International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

X.1375

(10/2020)

SERIES X: DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

Secure applications and services (2) – Intelligent transportation system (ITS) security

Guidelines for an intrusion detection system for in-vehicle networks

Recommendation ITU-T X.1375

International Telecommunication Union

ITU-T X-SERIES RECOMMENDATIONS

DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

PUBLIC DATA NETWORKS	X.1–X.199
OPEN SYSTEMS INTERCONNECTION	X.200–X.299
INTERWORKING BETWEEN NETWORKS	X.300–X.399
MESSAGE HANDLING SYSTEMS	X.400–X.499
DIRECTORY	X.500–X.599
OSI NETWORKING AND SYSTEM ASPECTS	X.600–X.699
OSI MANAGEMENT	X.700–X.799
SECURITY	X.800–X.849
OSI APPLICATIONS	X.850–X.899
OPEN DISTRIBUTED PROCESSING	X.900–X.999
INFORMATION AND NETWORK SECURITY	
General security aspects	X.1000–X.1029
Network security	X.1030–X.1049
Security management	X.1050–X.1069
Telebiometrics	X.1080–X.1099
SECURE APPLICATIONS AND SERVICES (1)	
Multicast security	X.1100–X.1109
Home network security	X.1110–X.1119
Mobile security	X.1120–X.1139
Web security	X.1140–X.1149
Security protocols (1)	X.1150–X.1159
Peer-to-peer security	X.1160–X.1169
Networked ID security	X.1170–X.1179
IPTV security	X.1180–X.1199
CYBERSPACE SECURITY	
Cybersecurity	X.1200–X.1229
Countering spam	X.1230–X.1249
Identity management	X.1250–X.1279
SECURE APPLICATIONS AND SERVICES (2)	
Emergency communications	X.1300–X.1309
Ubiquitous sensor network security	X.1310–X.1319
Smart grid security	X.1330–X.1339
Certified mail	X.1340–X.1349
Internet of things (IoT) security	X.1360–X.1369
Intelligent transportation system (ITS) security	**X.1370–X.1389**
Distributed ledger technology security	X.1400–X.1429
Distributed ledger technology security	X.1430–X.1449
Security protocols (2)	X.1450–X.1459
CYBERSECURITY INFORMATION EXCHANGE	
Overview of cybersecurity	X.1500–X.1519
Vulnerability/state exchange	X.1520–X.1539
Event/incident/heuristics exchange	X.1540–X.1549
Exchange of policies	X.1550–X.1559
Heuristics and information request	X.1560–X.1569
Identification and discovery	X.1570–X.1579
Assured exchange	X.1580–X.1589
CLOUD COMPUTING SECURITY	
Overview of cloud computing security	X.1600–X.1601
Cloud computing security design	X.1602–X.1639
Cloud computing security best practices and guidelines	X.1640–X.1659
Cloud computing security implementation	X.1660–X.1679
Other cloud computing security	X.1680–X.1699
QUANTUM COMMUNICATION	
Terminologies	X.1700–X.1701
Quantum random number generator	X.1702–X.1709
Framework of QKDN security	X.1710–X.1711
Security design for QKDN	X.1712–X.1719
Security techniques for QKDN	X.1720–X.1729
DATA SECURITY	
Big Data Security	X.1750–X.1759
5G SECURITY	X.1800–X.1819

For further details, please refer to the list of ITU-T Recommendations.

Recommendation ITU-T X.1375

Guidelines for an intrusion detection system for in-vehicle networks

Summary

Recommendation ITU-T X.1375 establishes guidelines for an intrusion detection system (IDS) for in-vehicle networks (IVNs). Recommendation ITU-T X.1375 mainly focuses on how to detect intrusion and malicious activities on IVNs such as those using a controller area network (CAN) that cannot be supported by general IDSs currently deployed on the Internet.

Recommendation ITU-T X.1375 includes classifications and analyses of attacks targeting IVNs. Recommendation ITU-T X.1375 then proposes methodologies and implementation guidelines for detecting intrusions and malicious activities within CAN-based IVNs that cannot be supported by general IDSs.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T X.1375	2020-10-29	17	11.1002/1000/14447

Keywords

Anomaly detection, controller area network (CAN), intelligent transportation system (ITS), intrusion detection system (IDS), in-vehicle network (IVN), misuse detection, security requirement, security threat, vehicle.

---

* To access the Recommendation, type the URL http://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID. For example, http://handle.itu.int/11.1002/1000/11830-en.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

Table of Contents

Recommendation ITU-T X.1375

Guidelines for an intrusion detection system for in-vehicle networks

1 Scope

This Recommendation establishes guidelines for an intrusion detection system (IDS) for in-vehicle networks (IVNs). This Recommendation identifies threats to IVNs such as a controller area network (CAN), which is widely used in modern vehicles.

This Recommendation mainly focuses on aspects of detecting intrusion and malicious activities in IVNs (such as a CAN) that cannot be supported by general IDSs currently deployed on the Internet.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T X.1371] Recommendation ITU-T X.1371 (2020), *Security threats to connected vehicles*.

[ITU-T X.1372] Recommendation ITU-T X.1372 (2020), *Security guidelines for vehicle-to-everything (V2X) communication*.

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 availability [b-ITU-T X.800]: The property of being accessible and useable upon demand by an authorized entity.

3.1.2 confidentiality [b-ITU-T X.800]: The property that information is not made available or disclosed to unauthorized individuals, entities, or processes.

3.1.3 controller area network (CAN) [b-ISO/IEC/IEEE 24765]: High-integrity bus system for networking intelligent devices within a system.

NOTE – Commonly used in embedded networks for vehicles or medical equipment.

3.1.4 exploit [b-ISO/IEC 27039]: Defined way to breach the security of information systems through vulnerability.

3.1.5 integrity [b-ITU-T X.800]: The property that data has not been altered or destroyed in an unauthorized manner.

3.1.6 intrusion [b-ISO/IEC 27039]: Unauthorized access to a network or a network-connected system, that is, deliberate or accidental unauthorized access to information systems, to include malicious activity against information systems, or unauthorized use of resources within information systems.

3.1.7 intrusion detection [b-ISO/IEC 27039]: Formal process of detecting intrusions, generally characterized by gathering knowledge about abnormal usage patterns, as well as what, how, and which vulnerability has been exploited to include how and when it occurred.

3.1.8 intrusion detection system (IDS) [b-ISO/IEC 27039]: Information systems used to identify that an intrusion has been attempted, is occurring, or has occurred.

3.1.9 threat [b-ISO/IEC 27000]: Potential cause of an unwanted incident, which can result in harm to a system or organization.

3.2 Terms defined in this Recommendation

None.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

ADAS	Advanced Driver Assistance System
AMP	Arbitration on Message Priority
AVN	Audio-Video Navigation
CAN	Controller Area Network
CANFD	Controller Area Network with Flexible Data rate
CNN	Convolutional Neural Network
CRC	Cyclic Redundancy Code
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CVE	Common Vulnerabilities and Exposures
DLC	Data Length Code
DNN	Deep Neural Network
DoS	Denial of Service
ECU	Electronic Control Unit
GAN	Generative Adversarial Network
HTTP	Hypertext Transfer Protocol
ID	Identifier
IDS	Intrusion Detection System
IFS	Interframe Space
IVI	In-Vehicle Infotainment
IVN	In-Vehicle Network
LEN	Length
LIN	Local Interconnected Network
LRD	Long-Range Dependence
MOST	Media-Oriented System Transport
MQTT	Message-Queuing Telemetry Transport
OBD	On-Board Diagnostic
OEM	Original Equipment Manufacturer
OTA	Over The Air

PLC	Power Line Communication
RTR	Remote Transmission Request
SD	Secure Digital
SIEM	Security Information and Event Management
SOC	Security Operations Centre
SOF	Start of Frame
TEC	Transmit Error Counter
TLS	Transport Layer Security
USB	Universal Serial Bus
VIN	Vehicle Identification Number
V2X	Vehicle-to-Everything
Wi-Fi	Wireless Fidelity

5 Conventions

None.

6 In-vehicle intrusion detection systems

An IVN is an internal network of a vehicle used for the communication between car components such as electronic control units (ECUs). Well-known examples of IVNs include a CAN, controller area network with flexible data rate (CANFD), local interconnected network (LIN), media-oriented system transport (MOST) and FlexRay.

The architecture of an IDS for IVN, e.g., in-vehicle IDS, as depicted in Figure 1, consists of elements such as a message queue, rule engine, interface manager, crypto module and storage. The general architecture of IDS is described in Appendix I.

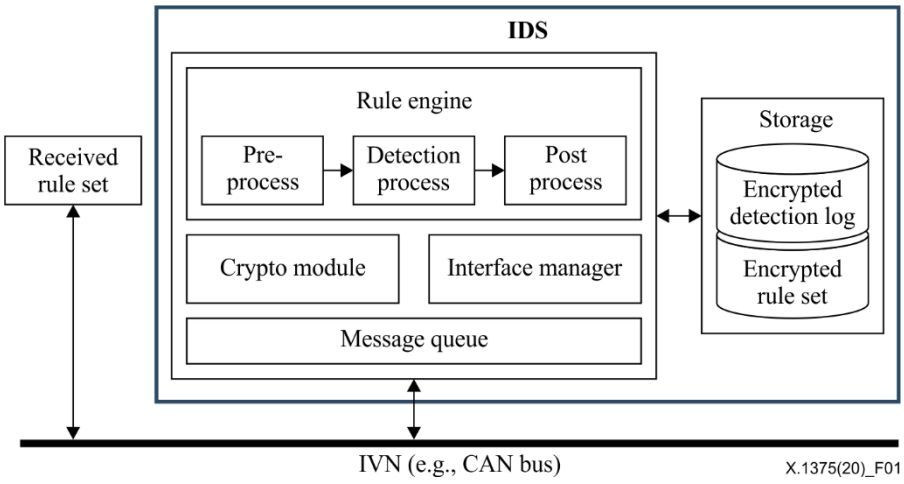

Figure 1 – Architecture of an in-vehicle intrusion detection system

The major elements illustrated in Figure 1 are described as follows:

– message queue, which collects all messages in a CAN bus and handles requests for collected traffic data from other IDS modules;

– rule engine, which includes three processes:

    - pre-process: comprising functions enabling a reset of the IDS and management of detection rules based on the rule set received from a backend server,
    - detection process: comprising the malicious messages detection function,
    - post process: comprising the function of decision handling of malicious messages,
- crypto module, which enables the encryption and decryption of detection logs, as well as management of encryption keys:
    - when IDS is running, the crypto module runs as a background server process (i.e., daemon),
    - when detected events are written in storage, the crypto module performs encryption functions,
    - likewise, when any process requests encryption or decryption tasks, the crypto module responds to the request;
- interface manager, which connects one or multiple CANs to download the rule set(s) and upload detection log(s) to storage or back-end server(s);
- detection log and rule sets, whose files are securely saved in storage.

6.1 Functions of in-vehicle intrusion detection system

The main function of IDS is to analyse incoming traffic (e.g., broadcast CAN messages) in order to detect attacks. The IDS generates detection logs and reports detected events to relevant recipients.

An in-vehicle IDS needs to perform the following functions:

- storage and updating rule sets;
- collection of IVN (e.g., CAN) messages;
- analysis of data (collected messages) using rule set(s);
- detection of intrusions resulting from the analyses performed on these data;
- reporting of detection results;
- storage of detection logs.

6.1.1 Collection of controller area network messages

An in-vehicle IDS collects and monitors the CAN message flow in IVNs. Collection functions are as follows:

- collection of messages and additional information, such as timestamp or bus channel information;
- data reduction, preferably at the source, to reduce data loads;
- pre-processing of raw data to eliminate redundancy and false alarms.

6.1.2 Detection of intrusion

An in-vehicle IDS functions to detect intrusions. This includes detection:

- of intrusions to a designated CAN bus;
- of denial of service (DoS) attacks, e.g., CAN message flooding;
- based on content, e.g., the data field in a CAN frame;
- based on context, e.g., the arbitration field in a CAN frame.

The detection procedure can be separated into three parts: pre-processing; detection processing; and post processing, as illustrated in Figure 2.

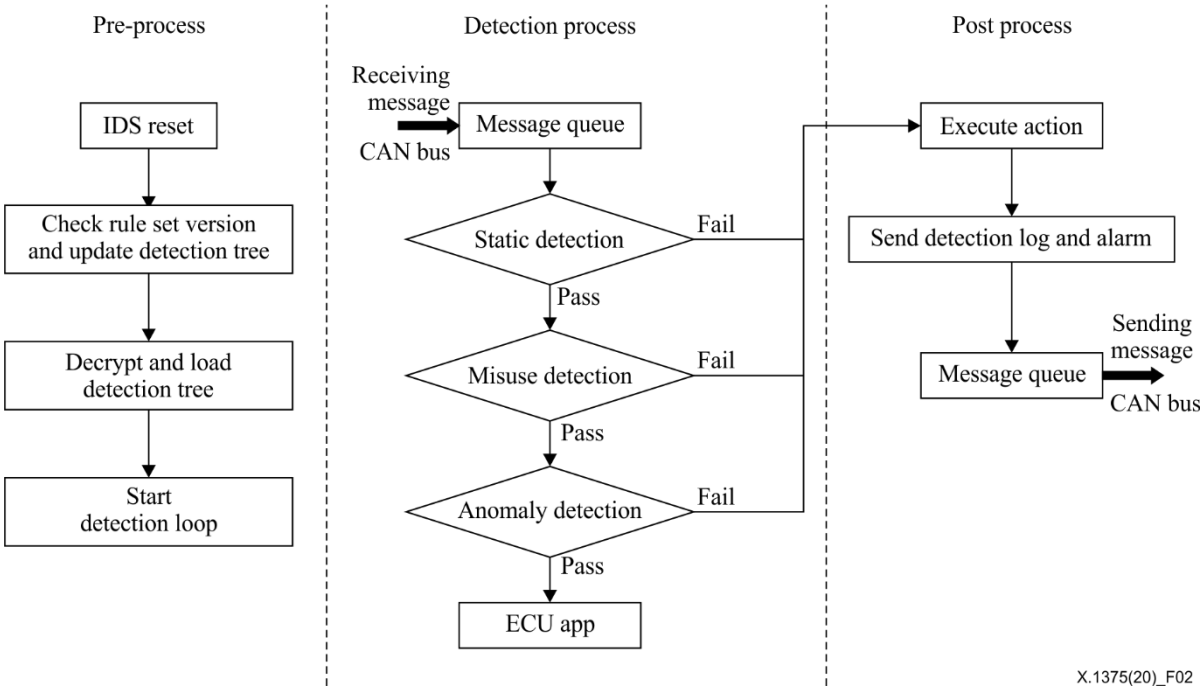

Figure 2 – Overall procedure in the rule engine

Details of the individual processes illustrated in Figure 2 follow.

– Pre-process is activated when the vehicle is started. This process includes the following steps:
  - resetting of IDS: this step resets all elements in the IDS;
  - check of rule set: this step checks the version of the rule set and updates detection rules, detection models, etc. (i.e., plugins);
  - decryption and loading of detection rules: this step decrypts and loads detection rules into the internal memory of the IDS;
  - start of detection loop: this step starts the detection loop while the vehicle is running.

– The detection process runs while the vehicle is operating. The detection process consists of the following.
  - Message queue: this represents the sub-process for message receipt from CAN, and their storage and forwarding.
  - Static detection: this represents the sub-process for checking the validity of CAN messages by comparing each message identifier (ID), data length code (DLC) and period with stored values in CAN database.
  - Misuse detection (signature-based model): represents the sub-process for detecting malicious messages based on the analysis of vehicle vulnerabilities (e.g., known vulnerabilities indexed in a common vulnerabilities and exposures (CVE) database) without considering the status of the vehicle or detecting exploit codes (i.e., hacking programs to gain high privilege of the target system or network).
  - Anomaly detection: this represents the sub-process for comparing the status of a vehicle to that of what a normal situation would be. This comparison is performed based on the detection rule. It includes detection models based on entropy, self-similarity, interval, sequence, offset and survival. If no anomaly is found, then the received messages can be sent to an ECU app (an ECU or ECU-related application program).

In each detection (static, misuse, anomaly) process, "Pass" means that nothing has been detected and "Fail" means that something has.

– The post process runs when any intrusion is detected. The post process consists of:
  - execute action: execute specific actions (e.g., display malfunction sign to the dashboard display cluster) by original equipment manufacturer (OEM) configurations;
  - send detection log and alarm: records that include the detected events or information that can be stored or displayed by OEM configurations.

If there is a failure in the detection process (e.g., IDS is not working or IDS cannot monitor IVNs), a detection log and alarm signal are transferred to the back-end server and driver in the post process.

6.1.3 Analysis of data

An in-vehicle IDS requires functions for data analyses. This includes the following.

– Determination of whether exploitation has occurred by using information about a discovered or potential vulnerability. Information pertaining to a discovered or potential vulnerability can be found in a CVE database or other cyber threat intelligence from various sources.
– Correlation of raw or pre-processed data, such as association of multiple attacks occurring at different targets with the same messages.
– Aggregation of inputs from several sources of the same message type.
– Fusion of inputs from multiple sources of disparate types.
– Support for other functions, such as trend analyses and next-target predictions.

6.1.4 Reporting of detection results

An in-vehicle IDS requires a reporting function to notify and report on detected events. This includes the following.

– reporting to designated recipients about detected suspicious events based on the configuration of the IDS (e.g., report to a console screen, dashboard or mobile device);
– logging of detected events and alarms (see clause 6.1.5);
– provision of one or more methods to report detected events, as well as warnings to the driver, manager or backend server;
– allowing only authorized persons to access reports;
– maintainence of integrity and confidentiality of data reporting to a backend server;
– creation of reports for later follow-up regarding suspicious messages;
– reports should include CAN messages, start and end times of events, as well as descriptions of suspicious events.

6.1.5 Storage of detection logs and rule sets

An in-vehicle IDS should be equipped with suitable local storage and the means to store detection logs and rule sets. This includes the following:

– automatic recording and storage of the detected events as logs;
– applyication of methods to encrypt and decrypt logs and rule sets;
– check or validation of the integrity of rule sets.

6.1.6 Updating rule sets

An in-vehicle IDS should provide the means to update rule sets. This includes the following:

– updating a rule set via means such as over-the-air (OTA) updates or a universal serial bus (USB) memory stick;
– application of methods to encrypt and decrypt rule sets in order to protect rule set confidentiality and integrity;

– support for failure recovery or rollback of rule sets.

7 Identified threats to in-vehicle networks

7.1 General

General security threats related to connected vehicles are specified in clause 6 of [ITU-T X.1371]. Among various threats to vehicles, external connectivity and connections can especially affect IVNs.

Likewise, security threats in the vehicle-to-everything (V2X) environment are specified in clause 7 of [ITU-T X.1372]. If any vehicle that has inter-connectivity between IVNs and external network interfaces connected with V2X communication, then threats identified in [ITU-T X.1372] can affect IVN security as a result.

In clauses 7.2 to 7.4, regardless of whether interconnectivity of IVNs and external network interfaces exists, threats directly related to IVNs are described. Note that the threats listed are well-known threats that can affect IVNs. Note also that state-of-the-art of intrusion detection methods cannot exhaustively detect all threats described in this Recommendation.

Because of the IVN characteristics described in Appendix I, there is no perfect countermeasure that can be implemented especially in CAN to block sniffing and wire-tapping attacks; thus, all attacks related to sniffing or wire-tapping cannot be easily detected by in-vehicle IDS. Attacks listed in clauses 7.3 and 7.4 can be detected by an in-vehicle IDS with various methods described in clause 8.2.

7.2 Threats to confidentiality

Attacks related to IVNs can pose threats to confidentiality. These threats affect most bus-type protocols. Clauses 7.2.1 and 7.2.2 provide examples of threats that can impact confidentiality.

7.2.1 Sniffing

In the case of IVNs, a sniffing attack involves an attacker gaining access to record all data on the network.

Data sniffing or eavesdropping has been demonstrated to be easily accomplished on the local CAN bus type of IVN since all its communication messages are sent across the entire CAN bus and a sniffer will simply accept all incoming data (see Figure 3).

CAN is a broadcasting-based bus protocol. This means any connected node (e.g., ECU) on the bus can listen to any messages. This characteristic of the CAN bus allows an attacker to analyse the CAN ID and the meaning of the payload. As a result, an attacker can know the relationship between a specific CAN ID and its related actuation to control the function of the vehicle.

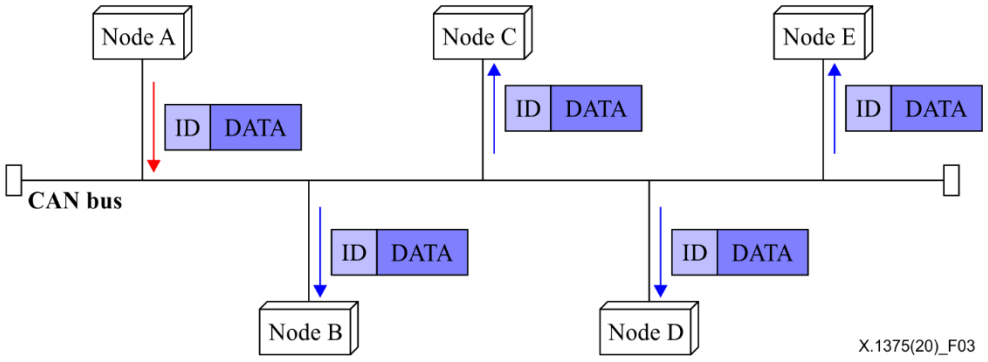

Figure 3 – Example of sniffing

7.2.2 Wire-tapping

In the case of IVNs, a wire-tapping attack involves an attacker gaining physical access to the network in order to record all data on the network.

Most IVNs are vulnerable to physical wire-tapping when an attacker can successfully access the target car physically. Once an attacker has accessed the target car, they can connect a tapping device to capture and monitor (sniff) all traffic on a CAN bus. Modern vehicles have multiple CAN buses. Access between buses is strictly controlled by a security gateway (i.e., in-vehicle firewall). Therefore, an attacker cannot monitor all traffic of all CAN buses simply by sniffing (as desrcribed in clause 7.2.1), unless the attacker can also obtain the access privilege to the security gateway. However, when attackers can access a target car physically, they can use wire-tapping to sniff all traffic (see Figure 4).

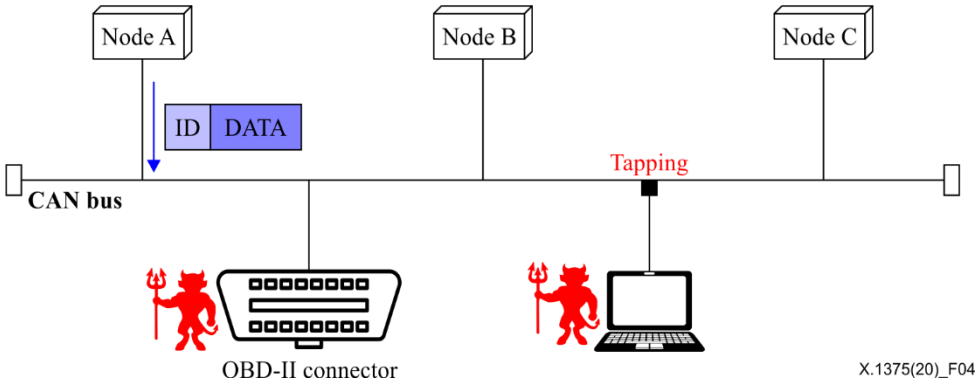

Figure 4 – Illustration of wire-tapping

7.3 Threats to integrity

Attacks on IVNs can pose threats to data integrity. These threats can affect most bus-type protocols. Clauses 7.3.1. to 7.3.3 provide examples of threats that can impact integrity.

7.3.1 Impersonation attack

In the case of IVNs, an impersonation attack is one where an attacker injects a well-crafted CAN frame with a specific CAN arbitration ID and data value.

An attacker can stop message transmission by controlling the target node and can plant or manipulate an impersonating node. If a victim node stops transmitting, all messages sent by the targeted node are removed from the bus. However, when an ECU receives a remote data frame, it is designed to transmit it immediately. If a receiver node does not receive a response to the remote frame, this can indicate that the existing node has been attacked or is broken. In this case, an attacker can plant an impersonating node to respond to a remote frame. Thus, the impersonating node broadcasts data frames periodically and responds to the remote frame as if pretending to be the targeted node (see Figure 5).

An attacker requires sufficient pre-knowledge about all CAN ID payload data to control vehicles. In order to trigger specific actuation of the vehicle by an impersonation attack, an attacker needs to send the exactly related CAN frame data with a specific arbitration ID and its payload data. Also, the attacker needs to know the timing and offset value to send the crafted CAN frame effectively.

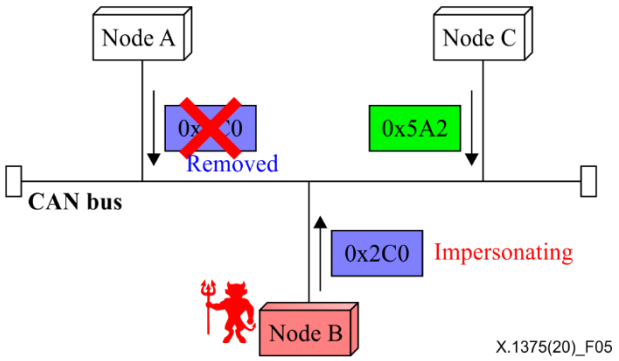

Figure 5 – Example of an impersonation attack

7.3.2 Fuzzy attack

In the case of IVNs, a fuzzy attack involves the injection of messages of spoofed random CAN IDs and payload data.

An attacker can inject messages of randomly spoofed IDs with arbitrary data. As a result, all nodes will receive numerous random messages with the intent to cause unintended vehicle behaviours (see Figure 6). To exploit the fuzzy attack, an attacker observes in-vehicle messages and selects target IDs to produce unexpected behaviours.

Unlike DoS attacks, fuzzy attacks paralyse functions of the vehicle rather than delaying normal messages via occupancy of the bus. In addition, attackers do not require pre-knowledge about all CAN IDs and their payload data (as in the impersonation attack in clause 7.3.1) to control vehicles. Simply sending CAN frame data with random arbitration IDs and associated payload data can cause serious malfunctions of the vehicle that may imperil driver safety.

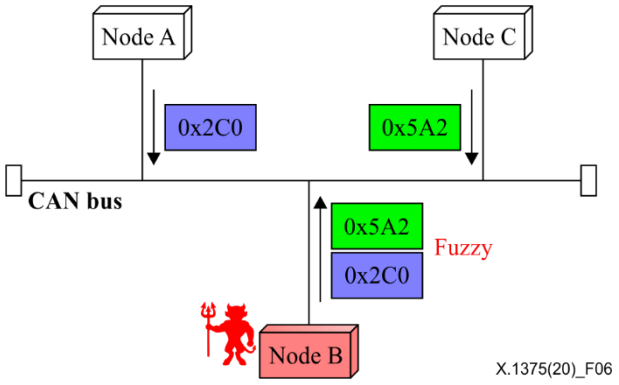

Figure 6 – Example of a fuzzy attack

7.3.3 Replay attack

In the case of IVNs, a replay attack is one that involves injecting messages with legitimate arbitration IDs and payload data.

An attacker can sniff any message and then retransmit it in order to make a target ECU or a target vehicle run certain instructions. Also, the sniffed messages can be stored in the dump file for future usage. In a replay attack, the attacker easily reproduces vehicle functions, such as manipulating windows, shifting gears and accelerating an engine. This is due to the fact that the arbitration ID and its payload of frames in the sniffed messages are valid since they were generated by legitimate CAN nodes. In addition, the CAN protocol does not consider information about the source of CAN frames. As a result, receiver nodes deal with payload that is suitable for the target vehicle. By continuously transmitting the sniffed messages, the attacker can override normal control of the vehicle (see Figure 7).

A replay attack can be performed successfully even when the attacker does not have pre-knowledge about any arbitration IDs and payload data. Meanwhile, two limitations exist in the replay attack:

1) the attacker must have captured CAN frames on the same model of the target vehicle;

2) the dump file should be captured under the specific circumstances for which the attacker is intending to mimic, e.g., if the attacker wants to blink the "door open" sign in the instrument cluster by using a replay attack, then the dump must capture data when the target door is open.

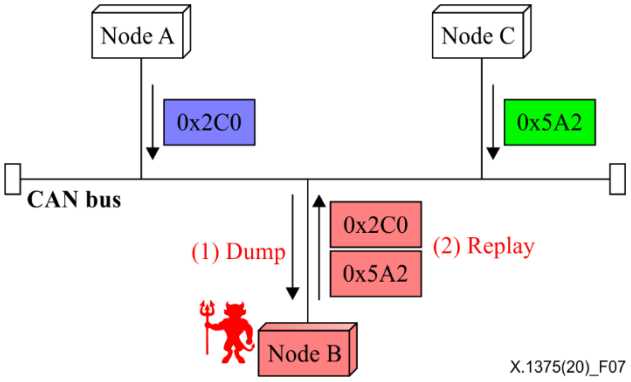

Figure 7 – Example of a replay attack

7.4 Threats to availability

Attacks to IVNs can pose threats to availability. These threats can affect most bus-type protocols. Clauses 7.4.1 and 7.4.2 provide examples of threats that can impact availability.

7.4.1 Denial of service attack

In the case of IVNs, a DoS attack involves an attacker injecting messages with high priority CAN frame data in a very short cycle.

An attacker can inject high priority messages in a short cycle on the bus. Because the CAN protocol does not provide authentication as a requirement for message insertion at the physical or link layer, any compromised node or malicious node can send modified CAN data frames. In addition, CAN has limited bandwidth (e.g., 1 Mbit/s). An attacker can easily cause a DoS attack to all nodes on the bus by overwhelming CAN with high priority messages. As a result, the vehicle may not be able to respond to normal control messages, which can seriously imperil the safety of the vehicle. Typical DoS attack messages aiming to occupy the CAN bus using the theoretically highest priority ID such as "0x000" are illustrated in Figure 8.

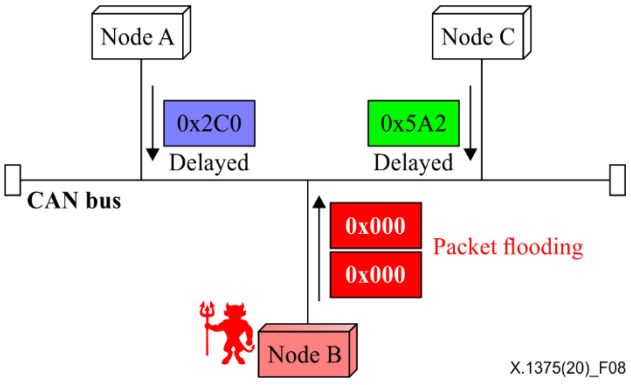

Figure 8 – Example of a packet flooding denial of service attack

7.4.2 Frame-drop attack

In the case of IVNs, a frame-drop attack involves an attacker injecting messages simultaneously with another node.

The frame-drop attack is a type of DoS attack, which makes transmissions of CAN frames containing a specific arbitration ID. According to the CAN protocol, this will fail to work. The frame-drop attack exploits a weakness of the arbitration phase of the CAN protocol that is invoked when nodes send CAN frames with the same arbitration ID.

Generally, when two or more nodes have CAN frames in the transmit (Tx) buffers, they send CAN frames immediately after an interframe space (IFS). The arbitration phase is responsible for preventing bus collisions in such situations by checking the arbitration ID field. However, if more than two nodes send the CAN frame with the same arbitration ID, arbitration does not work properly. In this situation, a bit-monitoring error could occur and the transmit error counter (TEC) of transmitting nodes rises.

When two or more nodes send CAN frames with the same arbitration ID, frame-drop occurs. As a result, legitimate frames cannot be delivered. If frame-drop attacks occur repetitively, some CAN nodes in a target vehicle will have high TEC values, possibly resulting in the node status changing to either error passive or bus off. In either case, such CAN nodes cannot send CAN frames for some period of time (see Figure 9).

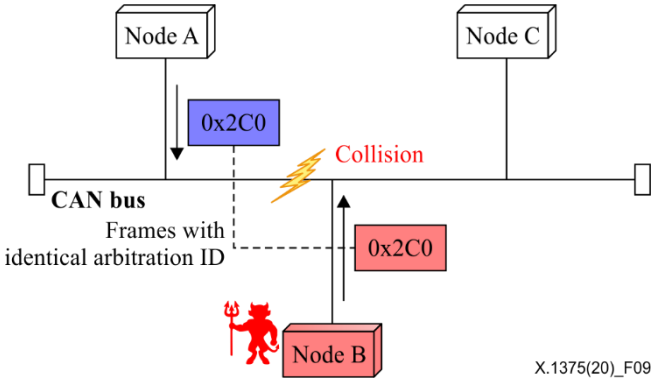

Figure 9 – Example of a frame-drop denial of service attack

8 Implementation guidelines for in-vehicle intrusion detection System

8.1 Points of installation for in-vehicle intrusion detection system

This clause identifies potential points of installation for an in-vehicle IDS and the features enabled or associated with each position.

Each identified point of installation has its own merits. The characteristics of each point of installation are summarized in Table 1. See Figure 10 for a reference to each point of installation. It is noted that IDSs can be deployed in any point of installation described in Table 1, but it should be guaranteed that all messages from each CAN bus are monitored and examined by the installed IDSs.

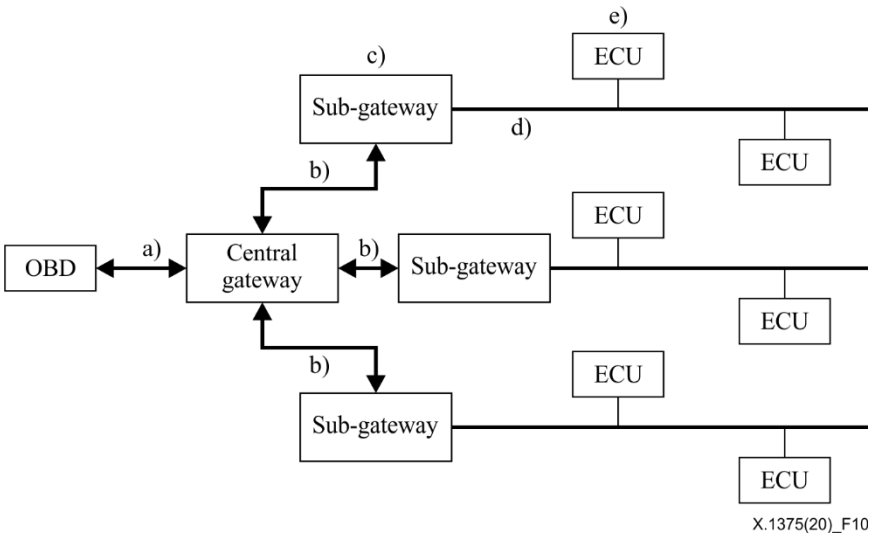

Figure 10 – General controller area network bus topology with possible points of installation (a) to (e) for an intrusion detection system

In Figure 10, the central gateway is the central communication node which acts as a router and is the gateway for all data coming into the vehicle. In addition, each sub-gateway represents the local communication node that is in charge of a specific sub-system domain, such as powertrain, chassis, body and multimedia.

Table 1 – Possible points of installation for an intrusion detection system and associated attributes

Point of installation	Attributes
(a) On central gateway	All attacks coming from outside the vehicle through the on-board diagnostic (OBD) port can be detected. However, too many messages may be collected and processed from this point of installation, making it difficult to decide on intent. That is, it may be hard to distinguish normal and malicious messages received at this point.
(b) Behind central gateway	The central gateway inherently provides some filtering of received messages. After all messages are filtered by the central gateway, a relatively small number of, but in aggregate more, informative messages can be gleaned at this point. An IDS positioned here can detect malicious messages injected directly to a CAN bus by an attacker.
(c) On sub-gateway	All messages to any of these points are related to a specific CAN domain, such as power train, chassis, multimedia or body. Accordingly, an IDS here can detect inconsistencies between messages in point (b) and messages used in a specific domain. Moreover, any attack from one domain to another can be detected by an IDS located at this point.
(d) Behind sub-gateway	If an ECU is damaged or replaced with a malicious ECU by an attacker and the attacker directly connects to a specific CAN bus domain, it will be impossible for an IDS in the central gateway and the sub-gateway to detect attacks. For example, when a certain ECU is wired directly with another device, the ECU does not need to broadcast a message. In such a case, the attack cannot be detected. Therefore, an IDS placed at this point is used when it is of interest to assess the trustworthiness of an ECU and to monitor a specific CAN domain.
(e) On ECU	Deploying an IDS at this point is useful for detecting loss of data in an ECU and for detecting any misbehaviour of the ECU by comparison with the ground-truth value of factory settings.
NOTE – See Figure 10, which illustrates the points of installation.	

8.2 Intrusion detection methodology

Any IDS can be classified as based on: a) static detection; b) misuse detection or signature; or c) anomaly (or a combination of any of these) according to the detection methodologies employed. Among the three types of detection method, the static detection method should be equipped in an IDS. This detection method can perform basic network examination based on the deterministic network configuration parameters, and does not require many computation overheads.

Misuse detection- or anomaly-based detection can be selectively used in IDS according to the IVN situation, such as hardware resource capacity. However, it is essential to detect misuse patterns (i.e., known attacks) as well as anomalous behaviour patterns (i.e., unknown attacks) in IVNs, because the consequences of an unknown attack could critically impact driver safety. To this end, and as highlighted in Table 1, IVNs should include IDSs with both misuse and anomaly detection-based methods in order to cover all attack possibilities.

Among the identified threats described in clause 7, some attacks that have an exact pattern can be detected by one or more detection methods such as static detection. Some attacks that have a dynamically changing pattern can be detected by anomaly-based detection or combinations of detection methods.

Each detection method is further described in clauses 8.2.1 to 8.2.4.

8.2.1 Static detection

The static detection method is used to detect violations in the data settings of the CAN ID, CAN DLC, CAN signal value and CAN bus domain.

Based on static detection methods, an IDS can check the validity of a CAN message by comparing its message ID, DLC and its period with stored values in a CAN database that includes the ground-truth values. To achieve high accuracy, the CAN database with ground-truth values can be provided by the OEM. As a result, static detection shows high accuracy because it does precise comparison tasks based on manufacturer factory values.

8.2.2 Misuse detection

The misuse patterns of IVNs will likely increase as attack techniques advance. Therefore, as known attack patterns are identified, these patterns should be updated regularly in the IDS by adequate methods (e.g., manual update, OTA).

Further, attack patterns should be developed and tested completely by trusted parties to minimize side-effect and false-positive errors.

To increase detection accuracy, an IDS can adopt multiple misuse detection models simultaneously. Also, misuse detection models can be flexibly changed, added and deleted according to vehicle circumstances. To this end, detection models need to be implemented as plugins.

To reduce the overall IDS overhead, the rule set (signatures for known attacks) should be loaded selectively. In general, the look-up speed depends on the number of loaded rules, as well as the performance of string match algorithms (typically based on the regular expression matching) used when interrogating CAN messages.

8.2.3 Anomaly detection

Known attacks can be handled well by using misuse detection models, which are nonetheless unable to detect attacks that are still not discovered (or not reported to the public) and covered by rule sets. In order to detect unknown attacks and anomalous behaviours, an anomaly detection model is required as well as a misuse detection model.

In general, anomaly detection models require more computation resources. Therefore, an IDS employing this technique requires a lightweight anomaly detection model, e.g., one thatis based on

entropy, self-similarity or statistics, in order to detect and respond to anomalous behaviours in real time.

In addition to using the misuse detection model, the IDS should adopt multiple anomaly detection models simultaneously. Also, anomaly detection models can be flexibly changed, added and deleted according to vehicle circumstances. To this end, the detection models need to be implemented as plugins.

8.2.3.1 Entropy-based model and plugin

Entropy-based IDS is one lightweight detection method that can be useful for finding an abnormal status in IVNs.

While monitoring traffic, the IDS routinely calculates the entropy value of the traffic. The entropy value can be calculated based on the variety of CAN message IDs. In a normal network, the frequency of a CAN message ID in a given time window has a stable value. The interval of CAN message ID occurrence also has a stable value. Several properties, such as frequency and intervals of CAN message IDs, can be used to estimate the entropy value. In an attack-free state, the entropy value does not show high fluctuations, because most CAN message IDs have static patterns, i.e., each CAN message ID has its own characteristics such as upcoming frequency and intervals between messages.

When an attack occurs in the network. e.g., under injection-based attacks such as a DoS attack, a distributed DoS attack or spoofing attack, there will be a drastic change in the entropy value because there have been no accompanying changes of network configuration and connected nodes (which could provide an otherwise plausible explanation for the change in entropy value).

Thus, by monitoring for drastic changes of entropy value, an IDS can determine whether there are incoming attacks.

8.2.3.2 Self-similarity-based model and plugin

The self-similarity property is used to find repetitive patterns, especially periodic trends, of the series of actions and their frequency. When attackers try to send any packets over the target network, the self-similarity property changes. This is especially important, since the self-similarity property does not easily change unless there is a drastic configuration change. Therefore, an IDS can detect the attack using this property.

The self-similarity property exists in all network environments, and in networks that rarely change, strong self-similarity properties are seen. This is especially relevant in automotive network environments.

Among many self-similarity measures, Hurst parameter-based monitoring [b-Kettani] is adaptable to monitor anomalous behaviours of IVNs. The Hurst parameter is a measure that characterizes long-range dependence (LRD), which is a statistical phenomenon shown in self-similar processes. A self-similar process that has LRD can be regarded as a process with a long memory. The Hurst parameter is used to measure the extent of LRD and is a real number between [0, 1]. When the property of LRD is significantly found, which means that the self-similarity property is strong, then the Hurst parameter is close to 1. Figure 11 shows a process for self-similarity-based anomaly detection.

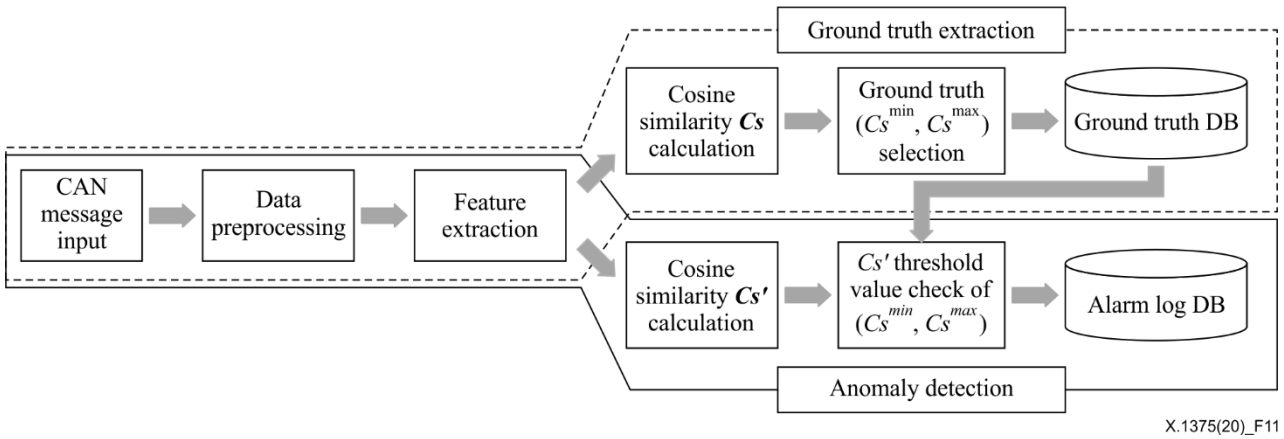

Figure 11 – Self-similarity-based anomaly detection

To estimate self-similarity, CAN messages are used as inputs. To select meaningful features, the collected CAN messages are preprocessed, and meaningful CAN messages (e.g., braking or acceleration related CAN messages) can be used.

To build ground truth (baseline) data, the CAN data needs to be collected in an attack-free state. The baseline distance of the selected features can then be estimated by cosine similarity (***Cs***); this ***Cs*** value has a boundary, $\boldsymbol{Cs}^{\min}$ and $\boldsymbol{Cs}^{\max}$.

This cosine similarity value needs to be estimated periodically. When a newly estimated cosine similarity value ***Cs***' does not fall within the boundary $\boldsymbol{Cs}^{\min}$ and $\boldsymbol{Cs}^{\max}$, this can be regarded as an anomaly event.

8.2.3.3 Survival-based model and plugin

Survival analysis uses statistical methods to analyse the expected duration of time until one or more events occurs, such as failure in mechanical systems. Survival analysis can estimate the proportion of a population of observed CAN message IDs by comparing with a certain past time window. Because CAN messages IDs have their own transmission interval or frequency, the overall survival ratio for a certain CAN message ID has a stable value.

Figure 12 shows anomaly detection based on the survival analysis model. The system contains two main parts: (1) the chunk-based threshold measurement; and (2) the detection algorithm. The survival analysis model is a statistical method used to discover which factors affect the survival rate and survival duration of a measurement object. This model focuses on the survival rate of an individual CAN ID in a specified chunk unit. A chunk unit is a time window for the observation of the distribution of a CAN ID.

The survival analysis model-based detection algorithm is used to determine whether the CAN message is normal by comparing the ground-truth value with the survival rate of the CAN message ID. When an injection attack such as flooding, fuzzy, or malfunction occurs, the survival rate of CAN IDs exhibits a different pattern from the ground-truth value.

For example, the survival rate of CAN IDs during flooding and fuzzy attacks is lower than the minimum value of the ground-truth. The injected CAN ID for flooding and fuzzy attacks was not a CAN ID specified in the manufacturers' guidelines. The flooding attack was performed with a higher priority of CAN IDs, while the fuzzy attack was performed with randomly generated CAN IDs. A large number of attack packets was injected into a chunk during the attacks, resulting in lower survival rates. On the other hand, the survival rate of CAN IDs during the malfunction attack was significantly higher than the maximum value of the ground-truth. Because the malfunction attack was performed through the CAN IDs specified by the manufacturer, the number of injected ones was added to those already existing, significantly increasing their total number.

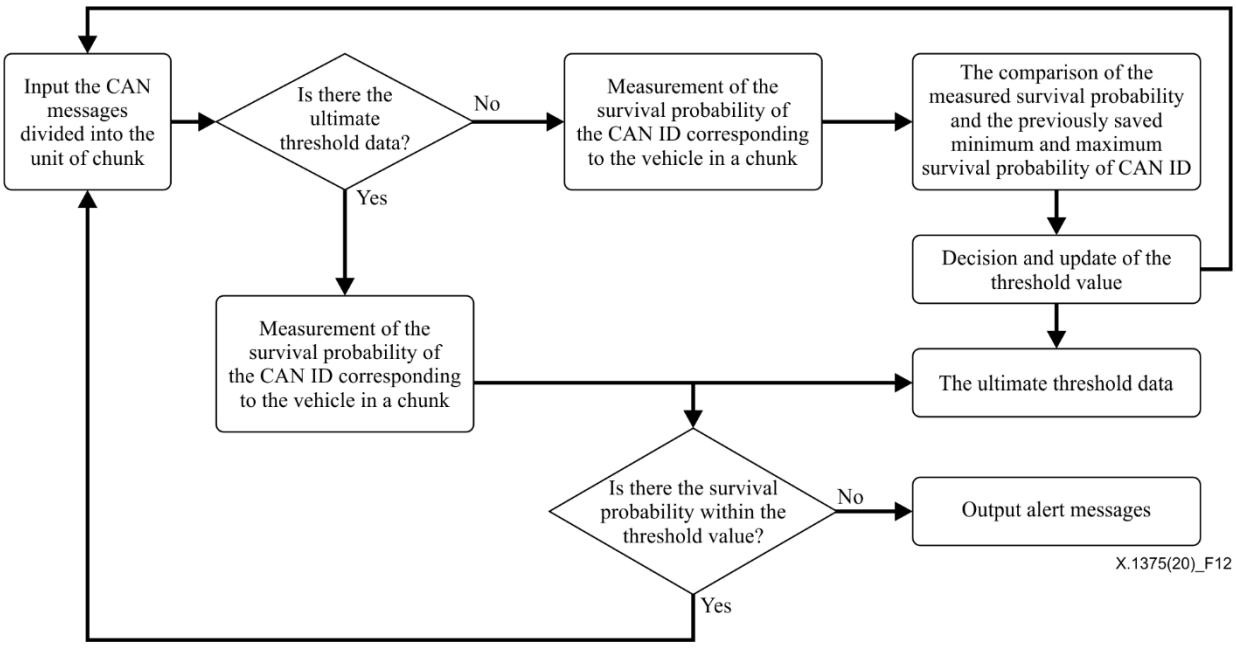

Figure 12 – Overview of the anomaly-intrusion detection scheme based on the survival analysis model

8.2.3.4 Interval-based model and plugin

If there are injected attacks, the arrival time interval for each CAN ID found in observed messages changes. By monitoring the drastic change of the time interval for each CAN message, IDS can detect the attacks.

Figure 13 illustrates the interval changes when an injection attack occurs, in which 0x1, 0x2 and 0x3 represent CAN IDs. These CAN IDs are routinely delivered by their own periodic time. However, when attackers attempt to inject an 0x2 message into the CAN bus, then the original interval of the message (0x2) is changed. By using these characteristics, the IDS can detect incoming attacks.

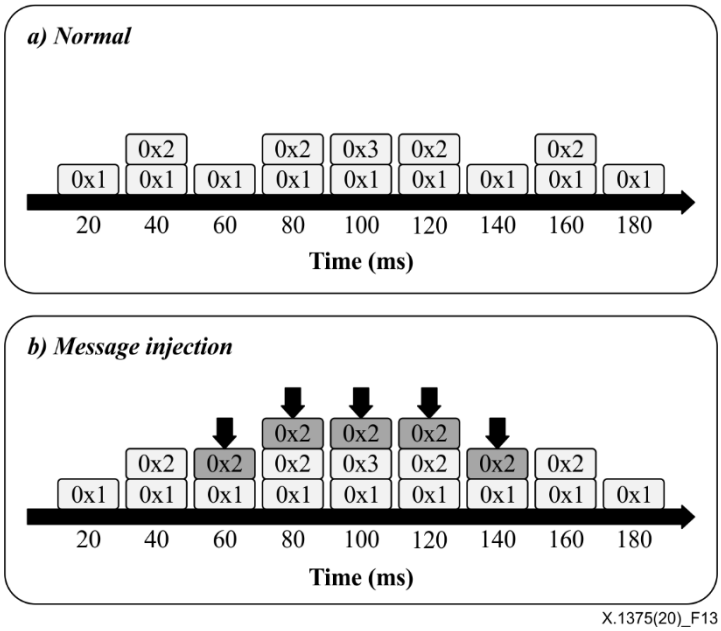

Figure 13 – Interval-based anomaly detection

8.2.3.5 Sequence-based model and plugin

In general, a single CAN message or combination of CAN messages are involved in actuation.

Thus, in normal (attack-free) status, an IDS should collect a CAN message or their combination to build the CAN message sequence library.

While monitoring traffic, if occurrences are detected of an unmatched abnormal CAN message sequence, the IDS can alert the relevant systems.

Figure 14 illustrates sequence changes when an attack occurs, in which, A, B, C and R represent CAN IDs. In normal conditions, A, B, and C messages are delivered in order (formed with a sequence of A-B-A-C) to actuate a specific function. Thus, A-B-A-C is a normally found CAN ID sequence pattern. If an attacker attempts to send R message without considering the adequate sequence, then the observed CAN ID sequence of R-B-C-A can be regarded as an abnormal sequence. By using these characteristics, an IDS can detect incoming attacks.

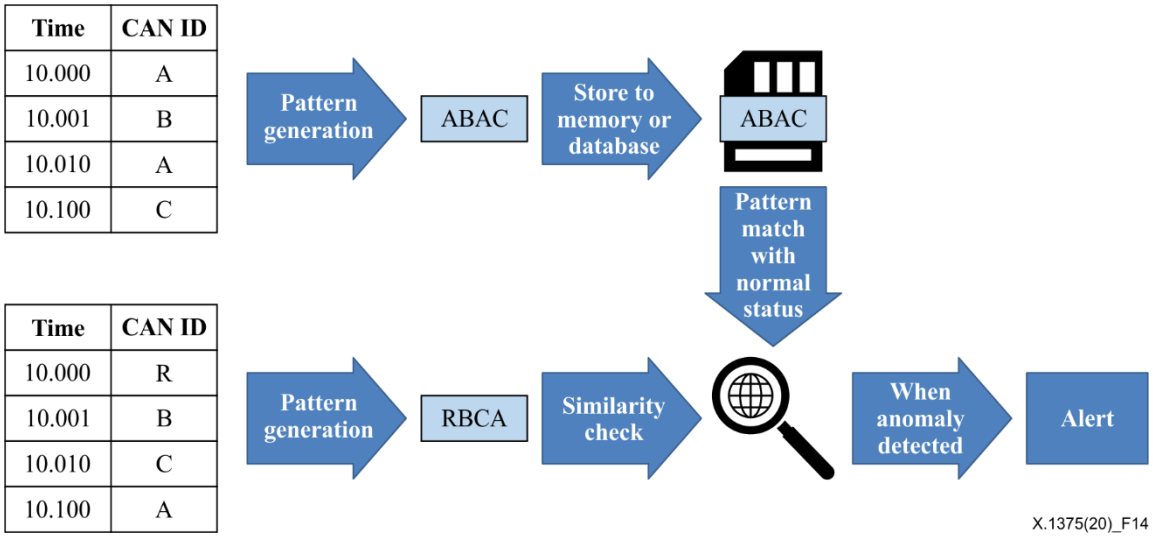

Figure 14 – Sequence-based anomaly detection

8.2.3.6 Offset-based model and plugin

CAN messages have a unique and regular interval time and offset when a vehicle is in normal (i.e., attack-free) status. This interval means the absolute elapsed time between the previous CAN message and the next message, while the offset means the number of packet or frame from the last request. In general, offset is within 5 when sending a remote frame request.

In addition to interval-based models being able to detect attacks, if there are injection attacks, then the arrival time interval for each CAN ID changes, as well as the offset value. This offset value variation can be used as a means of detection.

For example, when receiving a remote frame request, the reply packets should be delivered within a specific number of offsets. In the CAN protocol, a remote frame is designed to investigate the presence of the requested data frame. Thus, to detect anomaly status, the IDS can send remote frame requests periodically to all ECUs in the bus, and then estimate the offset response pattern.

– **Normal status**. Figure 15 shows the interval between the first CAN message (in the red dashed box) and the next message (in the orange dashed box) is short and immediate enough in the case of ID 0153. Also, the offset between the first message and the next one is 1, showing thats this CAN bus traffic is working normally as designed.

Figure 15 – Controller area network messages when a vehicle is in normal status

– **Under attack status**. Figure 16 shows that the first CAN message with 'ID 01f1' (in the red dashed box) hss arrived, but the response CAN message has not arrived within a short time and offsets; although many other CAN messages have also appeared in the time window. This shows that CAN bus traffic is under attack due to flooded messages.

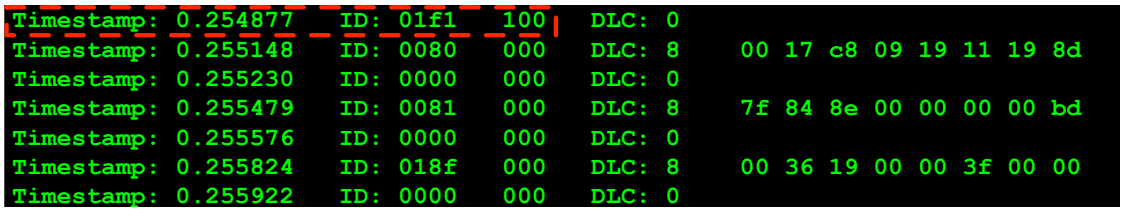

Figure 16 – Controller area network messages when a vehicle is in under attack status

Figure 17 shows changes in offset values of some CAN IDs (0x153, 0x164, 0x1F1, …, 0x5A0) when DoS and fuzzy attacks have occurred. The key on the right of Figure 17 indicates the offset value increase due to the attacks.

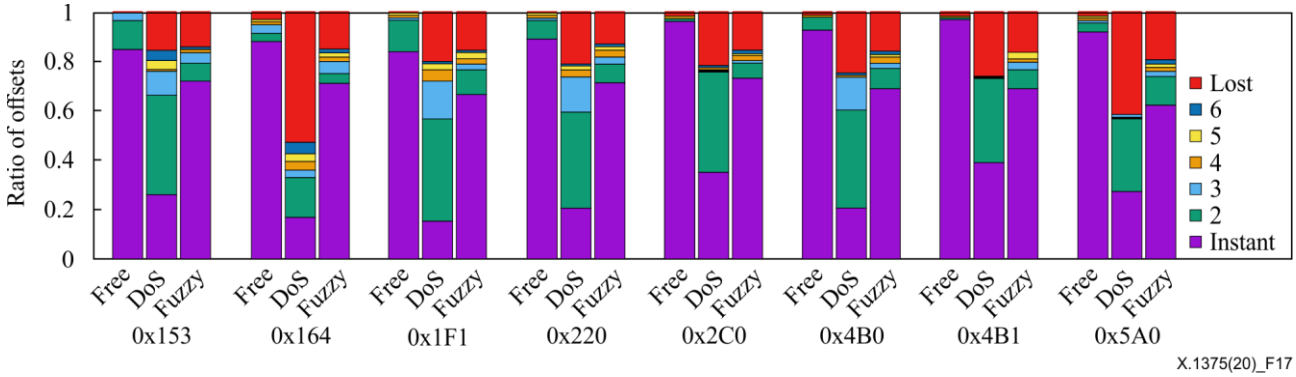

Figure 17 – Offset-based anomaly detection

For the groups of three bars for each CAN ID, the one on the left shows the ratio of offsets. For example, in the case of the 0x153 CAN ID, most messages are delivered without delay (in purple) when in an attack-free state. However, when DoS and fuzzy attacks occur, the offset values drastically increase. In some cases, many messages are lost, which has an adverse effect on security and safety of vehicles.

8.2.3.7 Machine learning-assisted intrusion detection model and plugin

If computing power allows, machine learning-based detection models (i.e., anomaly detection based on a convolutional neural network (CNN), deep neural network (DNN) and generative adversarial network (GAN)) can be considered.

– **GAN-based anomaly detection**. Many machine learning model-based approaches require real attack data or anomalous event data to construct and train an intrusion detection model. While normal data can be freely obtained, abnormal data is hard to obtain, because hackers usually do not release critical attack data and exploitative programs to the public. Also, vehicle manufacturers are reluctant to share case data about attacks they have experienced. This is the main limitation to the development of machine learning-based automobile IDS. To overcome this limitation, a GAN can play a role in detecting attacks not yet known to the

public. GAN is one of the deep learning models that can detect anomalies without knowledge of attacks.

GAN can generate anomaly data that are out of the normal range. These anomalous events contain unknown attacks. In other words, a model can be constructed using only normal data that can be easily obtained. GAN is composed of two sub-neural networks, a generator and a discriminator. The generator aims to generate fake data that are similar to actual CAN data. The discriminator aims to determine whether the input data is actual CAN data or fake data created by the generator.

CAN data can be represented in vector form (called one-hot vector encoding) as shown in Figure 18. CAN traffic data for GAN training and test can be treated as an image for efficient processing.

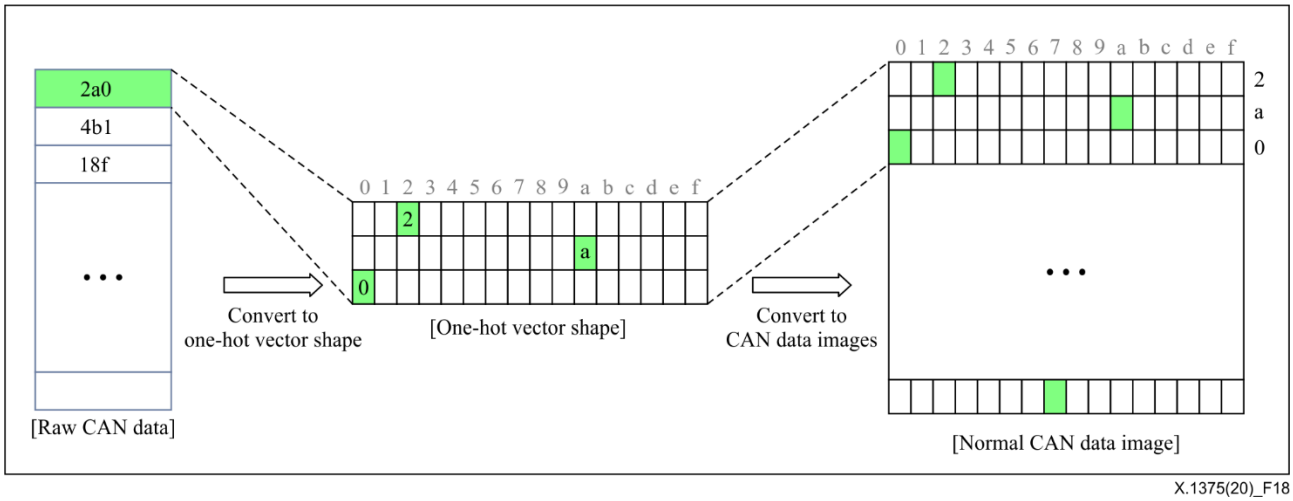

Figure 18 – One-hot vector encoding of a controller area network message

Figure 19 shows the overall architecture and process for a GAN training and test model. The generator produces fake images (i.e., abnormal CAN messages) by using random data as noise. The discriminator receives fake images produced by the generator and real images (i.e., normal CAN messages) of the actual CAN data. The generator competes with the discriminator and tries to make the discriminator identify which fake images are similar to real ones. Likewise, the discriminator competes with the generator. The discriminator tries to identify fake images similar to real images as fake ones. After training, the discriminator in the GAN model is used to detect anomalies.

The generator and the discriminator calculate the cost through back-propagation, reducing the errors between actual answers and outputs of the model.

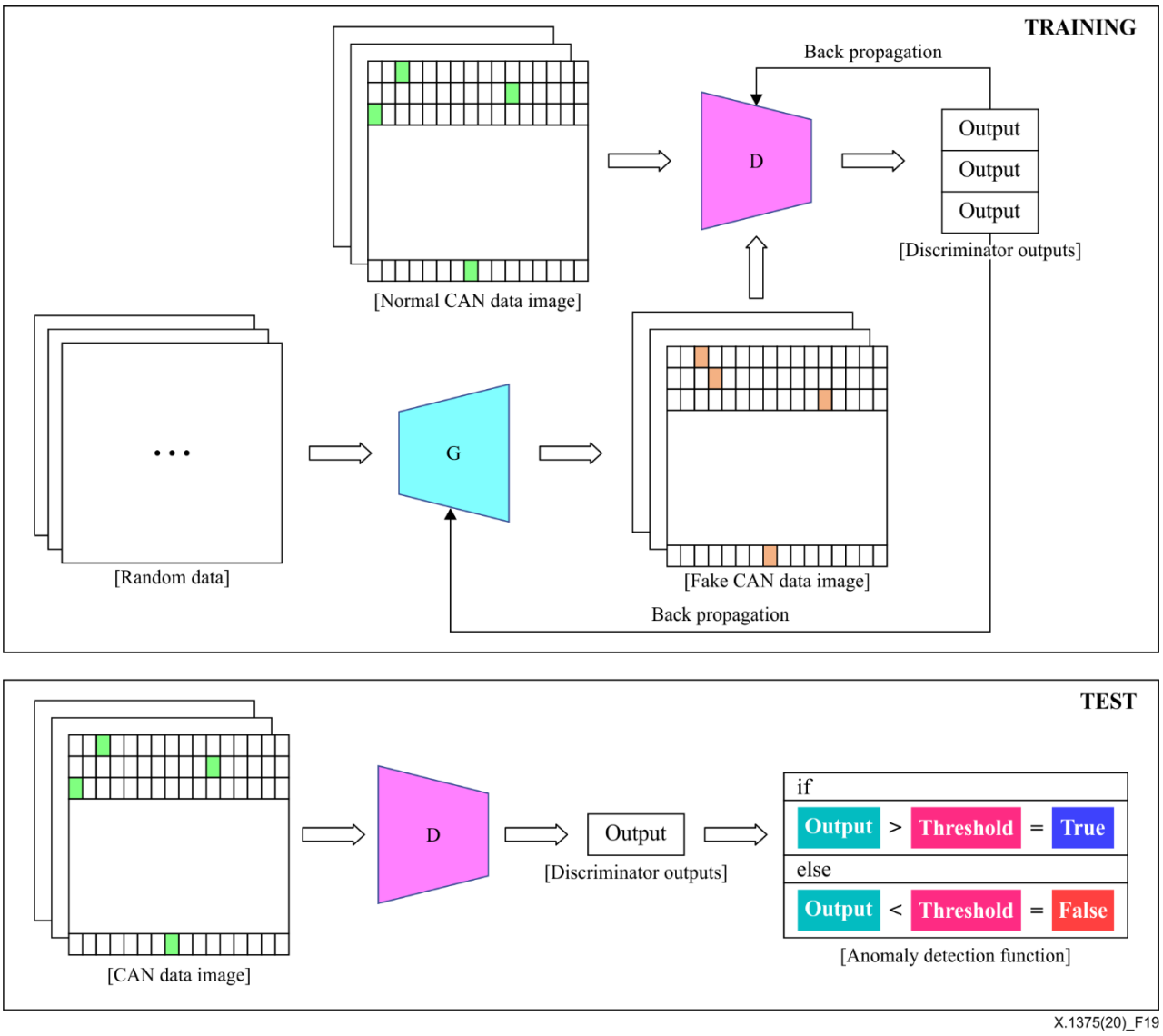

Figure 19 – GAN architecture and process for intrusion detection system
(G: generator; D: discriminator)

8.2.4 Hybrid detection with other intrusion detection systems located in edge, fog or cloud side

An in-vehicle IDS can immediately detect known attack patterns and respond to these attacks without consulting other IDSs located outside the vehicle. However, if an IDS detects unknown attacks, or suspicious or anomalous behaviours, the in-vehicle IDS can collaborate with ones outside to minimize the possibility of false-positive errors.

This type of hybrid detection model is effective in collecting suspicious unknown behaviours and developing new detection patterns.

The detected events can be transferred to other detection systems located in the edge, fog or cloud via available vehicular communications systems.

8.3 Detection rule set

Detection rule sets are classified into those based on: static detection; misuse detection; and anomaly detection. The main purpose of using detection rules is to identify any malicious message, i.e., any attacker injected or altered CAN message, from legitimate CAN message traffic.

– Rule sets for static detection are used to detect messages that violate predefined message field values.
– Rule sets for misuse detection are used to detect known patterns of malicious messages;.
– Rule sets for anomaly detection are used to detect abnormal-patterns of messages.

Note that the rule sets should be matched with the relevant detection method described in clause 8.2. For example, if an IDS detects misuse, then naturally the misuse rule set should also be prepared in the IDS. In addition, the information (or parameter value) used in each detection method should be included in the rule set structure.

Each rule set is further described in clauses 8.3.1 to 8.3.3.

8.3.1 Static detection rule set

The static detection rule set is used to detect any unusual behaviour within the IVN by matching messages against defined patterns. Examples include the following rules:

– CAN ID violation rule, which is used to detect messages having a specific CAN ID that is not included in a CAN database;
– CAN DLC violation rule, which is used to detect messages having a specific DLC value that is not included in a CAN database;
– CAN signal value violation rule, which is used to detect messages having a specific signal value that is not included in a CAN database;
– CAN bus domain violation rule, which is used to detect any unregistered message within a specific CAN bus domain.

8.3.2 Misuse detection rule set

The misuse detection rule set is used to detect any malicious messages within the IVN having specific patterns specified in advance.

– CAN message pattern filtering rule, which is used to detect any patterns that are already known as an attack in a message.

8.3.3 Anomaly detection rule set

The anomaly detection rule set is used to detect attacks that lie outside the scope of normal behaviours specified in advance.

– CAN message rate violation rule, which is used to detect any behaviour having an anomalous transmission period that exceeds normal behaviours specified in a CAN database;
– CAN signal data change rate violation rule, which is used to detect any behaviour having an anomalous signal data change rate that exceeds normal behaviours defined in a CAN database;
– CAN signal correlation violation rule, which is used to detect any behaviour in which two highly correlated signals suddenly become de-correlated in a specific period of time.

8.3.4 Detection rule set structure

Detection rule sets are formulated as shown in Table 2. The definitions of several terms used in Table 2 are as follows:

– signal ID, number specifying the signal type for a given CAN ID;
– signal value: a boundary value of a signal when a certain signal ID is given;
– bus ID, number specifying a CAN bus channel when a vehicle has multiple buses;
– correlated signal ID, specifying the other signal ID correlated with a given signal;
– signature pattern, strings in messages that can be used as signatures to identify attacks.

CAN ID, DLC, signal ID, signal value and bus ID are used for static detection, and messages are considered as anomalous if the values of the messages lie outside the ranges specified in Table 2.

Message rate, signal value change rate and correlated signal ID with coefficient value are used for anomaly detection, which is dependent on detection algorithms to specify or calculate these values. Signal correlation coefficients should be calculated in advance and those are compared with calculated values in real-time to detect anomalous behaviour.

Signature pattern values are specified and are used to detect already-known attack patterns in real-time.

Table 2 – Examples of rule set

CAN ID	DLC	Signal ID	Signal value	Bus ID	Message rate (ms)	Signal value change rate (%)	Correlated signal ID (coefficient value)	Signature pattern
0x01	0x08	0x10	0x00~0xFF	0x01	90-100	10-20	0x30	0x FF 0x00 0x01 0x86
0x02	0x08	0x20	0x00~0x80	0x02	90-100	10-20	–	0x75 05 A5 F5 A2

8.3.5 Detection rule set updates

In normal networks, when a security operations centre (SOC) finds that a certain detection rule causes a false positive or finds that there is a rule that can improve the true positive detection capability of an IDS, the IDS detection rule set can be updated according to an SOC request. Therefore, an IDS located within an IVN should also have the same functionality of allowing updates to its detection rule set(s) periodically or dynamically. The detection rule set can be updated via OTA or a communication channel between the IDS and SOC.

8.4 Detection report

Detection reports should be delivered to the SOC, the infrastructure that manages and analyses detection reports from IDSs in each vehicle for incident responses. Generally, the SOC manages security information and event management (SIEM) systems using security experts to aggregate and correlate security events generated by IDSs. Therefore, the SOC should filter these reports and deliver to each stakeholder, such as OEMs, drivers (users) and third parties, the appropriate information according to their data access authority. Third parties may include an insurance company or driving-data analytic service providers that have gained the rightful authority to access reports from the SOC. The delivery of any detection reports should be conducted through secure channels to preserve the confidentiality and integrity of reports.

Figure 20 shows how the detection report is delivered from in-vehicle IDS to SOC and to each stakeholder.

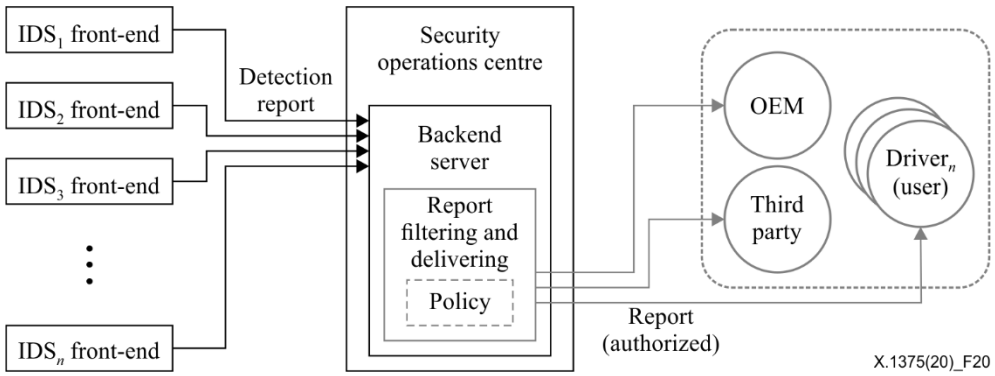

Figure 20 – Detection report flow from intrusion detection system to stakeholders

8.4.1 Detection report information

A detection report should include the following information:

– tag (message ID), the identification number of the delivered message in the detection report;

– LEN, the length of the message;

– vehicle identification number (VIN);

– vehicle status with recording time (e.g., vehicle speed provided by transmission control unit, engine coolant temp provided by engine control unit and steering wheel angle provided by body control unit);

– event time stamp (event date, event time);

– rule ID, which can work as an ID of the event detected;

– CAN ID, which is retrieved from message;

– message contents.

Table 3 shows an example of data list of detection report.

Table 3 – Example of data list of detection report

No.	Data category	Data	Description
1	VIN	Serial number specified by OEM	xx
2	Vehicle status	Vehicle speed	xx (km/h)
3	Vehicle status	Engine coolant temp	xx (°C)
4	Vehicle status	Steering wheel angle	xx°
5	Event date	Year:month:day	YY:MM:DD
6	Event time	Hour:minute second.millisecond	h:min:s.ms

8.4.2 Detection report format

Figure 21 shows an example of detection report format. It can include detection report initially consists of tag, LEN, VIN, vehicle status, date, timestamp, ruleID, canID and message contents. The length field should be added to the detection report to parse each datum in the backend server of the SOC. After generating the initial report, the detection report, except for tag and LEN data fields

should be encrypted to maintain confidentiality of the report. In addition, the digital signature should be generated using the whole detection report, including tag and LEN data fields, and appended to the detection report.

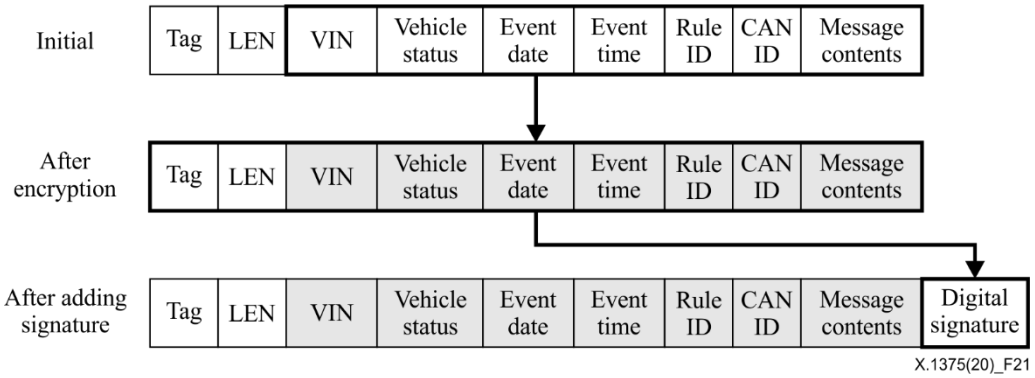

Figure 21 – Detection report format for the security operations centre process

8.4.3 Detection report delivery

The delivery of any detection reports should be conducted through secure channels to preserve the confidentiality and integrity of reports. For example, hypertext transfer protocol (HTTP) over transport layer security (TLS) or message-queuing telemetry transport (MQTT) over TLS should be used rather than native HTTP or MQTT alone.

In addition, it is necessary to determine the priority of the detection report. If there are multiple security events that IDS detects in a short period of time, IDS should determine which of the detection reports that contains a security event requires delivery to SOC first.

The priority of a detection report should be determined according to a severity score and confidence score. The severity score is a value that represents how the security event that violates a certain detection rule is dangerous or critical. The severity score of an individual detection rule should be specified in advance by security experts.

The confidence score is a value that represents the possibility of the security event detected by IDS being a real attack. The confidence score should be calculated dynamically whenever the violation is detected by IDS, and the degree of difference with the normal characteristic of IVN should be reflected in the confidence score. For example, if there are two suspicious messages that violate the normal message transmission period and the differences in the transmission period are10 ms and 20 ms, respectively, the confidence score of the message that has the greater difference should be larger than that with the smaller. The final priority of detection report can simply be calculated by the security score and confidence score.

Appendix I

General architecture of and background information about in-vehicle networks

(This appendix does not form an integral part of this Recommendation.)

I.1 General architectural framework of an intrusion detection system

This appendix describes the general architectural framework of an IDS. Figure I.1 depicts the general architecture of an IDS, consisting of: a collector; detector; updater;, responder; crypto engine; and storage, which modules perform the following functions.

– A collector provides an interface with input data for data access by other IDS modules such as a detector.

– A detector carries out a detection process. It accesses data provided by the collector and storage, and determines the resulting response.

– A responder reacts to detected intrusions with actions, such as warning the owner of the intrusion occurrence and reporting the results to the IDS control centre.

– An updater updates a detector based on the latest detection rule set.

– A crypto engine carries out cryptographic operations for various types of data to keep them secure from any unauthorized access.

– Storage provides a permanent depot for data required by the detector or the updater and control of the database system.

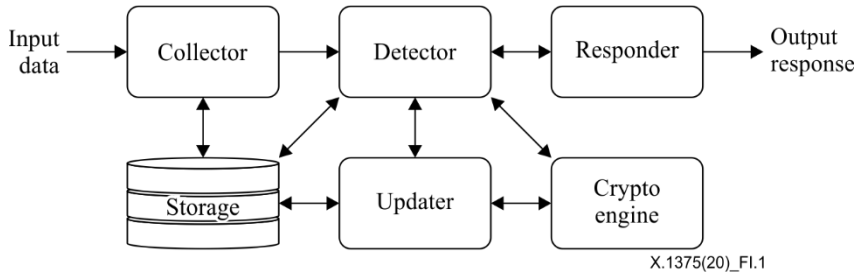

Figure I.1 – General architecture of intrusion detection system

I.2 In-vehicle network and its characteristics

Nowadays, communication technology plays an important role in the vehicle. Vehicular communication can be classified into communication external and internal to the vehicle as shown in Figure I.2. The IVN involves vehicle components such as sensors and ECUs, which are used in several domains, such as chassis control, body control and powertrain control of the vehicle. Moreover, these components are used in an advanced driver assistance system (ADAS), which supports a driver while driving. such as maintaining a vehicle within a driving lane and cruise control functionality. The head unit is a component of automotive infotainment, which gives the user control over vehicle information and entertainment media, e.g., audio and video.

This Recommendation mainly focuses on internal communications in IVNs such as CAN or CAN FD that cannot be supported by general IDSs as currently used on the Internet. The IDS aspects of this Recommendation are meant to ensure the detection of threat that impact ECU communications by using various efficient lightweight detection models such as those based on signature, entropy, self-similarity or survival. It includes the classification and understanding of threats on the internal communication network such as a CAN in vehicles that works with specialized protocols (e.g., bus type IVNs).

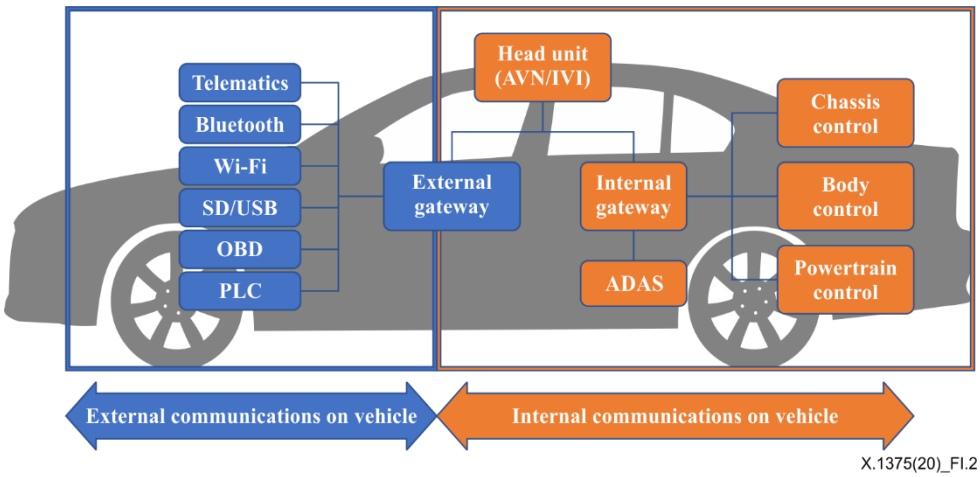

Figure I.2 – Communications environments in a vehicle

AVN: audio-video navigation; IVI: in-vehicle infotainment; PLC: power line communication; SD: secure digital

I.2.1 Overview of controller area networks

Among many IVNs such as CAN, CAN FD, LIN, MOST, FlexRay and automotive Ethernet, CAN is the most representative and widely used network type.

CAN is a message-based protocol. All messages are broadcast to existing nodes on the CAN bus and the reception filter of each node makes the selection using the arbitration ID of the message. Each node can extract and use the data after determining whether a message is accepted. Thus, the arbitration ID is not used as a concept of address, but acts as a priority when a collision occurs. The CAN protocol adopts carrier sense multiple access with collision avoidance (CSMA/CA) schemes and arbitration on message priority (AMP) to share the same bus with multiple nodes. When some nodes attempt to transmit at the same time, these messages should compete to occupy the bus. In this case, the arbitration process between messages starts according to a bit level of the ID. The competition procedure is done by sequentially comparing an arbitration field from first to end bit. It shows a dominant bit when a value is 0, because two bits operate on an AND-gate. The result is that a lower value between two IDs has a higher priority.

Figure I.3 shows the structure of a CAN frame. The description of each field in CAN frame is as follows.

– Start of frame (SOF) is a start bit composed of a single dominant bit and informs all nodes of a start of transmission. It can start if the bus is in a rest state.

– Arbitration field consists of 11 bits as an ID and 1 bit as remote transmission request (RTR). The ID is used as a priority during the arbitration process, and the RTR is determined according to the kind of CAN frame.

– Control field indicates two reserved bits and four bits as DLC.

– Data field refers to actual data for transferring information to another node. Any value from 0 to a maximum of 8 bytes can be made.

– Cyclic redundancy code (CRC) field guarantees the validity of a message as CRC; all nodes are subjected to the verification process of a message generated by a sender using this field.

– Acknowledge field consists of two bits as ACK part and ACK delimiter part. If a node receives a valid message normally, it replaces the ACK part, which was a recessive bit, with a dominant bit.

– End of frame – a CAN frame is terminated by a flag consisting of seven recessive bits.

– IFS – Data frames and remote frames are separated from preceding frames by a bit field called an IFS, which consists of at least three consecutive recessive bits.

– Data frame among CAN frames is the only frame for data transmission. Thus, nodes on the bus interact with others by transmitting data frames. If the RTR bit of an arbitration field is a dominant bit, it becomes a data frame.

– Remote frame – all nodes periodically broadcast messages regardless of receiver status. However, all receiver nodes can require certain kinds of information to run a given task. The remote frame arbitration field is composed of an ID and RTR bit, which is recessive. The other fields of a remote frame are the same as a data frame, but it is not intended to transmit information so that the data field can be empty.

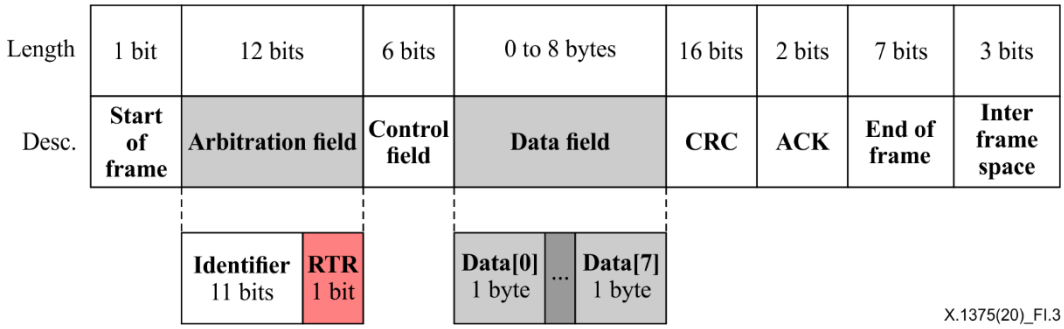

Figure I.3 – Structure of a controller area network frame

Regarding the attack sources in IVNs, the following remarks apply.

– Any connected device internally or externally can be an attack source of the IVN.

– Although there is no directly linked line, attacks from inside or outside can propagate to all connected networks when access control fails.

– Moreover, a traditional IVN does not have strong security features, e.g., device authentication or traffic encryption. That means any compromised node can send attack traffic without any restriction. Also, a bus type of IVN is a broadcasting type of network. Thus, all nodes connected to the same bus can listen to all traffic; and all nodes can be affected by any malicious broadcast messages because the broadcasted traffic in the same network segment cannot be controlled by a network switch or security gateway.

Therefore, there can be two severe problems from the network security perspective, as follows.

1) First, intrusion can be detected, but not prevented in real time. Some extreme measure (e.g., bus-off way) can block the attack source, but it can cause a serious adverse effect on vehicle functional safety.

2) Second, the detection should be done in a very short time. Some attacks can be successful by only sending a small number of packets. That is, the detection algorithms should be lightweight.

I.2.2 Content characteristics

A CAN ID itself is not an ID of an ECU (sender or receiver); thus, any ECU can send multiple CAN IDs regardless of ECU functionality. As shown in Figure I.4, basically, there are repetitive CAN messages from senders, but the ECU does not only send fixed messages. ECUs can transmit multiple types of message to be operated. For example, the node shown in Figure I.4 sends 0x1, 0x2 and 0x3 messages at the beginning, then it sends an 0x1 message only at 20 ms. Likewise, this node sends 0x1 and 0x3 messages at 100 ms. Also, such messages (e.g., 0x1, 0x2, 0x3) can be sent by another ECU. Thus, even though an IDS can detect a suspicious message whose message ID is 0x2, it cannot be used to point out the original sender.

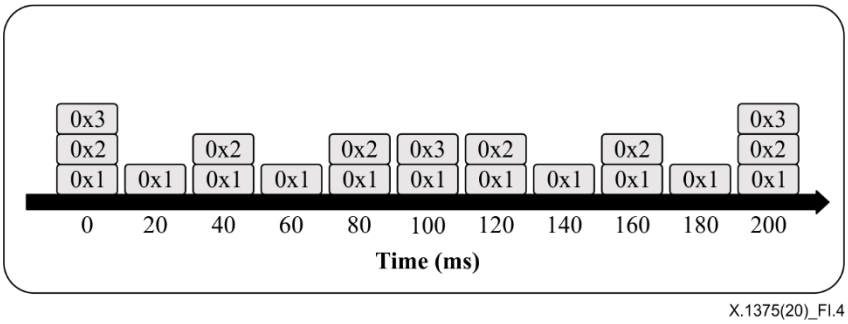

Figure I.4 – Transmission of controller area network messages

I.2.3 Technical characteristics

Because of the characteristics of the bus type of network, there is no specific packet header to identify the source (sender) and the destination (receiver) node.

Therefore, in the same bus, hardware signal analysis for each connected node is the only way to identify the attack source accurately. However, it requires additional installation of hardware for signal analysis; thus, this method is not always feasible for all vehicles and will increase the overall cost of vehicle production. A software-based approach can identify the attack source, but it is very hard to implement reliably. Both hardware- or software-based approaches can make false-positive errors.

As can be seen in Figure I.2 and Figure I.5, there are various possible attack sources (e.g., Bluetooth, wireless fidelity (Wi-Fi), OBD and telematics devices). Although they are not directly inserted in the internal network, an IDS needs to understand its traffic input patterns to classify regular traffic input and abnormal traffic input. (The possible various attack sources are depicted in Figure I.5.)

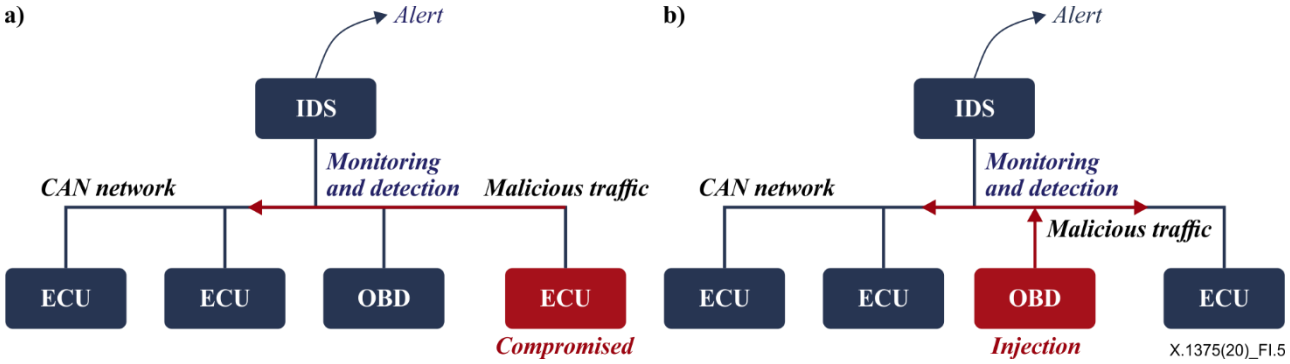

Figure I.5 – Controller area network message injection by various attack sources

I.2.4 Behaviour characteristics

An IVN is a closed network. That means there will be no dramatic traffic pattern change unless the vehicle is under attack, malfunctioning or adding many new nodes. Therefore, basically, IVNs have a self-similarity property (i.e., the traffic behaviour pattern is very similar to that in the previous time-window). This self-similarity property can be measured by using generic network entities (e.g., the number of packets, frequency of incoming packets and interval of arrival for each packet in a specific time-window). If an attack occurs in the network, the number of CAN packets, the frequency of CAN messages and the interval of CAN packets would be changed. Also, the response packets from ECU behavioural patterns will also be changed along with incoming malicious packets.

Therefore, monitoring network specific features from traffic is required to detect various attacks, whether or not some attack patterns are already known to the public.

I.2.5 Specialized characteristics

– Detection location

If ECUs can install an IDS in their operating system, they can increase the security level. However, installation requires more computing power, and because of the cost increase, this approach is not preferred by OEMs.

Therefore, the most flexible and preferred detection method is network-based intrusion detection.

– Blacklist- or whitelist-based traffic control derived from manufacturer ground-truth data

To reduce the detection cost by network-based intrusion detection, if the vehicle supports the OTA method to update ECU firmware, then it will be better to update ECUs to send allowed CAN messages only (blacklist-based traffic control); or update the ECUs to listen to the allowed CAN messages only (whitelist-based traffic control).

This treatment is helpful to reduce unneeded traffic in IVNs; however, it still requires critical ECU updates. Thus, it is not always preferred by OEMs.

Bibliography

[b-ITU-T X.800] Recommendation ITU-T X.800 (1991), *Security architecture for Open Systems Interconnection for CCITT applications*.

[b-ISO/IEC 27000] ISO/IEC 27000:2018, *Information technology – Security techniques – Information security management systems – Overview and vocabulary*.

[b-ISO/IEC 27039] ISO/IEC 27039:2015, *Information technology – Security techniques – Selection, deployment and operations of intrusion detection and prevention systems (IDPS)*.

[b-ISO/IEC/IEEE 24765] ISO/IEC/IEEE 24765:2017, *Systems and software engineering – Vocabulary*.

[b-Kettani] Kettani, H., Gubner, J. A. (2006). A novel approach to the estimation of the long-range dependence parameter. *IEEE Trans. Circuits Systems II: Express Briefs*, **53**(6), pp. 463-467.

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	Tariff and accounting principles and international telecommunication/ICT economic and policy issues
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Environment and ICTs, climate change, e-waste, energy efficiency; construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling, and associated measurements and tests
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	**Data networks, open system communications and security**
Series Y	Global information infrastructure, Internet protocol aspects, next-generation networks, Internet of Things and smart cities
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2021