Computer Systems Science & Engineering
DOI:10.32604/csse.2022.018801
Article

Tech Science Press

IoT Wireless Intrusion Detection and Network Traffic Analysis

Vasaki Ponnusamy[1], Aun Yichiet[1], NZ Jhanjhi[2,*], Mamoona humayun[3] and Maram Fahhad Almufareh[3]

[1]University Tunku Abdul Rahman, Kampar, 31900, Malaysia
[2]School of Computer Science and Engineering (SCE), Taylor's University, Selangor, Malaysia
[3]Department of Information Systems, College of Computer and Information Sciences, Jouf University, Al-Jouf, Saudi Arabia
[*]Corresponding Author: NZ Jhanjhi. Email: noorzaman.jhanjhi@taylors.edu.my
Received: 22 March 2021; Accepted: 05 May 2021

Abstract: Enhancement in wireless networks had given users the ability to use the Internet without a physical connection to the router. Almost every Internet of Things (IoT) devices such as smartphones, drones, and cameras use wireless technology (Infrared, Bluetooth, IrDA, IEEE 802.11, etc.) to establish multiple inter-device connections simultaneously. With the flexibility of the wireless network, one can set up numerous ad-hoc networks on-demand, connecting hundreds to thousands of users, increasing productivity and profitability significantly. However, the number of network attacks in wireless networks that exploit such flexibilities in setting and tearing down networks has become very alarming. Perpetrators can launch attacks since there is no first line of defense in an *ad hoc* network setup besides the standard IEEE802.11 WPA2 authentication. One feasible countermeasure is to deploy intrusion detection systems at the edge of these *ad hoc* networks (Network-based IDS) or at the node level (Host-based IDS). The challenge here is that there is no readily available benchmark data available for IoT network traffic. Creating this benchmark data is very tedious as IoT can work on multiple platforms and networks, and crafting and labelling such dataset is very labor-intensive. This research aims to study the characteristics of existing datasets available such as KDD-Cup and NSL-KDD, and their suitability for wireless IDS implementation. We hypothesize that network features are parametrically different depending on the types of network and assigning weight dynamically to these features can potentially improve the subsequent threat classifications. This paper analyses packet and flow features for the data packet captured on a wireless network rather than a wired network. Combining domain heuristcs and early classification results, the paper had identified 19 header fields exclusive to wireless network that contain high information gain to be used as ML features in Wireless IDS.

Keywords: IoT; machine learning; traffic features; IDS; KDD-CUP; NSL-KDD

1 Introduction

Humayun et al. [1] has mentioned that the automatic exchange of information between two systems or two devices without any manual input is the main objective of the Internet of Things (IoT). IoT is such a device that can easily trust other devices and exchange information, and this situation results in IoT devices becoming the target of attacks. Moreover, most IoT devices use existing wireless connections due to their convenience and flexibility without considering their weakness. A wireless access point usually is not configured for a secure operation which comes only with front end authentication. Like Distributed Denial of Service (DDoS), some common attacks are not preventable through traffic filtering since ICMP traffic is considered legitimate. Many computers start performing denial of service attack towards the same targeted server in distributed denial of service attacks. There are three types of DDOS attacks, application-layer DDOS attack, protocol DDOS attack, and volume-based DDOS attack. DDOS attacks can severely damage an organization's the business and network security. A DDOS attack can last anywhere from a few hours to several days, making the organizations website and network unreachable during the attack. To improve the IoT security on the network, an Intrusion Detection System (IDS) can be deployed to analyze the network traffic [2]. IDS is a system that monitors a network or a method for malicious activities and reports or alerts the user of the system. The intrusion detection system investigates application vulnerabilities and identifies abnormal activity and data injection in a system as they are designed to observe the activities in the system. The IDS helps the network administrator detect any malicious activity on the network and alerts the administrator to secure the data by taking appropriate actions against those attacks. To implement an effective IDS in a wireless environment, careful selection of datasets or network traffic is also of utmost importance. To that, this research presents an analysis of network traffic from the wired and wireless (IEEE802.11) environment. The study presented here can be contributing to future research, mainly for IoT and wireless security and researchers who wish to implement intrusion detection systems for their IoT networks. A careful selection of network traffic features can contribute towards an exemplary implementation of wireless networks IDS. Therefore, a comparison between the wired and wireless network and traffic characteristics is presented in the following sections, followed by traffic characteristics for wireless (IEEE802.11) networks

2 Classification of IDS

Two types of IDS are commonly deployed for intrusion detection, namely (1) Wired IDS and (2) Wireless IDS.

2.1 Wired Intrusion Detection System

A wired IDS is the standard IDS connected with all the network components (e.g. router, switch, IDS manager server, end devices) to perform intrusion detection processes [3]. The most common techniques used by wired IDS are signature-based, anomaly-based and hybrid technique. The signature-based method works by comparing the known information with the signatures database. Still, it cannot identify unknown attacks—the anomaly-based method used to compare current user activities against previously loaded logs of users. The bad review of this technique produces a larger number of false alarms because of irregular network and user behaviour. The hybrid detection technique can be used to combine the signature and anomaly-based detection techniques. The weakness of this technique is the complexity due to the integration of both signatures and anomalies. Tab. 1 shows a simple analysis of the intrusion detection technique.

The wired or standard IDS architecture used to connect all the devices with a cable. The IDS console will play the role to monitor and analyze the network traffic. When traffic or packet is coming from the internet, the router will pass the data to the IDS server; the IDS server would collect the traffic and perform the

analysis. The IDS do not drop any packets since the job of IDS is to collect and analyze the data. The wired IDS require more components and devices for the network setup, such as routers, switches, IDS consoles, IDS servers, and other end devices, as shown in Fig. 1.

Table 1: Analysis of intrusion detection techniques

IDS Technique	Detection time	Data source	Weakness
Signature based	Real-time	Network traffic	Unable to identify unknown attacks.
Anomaly based	Real-time	Network traffic, User behaviours	Consumes more resources for high level users.
Hybrid based	Real-time	Network packet, prior events	Complexity will increased due to integration of both, signatures and anomalies.

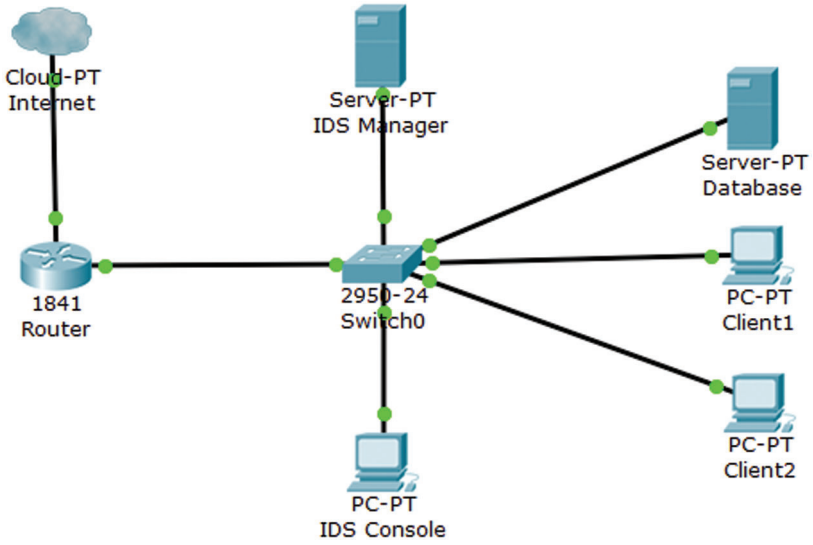

Figure 1: Wired IDS architecture

One of the weaknesses of traditional wired IDS for wireless implementation as in IoT is, it does not generally detect network intrusion from internal hosts of the network. Although it is possible to protect an organization's internal network from wireless attackers, there can be only one link between the wireless network and the main network, such a network intrusion system will not cover all of the traffic on the wireless network [4]. The traditional wired IDS may meet some challenges of securing the wireless network because it fundamentally ignores the monitoring of airspace from which most attacks are perpetrated. As a conclusion, the wired IDS is not suitable for deployment in the IoT network because most of the IoT devices are connected via wireless mediums such as *IEEE802.11*, *IEEE 802.15.4* and *IEEE 802.15.1*.

2.2 Wireless Intrusion Detection System

Compared with wired IDS, wireless IDS is more suitable for monitoring IoT networks. Wireless IDS is a better version of wired IDS because it has characteristics to be covered in a wireless network [3]. The wireless

network is vulnerable due to its open medium, dynamic changing of topology, cooperative algorithm, and lack of centralized monitoring. Therefore wired IDS is no longer suitable and sufficient. Compared to a wired IDS, wireless IDS can detect many possible attacks from the wireless access point due to an open area access network. According to [5], wireless IDS has more intrusion detection techniques to discover possible attacks on a wireless network. The methods included target system, detection technique, collection process, trust model, analysis and response to identify and analyze the network traffic. A wireless IDS has a more efficient method for investigating wireless network traffics.

The wireless IDS architecture looks like a wired IDS architecture, but it uses a wireless access point for network connectivity. The wireless IDS architecture is more suitable for IoT network due to the wireless sensor deployment. Furthermore, the typical components in a wireless IDS are the console, database server, and sensors, as shown in Fig. 2. In conclusion, wireless IDS is more suitable for investigating IoT network traffic from the technique used to the architecture and the network setup.

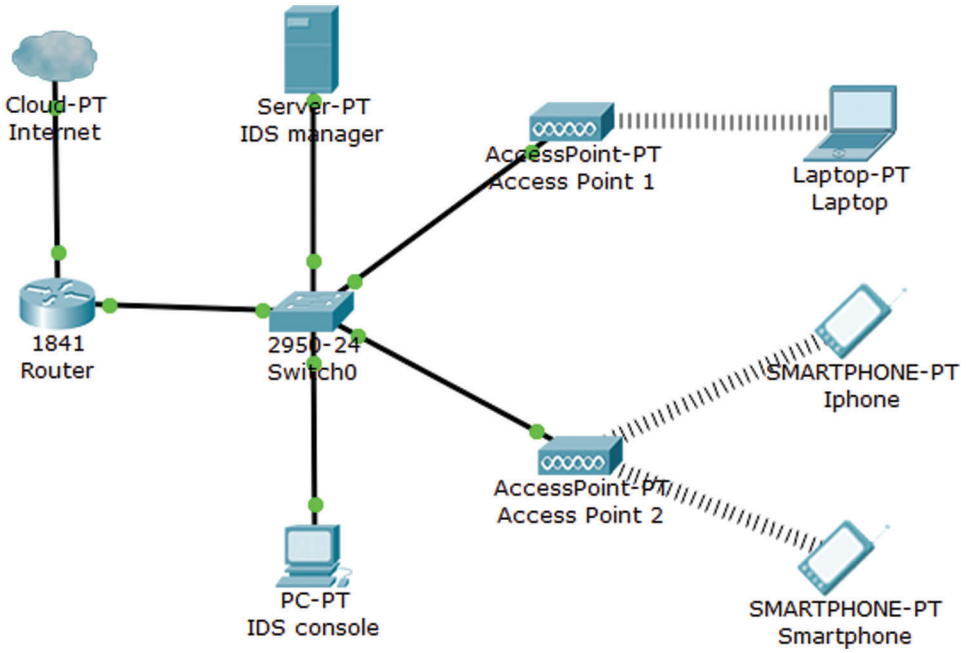

Figure 2: Wireless IDS architecture

3 Internet of Things Dataset

Network traffic dataset can be sniffed from both wired network and wireless network. There are considerable differences in the attacks that targets wired and wireless networks. Fadlullah et al. [6] states that a wired network has an access medium that is physically secured compared to the wireless network as it does not require the monitoring of airspace. The datasets used in most studies comprise sample datasets such as KDD Cup '99, NSL-KDD, and Kyoto 2006+ datasets. However, there is a lack of studies involving intrusion detection based on wireless networks, which states the essential parameters crucial in detecting intrusions in the wireless network for the classification algorithms. Furthermore, the wireless packets used as the dataset consist of multiple parameters and fields that require feature selection to be implemented in the algorithms to detect the intrusions in a network. This research aims to identify the critical parameters and fields required in a wireless network dataset to produce optimal results. Besides that, an analysis of traditional network traffic characteristics at the packet level, flow level, connection level and host level is studied to investigate if existing parameters are fit to use in a wireless network.

3.1 Network Traffic Analysis

Understanding the network data is fundamental before proceeding to the network traffic analysis process. Before investigating the network traffic dataset, understanding the type of network traffic data is crucial. Humayun et al. [7,8] have clearly summarized and compared these data categories. And since network traffic plays an important role in wireless intrusion detection design, the remaining sections focus on the types and techniques to analyze wireless traffic. Moreover, some of the challenges in using existing benchmark data for wireless intrusion detection is discussed in details. The network data has been categorized into 4 categories, that is at: packet, flow, connection, and host-level data. The atomic unit for network traffic are packets. These are captured by a specific application, Libpcap, WinPCap, Wireshark, and Libtrace [9].

3.2 Packet Level Data

Network applications generate traffic (packets) containing headers from multiple protocols through the encapsulation process. Some examples of these protocols are Transmission Control Protocol (TCP), User Datagram Protocol (UDP), Internet Control Message Protocol (ICMP). See Tab. 2 for a snippet of packet payload information and packet activity information. This traffic is then used to detect the DDoS and worm attacks. The packet-level data can commonly collected using TCPDump, Snort [10] and Nmap [11]. TCPDump [12] is similar to Wireshark but without a GUI, whereas snort is a standard tool used in intrusion detection while capturing packets simultaneously. Nmap is used for host discovery, service and operating system detection.

Table 2: Packet headers of datasets [8]

IP Header (IPV4)	
Internet Header Length	The number of 32-bit words in the header
Total Length	The entire packet size, including header and data in bytes
Time To Live	This filed limits a datagram's lifetime in hops (or time)
Protocol	The protocol used in the data portion of the IP datagram
Source address	This field is the IPV4 address of the sender of the datagram
Destination address	This field is the IPV4 address of the receiver of the datagram
TCP Packet	
Source port	Identifies the sending port
Destination port	Identifies the receiving port
Sequence number	Initial or accumulated sequence number
Acknowledgement number	The next sequence number that the receiver is expecting
Data offset	Specifies the size of the TCP header in 32-bit words
Flags (control bits)	NS, CWR, ECE, URG, ACK, PSH, RST, SYN, FIN
UDP Packet	
Source port	Identifies the sending port
Destination port	Identifies the receiving port
Length	The length in bytes of the UDP header and UDP data
ICMP Packet	
Type	Control (e.g. ping, destination unreachable, trace route)
Code	Details with the type
Rest of Header	More details

On the other hand, port mirroring can be another method that can be used for data collection. This is a hardware-based data collection whereby packet forwarding devices can be used to forward packets from one port to another port where packet capture devices can be connected. By using this method, the entire incoming and outgoing packets can be viewed within a whole network. But this approach requires enough bandwidth as mirroring can cause loss and packet delay [13]. Based on the detailed analysis by Mahoney et al. [14] on the important features for detecting attacks, the authors have identified 33 header features crucial in detecting an anomaly in networks. The author has performed a very comprehensive analysis of the types of attacks detected, their occurrences in terms of percentage, and the header fields that contribute to detection effectiveness. A detailed analysis of these seven fields and their significance to detecting attacks is well presented by Jing et al. [15]. Out of that, [15] have identified that source/destination IP address, source/destination port, time to live, timestamp, a packet payload, packet size and the number of packets is crucial in detecting attacks. As for the IP address, the way the addresses are distributed, and their changing patterns are useful for detecting botnet attacks. Whereas similarly, the changing and distribution of port address can be used to detect worn attacks. IP spoofing attack can be seen by looking at the TTL value because a TTL can show if the IP address is coming from an internal spoofed IP address. At the same time, a timestamp value can tell the inter-arrival time and round trip time. A non-uniform pattern in this can indicate a non-repudiation attack. Packet payload typically carries important information destined to the victim network.

For example, a deep inspection of the packet application header and its payload can indicate whether the header carries malware. One weakness of such a method is that some protocols such as SSL and TLS encrypt the payload information. Therefore, it is hard to detect malicious information carried by the payload. The packet size can indicate whether the payload information carried is initiated by a bot because the attack packets coming from all the bots would have almost similar packet size. A DDoS attack can be detected by analyzing the packets count, based on the inbound and outbound packet counts. Some examples include checking ICMP's request/reply count; traffic distribution among different network protocols; or the ratio of TCP packets with different flags value [15]. With the high-speed networks, packet-level data collection may require expensive hardware, and sophisticated encryption and obfuscation method may hinder the packet inspection.

3.3 Packet Level Data

A flow-level data is usually used by using a flow-key aggregated with the relevant data depending on the application. The applications can be summarized as network/application/host monitoring, intrusion detection, security awareness and network application classification. A collection of a flow-based dataset for intrusion detection is outlined in Umer et al. [16]. Flow-based data can be collected either by using depth-first or breadth-first methods. Choosing a specific flow-key to aggregate data in the first approach, and the latter collects as much information as possible to have a more comprehensive view of the network traffic [17]. A complete discussion of flow collection (Fig. 3), types of flow and how to analyze is discussed in Li et al. [17–20].

A flow is defined as a unidirectional sequence of packets that belongs to a same TCP session. The purposes of flow are to provide an overview of network traffic and attempt to deal with the encrypted packets. Flow level data comprises (as shown in Tab. 3) a tuple with flow-key aggregated with a collection of information such as srcIP, dstIP, src_port, dest_port as in the case of Cisco routers [18]. The reason for collecting flow-level data is to reduce the amount of network traffic to be analysed. Flow data is typically not helpful for deep analysis that requires packet payload. Flow-level data is the statistical information about the flow which comprises of flow count (number of discharges with the same flow key), flow type (ICMP, TCP, UDP, HTTP, DNS etc.), flow size, flow direction (inflow, outflow), flow duration (time between first packet arrival time to end of flow time) and flow rate.

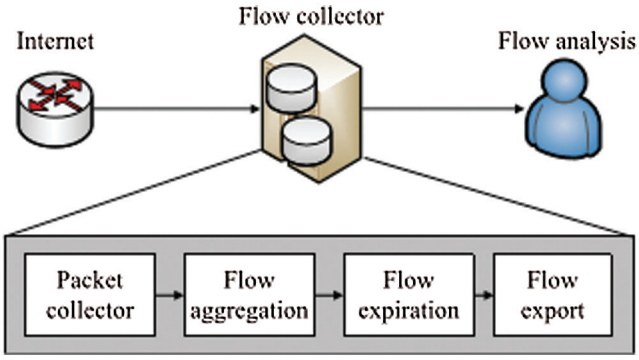

Figure 3: Flow Collection Process [15]

Table 3: NetFlow packet header for cybersecurity [8]

NetFlow Data — Simple Network Management Protocol (SNMP)	
Ingress Interface	Router Information
Source IP address	na
Destination IP address	na
IP protocol	IP protocol number
Source Port	UDP or TCP ports, 0 for other ports
Destination Port	UDP or TCP ports, 0 for other ports
IP type of service	Priority level of the flow
NetFlow Data — Flow Statistics	
IP protocol	IP protocol number
Source IP address	na
Destination IP address	na
Source Port	na
Destination Port	na
Bytes per packet	The flow analyzer captures their statistics
Packets per flow	Number of packets in the flow
TCP flags	NS, CWR, ECF, URG, ACK, PSH, RST SYN, FIN

3.4 Connection Level Data

The connection-level data records global information between two IP addresses from the viewpoint of a particular network, providing a finer level of network traffic granularity than the flow-level data. Using connection-level data, packet-level data, or flow-level data, we may obtain detailed information about network activities. Connection size (size of packets and length of flow) can be summarised as connection period (time from connection establishment to connection termination), connection count (number of connections per unit time), and connection form (TCP, UDP ICMP etc.).

3.5 Host Level Data

Any internal changes in the host can be seen in the host-level data, and most attacks have a direct effect on the host's reliability. Internal attacks such as unauthorised logging or entry, file system alteration, and privilege escalation can be detected using host-level data. They are commonly used in HIDS for monitoring abnormalities in the internals. Host level data can be collected using open source tools like Collect in Linux machines [21] and Load runner in Windows machines [22]. The collected data comprises CPU and memory usage and operation log (equipment and application operation log). Operation logs include events with the equipment such as mouse click, keyboard, cursor changes etc. In contrast, the application log relates directly to a specific application on the local port creation/ destruction, login attempts, system calls and usage of software events etc.

3.6 Summary of Network Traffic Dataset

Tab. 4 summarize and compares traffic in terms of data types; based on their strengths and weaknesses. Packet level data is useful because it has a full view of the entire packet information (payload and header) and can conclude the network activities in a much granular way, hence suitable for real-time detection. On the other hand, this method needs much-sophisticated hardware. The network speed increases exponentially, and packet inspection also breaches data privacy since it is done at a significantly deeper level. Instead, flow level data can compromise the speed of traffic by aggregating data into flows and reducing the number of traffic to be inspected. One setback with this method is that it is not suitable for payload inspection because inflow level packet payload is not considered. In this regard, connection-level data is preferable because it provides a more comprehensive picture of network traffic between two hosts. However, link-level data collection necessitates keeping track of connection status, which consumes more resources. As a result, examining the advantages and disadvantages of each packet-level, flow-level, and connection-level data will complement one another. A combination of all three is ideal since it provides comprehensive information on network activity. On the other hand, host-level data gives a complete view of the events in the system but not about any network activities. Therefore, host-level data cannot be used alone as it can give a very high false-positive rate even when normal user activities are performed.

Table 4: Summarization and comparison of data categories [15]

Data category	Classification		Advantages	Disadvantages
	Type	Explanation		
Packet level data	Source address	The IP address of source network interface	Access to raw packet data so that make real-time detection possible Allow pattern matching in payload content Have full information about network activities	Collection methods are not suitable for high-speed networks Touch private and sensitive information
	Destination address	The IP address of destination network interface		
	Source Port	The end-point of source network interface		
	Destination Port	The end-point of destination network interface		
	TTL	The maximum hop count		
	Timestamp	The point-in-time of sending or receiving packets		
	Packet payload	The content that packet carries		
	Packet size	The size of a packet in bytes		
	The number of transmitting bytes	The number of bytes accumulated in a certain time period		
	The number of packets	The packet count accumulated in a certain time period		
	Packet rate	The number of transmitting packets per unit time		

Table 4 (continued).

Data category	Classification		Advantages	Disadvantages
	Type	Explanation		
Flow-level data	Flow count	The number of flows	Independent of encrypted payload Collection methods are suitable for high-speed networks Collection methods are widely deployed and well understood	Collection methods cause some inevitable delay Have no information about packet payload
	Flow type	The protocol information of flow		
	Flow size	The number of packets in a flow		
	Flow direction	The transmission direction of a flow		
	Flow duration	The time duration of a flow		
	Flow rate	The number of transmitting packets per unit time of a flow		
Connection-level data	Connection size	The number of packets or flows in a connection	Provide global information of exchanged traffic between two IP addresses in a given time; Support a good detection performance when being used with other categories of data	Collection methods need to keep track of each connection status; Have poor performance when being used separately from other categories of data
	Connection duration	The time duration of a connection		
	Connection count	The number of connections		
	Connection type	The protocol information of a connection		
Host-level data	CPU usage	The load information about running software programs	Have full information about system performance and behaviors; Record and internal changes of a host system	Have a high false alarm rate when being used separately from other categories of data; Collection methods take up host-side resources
	Memory usage	The information about data exchange		
	Equipment operation logs	The running records of equipment that connects with a host		
	Application operation logs	The running records of an application		

As a summary, the analysis given based on each type of data can be used to detect specific attacks by considering the detection method and the network environment. Any application-specific attack detection may require packet-level and connection/flow level data. Whereas inspecting malware within a host might require more host-level data and some extend of connection data. Much network-based attacks such as DDoS and Botnet may need to utilize packet level and connection-level data. This attack can be further granularized by integrating host-level data to see the effect of DDoS on the host performance. A higher accuracy of detection can be achieved by doing a thorough analysis of the nature of the attacks and their impacts on the host or the network.

3.7 Summary of Network Traffic Dataset

The live dataset collected from the network is used for intrusion detection, but many compiled datasets are available on the internet for network intrusion detection. One of the most widely used datasets is the KDD Cup data set [23]. According to Can et al. [24], the KDD Cup data set consists of approximately 4,900,000 training instances and 41 features (Tab. 5). It is labelled with exactly one specific attack type, i.e., either standard or an attack. The training data set includes 80% attack and 20% normal data. Some of KDD features may not apply in intrusion detection because they do not have any useful role to classify the outputs. The network intrusion detection by using KDD cup has been done by Can et al. [24] shows 89.17% success rate on 67500 samples of attacks in the artificial neural network-based intrusion detection system for wireless sensor network. Due to the rapid change of technology, KDD datasets might have some limitations in detecting abnormal traffic in *ad hoc* wireless network [25]. Traffic collectors such as TCPdump, KDD dataset are very likely to become overloaded and drop packets in heavy traffic load. Some of the network traffic datasets are based on the current operating systems and hardware, so the

KDD dataset might be a challenge for investigating the *ad hoc* network traffic since the *ad hoc* network and IoT devices are built with different operating systems and hardware. Finally, most of the work on using the KDD dataset is mainly deployed for wired intrusion detection, and therefore wireless network features and attacks are not included in KDD datasets.

Table 5: KDD dataset features [8]

Basic Features		
Duration	Integer	Duration of the connection
Protocol_type	Nominal	Protocol type of the connection; TCP; UDP and ICMP
Service	Nominal	http, ftp, smtp, telnet … and other
Flag	Nominal	Connection status
Src_bytes	Integer	Bytes sent in one connection
Dst_bytes	Integer	Bytes received in one connection
Land	Binary	If src/dst IP address and port numbers are same, then 1
Wrong_fragment	Integer	Sum of bad checksum packets in a connection
Urgent	Integer	Sum of urgent packets in a connection
Content Features		
Hot	Integer	Sum of hot actions in a connection such as; entering a system directory, creating programs and executing programs
Num_failed_logins	Integer	Number of incorrect logins in a connection
Logged_in	Binary	If the login is correct, then 1, else 0
Num_compromised	Integer	Sum pf times appearance “not found” error in a connection
Root_shell	Binary	If the root gets the shell, then 1, else 0
Su_attempted	Binary	If the su command has been used, then 1 else 0
Num_root	Integer	Sum of operations performed as root in a connection
Num_file_creations	Integer	Number of logins of normal users
Num_access_files	Integer	Sum of operations in control files in a connection
Num_outbound_cmds	Integer	Sum of outbound commands in an ftp session
Is_hot_login	Binary	If the user is accessing as root or admin
Is_guest_login	Binary	If the user is accessing as guest, anonymous, or visitor
Traffic Features – Same Host – 2-second Window		
Duration	Integer	Duration of the connection
Protocol_type	Nominal	Protocol type of the connection; TCP, UDP, and ICMP
Service	Nominal	http, ftp, smtp, telnet..and other
Flag	Nominal	Connection status
Src_bytes	Integer	Bytes sent in one connection
Dst_bytes	Integer	Bytes received in one connection
Land	Binary	If src/dst IP address and port numbers are same, then 1
Wrong_fragment	Integer	Sum of bad checksum packets in a connection
Urgent	Integer	Sum of urgent packets in a connection

Sobh [26] performed a complete analysis of the KDD data set and had found some severe limitation of the dataset. Further analysis by Sobh (2006) [26] comprises the top 20 alarms generated, attacks detected, the contribution of fields to detection attacks not seen and overhead in terms of time and space. For the top 20 alarms, the authors conclude that out of 20, 8 has been identified to correctly detect attacks with the destination IP address as the important field. 9 other alarms were false positives (FP), and three others were able to detect arp poison but without the IP address since it only involves ARP packets. As for the attack detections, out of 201 attacks instances created, 67 episodes were detected such as DOS, probe and R2L, and the detection rate is above 50% for all these cases with TTL as one of the main header features contributed to the detection with 33 detections out of the 19 types of attacks. And 8 out of the 67 attacks were detected by IP addresses and none by port numbers. 30 checksum errors were created due to IP fragmentation and no proper reasoning for using smaller fragments. The author performed the statistical evaluation and own classification methods and found some shortcomings in the KDD Cup data. So some of the weaknesses identified by the author are:- very ambiguous attack definition, packet drop due to traffic overflow, too many redundant records (75% in training and 75% in the testing), weird input to unwell configured software, odd data from impractical attacks and some unrealistic data to hide some of the attacks. Furthermore, just by some random selection of data for training and data for testing, the results show a very unrealistic accuracy value.

Therefore [27] proposed a new dataset known as NSL-KDD with properly selected KDD data records. NSL-KDD (Tab. 6) seems to be a refined version of KDD cup data with all the essential 42 features with 5 classes and 4 attacks. According to Chae et al. [27], the dataset is a better selection as it does not have many redundant data in the training set, and therefore no business would occur during classification. Also, the number of records selected according to the difficulty level is inversely proportional to the number of records in the original KDD Cup data. Therefore, classification using various machine learning methods yields diverse accuracy rates that evaluate various classifications methods. Likewise, the number of data selected for training and testing is reasonable without being randomly selected as in the KDD Cup data. The investigation conducted by various researchers [27–33] show that NSL-KDD gives consistent and more realistic results.

Table 6: NSL-KDD features extracted from [28]

Type	Features
Nominal	Protocol_type(2), service(3), flag(4)
Binary	Land(7),logged_in(12),root_shell(14),su_attempted(15), is_host_login(21),,is_guset_login(22)
Numeric	Duration(10,src_bytes(5), dst_bytes(6), wrong_fragment(8), urgent(9), hot(10), num_failed_logins(11),num_compromised(13), num_root(16), num_file_creations(17), num_shells(18), num_access_files(19), num_outbound_cmds(20), count(23), srv_count(24), serror_rate(25), srv_error_rate(26),rerror_rate(27), srv_reeror_rate(28), same_srv_rate(29), diff_srv_rate(30), srv_diff_host_rate(31), dst_host_count(32), dst_host_srv_count(33), dst_host_same_srv_rate(34), dst_host_diff_srv_rate(35), dst_host_same_src_port_rate(36), dst_host_srv_diff_rate(37), dst_host_serror_rate(38), dst_host_srv_serror_rate(39), dst_host_rerror_rate(40), dst_host_srv_rerror_rate(41)

In an earlier study, McHugh (2010) [34] addresses the following issues with the KDD Cup data:- i) the TCP dump data is overloaded, and it tends to drop packets, ii) the Solaris dump data implications are not investigated, iii) not even one intrusion detection methods has used the Receiver Operating Curve (ROC) and Relative operating Characteristics, iv) errors per unit time could have been a better parameter to be analysed and v) an improvement in the false alarm rate in keeping the workload constant should have been addressed.

3.8 A Preliminary Wireless Network Dataset

IEEE 802.11 based WLANs consists of several frames that contain the information of the packets. The expanded packets in the dataset provide a clear comparison between the types of structures. For example, the number of fields of each packet differs according to the type of frame the packet is in. Therefore, the feature selection highly relies on similar fields that all the frames possess to compare the differences in the same fields of different frames. The types of frames are an important feature in the study of wireless networks as they affect the number of fields and the types of fields present in the frame. Tab. 7 shows a shows important frame information in the IEEE 802.11 dataset for traffic analytics. The features presented here can be used as a guideline for designing an effective wireless intrusion detection system. Besides the traditional TCP, UCP and IP packet headers, it is suggested to include the IEEE802.11 frame headers for effective wireless intrusion detection design.

Table 7: IEEE 802.11 expanded view of frame information

Info	Frame	802.11 Radio Information	Layers in TCP/IP Stack
Beacon Frame	Management	PHY type = 802.11(b)	S- Band DSSS Physical Layer
Null Function	Data	PHY type = 802.11(b) = 802.11(g)	(b)= DSSS Physical Layer (g)= S-Band ISM OFDM Physical Layer
Acknowledgement	Control	PHY type = 802.11(b) = 802.11(g)	(b)= DSSS Physical Layer (g)= S-Band ISM OFDM Physical Layer
Data	Data	PHY type = 802.11(b) = 802.11(n)	(b)= DSSS Physical Layer (g)= MIMO Physical Layer
Action	Management	PHY type = 802.11(b)	(b)= DSSS Physical Layer
Probe Response	Management	PHY type = 802.11(b)	(b)= DSSS Physical Layer
Clear-to-send	Control	PHY type = 802.11(b)	(b)= DSSS Physical Layer
QoS Data	Data	PHY type = 802.11(b)	(b)= DSSS Physical Layer
Probe Request	Management	PHY type = 802.11(b)	(b)= DSSS Physical Layer
Block Ack	Control Frame	PHY type = 802.11(b) = 802.11(g)	(b)= DSSS Physical Layer (g)= S-Band ISM OFDM Physical Layer
Request-to-send	Control	PHY type = 802.11(b) = 802.11(g)	(b)= DSSS Physical Layer (g)= S-Band ISM OFDM Physical Layer
Block Ack Request	Control	PHY type = 802.11(b) = 802.11(g)	(b)= DSSS Physical Layer (g)= S-Band ISM OFDM Physical Layer
QoS Null Function	Data	PHY type = 802.11(b) = 802.11(g)	(b)= DSSS Physical Layer (g)= S-Band ISM OFDM Physical Layer
Key	Data	PHY type = 802.11(b)	(b)= DSSS Physical Layer
DE authentication	Management	–	No layer
CF - End	Control	PHY type = 802.11(g)	S-Band ISM OFDM Physical Layer
Authentication	Management	PHY type = 802.11(b)	DSSS Physical Layer
Association Response	Management	PHY type = 802.11(b)	DSSS Physical Layer
Association Request	Management	PHY type = 802.11(b)	DSSS Physical Layer

4 Discussion

There is no fundamental research in IoT intrusion detection (ID) that mainly focuses on wireless networks to the best of our knowledge. Most of the research area in IDS focuses on traditional wired networks. Applying the wired network research of IDS at wireless network may not be feasible due to the architectural differences of IoT. Traditional security countermeasures and privacy enforcement cannot be directly applied to IoT technologies due to the three fundamental aspects:

1. The limited computing power of IoT components
2. The high number of interconnected devices
3. Sharing of data among objects users

Moreover, intrusion response to wireless networks depends on the type of intrusion, network protocols and applications in use and the confidence in the evidence, which is different from wired networks. A few works have been conducted using IDS to counter the wireless network attacks in IoT security. The main challenge is the nature of the wireless network. Unlike a wired network, in the wireless network, centralized access control is hard to be implemented due to the distributed nature of a wireless network. IoT devices and networks are the sources to generate massive unstructured data. Until now, researchers usually do not have access to the complete IoT network data that can be used for intrusion detection research. The wireless intrusion detection system will need to collect as much protocol data from the wireless network as needed.

Moreover, there are specific vulnerabilities in the physical and data link (MAC vulnerability) layer in wireless networks, which was not attempted in designing wired IDS. Therefor just deploying a wired IDS into wireless IDS would be just a false hope as it may not detect some specific wireless attacks, especially at the data link layer. No reliable research work has been conducted to create a standard benchmark dataset in a wireless IoT environment.

5 Conclusion

Careful selection of datasets is important in training ML-based wireless intrusion detection systems. As discussed, KDD Cup datasets and NSL-KDD Datasets contain traffic features that are detrimental to detect model accuracy when they are used to train to detect IoT variants kind of network intrusions. In IoT networks, wireless traffic carries more critical information at the data link. A detailed comparison between wired and wireless data showed that most wireless IDS' relevant features are found in the physical and data link layers. The findings indicate that adjusting features' weight for wireless-specific header information can potentially improve intrusions classification. Currently, to our best knowledge, no reliable research has been conducted to create a standard benchmark dataset in a wireless IoT environment. This paper identified a set of high gain features (in Tab. 7) that is highly correlated to network intrusion on wireless networks. The feature sets are filtered through a combination of domain heuristics and preliminary testing results of ML models trained with these custom feature sets [34–39]. Future investigation can leverage these feature set to customize the scope of data collection for any ML-based Wireless IDS design for IoT infrastructure.

Funding Statement: The author(s) received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflict of interest to report regarding the present study.

References

[1] M. Humayun, N. Jhanjhi, M. Alruwaili, S. S. Amalathas, V. Balasubramanian *et al.,* "Privacy protection and energy optimization for 5G-aided industrial internet of things," *IEEE Access*, vol. 8, pp. 183665–183677, 2020.

[2] T. Mohit, K. Raj, B. Akash and K. Jai, "Intrusion detection system," *International Journal of Technology Research and Applications*, vol. 5, no. 2, pp. 2320–8163, 2017.

[3] S. Boob and P. Jadhav, "Wireless intrusion detection system," *International Journal of Computer Application*, vol. 5, no. 8, pp. 0975–8887, 2010.

[4] Z. Tao and A. Ruighaver, "Wireless Intrusion Detection: Not as easy as traditional network intrusion detection," in *Proc TENCON 2005*, Melbourne, VIC, Australia, pp. 21–24, 2005.

[5] R. Mitchell and R. Chen, "A survey of intrusion detection in wireless network applications," *Computer Communications*, vol. 42, pp. 9–13, 2014.

[6] Z. M. Fadlullah, H. Nishiyama, N. Kato and M. Fouda, "Intrusion detection system (IDS) for combating attacks against cognitive radio networks," *IEEE Network*, vol. 27, no. 3, pp. 51–56, 2013.

[7] M. Humayun, N. Z. Jhanjhi and M. Z. Alamri, "IoT-based secure and energy efficient scheme for e-health applications," *Indian Journal of Science and Technology*, vol. 13, no. 28, pp. 2833–2848, 2020.

[8] L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.

[9] S. Alcock, P. Lorier and R. Nelson, "Libtrace: A packet capture and analysis library," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 2, pp. 42–48, 2012.

[10] Snort, [Accessed 14 August 2018]. *Available at:* https://www.snort.org.

[11] G. F. Lyon, "Nmap network scanning: The official Nmap project guide to network discovery and security scanning," *Insecure*, Com LLC (US), 2019.

[12] V. Jacobson, C. Leres and S. McCanne, "The tcpdump manual page," *Lawrence Berkeley Laboratory*, vol. 143, pp. p.–117, 1998.

[13] J. Weber, "The Fundamentals of passive monitoring access," in *Net Opt. Inc*, Santa Clara, CA, USA, 2006.

[14] M. V. Mahoney and P. K. Chan, "PHAD: Packet header anomaly detection for identifying hostile network traffic," 2021.

[15] X. Jing, Z. Yan, X. Jiang and W. Pedrycz, "Network traffic fusion and analysis against DDoS flooding attacks with a novel reversible sketch," *Information Fusion*, vol. 51, pp. 100–113, 2019.

[16] M. F. Umer, M. Sher and Y. Bi, "Flow-based intrusion detection: Techniques and challenges," *Computers & Security*, vol. 70, pp. 238–254, 2017.

[17] B. Li, J. Springer, G. Bebis and M. H. Gunes, "A survey of network flow applications," *Journal of Network and Computer Applications*, vol. 36, no. 2, pp. 567–581, 2013.

[18] Netflow, [Accessed 14 August 2018]. *Available at:* https:// www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-net flow/prod_white_paper0900aecd80406232.html.

[19] T. Fioreze, M. O. Wolbers, v. d. R. Meent and A. P. Pras, "Finding Elephant flows for optical networks," *Integrated Network Management*, *Proc. of the 6th ACM SIGCOMM Conf. on Internet Measurement (IMC06)*, vol. 2006, pp. 627–640, 2007.

[20] R. Hofstede, P. Celeda, B. Trammell, I. Drago, R. Sadre *et al.,* "Flow monitoring explained: From packet capture to data analysis with netflow and ipfix," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 4, pp. 2037–2064, 2014.

[21] Collectl, [Accessed 14 August 2018]. *Available at:* http:// collectl.sourceforge.net/index.html.

[22] Loadrunner, [Accessed 14 August 2018]. *Available at:* https://saas.hpe.com/zh-cn/software/loadrunner.

[23] S. J. Stolfo, W. Fan, W. Lee and A. Prodromidis, "KKD Cup 1999 Data," 2011. [Online]. Available at: http://kdd.ics. uci. Edu/databases/kddcup99/kddcup99. Html.

[24] O. Can and O. K. Sahingoz, "An intrusion detection system based on neural network," in *Proc SIU*. Malatya, Turkey, 2015.

[25] M. Ahmed, A. N. Mahmood and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 2016.

[26] T. S. Sobh, "Wired and wireless intrusion detection system: Classifications, good characteristics and state-of-the-art," *Computer Standards & Interfaces*, vol. 28, pp. 670–694, 2006.

[27] H. S. Chae, B. O. Jo, S. H. Choi and T. K. Park, "Feature selection for intrusion detection using nsl-kdd," *Recent advances in computer science*, vol. 20132, no. 2013, pp. 184–187, 2013.

[28] L. Dhanabal and S. P. Shantharajah, "A study on NSL-KDD dataset for intrusion detection system based on classification algorithms," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 4, no. 6, pp. 446–452, 2015.

[29] S. Lakhina, S. Joseph and B. Verma, "Feature reduction using principal component analysis for effective anomaly-based intrusion detection on NSL-KDD," *International Journal of Engineering Science and Technology*, vol. 2, no. 6, pp. 3175–3180, 2010.

[30] S. Revathi and A. Malathi, "A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection," *International Journal of Engineering Research & Technology (IJERT)*, vol. 2, no. 12, pp. 1848–1853, 2013.

[31] B. Ingre and A. Yadav, "Performance analysis of NSL-KDD dataset using ANN," in *Proc SPACES*. Vijayawada, India, 92–96, 2015.

[32] V. Kumar, H. Chauhan and D. Panwar, "K-means clustering approach to analyze NSL-KDD intrusion detection dataset," *International Journal of Soft Computing and Engineering (IJSCE)*, vol. 3, no. 4, pp. 2231–2307, 2013.

[33] K. Shrivas and A. K. Dewangan, "An ensemble model for classification of attacks with feature selection based on KDD99 and NSL-KDD data set," *International Journal of Computer Applications*, vol. 99, no. 15, pp. 8–13, 2014.

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

[35] S. J. Hussain, M. Irfan, N. Z. Jhanjhi, K. Hussain and M. Humayun, "Performance enhancement in wireless body area networks with secure communication," *Wireless Personal Communications*, vol. 116, no. 1, pp. 1–22, 2021.

[36] S. G. Kene and D. P. Theng, "A review on intrusion detection techniques for cloud computing and security challenges," in *Proc ICECS*, Coimbatore, India, 2015.

[37] M. Robert and I. R. Chen, "A survey of intrusion detection in wireless network applications," *Computer communications*, vol. 42, pp. 1–23, 2014.

[38] M. Shafiq, H. Ashraf, A. Ullah, M. Masud, M. Azeem *et al.,* "Robust cluster-based routing protocol for IoT-assisted smart devices in WSN," *Computers Materials and Continua*, vol. 67, no. 3, pp. 3505–3521, 2021.

[39] G. K. Snehal and P. T. Deepti, "A review on intrusion detection techniques for cloud computing and security challenges," in *Proc ICECS*, Coimbatore, India, 2015.