nodes, and between intermediate nodes and gateway, must be secure. Provenance information is encrypted using symmetric cryptography and selected data (for integrity check) is hashed using a one-way hash function.
- Symmetric cryptography is restricted to the encryption of short binary strings forming the extracted data feature sub-watermark. Source node, intermediate node and gateway share secret-keys to be used in different steps of the algorithms (encryption/decryption).
- Secret keys are changed and redistributed after a short random number of watermark generation processes.

- The zero-watermarking method used to embed provenance information is transparent, fragile and secure enough for IoT network applications.

4.4 Threat Model

There is a number of different attacks that may be applied against the proposed system. Attack models used to deceive and perform security breaches on different network entities require another party to obtain secret information or access the network database. A threat model similar to the one used in [225] is applied in our scheme. The attacker can perform two types of attacks: passive and active attacks (e. g., external attacks).

1. **Passive attack:** An attacker observes secret information by passively eavesdropping data. The attacker performs an eavesdropping attack that aims to obtain data information through listening to data transmission line in the wireless medium.

2. **Active or External attack:** A malicious attack aiming to destroy information by modifying data packets through launching different kinds of operations. An external or active adversary can launch the following main attacks:

   (a) **Replay attack:** Data packets are captured by an adversary and then resent in the future at a different time interval to deceive intermediate nodes or the gateway.

   (b) **Integrity attack:** An attacker inserts false value(s) into the data packet at the transmission channel to deceive the gateway. Also, the attacker may delete elements of the data packet.

   (c) **Modification attack:** In this attack, data is modified by an attacker without knowledge of the data content.

   (d) **Packet drop attack:** Refers to the intentional dropping or discarding of network packets by an attacker. This attack disrupts communication between devices, leading to the loss or interruption of data transmission. The attacker selectively discards packets, targeting specific devices or specific types of data.

   (e) **Database authentication attack:** An attacker aims to detect and retrieve provenance information stored in the network database.

   (f) **DoS attack:** Is a malicious attempt to disrupt the normal functioning of IoT devices or services by overwhelming the network with illegitimate traffic, exhausting its resources, and making it unavailable to legitimate users.

   (g) **Man-in-the-Middle (MITM) attack:** MITM attacks involve intercepting communication between IoT devices and altering the data exchanged.

4.5 Proposed Algorithms

In this section, a precise algorithmic presentation of ZIRCON to conduct the zero-watermarking scheme is described in details. The interaction between source nodes, intermediate nodes, the gateway, and the tamper-proof network database is described in Figure 4.5.

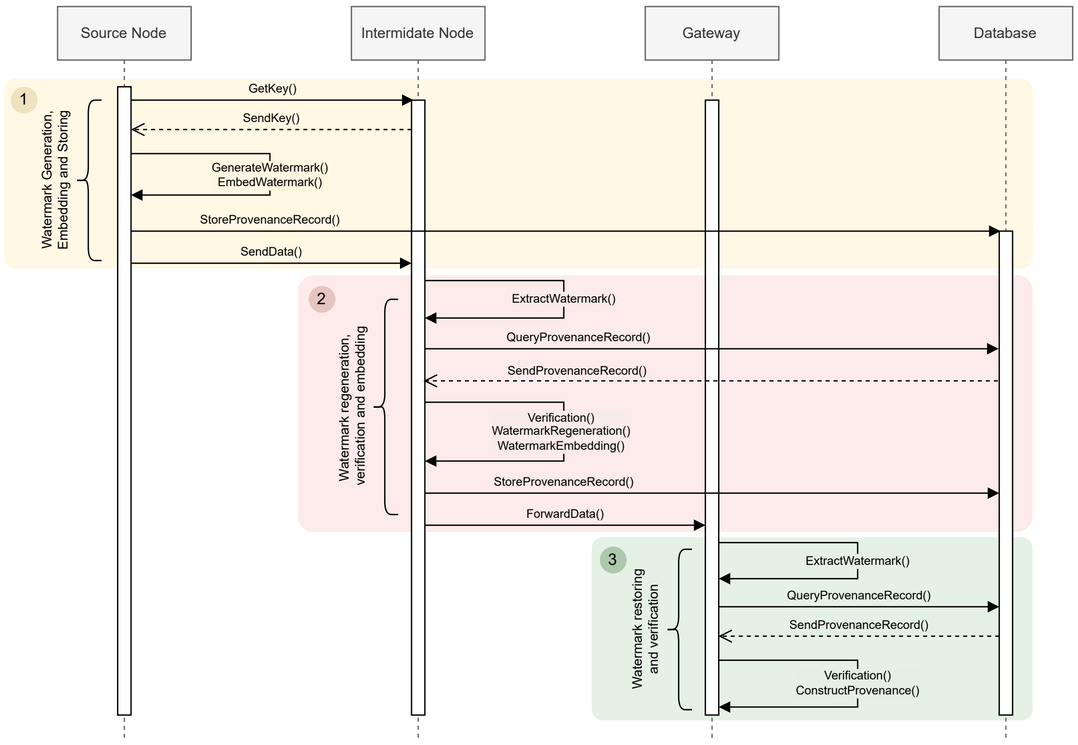


FIGURE 4.5: Sequence Diagram of ZIRCON. The dotted arrows represent a return. Normal arrow represent a message or a request.

Algorithm 1 : Watermark Generation and Storage

```
    input: $d_{n,k}$
    output: $W_{F_{n,k}}$
 1: procedure WATERMARK GENERATION AND STORAGE
 2:     $w_{ip} \leftarrow$ IoT Device $n$ IP Address
 3:     $w_t \leftarrow$ captured data.sensing time $(d_{n,k})$
 4:     $w_{sq} \leftarrow$ packet sequence number $(seq(d_{n,k}))$
 5:     $sw_{f_{n,k}} \leftarrow w_{ip} \mid\mid w_t \mid\mid w_{sq}$
 6:     $p_{n,k} \leftarrow E(sw_{f_{n,k}}) \leftarrow \text{ENC}(K_j, sw_{f_{n,k}})$
 7:     $sw_{h_{n,k}} \leftarrow H(d_{n,k})$                ▷ Select first 8 bytes of hash output
 8:     $W_{F_{n,k}} \leftarrow E(sw_{f_{n,k}}) \mid\mid sw_{h_{n,k}}$
 9:     $STR(W_{F_{n,k}})$
10:     $Send(d_{n,k})$
```

4.5.1 Single Hop Scenario.

In this scenario, two algorithms are proposed: watermark generation and storing, and watermark verification.

1. **Watermark Generation and Storage:** Algorithm 1 describes the process of generating and storing a watermark in a single-hop scenario. It accepts sensed data from the IoT device to produce a final watermark. The algorithm extracts the IP address and the data capturing time from the source node and combines it with a generated unique packet sequence number (seq) to generate a sub-watermark $sw_{f_{n,k}}$ as shown in Lines 2-5 of Algorithm 1. The sub-watermark is then encrypted using the secret key K_j to obtain a provenance record $p_{n,k} = E(sw_{f_{n,k}}, K_j)$. Another sub-watermark $sw_{h_{n,k}}$ is generated from the hash value of data payload using a one-way hash function. These two generated sub-watermarks are concatenated to form a final watermark $W_{F_{n,k}}$ as:

$$W_{F_{n,k}} = E(w_{ip} \,||\, w_t \,||\, w_{sq}\,, K_j) \,||\, H(d_{n,k}) = E(sw_{f_{n,k}}, K_j) \,||\, sw_{h_{n,k}} \tag{4.1}$$

   where || denotes the concatenation operator, $W_{F_{n,k}}$ $(1 \leq n \leq N$, is the final watermark, N is the number of nodes in the network, H is a lightweight (and secure) hash function, and n is the particular node number. The watermark generation algorithm uses the SHA-2 hash function to calculate the hash value. The advantage of using the SHA-2 hash function over other hash algorithms is that SHA-2 has a lightweight feature that uses 65% less memory than other algorithms, such as the MD5 hash function (which has several vulnerability issues), which is needed in resource-constrained networks [226]. After the generation procedure, the final watermark is stored in the network database and the data packet is sent to the base station as shown in Lines 7-9 of Algorithm 1.

2. **Watermark Querying and Verification:** The process of verifying data integrity and validating data provenance in a single-hop scenario is described in Algorithm 2, which takes the received data $d'_{n,k}$ as an input. Then, a re-generation procedure is performed to re-generate the watermark $R(W'_{n,k})$ and the stored watermark $W_{F_{n,k}}$ is queried from the database. A comparison operation is then applied on the re-generated sub-watermark $R(sw'_{h_{n,k}})$ and the queried sub-watermark $sw_{h_{n,k}}$. If the sub-watermarks are the same, data integrity is verified. Then, another comparison operation is performed that compares the re-generated sub-watermark $E(R(sw'_{f_{n,k}}))$ and the second queried sub-watermark $E(sw_{f_{n,k}})$, for provenance integrity check, as shown in Line 11 of Algorithm 2. If these two sub-watermarks are the same, provenance integrity is verified and the provenance record $p_{n,k}$ that contains provenance information is decrypted using the secret key K_j. The IP address, data capturing time and packet sequence number are obtained from $p_{n,k}$. Provenance is then validated and data is ready for processing. After that, the stored provenance record $p_{n,k}$ of received data packet $d'_{n,k}$ may be deleted from the database, after being used for security analysis. Whereas, if sub-watermarks were not the same, data $d'_{n,k}$ will be discarded and an attack procedure is performed (check the type of attack or origin of data is being altered). The stored provenance $p_{n,k}$ of the discarded data, after attack detection, will be deleted from the database.

Algorithm 2 : Watermark Querying and Verification

```
    input: $d^{'}_{n,k}$
    output: verified/not verified
 1: procedure Watermark querying and verification
 2:     $Receive(d^{'}_{n,k})$
 3:     $R(W^{'}_{n,k}) \leftarrow$ REDO Algorithm 1
 4:     $R(sw^{'}_{h_{n,k}}) \leftarrow EXTRACT(R(W^{'}_{n,k}))$
 5:     $W_{F_{n,k}} \leftarrow QRY(W_{F_{n,k}})$
 6:     $sw_{h_{n,k}} \leftarrow EXTRACT(W_{F_{n,k}})$
 7:     if $(R(sw^{'}_{h_{n,k}}) = sw_{h_{n,k}})$ then
 8:         Data Integrity Verified
 9:         $E(sw_{f_{n,k}}) \leftarrow EXTRACT(W_{F_{n,k}})$
10:         $E(R(sw^{'}_{f_{n,k}})) \leftarrow EXTRACT(R(W^{'}_{n,k}))$
11:         if $(E(R(sw^{'}_{f_{n,k}})) = E(sw_{f_{n,k}}))$ then
12:             Provenance Verified
13:             $p_{n,k} \leftarrow DEC(E(sw_{f_{n,k}}), K_j)$
14:             Extract IoT device $(n)$ IP address
15:             Check provenance information of $(d^{'}_{n,k})$
16:             Process data $(d^{'}_{n,k})$
17:         else
18:             Provenance Not Verified
19:             Discard data $(d^{'}_{n,k})$
20:             Perform attack procedure
21:             Delete $W_{F_{n,k}}$ from network database
22:         end if
23:     else
24:         Integrity Not Verified
25:         Discard data $(d^{'}_{n,k})$
26:         Perform attack procedure
27:         Delete $W_{F_{n,k}}$ from network database
28:     end if
```

4.5.2 Multi Hop Scenario

Three algorithms are proposed in this scenario: watermark generation and embedding, watermark verification and re-embedding, and data integrity verification and provenance reconstruction.

1. **Watermark Generation and Embedding:** Algorithm 3 describes the working process of two procedures: watermark generation and storing, and watermark embedding in the multi-hop scenario. The algorithm accepts the captured data $d_{n,k}$ as an input obtained from the source node that is sensing data from the surrounding environment. In the first procedure, three inputs are used for generating the first sub-watermark such as the IoT device IP address w_{ip}, the data sensing time w_t and the generated packet sequence number w_{sq_i} at node n. The sub-watermark $sw_{f_{n,k,i}}$ is formed by appending these values. To secure the provenance information, $sw_{f_{n,k,i}}$ is encrypted using secret key K_j. The encrypted value forms the provenance record $p_{n,k,i}$. Another sub-watermark $sw_{h_{n,k,i}}$ is generated from the hash value of the data payload. Finally, the final watermark $W_{F_{n,k,i}}$ is produced by concatenating the two sub-watermarks $sw_{f_{n,k,i}}$ and $sw_{h_{n,k,i}}$ as in Equation (4.2). Provenance record $p_{n,k,i}$ is then stored in the network database as shown in Line 8. In the second procedure, the watermarked data $d_{(n,k)W_{F_{n,k,i}}}$ is produced by concatenating the final watermark $W_{F_{n,k,i}}$ with the captured data packet $d_{n,k}$ as shown in Equation (4.3).

$$W_{F_{n,k,i}} = E(w_{ip} \,||\, w_t \,||\, w_{sq_i}, K_j) \,||\, H(d_{n,k}) = E(sw_{f_{n,k,i}}, K_j) \,||\, sw_{h_{n,k,i}} \tag{4.2}$$

$$d_{(n,k)W_{F_{n,k,i}}} = d_{n,k} \,||\, W_{F_{n,k,i}} \tag{4.3}$$

Algorithm 3 : Watermark Generation and Embedding

```
    input: $d_{n,k}$
    output: $d_{(n,k)W_{F_{n,k,i}}}$
 1: procedure WATERMARK GENERATION AND STORING
 2:     $w_{ip} \leftarrow$ IoT Device ($n$) IP Address
 3:     $w_t \leftarrow$ captured data.sensing time ($d_{n,k}$)
 4:     $w_{sq_i} \leftarrow seq$ of ($d_{n,k}$) at $n$
 5:     $sw_{f_{n,k,i}} \leftarrow w_{ip} \,||\, w_t \,||\, w_{sq_i}$
 6:     $p_{n,k,i} \leftarrow E(sw_{f_{n,k,i}}) \leftarrow ENC(sw_{f_{n,k,i}}, K_j)$
 7:     $sw_{h_{n,k,i}} \leftarrow H(d_{n,k})$          ▷ Select first 8 bytes of hash output
 8:     $W_{F_{n,k,i}} \leftarrow p_{n,k,i} \,||\, sw_{h_{n,k,i}}$
 9:     $STR(p_{n,k,i})$
10: procedure WATERMARK EMBEDDING
11:     $d_{(n,k)W_{F_{n,k,i}}} \leftarrow d_{n,k} \,||\, W_{F_{n,k,i}}$
12:     $Send(d_{(n,k)W_{F_{n,k,i}}})$
```

2. **Watermark verification and re-embedding:** At the next hop, a watermark verification and re-embedding algorithm is applied as shown in Algorithm 4. To verify data integrity at the next node, the algorithm accepts the watermarked data $d'_{(n,k)W_{F_{n,k,i}}}$ as an input. The captured data $d'_{n,k}$ and watermark $W'_{F_{n,k,i}}$ are extracted from $d'_{(n,k)W_{F_{n,k,i}}}$. A new sub-watermark $R(sw_{h_{n,k,i}})$ is re-generated from $d'_{n,k}$ by using the first procedure of Algorithm 3 and $sw'_{h_{n,k,i}}$ is extracted from $W'_{F_{n,k,i}}$. Then a comparison operation is applied on the sub-watermark

values of $R(sw_{h_{n,k,i}})$ and $sw'_{h_{n,k,i}}$ to check whether data is altered or not. If data integrity is verified, $E(sw_{f_{n,k,i}})$ is obtained from querying the provenance record $p_{n,k,i}$ from the network database. Another sub-watermark $E(sw'_{f_{n,k,i}})$ is extracted from $W'_{F_{n,k,i}}$ for provenance validation. Then, a comparison operation is applied on $E(sw_{f_{n,k,i}})$ and $E(sw'_{f_{n,k,i}})$. If both sub-watermarks are the same, provenance integrity is verified and a new watermark is generated using the same procedure of Algorithm 3 as shown in Lines 12-19. The new generated watermark $W_{F_{n,k,i}}$ is formed of the next hop node IP address, the watermarked data packet receiving time and a new generated packet sequence number w_{sq_i} of the next hop node, and the same hash value of the data packet obtained from the re-generated sub-watermark $R(sw_{h_{n,k,i}})$ using Equation (4.2). The watermark $W_{F_{n,k,i}}$ is concatenated with data $d'_{n,k}$ to form a watermarked data packet as shown using Equation (4.3). Then, the new generated sub-watermark $E(sw_{f_{n,k,i}})$ or provenance record $p_{n,k,i}$ is stored in the network database as shown in Line 20. However, if $E(sw_{f_{n,k,i}})$ and $E(sw'_{f_{n,k,i}})$ are not the same, the provenance is not verified and the data is discarded. Also, $P_{n,k}$ of received watermarked data packet $d'_{(n,k)W_{F_{n,k,i}}}$ is deleted from the database and an attack procedure is applied. If data integrity is not verified, data will be also discarded and an attack procedure will be applied. Also, all stored provenance records of $P_{n,k}$ related to this data packet will be deleted from the database.

3. **Integrity verification and provenance reconstruction:** The process of verifying data integrity and reconstructing provenance at the gateway is described in Algorithm 5. The verification procedure relies on five main conditions:

   (a) The origin of data packet based on the source IP address.

   (b) The freshness of the timestamp w_t included in the watermark.

   (c) The provenance record sequence number integrity.

   (d) The hop by hop integrity and provenance validation.

   (e) Verifying the data measurement using the hash value.

   The received watermarked data $d''_{(n,k)W_{F_{n,k,i}}}$ is extracted into $d''_{n,k}$ and $W''_{F_{n,k,i}}$. The gateway re-generates the sub-watermark $R(sw_{h_{n,k,i}})$ by performing the generation process of Algorithm 3 as shown in Line 4 Algorithm 5 and $sw''_{h_{n,k,i}}$ is extracted from $W''_{F_{n,k,i}}$. The extracted sub-watermark $sw''_{h_{n,k,i}}$ and the re-generated sub-watermark $R(sw_{h_{n,k,i}})$ will be compared using a comparison operation to check data integrity. If data is not altered, the gateway queries the last provenance record $p_{n,k,i}$ of the received watermarked data packet $d''_{(n,k)W_{F_{n,k,i}}}$ from the database. Then, $E(sw''_{f_{n,k,i}})$ is extracted from $W''_{F_{n,k,i}}$. The gateway performs a comparison operation for $E(sw''_{f_{n,k,i}})$ and $E(sw_{f_{n,k,i}})$ (i.e. last stored provenance record). If both values are the same, provenance is verified, the gateway queries the set of stored provenance records $P_{n,k}$ from the database and extracts the encrypted sub-watermarks $E(sw_{f_{n,k,i}})$ of each $p_{n,k,i}$. At Line 15, the secret key K_j is used to decrypt $E(sw_{f_{n,k,i}})$ and obtain the sub-watermarks $sw_{f_{n,k,i}}$ containing provenance information of the received data packet. The gateway constructs the data path from provenance information obtained and uses packet sequence record to identify any provenance record drop attack or any modification in the packet forwarding path. If data integrity or provenance is not verified, data will be discarded and an attack procedure is performed and $P_{n,k}$ of received watermarked data packet $d''_{(n,k)W_{F_{n,k,i}}}$ is deleted from the database.

Algorithm 4 : Watermark Verification and Re-embedding

```
   input: $d'_{(n,k)W_{F_{n,k,i}}}$
   output: verified/not verified, $W_{F_{n,k,i}}$, $d'_{(n,k)W_{F_{n,k,i}}}$
1: procedure WATERMARK VERIFICATION AND RE-EMBEDDING
2:     $Receive(d'_{(n,k)W_{F_{n,k,i}}})$
3:     Extract Watermarked Data into $d'_{n,k}$ and $W'_{F_{n,k,i}}$
4:     $R(sw_{h_{n,k,i}}) \leftarrow$ REDO Algorithm 1
5:     $sw'_{h_{n,k,i}} \leftarrow EXTRACT(W'_{F_{n,k,i}})$
6:     if $(R(sw_{h_{n,k,i}}) = sw'_{h_{n,k,i}})$ then
7:         Integrity Verified
8:         $E(sw_{f_{n,k,i}}) \leftarrow QRY(p_{n,k,i})$
9:         $E(sw'_{f_{n,k,i}}) \leftarrow EXTRACT(W'_{F_{n,k,i}})$
10:        if $(E(sw'_{f_{n,k,i}}) = E(sw_{f_{n,k,i}}))$ then
11:            Provenance Integrity Verified
12:            Generate next hop watermark{
13:                $w_{ip} \leftarrow$ next hop device IP Address
14:                $w_t \leftarrow$ received time $(d'_{(n,k)W_{F_{n,k,i}}})$
15:                $w_{sq_i} \leftarrow (seq(d'_{(n,k)}))$                ▷ new sequence number
16:                $i++$                              ▷ update next hop watermark index
17:                $sw_{f_{n,k,i}} \leftarrow w_{ip} \,||\, w_t \,||\, w_{sq_i}$
18:                $p_{n,k,i} \leftarrow E(sw_{f_{n,k,i}}) \leftarrow ENC(sw_{f_{n,k,i}}, K_j)$
19:                $sw_{h_{n,k,i}} \leftarrow$ Hash value from $R(sw_{h_{n,k,i}})$
20:                $W_{F_{n,k,i}} \leftarrow E(sw_{f_{n,k,i}}) \,||\, sw_{h_{n,k,i}}$}
21:            $d'_{(n,k)W_{F_{n,k,i}}} \leftarrow d'_{n,k} \,||\, W_{F_{n,k,i}}$
22:            $STR(p_{n,k,i})$
23:            $Send(d'_{(n,k)W_{F_{n,k,i}}})$
24:        else
25:            Provenance no verified/attack detection
26:            Discard data $d'_{n,k}$ & Perform attack procedure
27:            Delete $P_{n,k}$ from network database
28:        end if
29:    else
30:        Not verified/attack detection
31:        Discard data $d'_{n,k}$
32:        Perform attack procedure
33:        Delete $P_{n,k}$ from network database
34:    end if
```

Algorithm 5 : Watermark Restoring and Verification

```
    input: $d''_{(n,k)W_{F_{n,k,i}}}$
    output: verified/not verified, provenance reconstruction
 1: procedure Watermark Restoring and Verification
 2:     $Receive(d''_{(n,k)W_{F_{n,k,i}}})$
 3:     Extract Watermarked Data into $d''_{n,k}$ and $W''_{F_{n,k,i}}$
 4:     $R(sw_{h_{n,k,i}}) \leftarrow$ REDO Algorithm 3
 5:     $sw''_{h_{n,k,i}} \leftarrow EXTRACT(W''_{F_{n,k,i}})$
 6:     if $(R(sw_{h_{n,k,i}}) = sw''_{h_{n,k,i}})$ then
 7:         Data integrity verified
 8:         $E(sw_{f_{n,k,i}}) \leftarrow QRY(p_{n,k,i})$
 9:         $E(sw''_{f_{n,k,i}}) \leftarrow EXTRACT(W''_{F_{n,k,i}})$
10:         if $(E(sw_{f_{n,k,i}}) = E(sw''_{f_{n,k,i}}))$ then
11:             Provenance integrity verified
12:             $P_{n,k} \leftarrow QRY(P_{n,k})$
13:             for (index $i$, $1 \le i \le H$, $i++$) do
14:                 Extract $E(sw_{f_{n,k,i}})$ of each $p_{n,k,i}$ from $P_{n,k}$
15:                 $sw_{f_{n,k,i}} = DEC(E(sw_{f_{n,k,i}}), K_j)$
16:                 Extract provenance information
17:             end for
18:             Construct data path of $d''_{n,k}$
19:         else
20:             Provenance integrity is not verified
21:             Attack detection
22:             Discard data $d''_{n,k}$
23:             Perform attack procedure
24:             Delete $P_{n,k}$ from network database
25:         end if
26:     else
27:         Data integrity not verified/ attack detected
28:         Discard data $d''_{n,k}$
29:         Perform attack procedure
30:         Delete $P_{n,k}$ from network database
31:     end if
```

4.6 Managing Internal Datagrams

In this section, we propose the idea of labeling IP datagrams that are used internally for network management. These datagrams should not be analyzed by the IDS and will undergo an internal security procedure. This optimizes the scheme by minimizing the number of IDS operations on data packets. The advantage of this protocol is the use of the Identification field, flags and fragment offset as the embedding positions in the IP datagram header which will appear random-like and will not show an evident pattern that an attacker may try to exploit (cf. Section 4.4). The management of IP datagrams by network nodes is formally described in Algorithms 6 and 7.

Algorithm 6 : Internal Managing at the Source Node

```
    input: IP datagram $d_{IP}$
    output: Embedding hash value
 1: procedure INTERNAL MANAGING EMBEDDING PROCESS
 2:     if($d_{IP}$ = internal managing packet) then
 3:         Compute H(Destination IP || First 20 bytes of $d_{IP}$)
 4:         $d_{IP}$(header) ← H(Dest. IP || First 20 bytes of $d_{IP}$)
 5:     else
 6:         perform watermark generation and embedding
 7:     end if
```

Algorithm 7 : Internal Managing at the Destination Node

```
    input: IP datagram $d_{IP}$
    output: Require IDS/internal-managing
 1: procedure INTERNAL MANAGING PROCESS
 2:     Receive ($d_{IP}$)
 3:     if(IP datagram ← ($src$, $dest$)) then
 4:         if(($src$, $dest$ = internal) & L($R_D$) = L($D$)) then
 5:             Compute H(Destination IP || First 20 bytes of $d_{IP}$)
 6:             Extracted H ← $d_{IP}$(Identification+Flags+Offset)
 7:             if(Computed H = Extracted H) then
 8:                 $d_{IP}$ is authenticated, i.e.,
 9:                 $d_{IP}$ is not examined by the IDS
10:             else
11:                 attack detection
12:                 $d_{IP}$ is discarded
13:             end if
14:         else
15:             $d_{IP}$ must be examined by the IDS
16:         end if
17:     else
18:         $d_{IP}$ must be examined by the IDS
19:     end if
```

At each node or gateway, internal managing packets are labeled with a hash value that is computed and embedded before the packet is sent. The hash value is computed as H(Destination IP || First 20 bytes of the datagram content), where the operator

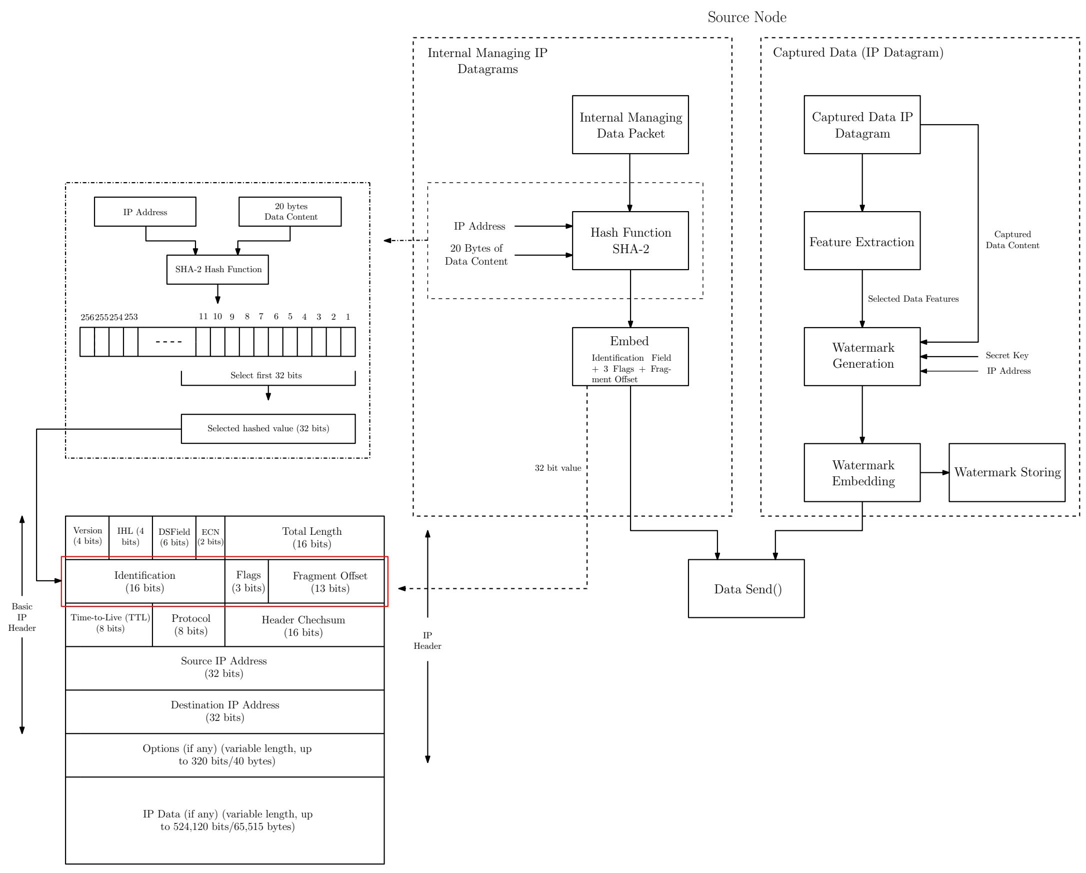


FIGURE 4.6: Embedding hash value in the Internal Managing Protocol. The red square represents the embedding position of the selected bit from the generated hash function.

|| denotes concatenation. The value is then embedded in the IP datagram header as shown in Algorithm 6 and Figure 4.6. We use the Identification field (16 bits), Flags(3 bits) and Fragment offset(13 bits) to embed the selected 32 bit from the hash value. After receiving any IP Datagram at the Gateway or any node in the internal network, an internal managing protocol is performed (before any IDS procedure) as shown in Algorithm 7. The datagram is subjected to a first condition that checks whether these datagrams have both a source and a destination address in our local network, since this is a first condition (filter). Then it checks if both the source and the destination address are internal and the size of the received data packet is equal to an internal managing packet size. Sensed data packets by sensor nodes are watermarked and have different size (data packet + watermark) as shown using Equation (4.3). If it is not the case, the datagram must be examined by the IDS (it is not an internal data packet). However, if both IP addresses are internal and the size is confirmed, the device computes H(Destination IP address || First 20 bytes of the datagram content) and extracts the hash value embedded in the header of the data packet. Then the node compares these two hash values. If these values are the same, the datagram is authenticated as "authorized internal-managing packet". Otherwise, an attack is detected and the datagram is discarded.

The hash function used in obtaining the IP datagram label is SHA-2. SHA-2 takes an input of any size and produces a 256-bit hash value. Since the Identification field is 16 bit long and the size of Flags and offset take another 16 bits, making a total of 32 bits, the device selects 32 LSB bits from the hash value as shown in Figure 4.6. We can also randomize the selection of these 32 bits by using pseudo-random number generator and obtain randomized bit positions that can be selected. This randomization would add another level of security for the system.

4.7 Summary

In this chapter, we have focused on designing a zero-watermarking approach to ensure data integrity and securely transmit provenance information in IoT networks. We consider an adversary that applies different types of attacks, as stated in the threat model, within the network and is able to acquire knowledge about the network dynamics prior to starting the attack.

To implement the approach, we have used two case scenarios of single- and multi-hop IoT networks, consisting of source nodes, intermediate nodes, and a gateway. The new design introduces an internal database for storing zero-watermarks that represent provenance records. These records are generated at source and intermediate nodes in each hop and are verified at each step in the data path from source to destination. We also propose a protocol to manage internal datagrams to reduce the overhead of the generation, embedding, and verification process on all data packets in the network.

Chapter 5

Security Analysis and Numerical Simulation

5.1 Introduction

The previous chapter explores the proposed model of zero-watermarking for securing data integrity and provenance information within IoT networks. Zero-watermarking offers a unique approach to embed essential data features and provenance details directly within data packets. These watermarks are then stored securely within a tamper-proof network database. This chapter delves into the security robustness of our proposed scheme by employing formal security analysis against various attack types commonly encountered in IoT networks. We will present a series of theorems accompanied by their proofs to demonstrate ZIRCON's resilience against these attacks.

To further validate our findings, the chapter utilizes representative simulations conducted within MATLAB™. These simulations allows for a comparative analysis of ZIRCON's performance against existing schemes. The comparison focuses on key parameters such as computational efficiency, energy consumption, and the additional data overhead introduced by the provenance information. Finally, we explore various cryptographic and hashing techniques for watermark generation through experimentation. This exploration aims to identify the optimal technique that offers a balance between security and efficiency within the ZIRCON framework.

5.2 Security Analysis

The IoT network can be subject to two main security breaches in the transmission phase: passive and active attacks on both data and watermark. An adversary can launch various attacks based on the threat model described in Section 4.4. In this section, we provide an analysis for the security of the proposed scheme against the attacks detailed in Section 4.4. We assume that the network gateway and database are trusted and cannot be compromised by an attacker.

Theorem 1. *An unauthorized party cannot access or obtain the secret information generated by the source node S_n.*

Proof. The source node S_n generates a final watermark $W_{F_{n,k,i}}$ by concatenating two sub-watermarks $sw_{f_{n,k,i}}$ and $sw_{h_{n,k,i}}$. The first sub-watermark $sw_{f_{n,k,i}}$ is obtained from extracted data features as follows: IP address w_{ip}, sensed data capturing time w_t and data packet sequence number w_{sq}. These data features are encrypted using Advanced Encryption Standard (AES) algorithm using a symmetric secret key K_j. $sw_{f_{n,k,i}}$ is generated and encrypted as $E(sw_{f_{n,k,i}}) = ENC(sw_{f_{n,k,i}}, K_j)$. Thus, an attacker being unaware of K_j cannot decrypt $sw_{f_{n,k,i}}$ (only authorized parties are aware of K_j, i. e., intermediate nodes and gateway). Note that K_j is changed and redistributed after a short random number of watermark generation sessions (see Section 4.4). For the second sub-watermark $sw_{h_{n,k,i}}$, a source node uses a one-way cryptographic hash function $H()$ to obtain $sw_{h_{n,k,i}}$ used for data integrity check. It is computationally infeasible to to find a pair (x, y) such that $h(x) = h(y)$, which make the function secure and cannot be inverted as assumed in Section 4.4. Additionally, we use SHA-2 hash function in our scheme with 256 bit hash value, which make it computationally infeasible for an attacker to carry out 2^{128} calculations to find the second sub-watermark. The generated watermark is computed as $W_{F_{n,k,i}} = E(sw_{f_{n,k,i}}) \,||\, sw_{h_{n,k,i}}$ for each captured data. Thus, an adversary cannot access watermark information generated by source nodes. □

Theorem 2. *An attacker, cannot successfully deceive an intermediate node I_l or gateway G by inserting fake data or deleting data from the data flow generated by a legitimate node S_n and transmitted to I_l or G.*

Proof. In case an attacker inserts fake data into a watermarked data-packet $d_{(n,k)W_{F_{n,k,i}}}$ being transmitted to I_l or G, the destination node extracts $d_{(n,k)W_{F_{n,k,i}}}$ into sensed data $d_{n,k}$ and watermark $W_{F_{n,k,i}}$. Then, a re-generated sub-watermark $R(sw_{h_{n,k,i}})$ is computed from the received captured data $d_{n,k}$ and compared to the extracted watermark $sw'_{h_{n,k,i}}$ from $W_{F_{n,k,i}}$. The process of re-generation is based on the previously mentioned generation process (i. e., SHA-2 hash function for $sw_{h_{n,k,i}}$). Hence, any change in $d_{n,k}$ content produces an altered re-generated sub-watermark. The assumption of secure communication of extracted data features and provenance information using symmetric cryptography and a one-way hash function applies (Section 4.4). Then, even if an attacker inserts fake data into $W_{F_{n,k,i}}$ without altering $d_{n,k}$, $R(sw_{h_{n,k,i}})$ will not match $sw'_{h_{n,k,i}}$ in the comparison process. Also, if the attacker inserts fake data to the second sub-watermark $E(sw'_{f_{n,k,i}})$ the next hop intermediate node or gateway queries the stored provenance record $p_{n,k,i} = E(sw_{f_{n,k,i}})$ from the data base and compares it with the extracted sub-watermark $E(sw'_{f_{n,k,i}})$. Any change in $E(sw'_{f_{n,k,i}})$ yields to alternation in the provenance information.

In the second case, the attacker aims to delete data content from $d_{n,k}$ or $W_{F_{n,k,i}}$, or drop an entire data-packet $d_{(n,k)W_{F_{n,k,i}}}$ being routed from S_n to I_l or from I_l to G. The deletion of q bits from $d_{n,k}$ results in the modification of $R(sw_{h_{n,k,i}})$ and thus $sw'_{h_{n,k,i}}$ will not match $R(sw_{h_{n,k,i}})$. Again, the previously mentioned assumption of secure communication of $W_{F_{n,k,i}}$ applies. Furthermore, if the attacker deletes q bits from $W_{F_{n,k,i}}$ it will be detected in the comparison process of the two sub-watermarks $R(sw_{h_{n,k,i}})$ and $sw'_{h_{n,k,i}}$ or between the queried sub-watermark $E(sw_{f_{n,k,i}})$ and the extracted one $E(sw'_{f_{n,k,i}})$. Obviously, such an adversary may drop $d_{(n,k)W_{F_{n,k,i}}}$ routed through I_l. This attack can be detected at G by accessing the tamper-proof database and querying the provenance records of $d_{n,k}$, and detecting where the packet drop attack occurred. The database stores provenance records securely, which cannot be accessed by an attacker (as described in Section 4.4). □

Theorem 3. *An attacker, attempting to alter provenance information: (i) cannot add legitimate nodes to the provenance of data generated by an unauthorized node, (ii) cannot successfully add or remove nodes from the provenance of data generated by legitimate nodes.*

Proof. I_l stores a provenance record $W_{F_{n,k,i}}$ after checking data integrity and provenance of the received data-packet $d_{(n,k)W_{F_{n,k,i}}}$. The symmetric secret key K_j shared between legitimate nodes is used to obtain the generated watermark $W_{F_{n,k,i}}$ used in data integrity and provenance validation. An unauthorized node generates watermarks using its own secret key that cannot match a generated watermark at I_l using K_j. As stated in Section 4.4, the source node, the intermediate node and the gateway share secret-keys to be used in different steps of the algorithms (encryption/decryption). These keys are changed and redistributed between legitimate nodes after a random number of sessions. Thus, in order to add a legitimate node, an attacker needs to obtain the same symmetric secret key that is only shared within legitimate network nodes of internal registered IP addresses. In the case of two malicious nodes I_m and I_v attempting to execute an attack, a captured data-packet $d_{n,k}$ by a legitimate source node S_n is routed through S_m. $d_{n,k}$ has a provenance record of (I_1, I_2, I_4). The malicious node I_m aims to remove I_2 and replace it with I_v. To add I_v as a provenance record to the database, the malicious node needs to compute the next-hop watermark which requires, as mentioned above, the knowledge of K_j and hash function variables. Hence, the provenance integrity check at the next I_j will fail and an attack is detected. Thus, I_m will fail to add or remove any provenance record from network database. Moreover, provenance records ($W_{F_{n,k,1}}$, $W_{F_{n,k,2}}$, ..., $W_{F_{n,k,i}}$) of a data-packet $d_{n,k}$ are stored in a tamper-proof database that is assumed to be resistant to any alternation of its entities, attackers cannot alter any record stored in it (see Section 4.4). □

Theorem 4. *It is impossible for an attacker, whether acting alone or in collaboration with others, to add or authenticate nodes to the provenance of data produced by a compromised node.*

Proof. An attacker may generate fake data and store provenance information in the database as a legitimate node with its secret key. The packet is then forwarded to the next hop intermediate node to store the next hop provenance information in the set of provenance records $P_{n,k}$ for this data packet in the database. The attacker's aim is to construct the provenance from innocent forwarding nodes and make them responsible for false data forwarding, thus marking them as untrustworthy nodes. However, there is an integrity and provenance validation procedure at the next hop node, which includes a watermark re-generation process $W_{F_{(n,k,i)}}$ using the secret key K_j, the attacker do not know the key for legitimate nodes. Thus, this attack will fail at the first hop. □

Theorem 5. *Any unauthorized attempt to modify data content through transmission channel would be detected.*

Proof. An adversary may perform a modification to the embedded watermark (computed as $W_{F_{n,k,i}} = E(sw_{f_{n,k,i}}) \mid\mid sw_{h_{n,k,i}})$ or data elements $d_{(n,k)W_{F_{n,k,i}}}$. If data elements are modified and $W_{F_{n,k,i}}$ remains unchanged, a different watermark is obtained based on a wrong hash value at an intermediate node or gateway. Since the first generated sub-watermark at source node $sw_{h_{n,k,i}}$ is the output of a hash function SHA-2 obtained as $sw_{h_{n,k,i}} = H(d_{n,k})$. Again, the assumption of hash functions used in the system (Section 4.4) applies. The wrong re-generated sub-watermark $R(sw_{h_{n,k,i}})$ will not match the extracted sub-watermark $sw'_{h_{n,k,i}}$. The intermediate node or gateway detects the modification attack and discards the data. Furthermore, if the attacker modifies $W_{F_{n,k,i}}$ and the data payload remains unchanged, the intermediate node or gateway re-generates the right sub-watermark, extract the modified watermark from the received data packet and queries the provenance record $p_{n,k,i}$ from the database. This results in a failed comparison operation for data integrity or for provenance validation and data will be discarded. □

Theorem 6. *By including a timestamp in the generation process of watermarks, any fraud transmission of previously captured data packets will be discovered.*

Proof. An attacker may provide a false idea about the sensing environment by fraudulently transmitting previously heard data packets that are captured and transmitted by a legitimate source node [227]. The attacker also detects the timing characteristics to be used later during the packet replay attack. To deceive an intermediate node or gateway, the attacker updates the timestamp w_t of the heard data packet $d_{n,k}$, based on timing characteristics, to a new recent time value. In the proposed scheme, a source node generates a watermark $W_{F_{n,k,i}}$ for each data packet captured ($d_{n,k}$). The generation process is based on provenance information, a timestamp and a hash value as described in Equation (4.1). Provenance information and timestamp will be encrypted using a secret key K_j to form the first sub-watermark (i. e., encrypting $sw_{f_{n,k,i}} = w_{ip} \mid\mid w_t$ as $E(sw_{f_{n,k,i}}) = ENC(sw_{f_{n,k,i}}, K_j))$. At next hop I_l or G, a new sub-watermark is generated from the replayed packet that will be compared to the extracted sub-watermark. If the attacker changed the timestamp of the data packet $d_{n,k}$ the comparison operation will fail. Since timestamps are different the new re-generated sub-watermark will not match the extracted one. Note that the attacker cannot modify the timestamp w_t embedded in the watermark $W_{F_{n,k,i}}$, due to the encryption process performed on the generated sub-watermark $sw_{f_{n,k,i}}$. The sub-watermark is encrypted using the source secret key K_j, which is only shared with legitimate entities (intermediate node and gateway) where an attacker uses a different secret key as stated in Section 4.4. Hence, replaying an old packet with an updated timestamp will lead to a failed authentication procedure. □

Theorem 7. *Any attempt from an attacker to selectively drop a provenance record from the database or alter the provenance will be detected at the base station.*

Proof. If an attacker manages to compromise the database and selectively remove a provenance record $p_{n,k,i}$ from a data packet's provenance $P_{n,k}$, the base station will query the complete provenance information from the database. This query occurs after a final integrity validation of both the data payload and provenance. After decrypting the retrieved sub-watermarks of $P_{n,k}$, the base station extracts the provenance information and checks the IP address w_{ip}, timestamp w_t, and packet sequence number w_{sq_i} of each provenance record. This information is used to construct the data path. If any provenance records are missing or have out-of-order sequence numbers in the stored forwarding provenance records, they will be detected. Consequently, the base station is able to identify any provenance record dropping attack. This method is also applicable in the single-hop scenario, where the base station detects packet drop attacks using the sequence number w_{sq} stored in the provenance record of each data packet in the network database as shown in Algorithm 1. □

Theorem 8. *In Algorithm 7, an attacker trying to deceive network devices to accept malicious datagrams as trusted internal managing datagrams will be detected and examined by implemented security algorithms.*

Proof. If an attacker succeeds to modify an internal managing datagram, the datagram will be forwarded to the next hop node. At the receiving node, a hash value is computed from the content of the datagram using a one way hash function as detailed in Section 4.6. It also extracts the hash value embedded by the source node from the identification field of the IP header. Both hash values are then compared to detect any attempt of forgery attack. If the values do not match, the device applies the implemented security algorithms to the received IP datagram and an attack procedure is performed. Note that a source node uses a one-way cryptographic hash function $H()$ using SHA-2 to obtain the hash value (embedded in internal managing IP datagram's header) so that it is computationally infeasible to find a pair (x, y) such that $h(x) = h(y)$, making it impossible for an attacker to invert the hash value and embed it to deceive the system (Section 4.4). Hence, a malicious entity trying to deceive the forwarding nodes using internal managing datagrams will be detected and discarded. □

Theorem 9. *The proposed scheme demonstrates robustness against DoS attacks.*

Proof. A DoS attack on an IoT network is a malicious attempt to disrupt the normal functioning of the network by overwhelming it with a flood of illegitimate requests or traffic. This type of attack can render IoT devices or services unavailable to legitimate users by exhausting the network's resources, such as bandwidth, processing power, or memory. The characteristics and impact of DoS attacks on IoT networks are as follows:

- *Resource Limitation:* IoT devices typically have limited computational resources, memory, and bandwidth. This makes them particularly vulnerable to DoS attacks as they can be easily overwhelmed by a relatively low volume of malicious traffic compared to traditional network devices.
- *Diverse and Distributed Nature:* IoT networks often consist of a vast number of heterogeneous devices distributed across various locations, making it challenging to secure the entire network effectively and to identify and mitigate attacks promptly.
- *Critical Applications:* Many IoT applications, such as smart grids, healthcare monitoring systems, and industrial control systems, are critical and require high availability and reliability. A DoS attack on such networks can lead to significant disruptions

DoS attacks on IoT networks can be launched using various methods:

1. *Flooding Attacks:* Attackers send an overwhelming amount of traffic to the target device or network, consuming its bandwidth and processing capacity. Flooding attacks include:
   - *HTTP Flooding:* Overloading the device with HTTP requests.
   - *UDP Flooding:* Sending a large number of User Datagram Protocol (UDP) packets.
   - *TCP SYN Flooding:* Exploiting the TCP handshake process by sending numerous SYN requests without completing the handshake.
2. *Exploitation of Vulnerabilities:* Attackers exploit specific vulnerabilities in the IoT devices' firmware or software to cause them to crash or become unresponsive. This exploitation includes:

- *Buffer Overflow:* Sending specially crafted packets that overflow the buffer memory of the device, causing it to crash.
- *Firmware Exploits:* Targeting known vulnerabilities in the device firmware to disrupt its operation.

ZIRCON can mitigate DoS attacks on IoT networks by ensuring the authenticity and integrity of the data packets at each hop and gateway. By embedding the watermark $W_{F_{n,k,i}}$, the system can verify the source and legitimacy of each packet, discarding any that fails the verification process as described in Algorithms 4 and 5. This prevents malicious packets from overwhelming the network, as only authenticated traffic is allowed to pass through. The continuous verification at each hop and the gateway helps in early detection and filtering of illegitimate traffic, thus protecting the network from being flooded with malicious data such as HTTP, UDP and TCP SYN flooding attacks. Since the approach verifies the authenticity of the source at each hop, it prevents attackers from easily injecting illegitimate traffic into the network. Attackers would need access to encryption keys and provenance information used in watermark generation and verification process, which is significantly more challenging. Each intermediate node I_l can verify if the packet has traversed legitimate source nodes S_n, ensuring that the packet's data path or provenance $P_{n,k}$ through the network is as expected and not deviated from normal behavior. Any deviation can trigger an alert or the dropping of the packet $d_{(n)}$, preventing malformed or spoofed packets from consuming network resources. Hence, ZIRCON verification process at each hop and at the gateway filters out packets without valid watermarks, thereby reducing the bandwidth and processing load on IoT devices and preventing them from being overwhelmed by illegitimate traffic. □

Theorem 10. *An attacker aiming at the interception of data communication between IoT devices using MITM attacks can be detected.*

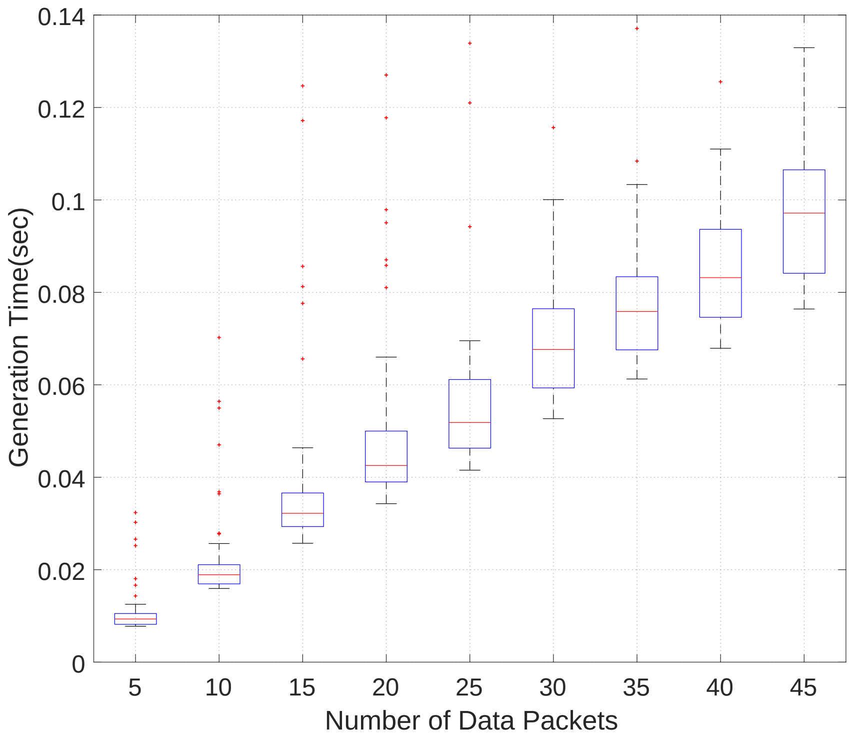
Proof. Many IoT devices have weak security features and lack the processing power to implement complex encryption protocols. A MITM attack is a serious threat to these devices on an IoT network. In this attack, an attacker secretly inserts themselves into the communication between two devices. This allows them to eavesdrop on the conversation, steal sensitive data, or even modify the data being exchanged. The attacker intercepts communication between the IoT devices and the legitimate destination (maybe I_l or G). This can be done through various methods like ARP spoofing or setting up a fake WiFi network. Once in the middle, the attacker can listen to the data flowing between the devices. This could include sensitive information such as sensor data or control commands (internal packets). The attacker can also modify the data before it reaches its destination, potentially causing malfunctions or disrupting operations. Also, the attacker can modify the intercepted packets to inject false data, alter commands, or introduce malicious payloads into the communication data stream. In our scheme, by embedding watermarks into data packets using provenance information, ZIRCON ensures that any alteration of the watermarked data packets $d_{(n,k)W_{F_{n,k,i}}}$ will either modify the watermark $W_{F_{n,k,i}}$ or alter the packet payload. At the next hop or gateway, the data packet undergoes verification. If a packet's watermark is invalid or altered, it is identified as compromised and the packet is discarded. Hence, the process of verifying the watermark at each hop means that even if an attacker intercepts and modifies a packet between two nodes, the modification will be detected at the next node, preventing the altered packet from proceeding further. Moreover, provenance information helps ensure that the packet is from a legitimate source node S_n. A MITM attacker cannot easily forge provenance information because it is cryptographically bound to the packet by a secret key K_j. The use of encryption keys to embed watermarks means that an attacker would need access to these keys to generate or modify a valid watermarks. Without the key, any attempt to alter the packet will result in an invalid watermarks, making it easier to detect and discard tampered packets. $\square$

Based on the above analysis, the proposed scheme is proven to be resistant against various malicious attacks of IoT networks, such as modification attack, integrity attack, packet replay, database authentication attack and passive attacks. It guarantees the integrity of data and ensures security against identifying and retrieving provenance information in IoT networks.

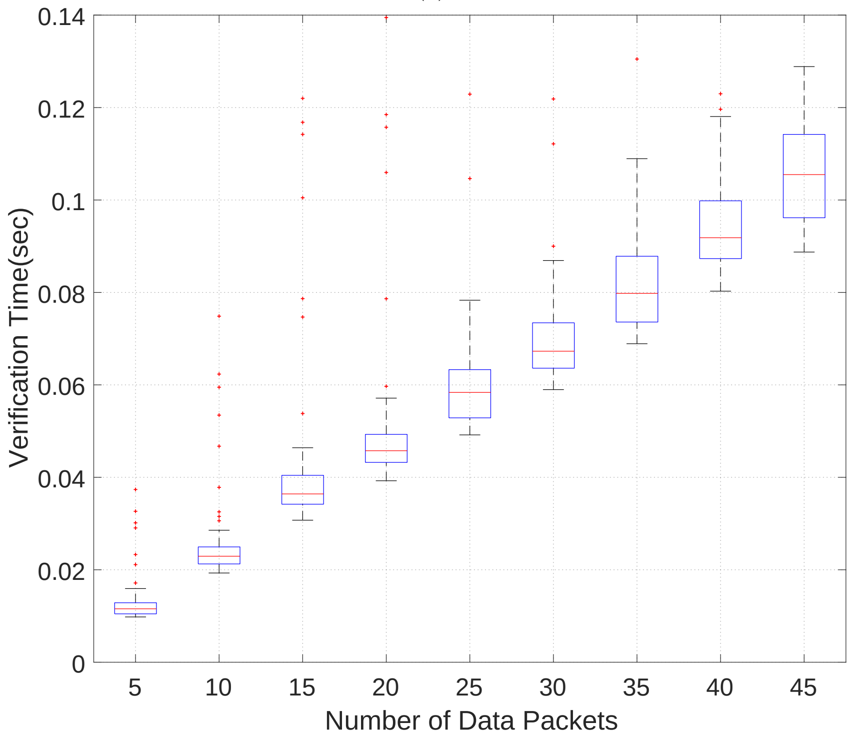
5.3 Simulation Results and Analysis

In this section, we evaluate the performance of the proposed scheme based on two features: data integrity and data provenance. For data integrity, the proposed scheme is evaluated based on watermark generation, embedding and verification time. Also, we have measured how this scheme performs in terms of energy utilization. The results are then compared to three state of the art techniques: RWFS [100], Asymmetric Cryptography Technique (ACT) [98] and Zero-Watermarking Scheme (ZWT) [48] based methods. We selected these three state-of-the-art methods to assess the performance of our new security technique based on their use of different security techniques deployed in a similar network model. For data provenance, we compare our scheme with MAC-based provenance scheme (MP), a secure provenance framework *SProv* [67], and a lightweight secure scheme BFP [33] in terms of cost analysis. The algorithms were implemented in MATLAB™ on Intel core i7 processor with a 2.59 GHz clock cycle and 16 GB of memory. Sensor data is represented as an integer data type, since most sensor readings are of numeric form such as temperature, humidity, motion and intensity.

In our algorithm, we use AES with 128 bit key size for encryption of generated watermarks. Despite the fact that AES has a larger key size than Data Encryption Standard (DES), AES is a more secure and advanced encryption algorithm compared to DES, which makes it more resistant to cryptanalysis attacks. Another reason for using AES is its performance and efficiency. AES is a fast and efficient algorithm. We provide, in Figures 5.1a, 5.1b, 5.2a and 5.2b, a comparison of using AES and DES algorithms in the generation and verification processes at each sensor node in the proposed model. The results demonstrate superior performance of our scheme when applying the AES algorithm, achieving approximately 10 times faster speeds in both watermark generation and verification. The use of substitution-permutation network (SPN) structure, which is optimized for hardware implementation and allows for parallel processing in AES shows a faster performance than DES, which uses a Feistel network structure. Regarding the hash function, we use a one way hash function SHA-2, specifically SHA-256, for generating the second sub-watermark sw_h. Although SHA-1 is faster than SHA-2 functions since it uses a smaller block size and has a simpler construction, however it is important to note that the slower performance of SHA-2 functions is outweighed by their improved security compared to SHA-1. We compared the generation and verification time of the proposed model using different hash functions in Figures 5.3a and 5.3b. The results shows that SHA-1 is faster than SHA-2 functions and SHA-2(256) function requires less processing time than SHA-2(384) and SHA-2(512). Hence, we use SHA-2(256), which provides

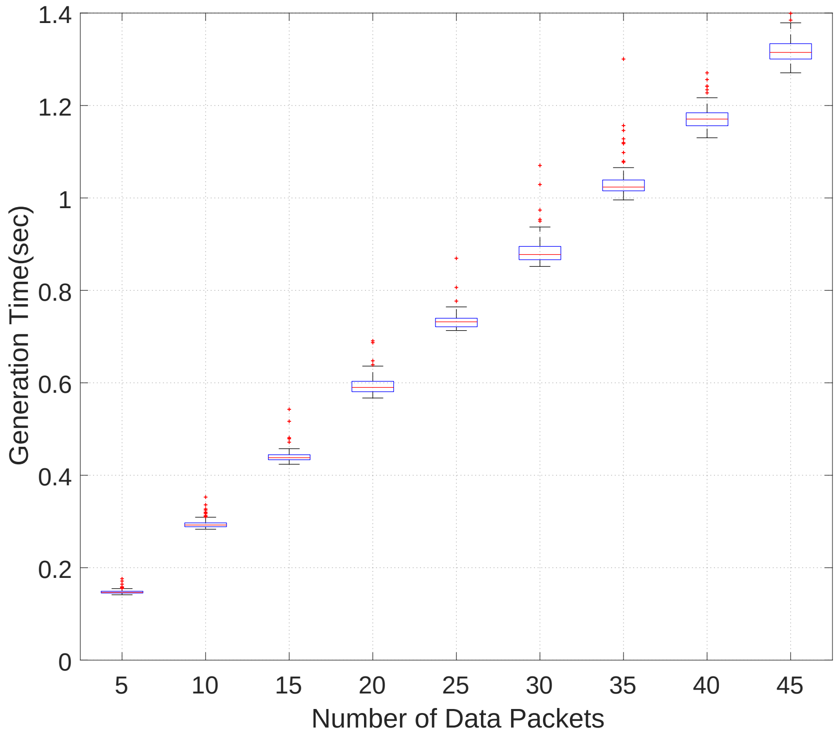


(a)

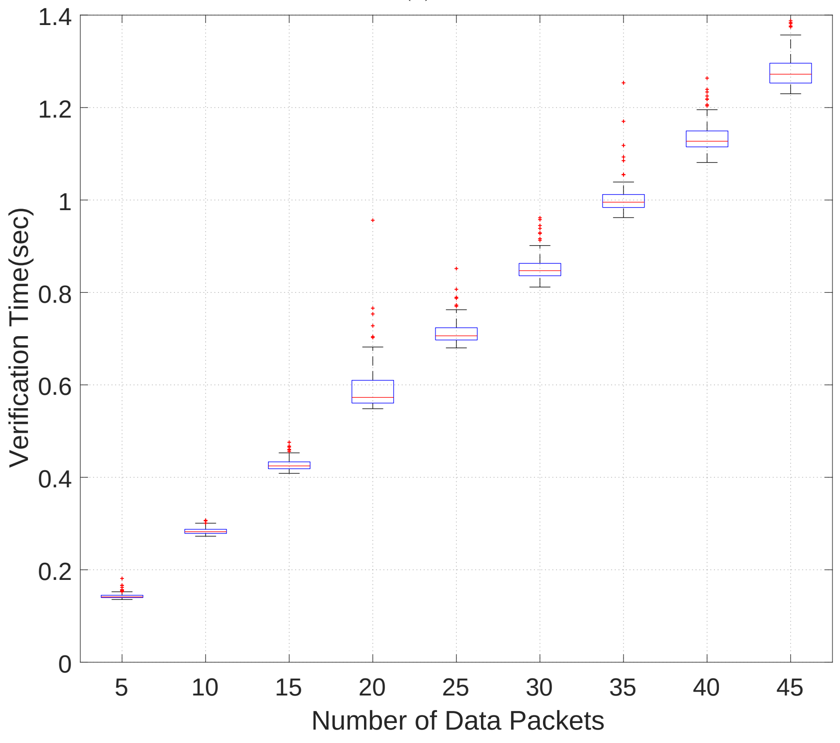


(b)

FIGURE 5.1: Computational time. (a) Watermark generation and embedding time using AES. (b) Watermark verification time using AES.

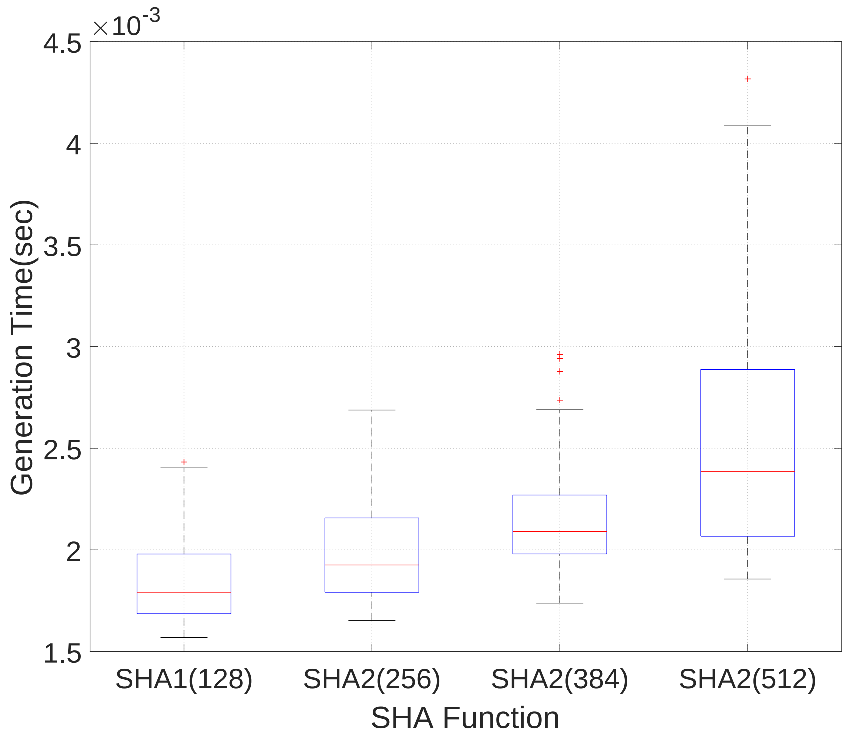


(a)

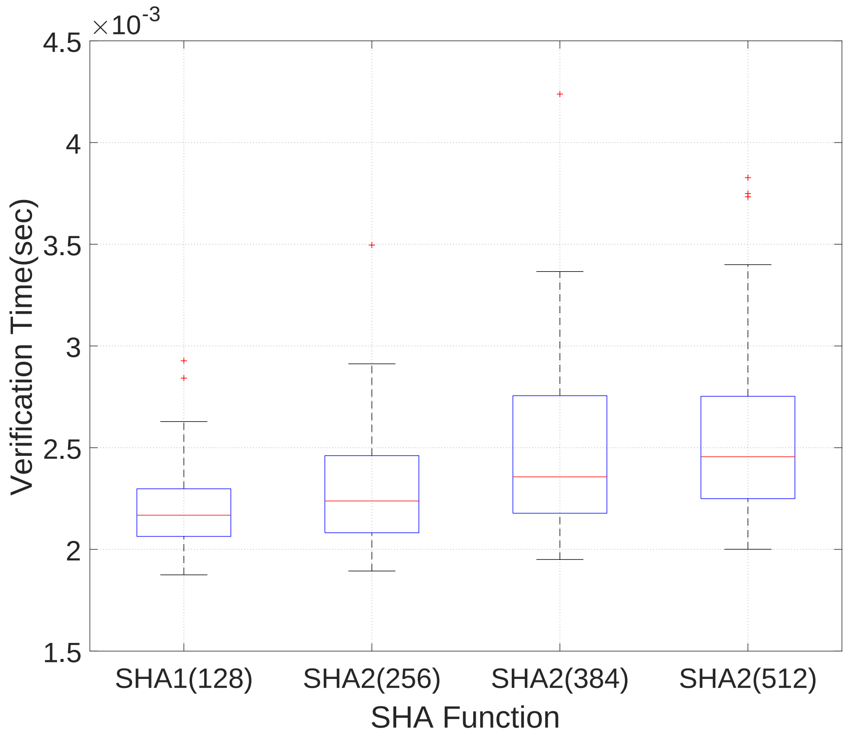


(b)

FIGURE 5.2: Computational time. (a) Watermark generation and embedding time using DES. (b) Watermark verification time using DES.



(a)



(b)

FIGURE 5.3: SHA Comparison. (a) Watermark generation and embedding time using different SHA functions. (b) Watermark verification time using different SHA functions.

the best performance in SHA-2 functions, as our hash function in the generation of watermarks.

5.3.1 Performance Evaluation

To evaluate the performance, we measure the computational time such as watermark generation, embedding, and watermark verification time of the proposed scheme, RWFS [100], ACT [98] and ZWT [48]. Additionally, we used energy utilization as another performance metrics and compared the results with existing methods [48, 98, 100]. We select these three works from the literature to compare our model with a regular watermarking technique, asymmetric cryptography technique and a zero-watermarking technique. From our research work, these papers provide these three methods and deploys it in a scenario similar to what we are analyzing and studying. Note that a confidence interval is added to show the average generation and verification time after a 100 simulation runs.

Computational Time

Computational time is described as the time required to complete the following processes: watermark generation, embedding and verification at sensor nodes and gateway.

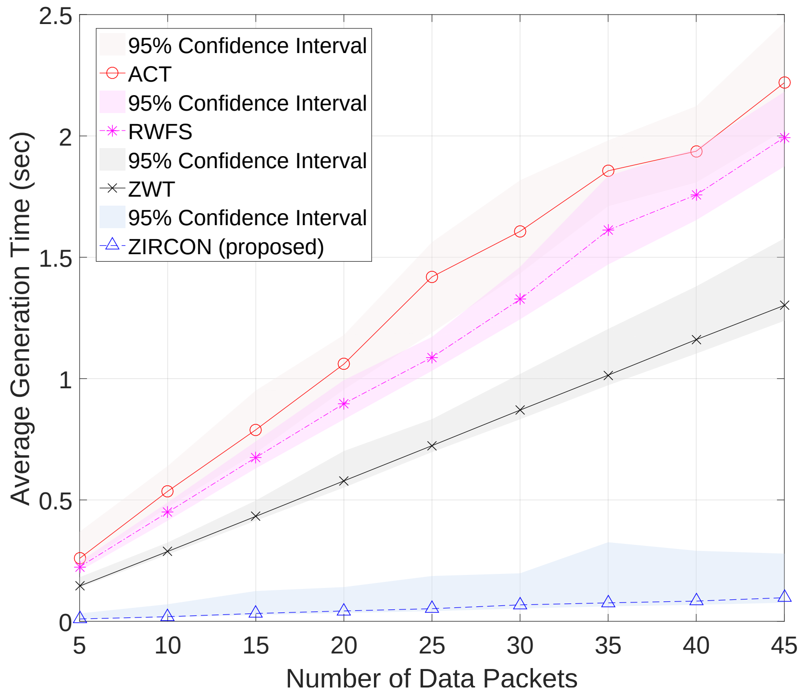
1. **Watermark generation and embedding time:** This metric measures the time taken by a node to generate a watermark and embed it into the data packet. It reflects the efficiency of the watermark generation and embedding process, which is important for real-time applications where delays must be minimized. Faster watermark generation and embedding reduce the end-to-end processing time, improving the responsiveness of the system. This is particularly important in applications like IoT, where timely data processing is essential. The existing RWFS [100] generates a watermark by encrypting the sensed data with a homomorphic encryption algorithm proposed by Castelluccia et al. [228] and passing it as an input to a keyed-hash message authentication code (HMAC). The watermark is then embedded randomly by computing each position of watermark bits using a pseudo-random number generator (PRNG) for each captured data at source node. In ACT [98], the watermark generation is based on an asymmetric cryptography function and uses group hashing for a set of data values that need to be captured in different time intervals before generating the watermark. Additionally, ZWT [48] uses DES for watermark encryption in the watermark generation process. Comparing these approaches [48, 98, 100], the proposed scheme uses a zero-watermarking technique that generates a fixed size watermark from provenance information and data features. It applies a one-way hash function to extracted data features and symmetric encryption (i. e., AES) for provenance information. Simulation results shows that the proposed scheme requires less watermark generation and embedding time than existing approaches [48, 98, 100] as observed in Figure 5.4a. Using AES as an encryption and SHA-2 to generate watermarks shows a significant improvement in the performance of sensor nodes. This results in decreasing the end-to-end time from capturing data to processing it at the destination gateway.

2. **Watermark verification time:** The verification algorithm is used to extract watermark and verify data integrity at the destination node. This procedure

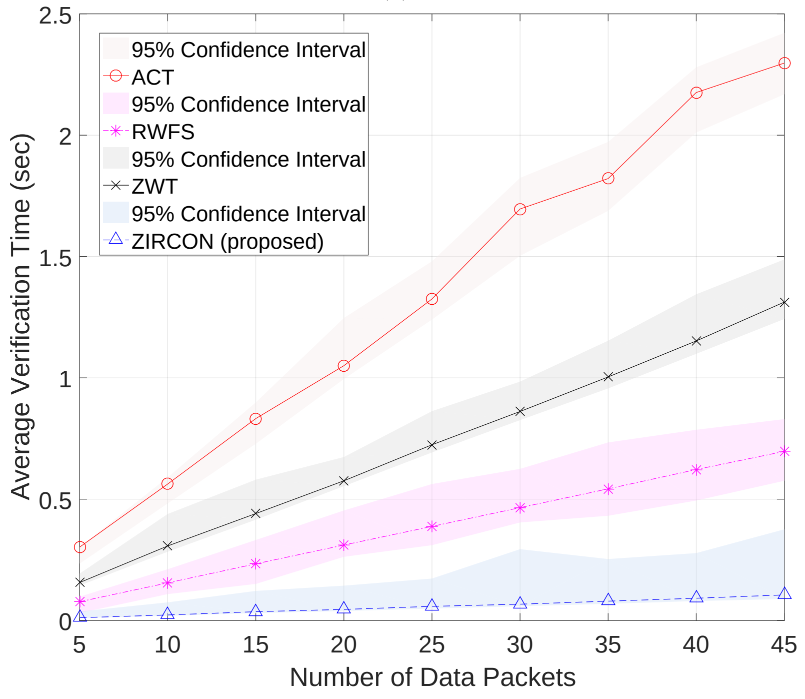
is performed at an intermediate node or gateway which have more computational and power capabilities than source nodes. Hence, this metric evaluates the time needed to extract and verify the watermark at the destination node. It indicates the efficiency of the verification process in ensuring data integrity. Shorter verification times enhance the overall system performance by reducing the processing overhead at destination nodes. They also contribute to lower latency in data verification, which is critical for time-sensitive applications. In the proposed approach, the watermark is concatenated to the data payload and each watermark is generated using AES and SHA-2 for each data packet which requires less extraction and verification time than RWFS [100], ACT [98] and ZWT [48]. The time for extracting and verifying data integrity in [100] depends on computing each watermark bit position and computing a hash value after re-encrypting the extracted data. In [98], the intermediate node or gateway requires receiving several data packets to perform watermark extraction and re-calculating the watermark based on asymmetric encryption to perform verification. Moreover, in ZWT [48], the intermediate node or gateway needs to extract data features from the received data packet and encrypt these features using DES algorithm to re-generate the watermark for verification. Figure 5.4b shows that the proposed zero-watermark approach requires less time to extract and verify data integrity than existing schemes [48, 98, 100]. It is worth pointing out that the proposed approach provides both data integrity and data provenance. The time shown in Figure 5.4b for the proposed scheme includes also the time needed for querying the stored watermarks from the database.

Energy Consumption

Energy consumption evaluates the energy consumed by a sensor node from the power utilized by each node and the total time consumed in the sensor node operation steps as shown in Figure 5.7. The energy consumed by a sensor node varies based on several basic energy consumption sources: processing time cost, radio transmission, sensor sensing, transient energy, and sleeping time cost [229–231]. It is crucial to utilize less energy-consuming security mechanisms for IoT networks due to the limited computation and power capabilities of sensor nodes. Energy consumption plays a critical role in the viability and sustainability of IoT networks, particularly those relying on battery-powered sensor nodes. Optimizing energy usage is paramount for extending the operational lifespan of these nodes, reducing maintenance costs, and minimizing environmental impact. By developing energy-efficient algorithms and protocols, the proposed scheme not only enhances the longevity of sensor nodes but also contributes to the overall resilience and affordability of IoT deployments. In the proposed scheme, we made our assumptions regarding energy consumption due to the fixed space required for watermark embedding. The phases that affect energy consumption in a sensor node are sensor node activation cost, watermark generation and embedding cost, data capturing cost, data transmission cost and cost for going to sleeping mode. The energy (E_n) of each sensor node in the network is computed according to Equation (4.3). The power (P_n) utilized by each node is determined by node's hardware components, the network's data rate, and the communication protocols used by the network. In order to estimate the power consumption of the sensor node for numerical simulation we use the energy model in [232] based on Mica2 Motes. The time to complete a round of a sensor node operation specified in Figure 5.7 is T_n which varies according to the data processing method and functionality of this node as shown in the figure. We assume, as in [232], that the parameters used in



(a)



(b)

FIGURE 5.4: Computational time. (a) Watermark generation and embedding time. (b) Watermark verification time.

the energy calculation are as follows: T_A = 1ms (Active time cost), T_S = 0.5 ms (Data sensing time), T_C (Computation and processing time), T_{TR} = 300 ms (Data transmission time), T_{SL} = 299 ms (sleeping time cost), and P_n = 30mW (Average power consumption of a single sensor node). Using Equation (5.2) we compute the energy of sensor nodes based on the previously specified parameters.

$$E_n = P_n \times T_n \tag{5.1}$$

$$E_n = P_n \times (T_A + T_S + T_C + T_{TR} + T_{SL}) \tag{5.2}$$

The analysis of the energy consumption of ZIRCON scheme compared to RWFS [100], ACT [98] and ZWT [48] shows that our approach requires less energy for each operating node. This results in an increase in life time of our network compared to other networks. The higher energy consumption in RWFS [100] is based on the computation of bit positions for watermark embedding and the encryption of captured data (that is used as an input to an HMAC function to obtain a final watermark) using homomophic encryption algorithm. This method is slower than conventional symmetric encryption methods because of the complex mathematics it requires. In comparison to homomorphic encryption, symmetric encryption is quicker and easier to use because it uses a single key to encrypt and decrypt data. Also, in ACT [98] the use of asymmetric cryptography functions and group hashing requires more energy at each sensor node due to the additional computational overhead required for the public and private key operations. The existing scheme ZWT [48], which uses DES for watermark encryption, dissipates higher energy than the proposed scheme which uses AES for sub-watermark encryption. Figures 5.5 and 5.6 shows the energy consumption of the proposed scheme compared to existing state-of-the-art methods RWFS [100], ACT [98] and ZWT [48], for a single source node and an intermediate node respectively. It is clearly shown that ZIRCON requires less energy consumption at each node of the network.

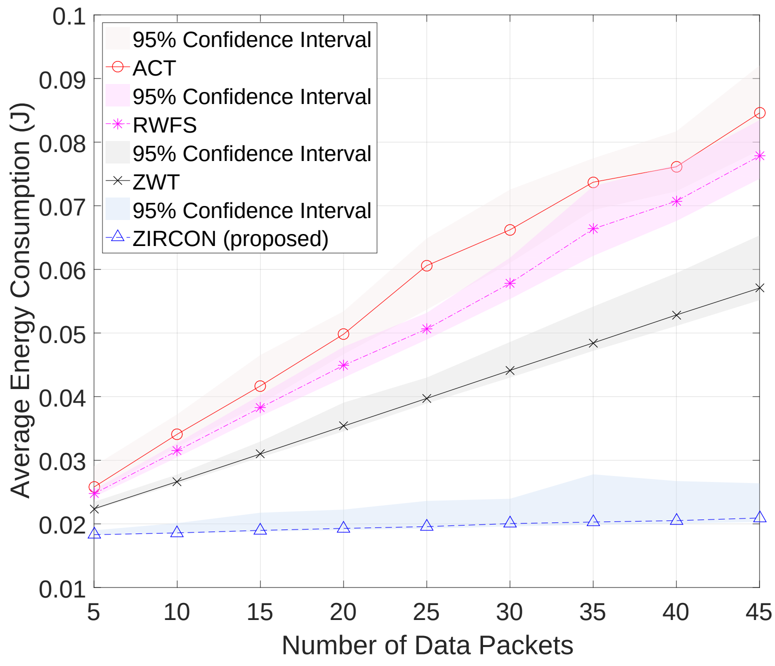


FIGURE 5.5: Energy consumption cost per single source node.

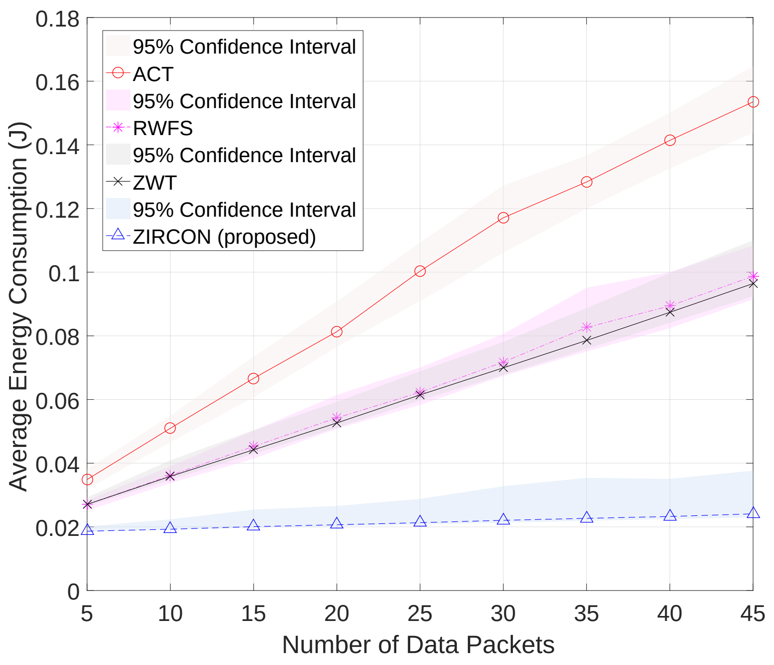


FIGURE 5.6: Energy consumption cost per single intermediate node.

5.3.2 Cost Analysis

Regarding cost analysis, we compare ZIRCON with three state-of-the-art methods in terms of transmission data size and data packet length. Transmission data size refers to the amount of data transferred over the network for each communication session or operation. It assesses the efficiency of data transmission in IoT networks by quantifying the volume of information exchanged between nodes. It accounts for both payload data and any additional metadata, such as provenance information or watermarks. Understanding transmission data size is important for optimizing network bandwidth usage and resource allocation. Smaller data sizes reduce transmission overhead, leading to faster communication, reduced latency, and improved network scalability. Moreover, minimizing data size conserves energy and extends the battery life of constrained IoT devices, enhancing overall network sustainability and operational efficiency. The second metric is packet length which refers to the size or length of individual data packets transmitted within the network. It evaluates the granularity of data transmission and the size of individual units of information exchanged between nodes. It includes factors such as payload size, header information, and any additional protocol-specific overhead. Optimizing packet length helps minimize transmission delays, enhance real-time responsiveness, and facilitate efficient use of network resources, making it essential for building robust and scalable IoT infrastructures. The state-of-the-art approaches for cost analysis are as follows:

1. The secure provenance framework $SProv$ [67] that is adapted to sensor networks by [33]. The provenance record at a node n_i is $p_i =< n_i, \text{hash}(D_i), C_i >$, where $\text{hash}(D_i)$ is a one way hash function of the updated data and $C_i = \text{sign}(\text{hash}(n_i, \text{hash}(D_i) \,|\, C_{i-1})$ is an integrity checksum. This method is referred to as SSP.

2. The MAC-based provenance scheme which computes a MAC value and send it with the node ID as the provenance record. This method is referred to as MP [33].

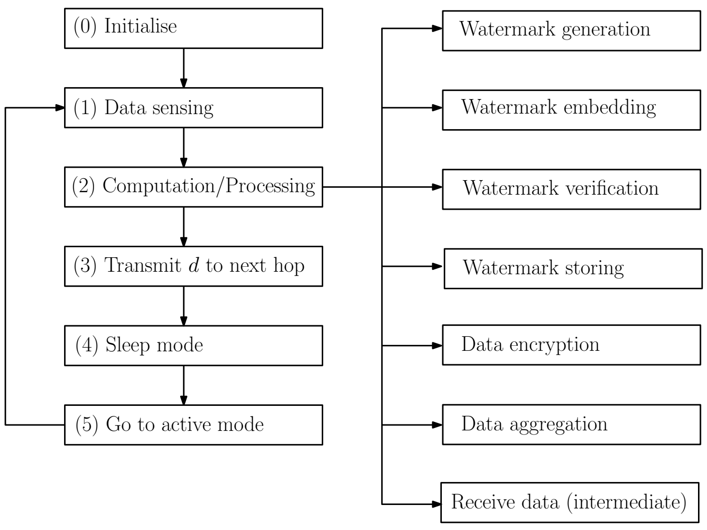
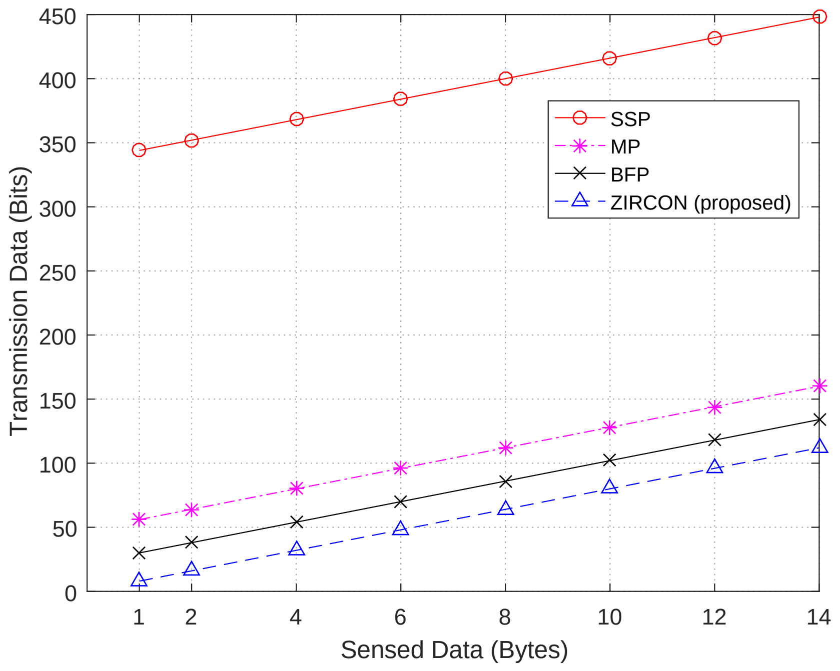


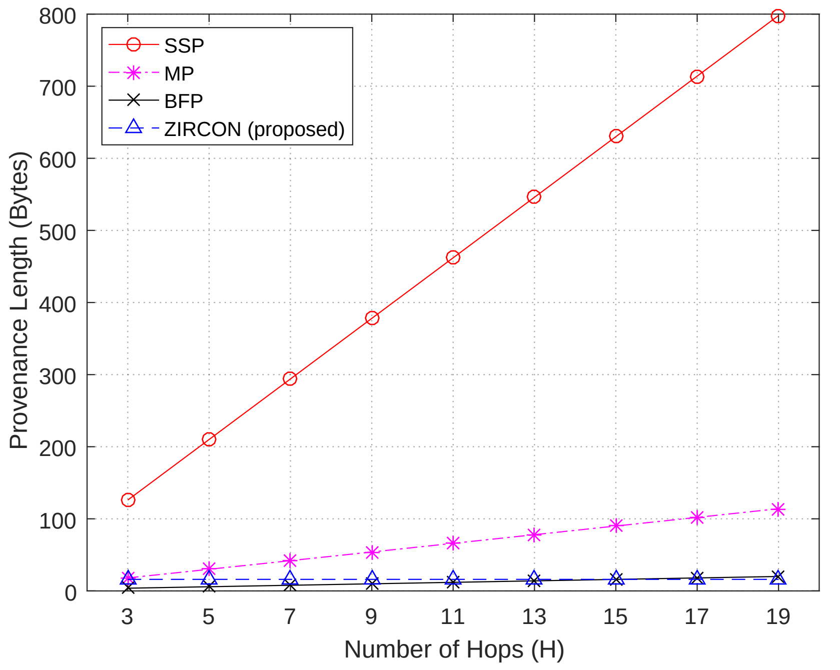
FIGURE 5.7: Sensor node operation cycle. The initialization phase is represented as step 0 in the cycle, which is not included in the performance evaluation of computational time and energy utilization of a node. The processes on the right side indicate various computation and processing tasks that a sensor node may perform.

3. A lightweight secure scheme BFP [33] that uses Bloom Filters to encode provenance information, which is sent along the data path with the data packet.

In the proposed scheme, a sensor node transmits both provenance information (IP address,timestamp, and sequence number) and a hash value as a zero-watermark. The IP address, packet timestamp and sequence number have a size of 4 bytes each. The source node encrypts the sub-watermark $sw_{f_{n,k,i}}$ and produce an encrypted sub-watermark of 16 bytes. Also, the source node computes the hash value from the extracted data payload, as shown in Algorithm 1, and selects the first 8 bytes. This implies that the generated zero-watermark including the provenance record is 24 bytes. The provenance record is stored in a tamper-proof network database at each hop. Hence, each data packet holds only one generated zero-watermark in each hop. For SSP, to perform cryptographic hash operations, they utilize SHA-1 with a bit length of 160, and for generating digital signatures of 160 bits (ECDSA), they make use of the TinyECC library [233]. The node ID, which is 2 bytes long, results in each provenance record being 42 bytes in length. To implement MP, the provenance record is formed of node ID and a MAC value computed on each source node. It uses the TinySec library [234] to compute the sensor CBC-MAC of size 4 bytes. Thus, the provenance record is of 6-byte size. In both schemes, SSP and MP, each node embeds its provenance information as a record with data packet, as the path length increases the provenance size increases linearly. This increase in provenance leads to an increase in the transmitted data packet size. In a multi-hop scenario, the provenance is $6 \times H$ bytes (i. e., the path is formed of H hops) for MP and $42 \times H$ bytes for SSP. However, in BFP, the provenance length depends on parameter selection of the Bloom Filter. For a given H and a false positive probability $P_{fp} = 0.02$, the number of required bits to encode the provenance information is $m = (-H \cdot \ln(P_{fp}))/(\ln 2)^2$. In this case the length of a BF grows with the number of nodes. Figure 5.8a shows a comparison between ZIRCON, SSP, MP and BFP approaches in-terms of transmission data size in a single hop scenario. Similarly, the results for data packet length in a Multi-hop scenario for both schemes is shown in Figure 5.8b. In resource constrained networks,



(a)



(b)

FIGURE 5.8: Cost comparison. (a) Transmission data size in the single hop scenario. (b) Provenance length in the multi-hop scenario.

energy is mainly affected by data transmission, which increases as the data packet increase. The results show that ZIRCON performs better than SSP and MP as the number of hops increases in the sensor network. Also, our algorithm outperforms BFP in terms of provenance length and scalability as the size of the network increases, and as the number of hops exceeds 11. the proposed model only encodes one provenance record $p_{n,k,i}$ with each data packet $d_{n,k}$ during transmission.

5.4 Discussion

The related literature includes many proposed schemes for ensuring data integrity and secure provenance transmission in WSNs using digital watermarking. These models are elaborated in Chapter 3. The limitations of such solutions were addressed in the proposed scheme. In this scheme we combine both data integrity and secure provenance transmission, taking into consideration the computational capabilities of sensor nodes in IoT networks, while maintaining security standards. IoT networks are vulnerable to many type of attacks. These networks are used in decision making processes that require high level of security. Moreover, it is essential in many situations to keep track of the data captured from sensor nodes to identify any malicious traffic in the context of intrusion detection systems. For this, it requires to overcome a set of challenges in order to securely transmit provenance information.

The challenges include managing the processing overhead of each network node, efficiently transmitting information about the data's origin without using additional bandwidth, and quickly responding to security breaches. Provenance information grows very fast, which requires transmitting large amount of provenance information with data packets. In fact, building the lineage of each data-packet requires storing the information of the data-packet including the complete set of nodes that were covered from source to destination. Embedding such vast amount of information with the data packet will result in a massive network overhead. This requires a solution for handling this amount of provenance information. This critical problem was not addressed in the related literature. In this context, we propose the use of a tamper-proof database to store these information that are embedded in watermarks at each node covered in the network. Hence, to obtain the required security standards, the proposed zero-watermarking approach generates two sub-watermarks that are used for integrity verification and secure provenance transmission. The sub-watermarks are based on one-way hash function (i. e., SHA-2) and symmetric encryption (i. e., AES). In our work, we provide an efficient and secure way to keep track of the whole network route that a piece of information has taken despite bandwidth overhead, storage limitations and computational overhead, while ensuring data integrity.

Managing internal data packets is a key component of improving IDS efficiency. Analyzing each data packet by the IDS at each node implies additional computational overhead. This issue was not addressed in the related literature, which only focus on data packets that are specified for sensed data. In our model, we propose a protocol for labeling internal managing data packets which allows to check for any attack at the level of these packets without the need to analyze it by IDS. In our work, we validate our zero-watermarking algorithm through a security analysis that shows our approach is robust to many attacks based on an attack model.

We provide a performance evaluation to analyze computational time, energy consumption and cost analysis in comparison with related literature. As a result of our security scheme outlined and proposed in this model, there are several areas for future study and improvement. The fast evolution of security attacks against IoT networks, such as Distributed Denial of Service (DDoS), Botnets, Privacy invasion and Physical attack, requires the advancement in security measures to protect against these types of attacks and to ensure the security of IoT networks. This presents an important call to discover ways to tackle the vast number of security attacks that are not yet studied in the area of securing data provenance and integrity in IoT networks. Another issue is that large-scale IoT networks that introduce the problem of large-scale provenance need to be analyzed. How to handle the huge lineage of data being transmitted over long data-path. Even in the presence of a database, how to manage methods to efficiently overseeing a large quantity of sensor nodes and the data they collect, along with information about its origin and the path it covers.

5.5 Summary

In Chapter 4, we address the problem of data integrity and secure transmission of provenance information for IoT networks. We propose a zero-watermarking approach that embeds provenance information and data features within data packets, storing these watermarks in a tamper-proof network database.

In this chapter, the security capabilities of our approach, ZIRCON, are demonstrated to be robust against various types of sensor network attacks through formal security analysis. We present a set of theorems and their proofs against different type of attack scenarios. Moreover, our findings are validated by conducting representative simulations in MATLAB™ and comparing our results with existing schemes based on different performance parameters such as computational time, energy consumption, and provenance length overhead. To choose the best technique for watermark generation, we conduct experiments on using different cryptographic techniques and hash functions. The results indicate that the proposed scheme is lightweight, offers better computational efficiency, and consumes less energy compared to prior methods.

Chapter 6

Zero-watermarking and Intrusion Detection System

6.1 Introduction

In the rapidly evolving digital world, network security stands as a critical concern. With vast amounts of data generated by organizational and individual activities, and the increase of cyber-attacks, securing information systems, computer and IoT networks from malicious intrusions is crucial. Traditional security measures like firewalls and digital signatures fall short in detecting unknown attacks. Anomaly-based NIDS provide a powerful tool for detecting abnormal network behavior. Using ML, these systems aim to differentiate between normal and malicious activity. However, challenges persist, including the computational complexity and the risk of misclassification errors.

In this chapter, we introduce ZW-IDS, a novel approach to improve anomaly-based NIDS performance. We integrate zero-watermarking using data provenance and ML-based approach as a two-layer classification NIDS. The first layer uses SVM with feature selection using ensemble learning model. The second layer employs data provenance information extracted from data features to generate unique zero-watermarks for each data packet. This two-layer approach aims to minimize the false positives and false negatives, improve computational complexity, and enhance the classification performance of NIDS.

We evaluate our approach using the network traffic NSL-KDD *NSL-KDD Dataset* [235] and CICIDS2017 Sharafaldin, Lashkari, and Ghorbani [27] datasets and assess its effectiveness through classification performance and computational time metrics. We study the effect of introducing the generated zero-watermark as a new feature for classification. Furthermore, we conduct a comparative analysis to demonstrate the performance of our scheme over other multi-method ML and DL existing solutions.

6.2 Dataset Description

6.2.1 NSL-KDD

In our first implementation we use the NSL-KDD dataset. It is a version of the KDD Cup 99 dataset, which is widely used for evaluating IDS. The NSL-KDD dataset is a pre-processed version of the original KDD Cup 99 dataset, designed to address some of its limitations, such as redundancy and irrelevant features. The pre-processing steps applied to the NSL-KDD dataset include removing duplicate records, converting categorical features to numerical representations, and balancing the class distribution.

The NSL-KDD dataset consists of network traffic data collected from a Local Area Network (LAN) testbed. It contains a mix of normal and attack traffic, with attacks belonging to four main categories: DoS, Probe, User to Root (U2R), and Remote to Local (R2L). The dataset provides various features extracted from network packets, including information about protocols, services, flags, and more. Each instance in the dataset is labeled as either normal or belonging to one of the attack categories. It includes 42 features. The train dataset consists of $125,973$ records and the test dataset contains $22,544$ records.

Researchers often use the NSL-KDD dataset to develop and evaluate IDS and ML models for network security tasks. Its balanced class distribution and reduced feature space make it suitable for training and testing IDS in realistic scenarios. In our first implementation we use NSL-KDD to evaluate the performance of different SVM models and our proposed model with IDS. We transform the dataset into a binary-classification dataset of two classes 'normal' and 'attack' class. An overview of the dataset is shown in Table 6.1

TABLE 6.1: Overview on NSL-KDD data.

Dataset type	Number of data samples		
	Records	Normal	Attack
NSL-KDD Train	125973	67343	58630
	%	53.45	46.54
NSL-KDD Test	22544	11245	11299
	%	49.88	50.11

6.2.2 CICIDS2017

CICIDS2017, Canadian Institute for Cybersecurity Intrusion Detection Evaluation Dataset 2017, is a network dataset widely used in cybersecurity research for evaluating IDS. It was created by the Canadian Institute for Cybersecurity (CIC) at the University of New Brunswick. The dataset contains a set of network traffic data, including normal traffic as well as various types of cyber attacks and intrusions. These attacks cover a range of different attack scenarios and techniques, providing a realistic and challenging dataset for testing the effectiveness of IDS. CICIDS2017 includes traffic collected from a realistic network environment, making it suitable for training and testing intrusion detection models under conditions that describes those encountered in real-world network environments. The dataset is labeled, where each traffic instance is annotated with information about whether it represents normal benign traffic or a specific type of attack.

One of the important features of CICIDS2017 is its scale. It contains a substantial number of samples, providing a rich source of data for experimentation and evaluation. The dataset includes tens of millions of traffic instances, allowing to perform analyses and assessments of IDS performance. It provides a CSV file for each type of attack which makes the number of samples applicable for training and testing. It can be used for IoT scenarios, particularly those involving network traffic analysis and intrusion detection in IoT environments. While the dataset may not specifically focus on IoT traffic, it contains a diverse range of network traffic data, including various types of attacks and normal traffic. Many of the attack scenarios present in the dataset are applicable to IoT environments, as IoT devices are vulnerable to many of the same types of attacks as traditional networked systems.

The dataset includes a number of attacks such as Brute Force FTP, Brute Force SSH, DoS, Web Attack, Botnet and DDoS. The dataset includes 2.8 million samples of network traffic. IoT devices have been increasingly targeted in DDoS attacks due to their large numbers, poor security configurations, limited computational capabilities, and always-on nature. To make evaluation of our model feasible, we use the DDoS network traffic CSV file and split the dataset into 70% training and 30% testing samples. The chosen dataset shows a good balance between normal and attack packets and includes 85 data features extracted from network traffic. Table 6.2 shows an overview of the classes within the dataset.

TABLE 6.2: Overview on CICIDS2017 data.

Dataset type	Number of data samples		
	Records	Normal	Attack
CICIDS2017 Train	158021	68311	89710
	%	43.22	56.77
CICIDS2017 Test	67724	29407	38317
	%	43.42	56.57

6.2.3 Justification for the use of NSL-KDD and CICIDS2017

In this work, we have chosen to evaluate the proposed approach using the NSL-KDD and CICIDS2017 datasets. While these datasets are not specifically tailored to IoT environments and are considered relatively older, their selection is justified based on several factors, which are widely supported by the existing literature.

Established Benchmarks in Intrusion Detection Research

NSL-KDD and CICIDS2017 have long been used as benchmark datasets for evaluating NIDS, owing to their structured labeling, diversity of attack types, and balanced distribution of attack classes. These characteristics provide a robust testing ground for intrusion detection algorithms. Numerous studies have demonstrated that using these datasets facilitates consistent and comparable evaluations of novel methods, including those intended for IoT scenarios that intrusion detection techniques often share underlying principles regardless of the specific network environment, leveraging these established datasets allows researchers to validate the core functionalities of detection systems before adapting them to more specialized IoT datasets.

Applicability to IoT Scenarios

While NSL-KDD and CICIDS2017 were originally created for traditional IT networks, many recent studies have successfully applied these datasets to IoT research. The datasets encompass a range of network traffic characteristics, including attack patterns and normal behaviors, which are also relevant in IoT environments. For instance, DDoS attacks, probing, and remote-to-local exploits, commonly seen in traditional networks, are likewise prevalent in IoT networks due to their inherent vulnerabilities .

By using these datasets, researchers can assess the performance of intrusion detection techniques in identifying generalized attack behaviors, which are transferable to IoT contexts. The ability to generalize results across different environments is important in developing scalable, robust NIDS solutions that can adapt to emerging IoT-specific threats.

6.3 Evaluation Metrics

The performance metrics used to evaluate the approach include accuracy, precision, recall, F-score, false negative rate (FNR), false positive rate (FPR) and computational time. These metrics are further categorized into True Positive (TP), False Negative (FN), False Positive (FP), and True Negative (TN) values. TP represents correctly identified positive cases, FN denotes negative cases incorrectly labeled as positive, FP indicates positive cases incorrectly labeled as negative, and TN signifies correctly identified negative cases. A positive case is when an attack occurs.

Accuracy which is the overall percentage of correct predictions made by the model.

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

The proportion of positive predictions that were actually correct is measured by precision.

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

The proportion of actual positive cases that were identified correctly is evaluated by recall.

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

The F-score, as a metric of effectiveness, computes the harmonic mean of precision and recall.

$$\text{F-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

FNR, or miss rate, is the proportion of negative cases that were incorrectly labeled as negative (FN) divided by the total number of actual negative cases.

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

FPR is the proportion of positive cases that were incorrectly labeled as positive (FP) divided by the total number of actual negative cases.

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

Additionally, in the evaluation of our model, we employ two critical metrics: the confusion matrix and ROC curves. The confusion matrix shows a detailed breakdown of our system's performance, showing the counts of true positives, true negatives, false positives, and false negatives. This breakdown is important for precisely assessing the efficacy of our intrusion detection approach, allowing us to identify areas of strength and weakness with clarity. Figure 6.1 shows the description of a confusion matrix.

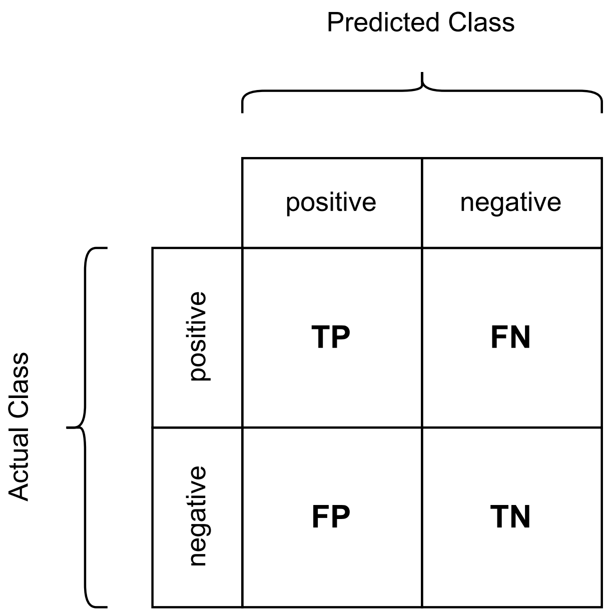


FIGURE 6.1: Description of confusion matrix.

Moreover, ROC curves provide a detailed understanding of our system's ability to distinguish between legitimate and malicious activity. By plotting the true positive rate against the false positive rate at various threshold settings, ROC curves offer insights into the trade-offs between sensitivity and specificity. This visual representation helps optimize our system's performance, allowing us to balance minimizing false alarms and maximizing the detection of actual intrusions. Figure 6.2 shows the description and elements of a ROC curve.

6.4 Augmenting Zero-Watermarking with Intrusion Detection System

In this section, we introduce a novel two-layer classification approach called ZW-IDS for intrusion detection to enhance the performance of NIDS. The approach is based

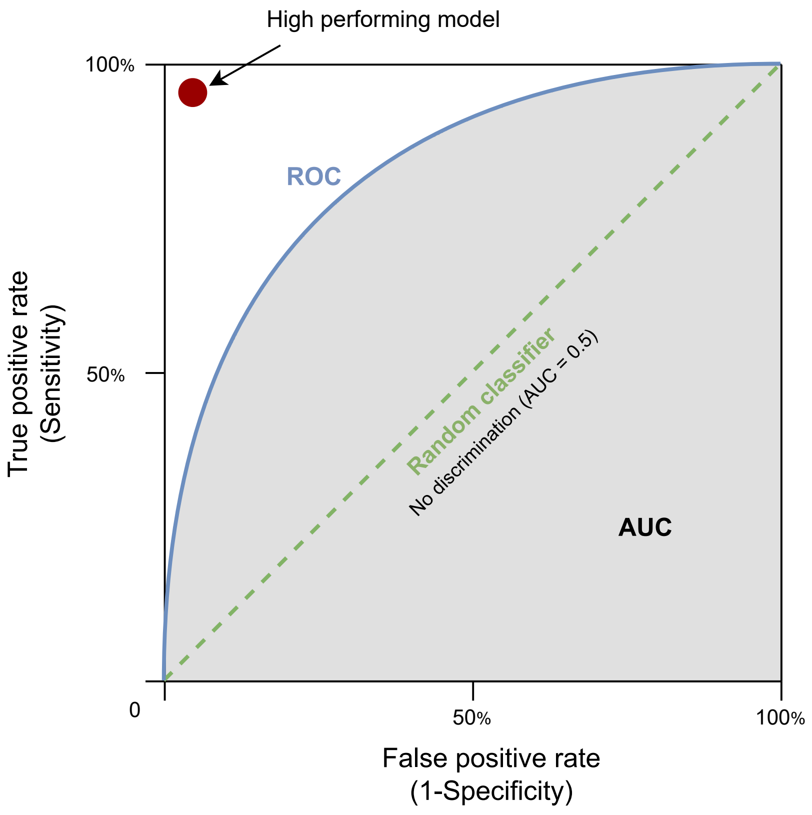


FIGURE 6.2: Description of a ROC curve.

on integrating a zero-watermarking-based scheme with an anomaly-based NIDS. The proposed approach is divided into two classification layers: (1) the first one is carried out by applying ML using SVM and feature engineering, and (2), in the second layer, the classification is performed on the classified data from the first layer using a zero-watermarking scheme with data provenance information. The workflow of the proposed model is given in Figure 6.3. The placement of the NIDS is at the gateway to capture the flow of data packets from source devices and incoming network traffic. The framework includes different components for processing data before applying the classification methodology. The proposed model is thoroughly discussed in this section.

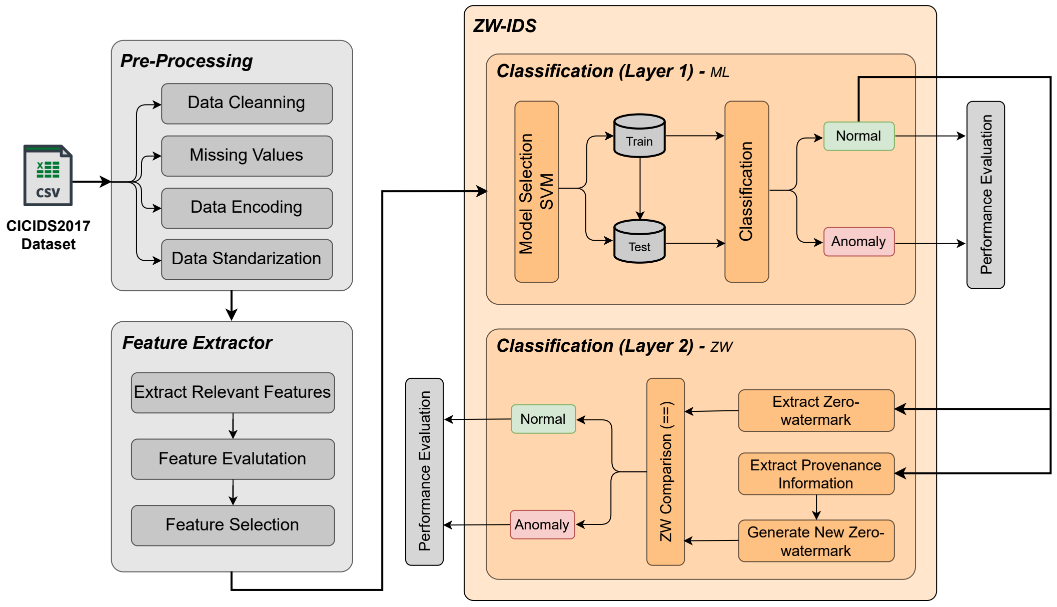


Figure 6.3: Proposed ZW-IDS Workflow. ZW-IDS consist of three main blocks. The Data Pre-processing block prepares input data for classification, while the Feature Extractor block identifies and selects optimal features for intrusion detection. The Classification block features two layers: the first employs SVM for ML-based classification, and the second layer focuses on extracting and regenerating zero-watermarks to enhance attack detection.

6.4.1 Data Pre-processing Stage

This stage cleans and prepares data from the dataset for further analysis, training and testing. This includes handling missing values, outliers, formatting inconsistencies, data normalization and standardization. The pre-processing process in our work includes the following steps:

1. **Calculating ranges for numeric features:** This function calculates and returns the range, which is the difference between the maximum and minimum values, for each numeric feature within the dataset. Range calculation is very important to provide insight into the spread or variability of numerical data. By understanding the range of each numeric feature, we can assess the scale of the data and identify potential outliers or anomalies. It helps in determining the relative importance of features during analysis and ensures that the features are on a similar scale for the SVM model, thus preventing bias towards features with

larger ranges. We notice that there are two features having infinite numbers in many data packets of the CICIDS2017 dataset which will affect the model performance. Consequently, we drop these two features that are *Flow Bytes* and *Flow Packets*. In NSL-KDD dataset, there are no infinite numbers in the numeric ranges.

2. **Extracting categorical values:** In this step, we identify and extract unique categorical values (categories) from each feature in the dataset. Categorical values represent qualitative data and are often non-numeric in nature, such as flags, protocol type, service and labels. Extracting categorical values is essential for understanding the diversity and distribution of categorical features within the dataset. It helps in identifying the different categories present in each categorical feature, which is needed for encoding categorical variables into a format suitable for SVM training.

3. **Encoding categorical values and labels:** This process encodes categorical values and labels. For labels, we have two classes *attack* and *normal*. The dataset includes a label feature where normal traffic is labeled as *normal*, and any intrusion is labeled as *attack*. Encoding categorical variables converts non-numeric data into a numerical format understandable by SVM models. In CICIDS2017 dataset, we drop some categorical values that are not useful in the classification procedure such as: *Flow ID*, *Source IP*, *Destination IP*, *Timestamp*, and *Label*. This results in a 78 numeric feature in CICIDS2017 and 41 in NSL-KDD.

4. **Data standardization:** Also known as data scaling or normalization, is a pre-processing technique used to transform the numeric features of the dataset to have a mean of 0 and a standard deviation of 1. This process ensures that all features are on a similar scale, preventing features with larger magnitudes from dominating those with smaller magnitudes during model training. In our work, `StandardScaler()` from `scikit-learn` is used to standardize the data. This scaler calculates the mean and standard deviation for each feature and then scales each feature such that it has a mean of 0 and a standard deviation of 1. By standardizing the data before training our SVM model, we ensure that the decision boundary is not biased by features with larger scales. The standardization of these features is carried out using the following equation:

$$z_i = \frac{x_i - \mu_i}{\sigma_i},$$

where z_i is the standardized value of the i-th feature, x_i is an individual observation of the i-th feature, μ_i is the mean of the i-th feature, and σ_i is the standard deviation of the i-th feature.

6.4.2 Zero-watermark Generation and Embedding

Provenance Information Extraction

Within the selected CICIDS2017 dataset, we encounter 85 data features, with 5 of them being non-numeric: source IP address, destination IP address, timestamp, flow ID, and label. These particular features are consistently removed by existing ML-based intrusion detection approaches during the data pre-processing phase, as they hold no relevance to the classification process. However, we recognize their significance as provenance information and choose to extract them for zero-watermark generation process. Initially, a sequence number, part of provenance information, is created for each packet, using extracted features as the source and destination IP addresses, timestamp, and flow ID. These accumulated provenance information, along with the generated sequence number, is used to generate the zero-watermark for each individual observation or data packet x. In the NSL-KDD dataset, we encounter 41 numeric features. In NSL-KDD dataset there is no direct provenance information such as source IP address, destination IP address and timestamp that we use to generate our zero-watermark. To evaluate our approach using this dataset we used source bytes, destination bytes, duration and sequence number to uniquely distinguish packets from source nodes.

Zero-watermark Generation and Embedding Procedure

Introducing a zero-watermarking approach to augment an ML-based IDS requires a watermark generation and embedding process at each legitimate source device. We propose a new zero-watermark generation and embedding algorithm to embed a watermark to the transmitted data packets using provenance information. Algorithm 8 describes the process of generating and embedding a watermark. It accepts data packets from the source device to generate a final zero-watermark. The algorithm extracts provenance information from the data features of each data point such as, *source IP address*, *destination IP address*, *timestamp* and combines it with a generated unique data packet sequence number (seq) to generate a sub-watermark $sw_{f_{n,k}}$ as shown in Algorithm 8. Then, the sub-watermark is encrypted using the AES. The input data is padded to ensure its length is a multiple of the AES block size. $sw_{f_{n,k}}$ is encrypted using the secret 128 bit key K_j to obtain a provenance record $p_{n,k} = E(sw_{f_{n,k}}, K_j)$. Another sub-watermark $sw_{h_{n,k}}$ is generated from the hash value of data payload using a one-way hash function, SHA-2. SHA-2 is preferred over other hash functions like MD5 due to its lightweight nature, consuming 65% less memory. MD5, while widely used in the past, has vulnerabilities that compromise its security [226]. Finally, these two generated sub-watermarks are concatenated to form a final zero-watermark $W_{F_{n,k}}$ using the following equation:

$$W_{F_{n,k}} = E(w_{ip} \,||\, w_t \,||\, w_{sq}\, , K_j) \,||\, H(x_{n,k}) = E(sw_{f_{n,k}}, K_j) \,||\, sw_{h_{n,k}}.$$

where $W_{F_{n,k}}$ $(1 \leq n \leq N)$, is the final watermark, N is the number of devices in the network, $||$ denotes the concatenation operator, H is a secure and lightweight one-way hash function, and n is the source device number. After the generation procedure, the final zero-watermark is embedded in the data packet x as shown in Equation 6.1 and, then, undergoes transmission.

$$x_{(n,k)W_{F_{n,k}}} = x_{n,k} \,||\, W_{F_{n,k}}. \tag{6.1}$$

6.4.3 Feature Extraction and Selection

Network connections can be described using a collection of data features, but these vary in their impact for understanding the connection's behavior. Some features provide little to no relevant information and are considered irrelevant. Others contain repetitive data and are redundant [236]. For this, we use a feature extraction and selection procedure to identify and extract relevant features from the pre-processed

Algorithm 8 : Watermark Generation and Embedding

```
input: $x_{n,k}$
output: $W_{F_{n,k}}$
1: procedure WATERMARK GENERATION
2:     $w_{ip_s} \leftarrow$ network device $n$ IP Address
3:     $w_{ip_d} \leftarrow$ destination device IP Address
4:     $w_t \leftarrow$ extracted timestamp $(x_{n,k})$
5:     $w_{sq} \leftarrow$ packet sequence number $(seq(x_{n,k}))$
6:     $sw_{f_{n,k}} \leftarrow w_{ip_s} \,||\, w_{ip_d} \,||\, w_t \,||\, w_{sq}$
7:     $p_{n,k} \leftarrow E(sw_{f_{n,k}}) \leftarrow \mathrm{ENC}(K_j, sw_{f_{n,k}})$
8:     $sw_{h_{n,k}} \leftarrow H(x_{n,k})$                ▷ select first 8 bytes of hash output
9:     $W_{F_{n,k}} \leftarrow E(sw_{f_{n,k}}) \,||\, sw_{h_{n,k}}$
10: procedure WATERMARK EMBEDDING
11:     $x_{(n,k)W_{F_{n,k}}} \leftarrow x_{n,k} \,||\, W_{F_{n,k}}$
12:     $Send\left(x_{(n,k)W_{F_{n,k}}}\right)$
```

data that are informative for anomaly detection, and evaluate and select the most important features. We test the different SVM models on different feature selection procedures. The two main cases are: (1) using all dataset features, and (2) selecting the k-important features using the ensemble learning method `ExtraTreesClassifier`. The method is as follows:

- **`ExtraTreesClassifier():`** Extra Trees, which stands for Extremely Randomized Trees, is an ensemble learning method based on decision trees. It creates a forest of random decision trees and splits nodes using random thresholds. This randomness helps to reduce overfitting and variance in the model. We train the model on the input features (78, 41 features) of the CICIDS2017 and NSL-KDD datasets, and the target variable which is the labels (normal, attack). In this step, the model's parameters is adjusted so that it can map the input data to the correct output labels.

- After training the model, the feature importance scores is calculated. Feature importance indicates the relative importance of each feature in predicting the target variable. The model returns an array containing the importance scores for each feature. Finally, we plot the obtained results showing the feature importance scores. We select the features that affect the decision of mapping each data point to a target label as shown in Figure 6.4 and Figure 6.5 for NSL-KDD and CICIDS2017, respectively. Thus, we suppress the 11 features with the least important scores, resulting in 30 features for the NSL-KDD dataset, and suppress 30 features, resulting in 48 features for the CICIDS2017 dataset.

6.4.4 Classification

This section describes the two-layered approach for intrusion detection. The first layer uses an SVM classifier to identify anomalies in the network traffic. The second layer leverages a zero-watermark approach, which uses provenance information to be embedded within the data packets. This layer adds an extra layer of security by extracting important information to verify data integrity and further enhance intrusion detection performance.

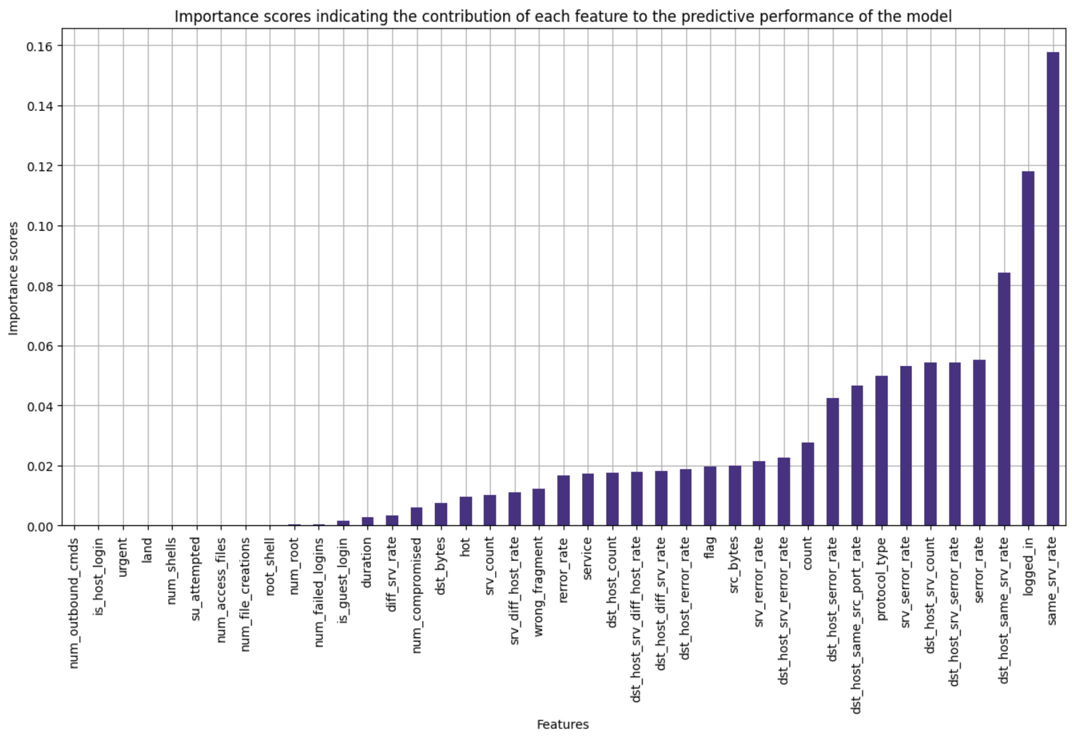


FIGURE 6.4: Feature importance scores using ExtraTreesClassifier for NSL-KDD.

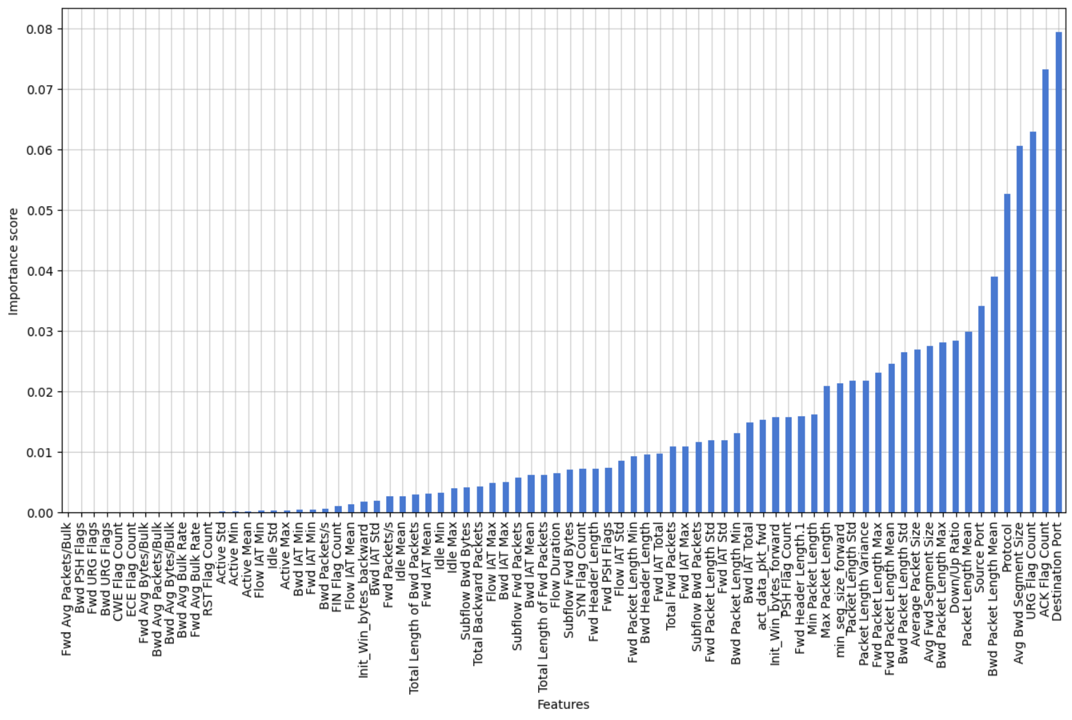


FIGURE 6.5: Feature importance scores using ExtraTreesClassifier for CICIDS2017.

Classification (Layer 1) using ML

An SVM is a supervised learning algorithm that excels at separating data packets into two categories, which is a generalization of maximal margin classifier. It sets a dividing line (hyperplane) in a multidimensional space. SVM aims to find the maximum margin (distance) between this hyperplane and the closest data points (support vectors) from each category. These support vectors define the best possible separation. New data packets are then classified based on which side of the hyperplane they fall on, although some misclassifications can occur. SVMs can handle some margin violations, making them a flexible and powerful tool for classifying received data packets as normal or attack. The separating hyperplane of $p-$dimensional space is defined by the equation:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0,$$

where $\beta_0, \beta_1, \ldots, \beta_p$ are coefficients of the hyperplane, and $X = (X_1, X_2, \ldots, X_p)^{\mathrm{T}}$ is a data-packet having $p-$features of length p. In our network dataset, we can represent it as an $n \times p$ data matrix X, which includes n training data packets in a $p-$dimensional feature space $x_1 = \begin{pmatrix} x_{11} & \cdots & x_{1p} \end{pmatrix}, \ldots, x_n = \begin{pmatrix} x_{n1} & \cdots & x_{np} \end{pmatrix}$. These data packets belong to two main classes, $y_1, \ldots, y_n \in \{-1, 1\}$, where -1 represents attack class and 1 normal class. A new data packet is received, a $p-$vector with data features $x^* = (x_1^*, \ldots, x_p^*)^{\mathrm{T}}$. Our objective is to construct a classifier utilizing our training dataset, enabling the classification of incoming data packets based on its set of features.

For this, we use a separating hyperplane that has a soft margin having the following property:

$$\begin{aligned} &\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p > M(1-\varepsilon_i), \text{ if } y_i = 1, \\ &\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p < M(1-\varepsilon_i), \text{ if } y_i = -1. \end{aligned}$$

The hyperplane is chosen to correctly separate most of the training observations into the two classes, but may misclassify a few observations. It is the solution to the following optimization problem:

$$\begin{aligned} &\underset{\beta_0,\beta_1,\ldots,\beta_p,\varepsilon_1,\ldots,\varepsilon_n}{\text{maximize}} \quad M \\ &\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1, \\ &y_i \left(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}\right) \geq M\left(1-\varepsilon_i\right), \\ &\varepsilon_i \geq 0, \sum_{i=1}^{n} \varepsilon_i \leq C, \end{aligned}$$

where C is a non-negative tuning parameter, and M is the width of the margin to be maximized. The variables $\varepsilon_1, \ldots, \varepsilon_n$ act as slack variables, used for data packets that fall on the wrong side of the margin or hyperplane. After optimizing the model, we classify a new data packet x^* by assessing its position relative to the hyperplane. Hence, we classify new data based on the sign of $f(x^*) = \beta_0 + \beta_1 x_1^* + \beta_2 x_2^* + \ldots + \beta_p x_p^*$.

The support vector classifier is effective for linear classification in a two-class scenario, but real-world boundaries are often nonlinear. SVM extends this by enlarging the feature space using kernels. Solving the SVM problem relies on the inner products of of the data points. Thus, the inner product of two data points $x_i, x_{i'}$ is as follows:

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}.$$

The inner product appears every time in the representation or the calculation of the solution for the support vector classifier, for that it is replaced with a generalization of the inner product of the following form: $K(x_i, x_{i'})$, where K is a function called *kernel*. There are several kernels that can be used for classification in SVM method such as *linear*, *polynomial*, *radial*, and *sigmoid*. In our approach, we train these four models on labeled data to learn a decision boundary between normal and anomaly packets based on the selected features in Section 6.4.3. Then, we apply the trained model to classify new, unlabeled data packets as normal or anomaly. After classification, the classified normal packets are placed in a CSV file to be used as an input to the next classification procedure. This file holds the predicted normal packets based on the specified decision boundary by the SVM model. In this stage, we apply all possible SVM models to test which one gives the best performance in terms of accuracy, precision, recall, F-score, computational performance and highest Area Under Curve (AUC) in the Receiver Operating Characteristic (ROC) curve. This is carried out by applying a *GridSearchCV* for testing the best SVM model using $5-$fold cross-validation and get the best parameters C and γ. After obtaining the highest performance from parameter tuning, we also apply Principal Component Analysis (PCA) to reduce the dimentionality of the feature space and check the performance using two PCAs. After extensive experiments, we found that the best performance is obtained using an RBF kernel with $C = 100$ and $\gamma = 0.1$. The different kernels uses the following functions for attack classification:

1. *Linear* kernel, which is explained before as support vector classifier. It is of the form:

$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

2. *Polynomial* kernel, it is of the form:

$$K(x_i, x_{i'}) = \left(1 + \sum_{j=1}^{p} x_{ij} x_{i'j}\right)^d$$

   The *polynomial* kernel has a degree parameter d which functions to find the optimal value in each dataset. The d parameter is the degree of the *polynomial* kernel function with a default value of $d = 2$. The greater the d value, the resulting system accuracy will be fluctuating and less stable. This happens because the higher the d parameter value, the more curved the resulting hyperplane line.

3. *Radial* kernel, it is of the form:

$$K(x_i, x_{i'}) = \exp\left(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2\right).$$

where γ is a positive constant parameter that determines the influence of each training sample on the model. It defines the reach of the kernel function, controlling the flexibility of the decision boundary.

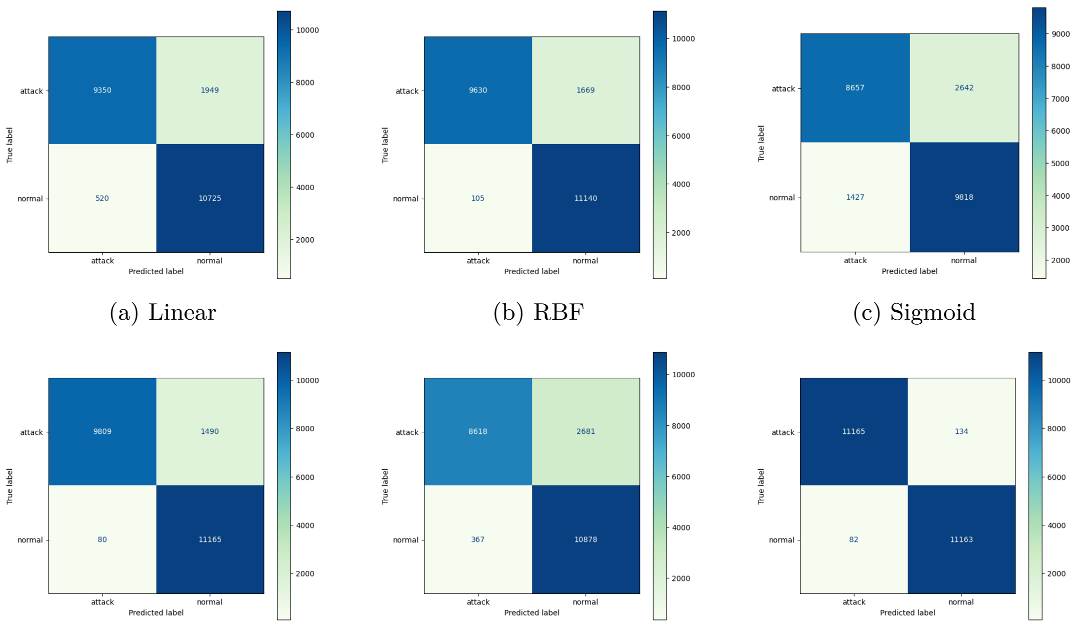
4. *Sigmoid* kernel,

$$K\left(x_i, x_{i'}\right) = \tanh\left(\alpha x_i, x_{i'}\right)$$

In this layer, we explore two scenarios using the NSL-KDD and CICIDS2017 datasets. For each scenario, we conducted multiple experiments varying the number of features and the kernel employed. The scenarios are as follows:

1. **Experiments with NSL-KDD:**

   - *Experiment 1:* In the first experiment we train the model based on all the features of the NSL-KDD dataset except the labels feature. The total number of feature is 41 feature. Figure 6.6 shows the confusion matrix of the different models used. Table 6.3 shows the evaluation metrics obtained from model testing. ROC curve is shown in Figure 6.7 to provide a detailed understanding of the performance of each model. The results demonstrate that the best performance is achieved using the *RBF* kernel with parameters $d = 3$, $C = 100$ and $\gamma = 0.1$. Moreover, this kernel configuration also provides the best computational time.



(a) Linear (b) RBF (c) Sigmoid

(d) GridSearchCV (e) GridSearchCV + PCA (f) ZW-IDS

FIGURE 6.6: Confusion matrix of the different SVM models with all features in NSL-KDD.

   - *Experiment 2:* In the second experiment, we trained the model using the most important features of the NSL-KDD dataset. The total number of selected features was 30. The selection of important features was performed using the previously described ExtraTreesClassifier. The feature importance scores are shown in Figure 6.4. We eliminated 11 features that had no significant impact on the classification process. Figure 6.8 displays

Table 6.3: Results of different SVM models with all features in NSL-KDD.

Model	Kernel	Features	Classification Performance Metrics								Computational Time (s)	
			Accuracy		**Precision**	**Recall**	**F1-score**	**FNR**	**FPR**	**AUC**	**Training Time**	**Testing Time**
SVM	Linear	All (41)	89.0481	A	94.7315	82.7507	88.3367	17.2493	4.6242	0.89063	4 m 0.3s	8.9 s
				N	84.6221	95.3757	89.6777					
SVM	RBF	All (41)	92.1309	A	98.9214	85.2288	91.566	14.7712	0.9337	0.92147	41 s	7.8 s
				N	86.9701	99.0663	92.6249					
SVM	Sigmoid	All (41)	81.9509	A	85.8489	76.6174	80.9709	23.3826	12.69	0.81963	2 m 58 s	13.7 s
				N	78.7961	87.3099	82.8348					
SVM	GridSearchCV, k = RBF	All (41)	93.0358	A	99.191	86.813	92.5901	13.187	0.7114	0.9305	24.6 s	2.8 s
	$d = 3, C = 100, \gamma = 0.1$			N	88.2886	99.2886	93.431					
SVM + PCA	GridSearchCV, k = RBF	All (41) - 2 PC	86.4798	A	95.9154	76.2722	84.9734	23.7277	3.2636	0.86504	3 m 11 s	26.1 s
	$d = 3, C = 100, \gamma = 0.1$			N	80.2272	96.7363	87.7117					
ZW-IDS	GridSearchCV, k = RBF	All (41)	99.0419	A	99.2709	98.8141	99.042	1.1859	0.7292	0.99042	39.9 s	3.2 s
	$d = 3, C = 100, \gamma = 0.1$			N	98.8138	99.2708	99.0418					

(A): *Attack class.*, (N): *Normal class.*

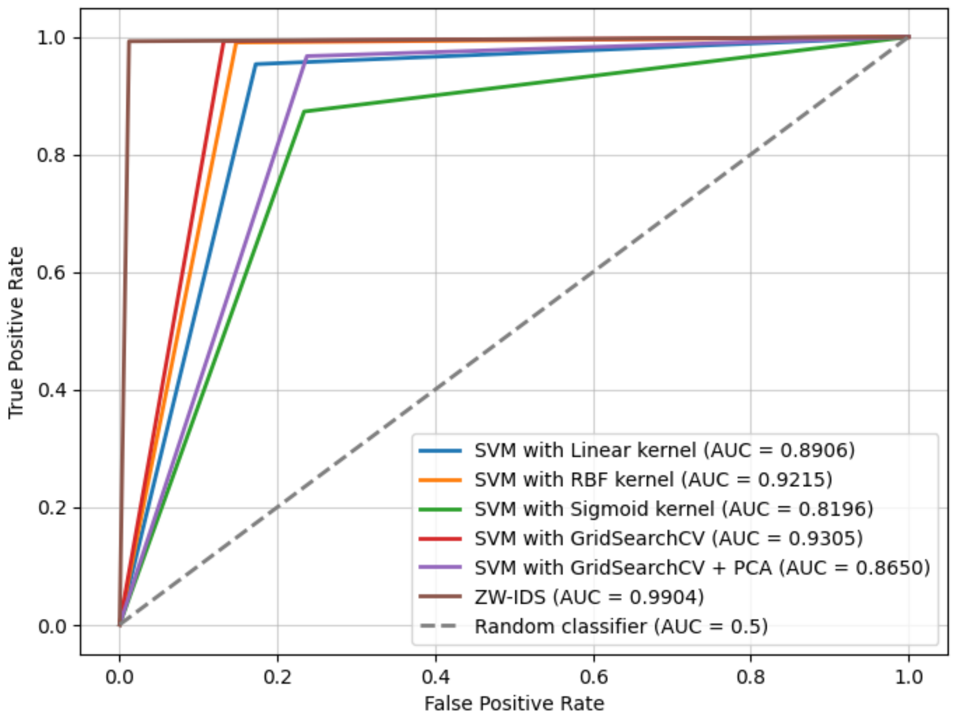
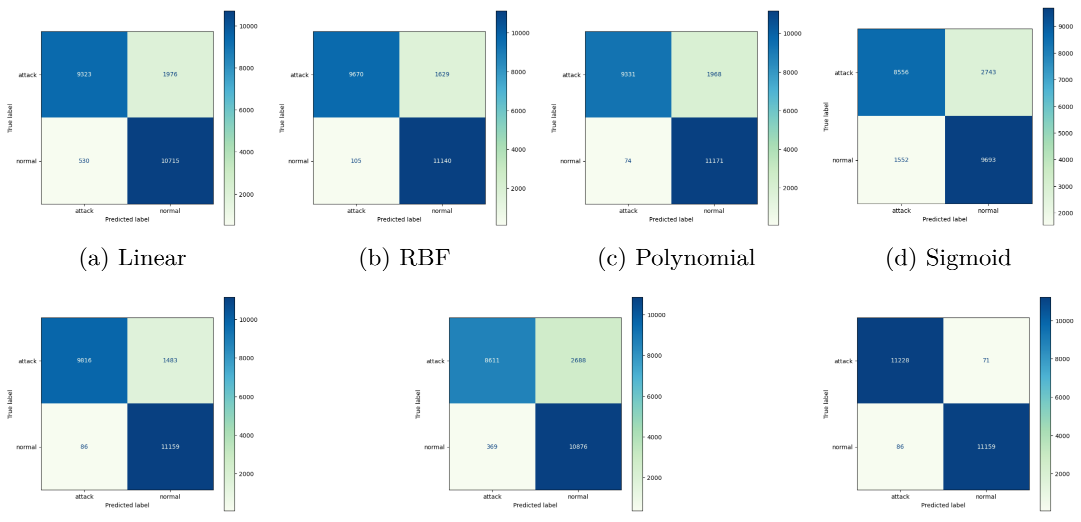


Figure 6.7: ROC curve using all features in NSL-KDD.

the confusion matrix for the different models used. Table 6.4 presents the evaluation metrics obtained from model testing. The ROC curves for the different models are shown in Figure 6.9. The results demonstrate that the RBF kernel outperforms other kernels in terms of classification accuracy and computational time. Specifically, the RBF kernel with parameters $d = 3$, $C = 100$ and $\gamma = 0.1$ achieved the best performance. Moreover, compared to the first experiment, reducing the number of features to the most important ones significantly improved the overall system performance, both in terms of classification accuracy and computational efficiency. Reducing the number of features improves intrusion detection by lowering dimensionality, which enhances generalization and reduces overfitting. It eliminates noise from irrelevant features, allowing the model to focus on the most informative data, resulting in simpler, more interpretable model. This reduction enhances computational efficiency, leading to faster processing times and better handling of larger datasets. Focusing on the most relevant features improves detection accuracy by capturing essential patterns and correlations, ultimately making the model more accurate, efficient, and robust.



(a) Linear (b) RBF (c) Polynomial (d) Sigmoid

(e) GridSearchCV (f) GridSearchCV + PCA (g) ZW-IDS

FIGURE 6.8: Confusion matrix of the different SVM models with k important features in NSL-KDD.

- *Experiment 3:* After evaluating the performance of each experiment, we found that the setup with the selected k-important features yielded the best results in terms of evaluation metrics and time efficiency. Consequently, we used this model to study the effect of introducing a zero-watermark from provenance information as a new feature to the dataset. Feature importance scores are shown in Figure 6.10, indicating that the impact of the zero-watermark is almost negligible. The correlation between features and the zero-watermark with PCA components is illustrated in Figures 6.11 which shows that the two PCA components do not depend on the introduced zero-watermark in the classification process. Table 6.5 presents the evaluation metrics from model testing. Comparing the results of using k-important features with and without the zero-watermark reveals that

Table 6.4: Results of different SVM models with k-important features in NSL-KDD.

Model	Kernel	Features	Classification Performance Metrics								Computational Time (s)	
			Accuracy		Precision	Recall	F1-score	FNR	FPR	AUC	Training Time	Testing Time
SVM	Linear	30	88.884	A	94.6209	82.5117	88.1524	17.4882	4.7132	0.8889	3 m 20s	7.2 s
				N	84.4299	95.2868	89.5304					
SVM	RBF	30	92.3084	A	98.9258	85.5828	91.7719	14.4172	0.9337	0.92324	30.5 s	7.3 s
				N	87.2425	99.0663	92.7792					
SVM	Polynomial	30	90.9422	A	99.2132	82.5825	90.1372	17.4174	0.6581	0.90962	54.9 s	3.3 s
				N	85.0217	99.3419	91.6257					
SVM	Sigmoid	30	80.9484	A	84.6458	75.7235	79.9365	24.2764	13.8016	0.80961	3 m 6.1 s	15.3 s
				N	77.9431	86.1983	81.8631					
SVM	GridSearchCV, k = RBF	30	93.0403	A	99.1315	86.8749	92.5994	13.125	0.7647	0.93055	20.2 s	3.0 s
	$d = 3, C = 100, \gamma = 0.1$			N	88.2693	99.2352	93.4316					
SVM + PCA	GridSearchCV, k = RBF	30 - 2 PC	86.4399	A	95.8909	76.2103	84.9253	23.7897	3.2903	0.86464	3 m 8 s	26.7 s
	$d = 3, C = 100, \gamma = 0.1$			N	80.1828	96.7185	87.6779					
ZW-IDS	GridSearchCV, k = RBF	30	98.2922	A	99.2242	97.3537	98.2801	0.6283	0.7647	0.98294	30.4 s	5.6 s
	$d = 3, C = 100, \gamma = 0.1$			N	97.3905	99.2352	98.3042					

(A): *Attack class.*, (N): *Normal class.*

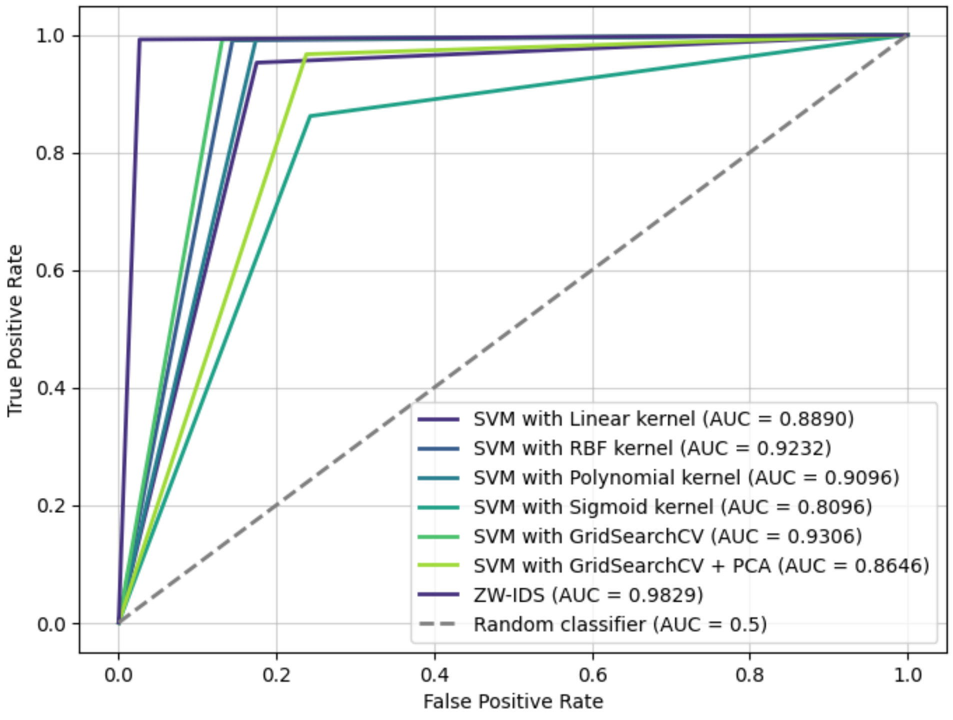


Figure 6.9: ROC curve using k-important features in NSL-KDD.

the zero-watermark has no significant effect on the classification process. Instead, it introduces additional overhead and complexity to the model, and even decreases accuracy, precision, recall, and F1-score. Therefore, we propose incorporating the zero-watermark approach within a second-level classification stage to enhance the overall performance of the IDS.

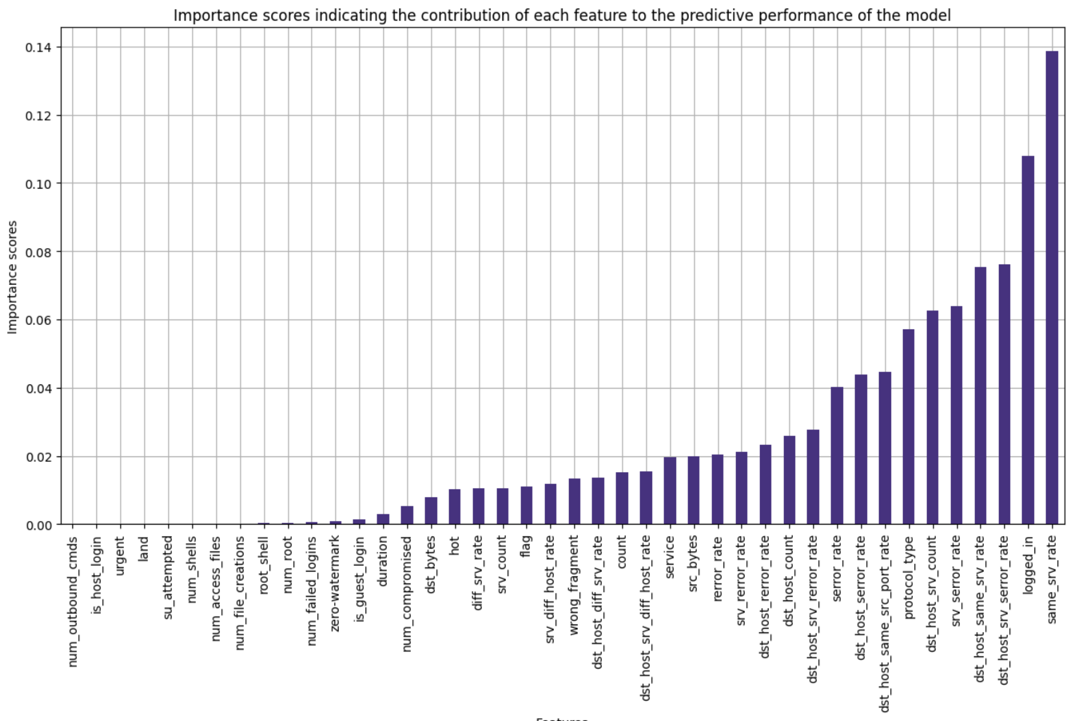


FIGURE 6.10: Feature importance score with zero-watermark in NSL-KDD.

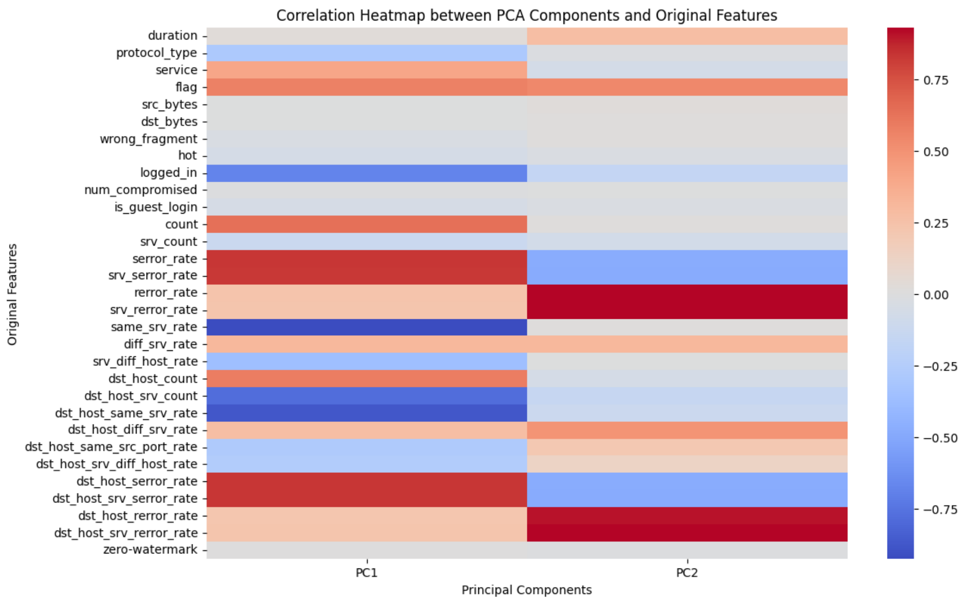


FIGURE 6.11: Correlation heatmap between PCA components, original features and zero-watermark

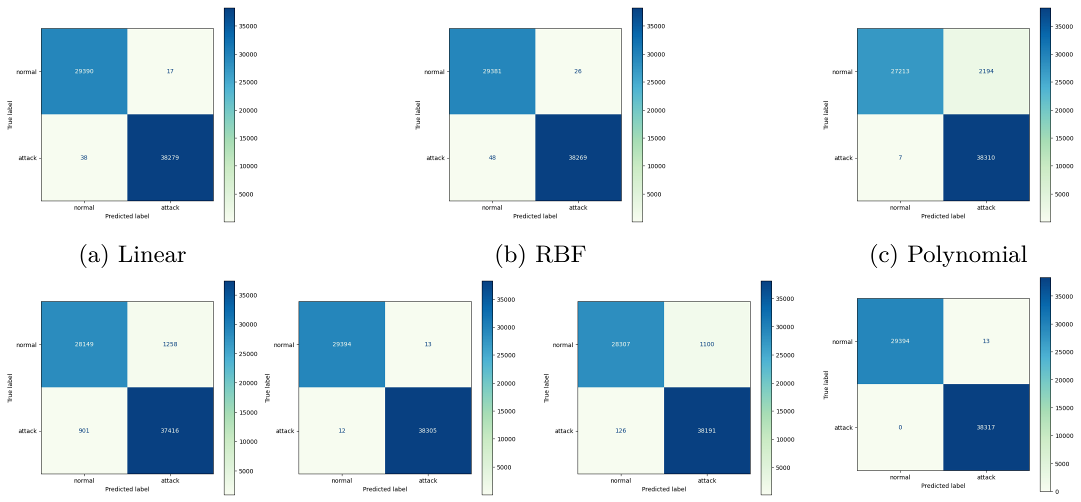
TABLE 6.5: Results of different SVM models with k-important features and zero-watermark in NSL-KDD.

Model	Kernel	Features	Classification Performance Metrics								Computational Time (s)	
			Accuracy		Precision	Recall	F1-score	FNR	FPR	AUC	Training Time	Testing Time
SVM	Linear	30 + ZW	88.8928	A	94.6311	82.5206	88.1619	17.4882	4.6954	0.88908	4 m 13.1s	7.0 s
				N	84.4378	95.2957	89.5388					
SVM	RBF	30 + ZW	92.2906	A	98.9454	85.5297	91.7497	14.4703	0.9426	0.92306	33 s	8.0 s
				N	87.2036	99.084	92.765					
SVM	Polynomial	30 + ZW	90.8623	A	99.2012	82.4321	90.0425	17.5767	0.658	0.90882	56 s	3.6 s
				N	84.9107	99.333	91.5574					
SVM	Sigmoid	30 + ZW	80.9883	A	84.7144	75.7324	79.972	24.2587	13.7216	0.810009	4 m 11.7 s	15.5 s
				N	77.9635	86.2695	81.9065					
SVM	GridSearchCV, k = RBF	30 + ZW	92.8584	A	99.1179	86.5209	92.392	13.479	0.7025	0.92873	59.9 s	3.9 s
	$d = 3, C = 100, \gamma = 0.1$			N	87.9899	99.2263	93.2709					
SVM + PCA	GridSearchCV, k = RBF	30 + ZW - 2 PC	86.4354	A	95.8802	76.2103	84.9211	23.7808	3.2903	0.86459	3 m 20.3 s	25.9 s
	$d = 3, C = 100, \gamma = 0.1$			N	80.1814	96.7096	87.6733					

(A): *Attack class.*, (N): *Normal class.*

2. **Experiments with CICIDS2017:**

- *Experiment 1:* In our initial scenario, we built an IDS using all features from the CICIDS2017 dataset, incorporating our novel zero-watermarking classification layer. We evaluated different combinations of our proposed ZW-IDS model with different SVM kernels, including linear, RBF, polynomial, and sigmoid. Using *GridSearchCV*, we optimized the hyper-parameters (C, γ, d) to achieve the best performance in terms of classification metrics and computational efficiency. Notably, we found that setting hyper-parameters to $C = 100$, $\gamma = 0.1$, and $d = 3$ with the RBF kernel, combined with zero-watermarks generated using AES encryption with a 128-bit secret key, show the best results as shown in the confusion matrix in Figure 6.12 and Table 6.6. This configuration achieved an accuracy of 98%, with training and testing times of 9.8 and 2.5 seconds, respectively. Additionally, we applied dimensionality reduction using PCA to reduce the feature space from 78 dimensions to 2 components. However, the integration of PCA led to a decline in classification performance, reducing accuracy to 96% and increasing training and testing times to 70.2 and 49.5 seconds, respectively. Thus, our findings suggest that in our context, dimensionality reduction does not effectively improve classification accuracy or reduce computational overhead. The ROC curve of this scenario is shown in Figure 6.13.
- *Experiment 2:* In the second scenario, we employed feature selection using the ExtraTreesClassifier ensemble learning model to assess the importance scores of all features within the CICIDS2017 dataset as shown in Figure 6.5. We identified and removed 30 unimportant features that did not significantly impact the classification process between normal and attack classes. The number of selected features not only enhanced processing time but also improved classification performance. Repeating experiments under the same conditions as the first scenario, with optimal hyper-parameter values of $C = 100$, $\gamma = 0.1$, and $d = 3$, along with the



(a) Linear (b) RBF (c) Polynomial
(d) Sigmoid (e) GridSearchCV (f) GridSearchCV + PCA (g) Proposed IDS

FIGURE 6.12: Confusion matrix of the different SVM models with all features in CICIDS2017.

TABLE 6.6: Results of different SVM models with all features in CICIDS2017.

Model	**Kernel**	**Features**	**Classification Performance Metrics**								**Computational Time (s)**	
			Accuracy		**Precision**	**Recall**	**F1-score**	**FNR**	**FPR**	**AUC**	**Training Time**	**Testing Time**
SVM	Linear	All (78)	99.9188	A	99.9556	99.9008	99.9282	0.0991	0.0578	0.99921	37.5 s	2.7 s
				N	99.8709	99.9422	99.9065					
SVM	RBF	All (78)	99.8907	A	99.9321	99.8747	99.9034	0.1252	0.0884	0.99893	1 m 2.1 s	22 s
				N	99.8369	99.9116	99.8742					
SVM	Polynomial	All (78)	96.75	A	94.5833	99.9817	97.2076	0.0182	7.4608	0.96260	10 m 49.5 s	56.6 s
				N	99.9743	92.5392	96.1132					
SVM	Sigmoid	All (78)	96.8121	A	96.7472	97.6486	97.1958	2.3514	4.2778	0.96685	14 m 17.2 s	20.3 s
				N	96.8985	95.7221	96.3067					
SVM	GridSearchCV, k = RBF	All (78)	99.9631	A	99.9661	99.9687	99.9674	0.0313	0.0442	0.99962	10 s	2.6 s
	$d = 3, C = 100, \gamma = 0.1$			N	99.9592	99.9558	99.9575					
SVM + PCA	GridSearchCV, k = RBF	All (78) - 2 PC	98.1897	A	97.2004	99.6712	98.4203	0.3288	3.740	0.97965	1 m 10.2 s	49.5 s
	$d = 3, C = 100, \gamma = 0.1$			N	99.5569	96.2594	97.8804					
ZW-IDS	GridSearchCV, k = RBF	All (78)	99.9808	A	100.0	99.9558	99.9779	0.0	0.0442	0.99977	9.8 s	2.5 + 2.6 s
	$d = 3, C = 100, \gamma = 0.1$			N	99.9661	100.0	99.983					

(A): *Attack class.*, (N): *Normal class.*

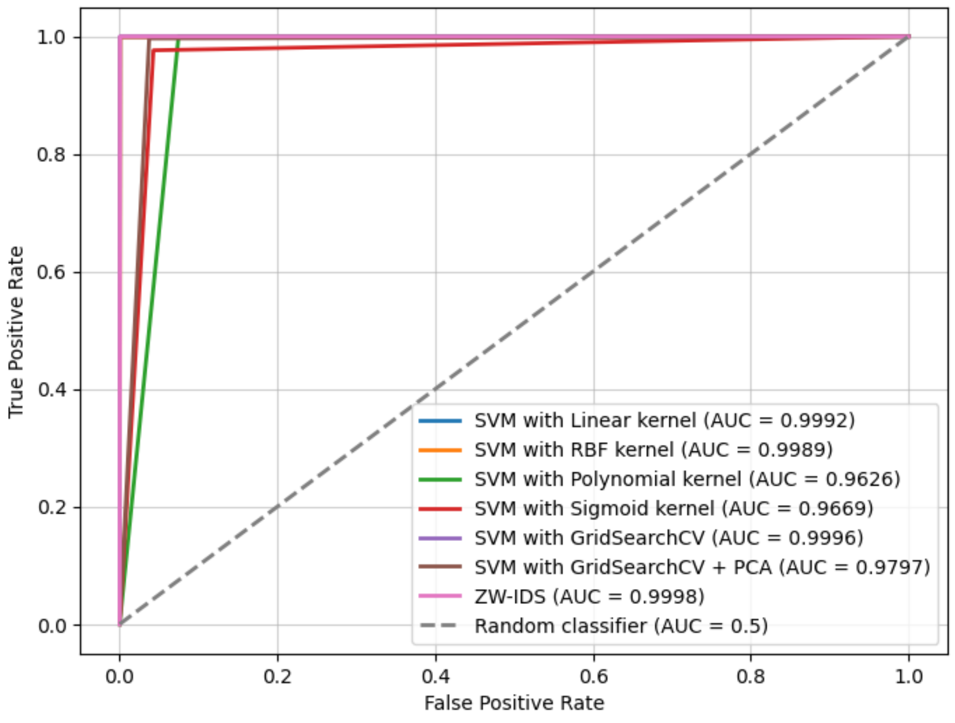
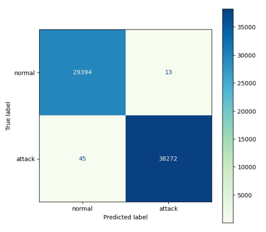


FIGURE 6.13: ROC curve using all features in CICIDS2017.

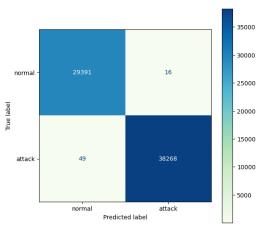
same zero-watermark generation and verification processes, shows better performance results. Applying feature selection with the zero-watermark approach using the RBF kernel resulted in a significant performance improvement, achieving a 99.98% accuracy and a very low FPR of 0.034% and 0.0% for FNR. The classification results of data packets, categorized into two classes –'normal' and 'attack'– using our approach, are shown in the confusion matrix presented in Figure 6.14 and Table 6.7. Moreover, the proposed model achieve better computational efficiency, resulting in 8.1 seconds in training time and 4.8 seconds in testing time (including zero-watermark regeneration and verification time). Figure 6.15 shows the ROC curve of the different SVM models. Thus, integrating feature selection and hyper-parameter tuning with our proposed ZW-IDS effectively mitigates model biasing and overfitting issues, enhancing the effectiveness and speed of IDS in detecting attacks and minimizing misclassification of data packets.

Classification (Layer 2) using Zero-Watermarking and Data Provenance

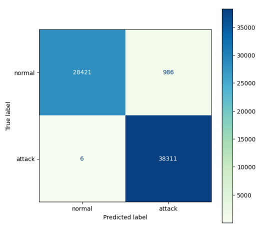
In the second layer classification, we use the best model and parameters of SVM that we already tested over the network dataset which is the RBF kernel. The aim of this layer is to check the misclassified data packets which are actual attack packets but are classified as "normal" ones. This is the Type 2 error in the classification process which is misclassifying a sample that belongs to the attack class as belonging to the normal class. In other words, it is a false negative. In network security applications, the attack class represents any type of an intrusion detected within the devices or network, while the normal class represents regular or expected behavior. Reducing Type 2 errors is critical in such applications, because missing an actual attack can lead to significant consequences overwhelming targeted servers, devices or networks.



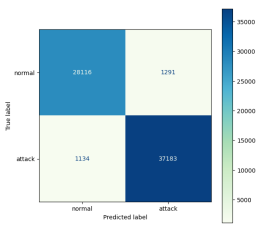
(a) Linear



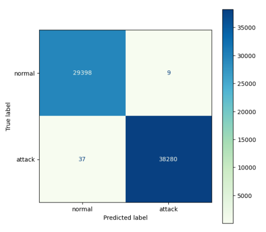
(b) RBF



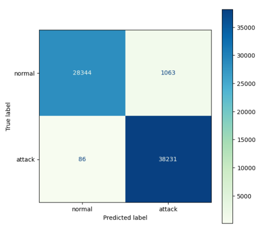
(c) Polynomial



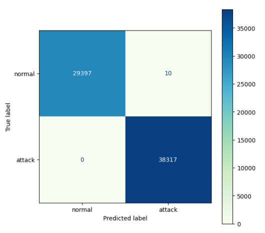
(d) Sigmoid



(e) GridSearchCV



(f) GridSearchCV + PCA



(g) Proposed IDS

FIGURE 6.14: Confusion matrix of the different SVM models with k-important features in CICIDS2017.

TABLE 6.7: Results of different SVM models with k-important features in CICIDS2017.

Model	**Kernel**	**Features**	**Classification Performance Metrics**								**Computational Time (s)**	
			Accuracy		**Precision**	**Recall**	**F1-score**	**FNR**	**FPR**	**AUC**	**Training Time**	**Testing Time**
SVM	Linear	48	99.9144	A	99.966	99.8826	99.9243	0.1174	0.0442	0.99919	32 s	1.9 s
				N	99.8471	99.9558	99.9014					
SVM	RBF	48	99.904	A	99.9582	99.8721	99.9151	0.1278	0.0544	0.99908	52 s	13.9 s
				N	99.8336	99.9456	99.8895					
SVM	Polynomial	48	98.5352	A	97.4909	99.9843	98.7219	0.0156	3.3529	0.98315	9 m 37.9 s	22.6 s
				N	99.9789	96.6471	98.2847					
SVM	Sigmoid	48	96.4193	A	96.6445	97.0405	96.8421	2.9595	4.3901	0.96325	9 m 56.7 s	13.7 s
				N	96.1231	95.6099	95.8658					
SVM	GridSearchCV, k = RBF	48	99.9321	A	99.9765	99.9034	99.94	0.0965	0.0306	0.99936	7.7 s	2.1 s
	d = 3, C = 100, $\gamma = 0.1$			N	99.8743	99.9694	99.9218					
SVM + PCA	GridSearchCV, k = RBF	48 - 2 PC	98.3034	A	97.2948	99.7756	98.5195	0.2244	3.6147	0.98080	1 m 0.5 s	44.6 s
	d = 3, C = 100, $\gamma = 0.1$			N	99.6975	96.3852	98.0134					
ZW-IDS	GridSearchCV, k = RBF	48	99.9852	A	100.0	99.966	99.983	0.0	0.034	0.99982	8.1 s	2.2 + 2.6 s
	d = 3, C = 100, $\gamma = 0.1$			N	99.9739	100.0	99.987					

(A): *Attack class.*, (N): *Normal class.*

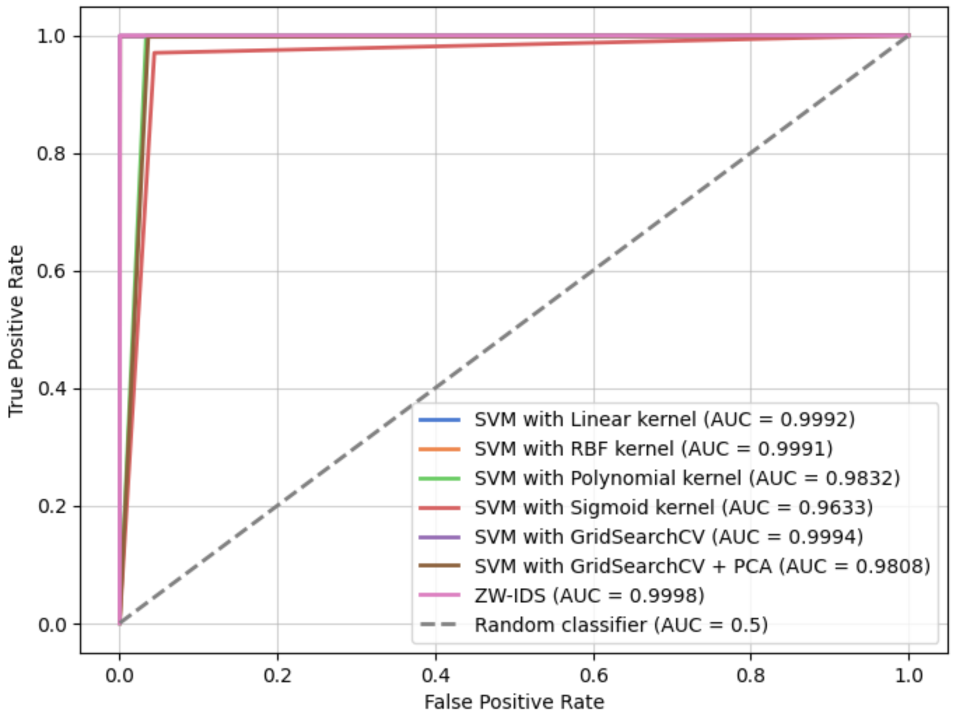


FIGURE 6.15: ROC curve using k-important features in CICIDS2017.

In our model, we generate a zero-watermark at the source device and embed it in the data packet before transmission using extracted provenance information, as shown in Algorithm 8.

After applying SVM classification in the first layer, the classified normal packets are used as an input to the zero-watermark algorithm to detect whether a packet is misclassified as normal or it is an actual normal packet. After receiving the initially flagged normal packet $x'_{(n,k)W_{F_{n,k}}}$, we extract provenance information (w_{ip_s}, w_{ip_d}, w_t, w_{sq}) from data features and re-generate a new zero-watermark $R(W'_{n,k})$ based on Algorithm 9. The sequence number that we generated is based on flow ID, source IP, destination IP and timestamp to uniquely distinguish data packets from source devices. Then, we extract the zero-watermark $W_{F_{n,k}}$ from the data packet $x'_{(n,k)W_{F_{n,k}}}$. If both zero-watermarks are equal then it remains flagged as a normal packet. Otherwise, it is re-flagged as an anomaly. In this scenario, there are two possibilities: either the data packet undergoes modification in its payload or zero-watermark, or the attacker generates zero-watermarks using their own secret key, which won't be authenticated by the IDS. This is a misclassification from the SVM layer 1 model. However, if the received packet lacks a zero-watermark, it falls into one of two categories. First, it may be a control packet, which follows a different procedure. Alternatively, if it does not meet the criteria of a control packet, it is classified as an intrusion. After the second layer of attack detection, we evaluate the performance by applying the same metrics that are used in the SVM model evaluation, as shown in Figure 6.3.

Algorithm 9 : Zero-watermark Re-generation and Re-classification

```
input: $x'_{(n,k)W_{F_{n,k}}}$
output: normal/attack
1: procedure Watermark Re-classification
2:     Receive $\left(x'_{(n,k)W_{F_{n,k}}}\right)$
3:     $R(W'_{n,k}) \leftarrow$ REDO Algorithm 1
4:     $W_{F_{n,k}} \leftarrow$ extract $W_{F_{n,k}}$ from $x'_{(n,k)W_{F_{n,k}}}$
5:     if $W_{F_{n,k}} \notin x'_{(n,k)}$ then
6:         flag $x'_{(n,k)}$ as 'attack'
7:     elif $R(W'_{n,k}) = W_{F_{n,k}}$ then
8:         flag $x'_{(n,k)}$ as 'normal'
9:     else
10:        flag $x'_{(n,k)}$ as 'attack'
11:    end if
```

6.5 Experimental Evaluation with Existing Approaches and Discussion

The ML and zero-watermarking algorithms implementation were done using Scikit-learn[1] library in Python[2] as backend. The code was developed in the Visual Studio™ environment using 16 GB RAM and an Intel™ i7 2.59 GHz processor. The experiments were carried out using CICIDS2017 [27] dataset.

Table 6.8: Performance evaluation with existing approaches.

Approach	Year	Classification Performance Metrics (%)						Computational Time (s)	
		Accuracy	Precision	Recall	F-score	FPR	FNR	Training Time	Testing Time
Tao, Sun, and Sun [23]	2018	98.01	98.39	98.16	98.27	1.89	2.06	49953.13	3.13
Jan et al. [186]	2019	98.03	98.43	97.99	98.20	1.91	2.01	208.90	2.28
Ravi, Chaganti, and Alazab [187]	2022	98.77	97.84	98.74	98.21	1.18	1.34	–	–
Rao and Suresh Babu [188]	2023	98.97	99.06	98.17	99.73	3.93	5.02	–	–
Alghushairy et al. [189]	2024	88.74	–	98.82	–	12.19	1.17	–	**0.007**
Proposed ZW-IDS	2024	**99.98**	**100.0**	**99.96**	**99.97**	**0.034**	**0.0**	**8.1**	4.8

(–): *Performance metric is not reported in the approach.*

To assess the effectiveness of our approach, we conducted a performance comparison with five state-of-the-art methods proposed by Tao, Sun, and Sun [23], Jan et al.

[1]https://scikit-learn.org/
[2]https://python.org/

[186], Ravi, Chaganti, and Alazab [187], Rao and Suresh Babu [188], and Alghushairy et al. [189] in terms of classification performance and computational efficiency, as detailed in Table 6.8. Our approach demonstrates better performance, achieving the highest accuracy of 99.98%, precision of 100%, recall of 99.96%, and F-score of 99.97%. These results shows the effectiveness of introducing a second-layer attack detection based on data provenance (zero-watermark) to augment an ML-based IDS.

While other methods employ various feature selection techniques with SVM, such as genetic algorithms proposed by Tao, Sun, and Sun [23], focusing on specific attributes like packet arrival rate as suggested by Jan et al. [186], or combining SVM with GNB presented by [189], our approach consistently outperforms them. Even when compared to other ML algorithms like Naive Bayes, Logistic Regression, KNN, DT, Random Forest proposed by Ravi, Chaganti, and Alazab [187], as well as DL models like RNN, LSTM, and GRU proposed by [188], our proposed ZW-IDS approach demonstrates better performance. Furthermore, ZW-IDS achieves significantly lower error rates, with an FPR of 0.034% and no instances of Type 2 errors with FNR of 0%, misclassifying attack data packets as normal packets, as shown in the confusion matrix of Figure 6.14.

In terms of computational efficiency, while not all approaches consider this metric, our approach outperforms Tao, Sun, and Sun [23] and Jan et al. [186] in training time, offering significantly lower processing time with 8.1 seconds compared to both approaches with 49953.13 and 208.9 seconds, respectively. Although Alghushairy et al. [189] report a testing time of 0.007 seconds, their approach's classification performance falls short, with an accuracy of 88.74% and an FPR of 12.19%. In ZW-IDS, testing time includes the computational overhead of zero-watermark regeneration and verification procedure of layer 2 classification. This highlights the lightweight, effective, and efficient nature of our two-layer IDS for intrusion detection in the CICIDS2017 dataset. Moreover, we demonstrate that integrating zero-watermarking with data provenance in ML-based IDS enhances performance and facilitates effective intrusion detection while minimizing misclassification errors.

6.5.1 Effectiveness of Two-layered Approach

Combining zero-watermarking-based provenance technique with a ML-based IDS to form a two layer intrusion detection approach provide an effective solution to a number of issues that can not be achieved when applying only one layer of the model. These issues are as follows:

- The analysis of the dataset reveals a significant portion of attack packets as shown in Table 6.2, comprising nearly 57% of the total dataset. Applying SVM followed by the zero-watermarking model means only 43% of the data undergoes reclassification with zero-watermarking. In this case, if we want to only apply the second layer, it requires additional computational time due to the need for regeneration of zero-watermarks for each packet and subsequent comparison across the entire dataset. However, SVM classification layer mitigates this computational burden by classifying the majority of the dataset (57%).

- A critical concern arises if an attacker launches an attack from a compromised device and gains access to the AES encryption secret key used in the zero-watermark generation process. In such a scenario, the attacker can execute attacks with legitimately generated zero-watermarks, thereby bypassing detection at the IDS level. However, with the inclusion of the first layer, a significant number of attack packets can be identified prior to undergoing zero-watermark checks. Removing this initial layer and given knowledge of the secret key, the IDS would fail to detect any attacks.

- Another important consideration is that not all attacks can be effectively detected using zero-watermarking-based provenance data alone. The first layer of the IDS uses network traffic information to infer deviations from normal behavior in packet classification, a capability not inherently present in the zero-watermarking provenance solution and can not be achieved through relying only on the second layer.

The augmentation between both layers enhances the robustness and efficacy of the IDS, enabling effective detection of most attacks while minimizing computational overhead and improving classification performance.

6.6 Summary

This chapter presents a novel approach to enhancing the performance of anomaly-based NIDS by integrating zero-watermarking with data provenance information and a ML-based approach. We introduce two network traffic datasets, NSL-KDD and CICIDS2017, to test the effectiveness of our model. First, we describe the various attack classes within both datasets and their significance in evaluating IDS. Next, we discuss the evaluation metrics employed to analyze IDS performance. Finally, we introduce the proposed model, detailing the experiments conducted to develop our system and achieve the best performance among all SVM models. The proposed approach addresses the limitations of traditional security measures by using ML techniques to differentiate between normal and malicious network activity. Through the implementation of SVM with feature selection in the first layer and data provenance-based zero-watermarking in the second layer, our method aims to reduce false positives and false negatives, improve computational efficiency, and enhance classification accuracy. Evaluation using the NSL-KDD and CICIDS2017 datasets shows the effectiveness of our approach in terms of classification performance and computational overhead. Additionally, a comparative analysis with existing multi-method ML and DL solutions highlights the improvement of our scheme in detecting and mitigating network intrusions using CICIDS2017 dataset. Overall, our proposed model contributes to advancing the field of network security by providing a practical and efficient solution for detecting and preventing cyber-attacks in information systems and computer networks.

Chapter 7

Conclusion and Future Directions

7.1 Summary and Conclusions

In the era of the digital age, the Internet of Things (IoT) has emerged as an intelligent system composed of interconnected physical objects capable of collecting and exchanging data dynamically. Despite its potential in various applications, from healthcare to environmental monitoring, IoT networks face significant security challenges due to their vulnerability to intrusions. Traditional protection techniques often fall short in addressing the specific needs of IoT environments, characterized by constrained resources and unique network architectures. In traditional networks, specific nodes like routers and switches handle the task of forwarding packets to their final destinations, which are directly connected to end systems. However, IoT networks typically use a multi-hop approach. In these networks, nodes not only forward packets but also function as end systems themselves. Consequently, for the sensed data packets to reach their final destination (such as a gateway or central processing unit), they must traverse a path consisting of sensor nodes placed on various light poles. Transmitting these data packets over a shared wireless medium makes the network susceptible to numerous security threats, including data forgery, packet replay, data modification, data insertion, and packet drop attacks. Therefore, it is crucial to develop advanced Intrusion Detection System (IDS) that can effectively secure IoT networks against malicious activities. IDS are essential for identifying various types of attacks and are key tools for the in-depth defense of computer and sensor networks. They monitor network traffic for malicious activities and alert when such activities are detected. IDS are classified into two main types: misuse and anomaly detection systems. Misuse IDS detect intrusions using known attack signatures and system vulnerabilities, but they cannot identify new attacks. Anomaly IDS, on the other hand, detect deviations from normal behavior to identify potential threats.

The integration of Machine Learning (ML) into IDS has significantly enhanced their capabilities. ML-based IDS learn from normal and attack behaviors to identify classes of attacks, although misuse detection techniques, while accurate for known attacks, fail to detect unknown ones. Various ML techniques, including supervised and unsupervised learning, have been implemented to develop anomaly-based IDS. There is a recognized need for new security methods to address the limitations of current ML-based IDS. In this thesis we propose a robust and lightweight system to tackle data integrity issues, securely transmit provenance information, and improve the effectiveness of anomaly-based IDS. The proposed solution introduces a novel approach based on zero-watermarking, utilizing data provenance information as a lightweight methodology.

In terms of contributions, we have started this dissertation with a global overview of the existing ML-based IDS. Then, we went further by surveying watermarking, zero-watermarking, and data provenance security techniques in IoT networks. We identified the need for a robust and lightweight framework for IoT networks to ensure data integrity, secure provenance information, and enhance anomaly-based IDS.

In the first contribution, we review recent IDSs using ML approaches for IoT networks. The study began by addressing the fundamental concepts of IDS and proceeded with a detailed review and classification of recent techniques. Our review identified several open issues and research challenges that need to be addressed to improve IDS schemes, such as detection rates, false positive rates, real-time detection, computational overhead, and energy consumption. We highlighted the necessity for comprehensive research that covers all attack types and recent IoT technologies. Additionally, the study showed the importance of finding optimal placement strategies for ML-based detection to enhance IoT network security while minimizing the risk of increasing the attack surface. We point out the need to explore watermarking algorithms, which offer a lightweight and resource-efficient alternative for IDS implementation.

Our second contribution examine the integration of IoT and data provenance, addressing vulnerabilities and the need for data trustworthiness, quality, traceability, and security. This systematic review provided insights into existing data provenance techniques in IoT networks, their advantages, limitations, and application in ensuring security requirements. We developed a taxonomy to categorize attributes related to the development of data provenance in IoT, providing researchers with better understanding this field. The study also identified significant research gaps and challenges, emphasizing the underexplored nature of data provenance in IoT networks and the need for further research and practical implementations to enhance network security. We also highlighted several areas for improvements in the existing security techniques for data provenance in IoT.

In the third contribution, we focus on addressing data integrity and secure transmission of provenance information in IoT networks. We proposed a zero-watermarking approach, named ZIRCON, which embeds provenance information extracted from data features with data packets, storing these watermarks in a tamper-proof network database. We consider a threat model based on both passive and active attacks, addressing single-hop and multi-hop scenarios. We present algorithms and modeling for each scenario, incorporating procedures for watermark generation, embedding, storage, and verification. Additionally, we account for various verification and attack cases, detailing the protocol applicable in each scenario. The security capabilities of ZIRCON were validated through formal security analysis, covering 10 attack scenarios. We further demonstrated and validated our approach via numerical simulations using MATLAB, applying various cryptographic techniques and SHA functions. The results indicate that AES with a 128-bit secret key and the SHA-2 hash function provide the best performance for our proposed model. We also propose a protocol to manage internal datagrams, which reduces the overhead associated with generating, embedding, and verifying processes for all data packets in the network. Our findings demonstrated that the proposed scheme is lightweight, computationally efficient, and consumes less energy compared to existing solutions.

Finally, the fourth contribution present a novel approach to enhancing the performance of anomaly-based Network Intrusion Detection System (NIDS) by integrating zero-watermarking using data provenance information and ML-based techniques. Our approach utilized Support Vector Machine (SVM) with feature selection in the first layer and data provenance-based zero-watermarking in the second layer to reduce false positives and false negatives, improve computational efficiency, and enhance classification accuracy. Evaluation using the NSL-KDD and CICIDS2017 datasets demonstrated the effectiveness of our approach in terms of classification performance and computational overhead. We perform extensive experiments to determine the optimal SVM model and parameters in the first layer. This layer involves three stages: first, data pre-processing, which includes calculating ranges for numeric features, extracting categorical values, encoding categorical values and labels, and standardizing the data. Second, zero-watermark generation and embedding at the source node, where generated watermarks from provenance information are embedded in data packets. Third, feature extraction and selection using an ensemble extra trees classifier to identify the most important features. The packets then undergo classification in two layers: initially, we classify network traffic using the SVM model. Subsequently, the classified normal packets undergo a second layer of classification using the zero-watermark model. Our approach achieves very high accuracy (99.98%), precision (100%), recall (99.96%), and F-score (99.97%). It significantly reduces false error rates, with an FPR of 0.034% and no instances of Type 2 errors with FNR of 0%, misclassifying attack data packets as normal packets. Additionally, the proposed model is computationally efficient, requiring only 8.1 seconds for training and 4.8 seconds for testing and watermark verification. These results shows the effectiveness of introducing a second-layer attack detection based on data provenance (zero-watermark) to augment an ML-based IDS. Comparative analysis with existing multi-method ML and Deep Learning (DL) solutions highlighted the improvements achieved by our scheme in detecting and mitigating network intrusions.

Research in IDS for IoT has still several actions to be done. Hence, there are wide opportunities for future research perspectives to extend and improve the existing field knowledge. In the next section, we point out several promising directions.

7.2 Future Directions

In terms of perspectives for future research, as a result of the work initiated in this thesis, there are several directions for improvement. This section discusses the limitations related to the existing security approaches for data provenance and ML-based IDS in IoT networks. This creates a great opportunity for researchers to find solutions to address such limitations and reduce the number of open problems.

- *Addressing Scalability and Real-Time Detection in IoT Networks:* one major limitation of current IDS in IoT networks is their scalability and ability to perform real-time detection. Most existing solutions struggle with high detection rates and low false positives when scaled to large networks. Future research should focus on developing IDS frameworks that can handle the vast number of devices and high data throughput characteristic of IoT environments without compromising on performance. This includes optimizing algorithms to function efficiently within the resource-constrained nature of IoT devices. Research should also explore the integration of more sophisticated ML techniques such as federated learning, which allows for model training across decentralized devices while maintaining data privacy.

- *Exploring New Attack Vectors and Defense Mechanisms:* current IDS solutions may not adequately cover all potential attack vectors, particularly as IoT technology evolves and new threats emerge. Research should focus on identifying and mitigating novel attack strategies that could exploit vulnerabilities in IoT networks. This includes studying the impact of attacks that target specific IoT protocols and developing modified defense mechanisms. For instance, investigating the security implications of emerging IoT communication standards like 5G and LoRaWAN could reveal new vulnerabilities. Additionally, adaptive IDS that can evolve and learn from new types of attacks in real-time should be developed to ensure ongoing protection against an ever-changing threat models.

- *Integrating IDS with IoT-Specific Protocols and Architectures:* the unique network architectures and communication protocols used in IoT networks pose challenges for traditional IDS. Future work should aim to integrate IDS seamlessly with these specific protocols and architectures, such as MQTT, CoAP, and Zigbee. This involves designing IDS algorithms to better understand the traffic patterns and behaviors associated with these protocols. Moreover, the development of IDS that can operate efficiently in multi-hop and mesh network topologies commonly used in IoT can enhance security. Research should also consider the implications of network topology changes, such as node mobility and varying network densities, on IDS performance and adapt accordingly.

- *Advancing the Integration of Blockchain with IDS for IoT Security:* Blockchain technology has the potential to significantly enhance the security and reliability of IDS in IoT networks by providing a decentralized and tamper-proof mechanism for data verification and storage. However, the integration of blockchain with IDS poses challenges such as scalability, latency, and resource consumption. For future work, we could focus on developing lightweight blockchain frameworks that can be effectively integrated with IDS without imposing excessive computational and energy overheads on IoT devices. Exploring sidechains, sharding, and other scalability solutions within blockchain can help address these challenges. Additionally, the application of smart contracts to automate security responses and incident management in real-time should be investigated to enhance the overall security posture of IoT networks.

- *Advancing Watermarking Techniques in IoT Networks:* watermarking, particularly in the context of IoT, remains an underexplored area that offers numerous research opportunities. One significant limitation of current watermarking techniques is their limited robustness and adaptability to the diverse and dynamic nature of IoT environments. Traditional watermarking methods, primarily designed for static digital content like images and videos, do not adequately address the unique challenges posed by IoT networks, such as continuous data flow, heterogeneous device capabilities, and varying network conditions. A future perspective shall focus on developing advanced watermarking algorithms directed specifically for IoT applications. These algorithms need to be lightweight to accommodate the constrained resources of IoT devices while being robust enough to withstand various types of attacks and network anomalies.

- *Efficient Query and Provenance Support Mechanisms* The efficient querying and management of provenance data in IoT networks present significant challenges due to the massive and continuous data generation by numerous interconnected devices. Future work should focus on developing scalable and efficient query mechanisms designed to the unique characteristics of IoT environments. These mechanisms need to handle the dynamic nature of IoT networks, where devices frequently join and leave, and data is constantly updated and transmitted across distributed systems. Innovative solutions might include advanced indexing techniques that facilitate rapid query responses. Additionally, employing ML algorithms to predict query patterns and pre-fetch relevant provenance data could significantly enhance query efficiency. Developing provenance support systems that integrate seamlessly with existing IoT infrastructures is also important. These systems should be capable of managing the provenance of data across heterogeneous devices and networks, ensuring that provenance information is consistently and accurately recorded. Leveraging technologies like edge computing can help by processing and storing provenance data closer to the data sources, thus reducing latency and improving response times for provenance queries.

- *Exploring Unsupervised Learning for Zero-Watermarking Approach in IoT Networks:* while our thesis has focused on utilizing supervised learning methods for enhancing IoT network security through a zero-watermarking approach, there is substantial potential in exploring unsupervised learning techniques. The integration of unsupervised learning with zero-watermarking could provide significant advantages in terms of anomaly detection, particularly for identifying previously unseen attack patterns without the need for labeled training data. One potential future research direction involves developing an unsupervised anomaly detection framework that leverages zero-watermarking. This framework could implement clustering techniques, such as k-means, DBSCAN, or hierarchical clustering, to group similar data points and identify anomalies as outliers. Additionally, autoencoders and other DL models could be utilized to learn normal network traffic patterns and detect deviations indicative of potential intrusions. Benchmarking and performance evaluation of unsupervised learning-based zero-watermarking approaches are essential to assess their effectiveness. Conducting comparative analyses of unsupervised learning models with existing supervised learning models in terms of detection accuracy, false positive rates, and computational efficiency will provide valuable insights. Additionally, deploying the proposed models in real-world IoT environments will help evaluate their practical effectiveness and robustness against various attack scenarios.

Appendix A

Data Provenance Comparison Tables

Table A.1: Comparative analysis of selected studies.

Reference	Year	Provenance Encoding Method	Provenance Storage Method	Application	Security Analysis	Pros	Cons
Cryptography-based							
Lim, Moon, and Bertino [105]	2010	Trust scores	N/A	Sensor Network	✗	Practical solution for trustworthiness assessment.	The method is based on the principle that the more trustworthy data a source provides, the more trusted the source is considered. Many attacks can deceive the system and overcome the trustworthiness of data.
Wang, Hussain, and Bertino [110]	2016	Path Index and MAC	Distributed database	WSN	✓	Embeds path index instead of data path, which reduces the size of the embedded information in each packet. Each packet path is stored at the forwarding nodes in the network.	Provenance information is only accessible at the BS and cannot be verified at each stage of the path, provenance data is not securely transmitted and stored.
Chia, Keoh, and Tang [111]	2017	Shamir secret sharing and threshold cryptography	Smart meter	Home Energy Monitoring Networks	✗	Achieves source identity authenticity, location authenticity, data consistency and source data authenticity.	The proposed solution does not consider the storage of provenance information, multi-hop architecture and provenance encoding.
Suhail et al. [113]	2018	Hash chain	In-packet	Sensor network	✗	Keeps track of data packets using a chain of provenance records that store that hash of traversed node ID.	Integrity is verified at the destination. Provenance size grows very fast as the number of forwarding nodes increase, link overhead due size increase in forwarded packets.
Xu, Zhang, and Wang [116]	2019	Path index differences	Network nodes	WSN	✗	High provenance compression rate. Whenever the number of hops increases the provenance size nearly remains at a constant level.	Provenance information is based on a few number of data features. Dictionaries at each node increase in size as the network scale and the data packets increase.
Suhail et al. [119]	2020	In-packet embedding	Routing table	RPL-based IoT network	✓	Constant provenance size, used energy consumption, enhanced provenance generation time.	Considers robustness against only three attacks. Provenance information is node ID and sequence number.
Liu and Wu [65]	2020	Common substring matching	Distributed database	Multihop IoT network	✗	Malicious node identification. High provenance decoding accuracy. Stable provenance length after all the path have been traversed	Many provenance fields that increase the size compared to other techniques. Lack of security analysis against different types of attacks.
Tang and Keoh [120]	2020	MAC	in-packet	HAN with smart metering	✓	Use of a symmetric key approach to improve efficiency over asymmetric key-based approaches.	Needs to integrate a third sender to the system to achieve location verification. Solution for single-hop data transfer scenario only.
Porkodi and Kesavaraja [121]	2021	Hash function and symmetric encryption	N/A	IoT network	✓	Multiple levels of authorities. Lightweight key management.	Verifying data origin is not satisfied. Needs for provenance storage. Provenance information is not sufficient to establish trustworthiness in the system.
Xu, Bertino, and Wang [122]	2022	Path index and Packet path hash value	Distributed node database	Zigbee WSN	✗	Reduces the negative impact of network topology changes. Maintains high provenance compression and keeps the provenance size constant	Increase in provenance size as network becomes larger in the PIDP scheme. Lacks analysis to different security threats.
Xu and Wang [123]	2022	Dictionary-based provenance scheme/ Hash functions	Network nodes	ZigBee sensor network	✗	WSN topology graph is presented as a series of different granularity topology graphs. Encoding provenance on high granularity levels which skips provenance updating at some nodes. Decrease in the provenance updating latency.	The scheme yields high cost in terms of computation and storage in sensor nodes, which are resource constrained and require lightweight schemes.
Bloom Filters							
Sultana et al. [33]	2015	in-packet Bloom filters	in-packet chain	WSN	✓	Simple encoding scheme. Reduces the size of provenance length. Scalable for a high number of nodes.	Provenance information is not sufficient (only node ID). Sends the complete data path within the packet.
Siddiqui, Rahman, and Nadeem [124]	2019	Bloom Filters with cryptographic mechanisms	in-packet	N/A	✗	The computation of the partial digital signature is carried out by the IoT node, while the more resource-intensive calculations are offloaded and performed by the edge node. Enhanced storage capabilities.	Fast increment of provenance size. Complete provenance is appended to the data packet.
Harshan et al. [125]	2020	Bloom Filter	In-packet	Raspberry Pi network	✗	At most half the nodes in the path modify the provenance. Reduction in the delay on packets.	Storage overhead in large-scale networks as the number of hops increases.

Reference	**Year**	**Provenance Encoding Method**	**Provenance Storage Method**	**Application**	**Security Analysis**	**Pros**	**Cons**
				Fingerprints			
Ali et al. [129]	2014	Link Fingerprints	Device Database	Bodyworn Sensors	✗	Use of quantization to distill the RSSI data into a much smaller size to overcome storage challenges in memory constrained sensor devices.	Considers only single-hop scenarios. Data provenance operates on a per-session basis and is not validated for each packet sent.
Alam and Fahmy [130]	2014	Prime multiplication and Rabin Fingerprints	In-packet buffer	IoT sensor network with TelosB motes	✗	The use of small number of bits to encode higher number of node IDs and requires fewer packets to construct network-wide provenance.	Provenance information is based on only node IDs, which is not sufficient to encounter many attacks.
Kamal and Tariq [131]	2018	Link Fingerprint	in-packet and server	Multihop IoT network usinf MICAz motes	✗	The provenance relies on the next hop node using the RSSI, which is related to any secrets stored in the nodes.	The higher the value of correlation coefficient, the higher the percentage of the secured data transfers that can be deceived by an attacker when compromising a group of colliding nodes.
				Blockchain-based			
Baracaldo et al. [132]	2017	Keyless Signature Infrastructure Module (KSI)	Blockchain	N/A	✗	Provenance data that is secured can only be accessed by authorized users. Lightweight and scalable architecture for IoT applications.	Data points generated from IoT devices/sensors are not secured when communicated to the gateway and then to the policy engine and KSI before storing the provenance data as a transaction in the blockchain.
Zeng et al. [135]	2018	Ethereum blockchain using edge computing nodes	blockchain database	Raspberry Pi Nodes and micaz motes	✗	No provenance compression is needed. Secure provenance storage through blockchain database.	Each data packet requires a transaction for updating provenance information. Large number of generated data packets from sensor nodes, requiring a complex and costly method to store and query each provenance record for the data packets from the blockchain.
Javaid, Aman, and Sikdar [138]	2018	PUFs	Blockchain	Linux working environment	✓	Prone to single point of failure due to the decentralized architecture. Smart contracts enable a safe and secure mechanism for the transmission, authentication and storage of requests.	Each PUF Challenge Response Pair (CRP) and the address of IoT device is stored by the smart contract. The huge number of data generated makes it complex to store this amount of data using a blockchain.
Sigwart et al. [139]	2020	Data points	Blockchain-based	General	✓	A general framework for different IoT applications. Layered architecture of smart contracts.	No consideration for the secure transmission of provenance records. New architecture is needed in the IoT platform.
Liu et al. [142]	2020	Hash function and K-Hop Ancestor	Blockchain	IoT application with blockchain network	✓	Examines existing security concerns in the distributed IoT network architecture. Uses blockchain as the fundamental architecture for storing and retrieving cross-domain provenance data by using its decentralization and immutability.	Querying provenance information needs to be optimized due to the cost of retrieving on/off blockchain storage.
Siddiqui et al. [143]	2020	Ciphertext-Policy Attribute based Encryption (CP-ABE)	Blockchain and centralized database	Cloud based IoT	✗	By applying partial signatures, it is possible to offload the blockchain method and associated overhead from the IoT node to the edge nodes.	Each provenance record must be stored twice (requiring additional storage and communication overhead), with the block being saved in the provenance database and the provenance data being published on the blockchain network by a provenance auditor.
Yin and Fu [144]	2022	Smart contract	Blockchain	Ethereum network	✗	Users can not access provenance information without access permission from data owner. Each time an operation is performed the authority is checked.	Provenance information needs to be communicated securely with the blockchain storage. Each operation need to be set as a transaction to be stored within the blockchain, which makes it complex.
Sun, Tang, and Du [145]	2022	Hash function and Homomorphic Signature	Blockchain	Multi-layer IoT applications	✗	Detects abnormal data operations using the provenance graph with integrity verification using a signature and a hash function on each IoT node.	Provenance information is not securely communicated with the edge node or gateway.
Chenli et al. [147]	2022	Directed graph and Chameleon Hash	Blockchain	Decentralized data sharing applications	✓	Performs both forward and backward tracking. Uses networked blockchain instead of single blockchain for storing provenance records.	Costly in terms of computation and storage for resource limited devices such as sensors and devices in IoT networks.
				Watermarking			
Sultana, Bertino, and Shehab [154]	2011	Watermarking based	inter packet delays	sensor network	✗	High detection accuracy, energy efficiency	Drastic increase in provenance size as number of nodes increases
				Physical Unclonable Functions			
Aman, Basheer, and Sikdar [156]	2019	PUFs and RSSI	Device memory and server	Indoor laboratory IoT environment	✓	Unclonability and robustness against physical attacks through avoiding the need to store secret keys.	Provenance information is based only on device's pseudonym identity and RSSI, which is not enough to obtain any attack attempt in packet drop, replay, and modification.
Aman, Basheer, and Sikdar [157]	2021	PUFs, wireless fingerprints	server database	IoT network with MICA-Z motes	✓	IoT devices do not store secrets in their memory. Privacy preservation. Resilience against Ephemeral Secret Leakage (ESL) attacks.	Requires computation of many session keys for each node. Lacks security analysis against different type of attacks.
Hamadeh and Tyagi [158]	2021	PUFs	Network nodes and server	FPGA Altera Cyclone	✗	Source identity authenticity through PUF. IoT node is able to send anonymously data to the server.	The system does not consider the traceability of data packets along the data path. Multi-hop model with the presence of forwarding devices is not considered. Selecting and computing a secret key in each round results in computational overhead.

Reference	**Year**	**Provenance Encoding Method**	**Provenance Storage Method**	**Application**	**Security Analysis**	**Pros**	**Cons**
Graph-based							
Chang et al. [159]	2022	Event-flow graphs	Provenance server	IoT smart apps	✗	Observes provenance without utilizing a sophisticated query language by just selecting the appropriate nodes on an event flow graph.	As the number of events and actions increases, the provenance data becomes large in scale over time, provenance data is not securely stored in the provenance server.
Jaigirdar et al. [160]	2023	Provenance graphs	Cloud	IoT Health applications	✗	The model includes security-aware properties at every step of data transmission. Provides the status of each device as data processing mechanism, software running and communication channels properties	Single point of failure where provenance information is stored in the cloud. All the devices/sensors and users need to forward and retrieve this information from it, including auditor, doctor, user, and gateway.
Al-Rakhami and Al-Mashari [163]	2022	Directed Acyclic Graph	Cloud server	Industrial Internet of Things (IIoT)	✗	Scalability, affordability, and quantum robustness are all associated with the adoption of IOTA's distributed ledger technology (DLT).	MQTT protocol is used to store and manage the majority of the data that is gathered by our system, but because it does not execute or impose data encryption, it is not completely safe against tampering.
Data Sanization							
Lomotey et al. [165]	2019	Data sanitization	Local database	Sensor network	✗	Hides sensitive data that users do not want to share with others by tagging attributes in the provenance information.	Provenance information is not secure. Storing and querying provenance information is not defined. Integrity is not ensured.
Lexial Chaining							
Lomotey, Pry, and Chai [167]	2018	Associative rules and statistical lexical chaining	Centralized database (CouchDB)	Devices with Bluetooth in a machine-to-machine (M2M) scenario	✗	During machine-to-machine communication, it is possible to track and categorize the communication routes taken by various IoT devices as well as their previous connections.	Lexical chains are not securely communicated between the different entities and the database.
Path Difference							
Gao et al. [168]	2013	Path difference and speculation	In-packet/ PC database	WSN	✗	Lightweight approach that does not require complex computation at sensor nodes. In case of inability to record path difference, a path speculation method can reconstruct the routing path.	The path field container is limited to a number of bits that cannot be exceeded. When the path difference is large and exceeds this limit, the path cannot be recorded completely.
Logging-based							
Sadineni, Pilli, and Battula [170]	2023	Provenance logs and network traffic	Centralized database	Link-Layer Forensics in IoT	✗	Detects network and link layer attacks in IoT networks. Detects stealthy attacks.	Provenance information is not securely transmitted through transmission channel. The stored provenance information can be altered. Multi-hop provenance path construction is not studied. Due to rapid increase in provenance information, there is a storage overhead at the database.
Frameworks using storing methods							
Alkhalil, Ramadan, and Ahmad [171]	2019	Trust model based on fuzzy logic	Network Nodes	WSN	✗	Node's trust is based on availability, neighboring nodes evaluation, and message drop rate.	The study needs to consider security issues in-terms of integrity and secure transmission of data. Is not clear how provenance information is encoded and stored.
Wang et al. [172]	2018	Provenance as sources and sinks	Centralized database	Samsung SmartThings IoT Platform	✗	Complete platform for provenance tracking in IoT applications. Records provenance information by a provenance collector directly from IoT devices.	No security mechanism is applied to secure the transmission of provenance information. Multi-hop tracking is not considered.
Elkhodr and Alsinglawi [174]	2020	Store device status	Local database	IoT Management Platform	✗	Stores the provenance information as device status in a centralized database including the visiting and returning devices.	Provenance information is not secured. Multi-hop model is not taken into consideration. Security requirements in terms of data and provenance are not studied.

TABLE A.2: Provenance category and evaluation methodology used in the selected security techniques.

Reference	**Provenance Category**	**Implementation**	**Simulation**
Lim, Moon, and Bertino [105]	Cryptography-based	✗	Java
Wang, Hussain, and Bertino [110]	Cryptography-based	TelosB motes	TinyOS 2.1.2 TOSSIM
Chia, Keoh, and Tang [111]	Cryptography-based	Raspberry Pi 3 and DLink DSP-W215 Wi-Fi Smart Plug	Java
Suhail et al. [113]	Cryptography-based	✗	Contiki simulator, Cooja [237]
Xu, Zhang, and Wang [116]	Cryptography-based	Testbed with Zigbee sensor nodes	TinyOS 2.1.2/TOSSIM, PowerTOSSIMz
Suhail et al. [119]	Cryptography-based	✗	Contiki simulator, Cooja
Liu and Wu [65]	Cryptography-based	✗	Python
Tang and Keoh [120]	Cryptography-based	✗	✗
Porkodi and Kesavaraja [121]	Cryptography-based	✗	N/A
Xu, Bertino, and Wang [122]	Cryptography-based	Zigbee sensor nodes with TI CC2530 microcontroller	TOSSIM/TinyOS 2.1.2 and PowerTOSSIMz [238]
Xu and Wang [123]	Cryptography-based	ZigBee sensor/TI CC2530 microcontroller	TinyOS 2.1.2/TOSSIM, PowerTOSSIMz
Sultana et al. [33]	Bloom Filter	✗	TinyOS simulator
Siddiqui, Rahman, and Nadeem [124]	Bloom Filter	✗	Java
Harshan et al. [125]	Bloom Filters	Raspberry Pi 3+ and a Digi XBee S2C	✗
Ali et al. [129]	Link Fingerprints	MICAz motes, running TinyOS	✗
Alam and Fahmy [130]	Rabin Fingerprints	TelosB motes	TOSSIM
Kamal and Tariq [131]	Link Fingerprints	MICAz motes	✗
Baracaldo et al. [132]	Blockchain-based	✗	✗
Zeng et al. [135]	Blockchain-based	MICAz motes	TOSSIM/TinyOS 2.1.2
Javaid, Aman, and Sikdar [138]	PUFs and Blockchain	✗	Solidity IDE, Remix
Sigwart et al. [139]	Blockchain-based	✗	Ethereum networks Rinkeby and Ropsten
Liu et al. [142]	Blockchain-based	Python interface/Pinocchio [239] and libsnark interface [240] with Etheruem test network	✗
Siddiqui et al. [143]	Blockchain-based	✗	Hyperledger Fabric [241]/ Byzantine fault-tolerant state machine replication library - Java (BFT-SMaRt) [242]
Yin and Fu [144]	Blockchain-based	✗	Ethereum network test/ Ubuntu
Sun, Tang, and Du [145]	Blockchain-based	✗	Public IoT database MMASH [243]
Chenli et al. [147]	Blockchain-based	✗	Go language and Hyperledger Fabric v2.0
Sultana, Bertino, and Shehab [154]	Watermarking	✗	TinyOS simulator
Aman, Basheer, and Sikdar [156]	PUFs and Link Fingerprints	MICAz motes/ MATLAB	MICA 2 mote platform/ AVRORA
Aman, Basheer, and Sikdar [157]	PUFs	MICAz motes, CC2420 transceiver	✗
Hamadeh and Tyagi [158]	PUFs	FPGA Altera Cyclon	Verilog (HDL), ModelSim XE
Chang et al. [159]	Event-flow graphs	✗	Gentle Wake-Up, and Smart Security apps using SmartThings IDE [244, 245]
Jaigirdar et al. [160]	Provenance graphs	✗	Flask module in python/ Neo4j
Al-Rakhami and Al-Mashari [163]	Directed Acyclic Graph	✗	Eclipse Paho [1] and Python
Lomotey et al. [165]	Data Sanitization	✗	✗
Lomotey, Pry, and Chai [167]	Lexical chaining techniques	Bluetooth devices/Amazon EC2 platform	✗
Gao et al. [168]	Path difference	✗	TinyOS 2.1.1 with CTP [246]/ TOSSIM
Sadineni, Pilli, and Battula [170]	Logging-based	✗	Contiki simulator, Cooja
Alkhalil, Ramadan, and Ahmad [171]	Fuzzy logic	✗	Java program
Wang et al. [172]	Centralized Database	✗	Samsung SmartThings IDE [247]
Elkhodr and Alsinglawi [174]	Management platform	✗	✗

Table A.3: Performance metrics used in the selected studies.

Reference	Prov. Length	Energy Consumption	Data Packet Size	Link-Loss Rate	Detection Rate	False Positive Rate	False Negative Rate	Computation time
Lim, Moon, and Bertino [105]	✗	✗	✗	✗	✗	✗	✗	✓
Wang, Hussain, and Bertino [110]	✓	✓	✗	✓	✓	✗	✗	✓
Chia, Keoh, and Tang [111]	✗	✗	✗	✗	✗	✗	✗	✓
Suhail et al. [113]	✗	✓	✗	✗	✗	✗	✗	✗
Xu, Zhang, and Wang [116]	✓	✓	✓	✗	✗	✗	✗	✗
Suhail et al. [119]	✓	✓	✗	✗	✓	✗	✗	✓
Liu and Wu [65]	✓	✗	✗	✗	✓	✓	✓	✗
Tang and Keoh [120]	✗	✗	✗	✗	✗	✗	✗	✗
Porkodi and Kesavaraja [121]	✗	✗	✗	✗	✗	✗	✗	✓
Xu, Bertino, and Wang [122]	✓	✓	✓	✗	✗	✓	✗	✗
XXu and Wang [123]	✓	✓	✓	✗	✓	✗	✗	✓
Sultana et al. [33]	✓	✓	✗	✓	✓	✓	✗	✗
Siddiqui, Rahman, and Nadeem [124]	✗	✗	✗	✗	✗	✗	✗	✓
Harshan et al. [125]	✓	✗	✗	✗	✓	✗	✗	✓
Ali et al. [129]	✗	✓	✓	✗	✗	✗	✗	✓
Alam and Fahmy [130]	✓	✓	✓	✗	✗	✗	✗	✗
Kamal and Tariq [131]	✗	✓	✗	✗	✗	✗	✗	✓
Baracaldo et al. [132]	✗	✗	✗	✗	✗	✗	✗	✗
Zeng et al. [135]	✓	✗	✓	✗	✗	✗	✗	✗
Javaid, Aman, and Sikdar [138]	✗	✗	✗	✗	✗	✗	✗	✗
Sigwart et al. [139]	✗	✗	✗	✗	✗	✗	✗	✓
Liu et al. [142]	✓	✗	✓	✗	✗	✗	✗	✓
Siddiqui et al. [143]	✗	✗	✗	✗	✗	✗	✗	✓
Yin and Fu [144]	✗	✗	✗	✗	✗	✗	✗	✓
Sun, Tang, and Du [145]	✗	✗	✗	✗	✗	✗	✗	✓
Chenli et al. [147]	✗	✗	✗	✗	✗	✗	✗	✓
Sultana, Bertino, and Shehab [154]	✗	✗	✓	✗	✓	✓	✓	✗
Aman, Basheer, and Sikdar [156]	✓	✓	✓	✗	✗	✗	✗	✓
Aman, Basheer, and Sikdar [157]	✓	✓	✗	✗	✗	✗	✗	✗
Hamadeh and Tyagi [158]	✗	✗	✗	✗	✗	✗	✗	✓
Chang et al. [159]	✗	✗	✗	✗	✗	✗	✗	✓
Jaigirdar et al. [160]	✓	✗	✗	✗	✗	✗	✗	✓
Al-Rakhami and Al-Mashari [163]	✗	✗	✓	✗	✗	✗	✗	✓
Lomotey et al. [165]	✗	✗	✗	✗	✗	✗	✗	✗
Lomotey, Pry, and Chai [167]	✗	✗	✗	✗	✓	✓	✓	✓
Gao et al. [168]	✓	✗	✓	✓	✗	✓	✗	✓
Sadineni, Pilli, and Battula [170]	✓	✗	✓	✗	✗	✗	✗	✗
Alkhalil, Ramadan, and Ahmad [171]	✗	✗	✗	✗	✗	✗	✗	✓
Wang et al. [172]	✓	✗	✓	✓	✗	✗	✗	✓
Elkhodr and Alsinglawi [174]	✗	✗	✗	✗	✗	✗	✗	✗

TABLE A.4: Security requirements in the selected studies.

Reference	**Security Requirements**						
	Data Integrity	***Confidentiality***	***Availability***	***Privacy***	***Freshness***	***Non-repudiation***	***Unforgeability***
Lim, Moon, and Bertino [105]	N/A	N/A	YES	N/A	YES	N/A	N/A
Wang, Hussain, and Bertino [110]	YES	YES	YES	N/A	YES	YES	N/A
Chia, Keoh, and Tang [111]	YES	N/A	N/A	N/A	N/A	YES	YES
Suhail et al. [113]	YES	N/A	N/A	N/A	YES	YES	YES
Xu, Zhang, and Wang [116]	YES	N/A	N/A	N/A	N/A	YES	N/A
Suhail et al. [119]	N/A	N/A	YES	N/A	YES	YES	N/A
Liu and Wu [65]	YES	N/A	YES	N/A	YES	N/A	YES
Tang and Keoh [120]	YES	YES	N/A	N/A	N/A	N/A	YES
Porkodi and Kesavaraja [121]	YES	YES	YES	N/A	N/A	N/A	YES
Xu, Bertino, and Wang [122]	YES	YES	YES	N/A	N/A	YES	YES
Xu and Wang [123]	YES	N/A	YES	N/A	N/A	YES	YES
Sultana et al. [33]	YES	YES	YES	N/A	YES	N/A	N/A
Siddiqui, Rahman, and Nadeem [124]	YES	YES	YES	N/A	N/A	YES	N/A
Harshan et al. [125]	YES	N/A	YES	N/A	N/A	YES	YES
Ali et al. [129]	YES	YES	N/A	N/A	YES	YES	YES
Alam and Fahmy [130]	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Kamal and Tariq [131]	YES	N/A	YES	N/A	YES	N/A	N/A
Baracaldo et al. [132]	YES	YES	N/A	YES	N/A	YES	N/A
Zeng et al. [135]	YES	YES	N/A	N/A	YES	YES	YES
Javaid, Aman, and Sikdar [138]	YES	N/A	N/A	N/A	YES	YES	N/A
Sigwart et al. [139]	YES	N/A	YES	N/A	N/A	YES	N/A
Liu et al. [142]	YES	N/A	N/A	N/A	YES	YES	YES
Siddiqui et al. [143]	YES	YES	N/A	N/A	YES	YES	N/A
Yin and Fu [144]	YES	YES	N/A	YES	N/A	YES	N/A
Sun, Tang, and Du [145]	YES	YES	N/A	N/A	N/A	N/A	YES
Chenli et al. [147]	YES	YES	N/A	N/A	N/A	N/A	N/A
Sultana, Bertino, and Shehab [154]	YES	YES	YES	N/A	YES	N/A	N/A
Aman, Basheer, and Sikdar [156]	YES	YES	N/A	YES	YES	YES	N/A
Aman, Basheer, and Sikdar [157]	YES	N/A	N/A	N/A	YES	N/A	N/A
Hamadeh and Tyagi [158]	YES	N/A	YES	YES	YES	YES	N/A
Chang et al. [159]	N/A	N/A	N/A	N/A	YES	N/A	N/A
Jaigirdar et al. [160]	YES	N/A	YES	N/A	N/A	YES	YES
Al-Rakhami and Al-Mashari [163]	YES	YES	YES	N/A	N/A	N/A	YES
Lomotey et al. [165]	N/A	YES	N/A	YES	N/A	YES	N/A
Lomotey, Pry, and Chai [167]	YES	YES	YES	N/A	N/A	YES	N/A
Gao et al. [168]	N/A	N/A	N/A	N/A	YES	YES	N/A
Sadineni, Pilli, and Battula [170]	N/A	N/A	YES	N/A	N/A	N/A	YES
Alkhalil, Ramadan, and Ahmad [171]	N/A	N/A	YES	N/A	N/A	YES	N/A
Wang et al. [172]	YES	N/A	YES	YES	N/A	YES	N/A
Elkhodr and Alsinglawi [174]	N/A	N/A	N/A	N/A	YES	N/A	N/A

TABLE A.5: Attacks studied in the selected security techniques.

Reference	**Data Attacks**					**Provenance Attacks**					
	Drop	***Replay***	***Forgery***	***Modification***	***Eavesdrop***	***Record drop***	***Replay***	***Forging***	***Modification***	***Chain tamper-ing***	***Inference***
Lim, Moon, and Bertino [105]	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Wang, Hussain, and Bertino [110]	✓	✓	✓	✓	✗	✓	✓	✓	✓	✗	✗
Chia, Keoh, and Tang [111]	✓	✓	✓	✓	✗	✓	✓	✓	✓	✗	✗
Suhail et al. [113]	✓	✓	✓	✗	✓	✓	✓	✓	✗	✗	✗
Xu, Zhang, and Wang [116]	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Suhail et al. [119]	✓	✓	✗	✗	✗	✓	✓	✗	✓	✓	✗
Liu and Wu [65]	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗
Tang and Keoh [120]	✗	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗
Porkodi and Kesavaraja [121]	✗	✗	✓	✗	✗	✗	✗	✓	✗	✗	✗
Xu, Bertino, and Wang [122]	✓	✗	✓	✓	✗	✓	✗	✓	✓	✗	✓
Xu and Wang [123]	✓	✗	✗	✓	✗	✓	✗	✗	✓	✗	✗
Sultana et al. [33]	✓	✓	✗	✗	✓	✓	✗	✗	✗	✗	✓
Siddiqui, Rahman, and Nadeem [124]	✓	✓	✗	✓	✗	✗	✗	✗	✗	✗	✗
Harshan et al. [125]	✗	✗	✓	✓	✓	✗	✗	✓	✓	✗	✓
Ali et al. [129]	✗	✓	✓	✓	✓	✗	✓	✓	✗	✗	✗
Alam and Fahmy [130]	✗	✓	✓	✓	✗	✗	✓	✓	✓	✗	✗
Kamal and Tariq [131]	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗
Baracaldo et al. [132]	✗	✗	✗	✓	✗	✗	✗	✗	✓	✗	✗
Zeng et al. [135]	✓	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
Javaid, Aman, and Sikdar [138]	✓	✗	✓	✓	✗	✓	✗	✓	✓	✗	✗
Sigwart et al. [139]	✗	✗	✓	✓	✓	✗	✗	✓	✓	✗	✗
Liu et al. [142]	✗	✗	✓	✓	✗	✗	✗	✓	✓	✗	✗
Siddiqui et al. [143]	✗	✗	✓	✓	✗	✗	✗	✓	✓	✗	✗
Yin and Fu [144]	✗	✗	✓	✓	✗	✗	✗	✓	✓	✗	✗
Sun, Tang, and Du [145]	✗	✗	✓	✓	✗	✗	✗	✓	✓	✗	✗
Chenli et al. [147]	✗	✗	✓	✓	✗	✗	✗	✓	✓	✗	✗
Sultana, Bertino, and Shehab [154]	✓	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗
Aman, Basheer, and Sikdar [156]	✓	✓	✗	✓	✓	✓	✓	✗	✓	✗	✓
Aman, Basheer, and Sikdar [157]	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗
Hamadeh and Tyagi [158]	✗	✗	✓	✓	✗	✗	✗	✓	✓	✗	✗
Chang et al. [159]	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Jaigirdar et al. [160]	✗	✗	✓	✓	✗	✗	✗	✓	✓	✗	✗
Al-Rakhami and Al-Mashari [163]	✗	✗	✓	✓	✗	✗	✗	✓	✓	✗	✗
Lomotey et al. [165]	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Lomotey, Pry, and Chai [167]	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Gao et al. [168]	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Sadineni, Pilli, and Battula [170]	✓	✗	✓	✓	✗	✓	✗	✓	✓	✗	✗
Alkhalil, Ramadan, and Ahmad [171]	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Wang et al. [172]	✗	✗	✓	✓	✓	✗	✗	✓	✓	✓	✓
Elkhodr and Alsinglawi [174]	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗

Appendix B

Additional Figures for ZW-IDS Experiments

In this appendix, we provide additional figures for the experiments on both NSL-KDD and CICIDS2017 datasets.

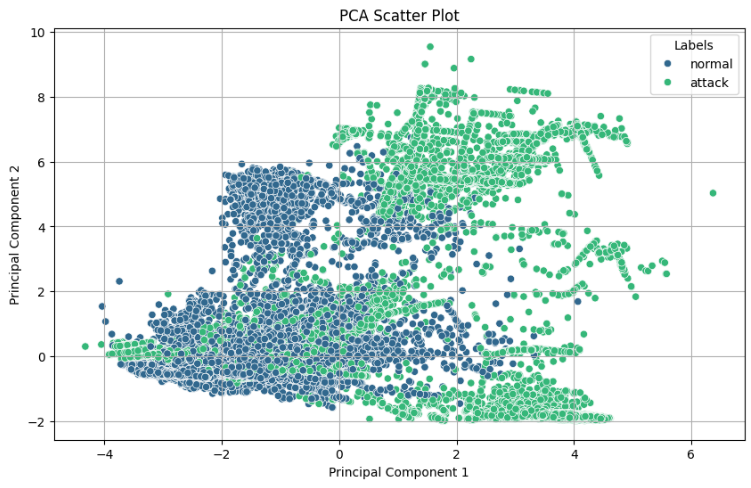


FIGURE B.1: Scatter plot of PCA components with all features with NSL-KDD dataset.

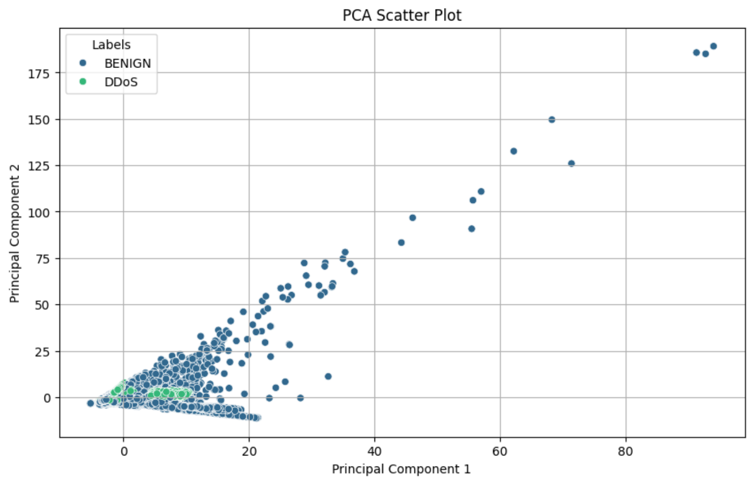
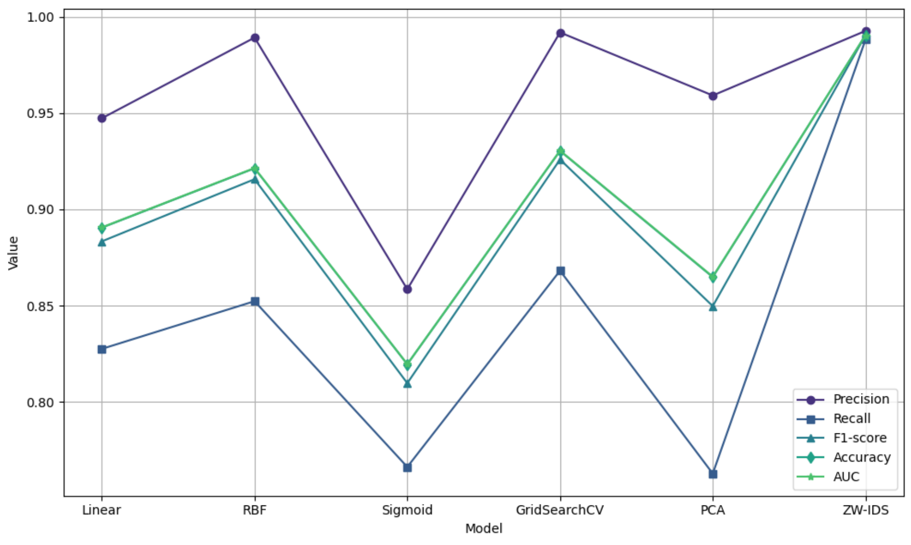
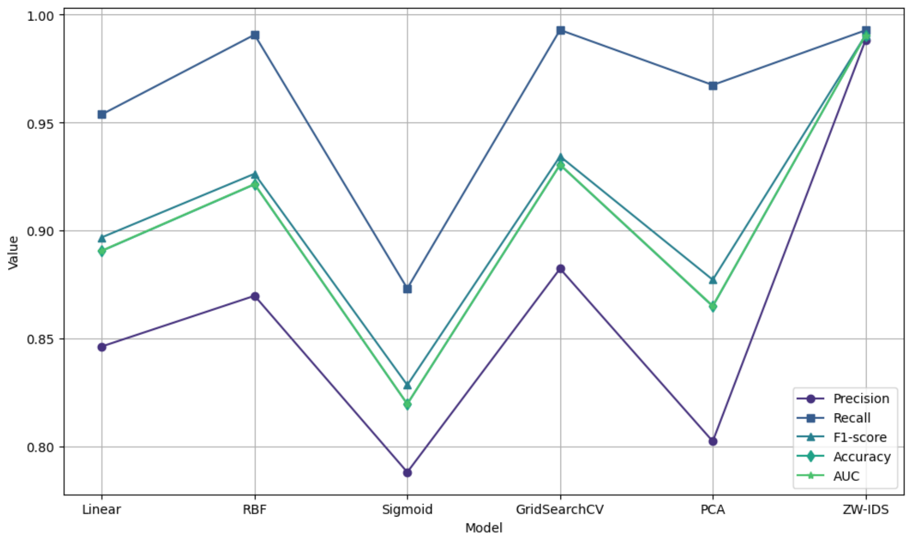


FIGURE B.2: Scatter plot of PCA components with all features with CICIDS2017 dataset.

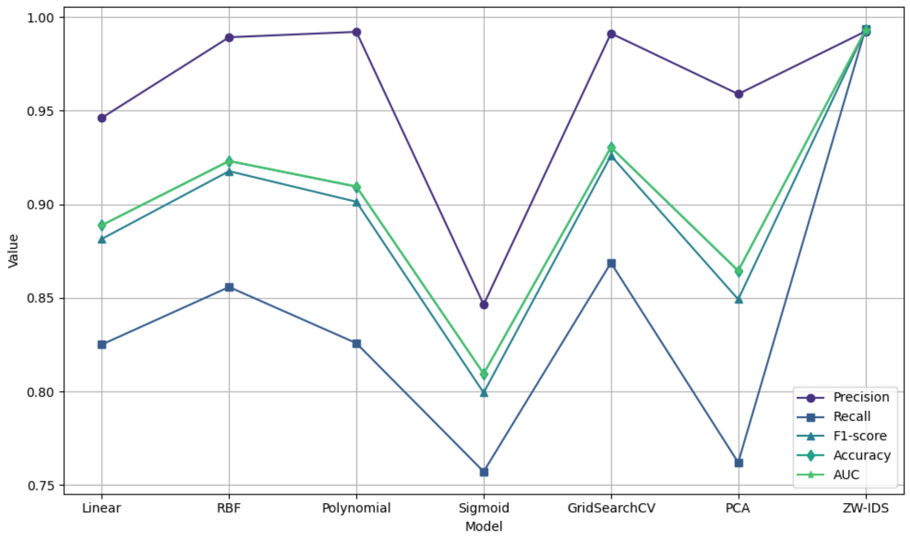


(a) Attack Label

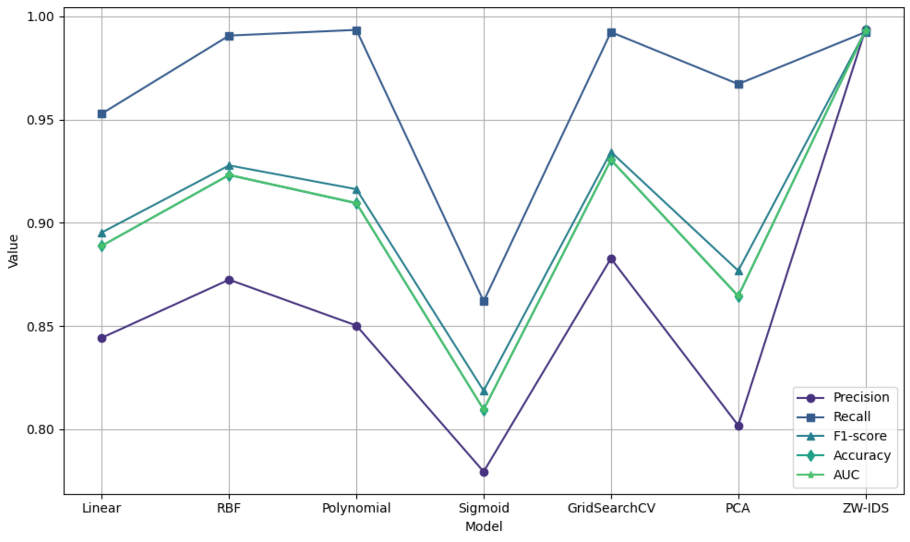


(b) Normal Label

FIGURE B.3: Visualization of performance metrics of different SVM models using all features with NSL-KDD dataset

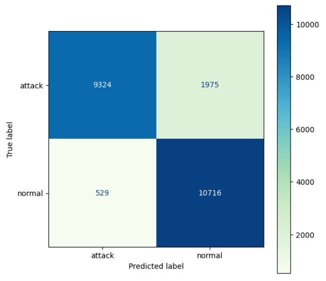


(a) Attack Label

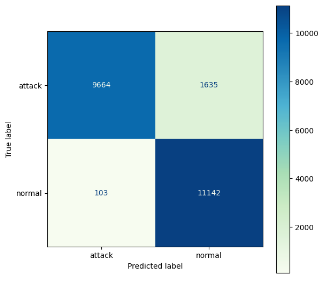


(b) Normal Label

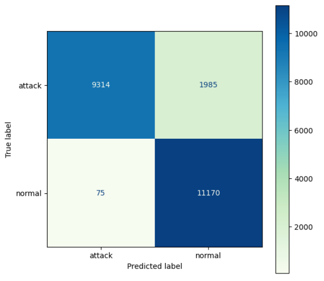
FIGURE B.4: Visualization of performance metrics of different SVM models using k-important features with NSL-KDD dataset.



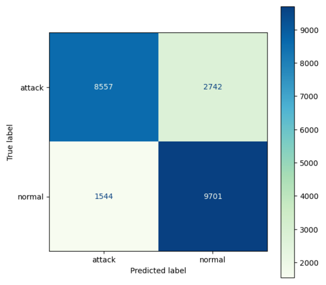
(a) Linear



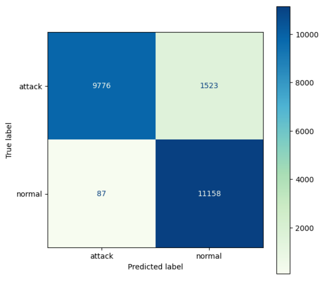
(b) RBF



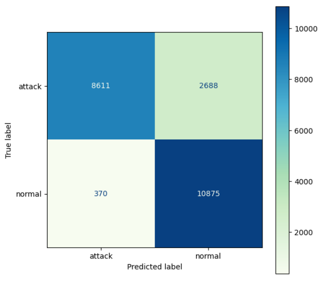
(c) Polynomial



(d) Sigmoid

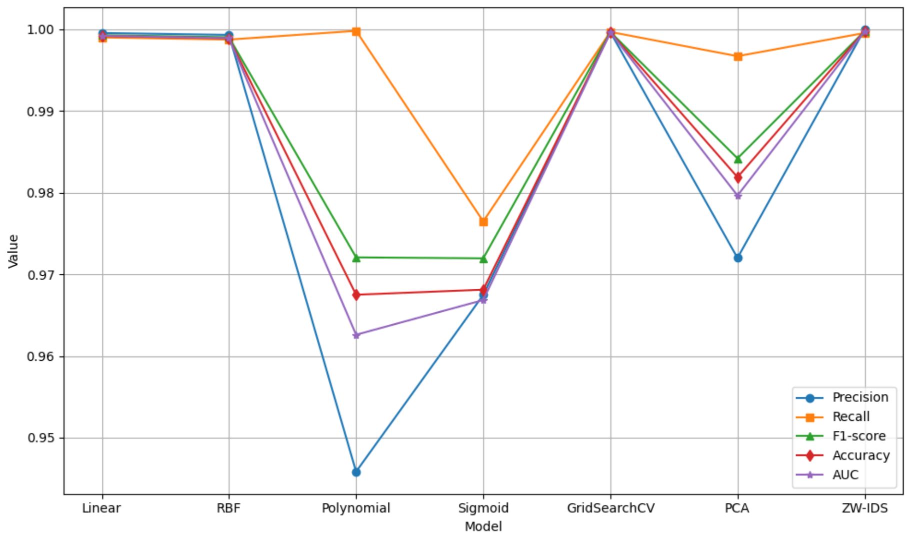


(e) GridSearchCV

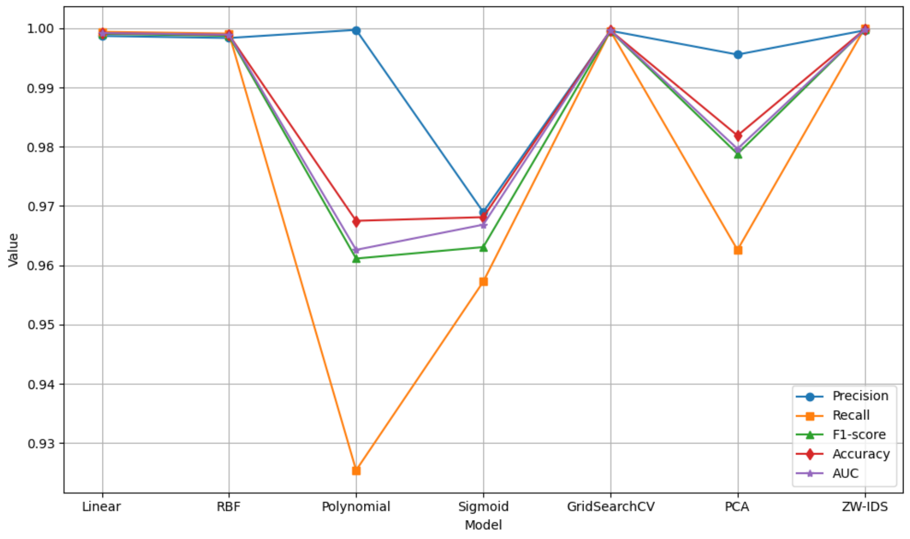


(f) GridSearchCV + PCA

FIGURE B.5: Confusion matrix of the different SVM models with k-important features and zero-watermark with NSL-KDD dataset.

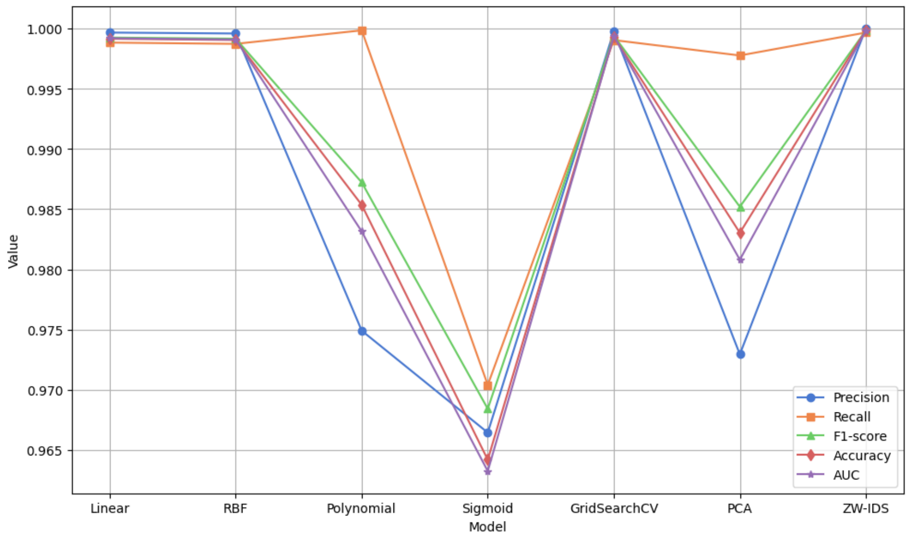


(a) Attack Label

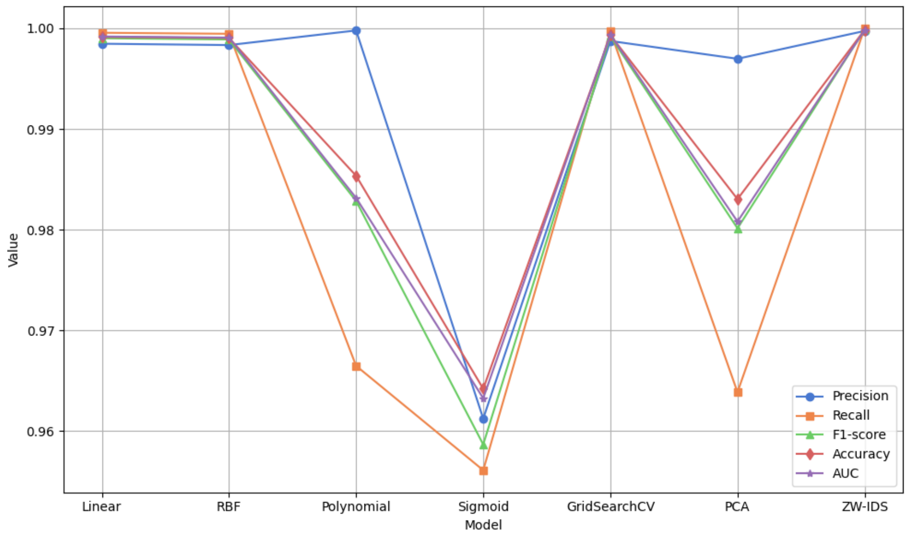


(b) Normal Label

FIGURE B.6: Visualization of performance metrics of different SVM models using all features with CICIDS2017 dataset.



(a) Attack Label



(b) Normal Label

FIGURE B.7: Visualization of performance metrics of different SVM models using k-important features with CICIDS2017 dataset.

Bibliography

[1] S. Sicari et al. "Security, privacy and trust in Internet of Things: The road ahead". In: *Computer Networks* 76 (2015), pp. 146–164. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2014.11.008.

[2] Bruno Bogaz Zarpelão et al. "A survey of intrusion detection in Internet of Things". In: *Journal of Network and Computer Applications* 84 (2017), pp. 25–37. DOI: https://doi.org/10.1016/j.jnca.2017.02.009.

[3] Kui Ren, Wenjing Lou, and Yanchao Zhang. "LEDS: Providing Location-Aware End-to-End Data Security in Wireless Sensor Networks". In: *IEEE Transactions on Mobile Computing* 7.5 (2008), pp. 585–598. DOI: 10.1109/TMC.2007.70753.

[4] Minhaj Ahmad Khan and Khaled Salah. "IoT security: Review, blockchain solutions, and open challenges". In: *Future Generation Computer Systems* 82 (2018), pp. 395–411. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2017.11.022.

[5] Xu sheng Gan et al. "Anomaly intrusion detection based on PLS feature extraction and core vector machine". In: *Knowledge-Based Systems* 40 (2013), pp. 1–6. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2012.09.004. URL: https://www.sciencedirect.com/science/article/pii/S0950705112002614.

[6] Preeti Mishra et al. "A Detailed Investigation and Analysis of Using Machine Learning Techniques for Intrusion Detection". In: *IEEE Communications Surveys & Tutorials* 21.1 (2019), pp. 686–728. DOI: 10.1109/COMST.2018.2847722.

[7] Chih-Fong Tsai et al. "Intrusion detection by machine learning: A review". In: *Expert Systems with Applications* 36.10 (2009), pp. 11994–12000. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2009.05.029.

[8] Jun Luo et al. "A novel intrusion detection method based on threshold modification using receiver operating characteristic curve". In: *Concurrency and Computation: Practice and Experience* 32.14 (2020), e5690. DOI: https://doi.org/10.1002/cpe.5690.

[9] W. Stallings. *Cryptography and Network Security: Principles and Practice*. Prentice Hall, 2011. ISBN: 9780136097044. URL: https://books.google.fr/books?id=wwfTvrWEKVwC.

[10] Hung-Jen Liao et al. "Intrusion detection system: A comprehensive review". In: *Journal of Network and Computer Applications* 36.1 (2013), pp. 16–24. ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2012.09.004.

[11] Muhammed Zekeriya Gündüz and Resul Das. "Analysis of Cyber-Attacks in IoT-based Critical Infrastructures". In: *INTERNATIONAL JOURNAL OF INFORMATION SECURITY SCIENCE* 8.4 (2019). ISSN: 2147-0030. URL: http://search/yayin/detay/383257.

[12] R.P. Lippmann et al. "Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation". In: *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00*. Vol. 2. 2000, 12–26 vol.2. DOI: 10.1109/DISCEX.2000.821506.

[13] Usman Shuaibu Musa et al. "Intrusion Detection System using Machine Learning Techniques: A Review". In: *2020 International Conference on Smart Electronics and Communication (ICOSEC)*. 2020, pp. 149–155. DOI: 10.1109/ICOSEC49089.2020.9215333.

[14] Chie-Hong Lee et al. "Machine learning based network intrusion detection". In: *2017 2nd IEEE International Conference on Computational Intelligence and Applications (ICCIA)*. 2017, pp. 79–83. DOI: 10.1109/CIAPP.2017.8167184.

[15] Fadi Salo et al. "Data Mining Techniques in Intrusion Detection Systems: A Systematic Literature Review". In: *IEEE Access* 6 (2018), pp. 56046–56058. DOI: 10.1109/ACCESS.2018.2872784.

[16] Anna L. Buczak and Erhan Guven. "A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection". In: *IEEE Communications Surveys & Tutorials* 18.2 (2016), pp. 1153–1176. DOI: 10.1109/COMST.2015.2494502.

[17] Juan Ignacio Iturbe Araya and Helena Rifà-Pous. "Anomaly-based cyberattacks detection for smart homes: A systematic literature review". In: *Internet of Things* 22 (2023), p. 100792. ISSN: 2542-6605. DOI: https://doi.org/10.1016/j.iot.2023.100792.

[18] Inês Martins et al. "Host-based IDS: A review and open issues of an anomaly detection system in IoT". In: *Future Generation Computer Systems* 133 (2022), pp. 95–113. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2022.03.001.

[19] Salmin Sultana, Mohamed Shehab, and Elisa Bertino. "Secure Provenance Transmission for Streaming Data". In: *IEEE Transactions on Knowledge and Data Engineering* 25.8 (2013), pp. 1890–1903. DOI: 10.1109/TKDE.2012.31.

[20] Chenyun Dai et al. "An Approach to Evaluate Data Trustworthiness Based on Data Provenance". In: *Secure Data Management*. Ed. by Willem Jonker and Milan Petković. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 82–98. ISBN: 978-3-540-85259-9. DOI: https://doi.org/10.1007/978-3-540-85259-9_6.

[21] Iftikhar Ahmad et al. "Performance Comparison of Support Vector Machine, Random Forest, and Extreme Learning Machine for Intrusion Detection". In: *IEEE Access* 6 (2018), pp. 33789–33795. DOI: 10.1109/ACCESS.2018.2841987.

[22] J. Mill and A. Inoue. "Support vector classifiers and network intrusion detection". In: *2004 IEEE International Conference on Fuzzy Systems (IEEE Cat. No.04CH37542)*. Vol. 1. 2004, 407–410 vol.1. DOI: 10.1109/FUZZY.2004.1375759.

[23] Peiying Tao, Zhe Sun, and Zhixin Sun. "An Improved Intrusion Detection Algorithm Based on GA and SVM". In: *IEEE Access* 6 (2018), pp. 13624–13631. DOI: 10.1109/ACCESS.2018.2810198.

[24] Yuan Zhang et al. "Anomaly-Based Network Intrusion Detection Using SVM". In: *2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)*. 2019, pp. 1–6. DOI: 10.1109/WCSP.2019.8927907.

[25] Seung-Ho Kang and Kuinam J. Kim. “A feature selection approach to find optimal feature subsets for the network intrusion detection system”. In: *Cluster Computing* 19.1 (2016), pp. 325–333. ISSN: 1573-7543. DOI: 10.1007/s10586-015-0527-8.

[26] Mahbod Tavallaee et al. “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. DOI: 10.1109/CISDA.2009.5356528.

[27] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. “Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization”. In: *Proceedings of the 4th International Conference on Information Systems Security and Privacy - Volume 1: ICISSP*, INSTICC. SciTePress, 2018, pp. 108–116. ISBN: 978-989-758-282-0. DOI: 10.5220/0006639801080116.

[28] Mahmoud Ammar, Giovanni Russello, and Bruno Crispo. “Internet of Things: A survey on the security of IoT frameworks”. In: *Journal of Information Security and Applications* 38 (2018), pp. 8–27. ISSN: 2214-2126. DOI: https://doi.org/10.1016/j.jisa.2017.11.002.

[29] Alessio Botta et al. “Integration of Cloud computing and Internet of Things: A survey”. In: *Future Generation Computer Systems* 56 (2016), pp. 684–700. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2015.09.021.

[30] Mihai T. Lazarescu. “Wireless Sensor Networks for the Internet of Things: Barriers and Synergies”. In: *Components and Services for IoT Platforms: Paving the Way for IoT Standards*. Springer International Publishing, 2017. Chap. 9, pp. 155–186. ISBN: 978-3-319-42304-3. DOI: 10.1007/978-3-319-42304-3_9.

[31] Narayanan Manikandan and Srinivasan Subha. “Parallel AES algorithm for performance improvement in data analytics security for IoT”. In: *International Journal of Networking and Virtual Organisations* 18.2 (2018), pp. 112–129. DOI: 10.1504/IJNVO.2018.091575.

[32] Jayavardhana Gubbi et al. “Internet of Things (IoT): A vision, architectural elements, and future directions”. In: *Future Generation Computer Systems* 29.7 (2013), pp. 1645–1660. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2013.01.010.

[33] Salmin Sultana et al. “A Lightweight Secure Scheme for Detecting Provenance Forgery and Packet DropAttacks in Wireless Sensor Networks”. In: *IEEE Transactions on Dependable and Secure Computing* 12.3 (2015), pp. 256–269. DOI: 10.1109/TDSC.2013.44.

[34] Rui Hu et al. “A survey on data provenance in IoT”. In: *World Wide Web* 23.2 (2020), pp. 1441–1463. ISSN: 1573-1413. DOI: 10.1007/s11280-019-00746-1.

[35] Y. Richard Wang and Stuart E. Madnick. “A Polygon Model for Heterogeneous Database Systems: The Source Tagging Perspective”. In: *Proceedings of the Sixteenth International Conference on Very Large Databases*. Brisbane, Australia: Morgan Kaufmann Publishers Inc., 1990, 519–533. ISBN: 055860149X. DOI: https://dl.acm.org/doi/10.5555/94362.94604.

[36] Changda Wang, Wenyi Zheng, and Elisa Bertino. “Provenance for Wireless Sensor Networks: A Survey”. In: *Data Science and Engineering* 1.3 (2016), pp. 189–200. ISSN: 2364-1541. DOI: 10.1007/s41019-016-0017-x.

[37] Peter Buneman, Sanjeev Khanna, and Tan Wang-Chiew. "Why and Where: A Characterization of Data Provenance". In: *Database Theory — ICDT 2001*. Ed. by Jan Van den Bussche and Victor Vianu. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 316–330. ISBN: 978-3-540-44503-6. DOI: https://doi.org/10.1007/3-540-44503-X_20.

[38] Yogesh L. Simmhan, Beth Plale, and Dennis Gannon. "A Survey of Data Provenance in E-Science". In: *SIGMOD Rec.* 34.3 (2005), 31–36. ISSN: 0163-5808. DOI: 10.1145/1084805.1084812.

[39] Faheem Zafar et al. "Trustworthy data: A survey, taxonomy and future trends of secure provenance schemes". In: *Journal of Network and Computer Applications* 94 (2017), pp. 50–68. ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2017.06.003.

[40] Luigi Atzori, Antonio Iera, and Giacomo Morabito. "The Internet of Things: A survey". In: *Computer Networks* 54.15 (2010), pp. 2787–2805. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2010.05.010.

[41] I.F. Akyildiz et al. "Wireless sensor networks: a survey". In: *Computer Networks* 38.4 (2002), pp. 393–422. ISSN: 1389-1286. DOI: https://doi.org/10.1016/S1389-1286(01)00302-4.

[42] Ron G Van Schyndel, Andrew Z Tirkel, and Charles F Osborne. "A digital watermark". In: *Proceedings of 1st international conference on image processing*. Vol. 2. IEEE. 1994, pp. 86–90. DOI: 10.1109/ICIP.1994.413536.

[43] Chuhong Fei, Deepa Kundur, and Raymond H Kwong. "Analysis and design of secure watermark-based authentication systems". In: *IEEE transactions on information forensics and security* 1.1 (2006), pp. 43–55. DOI: 10.1109/TIFS.2005.863505.

[44] Ingemar Cox et al. *Digital Watermarking and Steganography*. Morgan Kaufmann Publishers Inc., 2007. DOI: https://doi.org/10.1016/B978-0-12-372585-1.X5001-3.

[45] David Megías. "Data Hiding: New Opportunities for Security and Privacy?" In: *Proceedings of the European Interdisciplinary Cybersecurity Conference*. EICC 2020. Rennes, France, 2021, pp. 1–6. DOI: 10.1145/3424954.3425511.

[46] Luis Perez-Freire et al. "Watermarking Security: a Survey". In: *Transactions on data hiding and multimedia security I*. Springer, 2006, pp. 41–72. URL: https://dl.acm.org/doi/10.5555/2172238.2172240.

[47] Guoyin Zhang et al. "A New Digital Watermarking Method for Data Integrity Protection in the Perception Layer of IoT". In: *Security and Communication Networks* 2017 (2017). ISSN: 1939-0114. DOI: https://doi.org/10.1155/2017/3126010.

[48] Khizar Hameed et al. "Towards a formally verified zero watermarking scheme for data integrity in the Internet of Things based-wireless sensor networks". In: *Future Generation Computer Systems* 82 (2018), pp. 274–289. DOI: https://doi.org/10.1016/j.future.2017.12.009.

[49] Bofeng Pan, Natalia Stakhanova, and Suprio Ray. "Data Provenance in Security and Privacy". In: *ACM Comput. Surv.* (2023). ISSN: 0360-0300. DOI: 10.1145/3593294.

[50] Luc Moreau et al. “A Templating System to Generate Provenance”. In: *IEEE Transactions on Software Engineering* 44.2 (2018), pp. 103–121. DOI: 10.1109/TSE.2017.2659745.

[51] Yingwei Cui, Jennifer Widom, and Janet L. Wiener. “Tracing the Lineage of View Data in a Warehousing Environment”. In: *ACM Trans. Database Syst.* 25.2 (2000), 179–227. ISSN: 0362-5915. DOI: 10.1145/357775.357777.

[52] James Cheney, Laura Chiticariu, and Wang-Chiew Tan. “Provenance in Databases: Why, How, and Where”. In: *Foundations and Trends® in Databases* 1.4 (2009), pp. 379–474. ISSN: 1931-7883. DOI: 10.1561/1900000006.

[53] Boris Glavic and Klaus R. Dittrich. “Data Provenance: A Categorization of Existing Approaches”. In: *Datenbanksysteme für Business, Technologie und Web*. 2007, pp. 227–241. URL: https://www.zora.uzh.ch/id/eprint/24450/.

[54] Peter Buneman, Sanjeev Khanna, and Tan Wang-Chiew. “Why and Where: A Characterization of Data Provenance”. In: *Database Theory — ICDT 2001*. Ed. by Jan Van den Bussche and Victor Vianu. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 316–330. ISBN: 978-3-540-44503-6.

[55] Ming Gao et al. “A Survey on Management of Data Provenance: A Survey on Management of Data Provenance”. In: *Chinese Journal of Computers* 33 (2010), pp. 373–389.

[56] Magesh Jayapandian et al. “Michigan Molecular Interactions (MiMI): putting the jigsaw puzzle together”. In: *Nucleic Acids Research* 35.suppl1 (Nov. 2006), pp. D566–D571. ISSN: 0305-1048. DOI: 10.1093/nar/gkl859.

[57] Tara Salman et al. “Security Services Using Blockchains: A State of the Art Survey”. In: *IEEE Communications Surveys and Tutorials* 21.1 (2019), pp. 858–880. DOI: 10.1109/COMST.2018.2863956.

[58] Jin Li et al. “Digital provenance: Enabling secure data forensics in cloud computing”. In: *Future Generation Computer Systems* 37 (2014), pp. 259–266. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2013.10.006.

[59] Oludare Isaac Abiodun et al. “Data provenance for cloud forensic investigations, security, challenges, solutions and future perspectives: A survey”. In: *Journal of King Saud University - Computer and Information Sciences* 34.10, Part B (2022), pp. 10217–10245. ISSN: 1319-1578. DOI: https://doi.org/10.1016/j.jksuci.2022.10.018.

[60] Wei-Zhe Zhang et al. “Secure and Optimized Load Balancing for Multitier IoT and Edge-Cloud Computing Systems”. In: *IEEE Internet of Things Journal* 8.10 (2021), pp. 8119–8132. DOI: 10.1109/JIOT.2020.3042433.

[61] Shams Zawoad, Ragib Hasan, and Kamrul Islam. “SECProv: Trustworthy and Efficient Provenance Management in the Cloud”. In: *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*. 2018, pp. 1241–1249. DOI: 10.1109/INFOCOM.2018.8485824.

[62] Changhao Chenli and Taeho Jung. “ProvNet: Networked Blockchain for Decentralized Secure Provenance”. In: *Blockchain – ICBC 2020*. Ed. by Zhixiong Chen et al. Cham: Springer International Publishing, 2020, pp. 76–93. ISBN: 978-3-030-59638-5. DOI: https://doi.org/10.1007/978-3-030-59638-5_6.

[63] Syed Rafiul Hussain et al. “Secure data provenance compression using arithmetic coding in wireless sensor networks”. In: *2014 IEEE 33rd International Performance Computing and Communications Conference (IPCCC)*. 2014, pp. 1–10. DOI: 10.1109/PCCC.2014.7017068.

[64] Xueping Liang et al. “ProvChain: A Blockchain-Based Data Provenance Architecture in Cloud Environment with Enhanced Privacy and Availability”. In: *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. 2017, pp. 468–477. DOI: 10.1109/CCGRID.2017.8.

[65] Zhe Liu and Yuting Wu. “An Index-Based Provenance Compression Scheme for Identifying Malicious Nodes in Multihop IoT Network”. In: *IEEE Internet of Things Journal* 7.5 (2020), pp. 4061–4071. DOI: 10.1109/JIOT.2019.2961431.

[66] Aravind Ramachandran and Murat Kantarcioglu. “SmartProvenance: A Distributed, Blockchain Based DataProvenance System”. In: *Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy*. CODASPY ’18. Tempe, AZ, USA: Association for Computing Machinery, 2018, 35–42. ISBN: 9781450356329. DOI: 10.1145/3176258.3176333.

[67] Ragib Hasan, Radu Sion, and Marianne Winslett. “The Case of the Fake Picasso: Preventing History Forgery with Secure Provenance”. In: *7th USENIX Conference on File and Storage Technologies (FAST 09)*. San Francisco, CA: USENIX Association, Feb. 2009. URL: https://www.usenix.org/conference/fast09/technical-sessions/presentation/hasan.

[68] Mohammed Rangwala et al. “A mutual agreement signature scheme for secure data provenance”. In: *The 23rd International Conference on Computer Communications and Networks*. 2016, pp. 726–733. URL: https://cs.iupui.edu/~xzou/Papers/ICCCN14_Digital_Provenance.pdf.

[69] Fuzel Jamil et al. “Secure provenance using an authenticated data structure approach”. In: *Comput. Secur.* 73 (2018), pp. 34–56. URL: https://api.semanticscholar.org/CorpusID:3463031.

[70] Xinlei Wang et al. “Chaining for securing data provenance in distributed information networks”. In: *MILCOM 2012 - 2012 IEEE Military Communications Conference*. Oct. 2012, pp. 1–6. ISBN: 978-1-4673-1729-0. DOI: 10.1109/MILCOM.2012.6415609.

[71] Muhammad Rizwan Asghar et al. “Securing Data Provenance in the Cloud”. In: *Open Problems in Network Security*. Ed. by Jan Camenisch and Dogan Kesdogan. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 145–160. ISBN: 978-3-642-27585-2. DOI: https://doi.org/10.1007/978-3-642-27585-2_12.

[72] Michael Backes, Niklas Grimm, and Aniket Kate. “Data Lineage in Malicious Environments”. In: *IEEE Transactions on Dependable and Secure Computing* 13.2 (2016), pp. 178–191. DOI: 10.1109/TDSC.2015.2399296.

[73] Mohammad M. Bany Taha, Sivadon Chaisiri, and Ryan K. L. Ko. “Trusted Tamper-Evident Data Provenance”. In: *2015 IEEE Trustcom/BigDataSE/ISPA*. Vol. 1. 2015, pp. 646–653. DOI: 10.1109/Trustcom.2015.430.

[74] Raja Naeem Akram and Ryan K. L. Ko. “Unified Model for Data Security - A Position Paper”. In: *Proceedings of the 2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications*. TRUSTCOM ’14. USA: IEEE Computer Society, 2014, 831–839. ISBN: 9781479965137. DOI: 10.1109/TrustCom.2014.110.

[75] Emmanouil Vasilomanolakis et al. “Taxonomy and Survey of Collaborative Intrusion Detection”. In: *ACM Comput. Surv.* 47.4 (2015). ISSN: 0360-0300. DOI: 10.1145/2716260.

[76] Michael Zipperle et al. “Provenance-based Intrusion Detection Systems: A Survey”. In: *ACM Comput. Surv.* 55.7 (2022). ISSN: 0360-0300. DOI: 10.1145/3539605.

[77] Atefeh Torkaman, Ghazaleh Javadzadeh, and Marjan Bahrololum. “A hybrid intelligent HIDS model using two-layer genetic algorithm and neural network”. In: *The 5th Conference on Information and Knowledge Technology*. 2013, pp. 92–96. DOI: 10.1109/IKT.2013.6620045.

[78] N. Chaabouni et al. “Network Intrusion Detection for IoT Security Based on Learning Techniques”. In: *IEEE Communications Surveys Tutorials* 21.3 (2019), pp. 2671–2701. ISSN: 2373-745X. DOI: 10.1109/COMST.2019.2896380.

[79] Rami Puzis et al. “Optimization of NIDS Placement for Protection of Intercommunicating Critical Infrastructures”. In: *Intelligence and Security Informatics*. Ed. by Daniel Ortiz-Arroyo et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 191–203. ISBN: 978-3-540-89900-6. DOI: https://doi.org/10.1007/978-3-540-89900-6_20.

[80] Chirag Modi et al. “A survey of intrusion detection techniques in Cloud”. In: *Journal of Network and Computer Applications* 36.1 (2013), pp. 42–57. ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2012.05.003.

[81] Kuldeep Yadav and Avinash Srinivasan. “iTrust: an integrated trust framework for wireless sensor networks”. In: *Proceedings of the 2010 ACM Symposium on Applied Computing*. SAC ’10. Sierre, Switzerland: Association for Computing Machinery, 2010, 1466–1471. ISBN: 9781605586397. DOI: 10.1145/1774088.1774402.

[82] Luca Arnaboldi and Charles Morisset. *A Review of Intrusion Detection Systems and Their Evaluation in the IoT*. 2021. DOI: https://doi.org/10.48550/arXiv.2105.08096. arXiv: 2105.08096 [cs.CR].

[83] Liqun Liu et al. “An intrusion detection method for internet of things based on suppressed fuzzy clustering”. In: *EURASIP Journal on Wireless Communications and Networking* 2018.1 (2018), p. 113. ISSN: 1687-1499. DOI: 10.1186/s13638-018-1128-z.

[84] Christian Cervantes et al. “Detection of sinkhole attacks for supporting secure routing on 6LoWPAN for Internet of Things”. In: *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*. 2015, pp. 606–611. DOI: 10.1109/INM.2015.7140344.

[85] Zeeshan Ali Khan and Peter Herrmann. “A Trust Based Distributed Intrusion Detection Mechanism for Internet of Things”. In: *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*. 2017, pp. 1169–1176. DOI: 10.1109/AINA.2017.161.

[86] Stefan Axelsson. *Intrusion Detection Systems: A Survey and Taxonomy*. Tech. rep. Chalmers University of Technology, 2000. URL: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=7a15948bdcb530e2c1deedd8d22dd9b54788a634.

[87] Xueyuan Han et al. “FRAPpuccino: Fault-detection through Runtime Analysis of Provenance”. In: *9th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 17)*. Santa Clara, CA: USENIX Association, July 2017. URL: https://www.usenix.org/conference/hotcloud17/program/presentation/han.

[88] Tom M. Mitchell. *Machine Learning*. McGraw Hill, 1997. ISBN: 97800704280720070428077.

[89] Taiwo Oladipupo Ayodele. *New Advances in Machine Learning*. InTech, 2010. ISBN: 978-953-307-034-6.

[90] P. Langley and H.A. Simon. “Applications of Machine Learning and Rule Induction”. In: *Communications of the ACM* 38.11 (1995), pp. 54–64. DOI: 10.1145/219717.219768.

[91] C. Kolias et al. “Intrusion Detection in 802.11 Networks: Empirical Evaluation of Threats and a Public Dataset”. In: *IEEE Communications Surveys Tutorials* 18.1 (2016), pp. 184–208. ISSN: 2373-745X. DOI: 10.1109/COMST.2015.2402161.

[92] D. Praveen Kumar, Tarachand Amgoth, and Chandra Sekhara Rao Annavarapu. “Machine learning algorithms for wireless sensor networks: A survey”. In: *Information Fusion* 49 (2019), pp. 1 –25. ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2018.09.013.

[93] Huiping Guo, Yingjiu Li, and Sushil Jajodia. “Chaining watermarks for detecting malicious modifications to streaming data”. In: *Information Sciences* 177.1 (2007), pp. 281–298. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2006.03.014.

[94] Ibrahim Kamel and Hussam Juma. “Simplified Watermarking Scheme for Sensor Networks”. In: *International Journal of Internet Protocol Technology* 5.1/2 (2010), 101–111. ISSN: 1743-8209. DOI: 10.1504/IJIPT.2010.032619.

[95] I. Kamel and Hussam Juma. “A Lightweight Data Integrity Scheme for Sensor Networks”. In: *Sensors (Basel, Switzerland)* 11 (Dec. 2011), pp. 4118–4136. DOI: 10.3390/s110404118.

[96] Baowei Wang, Weiwen Kong, and Naixue Xiong. “A Dual-Chaining Watermark Scheme for Data Integrity Protection in Internet of Things”. In: *Computers, Materials and Continua* 58 (Jan. 2019), pp. 679–695. DOI: 10.32604/cmc.2019.06106.

[97] Xi Shi and Di Xiao. “A Reversible Watermarking Authentication Scheme for Wireless Sensor Networks”. In: *Information Sciences* 240 (2013), 173—183. ISSN: 0020-0255. DOI: 10.1016/j.ins.2013.03.031.

[98] Xingming Sun et al. “Digital Watermarking Method for Data Integrity Protection in Wireless Sensor Networks”. In: *International Journal of Security and Its Applications* 7.4 (2013), pp. 407–416. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84883411518&partnerID=40&md5=1f268560dc663749dbfb41b10033b764.

[99] Lingjing Zhou and Zhengdao Zhang. “A secure data transmission scheme for wireless sensor networks based on digital watermarking”. In: *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery*. 2012, pp. 2097–2101. DOI: 10.1109/FSKD.2012.6234194.

[100] Arwa Alromih, Mznah Al-Rodhaan, and Yuan Tian. “A Randomized Watermarking Technique for Detecting Malicious Data Injection Attacks in Heterogeneous Wireless Sensor Networks for Internet of Things Applications”. In: *Sensors (Basel, Switzerland)* 18.12 (2018). ISSN: 1424-8220. DOI: 10.3390/s18124346.

[101] Farid Lalem et al. “Data Authenticity and Integrity in Wireless Sensor Networks Based on a Watermarking Approach”. In: *The 29th International Florida Artificial Intelligence Research Society*. FLAIRS-29. Key Largo, Florida, United States, May 2016, pp. 276–281. URL: https://hal.univ-brest.fr/hal-01294146.

[102] Simone Soderi. “Acoustic-based security: A key enabling technology for wireless sensor networks”. In: *International Journal of Wireless Information Networks* 27.1 (2020), pp. 45–59. DOI: https://doi.org/10.1007/s10776-019-00473-4.

[103] Djallel Boubiche et al. “SDAW: Secure Data Aggregation Watermarking-based Scheme in Homogeneous WSNs”. In: *Telecommunication Systems* 59 (Apr. 2015), pp. 277–288. DOI: 10.1007/s11235-015-0047-0.

[104] Unkyu Park and John Heidemann. “Provenance in Sensornet Republishing”. In: *Provenance and Annotation of Data and Processes: Second International Provenance and Annotation Workshop, IPAW 2008, Salt Lake City, UT, USA, June 17-18, 2008. Revised Selected Papers*. Berlin, Heidelberg: Springer-Verlag, 2008, 280–292. ISBN: 9783540899648. URL: https://doi.org/10.1007/978-3-540-89965-5_28.

[105] Hyo-Sang Lim, Yang-Sae Moon, and Elisa Bertino. “Provenance-Based Trustworthiness Assessment in Sensor Networks”. In: *Proceedings of the Seventh International Workshop on Data Management for Sensor Networks*. DMSN ’10. Singapore: Association for Computing Machinery, 2010, 2–7. ISBN: 9781450304160. DOI: 10.1145/1858158.1858162.

[106] Elisa Bertino. “Security and Privacy in the IoT”. In: Springer, Cham, Jan. 2018, pp. 3–10. ISBN: 978-3-319-75159-7. DOI: 10.1007/978-3-319-75160-3_1.

[107] Md Morshed Alam and Weichao Wang. “A Comprehensive Survey on Data Provenance: State-of-the-Art Approaches and Their Deployments for IoT Security Enforcement”. In: *J. Comput. Secur.* 29.4 (2021), 423–446. ISSN: 0926-227X. DOI: 10.3233/JCS-200108.

[108] Wolali Ametepe et al. “Data provenance collection and security in a distributed environment: a survey”. In: *International Journal of Computers and Applications* 43.1 (2021), pp. 11–25. DOI: 10.1080/1206212X.2018.1501937.

[109] Emrullah Gultekin and Mehmet S. Aktas. “Systematic Literature Review on Data Provenance in Internet of Things”. In: *Computational Science and Its Applications – ICCSA 2022 Workshops*. Ed. by Osvaldo Gervasi et al. Cham: Springer International Publishing, 2022, pp. 31–46. ISBN: 978-3-031-10542-5. DOI: https://doi.org/10.1007/978-3-031-10542-5_3.

[110] Changda Wang, Syed Rafiul Hussain, and Elisa Bertino. “Dictionary Based Secure Provenance Compression for Wireless Sensor Networks”. In: *IEEE Transactions on Parallel and Distributed Systems* 27.2 (2016), pp. 405–418. DOI: 10.1109/TPDS.2015.2402156.

[111] Ming Hong Chia, Sye Loong Keoh, and Zhaohui Tang. “Secure Data Provenance in Home Energy Monitoring Networks”. In: *Proceedings of the 3rd Annual Industrial Control System Security Workshop*. ICSS 2017. San Juan, PR, USA: Association for Computing Machinery, 2017, 7–14. ISBN: 9781450363334. DOI: 10.1145/3174776.3174778.

[112] Adi Shamir. “How to Share a Secret”. In: *Commun. ACM* 22.11 (1979), 612–613. ISSN: 0001-0782. DOI: 10.1145/359168.359176.

[113] Sabah Suhail et al. “Data trustworthiness in IoT”. In: *2018 International Conference on Information Networking (ICOIN)*. 2018, pp. 414–419. DOI: 10.1109/ICOIN.2018.8343151.

[114] Roger Alexander et al. *RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks*. RFC 6550. 2012. DOI: 10.17487/RFC6550.

[115] A. Dunkels, B. Gronvall, and T. Voigt. “Contiki - a lightweight and flexible operating system for tiny networked sensors”. In: *29th Annual IEEE International Conference on Local Computer Networks*. 2004, pp. 455–462. DOI: 10.1109/LCN.2004.38.

[116] Qinbao Xu, Xing Zhang, and Changda Wang. “Provenance Compression Using Packet-Path-Index Differences in Wireless Sensor Networks”. In: *2019 15th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN)*. 2019, pp. 200–205. DOI: 10.1109/MSN48538.2019.00047.

[117] Sahin Buyrukbilen and Spiridon Bakiras. “Secure Similar Document Detection with Simhash”. In: *Secure Data Management*. Ed. by Willem Jonker and Milan Petković. Cham: Springer International Publishing, 2014, pp. 61–75. ISBN: 978-3-319-06811-4. DOI: https://doi.org/10.1007/978-3-319-06811-4_12.

[118] Moses S. Charikar. “Similarity Estimation Techniques from Rounding Algorithms”. In: *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*. STOC ’02. Montreal, Quebec, Canada: Association for Computing Machinery, 2002, 380–388. ISBN: 1581134959. DOI: 10.1145/509907.509965.

[119] Sabah Suhail et al. “Provenance-enabled packet path tracing in the RPL-based internet of things”. In: *Computer Networks* 173 (2020), p. 107189. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2020.107189.

[120] Zhaohui Tang and Sye Loong Keoh. “An Efficient Scheme to Secure Data Provenance in Home Area Networks”. In: *2020 IEEE 3rd 5G World Forum (5GWF)*. 2020, pp. 115–120. DOI: 10.1109/5GWF49715.2020.9221402.

[121] S. Porkodi and D. Kesavaraja. “Secure Data Provenance in Internet of Things using Hybrid Attribute based Crypt Technique”. In: *Wireless Personal Communications* (2021). DOI: 10.1007/s11277-021-08157-0.

[122] Qinbao Xu, Elisa Bertino, and Changda Wang. “Compact Provenance Scheme Through Packet Path Index Differences in WSNs”. In: *IEEE Transactions on Information Forensics and Security* 17 (2022), pp. 3511–3524. DOI: 10.1109/TIFS.2022.3207897.

[123] Qinbao Xu and Changda Wang. “Stepwise Refinement Provenance Scheme for Wireless Sensor Networks”. In: *IEEE Internet of Things Journal* 9.13 (2022), pp. 11126–11140. DOI: 10.1109/JIOT.2021.3127496.

[124] Muhammad Shoaib Siddiqui, Atiqur Rahman, and Adnan Nadeem. “Secure Data Provenance in IoT Network using Bloom Filters”. In: *Procedia Computer Science* 163 (2019). 16th Learning and Technology Conference 2019Artificial Intelligence and Machine Learning: Embedding the Intelligence, pp. 190–197. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2019.12.100.

[125] J. Harshan et al. “Bloom Filter Based Low-Latency Provenance Embedding Schemes in Wireless Networks”. In: *2020 IEEE Wireless Communications and Networking Conference (WCNC)*. 2020, pp. 1–7. DOI: 10.1109/WCNC45663.2020.9120640.

[126] Xiaopei Lu et al. “PathZip: Packet path tracing in wireless sensor networks”. In: *2012 IEEE 9th International Conference on Mobile Ad-Hoc and Sensor Systems (MASS 2012)*. 2012, pp. 380–388. DOI: 10.1109/MASS.2012.6502538.

[127] Bilal Shebaro et al. “Demonstrating a Lightweight Data Provenance for Sensor Networks”. In: *Proceedings of the 2012 ACM Conference on Computer and Communications Security*. CCS ’12. Association for Computing Machinery, 2012, 1022–1024. ISBN: 9781450316514. DOI: 10.1145/2382196.2382312.

[128] David Megías. “Improved Privacy-Preserving P2P Multimedia Distribution Based on Recombined Fingerprints”. In: *IEEE Transactions on Dependable and Secure Computing* 12.2 (2015), pp. 179–189. DOI: 10.1109/TDSC.2014.2320712.

[129] Syed Taha Ali et al. “Securing First-Hop Data Provenance for Bodyworn Devices Using Wireless Link Fingerprints”. In: *IEEE Transactions on Information Forensics and Security* 9.12 (2014), pp. 2193–2204. DOI: 10.1109/TIFS.2014.2357998.

[130] S.M. Iftekharul Alam and Sonia Fahmy. “A practical approach for provenance transmission in wireless sensor networks”. In: *Ad Hoc Networks* 16 (2014), pp. 28–45. ISSN: 1570-8705. DOI: https://doi.org/10.1016/j.adhoc.2013.12.001.

[131] Mohsin Kamal and Muhammad Tariq. “Light-Weight Security and Data Provenance for Multi-Hop Internet of Things”. In: *IEEE Access* 6 (2018), pp. 34439–34448. DOI: 10.1109/ACCESS.2018.2850821.

[132] Nathalie Baracaldo et al. “Securing Data Provenance in Internet of Things (IoT) Systems”. In: *Service-Oriented Computing – ICSOC 2016 Workshops*. Ed. by Khalil Drira et al. Springer International Publishing, 2017, pp. 92–98. ISBN: 978-3-319-68136-8. DOI: https://doi.org/10.1007/978-3-319-68136-8_9.

[133] Ahto Buldas, Andres Kroonmaa, and Risto Laanoja. “Keyless Signatures’ Infrastructure: How to Build Global Distributed Hash-Trees”. In: *Secure IT Systems*. Ed. by Hanne Riis Nielson and Dieter Gollmann. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 313–320. ISBN: 978-3-642-41488-6.

[134] Ahto Buldas et al. “Efficient Record-Level Keyless Signatures for Audit Logs”. In: *Secure IT Systems*. Ed. by Karin Bernsmed and Simone Fischer-Hübner. Cham: Springer International Publishing, 2014, pp. 149–164. ISBN: 978-3-319-11599-3.

[135] Yu Zeng et al. “A Blockchain-Based Scheme for Secure Data Provenance in Wireless Sensor Networks”. In: *2018 14th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN)*. 2018, pp. 13–18. DOI: 10.1109/MSN.2018.00009.

[136] Weisong Shi et al. “Edge Computing: Vision and Challenges”. In: *IEEE Internet of Things Journal* 3.5 (2016), pp. 637–646. DOI: 10.1109/JIOT.2016.2579198.

[137] Gavin Wood. “Ethereum: A secure decentralised generalised transaction ledger”. In: *Journal of Computer and Communications* 151 (2014), pp. 1–32. URL: https://api.semanticscholar.org/CorpusID:261284440.

[138] Uzair Javaid, Muhammad Naveed Aman, and Biplab Sikdar. “BlockPro: Blockchain Based Data Provenance and Integrity for Secure IoT Environments”. In: *Proceedings of the 1st Workshop on Blockchain-Enabled Networked Sensor Systems*. BlockSys’18. Shenzhen, China: Association for Computing Machinery, 2018, 13–18. ISBN: 9781450360500. DOI: 10.1145/3282278.3282281.

[139] Marten Sigwart et al. “A secure and extensible blockchain-based data provenance framework for the Internet of Things”. In: *Personal and Ubiquitous Computing* (2020). ISSN: 1617-4917. DOI: 10.1007/s00779-020-01417-z.

[140] Marten Sigwart et al. “Blockchain-Based Data Provenance for the Internet of Things”. In: *Proceedings of the 9th International Conference on the Internet of Things*. IoT ’19. Bilbao, Spain: Association for Computing Machinery, 2019, pp. 1–8. ISBN: 9781450372077. DOI: 10.1145/3365871.3365886.

[141] Habeeb Olufowobi et al. “Data Provenance Model for Internet of Things (IoT) Systems”. In: *Service-Oriented Computing – ICSOC 2016 Workshops*. Springer International Publishing, Oct. 2017, pp. 85–91. ISBN: 978-3-319-68135-1. DOI: 10.1007/978-3-319-68136-8_8.

[142] Dongxiao Liu et al. “Secure and Efficient Distributed Network Provenance for IoT: A Blockchain-Based Approach”. In: *IEEE Internet of Things Journal* 7.8 (2020), pp. 7564–7574. DOI: 10.1109/JIOT.2020.2988481.

[143] Muhammad Shoaib Siddiqui et al. “BlockTrack-L: A Lightweight Blockchain-based Provenance Message Tracking in IoT”. In: *International Journal of Advanced Computer Science and Applications* 11.4 (2020). DOI: 10.14569/IJACSA.2020.0110462.

[144] Fei Yin and Zhiqiang Fu. “A Data Provenance Scheme Based on Blockchain for Internet of Things”. In: *2022 2nd International Conference on Computer Science and Blockchain (CCSB)*. 2022, pp. 42–45. DOI: 10.1109/CCSB58128.2022.00014.

[145] Shuang Sun, Huayun Tang, and Rong Du. “A Novel Blockchain-Based IoT Data Provenance Model”. In: *2022 2nd International Conference on Computer Science and Blockchain (CCSB)*. 2022, pp. 46–52. DOI: 10.1109/CCSB58128.2022.00015.

[146] Paolo Missier, Khalid Belhajjame, and James Cheney. “The W3C PROV Family of Specifications for Modelling Provenance Metadata”. In: *Proceedings of the 16th International Conference on Extending Database Technology*. EDBT ’13. Genoa, Italy: Association for Computing Machinery, 2013, 773–776. ISBN: 9781450315975. DOI: 10.1145/2452376.2452478.

[147] Changhao Chenli et al. “ProvNet: Networked bi-directional blockchain for data sharing with verifiable provenance”. In: *Journal of Parallel and Distributed Computing* 166 (2022), pp. 32–44. ISSN: 0743-7315. DOI: https://doi.org/10.1016/j.jpdc.2022.04.003.

[148] Giuseppe Ateniese et al. “Redactable Blockchain - Or - Rewriting History in Bitcoin and Friends”. In: *Proceedings - 2nd IEEE European Symposium on Security and Privacy, EuroS and P 2017*. Institute of Electrical and Electronics Engineers Inc., 2017, 111 – 126. DOI: 10.1109/EuroSP.2017.37.

[149] Christina Brzuska et al. “Security of Sanitizable Signatures Revisited”. In: *Public Key Cryptography – PKC 2009*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 317–336. ISBN: 978-3-642-00468-1. DOI: https://doi.org/10.1007/978-3-642-00468-1_18.

[150] Hugo Krawczyk and Tal Rabin. “Chameleon hashing and signatures”. In: *Cryptology ePrint Archive* (1998).

[151] Jan Camenisch et al. “Chameleon-Hashes with Ephemeral Trapdoors”. In: *Public-Key Cryptography – PKC 2017*. Ed. by Serge Fehr. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017, pp. 152–182. ISBN: 978-3-662-54388-7. DOI: https://doi.org/10.1007/978-3-662-54388-7_6.

[152] David Megías, Jordi Serra-Ruiz, and Mehdi Fallahpour. “Efficient self-synchronised blind audio watermarking system based on time domain and FFT amplitude modification”. In: *Signal Processing* 90.12 (2010), pp. 3078–3092. ISSN: 0165-1684. DOI: https://doi.org/10.1016/j.sigpro.2010.05.012. URL: https://www.sciencedirect.com/science/article/pii/S0165168410002148.

[153] Tanya Koohpayeh Araghi, David Megías, and Andrea Rosales. “Evaluation and Analysis of Reversible Watermarking Techniques in WSN for Secure, Lightweight Design of IoT Applications: A Survey”. In: *Advances in Information and Communication*. Ed. by Kohei Arai. Cham: Springer Nature Switzerland, 2023, pp. 695–708. ISBN: 978-3-031-28073-3.

[154] Salmin Sultana, Elisa Bertino, and Mohamed Shehab. “A Provenance Based Mechanism to Identify Malicious Packet Dropping Adversaries in Sensor Networks”. In: *2011 31st International Conference on Distributed Computing Systems Workshops*. 2011, pp. 332–338. DOI: 10.1109/ICDCSW.2011.54.

[155] Arun Kanuparthi, Ramesh Karri, and Sateesh Addepalli. “Hardware and Embedded Security in the Context of Internet of Things”. In: *Proceedings of the 2013 ACM Workshop on Security, Privacy & Dependability for Cyber Vehicles*. CyCAR ’13. Berlin, Germany: Association for Computing Machinery, 2013, 61–64. ISBN: 9781450324878. DOI: 10.1145/2517968.2517976.

[156] Muhammad Naveed Aman, Mohammed Haroon Basheer, and Biplab Sikdar. “Data Provenance for IoT With Light Weight Authentication and Privacy Preservation”. In: *IEEE Internet of Things Journal* 6.6 (2019), pp. 10441–10457. DOI: 10.1109/JIOT.2019.2939286.

[157] Muhammad Naveed Aman, Mohamed Haroon Basheer, and Biplab Sikdar. “A Lightweight Protocol for Secure Data Provenance in the Internet of Things Using Wireless Fingerprints”. In: *IEEE Systems Journal* 15.2 (2021), pp. 2948–2958. DOI: 10.1109/JSYST.2020.3000269.

[158] Hala Hamadeh and Akhilesh Tyagi. “An FPGA Implementation of Privacy Preserving Data Provenance Model Based on PUF for Secure Internet of Things”. In: *SN Computer Science* 2.2 (2021), p. 65. ISSN: 2661-8907. DOI: 10.1007/s42979-020-00428-0.

[159] Byeong-Mo Chang et al. “SmartProvenance: User-friendly provenance system for internet of things applications based on event flow graphs”. In: *IET Software* 16 (2022), 576–602. DOI: 10.1049/sfw2.12071.

[160] Fariha Tasmin Jaigirdar et al. “Security-aware provenance for transparency in IoT data propagation”. English. In: *IEEE Access* 11 (May 2023). Publisher Copyright: Author, pp. 55677–55691. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2023.3280928.

[161] Fariha Tasmin Jaigirdar, Carsten Rudolph, and Chris Bain. “Prov-IoT: A Security-Aware IoT Provenance Model”. In: *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. 2020, pp. 1360–1367. DOI: 10.1109/TrustCom50675.2020.00183.

[162] Luc Moreau et al. “The Open Provenance Model core specification (v1.1)”. In: *Future Generation Computer Systems* 27.6 (2011), pp. 743–756. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2010.07.005.

[163] Mabrook S. Al-Rakhami and Majed Al-Mashari. “ProChain: Provenance-Aware Traceability Framework for IoT-Based Supply Chain Systems”. In: *IEEE Access* 10 (2022), pp. 3631–3642. DOI: 10.1109/ACCESS.2021.3135371.

[164] Wellington Fernandes Silvano and Roderval Marcelino. “Iota Tangle: A cryptocurrency to communicate Internet-of-Things data”. In: *Future Gener. Comput. Syst.* 112 (2020), pp. 307–319. URL: https://api.semanticscholar.org/CorpusID:219753144.

[165] Richard Lomotey et al. “Data Verification and Privacy in IoT Architecture”. In: *2019 IEEE World Congress on Services (SERVICES)*. Vol. 2642-939X. 2019, pp. 66–71. DOI: 10.1109/SERVICES.2019.00026.

[166] Steffen Remus and Chris Biemann. “Three Knowledge-Free Methods for Automatic Lexical Chain Extraction”. In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Ed. by Lucy Vanderwende, Hal Daumé III, and Katrin Kirchhoff. Atlanta, Georgia: Association for Computational Linguistics, June 2013, pp. 989–999. URL: https://aclanthology.org/N13-1119.

[167] Richard K. Lomotey, Joseph C. Pry, and Chenshean Chai. “Traceability and visual analytics for the Internet-of-Things (IoT) architecture”. In: *World Wide Web* 21.1 (2018), pp. 7–32. ISSN: 1573-1413. DOI: 10.1007/s11280-017-0461-1.

[168] Yi Gao et al. “Pathfinder: Robust path reconstruction in large scale sensor networks with lossy links”. In: *2013 21st IEEE International Conference on Network Protocols (ICNP)*. 2013, pp. 1–10. DOI: 10.1109/ICNP.2013.6733600.

[169] Matthias Keller, Jan Beutel, and Lothar Thiele. “How Was Your Journey? Uncovering Routing Dynamics in Deployed Sensor Networks with Multi-Hop Network Tomography”. In: *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*. SenSys ’12. Toronto, Ontario, Canada: Association for Computing Machinery, 2012, 15–28. ISBN: 9781450311694. DOI: 10.1145/2426656.2426659.

[170] Lakshminarayana Sadineni, Emmanuel S. Pilli, and Ramesh Babu Battula. “ProvLink-IoT: A novel provenance model for Link-Layer Forensics in IoT networks”. In: *Forensic Science International: Digital Investigation* 46 (2023), p. 301600. ISSN: 2666-2817. DOI: https://doi.org/10.1016/j.fsidi.2023.301600.

[171] Adel Alkhalil, Rabie Ramadan, and Aakash Ahmad. “Bio-inspired Think-and-Share Optimization for Big Data Provenance in Wireless Sensor Networks”. In: *International Journal of Advanced Computer Science and Applications* 10.6 (2019). DOI: 10.14569/IJACSA.2019.0100650.

[172] Qi Wang et al. “Fear and Logging in the Internet of Things”. In: *Network and Distributed System Security Symposium*. 2018, pp. 1–15. DOI: 10.14722/NDSS.2018.23282.

[173] Paul Groth and Luc Moreau. *PROV-Overview: An Overview of the PROV Family of Documents*. 2013. URL: https://www.w3.org/TR/2013/NOTE-prov-overview-20130430/.

[174] Mahmoud Elkhodr and Belal Alsinglawi. “Data provenance and trust establishment in the Internet of Things”. In: *Security and Privacy* 3 (Nov. 2020), pp. 1–11. DOI: 10.1002/spy2.99.

[175] Mahmoud Elkhodr, Seyed Shahrestani, and Hon Cheung. “A Middleware for the Internet of Things”. In: *International Journal of Computer Networks and Communications* 8 (Apr. 2016), pp. 159–178. DOI: 10.5121/ijcnc.2016.8214.

[176] Xixun Yu, Zheng Yan, and Athanasios V. Vasilakos. “A Survey of Verifiable Computation”. In: *Mobile Networks and Applications* 22.3 (2017), pp. 438–453. ISSN: 1572-8153. DOI: 10.1007/s11036-017-0872-3.

[177] Zheng Yan, Xixun Yu, and Wenxiu Ding. “Context-Aware Verifiable Cloud Computing”. In: *IEEE Access* 5 (2017), pp. 2211–2227. DOI: 10.1109/ACCESS.2017.2666839.

[178] Xixun Yu, Zheng Yan, and Rui Zhang. “Verifiable outsourced computation over encrypted data”. In: *Information Sciences* 479 (2019), pp. 372–385. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2018.12.022.

[179] I. Foster. “The virtual data grid: a new model and architecture for data-intensive collaboration”. In: *15th International Conference on Scientific and Statistical Database Management, 2003.* 2003, pp. 11–. DOI: 10.1109/SSDM.2003.1214945.

[180] H. V. Jagadish and Frank Olken. “Database Management for Life Sciences Research”. In: *SIGMOD Rec.* 33.2 (2004), 15–20. ISSN: 0163-5808. DOI: 10.1145/1024694.1024697.

[181] Scott Jensen et al. “Provenance Capture and Use in a Satellite Data Processing Pipeline”. In: *IEEE Transactions on Geoscience and Remote Sensing* 51.11 (2013), pp. 5090–5097. DOI: 10.1109/TGRS.2013.2266929.

[182] Hadi Sabaa and Brajendra Panda. “Data authentication and provenance management”. In: *2007 2nd International Conference on Digital Information Management*. Vol. 1. 2007, pp. 309–314. DOI: 10.1109/ICDIM.2007.4444241.

[183] Salmin Sultana, Mohamed Shehab, and Elisa Bertino. “Secure Provenance Transmission for Streaming Data”. In: *IEEE Transactions on Knowledge and Data Engineering* 25.8 (2013), pp. 1890–1903. DOI: 10.1109/TKDE.2012.31.

[184] Adel Alkhalil and Rabie A. Ramadan. “IoT Data Provenance Implementation Challenges”. In: *Procedia Computer Science* 109 (2017). 8th International Conference on Ambient Systems, Networks and Technologies, ANT-2017 and the 7th International Conference on Sustainable Energy Information Technology, SEIT 2017, 16-19 May 2017, Madeira, Portugal, pp. 1134–1139. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2017.05.436.

[185] Sabah Suhail et al. “Introducing Secure Provenance in IoT: Requirements and Challenges”. In: *2016 International Workshop on Secure Internet of Things (SIoT)*. 2016, pp. 39–46. DOI: 10.1109/SIoT.2016.011.

[186] Sana Ullah Jan et al. “Toward a Lightweight Intrusion Detection System for the Internet of Things”. In: *IEEE Access* 7 (2019), pp. 42450–42471. DOI: 10.1109/ACCESS.2019.2907965.

[187] Vinayakumar Ravi, Rajasekhar Chaganti, and Mamoun Alazab. “Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system”. In: *Computers and Electrical Engineering* 102 (2022), p. 108156. ISSN: 0045-7906. DOI: https://doi.org/10.1016/j.compeleceng.2022.108156.

[188] Yamarthi Narasimha Rao and Kunda Suresh Babu. “An Imbalanced Generative Adversarial Network-Based Approach for Network Intrusion Detection in an Imbalanced Dataset”. In: *Sensors* 23.1 (2023). ISSN: 1424-8220. DOI: 10.3390/s23010550.

[189] Omar Alghushairy et al. “An Efficient Support Vector Machine Algorithm Based Network Outlier Detection System”. In: *IEEE Access* 12 (2024), pp. 24428–24441. DOI: 10.1109/ACCESS.2024.3364400.

[190] Arezou Soltani Panah et al. “In the shadows we trust: A secure aggregation tolerant watermark for data streams”. In: *2015 IEEE 16th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. 2015, pp. 1–9. DOI: 10.1109/WoWMoM.2015.7158149.

[191] Ragib Hasan, Radu Sion, and Marianne Winslett. “The Case of the Fake Picasso: Preventing History Forgery with Secure Provenance”. In: *7th USENIX Conference on File and Storage Technologies (FAST 09)*. USENIX Association, Feb. 2009, pp. 1–14. URL: http://usenix.org/event/fast09/tech/full_papers/hasan/hasan.pdf.

[192] R Manjunatha Prasad and Shivaprakash Koliwad. “A robust wavelet-based watermarking scheme for copyright protection of digital images”. In: *2010 Second International conference on Computing, Communication and Networking Technologies*. 2010, pp. 1–9. DOI: 10.1109/ICCCNT.2010.5591586.

[193] Mehdi Fallahpour and David Megías. “Audio Watermarking Based on Fibonacci Numbers”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 23.8 (2015), pp. 1273–1282. DOI: 10.1109/TASLP.2015.2430818.

[194] Reema Jain and Manish Jain. “Digital image watermarking using 3-level DWT and FFT via image compression”. In: *International Journal of Computer Applications* 124.16 (2015). DOI: 10.5120/ijca2015905808.

[195] Chhaya S Gosavi and Suresh N Mali. “Video authentication and copyright protection using unique watermark generation technique and singular value decomposition”. In: *International Journal of Computer Applications* 123.3 (2015). DOI: 10.5120/ijca2015905255.

[196] Aihab Khan, Syed Afaq Husain, et al. “A fragile zero watermarking scheme to detect and characterize malicious modifications in database relations”. In: *The Scientific World Journal* 2013 (2013). DOI: https://doi.org/10.1155/2013/796726.

[197] Gbadebo Ayoade et al. “Evolving Advanced Persistent Threat Detection using Provenance Graph and Metric Learning”. In: *2020 IEEE Conference on Communications and Network Security (CNS)*. 2020, pp. 1–9. DOI: 10.1109/CNS48642.2020.9162264.

[198] Mathieu Barre, Ashish Gehani, and Vinod Yegneswaran. “Mining Data Provenance to Detect Advanced Persistent Threats”. In: *11th International Workshop on Theory and Practice of Provenance (TaPP 2019)*. Philadelphia, PA: USENIX Association, June 2019, pp. 1–11. URL: https://www.usenix.org/conference/tapp2019/presentation/barre.

[199] Ghita Berrada and James Cheney. “Aggregating unsupervised provenance anomaly detectors”. In: *11th International Workshop on Theory and Practice of Provenance (TaPP 2019)*. Philadelphia, PA: USENIX Association, June 2019, pp. 1–9. URL: https://www.usenix.org/conference/tapp2019/presentation/berrada.

[200] Yulai Xie et al. “Unifying intrusion detection and forensic analysis via provenance awareness”. In: *Future Generation Computer Systems* 61 (2016), pp. 26–36. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2016.02.005.

[201] Yulai Xie et al. “Pagoda: A Hybrid Approach to Enable Efficient Real-Time Provenance Based Intrusion Detection in Big Data Environments”. In: *IEEE Transactions on Dependable and Secure Computing* 17.6 (2020), pp. 1283–1296. DOI: 10.1109/TDSC.2018.2867595.

[202] Xueyuan Han et al. “FRAPpuccino: Fault-detection through Runtime Analysis of Provenance”. In: *9th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 17)*. Santa Clara, CA: USENIX Association, July 2017, pp. 1–7. URL: https://www.usenix.org/conference/hotcloud17/program/presentation/han.

[203] Min Du et al. “DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning”. In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. CCS ’17. Dallas, Texas, USA: Association for Computing Machinery, 2017, 1285–1298. ISBN: 9781450349468. DOI: 10.1145/3133956.3134015.

[204] Peng Gao et al. “AIQL: Enabling Efficient Attack Investigation from System Monitoring Data”. In: *2018 USENIX Annual Technical Conference (USENIX ATC 18)*. Boston, MA: USENIX Association, July 2018, pp. 113–126. ISBN: ISBN 978-1-939133-01-4. URL: https://www.usenix.org/conference/atc18/presentation/gao.

[205] Yulai Xie et al. “P-Gaussian: Provenance-Based Gaussian Distribution for Detecting Intrusion Behavior Variants Using High Efficient and Real Time Memory Databases”. In: *IEEE Transactions on Dependable and Secure Computing* 18.6 (2021), pp. 2658–2674. DOI: 10.1109/TDSC.2019.2960353.

[206] Ghita Berrada et al. “A baseline for unsupervised advanced persistent threat detection in system-level provenance”. In: *Future Generation Computer Systems* 108 (2020), pp. 401–413. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2020.02.015.

[207] Peng Gao et al. “SAQL: A Stream-based Query System for Real-Time Abnormal System Behavior Detection”. In: *27th USENIX Security Symposium (USENIX Security 18)*. Baltimore, MD: USENIX Association, Aug. 2018, pp. 639–656. ISBN: 978-1-939133-04-5. URL: https://www.usenix.org/conference/usenixsecurity18/presentation/gao-peng.

[208] Mathieu Garchery and Michael Granitzer. “ADSAGE: Anomaly Detection in Sequences of Attributed Graph Edges applied to insider threat detection at fine-grained level”. In: *CoRR* abs/2007.06985 (2020). arXiv: 2007.06985. URL: https://arxiv.org/abs/2007.06985.

[209] Wajih Ul Hassan, Adam Bates, and Daniel Marino. “Tactical Provenance Analysis for Endpoint Detection and Response Systems”. In: *2020 IEEE Symposium on Security and Privacy (SP)*. 2020, pp. 1172–1189. DOI: 10.1109/SP40000.2020.00096.

[210] Qi Wang et al. “You Are What You Do: Hunting Stealthy Malware via Data Provenance Analysis”. In: *Proceedings 2020 Network and Distributed System Security Symposium* (2020). URL: https://www.ndss-symposium.org/wp-content/uploads/2020/02/24167-paper.pdf.

[211] Michael Zipperle et al. “Provenance-Based Intrusion Detection Systems: A Survey”. In: *ACM Comput. Surv.* 55.7 (2022), pp. 1–36. ISSN: 0360-0300. DOI: 10.1145/3539605.

[212] A. Meddeb. “Internet of things standards: who stands out from the crowd?” In: *IEEE Communications Magazine* 54.7 (2016), pp. 40–47. ISSN: 1558-1896. DOI: 10.1109/MCOM.2016.7514162.

[213] Shahid Raza, Linus Wallgren, and Thiemo Voigt. “SVELTE: Real-time intrusion detection in the Internet of Things”. In: *Ad Hoc Networks* 11.8 (2013), pp. 2661 –2674. ISSN: 1570-8705. DOI: https://doi.org/10.1016/j.adhoc.2013.04.014.

[214] Huiping Guo, Yingjiu Li, and Sushil Jajodia. “Chaining watermarks for detecting malicious modifications to streaming data”. In: *Information Sciences* 177.1 (2007), pp. 281 –298. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2006.03.014. URL: http://www.sciencedirect.com/science/article/pii/S0020025506000855.

[215] Wei Zhang et al. “Secure data aggregation in wireless sensor networks: A watermark based authentication supportive approach”. In: *Pervasive and Mobile Computing* 4.5 (2008), pp. 658 –680. ISSN: 1574-1192. DOI: https://doi.org/10.1016/j.pmcj.2008.05.005. URL: http://www.sciencedirect.com/science/article/pii/S157411920800059X.

[216] Sean Weerakkody et al. “Resilient Control in Cyber-Physical Systems: Countering Uncertainty, Constraints, and Adversarial Behavior”. In: *Foundations and Trends® in Systems and Control* 7.1-2 (2019), pp. 1–252. ISSN: 2325-6818. DOI: 10.1561/2600000018. URL: http://dx.doi.org/10.1561/2600000018.

[217] Khizar Hameed et al. “Towards a formally verified zero watermarking scheme for data integrity in the Internet of Things based-wireless sensor networks”. In: *Future Generation Computer Systems* 82 (2018), pp. 274 –289. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2017.12.009.

[218] Zhang Guoyin et al. “A New Digital Watermarking Method for Data Integrity Protection in the Perception Layer of IoT”. In: *Security and Communication Networks* 2017 (2017). ISSN: 1939-0114. DOI: https://doi.org/10.1155/2017/3126010.

[219] Arwa Alromih, Mznah Al-Rodhaan, and Yuan Tian. “A Randomized Watermarking Technique for Detecting Malicious Data Injection Attacks in Heterogeneous Wireless Sensor Networks for Internet of Things Applications”. In: *Sensors (Basel, Switzerland)* 18.12 (2018), p. 4346. ISSN: 1424-8220. DOI: 10.3390/s18124346. URL: https://pubmed.ncbi.nlm.nih.gov/30544877.

[220] Baowei Wang et al. “A Dual-Chaining Watermark Scheme for Data Integrity Protection in Internet of Things”. In: *Computers, Materials & Continua* 58.3 (2019), pp. 679–695. ISSN: 1546-2226. DOI: 10.32604/cmc.2019.06106. URL: http://www.techscience.com/cmc/v58n3/23040.

[221] Xi Shi and Di Xiao. “A Reversible Watermarking Authentication Scheme for Wireless Sensor Networks”. In: *Inf. Sci.* 240 (Aug. 2013), 173–183. ISSN: 0020-0255. DOI: 10.1016/j.ins.2013.03.031. URL: https://doi.org/10.1016/j.ins.2013.03.031.

[222] Qun Ding et al. “A Reversible Watermarking Scheme Based on Difference Expansion for Wireless Sensor Networks”. In: *International Journal of Grid Distribution Computing* 8.2 (2015), pp. 143–154. DOI: 10.14257/ijgdc.2015.8.2.14. URL: http://dx.doi.org/10.14257/ijgdc.2015.8.2.14.

[223] Xingming Sun et al. “Digital Watermarking Method for Data Integrity Protection in Wireless Sensor Networks”. In: *International Journal of Security and Its Applications* 7.4 (2013), pp. 407–416.

[224] Nuno Monteiro et al. “Interference analysis in a LTE-A HetNet scenario: Coordination vs. uncoordination”. In: *Wireless VITAE 2013*. IEEE. 2013, pp. 1–5.

[225] Jie Cui et al. “Data aggregation with end-to-end confidentiality and integrity for large-scale wireless sensor networks”. In: *Peer-to-Peer Networking and Applications* 11.5 (2018), pp. 1022–1037. DOI: https://doi.org/10.1007/s12083-017-0581-5.

[226] Prasanth Ganesan et al. “Analyzing and Modeling Encryption Overhead for Sensor Network Nodes”. In: *Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications*. WSNA ’03. San Diego, CA, USA: Association for Computing Machinery, 2003, 151–159. ISBN: 1581137648. DOI: 10.1145/941350.941372.

[227] Tanya Gazelle Roosta. “Attacks and Defenses of Ubiquitous Sensor Networks”. AAI3331772. PhD thesis. USA: University of California at Berkeley, 2008. ISBN: 9780549832898. URL: https://dl.acm.org/doi/book/10.5555/1559508.

[228] Claude Castelluccia et al. “Efficient and Provably Secure Aggregation of Encrypted Data in Wireless Sensor Networks”. In: *ACM Trans. Sen. Netw.* 5.3 (2009). ISSN: 1550-4859. DOI: 10.1145/1525856.1525858.

[229] Malka Nisha Halgamuge et al. “An estimation of sensor energy consumption”. In: *Progress In Electromagnetics Research B* 12.12 (2009), 259 – 295. DOI: 10.2528/PIERB08122303.

[230] C. Fu et al. “An energy balanced algorithm of LEACH protocol in WSN”. In: *IJCS* 10.1 (2013), 354 – 359. URL: https://www.proquest.com/scholarly-journals/energy-balanced-algorithm-leach-protocol-wsn/docview/1441692218/se-2.

[231] Malka N. Halgamuge et al. “An Estimation of Sensor Energy Consumption”. In: *Progress In Electromagnetics Research B* 12 (2009), pp. 259–295. DOI: doi:10.2528/PIERB08122303.

[232] M.J. Miller and N.H. Vaidya. “A MAC protocol to reduce sensor network energy consumption using a wakeup radio”. In: *IEEE Transactions on Mobile Computing* 4.3 (2005), pp. 228–242. DOI: 10.1109/TMC.2005.31.

[233] An Liu and Peng Ning. “TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks”. In: *2008 International Conference on Information Processing in Sensor Networks (ipsn 2008)*. 2008, pp. 245–256. DOI: 10.1109/IPSN.2008.47.

[234] Chris Karlof, Naveen Sastry, and David Wagner. “TinySec: A Link Layer Security Architecture for Wireless Sensor Networks”. In: *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*. SenSys ’04. Baltimore, MD, USA: Association for Computing Machinery, 2004, 162–175. ISBN: 1581138792. DOI: 10.1145/1031495.1031515.

[235] *NSL-KDD Dataset*. https://www.unb.ca/cic/datasets/index.html. Accessed: [13 April 2024].

[236] Sobin Soniya Sathiyadhas and Maria Celestin Vigila Soosai Antony. “A network intrusion detection system in cloud computing environment using dragonfly improved invasive weed optimization integrated Shepard convolutional neural network”. In: *International Journal of Adaptive Control and Signal Processing* 36.5 (2022), pp. 1060–1076. DOI: https://doi.org/10.1002/acs.3386.

[237] Fredrik Osterlind et al. “Cross-Level Sensor Network Simulation with COOJA”. In: *Proceedings. 2006 31st IEEE Conference on Local Computer Networks*. 2006, pp. 641–648. DOI: 10.1109/LCN.2006.322172.

[238] Enrico Perla et al. “PowerTOSSIM z: Realistic Energy Modelling for Wireless Sensor Network Environments”. In: *Proceedings of the 3nd ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks*. PM2HW2N ’08. New York, NY, USA: Association for Computing Machinery, 2008, 35–42. ISBN: 9781605582399. DOI: 10.1145/1454630.1454636.

[239] Bryan Parno et al. “Pinocchio: Nearly Practical Verifiable Computation”. In: *Commun. ACM* 59.2 (2016), 103–112. ISSN: 0001-0782. DOI: 10.1145/2856449.

[240] Ahmed Kosba, Charalampos Papamanthou, and Elaine Shi. “xJsnark: A Framework for Efficient Verifiable Computation”. In: *2018 IEEE Symposium on Security and Privacy (SP)*. 2018, pp. 944–961. DOI: 10.1109/SP.2018.00018.

[241] Christian Cachin. “Architecture of the hyperledger blockchain fabric”. In: *Workshop on distributed cryptocurrencies and consensus ledgers*. Vol. 310. 2016, pp. 4–8. URL: https://www.zurich.ibm.com/dccl/papers/cachin_dccl.pdf.

[242] João Sousa, Alysson Bessani, and Marko Vukolic. "A Byzantine Fault-Tolerant Ordering Service for the Hyperledger Fabric Blockchain Platform". In: *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. 2018, pp. 51–58. DOI: 10.1109/DSN.2018.00018.

[243] Alessio Rossi et al. "A Public Dataset of 24-h Multi-Levels Psycho-Physiological Responses in Young Healthy Adults". In: *Data* 5.4 (2020). ISSN: 2306-5729. DOI: 10.3390/data5040091.

[244] SmartThings. *SmartThings public GitHub Repo*. Accessed 1 August 2023. URL: https://github.com/SmartThingsCommunity/SmartThingsPublic/tree/master/smartapps.

[245] Samsung Electronics Co. *SmartThings Groovy IDE*. Accessed 1 August 2023. URL: https://graph.api.smartthings.com/.

[246] Omprakash Gnawali et al. "Collection tree protocol". English (US). In: *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems, SenSys 2009*. 2009, pp. 1–14. ISBN: 9781605587486. DOI: 10.1145/1644038.1644040.

[247] Samsung. *SmartThings IDE*. 2017. URL: https://graph.api.smartthings.com/.