基于CANBus的防拆与远程锁车方案

1.概述

锁车解决方案通过整合云平台、车载网关和控制器， 为用户提供一体化的远程车辆锁定/解锁方案。授权用户通过车载平台可以实时下发锁车/解锁命令来远程控制车辆锁定和解锁，保证车辆资产得到授权使用。

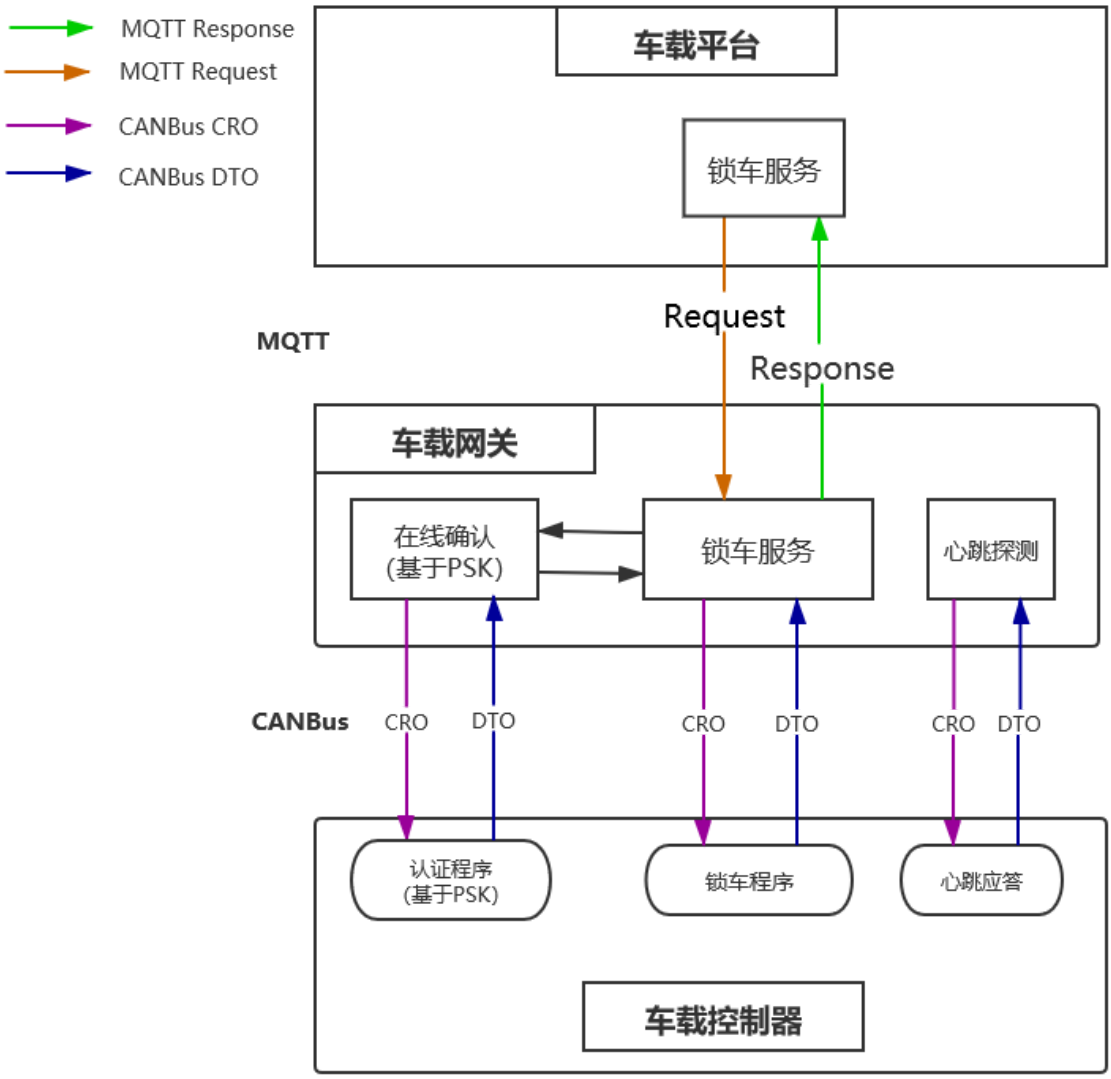
网关与车载控制器通过CANBus连接，网关与平台通过MQTT连接。

为确保锁车/解锁命令来自授权请求， 网关和控制器之间采用共享密钥进行认证。

在网关启动后或者CANBus重连后， 网关会执行在线确认过程， 确保控制器在线。

确认控制器在线后， 网关会周期性发送心跳指令， 用于判断网关与控制器之间的连接处于正常状态，在状态发生变化时将采取相关措施。 在连接正常情况下， 网关还会根据控制器收包统计判读是否有在线破解或干扰攻击。

由于锁车/解锁操作关系到安全， 执行相关指令前必须要重新进行认证， 确保是合法的请求。



2. 网关与控制器之间的交互协议

2.1 协议描述

2.1.1 默认的CAN参数

- 比特率(bitrate)：250kbps
- 网关CAN Id(local_can_id): 0x70
- 车载控制器CAN Id(target_can_id): 0x18A
- 如果需要更改参见3.1 CANBus参数配置

2.1.2 协议

网关和控制器之间采用请求/响应模式进行通信，由网关发送请求，控制器收到请求后发送响应。网关发送Command Receive Object(CRO)数据包，控制器在收到CRO后，响应Data Transmission Object(DTO)数据包。不同的服务(Action Service)是通过CRO数据包中的Action Service Number字段指定的。本协议约定Command code 都使用0x21。

2.1.2.1 Command Receive Object(CRO)

Parameters in message field:

Position	Type	Description
0	byte	Command code = 0x21(ACTION_SERVICE)
1	byte	Command counter = CTR
2	byte	Action Service Number
3...7	bytes	Action service related parameter and data area

Note: The data length code of the CRO must always be 8. Unused data bytes, marked as "don't care" in the command descriptions, may have arbitrary values.

2.1.2.2 Data Transmission Object(DTO)

Parameters in message field:

Position	Type	Description
0	byte	Packet ID = 0xFF(DTO contains a Command Return Message)
1	byte	Command Return Code
2	byte	CTR: Command Counter as received in CRO with the last command.
3...7	byte	Return data field

2.1.2.3 Command Return Codes

Table of Command Return Codes:

Command Return Code	Description
0x00	ACK(no error)
0x01	NACK(error)

2.1.2.4 Supported Action Services

Action Service	Action Service Number	Description	Note
get_seed	0x30	获取随机数	
online_confirm	0x31	在线确认	
heartbeat	0x32	心跳包	
lock	0x33	锁车	
unlock	0x34	解锁	
perm_lock	0x35	永久锁车	
perm_unlock	0x36	永久解锁	

2.2 在线确认

2.2.1 应用流程

采用预共享密钥，随机数和hash算法实现。

步骤1：预先在网关和控制器上配置好密钥【PSK】。

步骤2：在网关启动或CANBus重连后，发送**获取随机数指令**。控制器生成随机数并保存， 同时将生成的随机数返回给网关。

步骤3：网关将收到的随机数和密钥进行哈希运算，生成hash值A。【采用控制器和网关都支持的hash算法】

步骤4：网关将hash值A放入**认证指令**，发送至控制器。

步骤5：控制器在收到**认证指令**后，取出hash值A。

步骤6：控制器通过随机数和密钥计算出的hash值B。

步骤7：控制器比较hash值A与B，相同则通过认证，不同则表示认证失败或被拆卸。

步骤8：控制器将比较结果通过**认证指令**的应答数据包反馈给网关。ACK表示认证通过，NACK即表示认证失败。

2.2.2 获取随机数

- Structure of data in CRO:

Position	Type	Description
0	byte	Command code = 0x21(ACTION_SERVICE)
1	byte	Command counter = CTR
2	word	Action Service Number = 0x30
3...7	bytes	don't care

- Contents of returned DTO:

Position	Type	Description
0	byte	Packet ID: 0xFF(DTO contains a Command Return Message)
1	byte	Command Return Code
2	byte	Command Counter = CTR
3...6	byte	random number
7	byte	don't care

- Example:

网关发送:

CAN Id: 0x18A

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0x21	0x01	0x30	0x00	0x00	0x00	0x00	0x00

控制器发送:

CAN Id: 0x70

ACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x00	0x01	0xCC	0x23	0x21	0x2C	0x00

Note: The random number value is :0xCC 0x23 0x21 0x2C

NACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x01	0x01	0x00	0x00	0x00	0x00	0x00

2.2.3 认证

- Structure of data in CRO:

Position	Type	Description
0	byte	Command code = 0x21
1	byte	Command counter = CTR
2	byte	Action Service Number = 0x31
3...6	bytes	"hash" data
7	byte	don't care

- Contents of returned DTO:

Position	Type	Description
0	byte	Packet ID: 0xFF
1	byte	Command Return Code
2	byte	Command Counter = CTR
3...7	byte	don't care

- Example:

网关发送:

CAN Id: 0x18A

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0x21	0x01	0x31	0x32	0x3A	0x43	0xCD	0x00

Note: The hash value is :0x32 0x3A 0x43 0xCD

控制器发送:

CAN Id: 0x70

ACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x00	0x01	0x00	0x00	0x00	0x00	0x00

NACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x01	0x01	0x00	0x00	0x00	0x00	0x00

2.3 连接监控

2.3.1 心跳

确认控制器在线后， 网关会周期性发送心跳指令， 用于判断网关与控制器之间的连接是否处于正常状态， 在状态发生变化时将采取相关措施，本地输出告警信息，并将告警信息发送至平台。

- Structure of data in CRO:

Position	Type	Description
0	byte	Command code = 0x21(ACTION_SERVICE)
1	byte	Command counter = CTR
2	byte	Action Service Number = 0x32
3...6	bytes	don't care
7	byte	Bit 0：平台连接状态（0：未连接， 1：已连接） Bit 1: clear counter（0：do nothing， 1：clear) 控制器在收到clear counter后需要将Received packets counter和Error packets counter清零

- Contents of returned DTO:

Position	Type	Description
0	byte	Packet ID = 0xFF(DTO contains a Command Return Message)
1	byte	Command Return Code
2	byte	Command Counter = CTR
3...4	word	Received packets counter. 控制器收到的所有控制指令计数
5...6	word	Error packets counter. 控制器收到的验证失败的数据包计数
7	byte	Lock status. 锁车状态由控制器返回 Bits[2:0]: 锁车状态(0x00: 表示未锁车，0x01: 表示已锁车，0x02: 表示已永久锁车)

- Example:

网关发送:

CAN Id: 0x18A

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0x21	0x01	0x32	0x00	0x00	0x00	0x00	0x00

控制器发送:

CAN Id: 0x70

ACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x00	0x01	0x01	0x00	0x00	0x00	0x00

2.4 远程锁车/解锁

2.4.1 应用流程

网关采用心跳包，判断与控制器之间的连接状态。网关接收平台下发的锁车/解锁指令，并下发给控制器。

步骤1：网关在启动后，周期性发送心跳包。

步骤2：控制器在收到心跳包后，发送心跳应答包。

步骤3：网关通过心跳应答包，判断网关和控制器之间连接是否正常。

步骤4：网关在一段时间未收到心跳应答包，则心跳超时。网关输出本地告警并发送告警信息至平台，告知平台网关与控制器连接异常（在这种情况下，网关是不接受锁车/解锁指令，还是需要记录指令，在恢复连接后执行相应的指令？）。反之，网关与控制器连接正常，网关可正常接收平台发送的锁车/解锁指令，并下发指定的CAN数据报文至车载控制器。

步骤5：车载控制器收到锁车/解锁指令后，在满足预设的锁车条件下【锁车要求车辆未在行驶状态且未在作业中，以免产生安全隐患】，执行锁车/解锁指令。

2.4.2 锁车

步骤1：获取随机数

网关发送:

CAN Id: 0x18A

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0x21	0x01	0x30	0x00	0x00	0x00	0x00	0x00

控制器发送:

CAN Id: 0x70

ACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x00	0x01	0x19	0x23	0x21	0x2C	0x00

Note: The random number value is :0x19 0x23 0x21 0x2C

NACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x01	0x01	0x00	0x00	0x00	0x00	0x00

步骤2：下发锁车指令

网关发送:

CAN Id: 0x18A

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0x21	0x01	0x33	0x32	0x56	0x43	0xCD	0x00

Note: The hash value is :0x32 0x56 0x43 0xCD

控制器发送:

CAN Id: 0x70

ACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x00	0x01	0x00	0x00	0x00	0x00	0x00

NACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x01	0x01	0x00	0x00	0x00	0x00	0x00

2.4.3 解锁

步骤1：获取随机数

网关发送:

CAN Id: 0x18A

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0x21	0x01	0x30	0x00	0x00	0x00	0x00	0x00

控制器发送:

CAN Id: 0x70

ACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x00	0x01	0x22	0x23	0x21	0x2C	0x00

Note: The random number value is :0x22 0x23 0x21 0x2C

NACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x01	0x01	0x00	0x00	0x00	0x00	0x00

步骤2：下发解锁指令

网关发送:

CAN Id: 0x18A

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0x21	0x01	0x34	0x79	0x56	0x43	0xCD	0x00

Note: The hash value is :0x79 0x56 0x43 0xCD

控制器发送:

CAN Id: 0x70

ACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x00	0x01	0x00	0x00	0x00	0x00	0x00

NACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x01	0x01	0x00	0x00	0x00	0x00	0x00

2.4.4 永久锁车

步骤1：获取随机数

网关发送:

CAN Id: 0x18A

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0x21	0x01	0x30	0x00	0x00	0x00	0x00	0x00

控制器发送:

CAN Id: 0x70

ACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x00	0x01	0x22	0x23	0x21	0xBB	0x00

Note: The random number value is :0x22 0x23 0x21 0xBB

NACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x01	0x01	0x00	0x00	0x00	0x00	0x00

步骤2：下发解锁指令

网关发送:

CAN Id: 0x18A

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0x21	0x01	0x35	0x79	0x56	0x43	0x62	0x00

Note: The hash value is :0x79 0x56 0x43 0x62

控制器发送:

CAN Id: 0x70

ACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x00	0x01	0x00	0x00	0x00	0x00	0x00

NACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x01	0x01	0x00	0x00	0x00	0x00	0x00

2.4.5 永久解锁

步骤1：获取随机数

网关发送:

CAN Id: 0x18A

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0x21	0x01	0x30	0x00	0x00	0x00	0x00	0x00

控制器发送:

CAN Id: 0x70

ACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x00	0x01	0x22	0x23	0x21	0x45	0x00

Note: The random number value is :0x22 0x23 0x21 0x45

NACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x01	0x01	0x00	0x00	0x00	0x00	0x00

步骤2：下发解锁指令

网关发送:

CAN Id: 0x18A

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0x21	0x01	0x36	0x79	0x56	0x43	0x6C	0x00

Note: The hash value is :0x79 0x56 0x43 0x6C

控制器发送:

CAN Id: 0x70

ACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x00	0x01	0x00	0x00	0x00	0x00	0x00

NACK:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
0xFF	0x01	0x01	0x00	0x00	0x00	0x00	0x00

3. 平台与网关之间的交互协议

- 支持的CANBus参数配置表

Parameter Name	Description	Type	Note
client_token	确保请求/应答相对应的唯一标识符	string	
bitrate	CANbus波特率	int	
local_can_id	网关CAN id	int	
target_can_id	车载控制器CAN id	int	
heartbeat_interval	心跳间隔(单位：s)	int	

- 支持的指令控制表

Parameter Name	Description	Type	Note
client_token	确保请求/应答相对应的唯一标识符	string	
command	CAN控制指令，包括： 锁车: lock 解锁: unlock 永久锁车: perm_lock 永久解锁: perm_unlock	string	

3.1 CANBus参数配置

向下面topic发送消息，配置CANBus的通信参数。

Request Topic：`/v1/{client_id}/can/set`

Request payload：

```
{
    "client_token": "3bzJQ2OOUkLS6O6lMhW3muUv73ycUT7J",
    “bitrate”: 250000,
    “local_can_id”: 0x70,
    “target_can_id”: 0x18A,
    "heartbeat_interval": 60
}
```

Response Topic：`/v1/{client_id}/can/set/resp`

Response Payload：

Success：

```
{
    "client_token": "3bzJQ2OOUkLS6O6lMhW3muUv73ycUT7J",
    "result": {
        “bitrate”: 250000,
        “local_can_id”: 0x70,
        “target_can_id”: 0x18A,
        "heartbeat_interval": 60
    }
}
```

Failure:

```
{
    "client_token": "3bzJQ2OOUkLS6O6lMhW3muUv73ycUT7J",
    "error": "invalid_parameter",
    "error_desc": "Invalid request parameter"
}
```

3.2 下发指令

向下面topic发送消息，下发指令。

Request Topic: `/v1/{client_id}/can/control`

Request payload:

```
{
    "client_token": "3bzJQ2OOUkLS6O6lMhW3muUv73ycUT7J",
    "command": "lock|unlock|perm_lock|perm_unlock"
}
```

Response Topic: `/v1/{client_id}/can/control/resp`

Response Payload:

Success:

```
{
    "client_token": "3bzJQ2OOUkLS6O6lMhW3muUv73ycUT7J",
    "result": {
        "command": "lock|unlock|perm_lock|perm_unlock"
    }
}
```

Failure:

```
{
    "client_token": "3bzJQ2OOUkLS6O6lMhW3muUv73ycUT7J",
    "error": "invalid_parameter",
    "error_desc": "Invalid request parameter"
}
```