DOI: 10.21917/ijct.2022.0402

DESIGN OF INTRUSION PREVENTION SYSTEM IN INTERNET OF THINGS COMMUNICATION

K.P. Swaraj[1], G. Kiruthiga[2] and P.A. Shemitha[3]

[1]*Department of Computer Science and Engineering, Government College of Engineering Thirssur, India*
[2,3]*Department of Computer Science and Engineering, IES College of Engineering, India*

Abstract

In this paper, we model a CAN bus controller for data communication in Internet of Things (IoT) Network. However, most communication taking place via CAN bus may prone to attack. Hence aligning security with Intrusion Prevention System (IPS) to detect and mitigate attacks are required. In this paper, we hence model a IPS system over CAN communication. The model uses logs of communication to get trained and detect the attacks in the network. The simulation is conducted in NS2.34 tool to test the efficacy of the CAN-IoT Model. The results show that the proposed method has higher detection rate than other methods.

Keywords:

CAN Controller, Internet of Things, Intrusion Prevention System, Detection Rate

1. INTRODUCTION

Vehicles are now able to connect to the Internet of Things (IoT) because to recent advancements in embedded device technology and advances in internet access. Due to the crucial nature of the roles they serve, many technological systems present serious safety concerns. Modern high-end automobiles are equipped with a plethora of electronic control units (ECUs), all of which work together to enhance the vehicle functionality. One hundred electronic control units (ECUs) are not uncommon in modern automobiles. These features all add up to a more complex car system, which increases the attack surface and thus the likelihood of an attack. These ECUs are able to talk to one another across the vehicle internal network thanks to a variety of protocols, including the controller area network (CAN), the local interconnect network (LIN), FlexRay, and the media-oriented systems transport (MOST). The Controller Area Network (CAN) bus is the dominant networking technology in vehicles today. It uses a broadcast protocol to link together diverse electronic control modules (ECUs). Although the CAN bus was developed for use in automobiles, it has since found significant use in the automation of industrial processes and other industries. The electronic control units (ECUs) and their accompanying software are vulnerable due to the lack of built-in security mechanisms in the CAN bus.

The CAN bus has been found to be susceptible to both network-based and direct assaults [5], [9], [12] and [15]. This occurs because the message frames sent between ECUs are not encrypted or authenticated, and because it allows connection between illegal nodes. Wireless network connectivity allows for indirect attacks from a wider distance than the on-board diagnostic (OBD-II) connector, which allows for direct physical attacks. It was initially successfully demonstrated and implemented by Koscher et al. [12], who used reverse engineering on the ECU codes to manage a wide range of vehicle activities. Cyber threats against automotive networks are continually developing, as evidenced by the work, who show how hackers can connect to vehicles remotely and take command of their operations. Both of these scientists provide proof by showing how cybercriminals can connect to vehicles remotely and take command of its features. Despite the constant emergence of new dangers, we continue to lack adequate information for countering them.

Since the security requirements are not well-defined and the available bandwidth is limited by the protocols that govern the vehicular network, designing security for the in-vehicle network has proven to be difficult. Standards like ISO/SAE 21434 [2] and the SAE J3061 cybersecurity guideline for cyber-physical vehicle systems [17] have recently been developed and published in an effort to close this security engineering gap in modern autos. One such standard is ISO/SAE 21434 [2]. There is a need to consider the system memory and processing capacity limits while designing a safety device for usage in autos. This means that in-car network security must be lightweight while still taking into account the possibility of the network failing.

Recent literature [21] shows that numerous intrusion detection system (IDS) systems for CAN buses are in use. Since the process by which such IDSs react has been mostly overlooked, designers are left wondering how they might include an IDS warning into a vehicle system. This is due to the widespread disregard for these IDSs. In this paper, we present the architecture of an intrusion prevention system (IPS) that can be used in tandem with any existing intrusion detection system (IDS) to aid in recovering from attacks and preventing similar ones in the future by employing a restart-based technique. In order to bring a compromised node into a recovery state while it is being attacked, we make use of the busoff state, an error handling feature of the CAN bus. This characteristic is called the busoff state, and it is a special case of the busoff transition. Until it is reset, an ECU in a bus-off state cannot communicate with the network. Before the ECU may start sending messages again, this must be completed.

2. RELATED WORKS

Concerns concerning the feasibility of installing security measures to protect the WSN against unwanted access and data theft have been raised by researchers [5]. Unauthorized users can employ passive attacks against a network security to listen in on conversations and glean useful information. Passive attacks include eavesdropping on targets, studying traffic patterns, and sneaking up on people. When the packets are carrying control information, an attacker who eavesdrops has a much better chance of success than one who relies on the location server to acquire this data [6]. Malicious nodes initiate active eavesdropping attacks by sending requests to fake transmitters masquerading as legitimate nodes. Instead, in passive eavesdropping attacks, the

bad nodes simply listen in on the wireless channel and record whatever they find. In order to pose as a legitimate node and send queries to transmitters, a malicious node must first learn enough about the network architecture to convince it that it is, in fact, a valid node. To achieve this goal, an active eavesdropping attack must first be carried out [7]. In contrast, an attack is called active when the attacker actively snoops on, alters, or otherwise interferes with the information passing through the channel. During an active attack, the authenticity and freshness of data stored on nodes could be questioned. The vulnerabilities of wireless sensor networks make them susceptible to a wide variety of attacks. Selective forwarding, jamming, hello flood, Sybil assaults, sinkholes, and wormholes are all types of attacks that can be launched against a network. Security in wireless sensor networks is essential for keeping user identities, data, and privacy safe [8].

An intrusion occurs when an unauthorized user gains access to a wireless sensor network. An attacker use of eavesdropping is an example of passive information collecting. One method of actively obtaining data is via forwarding malicious packets, deleting packets, or setting up a hole node [9]. So that any malicious activity on the network can be discovered, intrusion detection systems (IDSs) are built and integrated into the network. These setups leverage individual computers within the network. If malicious nodes are found in the network, users must be warned immediately and the network architecture may need to be revised. Many intrusion detection systems exist today [10]-[12], but many of them either fail to detect intrusions effectively or have significant computational, power, or communication overheads.

Different types of IDS can be constructed depending on the considerations and goals that need to be accomplished. A definition of IDS based on factors including the nature of the intrusion, the nature of the invader, the technique of detection, the audit data source, the location of the computing where the data was collected, the underlying infrastructure, and the frequency of use is discussed in [13]. It states that signature-based detection and anomaly-based detection are the two primary categories of IDS. In addition, specification-based detection into its own subcategory for the sake of clarity [14] [15].

3. INTRUSION PREVENTION SYSTEM

An error frame is immediately added to the transmission queue whenever the intrusion prevention system (IPS) detects an attack. The first six dominant bits in a row at the start of the frame will be given the greatest priority in the next round of bus arbitration. Nodes that have already received the message will not be affected by any errors in the frame.

When an IPS detects an attack, it will send out an error frame. This will result in an 8-point rise in the TEC of the sending node and a 1-point increase in the REC of the receiving nodes. If the hacked node TEC value rises above 255, it will enter a bus-off condition and be rendered inoperable.

The electronic control units must detect 128 sets of 11 consecutive recessive bits on the bus before returning to the error-active state from the bus-off state. The final few seconds before the changeover may also see an ECU perform a self-reset or reboot. We recommend that when the bus is shut down, all ECUs with remote interface capabilities be rebooted. One recovery option that can be utilized to restore the computer to its previous state is to restart it. System restarting is advantageous because it can be easily automated, it returns the application to its original configuration (which has been thoroughly tested and understood), and it does not depend on the system running correctly when the reboot is complete [4].

High availability of these ECUs is still attainable with power cycling, even if the detection system is prone to false alarms or if it is unsure if the reboot operation can repair the issue. This holds true even if it is unclear whether a reboot will resolve the problem. Keep in mind that these ECUs are not accountable for the safety of the user. Thus, restarting them will not harm the system, but it may harm the user experience.

Wait-then-reset requires the application layer to wait until some time has passed after a fault situation has been recognized, whereas automatic reset begins as soon as the CAN controller changes to bus-off. As soon as the CAN controller goes into bus-off mode, a reset is initiated automatically. The frequency-limited-reset function prevents resets from occurring any less frequently than the minimum interval selected by the user. When the application layer reset vector is used to initiate a reset, the CAN controller enters a bus-off state and waits until it detects the required number of recessive bits before rejoining the network. While the reset is being executed, this occurs.

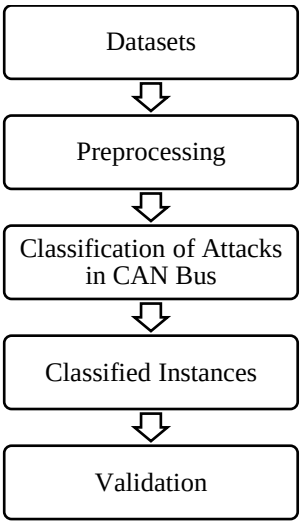

Fig.1. Proposed Model

Every agent is either a host agent or a network agent. Host agents are pieces of software that can be installed on hosts in order to keep an eye out for suspicious or malicious activities. Host agents can keep tabs on host positive and bad deeds. However, network monitoring is performed by agents hosted on switches and routers. In other words, these agents monitor the network and record every single event. The proposed approach will increase the model dependability despite introducing some unnecessary computational expense and task duplication between the host and network agents. A host agent location of operation and the data it collects are two of the most distinguishing features between it and a network agent. During this time, network agents are monitoring data packets and host agents are checking system files.

Detection of potentially harmful conduct is a major goal for both classes of agents. While agents are responsible for detecting intrusions, they are not held responsible for preventing them. Prevention is handled independently within the paradigm

structure. First, it is up to the learning agents to establish priority levels, and second, it is up to them to make the actual choices. The collected data will be given a priority level by the agent during the process of setting priorities. Important information, like the current state of the network, may be absent from particular datasets. Therefore, we don't give them much of a priority. However, it possible that some other data stores crucial information about the host or network, and as such, it should be labeled as high-priority information. This is because the data it holds is so crucial. The agent will next decide whether or not the current connection or host is malicious. The agent is responsible for making this decision.

In order to find information that is similar to that discovered in other clusters, the agents first cluster the data. Using the same distance scales discovered in the earlier stage of the procedure, we now evaluate the distance function for the data we have acquired from the datasets. The next step is to pinpoint the most accessible hub. The node in the center of this cluster is currently engaged in a computation to determine how far off from the present the data being processed actually is. The last, and most crucial, stage is for the agents to choose $2n+1$ data points that are geographically close to the new data and, hence, are the most comparable. In order to decide which nearest data point to employ in the next phase, the agents choose an odd number of data points. In addition, agents pick a lot of data points from the local region since they have to consider the likelihood that data will be inaccurate or noisy.

As a first stage in making a choice, agents look for the cluster that is geographically nearest to the new data and zero in on data points that have a high degree of similarity with the new information. Agents must first develop association rules to ascertain the validity of their assessment before reaching a final choice. If you utilize the conditional statements known as association rules, you may determine the most likely connection between two datasets. We have used confidence and support measures, which assess the strength of the association between two data points, to reach our objective. Support and confidence measures are computed using the entire dataset, even though we only use data from a single cluster.

3.1 INTRUSION PREVENTION

The only real function of a passive intrusion detection system is to alert the user that they have been the target of some form of suspicious activity. However, our DMAIDPS intrusion detection and prevention solution can help reduce the impact of these assaults on networks. The system administrator can create a matrix that shows how various attacks have been found and what countermeasures have been implemented. Network architecture, corporate regulations, and other contextual considerations may all play a role in the specifics of these mappings. However, the most common method of protection is to avoid attack altogether. In response to the discovered attack, DMAIDPS was created. Next to our firewall, we have placed a server whose main purpose is prevention. As soon as the agents detect malicious activity, they will notify this server with relevant details. The firewall rules are updated mechanically whenever a new threat is notified to the prevention server.

Two types of learning agents are considered within DMAIDPS. This raises the possibility that the attack the agents traced back to two distinct sources. If the prevention server detects that a host has been compromised, it will order the firewall to temporarily close all of the host ports. In this scenario, the malicious host is isolated from the rest of the network.

When an infected host is identified, any and all connections to it are immediately terminated. Such an event might occur if, for instance, the malicious host started launching probing attacks against the other hosts. Its inability to send and receive probing packets with other hosts is a direct result of this. Alternatively, if the intrusion is traffic-related, the prevention server will alter the firewall rules so that the routers and switches temporarily disable the malicious port.

Let pretend that a DDoS attack causes widespread network outages, but the perpetrators of the attack remain anonymous. On the other hand, agents within the network have discovered that the flooding packets are being forwarded to other nodes via the port of a certain router. The firewall port will be shut down to prevent the flood from spreading.

4. EXPERIMENTS

We begin by analyzing the time it takes for a detector node to decide what to do after receiving a message frame from the controller. In this article, we'll look at the similarities and differences between two IDS varieties: those that use interval time analysis and those that use response time analysis. We also provide an evaluation of the CAN bus speed minimal requirements for operation. The quantity of information within a single message frame served as the basis for this investigation. There are seven nodes in the experimental network, six of which are data collectors and one of which is the control node.

Every computer in the system can operate as a relay, transmitting data across the bus to any other computers that are tuned in. There is no limit to the number of copies of the same message that a single node can transmit. You can only send one message at a time, though. The supervisory node in a time-response IDS system is aware of the periodicity of each message and the window of time during which the first message from each node is expected to arrive. Having this data on hand helps find break-in attempts. The detector node stores the minimum interval threshold for each message sent, same like in the previous illustration.

Specifically, we are interested in how long it takes for a single message to go through the detector. The CAN controller uses the system clock to generate the clock signal for the detector block. Upon receipt of a message, the detector node examines it with a number of filters, such as the message instance ID and the time it arrived. Once it is established that the current instance of the message has passed both checks, the detector node will alter the arrival time for the succeeding instance of the message. Figure 8 depicts this operation, which is used by the IDS for analyzing msg response times. The source node will increase its TEC by eight and the detector node REC by one if the message fails to pass one of the checks.

If the detector receives a message frame with an ID that is not in its lookup table, it will take five clock cycles to determine if the ID is genuine. This holds true regardless of whatever IDS is in use. As soon as the injected message completes, the bus will be occupied by the error frame, making the message frame with the

unknown ID useless because it cannot be transmitted. Error frames take up valuable bus real estate once an injected message has finished. Depending on the IDS in question, the time it takes to scan the anomaly detection characteristics of a message and subsequently update the status can vary considerably. Each of the currently-considered IDS alternatives and their associated costs are detailed below.

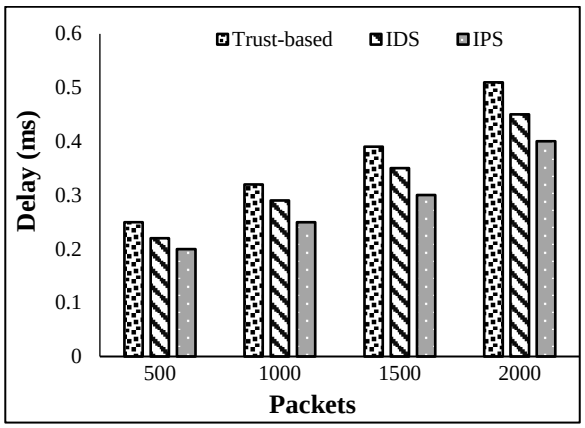

Fig.2. Delay

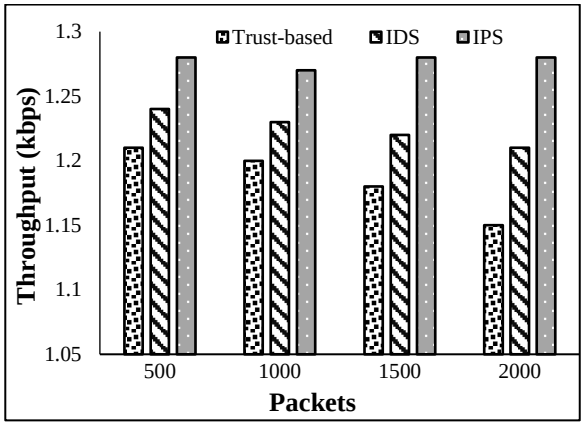

Fig.3. Throughput

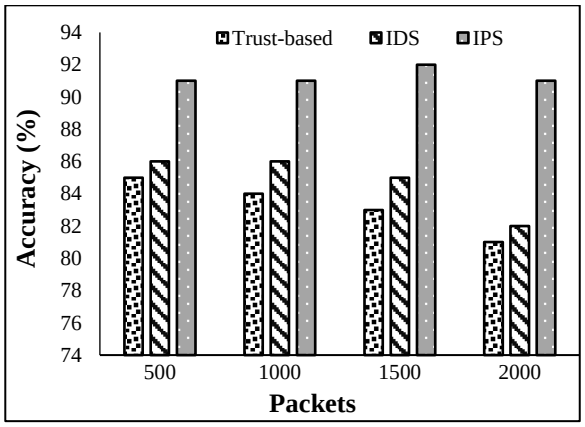

Fig.4. Detection Accuracy

In order to save money on electricity, it is best to run at the lowest possible frequency; nevertheless, the bus minimum data length and the total amount of valid messages must also be considered. So, if there is any message that uses 0 bytes as its data length, the 0-bytes analysis must be used to determine the lowest possible operational frequency. The graphic shows that as the number of data bytes increases from 0 to 8, the minimum operating frequency drops by roughly a third for each bus speed. This takes place regardless of how fast the bus goes.

However, fast the bus may be, the detector must place a call in the same amount of time it takes to transmit 25 bits. The detector must be able to cycle at least once every 5000 ns, even if there are no data bytes in the frame, for a 1 Mbps data bus. Due to the attached IDS, the detector is now capable of speeds of up to 400 MHz, and message validation can be completed in 12.5 ns. The procedure will be completed and a choice on the next data frame made before the 13th bit is finished processing, as it takes 1000 nanoseconds to do so.

Even if the message ID is not in the lookup table or the timing is off, the detector node will have already decided on the message and queued an error frame well before the bogus message is finished transmitting. With a 1 Mbps bus and 8-byte messages, the detector can function at a minimum frequency of 56 kilohertz.

With this method, you can rest assured that the message frame will still be OK in under 10 nanoseconds. Despite this, the error frame could still be sent before the end-of-frame bit at the end of the message is processed. Many vehicle transmission times are substantially higher than necessary because non-safety-critical ECUs are linked to medium- and low-speed buses. Since this is the case, a detector node running at the worst-case frequency of 56 kHz has significantly more time to complete its procedures and send out an invalidation message. Since our proposed approach is lightweight and computationally efficient, it is easy to implement and use, making it a viable option. This is due to its usefulness in preventing anomalous behavior in the transportation network.

5. CONCLUSION

Our team has created a new IDS that monitors CAN buses for unwanted intrusions. An ECU that has been compromised can recover from an attack by rebooting with the help of this IDS, which can also block malicious remote message injection attacks. We investigated the feasibility of using CAN error handling strategies to kick an attacker off the network. We also reconstructed proposed CAN IDSs and evaluated their ability to spot assaults with latency times that are consistent with bus line speeds. Between the time the last bit of the arbitration field is transferred and the conclusion of the message frame, our detector node can determine which message frame is being transmitted. Time measurements allow us to ascertain a 20-ms elapsed duration. In this example, the recovery procedure reduces the time it takes for the abnormal node to transition to a bus-off state to under 6 milliseconds, which is significantly shorter than the periodicity of the regular message frame. We plan to integrate the IPS into a working automotive system in the future and study the impact of false positives on threat prevention and the performance hit caused by reboot-based recovery.

REFERENCES

[1] Z. Liu, T. Tsuda and H. Watanabe, “Data Driven CyberPhysical System for Landslide Detection”, *Mobile Networks and Applications*, Vol. 24, No. 3, pp. 991-1002, 2019.

[2] N. Bibi, M.N. Majid, H. Dawood and P. Guo, "Automatic Parking Space Detection System", *Proceedings of International Conference on Multimedia and Image Processing*, pp. 11-15, 2017.

[3] S. Aljawarneh, M. Aldwairi and M.B. Yassein, "AnomalyBased Intrusion Detection System through Feature Selection Analysis and Building Hybrid Efficient Model", *Journal of Computational Science*, Vol. 25, pp. 152-160, 2018.

[4] Z. Li, C. Wang, C. Jiang and X. Li, "Multicast Capacity Scaling for Inhomogeneous Mobile Ad Hoc Networks", *Ad Hoc Networks*, Vol. 11, No. 1, pp. 29-38, 2013.

[5] S. Zhou and L. Ying, "On Delay Constrained Multicast Capacity of Largescale Mobile Ad-Hoc Networks", *Proceedings of International Conference on Communications and Networks*, pp. 1-7, 2010.

[6] J.P. Jeong, T. He and D.H.C. Du, "TMA: Trajectory-based Multicast Forwarding for Efficient Multicast Data Delivery in Vehicular Networks", *Computer Networks*, Vol. 57, No. 13, pp. 662-672, 2013.

[7] X. Ge, J. Yang, H. Gharavi and Y. Sun, "Energy Efficiency Challenges of 5G Small Cell Networks", *IEEE Communications Magazine*, Vol. 55, No. 5, pp. 184-191, 2017.

[8] Y. Arta and R. Kharisma, "Simulasi Implementasi Intrusion Prevention System (IPS) Pada Router Mikrotik", *IT Journal Research and Development*, Vol. 3, No. 1, pp. 104-114, 2018.

[9] A.S. Alqahtani and M. Alquraish, "On Implementing a Powerful Intrusion Prevention System Focused on Big Data", *The Journal of Supercomputing*, Vol. 77, No. 12, pp. 14039-14052, 2021.

[10] T. Prasetyo, "Pengamanan Jaringan Komputer Dengan Intrusion Prevention System (IPS) Berbasis Sms Gateway", *Journal Teknologi Pintar*, Vol. 2, No. 6, pp. 1-12, 2022.

[11] A. Krishna, A. Lal and M. Hari, "Intrusion Detection and Prevention System using Deep Learning", *Proceedings of International Conference on Electronics and Sustainable Communication Systems*, pp. 273-278, 2021.

[12] A. Dushimimana, T. Tao and R. Kindong, "Bi-Directional Recurrent Neural Network for Intrusion Detection System (IDS) in the Internet of Things (IoT)", *International Journal of Advanced Engineering Research and Science*, Vol. 7, No. 3, pp. 524-539, 2020.

[13] H. Larijani, J. Ahmad and N. Mtetwa, "A Heuristic Intrusion Detection System for Internet-of-Things (IoT)", *Proceedings of International Conference on Intelligent Computing*, pp. 86-98, 2019.

[14] T. Alves and T. Morris, "Embedding Encryption and Machine Learning Intrusion Prevention Systems on Programmable Logic Controllers", *IEEE Embedded Systems Letters*, Vol. 10, No. 3, pp. 99-102, 2018.

[15] G. Xian, "Cyber Intrusion Prevention for Large-Scale Semi-Supervised Deep Learning based on Local and Non-Local Regularization", *IEEE Access*, Vol. 8, pp. 55526-55539, 2020.