Odin Device-Phone Communication Protocol

v1.11

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Description |
| 0.3 | 2013/03/12 | Rocky Huang |  |
| 0.4 | 2013/05/14 | Rocky Huang | Major Change:   1. Add firmware upgrade mechanism 2. Modify the mechanism for the administrator to read back IPA client    1. Section 3.3.3 IPA mode ReadAddedClient command removed    2. Section 3.5.3, MANAGEMENT mode Sync command    3. Section 3.5.4, MANAGEMENT mode ValidatePINAndSync command 3. Add command to let client change its user name   3.1 Section 3.5.5, MANAGEMENT mode SetUserName command |
| 0.5 | 2013/05/27 | Rocky Huang | Major Change:   1. Change on administrator’s event log record synchronization related command.    1. Section 3.5.1, MANAGEMENT mode SetPIN command    2. Section 3.5.2, MANAGEMENT mode SetGIN command    3. Section 3.5.3, MANAGEMENT mode Sync command    4. Section 3.5.4, MANAGEMENT mode ValidatePINAndSync command    5. Section 3.5.5, MANAGEMENT mode SetUserName command    6. Section 3.5.8, MANAGEMENT mode ValidatePINAndSetPIN command    7. Section 3.5.9, MANAGEMENT mode ValidatePINAndSetGIN command    8. Section 3.5.10, MANAGEMENT mode AckSync command    9. Section 3.1.1, PAIRING mode SendRequest command    10. Section 3.6, new ActCode value 06h ‘Sync only’ 2. Mechanism to let phone side be able to explicitly abort mutual authentication process.    1. Section 3.2.5, AUTH mode Abort command |
| 0.6 | 2013/7/29 | Rocky Huang | Major Change:   1. Add Access Right passing flow. The following sections are changed:    1. Section 3.3.1, IPA mode SendRequest command.    2. Section 3.4.3, PRC mode RegisterClient command    3. Section 3.5.3 and section 3.5.4, MANAGEMENT mode Sync and ValidatePINAndSync command 2. Add method to passing device name back to client in order to reflect the change of device name (lock name) by administrator    1. Section 3.2.1, AUTH mode SendRequest command 3. Add new log event definition in section 3.8 4. Device will response FW version in PAIRING procedure in section 3.1.1 5. Add method to let client change its user name directly in AUTH mode Finish command in section 3.2.2 6. Add new status code to notify client its re-add attempt in section 3.2.2 7. Add new status code to notify client/admin its unlock door attempt is bypassed due to handle incorrect position in section 3.2.2 and 3.5.4 8. Add method to let app challenge device for App\_key correctness during PAIRING and IPA process, in section 3.1.1 and section 3.3.1 9. Add method for dealing with the challenge-response process in mutual authentication in the case that app side has more than 10 DIDs, in section 3.2.1, section 3.2.6, and section 3.2.7. 10. Remove log reading function in MANAGEMENT mode SetUsername command, in section 3.5.5. 11. Add MANAGEMENT mode SetProperty command, in section 3.5.11. 12. Modify MANAGEMENT mode GetProperty command to support get multiple properties, in section 3.5.7. 13. Add method to pass Admin\_name and Device\_name to client by AUTH mode FINISH command, in section 3.2.2. 14. Add method to let Admin change device name, in section 3.5.12. 15. Add method to differentiate DIN check error and SETUP state error upon PAIRING. In section 3.1.1 16. Add description about NDEF media-type interpolation, in section 2.1 17. Add AUTH mode SendRequest\_Wo\_AutoPairing and SendRequest\_Wo\_AutoPairingEx command that will not trigger auto pairing in SETUP mode, in section 3.2.8 and section 3.2.9 18. Modify PAIRING mode SendRequest command to add response admin\_rolling\_number to tell who is the latest admin 19. Modify MANAGEMENT mode Sync command’s command format and IPA mode SendRequest command’s response format because access right definition changes. Modify PRC mode register client command to transmit two-pass encrypted access right. 20. Modify 3.3.1, access right to 16 bytes. 21. Modify 3.5.3, access right to 16 bytes. The response data contains access right. 22. Modify 3.2.2, client read back item 23. Modify 3.5.7.1, Property ID and data size |
| 0.7 | 2014/2/5 | Rocky Huang | 1. Remove original ValidatePIN in section 3.2.3 and ValidateGIN in section 3.2.4 2. Remove original SetPIN in section 3.5.1 and SetGIN in section 3.5.2 and ValidatePINAndSync in section 3.5.4 3. Add UpdateAccessRight in section 3.4.5 |
| 0.8 | 2014/7/25 | Rocky Huang | 1. Add FACTORY test command |
| 0.9 | 2014/10/2 | Rocky Huang | 1. In section 3.5.1, add 2nd entry-ctrl-byte 2. New 2nd entry-ctrl-byte table in section 3.8 |
| 1.0 | 2014/10/9 | Rocky Huang | 1. In section 2.1, the latest protocol version code 80h 05h. 2. In section 3.1.1, let admin get latest sequence from device when pairing 3. In section 3.2.2, let client get battery status. 4. In section 3.5.1, Modify response data format to convey the latest sequence number of a client recorded at device side 5. In section 3.4.3 and section 3.4.5, modify command data format to make sequence\_number encrypted by admin’s DID-FID-Key 6. In section 3.4.1, let RequestTID to be encrypted by admin DID-FID-Key to improve the security level. |
| 1.1 | 2014/11/26 | Rocky Huang | 1. In section 3.4.1, correct command data format 2. In section 3.6, add extra act indicator in ActCode value. |
| 1.2 | 2015/1/20 | Rocky Huang | 1. In section 3.4.6, add new command RegisterHotelClient for registration using 64-byte new TID credential format. 2. In section 3.5.4.1, add new property, WIFI RSSI. 3. In section 3.6, add new command mode, HOTEL. |
| 1.3 | 2015/4/20 | Rocky Huang | 1. In section 3.5.1, add extra SYNC command response data ‘Password\_len’ and ‘Password’ 2. In section 3.5.4.1, add new properties WIFI module status, WIFI enable, and Channel Mode Schedule 3. Add section 3.5.10, which describe AddPassword command 4. Add section 3.7, which describe gateway related commands |
| 1.4 | 2015/7/13 | Rocky Huang | 1. In section 3.1.1, add TimeZone and DST information in pairing command 2. In section 3.2.2, add new item AR portion which can be used to sync back AR data of AR version 1. 3. In table 3.5.4.1, add new properties time zone, relock delay, activate gateway pairing, lock settings, feature selection, keypad illumination, Wifi enroll site ID, and Wifi enroll token. 4. In section 3.5.11, add new command SyncByFID which supports delete/change clients by FID 5. In section 3.5.12, add new command SetPropertyWithSeq which supports setting properties with sequence number 6. In section 3.11, rename bit1 of 2nd\_entry\_ctrl\_byte from EXP to SUS; add bit2 TECH and bit3 LML. 7. In section 3.12, increase DID-FID-SN field to 2-byte and add additional data for client information change command 8. In section 3.7.7, add new command GetPendingGWAction. 9. In section 3.7.8, add new command ResponseParamGet 10. In section 3.7.9, add new command ResponseParamSet 11. In section 3.7.10, add new command Ack 12. IPA add MAC |
| 1.8 | 2016/8/1 | Rocky Huang | 1. In table 3.5.4.1, rename the property ID 08h’s name WIFI\_TEST\_MACHINE to WIFI\_MACHINE since it is used to indicate the server idx for the gateway to connect to. 2. In section 3.7.2, modify StartPairingRequest such that lock can response the server idx for the gateway to connect to. 3. In section 3.7.8, modify StartAddingRequest such that lock can response the server idx for the gateway to connect to. 4. In section 3.7.3.2, add Bad\_gateway\_model error bit (bit 7) 5. In section 3.7.5, modify NotifyPairingResult command to support passing parameters to lock upon pairing error/succ. 6. In section 3.7.10, modify NotifyAddingResult command to support passing parameters to lock upon pairing error/succ. |
| 1.9 | 2016/11/10 | Rocky Huang | 1. In section 3.5.13, encryption add password command is supported. 2. In section 3.5.14, encryption clone card command is supported. |
| 1.10 | 2017/4/18 | Rocky Huang | 1. In section 3.5.15, new sync command SyncByFID\_NewLog\_Enc is added to support encrypted log content and new log record format which adds 1-byte to store initiator information. 2. In section 3.12, add new log event 3Ch and 3Dh to convey Netcode/Varicode information |
| 1.11 | 2017/11/8 | Rocky Huang | 1. In section 3.2.2, add new AR type and AR portion definition and new response status code 2. In section 3.4.4, add new new response status code 3. In table 3.5.4.1, add new property type 4. In section 3.5.11, add new AR type and AR portion definition 5. In table 3.12, add new log event code |

1. Odin device-phone communication protocol brief:

Odin DEVICE emulates NFC Forum Type 4 Tag and follows ISO 14443-4 standard.

For the PHONE and DEVICE to exchange data, NDEF message is used. Odin communication protocol packets are encapsulated in NDEF record’s PAYLOAD. The operation of the protocol is based on “NFC Forum Type 4 Tag Operation Specification 2.0”.

An Odin command utilizes NDEF UPDATE procedure and NDEF READ procedure if this Odin command expects response data from DEVICE. The Odin command data is encapsulated in NDEF message which NDEF UPDATE procedure will perform. The Odin response data is encapsulated in NDEF message which is read back via NDEF READ procedure.

*PHONE DEVICE*

*NDEF\_UPDATE*

*(Cmd\_mode, Cmd\_code, Cmd\_data\_len,*

*Command specified data) 🡪*

*NDEF\_READ () 🡪*

*🡨 (Status OK, Cmd\_mode, Cmd\_code, res\_data\_len, command specified response data))*

1. Odin device-phone communication protocol packet format:
   1. Command packet format: Odin command packets are transmitted in NDEF message. The NDEF message should contain only a single record and the record should be **short record**. This means **MB and ME are both 0x1** while **CF and IL being 00h**. **TNF should be 02h** which indicates a media-type. Type has prefix ‘pkinno/odin’ and thus type length is at least 0Bh. For example, there might be media-type ‘pkinno/odin/0000/00’ which includes app specific code ‘0001’ and protocol version ‘00’. Payload is the Odin command to issue and it is always 2-byte long. So the transmitted NDEF message is of the following format:

***Flags&TNF TLEN PLEN TYPE PAYLOAD***

*D2h 0Bh 43h 70h 6Bh 69h 6Eh 6Eh 6Fh 0x2Fh 6Fh 64h 69h 6Eh 01h 01h…*

The PAYLOAD in the NDEF message holds Odin command. An Odin command’s PAYLOAD consists of at least 4 bytes, which are 2-byte protocol version, 1-byte cmd\_mode, and 1 byte cmd\_code, respectively. If this command contains some command specified data to be sent to DEVICE, then a third byte command data length field follows, and then starting from the fourth byte there are variable bytes of command specified data. For example, here is Odin command (Pairing, SendRequest) which has command PAYLOAD cmd\_mode 01h (Pairing) and command code 01h (SendRequest) and then command data length which is 40h and then the 40h bytes of command data payload.

*Protocol\_Ver mode code data\_len Data*

*80h 05h 01h (PAIRING) 01h (SendRequest) 40h command specified 40h bytes of data*

* 1. Response packet format: Odin response packets are transmitted in NDEF format. The NDEF message should contain only a single record and the record should be **short record**. This means **MB and ME are both 01h** while **CF and IL being 00h**. **TNF should be 02h** which indicates a media-type. Type is ‘pkinno/odin’ and thus type length is 0xb. Payload is the Odin command to issue and it is always 2-byte long. So the transmitted NDEF message is of the following format:

***Flags&TNF TLEN PLEN TYPE PAYLOAD***

*D2h 0Bh 24h 70h 6Bh 69h 6Eh 6Eh 6Fh 2Fh 6Fh 64h 69h 6Eh 00h 01h 01h …*

The PAYLOAD in the NDEF message holds Odin response. An Odin response contains at least 3 bytes, which are 1 byte cmd status, 1 byte cmd\_mode, and 1 byte cmd\_code, respectively. If this response contains some command specified response data to be sent back to PHONE, then starting from the fourth byte there are the real command specified response data bytes. The leading 3 bytes represent the status of last Odin command issued by PHONE. Status byte value ‘00h’ means the command successfully performed, while non-zero status byte value indicates some error occurs during the command handling. The following example is the response packet of Odin command (PAIRING, SendRequest).

*Status mode code data\_len data*

*00h (OK) 01h (PAIRING) 01h (SendRequest) 31h command specified response data*

1. Command List

An Odin command contains at least 2 byte. The first byte is command mode, which describe the category of the command, such as PAIRING, AUTH, IPA, PRC, and MANAGEMENT. The second byte is command code, which indicates the action, such as SendRequest, Finish, and ReadAddedClient.

* 1. Command mode PAIRING:

The command mode byte value is 01h.

* + 1. SendRequest:

The command code byte value is 01h.

PHONE issues the command to send pairing procedure request with necessary data. It can only be issued by PHONE after DEVICE’s setup button being pressed. The command format is as follows, where DIN, DN, and FID are of 16 bytes, and Time is 4-byte current time, and USERNAME is no greater than 16 bytes. Then 16-byte App\_key\_seed and 16-byte Challenge.

Mode Code Data\_len Data\_bytes

01h 01h 54h + USERNAME length DIN, DN, FID, Time, USERNAME, App\_key\_seed,

Data\_bytes (cont.)

Challenge

The response format is as follows. The DEVICE’s 16-byte DID, the 16-byte scrambled output of the DID-FID-Key and a random number, denoted by S2(DID-FID-Key, Random), the 16-byte random number itself, and the 2-byte DID-FID-SN are sent back in response data bytes. Then there is a 2-byte integer, M, which indicates the number of log event record contained in the response data. The 4-byte long START\_LOG\_SN is the serial number of the first log event record sent back. Then there are variable byte of log event record data. Then there is 16-byte FW\_Version and 16-byte SHA256-HMAC response calculated by App\_key and the challenge sent by PHONE. Then there is the 4-byte admin\_rolling\_number. The there is 16-byte DID-FID-Time-Hash used for claiming lock from Asgard server. At the end there is 4-byte latest sequence number.

Status Mode Code Data\_len Data\_bytes

00h if OK 01h 01h 7Dh DID, S2(DID-FID-Key, Random), Random, DID-FID-SN,

Data\_bytes (cont.)

M, START\_LOG\_SN, DID-FID-SN\_Log\_0, LogEvent\_0, Time\_0, FW\_Version,

Data\_bytes (cont.)

SHA256-HMAC(App\_key, Challenge),

Data\_bytes (cont.)

Admin\_rolling\_number, DID-FID-Time-Hash, Seq\_number, BLE\_MAC

If some error happens, the status code is as follows and the response data is not used.

Status Mode Code Data\_len Data\_bytes

01h if misc fails 01h 01h N/A N/A

06h if device not in setup 01h 01h N/A N/A

07h if DIN check error 01h 01h N/A N/A

* + 1. Finish:

The command code byte value is 02h.

PHONE can issue the command to notify DEVICE the completion of reading necessary response data of a former SendReqeust command and all paring procedure successfully performed. Then there is 1-byte Ext\_flag descriptor which indicates whether there are extra data to set to the lock when pairing. If bit 0 of Ext\_flag is on, the 1-byte TimeZone, 4-byte DST\_start, 4-byte DST\_end, and 1-byte DST\_hour are provided. If bit 1 of Ext\_flag is on, the 6-byte LID is provided for the lock as a source to generated key for NetCode.

Mode Code Data\_len Data\_bytes

01h 02h 1 + {10} + {6} Ext\_flag, {TimeZone, DST\_start, DST\_end, DST\_hour}. {LID}

The response of the command might be skipped by the PHONE. The PHONE can use UpdateBinary command’s response SW1 and SW2 bytes in ISO14443-4 layer to tell if the command succeeds.

Status Mode Code Data\_len Data\_bytes

00h if OK 01h 02h N/A N/A

01h if PAIRING fails 01h 02h N/A N/A

* + - 1. Bits of Ext\_Flag

|  |  |
| --- | --- |
| Bit | Data to provide |
| 0 | TimeZone, DST\_start, DST\_end, DST\_hour |
| 1 | LID |

* 1. Command mode AUTH:

The command mode byte value is 02h.

* + 1. SendRequest

The command code byte value is 01h.

PHONE issues the command to let PHONE and DEVICE perform mutual authentication. The command format is as follows. 16-byte FID is the leading data to be sent. Then there is 16-byte Challenge2 and 4-byte current time. In protocol version 8008 and newer protocol version, additional information obtained from login is required – 4-byte Login\_Nbr, 4-byte Login\_time, and 32-byte Login-Nbr-HASH, and 1-byte Num\_of\_Lock. There will be at least one set of 16-byte DID and 2-byte DID-FID-SN, denoted by DID\_0 and DID-FID-SN\_0, if PHONE only relates to a single Odin DEVICE. If PHONE has registered itself to multiple Odin DEVICEs, more DID\_n and DID-FID-SN\_n will be sent.

Mode Code Data\_len Data\_bytes

02h 01h 4Dh + 12h x num\_of\_lock FID, Challenge2, Time, Login\_Nbr, Login\_Time,

Data\_bytes (cont.)

Login-Nbr-HASH, Num\_of\_lock, DID\_0, DID-FID-SN\_0, {DID\_1, DID-FID-SN\_1, …, DID\_n,

Data\_bytes (cont.)

DID-FID-SN\_n}

The response format is as follows. 16-byte DID, 16-byte response2, and 16-byte challenge1, and 16-byte device name, 16-byte FW\_Version, 1-byte Battery\_ADC\_Value, and 2-byte num\_of\_pedning\_log, will be sent back to PHONE. In protocol version 8008 or newer, an optional 16-byte Response2\_ex might be transmitted which is computed by DID-TID-Key for a registered PRC/OTA clients.

When the DEVICE accept the command, the status code is 00h. If PHONE does not send the DID which matches DEVICE’s, the status code is 01h. If PHONE sends out correct DID but is not a valid SN, the status code is 0Fh. If PHONE sends out correct DID but is not a valid SN, and currently DEVICE is in IPA state, the status code is 0Ch. If the DEVICE is in SETUP state, the status code is 0Dh. If the device contains some changes on client list which has not yet been reflected to ADMIN, the status code is 10h. Status code value 20h means login number is expired.

Status Mode Code Data\_len Data\_bytes

00h if OK 02h 01h 53h + {10h} DID, Response2, Challenge,

Data\_bytes (cont.)

Device\_name, FW\_Version, Battery\_ADC\_Value, Num\_of\_pending\_log, {Response2\_ex}

Following are other possible Status Code value.

Status

01h if fails (DID mismatch)

02h if the command packet is in incorrect format

03h if the command is issued but currently the state of the lock cannot accept the command

04h if the client owing this SN is TID client and the lock is in IPA mode

07h if the command is ok and the channel mode is active in this lock

08h if the command is ok and the lock is already opened (but not due to channel mode)

09h if the lock has not yet been correctly initialized (e.g. model name not yet set)

0Ah if the command is ok and the lock is in no-disturb mode

0Bh if the client owing this SN is TID client

0Ch if not valid SN and device in IPA

0Dh if device in SETUP

0Eh if not valid SN

0Fh if device in IPA

10h if sync for client list required (only used by admin)

20h if login number expired

* + 1. Finish:

The command code byte value is 02h.

PHONE issue the command to transmit 16-byte Response1, 1-byte ActCode, which describe the action to be taken after the completion of authentication. If ActCode is UNLOCK or MANAGEMENT, 1-byte UserName length, and variable length user name will be transmitted. If ActCode is UNLOCK\_WITH\_TEMP\_RELOCK\_DELAY, 1-byte temp\_relock\_delay will be transmitted. The unit of temp relock delay is 5 seconds. The DEVICE will verify the response 1. The ActCode table is described in section 3.6.

Mode Code Data\_len Data\_bytes

02h 02h 11h + variable data Response1, ActCode,

Data\_bytes (cont.)

{ (UsernameLen, Username) | (Temp\_relock\_delay) }

In case of a correct Response1, the AUTH procedure is complete.

The response might contain extra data which Device needs to notify client. The first byte ‘Num\_of\_item’ describe how many items the device will send to client. Then there are sets of {ItemID, ItemLen, ItemData} which contains the data item to transmit. M is the sum of all {ItemID, ItemLen, ItemData} sets transmitted.

Status Mode Code Data\_len Data\_bytes

00h if OK 02h 02h 1h + M Num\_of\_item,

Data\_bytes

{ItemID\_0, ItemLen\_0, ItemData\_0},, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

05h if OK but bypass motor action 02h 02h 1h + M Num\_of\_item,

Data\_bytes (cont.)

{ItemID\_0, ItemLen\_0, ItemData\_0}, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

08h if OK and door is already unlocked 02h 02h 1h + M Num\_of\_item,

Data\_bytes (cont.)

{ItemID\_0, ItemLen\_0, ItemData\_0}, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Here is the table of possible Items:

|  |  |  |
| --- | --- | --- |
| Item | Item ID | Length (in bytes) |
| Administrator’s name | 0 | 1~16 |
| Device’s name | 1 | 16 |
| Access right (plain) | 2 | 16 |
| Log event | 3 | 10 |
| Access right ext flag | 4 | 1 |
| AR portion (Only support in protocol 8007 or newer) | 5 | Variable  Pkinno type: 22  Codelocks type: 48  KIC type: 20 (Only support in 8012 or newer) |
| Suspend TS | 6 | 4 |

For item ID 4, the Access right ext flag bit value:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| RFU | RFU | RFU | RFU | RFU | RFU | SUS | OT EXP |

For Item ID 5, the AR portion data format is as follows (identical format as in MANAGEMENT mode Sync command)

AR portion:

{AR\_version, AR\_len, AR\_data, AR\_TS}

AR\_version is 1 byte.

AR\_len is 1 byte.

AR\_data is multiple bytes.

AR\_TS is 4 bytes.

AR portion format for AR version 0 (one 16-byte-AR):

{00h, 10h, 16-byte AR\_data, AR\_TS}

AR portion format for AR version 1 (Up to ten 4-byte-AR):

{01h, 02h + 4h \* i, 1st\_AR\_bitmsk, 2nd\_AR\_bitmsk, AR\_0, AR\_1, …, AR\_i-1, AR\_TS}

1st\_AR\_bitmsk:

Controlling whether to update the corresponding AR

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Bit 0 | Bit 1 | Bit 2 | Bit 3 | Bit 4 | Bit 5 | Bit 6 | Bit 7 |
| AR 0 | AR 1 | AR 2 | AR 3 | AR 4 | AR 5 | AR 6 | AR 7 |

2nd\_AR\_bitmsk:

Controlling whether to update the corresponding AR

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Bit 0 | Bit 1 | Bit 2 | Bit 3 | Bit 4 | Bit 5 | Bit 6 | Bit 7 |
| AR 8 | AR 9 | RFU | RFU | RFU | RFU | RFU | RFU |

AR portion format for AR version 2 (Up to three 4-byte-AR):

{01h, 02h + 4h \* i, 1st\_AR\_bitmsk, 2nd\_AR\_bitmsk, AR\_0, AR\_1, …, AR\_i-1, AR\_TS}

1st\_AR\_bitmsk:

Controlling whether to update the corresponding AR

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Bit 0 | Bit 1 | Bit 2 | Bit 3 | Bit 4 | Bit 5 | Bit 6 | Bit 7 |
| AR 0 | AR 1 | AR 2 | RFU | RFU | RFU | RFU | RFU |

2nd\_AR\_bitmsk:

Controlling whether to update the corresponding AR

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Bit 0 | Bit 1 | Bit 2 | Bit 3 | Bit 4 | Bit 5 | Bit 6 | Bit 7 |
| RFU | RFU | RFU | RFU | RFU | RFU | RFU | RFU |

Status Mode Code Data\_len Data\_bytes

01h if Response1 incorrect 02h 02h N/A N/A

02h if requiring PIN validation 02h 02h N/A N/A

03h if requiring PIN setup 02h 02h N/A N/A

04h if IPA/batch re-add attempt 02h 02h N/A N/A

06h if access right denied 02h 02h as normal as normal

07h if lock is in channel mode 02h 02h as normal as normal

09h if 1st auth succ in dual auth mode 02h 02h as normal as normal

After a successful Finish command, the DEVICE will perform requested action according to ActCode, such as unlock the door (ActCode 00h) or IAP flow (ActCode 01h).

The most significant 2 bits of status code byte contains the battery status. The 2-bit battery status has 4 possible values, 0, 1, 2, and 3. If the battery is full, the battery status value is 0. If the battery life left is 2/3, the value is 1. If the battery life left is 1/3, the value is 2. If the battery life is running out, the value is 3. So for example, if the battery life left is 2/3 and there are more data to sync, the status code will be 50h.

Battery Status (Bit 6 and bit 7 of Status Code Byte) Battery

00h Full

01h 2/3

02h 1/3

03h Low



* + 1. Abort:

The command code byte value is 05h.

This command is used to abort AUTH process explicitly.

Mode Code Data\_len Data\_bytes

02h 05h N/A N/A

The response is as follows. Note that a client is allowed to skip reading of the response.

Status Mode Code Data\_len Data\_bytes

00h if OK 02h 05h N/A N/A

01h if fail 02h 05h N/A N/A

* + 1. ChallengeResponse:

The command code byte value is 06h. This command is used to resend Challenge2, DID and 2-byte DID-FID-SN to challenge again if there is no DID match in previous AUTH mode SendRequest command. In protocol version 8008 and newer protocol version, additional information obtained from login is required – 4-byte Login\_Nbr, 4-byte Login\_time, and 32-byte Login-Nbr-HASH. In protocol version 8008 or newer, an optional 16-byte Response2\_ex which is computed by DID-TID-Key for a registered PRC/OTA clients might be sent back to the client app.

Mode Code Data\_len Data\_bytes

02h 06h 4Ah Challenge2, DID, DID-FID-SN, Login\_Nbr,

Data\_bytes (cont.)

Login\_Time, Login\_Nbr\_Hash

The response is

Status Mode Code Data\_len Data\_bytes

00h if OK 02h 06h 20h + {10h} Response2, Challenge, {Response2\_ex}

Following are other possible Status Code value.

Status

01h if fails (DID mismatch)

02h if the command packet is in incorrect format

03h if the command is issued but currently the state of the lock cannot accept the command

04h if the client owing this SN is TID client and the lock is in IPA mode

07h if the command is ok and the channel mode is active in this lock

08h if the command is ok and the lock is already opened (but not due to channel mode)

09h if the lock has not yet been correctly initialized (e.g. model name not yet set)

0Ah if the command is ok and the lock is in no-disturb mode

0Bh if the client owing this SN is TID client

0Ch if not valid SN and device in IPA

0Dh if device in SETUP

0Eh if not valid SN

0Fh if device in IPA

10h if sync for client list required (only used by admin)

20h if login number expired

* + 1. SendRequest\_Ex

The command code byte value is 07h.

This command has the same command data and response data format as SendRequest command. The only difference is that it allows to further issue ChallengeResponse command for the case no DID matched.

* + 1. SendRequest\_Wo\_AutoPairing

The command code byte value is 08h.

This command has the same command data and response data format as SendRequest command. The only difference is that it will not trigger auto pairing in SETUP mode.

* + 1. SendRequest\_Wo\_AutoPairing\_Ex

The command code byte value is 09h.

This command has the same command data and response data format as SendRequest\_Ex command. The only difference is that it will not trigger auto pairing in SETUP mode.

* 1. Command mode IPA:

The command mode byte value is 03h.

* + 1. SendRequest:

The command code byte value is 01h.

The to-be-added PHONE issues this command to add itself in the white list of the DEVICE. The DEVICE will only accept IPA command if the administrator has executed AUTH procedure with ActCode 01h. The command data are 16-byte FID, variable length username (max 16-byte), 16-byte App\_key\_seed, and 16-byte Challenge.

Mode Code Data\_len Data\_bytes

03h 01h 30h + USERNAME length FID, USERNAME, App\_key\_seed, Challenge

The response format is as follows. The DEVICE’s 16-byte DN, 16-byte DID, the 16-byte scrambled output of the DID-FID-Key and a random number, denoted by S2(DID-FID-Key, Random), the 16-byte random number itself, the 2-byte DID-FID-SN, 16-byte Access\_right, and SHA256-HMAC(App\_key, Challenge) are sent back in response data bytes.

Status Mode Code Data\_len Data\_bytes

00h if OK 03h 01h 6Fh DN, DID, S2(DID-FID-Key, Random), Random, DID-FID-SN

Data\_bytes (cont)

Access\_Right, SHA256-HMAC(App\_key, Challenge), Sequence\_number, Rolling\_number, MAC

Status Mode Code Data\_len Data\_bytes

01h if fails 03h 01h N/A N/A

* + 1. ClientFinish:

The command code byte value is 02h.

The to-be-added PHONE issues the command to notify DEVICE the completion of reading necessary response data of a former IPA SendReqeust command and the IPA procedure for thus user is successfully performed.

Mode Code Data\_len Data\_bytes

03h 02h N/A N/A

The response of the command might be skipped by the PHONE. The PHONE can use UpdateBinary command’s response SW1 and SW2 bytes in ISO14443-4 layer to tell if the command succeeds.

For IPA successful case, the response contains extra data which Device need to notify client. The first byte ‘Num\_of\_item’ describe how many items the device will send to client. Then there are sets of {ItemID, ItemLen, ItemData} which contains the data item to transmit. M is the sum of all {ItemID, ItemLen, ItemData} sets transmitted.

Status Mode Code Data\_len Data\_bytes

00h if OK 03h 02h 01h + M Num\_of\_item,

Data\_bytes

{ItemID\_0, ItemLen\_0, ItemData\_0},, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

For IPA fail case, there is no extra data.

Status Mode Code Data\_len Data\_bytes

01h if IPA fails 03h 02h N/A N/A

* + 1. AdminFinish:

The command code byte value is 04h.

The administrator PHONE issue the command to notify DEVICE it has read back the new client’s data.

Mode Code Data\_len Data\_bytes

03h 04h N/A N/A

The response of the command might be skipped by the PHONE. The PHONE can use UpdateBinary command’s response SW1 and SW2 bytes in ISO14443-4 layer to tell if the command succeeds.

The response contains status code 00h if the DEVICE has successfully received the notification. Otherwise the response contains status 01h.

Status Mode Code Data\_len Data\_bytes

00h if OK 03h 04h N/A N/A

01h if IPA fails 03h 04h N/A N/A

* 1. Command mode PRC:

The command mode byte value is 04h.

* + 1. RequestTID:

The command code byte value is 01h.

After issuing AUTH mode SendRequest command, the administrator PHONE can issue this command to request TIDs for PRC. The data bytes contains the 16-byte Response1 computed from Challenge1, and 1-byte Num\_of\_TIDs which is the number of TIDs Admin PHONE would like to acquire.

Mode Code Data\_len Data\_bytes

04h 01h 11h Response1, Num\_of\_TIDs (= n)

The response depends on the number of TIDs requested according to the command, if the response status code is OK, there is at least a set of 16-byte encrypted DID-TID-Key and 2-byte DID-TID-SN. Suppose the number if TIDs requested is n, then there are n pairs in the response data.

Status Mode Code Data\_len Data\_bytes

00h if OK 04h 01h 10h + n x 12h Random, AES\_ECB\_Enc(input = DID-TID-Key\_0,

Data\_bytes (cont.)

key = admin DID-FID-Key), DID-TID-SN\_0,

Data\_bytes (cont.)

AES\_ECB\_Enc(input = DID-TID-Key\_1, key = admin DID-FID-Key), DID-TID-SN\_1, … ,

Data\_bytes (cont.)

AES\_ECB\_Enc(input = DID-TID-Key\_n-1, key = admin DID-FID-Key), DID-TID-SN\_n-1,

Status Mode Code Data\_len Data\_bytes

01h if fail 04h 01h N/A N/A

* + 1. AdminFinish

The command code byte value is 02h.

The administrator PHONE issues this command to notify the DEVICE that it has finished reading all the pairs of 16-byte Scramble2(DID-TID-Key, Random) result and the DID-TID-SN.

Mode Code Data\_len Data\_bytes

04h 02h N/A N/A

The response of the command might be skipped by the PHONE. The PHONE can use UpdateBinary command’s response SW1 and SW2 bytes in ISO14443-4 layer to tell if the command succeeds.

Status Mode Code Data\_len Data\_bytes

00h if OK 04h 02h N/A N/A

01h if IPA fails 04h 02h N/A N/A

* + 1. RegisterClient:

The command code byte value is 04h.

The command is only valid from a PHONE which has performed P2P provision with administrator and should be issued after an AUTH mode SendRequest command. The PHONE issues the command to transmit 10h-byte Response1, USERNAME, and the 30h-byte encrypted data, which is composed of 4-byte sequence\_number + 16-byte access right + 16-byte DID-TID-Key + 2-byte CRC result (input are sequence\_number + access-right + DID-TID-Key) + 10-byte zero-padding.

The DEVICE will check if this is a TID tap by the DID-TID-SN from previous AUTH mode SendRequest command and verify the response1.

Format of Access\_Right field in encrypted data:

In 8006, the Access\_Right field is 16-byte v1 access right

In 8007, the Access\_Right field is

{1st\_ctrl\_mask, 1st\_ctrl\_byte, 2nd\_ctrl\_mask, 2nd\_ctrl\_byte, AR\_portion}

Note that only some bits of the ctrl byte can be set such as OT of 1st\_ctrl\_byte and TECH\_USER of 2nd\_ctrl\_byte.

AR portion:

{AR\_version, AR\_len, AR\_data, AR\_TS}

AR\_version is 1 byte.

AR\_len is 1 byte.

AR\_data is multiple bytes.

AR\_TS is 4 bytes.

Mode Code Data\_len Data\_bytes

04h 04h 40h+USERNAME length Response 1, USERNAME, AES\_ECB\_Enc (input =

Data\_bytes (cont.)

{Sequence\_number, Access\_Right, DID-TID-Key, CRC, padding}, key = Admin DID-FID-Key)

The response format is as follows. The 16-byte scrambled output of the DID-FID-Key and a random number, denoted by S2(DID-FID-Key, Random), the 16-byte random number itself, and the 2-byte DID-FID-SN are sent back in response data bytes.

Status Mode Code Data\_len Data\_bytes

00h if OK 04h 04h 22h S2(DID-FID-Key, Random), Random, DID-FID-SN

01h if fails 04h 04h N/A N/A

* + 1. ClientFinish:

The command code byte value is 05h.

The to-be-added PHONE issues the command to notify DEVICE the completion of reading necessary response data of a former RegisterPRCClient command and the client will be added to the DEVICE.

Mode Code Data\_len Data\_bytes

04h 05h N/A N/A

The response of the command might be skipped by the PHONE. The PHONE can use UpdateBinary command’s response SW1 and SW2 bytes in ISO14443-4 layer to tell if the command succeeds.

Status Mode Code Data\_len Data\_bytes

00h if OK 04h 05h 1h + M Num\_of\_item,

Data\_bytes

{ItemID\_0, ItemLen\_0, ItemData\_0},, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

05h if OK but bypass motor action 04h 05h 1h + M Num\_of\_item,

Data\_bytes (cont.)

{ItemID\_0, ItemLen\_0, ItemData\_0}, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

08h if OK and door is already unlocked 04h 05h 1h + M Num\_of\_item,

Data\_bytes (cont.)

{ItemID\_0, ItemLen\_0, ItemData\_0}, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

01h if fails 04h 05h N/A N/A

06h if access right denied 04h 05h as normal as normal

07h if lock is in channel mode 04h 05h as normal as normal

09h if 1st auth succ in dual auth mode 04h 05h as normal as normal

* + 1. UpdateAccessRight:

The command code byte value is 06h.

The command is only valid from a PHONE which has performed P2P provision with administrator and should be issued after an AUTH mode SendRequest command. The PHONE issues the command to transmit 10h-byte Response1 and the 30h-byte encrypted data, which is composed of 4-byte sequence\_number + multiple-byte access right + 16-byte recipient’s FID + 2-byte CRC (input are sequence\_number + access-right + recipient’s FID) + multiple-byte zero-padding.

Format of Access\_Right field in encrypted data:

In 8006, the Access\_Right field is 16-byte v1 access right

In 8007, the Access\_Right field is

{1st\_ctrl\_mask, 1st\_ctrl\_byte, 2nd\_ctrl\_mask, 2nd\_ctrl\_byte, AR\_portion}

Note that only some bits of the ctrl byte can be set such as OT of 1st\_ctrl\_byte and TECH\_USER of 2nd\_ctrl\_byte.

AR portion:

{AR\_version, AR\_len, AR\_data, AR\_TS}

AR\_version is 1 byte.

AR\_len is 1 byte.

AR\_data is multiple bytes.

AR\_TS is 4 bytes.

Mode Code Data\_len Data\_bytes

04h 06h 40h Response 1, AES\_ECB\_Enc (input =

Data\_bytes (cont.)

{Sequence\_number, 1st\_mask, 1st\_byte, 2nd\_mask, 2nd\_byte, AR\_Portion, recipient’s FID, CRC, padding}, key = Admin\_DID-FID-Key)

The response may contain items to sync back to admin.

Status Mode Code Data\_len Data\_bytes

00h if OK 04h 06h 1h + M Num\_of\_item,

Data\_bytes

{ItemID\_0, ItemLen\_0, ItemData\_0},, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

05h if OK but bypass motor action 04h 06h 1h + M Num\_of\_item,

Data\_bytes (cont.)

{ItemID\_0, ItemLen\_0, ItemData\_0}, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

08h if OK and door is already unlocked 04h 06h 1h + M Num\_of\_item,

Data\_bytes (cont.)

{ItemID\_0, ItemLen\_0, ItemData\_0}, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

01h if fails 04h 06h N/A N/A

* + 1. RegisterHotelClient:

The command code byte value is 07h.

The command is only valid from a PHONE which has a Dion Hotel OTA Credential after an AUTH mode SendRequest command. The PHONE issues the command to transmit 10h-byte Response1, USERNAME, and the 40h-byte encrypted data, which is composed of 4-byte sequence\_number + 16-byte access right + 16-byte DID-TID-Key + 16-byte FID + 2-byte CRC result (input are sequence\_number + access-right + DID-TID-Key) + 10-byte zero-padding.

The DEVICE will check if this is a TID tap by the entire AES encrypted block.

Mode Code Data\_len Data\_bytes

04h 07h 50h+USERNAME length USERNAME, AES\_ECB\_Enc (input =

Data\_bytes (cont.)

{Sequence\_number, Access\_Right, DID-TID-Key, FID, CRC, padding}, key = Admin DID-FID-Key)

The response format is as follows. The 16-byte scrambled output of the DID-FID-Key and a random number, denoted by S2(DID-FID-Key, Random), the 16-byte random number itself, and the 2-byte DID-FID-SN are sent back in response data bytes.

Status Mode Code Data\_len Data\_bytes

00h if OK 04h 07h 21h S2(DID-FID-Key, Random), Random, DID-FID-SN

01h if fails 04h 07h N/A N/A

* + 1. UpdateSuspendStatus:

The command code byte value is 08h.

The command is only valid from a PHONE which has performed P2P provision with administrator and should be issued after an AUTH mode SendRequest command. The PHONE issues the command to transmit 10h-byte Response1 and the 30h-byte encrypted data, which is composed of 4-byte sequence\_number + 1-byte SUS status + 4-byte SUS TS + 16-byte recipient’s FID + 2-byte CRC (input are sequence\_number + SUS status + SUS TS + recipient’s FID) + 5-byte zero-padding.

Mode Code Data\_len Data\_bytes

04h 08h 30h Response 1, AES\_ECB\_Enc (input =

Data\_bytes (cont.)

{Sequence\_number, SUS\_Status, SUS\_TS, FID, CRC, padding}, key = Admin\_DID-FID-Key)

The response may contain items to sync back to admin.

Status Mode Code Data\_len Data\_bytes

00h if OK 04h 08h 1h + M Num\_of\_item,

Data\_bytes

{ItemID\_0, ItemLen\_0, ItemData\_0},, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

05h if OK but bypass motor action 04h 08h 1h + M Num\_of\_item,

Data\_bytes (cont.)

{ItemID\_0, ItemLen\_0, ItemData\_0}, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

08h if OK and door is already unlocked 04h 08h 1h + M Num\_of\_item,

Data\_bytes (cont.)

{ItemID\_0, ItemLen\_0, ItemData\_0}, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

01h if fails 04h 06h N/A N/A

* + 1. ClientFinishWithActCode:

The command code byte value is 09h.

The command is available in protocol version 8009 and newer version.

The command is similar to ClientFinish.

The to-be-added PHONE issues the command to notify DEVICE the completion of reading necessary response data of a former RegisterPRCClient command and the client will be added to the DEVICE.

Mode Code Data\_len Data\_bytes

04h 09h 1 ActCode

The response of the command might be skipped by the PHONE. The PHONE can use UpdateBinary command’s response SW1 and SW2 bytes in ISO14443-4 layer to tell if the command succeeds.

Status Mode Code Data\_len Data\_bytes

00h if OK 04h 09h 1h + M Num\_of\_item,

Data\_bytes

{ItemID\_0, ItemLen\_0, ItemData\_0},, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

05h if OK but bypass motor action 04h 09h 1h + M Num\_of\_item,

Data\_bytes (cont.)

{ItemID\_0, ItemLen\_0, ItemData\_0}, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

08h if OK and door is already unlocked 04h 09h 1h + M Num\_of\_item,

Data\_bytes (cont.)

{ItemID\_0, ItemLen\_0, ItemData\_0}, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

01h if fails 04h 09h N/A N/A

* + 1. UpdateAccessRightWithActCode:

The command code byte value is 0Ah.

The command is available in protocol version 8009 and newer version.

The command is similar to UpdateAccessRight.

The command is only valid from a PHONE which has performed P2P provision with administrator and should be issued after an AUTH mode SendRequest command. The PHONE issues the command to transmit 10h-byte Response1 and the 30h-byte encrypted data, which is composed of 4-byte sequence\_number + multiple-byte access right + 16-byte recipient’s FID + 2-byte CRC (input are sequence\_number + access-right + recipient’s FID) + multiple-byte zero-padding.

Format of Access\_Right field in encrypted data:

In 8006, the Access\_Right field is 16-byte v1 access right

In 8007, the Access\_Right field is

{1st\_ctrl\_mask, 1st\_ctrl\_byte, 2nd\_ctrl\_mask, 2nd\_ctrl\_byte, AR\_portion}

Note that only some bits of the ctrl byte can be set such as OT of 1st\_ctrl\_byte and TECH\_USER of 2nd\_ctrl\_byte.

AR portion:

{AR\_version, AR\_len, AR\_data, AR\_TS}

AR\_version is 1 byte.

AR\_len is 1 byte.

AR\_data is multiple bytes.

AR\_TS is 4 bytes.

Mode Code Data\_len Data\_bytes

04h 0Ah 41h Response 1, AES\_ECB\_Enc (input =

Data\_bytes (cont.)

{Sequence\_number, 1st\_mask, 1st\_byte, 2nd\_mask, 2nd\_byte, AR\_Portion, recipient’s FID, CRC, padding}, key = Admin\_DID-FID-Key) , ActCode

The response may contain items to sync back to admin.

Status Mode Code Data\_len Data\_bytes

00h if OK 04h 0Ah 1h + M Num\_of\_item,

Data\_bytes

{ItemID\_0, ItemLen\_0, ItemData\_0},, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

05h if OK but bypass motor action 04h 0Ah 1h + M Num\_of\_item,

Data\_bytes (cont.)

{ItemID\_0, ItemLen\_0, ItemData\_0}, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

08h if OK and door is already unlocked 04h 0Ah 1h + M Num\_of\_item,

Data\_bytes (cont.)

{ItemID\_0, ItemLen\_0, ItemData\_0}, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

01h if fails 04h 0Ah N/A N/A

* + 1. UpdateSuspendStatusWithActCode:

The command code byte value is 0Bh.

The command is available in protocol version 8009 and newer version.

The command is similar to UpdateSuspendStatus.

The command is only valid from a PHONE which has performed P2P provision with administrator and should be issued after an AUTH mode SendRequest command. The PHONE issues the command to transmit 10h-byte Response1 and the 30h-byte encrypted data, which is composed of 4-byte sequence\_number + 1-byte SUS status + 4-byte SUS TS + 16-byte recipient’s FID + 2-byte CRC (input are sequence\_number + SUS status + SUS TS + recipient’s FID) + 5-byte zero-padding.

Mode Code Data\_len Data\_bytes

04h 0Bh 31h Response 1, AES\_ECB\_Enc (input =

Data\_bytes (cont.)

{Sequence\_number, SUS\_Status, SUS\_TS, FID, CRC, padding}, key = Admin\_DID-FID-Key), ActCode

The response may contain items to sync back to admin.

Status Mode Code Data\_len Data\_bytes

00h if OK 04h 0Bh 1h + M Num\_of\_item,

Data\_bytes

{ItemID\_0, ItemLen\_0, ItemData\_0},, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

05h if OK but bypass motor action 04h 0Bh 1h + M Num\_of\_item,

Data\_bytes (cont.)

{ItemID\_0, ItemLen\_0, ItemData\_0}, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

08h if OK and door is already unlocked 04h 0Bh 1h + M Num\_of\_item,

Data\_bytes (cont.)

{ItemID\_0, ItemLen\_0, ItemData\_0}, … ,{ItemID\_N-1, ItemLenN-1, ItemData\_N-1}

Status Mode Code Data\_len Data\_bytes

01h if fails 04h 0Bh N/A N/A

* 1. Command mode MANAGEMENT

The command mode byte value is 05h.

* + 1. Sync:

The command code byte value is 03h.

After issuing AUTH mode SendRequest command, the administrator PHONE can issue this command to synchronize client list with DEVICE in the following 2 cases. The first case happens when the administrator PHONE has changed/deleted clients since last synchronization with the DEVICE. The second case can be invoked when the administrator received status code 10h which indicates DEVICE has some changes on client lists to be reflected to the administrator. The command data byte format is as follows. The first 11h byte are 10h-byte Response1 and 1-byte ActCode. Then ADMIN need to specify the first record to read, by putting a 4-byte integer, START\_LOG\_SN, after ActCode, and then 1-byte integer, MAX\_READ\_NBR, which defines the maximum number of log the DEVICE should response. Then, there is a 4-byte integer, LAST\_LOG\_SN, which describes the last received log event record. Note that if it is the first Sync command sent after AUTH mode SendRequest command, the ADMIN should set LAST\_SYNC\_LOG\_SN to the same value as START\_LOG\_SN. If it is not, is should set LAST\_SYNC\_LOG\_SN as the last received log event record’s serial number, as an acknowledgment to let DEVICE know last successfully log event synchronization. By default the DEVICE deletes log event records till LAST\_SYNC\_LOG\_SN+1, but the ADMIN can use ActCode value 06h so that the DEVICE will not delete log event from its storage. Suppose there are N clients whose client list entry information needs to update to DEVICE side, then N sets of (DID-FID-SN, Mask, Entry\_ctrl\_byte, Access\_Right) will be appended in the command data byte, where DID-FID-SN is 2-byte, Mask is 1-byte, Entry\_ctrl\_byte is 1-byte, and Access\_Right is 16-byte. If DID-FID-SN here is set to max value 0xFF, then it does not modify any client’s access right, and instead it will be used at later IPA client.

Then there are data related to log event record synchronization.

Mode Code Data\_len Data\_bytes

05h 03h 1Eh + (14h x N) Response1, ActCode,

Data\_bytes (cont.)

START\_LOG\_SN, MAX\_READ\_NBR, LAST\_SYNC\_LOG\_SN,

Data\_bytes (cont.)

{DID-FID-SN\_0,Mask\_0, Enrty\_ctrl\_byte\_0, Access\_right\_0,

Data\_bytes (cont.)

DID-FID-SN\_1, Mask\_1, Entry\_ctrl\_byte\_1, Access\_right\_1, … , Access\_right\_N-1},

Data\_bytes (cont.)

Sequence\_number

If the DEVICE side also has some changes on client entries since latest synchronization event with administrator PHONE, it will put it in response. Also, the device is possible to append log events since last synchronization. The response data byte format is as follows. The first byte N describes the number of client to be synchronized to admin. M is a 1-byte integer which describes the number of logs to be synchronized to admin. L is 1-byte integer denoting offset to get the first byte of the four-byte LOG\_START\_SN, and this offset is relative to start of Data\_bytes. Suppose there are N clients whose client list entry information needs to be synchronized to ADMIN side, then N sets of (DID-FID-SN, 1st\_Mask, 1st\_ctrl\_byte, 2nd\_Mask, 2nd\_ctrl\_byte, FID, Username\_len, Username, Access\_right, Seq\_num) events will be appended in the response data byte. If there is changed on the client’s username, the corresponding bit must be set in Mask and Ctrl\_byte. FID is the client’s FID, Username\_len is the length is the length of new user name in byte, Username is the data bytes of the user name, access\_right is 16-byte, and Seq\_num is 4-byte.

If there is pending log to be synchronized to ADMIN, then a 4-byte integer, START\_LOG\_SN, will be appended in the response data, after clients’ changes. START\_LOG\_SN indicates the serial number of the first log, and it will increment one per log event, and it will wrap around at 4-byte boundary. Then M pairs of (DID-FID-SN\_Log, LogEvent, Time {, M\_FID}) will be appended in the response data. The definition of LogEvent is described in section 3.8. M\_FID only exists in the case when LogEvent are AUTH\_FAIL\_GIN\_INCORRECT (18h), AUTH\_FAIL\_GIN\_NOT\_EXIST (19h), and AUTH\_FAIL\_UNKNOWN\_CARD (20h). If LogEvent is AUTH\_FAIL\_GIN\_INCORRECT (18h) or AUTH\_FAIL\_GIN\_NOT\_EXIST (19h), meaning this a phone client, M\_FID is 16-byte long and represent the client’s FID. If LogEvent is AUTH\_FAIL\_UNKNOWN\_CARD (20h), meaning this a card client, then M\_FID is 16-byte long, and the first byte of M\_FID is card UID length, then there are UID bytes.

|  |  |  |
| --- | --- | --- |
| UID len | UID bytes | Padding bytes (must be all zero) |
| (1-byte) | (4, 7, or 10 bytes) | 16 - UID\_len - 1 |

In the command table depicted below, value M stands for the total number of log to be synchronized to ADMIN. The value Ma stands for the number of log of valid clients, and the value Mb stands for the number of log of invalid clients (phone or card not yet added or been deleted), whose FIDs also need to be reported to ADMIN. Note that M = Ma + Mb.

In the command table depicted below, the value P is introduced by the usernames of all clients sync.

Status Mode Code Data\_len Data bytes (if N >0 or M>0)

00h if OK 05h 03h 07h + 43 x N+ 6 x Ma + 22 x Mb + P N, M, L

Data\_bytes (cont.)

{SN\_0, 1st\_Mask\_0, 1st\_Ctrl\_byte\_0, 2nd\_Mask\_0, 2nd\_Ctrl\_byte\_0, FID\_0, Username\_len\_0,

Data\_bytes (cont.)

Username\_0, Access\_Right\_0, Seq\_0, SN\_1, 1st\_Mask\_1, 1st\_Ctrl\_byte\_1, 2nd\_Mask\_1,

Data\_bytes (cont.)

2nd\_Ctrl\_byte\_1, FID\_1, Username\_len\_1, Username\_1, Access\_Right\_1, Seq\_1, …, SN\_n-1,

Data\_bytes (cont.)

1st\_Mask\_n-1, 1st\_Ctrl\_byte\_n-1, 2nd\_Mask\_n-1, 2nd\_Ctrl\_byte\_n-1, FID\_n-1,

Data\_bytes (cont.)

Username\_len\_n-1, Username\_n-1, Access\_Right\_n-1, Seq\_n-1}, START\_LOG\_SN,

Data\_bytes (cont.)

DID-FID-SN\_Log\_0, LogEvent\_0, Time\_0, {M\_FID}, DID-FID-SN\_Log\_1, LogEvent\_1, Time\_1,

Data\_bytes (cont.)

DID-FID-SN\_Log\_M-1, LogEvent\_M-1, Time\_M-1}

Beside status value 00h for OK, if the device still has more log not yet read by ADMIN, the status code is 10h. If the ADMIN get this response status, it can send Sync command again until there is no more data to sync, and end the transaction with an AckSync command.

Status Mode Code Data\_len Data bytes (if N >0 or M>0)

10h if more to sync 05h 03h 07h + 42 x N+ 6 x Ma + 22 x Mb + P N, M, L

Data\_bytes (cont.)

{SN\_0, 1st\_Mask\_0, 1st\_Ctrl\_byte\_0, 2nd\_Mask\_0, 2nd\_Ctrl\_byte\_0, FID\_0, Username\_len\_0,

Data\_bytes (cont.)

Username\_0, Access\_Right\_0, Seq\_0, SN\_1, 1st\_Mask\_1,

Data\_bytes (cont.)

1st\_Ctrl\_byte\_1, 2nd\_Mask\_1, 2nd\_Ctrl\_byte\_1, FID\_1, Username\_len\_1, Username\_1,

Data\_bytes (cont.)

Access\_Right\_1, Seq\_1, …, SN\_n-1, 1st\_Mask\_n-1,

Data\_bytes (cont.)

1st\_Ctrl\_byte\_n-1, 2nd\_Mask\_n-1, 2nd\_Ctrl\_byte\_n-1, FID\_n-1, Username\_len\_n-1,

Data\_bytes (cont.)

Username\_n-1, Access\_Right\_n-1, Seq\_n-1 },

Data\_bytes (cont.)

START\_LOG\_SN, DID-FID-SN\_Log\_0, LogEvent\_0, Time\_0, {M\_FID}, DID-FID-SN\_Log\_1,

Data\_bytes (cont.)

LogEvent\_1, Time\_1, DID-FID-SN\_Log\_M-1, LogEvent\_M-1, Time\_M-1}

The most significant 2 bits of status code byte contains the battery status. The 2-bit battery status has 4 possible values, 0, 1, 2, and 3. If the battery is full, the battery status value is 0. If the battery life left is 2/3, the value is 1. If the battery life left is 1/3, the value is 2. If the battery life is running out, the value is 3. So for example, if the battery life left is 2/3 and there are more data to sync, the status code will be 50h.

Battery Status (Bit 6 and bit 7 of Status Code Byte) Battery

00h Full

01h 2/3

02h 1/3

03h Low

The following is error case response.

Status Mode Code Data\_len Data\_bytes

01h if Response1 incorrect 05h 03h N/A N/A

02h if requiring PIN validation 05h 03h N/A N/A

05h if OK but bypass unlock door 05h 03h Same as OK case Same as OK case

(because handle position incorrect)

08h if OK but bypass unlock door 05h 03h Same as OK case Same as OK case

(because door is already unlocked)

15h if has more to sync but bypass 05h 03h Same as OK case Same as OK case

unlock door (because handle position incorrect)

18h if has more to sync but bypass 05h 03h Same as OK case Same as OK case

unlock door (because door is already unlocked)

* + 1. SetUserName:

The command code byte value is 05h.

Only valid after ADMIN issuing MANAGEMENT mode SYNC command. Only ADMIN can issues this command to change its user name stored at DEVICE.

Mode Code Data\_len Data\_bytes

05h 05h 01h + username\_len Usename\_len, Username

The response contains status code 00h if the client’s new UserName has been successfully updated to the DEVICE.

Status Mode Code Data\_len Data\_bytes

00h if OK 05h 05h N/A N/A

01h if NG 05h 05h N/A N/A



* + 1. FWUPG:

The command code byte value is 06h.

After AUTH procedure completed either by AUTH mode Finish command or AUTH mode ValidatePIN command, with ActCode 05h. The administrator PHONE can issue this command to upgrade new FW image to DEVICE. The ADMIN must slice the FW upgrade package into 192-byte packet and correspond each packet with a SN, which is 2 byte long. The last packet might not be 192-byte long. The first 192-byte packet has SN 0, the second 192-byte packet has SN 1, and the third 192-byte packet has SN 2, and so on. The ADMIN initiates FW upgrade by sending first FWUPG command with UPG action set to 00h. Then ADMIN can transmit remaining packet by sending FWUPG command with UPG action set to 01h. When ADMIN transmit the last packet, it must set UPG action to 02h to indicate the end of the FW upgrade transmission.

Mode Code Data\_len Data\_bytes

05h 06h 04h + length of FW packet payload UPG action byte, FW packet SN,

Data\_bytes (cont.)

FW packet length byte, FW packet payload

Normally, the response status code is 00h upon device successfully receiving packet. However, the device might set status code 40h to indicate the

The DEVICE will response the next FW packet SN to receive. Note that the device can request ADMIN to send the packet of a specified SN, by set “Next FW packet SN” to its desired SN. For example, after receiving packet SN=1, the device might request ADMIN to send packet SN=100, by setting “Next FW packet SN” to 100.

Status Mode Code Data\_len Data bytes

00h if OK 05h 06h 02h Next FW packet SN

40h if NEXT\_IS\_LAST\_PACKET 05h 06h 02h Next FW packet SN

01h if fail 05h 06h 02h Next FW packet SN

There might be the case FW upgrade package transmission being interrupt because ADMIN remove PHONE from RF field accidentally. For ADMIN to recover the transmission, it first need to pass mutual authentication (either by AUTH mode Finish command or AUTH mode ValidatePIN command, with ActCode 05h) and then it can issue FWUPG with UPG action 3 to query the last received packet SN, and FW packet SN and FW packet length byte is ignored by DEVICE. Then ADMIN can continue the transmission by sending FWUPG command with UPG action byte set to 01h and the SN of the next packet.

The ADMIN can cancel previous transmitted FW upgrade packet and resend first packet by sending FWUPG command with UPG action byte set to 00h.

* + - 1. Layout of FW upgrade package

E-FW MAC 2 (32-byte)

E-FW MAC 1 (32-byte)

Control (4-byte)

Ver-ID (16-byte)

E-FWi size (4-byte)

Zero Padding (104-byte)

E-FW 1

E-FW 2

* + - 1. UPG action byte:

|  |  |
| --- | --- |
| UPG action | Description |
| 00h | Send first packet |
| 01h | Send remaining packets |
| 02h | Send the last packet |
| 03h | Query current packet SN |

* + 1. GetProperty:

The command code byte is 07h.

After ADMIN successfully issues SYNC, the ADMIN can issue GetProperty to read back device specified properties, such as FW version. PropertyID is 1-byte integer.

Mode Code Data\_len Data\_bytes

05h 07h 01h + N NUM\_OF\_PROPERTIES\_TO\_GET, {PropertyID\_0, PropertID\_1,

Data\_bytes (cont.)

PropertyID\_N-1}

The DEVICE will response the requested property data. PropertyLen is 1-byte integer to describe the data length of the property. For any unsupported property ID, the DataLen will be 0 and the DataBytes will be skipped.

Status Mode Code Data\_len Data bytes

00h if OK 05h 07h N + sum of all property data DataLen\_0, DataBytes\_0,

Data bytes (cont.)

DataLen\_1, DataBytes\_1, …, DataLen\_N-1, DataBytes\_N-1

* + - 1. Property ID and data size:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Data size | RW | Value |
| 00h | Mute | 1-byte | RW | 0/1 |
| 01h | Volume | 1-byte | RW | 0 ~ 255 |
| 02h | No Disturb Mode | 1-byte | RW | 0/1 |
| 03h | Channel Mode | 1-byte | RW | 0/1 |
| 04h | Double Authentication | 1-byte | RW | 0/1 |
| 05h | Time zone | 1-byte | RW |  |
| 06h | NetCode mode | 1-byte | RW | 1: STD multi & single  2: STD multi  3: STD single  4: URM  5: ACC |
| 07h | Activate built-in WIFI pairing | 1-byte | WO | 1 (One-shot) |
| 08h | Wifi machine | 1-byte | RW | Server idx. 0 is default. |
| 09h | Code client digit range | 1-byte | RO |  |
| 0Ah | Relock delay | 1-byte | RW | 2 ~ 30 |
| 0Bh | Activate gateway pairing | 1-byte | WO | 1 (One-shot) |
| 0Ch | Lock settings | 1-byte | RW | Bit 0: Lock/unlock status indicator  Bit 1: Lock down |
| 0Dh | Feature selection | 1-byte | RW | Bit 0: User code  Bit 1: Card  Bit 2: NetCode  Bit 3: Phone user |
| 0Eh | Gateway enable | 1-byte | RW | 0/1 |
| 0Fh | Day lockout | 1-byte | RW | MTWTFSS (Bit 0 ~ Bit 6) |
| 10h | Code client code length | 1-byte | RO | 4 ~ 16 |
| 11h | Touch Ignore Prefix | 1-byte | RW | 0/1 |
| 12h | WIFI enable | 1-byte | RW | 0/1 |
| 13h | Touch Random Digit | 1-byte | RW | 0/1 |
| 18h | Last get TID | 1-byte | RO | Valid SN value |
| 19h | Num of unread TIDs | 1-byte | RO |  |
| 20h | WIFI SSID | 32-byte maximum | RW |  |
| 21h | WIFI password | 32-byte maximum | WO |  |
| 22h | WIFI RSSI | 1-byte | RO | 0 ~ 255 |
| 23h | WIFI module status | 2-byte | RO | 2-byte reflected WIFI module status |
| 24h | WIFI enable (obsolete in the future) | 1-byte | RW | 0, 1 |
| 25h | Channel mode schedule | variable-byte | RW | 4-byte sequence\_number + 16-byte AR |
| 26h | Wifi enroll site ID | 8-byte | WO |  |
| 27h | Wifi enroll token | 8-byte | WO |  |
| 28h | Keypad illumination | 3-byte | RW | {mode, start\_hour, end\_hour} |
| 29h | Master code | 8-byte | WO |  |
| 2Ah | Sub-master Code | 8-byte | WO |  |
| 2Bh | Delete sub-master code | 1-byte | WO | 1 |
| 2Ch | Emergency open cancellation | 1-byte | WO | 1 (One-shot) |
| 2Dh | Block specific Netcode | 6-byte | RW | Netcode sequence |
| 2Eh | Odin protocol version | 2-byte | RO | {Start version, end version} |
| 2Fh | Set Guest-code Prefix | 3-byte | RW | Guest code prefix |
| 30h | Lock-unlock Status Indicator | 1-byte | RW |  |
| 31h | Lock down | 1-byte | RW |  |
| 32h | Clear Guest-code user code | 1-byte | RW | Write 1 to clear, and read 0 means no guest user code is registered while 1 means a guest user code is registered |
| 33h | Set Daylight Savings | 9-byte | RW | 1-byte DST hour, 4-byte start-time and 4-byte end-time |
| 34h | Delete Gateway | 1-byte | WO |  |
| 35h | Activate Gateway RSSI measurement | 1-byte | WO |  |
| 36h | Gateway BLE RSSI | 1-byte | RO |  |
| 37h | Gateway Model name | 4-byte | RO |  |
| 38h | Imp code version | 4-byte | RO |  |
| 39h | Imp device ID | 8-byte | RO |  |
| 3Ah | Imp MAC address | 6-byte | RO |  |
| 3Bh | Gateway FW version | 4-byte | RO |  |
| 3Ch | Activate gateway adding lock | 1-byte | WO |  |
| 3Dh | App-Gateway shared FID-MAC-Key | 16-byte | RO | FID-MAC-Key encrypted by admin’s DID-FID-Key |
| 3Eh | Battery status | 1-byte | RO |  |
| 3Fh | Reboot | 1-byte | WO |  |
| 40h | NetCode grace period | 1-byte | RW |  |
| 41h | Auto LR detect trigger | 1-byte | RW |  |
| 45h | REM sensor configuration | 1-byte | RW |  |
| 46h | One-time-code | variable |  |  |
| 47h | NetCode block policy | 1-byte | RW |  |
| 4Ch | First-man-in | 1-byte | RW | Enable/disable first man in |
| 80h | FW version | 16-byte | RO |  |

* + 1. ValidatePINAndSetPIN:

Obsolete.

* + 1. ValidatePINAndSetGIN:

Obsolete.

* + 1. AckSync

The command code byte value is 0Ah.

The command is used after the administrator issuing MANAGEMENT mode Sync or ValidatePINAndSync command, as the final acknowledgment to complete the log/client change synchronization. The first 16-byte command data is Response1, then there is ActCode. Then there is a 4-byte integer, LAST\_SYNC\_LOG\_SN, describing the last received log event record.

Note that the device will only check if ActCode value 06h is supplied, and other ActCode value is ignore. If there is ActCode value 06h, the device will keep the log event record. Otherwise, the log event records until LAST\_SYNC\_LOG\_SN+1 will be deleted.

Mode Code Data\_len Data\_bytes

05h 0Ah 15h Response1, ActCode, LAST\_SYNC\_LOG\_SN

The response is as follows.

Status Mode Code Data\_len Data\_bytes

00h if OK 05h 0Ah N/A N/A

01h if fails 05h 0Ah N/A N/A

* + 1. SetProperty:

The command code byte value is 0Bh.

After ADMIN successfully issues SYNC, the ADMIN can issue GetProperty to read back device specified properties, such as FW version. PropertyID is 1-byte integer.

Mode Code Data\_len Data\_bytes

05h 0Bh 01h + 2 x N + all property data NUM\_OF\_PROPERTIES\_TO\_SET,

Data\_bytes (cont.)

{PropertyID\_0, DataLen\_0, DataBytes\_0, PropertyID\_1, DataLen\_1, DataBytes\_1, … ,

Data\_bytes (cont.)

PropertyID\_N-1, DataLen\_N-1, DataBytes\_N-1}

The DEVICE will response the requested property data. Property\_Status is 1-byte integer to describe whether the property set activity is successful.

Status Mode Code Data\_len Data bytes

00h if OK 05h 0Bh N Property\_Status\_0, Property\_Status\_1,

Data bytes (cont.)

Property\_Status\_N-1

Property\_Status code explanation:

|  |  |
| --- | --- |
| Property Status Code | Description |
| 00h | OK |
| 01h | Invalid ID |
| 02h | Value out of range |
| FFh | Other errors |

* + 1. SetDeviceName:

The command code byte value is 0Ch.

Only valid after ADMIN issuing MANAGEMENT mode SYNC command successfully. Only ADMIN can issues this command to change the DEVICE’s name.

Mode Code Data\_len Data\_bytes

05h 0Ch 10h Device\_name

The response contains status code 00h if the client’s new UserName has been successfully updated to the DEVICE.

Status Mode Code Data\_len Data\_bytes

00h if OK 05h 0Ch N/A N/A

01h if NG 05h 0Ch N/A N/A

* + 1. AddPasswordClient:

The command code byte value is 0Dh.

Only valid after ADMIN issuing MANAGEMENT mode SYNC command successfully. The command data contains 4-byte sequence number, 16-byte FID, 1st\_mask, 1st\_byte, 2nd\_mask, 2nd\_byte, AR portion whose size depends on the AR version, 1-byte Password\_len, password data payload whose size limitation is defined by the individual lock model, 1-byte Username\_len, and maximum 16-byte Username.

Mode Code Data\_len Data\_bytes

05h 0Dh 18h + A + P + U Sequence\_number, FID, 1st\_mask, 1st\_byte,

Data\_bytes (cont.)

2nd\_mask, 2nd\_byte, AR\_portion, Password\_len, Password, Username\_len, Username

The response contains status code 00h if the new password client has been added successfully.

Status Mode Code Data\_len Data\_bytes

00h if OK 05h 0Dh N/A N/A

01h if add fails 05h 0Dh N/A N/A

* + 1. SyncByFID:

The command code byte value is 0Eh.

Only supports in protocol version 8007 and newer.

Command format:

Command data contains 4-byte START\_LOG\_SN, 1-byte MAX\_READ\_NBR, 4-byte LAST\_SYNC\_LOG\_SN, and 1-byte Num\_Of\_clients. Then there are multiple sets of sync data. Each set of sync data must contain at least 2-byte SN, 16-byte FID, 1-byte 1st\_Mask, 1-byte 1st\_ctrl\_byte, 1-byte 2nd\_Mask, 1-byte 2nd\_ctrl\_byte, 4-byte Seq\_number. Depending on the value of mask and ctrl bytes, each set of sync data can contains additional data. If Uname\_bit is 1 in 1st\_mask and 1st\_ctrl\_byte, Uname portion must exist. If AR\_bit is 1 in 1st\_mask and 1st\_ctrl\_byte, AR portion must exist. If LML\_bit is 1 in 2nd\_mask and 2nd\_ctrl\_byte, LML time portion must exist.

Mode Code Data\_len Data\_bytes

05h 0Eh Variable Response1, ActCode,

Data\_bytes (cont.)

START\_LOG\_SN, MAX\_READ\_NBR, LAST\_SYNC\_LOG\_SN, Num\_Of\_Clients,

Data\_bytes (cont.)

{SN\_0, FID\_0, 1st\_Mask\_0, 1st\_ctrl\_byte\_0, 2nd\_mask\_0, 2nd\_ctrl\_byte\_0, Seq\_number\_0,

Data\_bytes (cont.)

{TID\_Key\_portion}, {AR\_portion\_0},…

Data\_bytes (cont.)

{SN\_1, FID\_1, … }, … {SN\_n-1, FID\_n-1, …}

TID key portion:

16-byte DID-TID-Key, optional only for deleting an OTA client, should appear only when

AR portion:

{AR\_version, AR\_len, AR\_data, AR\_TS}

AR\_version is 1-byte.

AR\_len is 1-byte.

AR\_data is multiple bytes.

AR TS is 4-byte.

AR portion format for AR version 0 (one 16-byte-AR):

{00h, 10h, 16-byte AR\_data, AR\_TS}

AR portion format for AR version 1 (Up to ten 4-byte-AR):

{01h, 02h + 4h \* i, 1st\_AR\_bitmsk, 2nd\_AR\_bitmsk, AR\_0, AR\_1, …, AR\_i-1, AR\_TS}

1st\_AR\_bitmsk:

Controlling whether to update the corresponding AR

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Bit 0 | Bit 1 | Bit 2 | Bit 3 | Bit 4 | Bit 5 | Bit 6 | Bit 7 |
| AR 0 | AR 1 | AR 2 | AR 3 | AR 4 | AR 5 | AR 6 | AR 7 |

2nd\_AR\_bitmsk:

Controlling whether to update the corresponding AR

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Bit 0 | Bit 1 | Bit 2 | Bit 3 | Bit 4 | Bit 5 | Bit 6 | Bit 7 |
| AR 8 | AR 9 | RFU | RFU | RFU | RFU | RFU | RFU |

AR portion format for AR version 2 (Up to three 4-byte-AR):

{01h, 02h + 4h \* i, 1st\_AR\_bitmsk, 2nd\_AR\_bitmsk, AR\_0, AR\_1, …, AR\_i-1, AR\_TS}

1st\_AR\_bitmsk:

Controlling whether to update the corresponding AR

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Bit 0 | Bit 1 | Bit 2 | Bit 3 | Bit 4 | Bit 5 | Bit 6 | Bit 7 |
| AR 0 | AR 1 | AR 2 | RFU | RFU | RFU | RFU | RFU |

2nd\_AR\_bitmsk:

Controlling whether to update the corresponding AR

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Bit 0 | Bit 1 | Bit 2 | Bit 3 | Bit 4 | Bit 5 | Bit 6 | Bit 7 |
| RFU | RFU | RFU | RFU | RFU | RFU | RFU | RFU |

Response format:

The first byte of response data is num\_of\_clients which indicates the number of clients to sync from the lock to admin in this response. The second byte is num\_of\_log which indicates the number of log to sync from the lock to admin in this response. The 3rd byte is START\_LOG\_SN\_Offset, which indicates the byte offset related the start of response data. It can be used as a quick reference to START\_LOG\_SN byte, which is the first byte of log portion. Starting from the 4th byte is multiple sets the client data to sync from lock to admin. Each sets of data has the same format as in command packet. Then there are 4-byte START\_LOG\_SN, and the log records to sync. Each log record is at least 7-byte.

Status Mode Code Data\_len Data bytes

00h if OK 05h 0Eh Variable num\_of\_clients, num\_of\_log, START\_LOG\_SN\_Offset

Data\_bytes (cont.)

{SN\_0, FID\_0, 1st\_Mask\_0, 1st\_Ctrl\_byte\_0, 2nd\_Mask\_0, 2nd\_Ctrl\_byte\_0, Seq\_number\_0,

Data\_bytes (cont.)

{Uname\_portion\_0}, {AR\_portion\_0}, {LML\_time\_portion\_0}…

Data\_bytes (cont.)

{SN\_1, FID\_1, … }, … {SN\_Nc-1, FID\_Nc-1, …}

Data\_bytes (cont.)

{START\_LOG\_SN, LOG\_record\_0, LOG\_record\_1, … LOG\_record\_Nl-1}

LML time portion:

{time}

Time is 4-byte UTC second

LOG\_RECORD format:

{DID-FID-SN, Event\_Code, Time, {Additional\_data}}

DID-FID-SN is 2-byte.

Event\_code is 1-byte.

Time is 4-byte.

Additional data only exists in some special log which is used to convey FID and name. See Section 3.12.

* + 1. SetPropertyWithSeq:

The command code byte value is 0Fh.

Only supports in protocol version 8007 and newer.

PropertyID is 1-byte. Seq\_number is 4-byte.

Mode Code Data\_len Data\_bytes

05h 0Fh 05h + 2 x N + all property data NUM\_OF\_PROPERTIES\_TO\_SET, Seq\_number,

Data\_bytes (cont.)

{PropertyID\_0, DataLen\_0, DataBytes\_0, PropertyID\_1, DataLen\_1, DataBytes\_1, … ,

Data\_bytes (cont.)

PropertyID\_N-1, DataLen\_N-1, DataBytes\_N-1}

The DEVICE will response the requested property data. Property\_Status is 1-byte integer to describe whether the property set activity is successful.

Status Mode Code Data\_len Data bytes

00h if OK 05h 0Fh N Property\_Status\_0, Property\_Status\_1,

Data bytes (cont.)

Property\_Status\_N-1

* + 1. AddPasswordClient\_Enc:

The command code byte value is 10h.

Only valid after ADMIN issuing MANAGEMENT mode SYNC command successfully. The command data contains Response1 and encrypted password client information which is padded to multiple of 16. The encrypted content are 4-byte sequence number, 16-byte FID, 1st\_mask, 1st\_byte, 2nd\_mask, 2nd\_byte, AR portion whose size depends on the AR version, 1-byte Password\_len, password data payload whose size limitation is defined by the individual lock model, 1-byte Username\_len, and maximum 16-byte Username.

Mode Code Data\_len Data\_bytes

05h 10h Variable (Multiple of 16) Response 1, AES\_ECB\_Enc (input =

Data\_bytes (cont.)

{Sequence\_number, FID, 1st\_mask, 1st\_byte, 2nd\_mask, 2nd\_byte, AR\_Portion,

Data\_bytes (cont.)

Password\_len, password, Username\_len, Username, CRC, padding}, key = DID-FID-Key)

The response contains status code 00h if the new password client has been added successfully.

Status Mode Code Data\_len Data\_bytes

00h if OK 05h 10h N/A N/A

01h if add fails 05h 10h N/A N/A

* + 1. CloneCardClient\_Enc:

The command code byte value is 10h.

Only valid after ADMIN issuing MANAGEMENT mode SYNC command successfully. The command data contains Response1 and encrypted password client information which is padded to multiple of 16. The encrypted content are 4-byte sequence number, 16-byte FID, 1st\_mask, 1st\_byte, 2nd\_mask, 2nd\_byte, AR portion whose size depends on the AR version, 1-byte Username\_len, and maximum 16-byte Username.

Mode Code Data\_len Data\_bytes

05h 11h Variable (Multiple of 16) Response 1, AES\_ECB\_Enc (input =

Data\_bytes (cont.)

{Sequence\_number, FID, 1st\_mask, 1st\_byte, 2nd\_mask, 2nd\_byte, AR\_Portion,

Data\_bytes (cont.)

Username\_len, Username, CRC, padding}, key = DID-FID-Key)

The response contains status code 00h if the new password client has been added successfully.

Status Mode Code Data\_len Data\_bytes

00h if OK 05h 11h N/A N/A

01h if add fails 05h 11h N/A N/A

* + 1. SyncByFID\_NewLog\_Enc:

The command code byte value is 12h.

Only supports in protocol version 8011 and newer.

Command format:

Command data contains 4-byte START\_LOG\_SN, 1-byte MAX\_READ\_NBR, 4-byte LAST\_SYNC\_LOG\_SN, and 1-byte Num\_Of\_clients. Then there are multiple sets of sync data. Each set of sync data must contain at least 2-byte SN, 16-byte FID, 1-byte 1st\_Mask, 1-byte 1st\_ctrl\_byte, 1-byte 2nd\_Mask, 1-byte 2nd\_ctrl\_byte, 4-byte Seq\_number. Depending on the value of mask and ctrl bytes, each set of sync data can contains additional data. If Uname\_bit is 1 in 1st\_mask and 1st\_ctrl\_byte, Uname portion must exist. If AR\_bit is 1 in 1st\_mask and 1st\_ctrl\_byte, AR portion must exist. If LML\_bit is 1 in 2nd\_mask and 2nd\_ctrl\_byte, LML time portion must exist.

Mode Code Data\_len Data\_bytes

05h 12h Variable Response1, ActCode,

Data\_bytes (cont.)

START\_LOG\_SN, MAX\_READ\_NBR, LAST\_SYNC\_LOG\_SN, Num\_Of\_Clients,

Data\_bytes (cont.)

{SN\_0, FID\_0, 1st\_Mask\_0, 1st\_ctrl\_byte\_0, 2nd\_mask\_0, 2nd\_ctrl\_byte\_0, Seq\_number\_0,

Data\_bytes (cont.)

{TID\_Key\_portion}, {AR\_portion\_0},…

Data\_bytes (cont.)

{SN\_1, FID\_1, … }, … {SN\_n-1, FID\_n-1, …}

TID key portion:

16-byte DID-TID-Key, optional only for deleting an OTA client, should appear only when

AR portion:

{AR\_version, AR\_len, AR\_data, AR\_TS}

AR\_version is 1-byte.

AR\_len is 1-byte.

AR\_data is multiple bytes.

AR TS is 4-byte.

AR portion format for AR version 0 (one 16-byte-AR):

{00h, 10h, 16-byte AR\_data, AR\_TS}

AR portion format for AR version 1 (Up to ten 4-byte-AR):

{01h, 02h + 4h \* i, 1st\_AR\_bitmsk, 2nd\_AR\_bitmsk, AR\_0, AR\_1, …, AR\_i-1, AR\_TS}

1st\_AR\_bitmsk:

Controlling whether to update the corresponding AR

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Bit 0 | Bit 1 | Bit 2 | Bit 3 | Bit 4 | Bit 5 | Bit 6 | Bit 7 |
| AR 0 | AR 1 | AR 2 | AR 3 | AR 4 | AR 5 | AR 6 | AR 7 |

2nd\_AR\_bitmsk:

Controlling whether to update the corresponding AR

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Bit 0 | Bit 1 | Bit 2 | Bit 3 | Bit 4 | Bit 5 | Bit 6 | Bit 7 |
| AR 8 | AR 9 | RFU | RFU | RFU | RFU | RFU | RFU |

Response format:

The first byte of response data is num\_of\_clients which indicates the number of clients to sync from the lock to admin in this response. The second byte is num\_of\_log which indicates the number of log to sync from the lock to admin in this response. The 3rd byte is START\_LOG\_SN\_Offset, which indicates the byte offset related the start of response data. It can be used as a quick reference to START\_LOG\_SN byte, which is the first byte of log portion. Starting from the 4th byte is multiple sets the client data to sync from lock to admin. Each sets of data has the same format as in command packet. Then there are 4-byte START\_LOG\_SN, and the log records to sync. Each log record is at least 7-byte.

Status Mode Code Data\_len Data bytes

00h if OK 05h 12h Variable num\_of\_clients, num\_of\_log, START\_LOG\_SN\_Offset

Data\_bytes (cont.)

{SN\_0, FID\_0, 1st\_Mask\_0, 1st\_Ctrl\_byte\_0, 2nd\_Mask\_0, 2nd\_Ctrl\_byte\_0, Seq\_number\_0,

Data\_bytes (cont.)

{Uname\_portion\_0}, {AR\_portion\_0}, {LML\_time\_portion\_0}…

Data\_bytes (cont.)

{SN\_1, FID\_1, … }, … {SN\_Nc-1, FID\_Nc-1, …}

Data\_bytes (cont.)

{ AES\_ECB\_Enc (input = START\_LOG\_SN, LOG\_record\_new\_0, LOG\_record\_new\_1,

Data\_bytes (cont.)

… LOG\_record\_new\_Nl-1, padding, key = DID-FID-key)}

LML time portion:

{time}

Time is 4-byte UTC second

LOG\_RECORD\_NEW format:

{DID-FID-SN, Ext\_Flag, Event\_Code, Time, {Additional\_data}}

DID-FID-SN is 2-byte.

Ext\_flag is 1-byte. 0th bit indicates the command initiator. 0: Phone. 1: Web.

Event\_code is 1-byte.

Time is 4-byte.

Additional data only exists in some special log which is used to convey FID and Netcode/Varicode. See Section 3.12.

* 1. Command mode HOTEL

The command mode byte value is 0Eh. The command is not supported by normal app for Odin and is only valid for Dion Hotel Staff App.

* + 1. ExecStaffCmd

The command code byte value is 01h.

Staff App can issue the command to execute STAFF-CMD provided by PC admin. The command data can contains 20-byte E\_CMD or 40-byte E\_CMD.

Mode Code Data\_len Data\_bytes

0Eh 01h 20h or 40h E\_CMD

The response status code is 00h if the STAFF-CMD is executed successfully. The response data contains 1-byte CMD\_TYPE\_CODE, 4-byte SEQNUM, 1-byte RES\_LEN which indicates the length of RES\_DATA, and variable-byte RES\_DATA.

Status Mode Code Data\_len Data\_bytes

00h if OK 0Eh 01h 05h or 06h + n CMD\_TYPE\_CODE,

Data\_bytes (cont.)

SEQNUM, RES\_LEN, RES\_DATA

The other status code values upon error:

01h if invalid command

02h if integrity check fail

03h if DID check fail

04h if SEQNUM check fail

* + - 1. Table of STAFF-CMD

|  |  |  |  |
| --- | --- | --- | --- |
| STAFF-CMD | CMD\_TYPE\_CODE | E\_CMD  length | Constraint |
| DEPLOY | 00h | 40h | Only accepted if the lock is in SETUP mode |
| GETTID | 01h | 20h |  |
| DELCLIENT | 02h | 20h |  |
| GETLOG | 03h | 20h |  |
| SETCLOCK | 04h | 20h |  |
| GETBAT | 05h | 20h |  |

* + - 1. RES\_DATA format

|  |  |  |
| --- | --- | --- |
| STAFF-CMD | RES\_DATA Format | Remarks |
| DEPLOY | N/A |  |
| GETTID | E\_TID\_0, E\_TID\_1, … , E\_TID\_n-1 | Each E\_TID\_x: 20h-byte |
| DELCLIENT | N/A |  |
| GETLOG | Start\_Log\_SN, Log\_0, Log\_1, …, Log\_n-1 | Start\_log\_SN: 4-byte  Each Log\_x: 6-byte |
| SETCLOCK | N/A |  |
| GETBAT | Bat\_status | Bat\_status: 1-byte |

* + 1. GetDID

The command code byte value is 02h.

Staff App can issue the command to query the DID of a specified lock. The command is accepted only when the lock is in SETUP mode.

Mode Code Data\_len Data\_bytes

0Eh 02h N/A N/A

The response contains status code 00h if it is executed successfully.

Status Mode Code Data\_len Data\_bytes

00h if OK 0Eh 02h 10h DID

01h if cannot execute (not in SETUP) 0Eh 02h N/A N/A

* + 1. Abort

The command code byte value is 01h.

Mode Code Data\_len Data\_bytes

0Eh 03h N/A N/A

The response contains status code 00h if it is executed successfully.

Status Mode Code Data\_len Data\_bytes

00h if OK 0Eh 03h N/A N/A

01h if fail 0Eh 03h N/A N/A

* 1. Command mode GATEWAY

The command mode byte value is 0Dh. The command is not supported by App and only valid for gateway.

* + 1. Introduce:

The command code byte value is 01h.

The gateway issues the command to introduce its existence to the lock, with its Wifi module’s version and MAC address, and the LOCK will report its DID.

Mode Code Data\_len Data\_bytes

0Dh 01h 2 + m Wifi Version, Wifi MAC length, Wifi MAC address

The response contains status code 00h if it is executed successfully.

Status Mode Code Data\_len Data\_bytes

00h if OK 0Dh 01h 16 DID

01h if fail 0Dh 01h 16 DID

* + 1. StartPairingReq:

The command code byte value is 02h.

The gateway will initiate the command if the lock is advertising the GW\_PAIRING flag. The command data contains 1-byte Version, and Wifi module’s MAC length, and m-byte MAC address.

Mode Code Data\_len Data\_bytes

0Dh 02h 2 + m Wifi Version, Wifi MAC length, Wifi MAC address

The response contains status code 00h if it is executed successfully. The lock will send 16-byte DID, 16-byte FID, 4-byte Rolling\_nbr, 16-byte Rand, 4-byte Time, 16-byte Hash\_Key and the above are used in claim process. Then 8-byte SiteID and 8-byte Token which are essential for enroll process. Then there 1-byte SSID\_len which describes the length of multiple byte SSID and 1-byte PWD\_len which describes the length of multiple byte PWD. Then there is 32-byte scrambled (DID-MAC-Key, Random) content. Then there is 32-byte scrambled (FID-MAC-Key, Random\_2) hash content. Then 1-byte server idx to indicate which server the gateway agent should be connected to. If server idx byte does not exists by command data payload length checking, the gateway should connect to default server (K3).

Status Mode Code Data\_len Data\_bytes

00h if OK 0Dh 02h 7Bh + s + p +{20h} DID, FID, Rolling\_nbr, Rand, Time,

Data\_bytes (cont.)

Hash\_Key, SiteID, Token, SSID\_len, SSID, Pwd\_len, Pwd, S2(DID-MAC-Key, Random), Random,

Data\_bytes (cont.)

S2(FID-MAC-Key, Random\_2), Random\_2, Server\_idx

01h if lock not sup 0Dh 02h N/A N/A

02h if ver/cap err 0Dh 02h N/A N/A

* + 1. FinishPairingReq:

The command code byte value is 03h.

The claim\_status will at least contains CLAIM\_PENDING bit. The claim result will be further report by NotifyPairingResult.

Mode Code Data\_len Data\_bytes

0Dh 03h 1 Claim\_status

The response contains status code 00h if it is executed successfully.

Status Mode Code Data\_len Data\_bytes

00h if OK 0Dh 03h N/A N/A

01h if fail 0Dh 03h N/A N/A

* + - 1. Table: Claim status

|  |  |
| --- | --- |
| Bit of Claim status | Meaning |
| 0 | SSID/PWD Pending |
| 1 | Claim Pending |
| 2 | SSID/PWD Error |
| 3 | Claim error |
| 4 | Wifi error |
| 5 | Cloud error |
| 6 | Other error |
| 7 | Bad gateway model error |

* + 1. Auth:

The command code byte value is 04h.

Mode Code Data\_len Data\_bytes

0Dh 04h 11h + m Challenge\_2, Wifi\_MAC\_len, Wifi\_MAC\_addr

The response contains status code 00h if it is executed successfully.

Status Mode Code Data\_len Data\_bytes

00h if OK 0Dh 04h 20h Response\_2, Challenge\_1

01h if fail 0Dh 04h N/A N/A

* + 1. NotifyPairingResult:

The command code byte value is 05h.

The gateway should notify the pairing result. The CLAIM\_PENDING bit should be 0 in claim\_status byte. Other error bits will be set possibly. If the BAD\_GATEWAY\_MODEL error bit exists in Claim\_status, at least gateway model name parameter (ID 04h) will be put in the response data bytes.

Mode Code Data\_len Data\_bytes

0Dh 05h 11h Response1, Claim\_status

The response contains status code 00h if it is executed successfully.

Status Mode Code Data\_len Data\_bytes

00h if OK 0Dh 05h N/A N/A

01h if fail 0Dh 05h N/A N/A



* + 1. Exchange:

The command code byte value is 06h.

The gateway issues this command to send downlink data to lock and receive uplink data from lock. The gateway should issue this command if GW\_DATA flag is set or GW\_BUSY flag is not set by the lock.

The command data contains 16-byte Response1 and 1-byte type\_status which describes two kinds of information: the result of the uplink/get param in previous Exchange command and the types of data (downlink data or get param data) to transmit in this exchange command. Then the whole downlink data or param data.

An Exchange command also acts as the acknowledgement of successfully processing previous Exchange command (i.e. the uplink data is considered sent to cloud successfully).

type\_status Desc

Bit 0 Prev uplink result. 0 means prev uplink succ, 1 means prev uplink fail

Bit 4 Data type. 0 means downlink data transmission, 1 means sending param

Mode Code Data\_len Data\_bytes

0Dh 06h 11h + p Response1, type\_status, Downlink\_data

Mode Code Data\_len Data\_bytes

0Dh 06h 11h + p Response1, type\_status, param\_0\_id, param\_0\_len, param\_0\_data,

Data\_bytes (cont.)

param\_1\_id, param\_1\_len, param\_1\_data, … param\_n-1\_id, param\_n-1\_len, param\_n-1\_data

The response status code and data is defined as follows. Bit 0 ~ 3 of status code describe the command exec status

Status (Bit 0 ~ 3) Desc

00h OK

01h Fail

02h Time slot expire

Bit 4 ~ 7 of status code

Status (Bit 4 ~ 7) Desc

Bit 4 The lock contains more data to uplink; hint next Exchange required

Bit 5 The lock request get\_param; the param to get is indicated at payload

Bit 6 The lock request set\_param; the param to set is indicated at payload

Bit 0 ~ 3 of status code and it 4 ~ 7 of status code, the following status code value is possible. Data byte payload for each case is depicted.

Status Mode Code Data\_len Data\_bytes

00h 0Dh 06h p Uplink\_data

01h 0Dh 06h N/A N/A

02h 0Dh 06h p Uplink\_data

10h 0Dh 06h p Uplink\_data

20h 0Dh 06h variable Num\_of\_param, param\_0\_id, …, param\_n-1\_id

40h 0Dh 06h p Num\_of\_param, param\_0\_id, param\_0\_len, param\_0\_data, …, param\_n-1\_id, param\_n-1\_len, param\_n-1\_data

22h 0Dh 06h variable Num\_of\_param, param\_0\_id, …, param\_n-1\_id

42h 0Dh 06h p Num\_of\_param, param\_0\_id, param\_0\_len, param\_0\_data, …, param\_n-1\_id, param\_n-1\_len, param\_n-1\_data

30h 0Dh 06h variable Num\_of\_param, param\_0\_id, …, param\_n-1\_id

50h 0Dh 06h p Num\_of\_param, param\_0\_id, param\_0\_len, param\_0\_data, …, param\_n-1\_id, param\_n-1\_len, param\_n-1\_data

Here is the table of possible Params:

|  |  |  |  |
| --- | --- | --- | --- |
| Param | Param ID | Length (in bytes) | Get/Set |
| Wifi test machine | 00h | 1 | G/S |
| Gateway enable | 01h | 1 | G/S |
| Wifi RSSI | 02h | 1 | G |
| BLE RSSI | 03h | 1 | G |
| Gateway model name | 04h | 4 | G |
| Imp (Gungnir) code version | 05h | 4 | G |
| Imp device ID | 06h | 8 | G |
| Imp MAC address | 07h | 6 | G |
| Internet time | 08h | 4 | G |

* + 1. Ack:

The command code byte value is 0Ah.

The command is used by the gateway to notify it has processed the uplink data in previous response of Exchange command. If the last Exchange command is not followed by Ack command, the lock will consider that the gateway does not successfully handle the last uplink and will attempt to resend identical uplink data in the response of next Exchange command.

Mode Code Data\_len Data\_bytes

0Dh 0Ah N/A N/A

The response contains status code 00h if it is executed successfully.

Status Mode Code Data\_len Data\_bytes

00h if OK 0Dh 0Ah N/A N/A

* + 1. StartAddingReq:

The command code byte value is 0Bh.

The gateway will initiate the command if the lock is advertising the GW\_ADDING flag. The command data contains 1-byte Version, and Wifi module’s MAC length, and m-byte MAC address.

Mode Code Data\_len Data\_bytes

0Dh 0Bh 2 + m Wifi Version, Wifi MAC length, Wifi MAC address

The response contains status code 00h if it is executed successfully. The lock will send 16-byte DID, 16-byte FID, 4-byte Rolling\_nbr, 16-byte Rand, 4-byte Time, 16-byte Hash\_Key and the above are used in server-side add process. Then there is 32-byte scrambled (FID-MAC-Key, Random\_2) hash content. Then 1-byte server idx to indicate which server the gateway agent should be connected to. If server idx byte does not exists by command data payload length checking, the gateway should connect to default server (K3).

Status Mode Code Data\_len Data\_bytes

00h if OK 0Dh 0Bh 69h DID, FID, Rolling\_nbr, Rand, Time, Hash\_Key,

Data\_bytes (cont.)

S2(DID-MAC-Key, Random), Random, Server\_idx

01h if lock not sup 0Dh 0Bh N/A N/A

02h if ver/cap err 0Dh 0Bh N/A N/A

* + 1. FinishAddingReq:

The command code byte value is 0Ch.

The add\_status will at least contains ADD\_PENDING bit. The add result will be further report by NotifyAddingResult.

Mode Code Data\_len Data\_bytes

0Dh 0Ch 1 Add\_status

The response contains status code 00h if it is executed successfully.

Status Mode Code Data\_len Data\_bytes

00h if OK 0Dh 0Ch N/A N/A

01h if fail 0Dh 0Ch N/A N/A

* + 1. NotifyAddingResult:

The command code byte value is 0Dh.

The gateway should notify the lock adding result. The ADD\_PENDING bit should be 0 in add\_status byte. Other error bits will be set possibly. If the BAD\_GATEWAY\_MODEL error bit exists in Claim\_status, at least gateway model name parameter (ID 04h) will be put in the response data bytes.

Mode Code Data\_len Data\_bytes

0Dh 0Dh 11h Response1, Add\_status

The response contains status code 00h if it is executed successfully.

Status Mode Code Data\_len Data\_bytes

00h if OK 0Dh 0Dh N/A N/A

01h if fail 0Dh 0Dh N/A N/A

* + 1. Scenario
       1. Lock want get 2 one-byte parameters from gateway

*GATEWAY LOCK*

*(GATEWAY, Auth, 17h*

*10h-byte Challenge\_2,*

*06h,*

*06h-byte Mac)* 🡪

*🡨 (00h, GATEWAY, Auth, 20h,*

*10h-byte Response\_2,*

*10h-byte Challenge\_1)*

*(GATEWAY, Exchange, 11h,*

*10h-byte response\_1,*

*00h)* 🡪

*🡨 (40h, GATEWAY, Exchange, 03h,*

*02h,*

*1-byte param\_id\_0, 1-byte param\_id\_1)*

*(GATEWAY, Exchange, 18h,*

*10h-byte response\_1,*

*10h,*

*02h,*

*1-byte param\_id\_0, 1-byte parram\_len\_0, param\_data\_0,*

*1-byte param\_id\_1, 1-byte parram\_len\_1, param\_data\_1)*

🡪

*🡨 (00h, GATEWAY, Exchange, 00h)*

*(GATEWAY, Ack)* 🡪

*🡨 (00h, GATEWAY, Ack)*

* + - 1. Lock wants to get one 4-byte parameter from gateway but gateway need to process a downlink just before responding the parameter

*GATEWAY LOCK*

*(GATEWAY, Auth, 17h*

*10h-byte Challenge\_2,*

*06h,*

*06h-byte Mac)* 🡪

*🡨 (00h, GATEWAY, Auth, 20h,*

*10h-byte Response\_2,*

*10h-byte Challenge\_1)*

*(GATEWAY, Exchange, 11h,*

*10h-byte response\_1,*

*00h)* 🡪

*🡨 (40h, GATEWAY, Exchange, 02h,*

*01h,*

*1-byte param\_id\_0)*

*(GATEWAY, Exchange, 51h,*

*10h-byte, response,*

*00h,*

*40h-byte Downlink data)* 🡪

*🡨 (40h, GATEWAY, Exchange, 02h,*

*01h,*

*1-byte param\_id\_0)*

*(GATEWAY, Exchange, 18h,*

*10h-byte response,*

*10h,*

*01h,*

*1-byte param\_id\_0, 1-byte parram\_len\_0, param\_data\_0)*

🡪

*🡨 (00h, GATEWAY, Exchange, 00h)*

*(GATEWAY, Ack)* 🡪

*🡨 (00h, GATEWAY, Ack)*

* + - 1. Lock wants to set a 1-byte parameter to gateway

*GATWAY LOCK*

*(GATEWAY, Auth, 17h*

*10h-byte Challenge\_2,*

*06h,*

*06h-byte Mac)* 🡪

*🡨 (00h, GATEWAY, Auth, 20h,*

*10h-byte Response\_2,*

*10h-byte Challenge\_1)*

*(GATEWAY, Exchange, 11h,*

*10h-byte response\_1,*

*00h)* 🡪

*🡨 (20h, GATEWAY, Exchange, 05h,*

*03h,*

*01h*

*1-byte param\_id\_0,*

*1-byte param\_0\_len,*

*Param\_0\_data)*

*(GATEWAY, Ack)* 🡪

*🡨 (00h, GATEWAY, Ack)*

* 1. Command mode FACTORY

The command mode byte value is 0Fh. The command is not supported by normal app for Odin and is only valid for FACTORY app.

* + 1. SendRequest:

The command code byte value is 01h.

PHONE issues the command to start set-level FACTORY test procedure request with necessary data. It can only be issued by PHONE after DEVICE’s setup button being pressed. The command format is as follows, where DIN, DN, and FID are of 16 bytes, and Time is 4-byte current time, and USERNAME is no greater than 16 bytes. Then 16-byte App\_key\_seed and 16-byte Challenge. Then there is 2-byte model name which will set to device. Then 1-byte UID length and UID bytes which is the golden card UID which will be used in the test procedure.

Mode Code Data\_len Data\_bytes

0Fh 01h 67h + UID length DIN, DN, FID, Time, USERNAME, App\_key\_seed,

Data\_bytes (cont.)

Challenge, Model\_name, UID\_Length, UID

The response format is as follows. The DEVICE’s 16-byte DID, the 16-byte scrambled output of the DID-FID-Key and a random number, denoted by S2(DID-FID-Key, Random), the 16-byte random number itself, and the 2-byte DID-FID-SN are sent back in response data bytes. Then there is a 2-byte integer, M, which indicates the number of log event record contained in the response data. The 4-byte long START\_LOG\_SN is the serial number of the first log event record sent back. Then there are variable byte of log event record data. Then there is 16-byte FW\_Version and 16-byte SHA256-HMAC response calculated by App\_key and the challenge sent by PHONE. Then there is the 4-byte admin\_rolling\_number. Finally there is 16-byte DID-FID-Time-Hash used for claiming lock from Asgard server.

Status Mode Code Data\_len Data\_bytes

00h if OK 0Fh 01h 72h DID, S2(DID-FID-Key, Random), Random, DID-FID-SN,

Data\_bytes (cont.)

M, START\_LOG\_SN, DID-FID-SN\_Log\_0, LogEvent\_0, Time\_0, FW\_Version,

Data\_bytes (cont.)

SHA256-HMAC(App\_key, Challenge), Admin-rolling-number, DID-FID-Time-Hash

If some error happens, the status code is as follows and the response data is not used.

Status Mode Code Data\_len Data\_bytes

01h if misc fails 01h 01h N/A N/A

06h if device not in setup 01h 01h N/A N/A

07h if DIN check error 01h 01h N/A N/A

* + 1. Finish:

The command code byte value is 02h.

PHONE can issue the command to notify DEVICE the completion of reading necessary response data of a former SendReqeust command and all FACTORY entering procedure successfully performed.

Mode Code Data\_len Data\_bytes

0Fh 02h N/A N/A

The response of the command might be skipped by the PHONE. The PHONE can use UpdateBinary command’s response SW1 and SW2 bytes in ISO14443-4 layer to tell if the command succeeds.

Status Mode Code Data\_len Data\_bytes

00h if OK 0Fh 02h N/A N/A

01h if FACTORY entrance fails 0Fh 02h N/A N/A

* 1. ActCode Byte Format

The ActCode is 1-byte and it is composed of 6-bit act value and 2-bit extra act indicator.

|  |  |
| --- | --- |
| Bit 6 and 7 | Bit 0 ~ 5 |
| Extra Act Ind | Act Value |

* + 1. ActCode Act Value:

|  |  |
| --- | --- |
| ActCode Act Value  (Bit 0 ~ 5) | Description |
| 00h | Unlock door |
| 01h | Execute IPA flow |
| 02h | Unlock door with temp relock delay |
| 05h | MANAGEMENT activity |
| 06h | Sync only |
| 07h | Execute IPD flow |

* + 1. The above ActCode value only occupies bit 0 ~ bit 5, and bit 6 and bit 7 is extra act indicator.

Bit 6: Force BLE (if the lock has BLE interface) to stay at fast advertise for a while after this BLE connection is closed to allow app to get necessary PIDs.

* 1. Client List 1st Entry Control Byte:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| AccessRight | RFU | CARD | TID | PWD | SUS | OT | VALID |

**VALID** bit: Indicates whether the client is valid. Administrator can clear a client’s VALID bit to delete it from client list.

**OT** bit:In protocol version 8007 or newer, this bit can be used to set a client as one-time user.

**SUS** bit:In protocol version 8007 or newer, this bit can be set by admin to explicitly suspend the client.

**PWD** bit: Indicates whether the client is password client.

**TID** bit: Indicates whether the client is a TID user.

**CARD** bit: Indicates whether the client is a card.

**AccessRight** bit: Indicates change on the access right of a client.

* 1. Client List 2nd Entry Control Byte:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| RFU | RFU | TIDKEY | USERNAME | LML | TECH | EXP | INHERIT |

**INHERIT** bit: Indicates whether the client is inherited.

**EXP** bit: Indicates whether the client’s access right is expired. In protocol 8006, this bit will be set to 1 only if the client’s access right is set to one-time and the client has been accessed the door (one-time access right expired).

**TECH** bit: Indicates whether the client is a technician user.

**LML** bit: Indicates whether the last modification to the client is done by lock’s local operation

**USERNAME** bit: Indicates change on client’s username.

**TIDKEY bit:** Indicates whether to include extra 16-byte DID-TID-Key to delete the OTA clients

* 1. Log event definition:

|  |  |  |
| --- | --- | --- |
| Event Code | Event Description | Additional data |
| 01h | Client Unlock Door | N/A |
| 02h | Client Remove Pin | N/A |
| 03h | Admin remove GIN | N/A |
| 04h | IPA client added | N/A |
| 05h | Admin sync | N/A |
| 06h | Card Unlock Door | N/A |
| 07h | IPA admin tap | N/A |
| 08h | SetUserName | N/A |
| 09h | SetPIN | N/A |
| 0Ah | SetGIN | N/A |
| 0Bh | Pairing success | N/A |
| 0Ch | PRC client added | N/A |
| 0Dh | PRC request TID success | N/A |
| 0Eh | Client is deleted | N/A |
| 0Fh | Inherit client added | N/A |
| 10h | IPA client re-added attempt | N/A |
| 11h | Pairing fail | N/A |
| 12h | Phone client auth fail | N/A |
| 13h | TID auth fail | N/A |
| 14h | Phone client auth fail: PIN required | N/A |
| 15h | Phone client auth fail: PIN setup needed | N/A |
| 16h | Phone client auth fail: Invalid SN but DID match | N/A |
| 17h | Phone client auth fail: PIN incorrect | N/A |
| 18h | Phone client auth fail: GIN incorrect | N/A |
| 19h | Phone client auth fail: GIN not exist | N/A |
| 1Ah | Card client auth fail: unknown card | N/A (in 8006 and older version there are card UID as additional data) |
| 1Bh | Access Right denied | N/A |
| 1Ch | Lock Door | N/A |
| 1Dh | Double Auth 1st Pass | N/A |
| 1Eh | Login number expired | N/A |
| 1Fh | Unlock door jam | N/A |
| 21h | IPA fail: not in IPA | N/A |
| 22h | IPA fail: cannot set user name | N/A |
| 23h | IPA fail: cannot add user | N/A |
| 24h | OTA add (not by lock) | N/A |
| 33h | MANAGEMENT fail: auth fail when SYNC | N/A |
| 34h | MANAGEMENT fail: non-Admin attempt to execute sync | N/A |
| 35h | Delete user via keypad |  |
| 36h | Suspend user |  |
| 37h | Restore user |  |
| 38h | Delete user via keypad |  |
| 39h | Delete all users via keypad |  |
| 3Ah | Client SN-FID mapping change: FID[0~ 7] | 8-byte additional data: FID[0~7] |
| 3Bh | Client SN-FID mapping change: FID[8~15] | 8-byte additional data: FID[8~15] |
| 3Ch | Netcode/Varicode unlocking, with code value information | 8-byte additional data: Netcode/Varicode value in 32-bit value [0~3] and last 4-byte RFU |
| 3Dh | Netcode/Varicode blocked, with code value information | 8-byte additional data: Netcode/Varicode value in 32-bit value [0~3] and last 4-byte RFU |
| 3Fh | Firmware upgrade | N/A |
| 40h | Sub master unlock door |  |
| 41h | PASSWORD client added | N/A |
| 42h | PASSWORD auth fail | N/A |
| 43h | PASSWORD fail to add | N/A |
| 44h | Admin change |  |
| 45h | OPENLOCK password change |  |
| 46h | ONETIME password set |  |
| 48h | OPENLOCK password unlock door |  |
| 49h | ONETIME password unlock door |  |
| 4Ah | Admin PASSWORD unlock | N/A |
| 4Bh | Netcode PASSWORD unlock | 1-byte code\_len, 6-byte netcode (Phase in time TBD) |
| 4Ch | Guestcode PASSWORD unlock | N/A |
| 4Dh | Guescode registered |  |
| 4Eh | Guestcode cleared |  |
| 4Fh | Guestcode prefix set |  |
| 51h | Cylinder mechanical movement |  |
| 52h | Mechanical unlock |  |
| 53h | Mechanical lock |  |
| 54h | Press button manual unlock |  |
| 55h | REM 1 unlock |  |
| 56h | REM 2 Fire alarm trigger |  |
| 57h | REM 2 Fire alarm clear |  |
| 58h | Auto relock |  |
| 59h | Internal usage only |  |
| 5Ah | Internal usage only |  |
| 5Bh | One-button locking |  |
| 5Ch | Guest code disable prefix |  |
| 5Dh | Delete sub master code |  |
| 5Eh | Set master code |  |
| 5Fh | Set sub master code |  |
| 61h | Remote unlock | N/A |
| 62h | Remote add | N/A |
| 63h | Remote delete | N/A |
| 64h | Remote locking |  |
| 65h | Remote encrypted add password |  |
| 66h | Remote encrypted clone card |  |
| 70h | Lock down |  |
| 71h | Channel mode on |  |
| 72h | Channel mode off |  |
| 73h | REM 1 sensor trigger |  |
| 74h | REM 2 sensor trigger |  |
| 7Bh | Cancel Lock down |  |
| 7Ch | REM 1 sensor release |  |
| 7Dh | REM 2 sensor release |  |
| 81h | Watchdog reset | N/A |
| FFh | OTA delete (not by lock) |  |
|  |  |  |
|  |  |  |



1. Odin command transmission scenario

There are several communication scenarios between PHONE and DEVICE. They are pairing, IPA, unlock, set/change PIN, and FW update.

* 1. PAIRING procedure:

*PHONE DEVICE*

*NDEF\_UPDATE*

*(PAIRING, SendRequest, 64h,*

*10h byte DIN data,*

*10h byte DN data,*

*10h byte FID data,*

*04h-byte Time,*

*10h byte USERNAME data,*

*10h-byte App\_key\_seed,*

*10h-byte Challenge) 🡪*

*NDEF\_READ () 🡪*

*🡨 (00h for OK, PAIRING, SendRequest, 6Dh,*

*10h-byte DID data*

*10h-byte DID-FID-Key,*

*10h-byte Random,*

*1-byte DID-FID-SN,*

*2-byte M value (0x0000),*

*4-byte START\_LOG\_SN*

*1-byte SN of LOG event,*

*1-byte LOG event*

*4-byte TIME*

*10h-byte FW\_Version,*

*10h-byte SHA256-HMAC(App\_key, Challenge) )*

***OR***

*🡨 (Non-zero if NG, PAIRING, SendRequest)*

*NDEF\_UPDATE (PAIRING, Finish) 🡪*

* 1. A registered client UNLOCK the door:

*PHONE DEVICE*

*NDEF\_UPDATE*

*(AUTH, SendRequest, 46h,*

*10h-byte FID,*

*10h-byte Challenge2,*

*04h-byte Time,*

*10h-byte DID\_0,*

*1-byte DID-FID-SN\_0,*

*10h-byte DID\_1,*

*1-byte DID-FID-SN\_1)* 🡪

*NDEF\_READ () 🡪*

*🡨 (00h for OK, AUTH, SendRequest, 51h,*

*10h-byte DID data*

*10h-byte response2 data,*

*10h-byte challenge1 data,*

*10h-byte Device Name,*

*10h-byte FW\_version,*

*1-byte Battery\_ADC\_Value)*

***OR***

*🡨 (Non-zero if NG, AUTH, SendRequest)*

*NDEF\_UPDATE (AUTH, Finish, 22h,*

*10h-byte Response1, 00h,*

*1h-byte username length,*

*10-h byte Username) 🡪*

*NDEF\_READ 🡪*

*🡨 (00h for OK, AUTH, Finish)*

***OR***

*🡨 (01h if Response1 is incorrect, AUTH, Finish)*

***OR***

*🡨 (02h if requiring further PIN validation, AUTH, Finish)*

***OR***

*🡨 (03h if requiring PIN setup for the client, AUTH, Finish)*

* 1. A registered client UNLOCK the door, while DID not match in AUTH.SendRequest and the client is related to more than 10 devices:

*PHONE DEVICE*

*NDEF\_UPDATE*

*(AUTH, SendRequest\_Ex, 46h,*

*10h-byte FID,*

*10h-byte Challenge2,*

*04h-byte Time,*

*10h-byte DID\_0,*

*1-byte DID-FID-SN\_0,*

*10h-byte DID\_1,*

*1-byte DID-FID-SN\_1)* 🡪

*NDEF\_READ () 🡪*

*🡨 (01h for DID-mismatch, AUTH, SendRequest\_Ex, 51h,*

*10h-byte DID data*

*10h-byte response2 data,*

*10h-byte challenge1 data,*

*10h-byte Device Name,*

*10h-byte FW\_version,*

*1-byte Battery\_ADC\_Value)*

***OR***

*🡨 (Non-zero if NG, AUTH, SendRequest\_Ex)*

*NDEF\_UPDATE (AUTH, ChallengeResponse, 21h,*

*10h-byte Challenge2,*

*10h-byte DID,*

*1-byte DID-FID-SN)* 🡪

*NDEF\_READ 🡪*

*🡨 (00h for OK, AUTH, ChallengeResponse, 20h,*

*10h-byte response2 data,*

*10h-byte challenge1 data)*

*NDEF\_UPDATE (AUTH, Finish, 22h,*

*10h-byte Response1, 00h,*

*1h-byte username length,*

*10-h byte Username) 🡪*

*NDEF\_READ 🡪*

*🡨 (00h for OK, AUTH, Finish)*

***OR***

*🡨 (01h if Response1 is incorrect, AUTH, Finish)*

***OR***

*🡨 (02h if requiring further PIN validation, AUTH, Finish)*

***OR***

*🡨 (03h if requiring PIN setup for the client, AUTH, Finish)*

* 1. In Place Add (IPA):

*PHONE (Administrator tap) DEVICE*

*NDEF\_UPDATE*

*(AUTH, SendRequest, 35h,*

*10h-byte FID,*

*10h-byte Challenge2,*

*04h-byte Time,*

*10h-byte DID\_0,*

*1-byte DID-FID-SN\_0)* 🡪

*NDEF\_READ () 🡪*

*🡨 (00h for OK, AUTH, SendRequest, 51h,*

*10h-byte DID data*

*10h-byte response2 data,*

*10h-byte challenge1 data,*

*10h-byte device name,*

*10h-byte FW\_version,*

*1-byte Battery\_ADC\_Value)*

***OR***

*🡨 (Non-zero if NG, AUTH, SendRequest)*

*NDEF\_UPDATE (MANAGEMENT, Sync, 27h,*

*10h-byte Response1,*

*1-byte ActCode value 01h,*

*4-byte START\_LOG\_SN,*

*1-byte MAX\_READ\_SYNC\_NBR,*

*4-byte LAST\_SYNC\_LOG\_SN,*

*1-byte SN value FFh,*

*1-byte Mask value 80h,*

*1-byte Entry\_ctrl\_byte value 80h,*

*0Ah-byte Access\_Right)* 🡪

*PHONE (Client tap) DEVICE*

*NDEF\_UPDATE*

*(AUTH, SendRequest, 35h,*

*10h-byte FID,*

*10h-byte Challenge2,*

*04h-byte Time,*

*10h-byte DID\_0,*

*1-byte DID-FID-SN\_0)* 🡪

*NDEF\_READ () 🡪*

*🡨 (0Fh, AUTH, SendRequest, 51h,*

*10h-byte DID data*

*10h-byte response2 data,*

*10h-byte challenge1 data,*

*10h-byte device name,*

*10h-byte FW\_version,*

*1-byte Battery\_ADC\_Value)*

*NDEF\_UPDATE*

*(IPA, SendRequest, 40h,*

*10h-byte FID data,*

*10h-byte USERNAME,*

*10h-byte App\_key\_seed,*

*10h-byte Challenge) 🡪*

*NDEF\_READ () 🡪*

*🡨 (00h for OK, IPA, SendRequest, 5Bh,*

*10h-byte DN,*

*10h-byte DID,*

*10h-byte SC2(DID-FID-Key, Random)*

*10h-byte Random,*

*1-byte DID-FID-SN,*

*0Ah-byte Access\_Right,*

*SHA256-HMAC(App\_key, Challange) )*

***OR***

*🡨 (Non-zero if NG, IPA, SendRequest)*

*NDEF\_UPDATE (IPA, ClientFinish) 🡪*

*PHONE (Administrator tap) DEVICE*

*NDEF\_UPDATE*

*(AUTH, SendRequest, 46h,*

*10h byte FID,*

*10h byte Challenge2,*

*04h byte Time,*

*10h byte DID\_0,*

*1 byte DID-FID-SN\_0,*

*10h byte DID\_1,*

*1 byte DID-FID-SN\_1)* 🡪

*NDEF\_READ () 🡪*

*🡨 (10h for OK but SYNC required, AUTH, SendRequest, 51h,*

*10h-byte DID data*

*10h-byte response2 data,*

*10h-byte challenge1 data,*

*10h-byte device name,*

*10h-byte FW\_version,*

*1-byte Battery\_ADC\_Value)*

*NDEF\_UPDATE (MANAGEMENT, Sync, 1Ah,*

*10h-byte Response1,*

*1-byte ActCode value 00h,*

*4-byte START\_LOG\_SN,*

*1-byte MAX\_READ\_SYNC\_NBR,*

*4-byte LAST\_SYNC\_LOG\_SN)* 🡪

*NDEF\_READ 🡪*

*🡨 (00h for OK, MANAGEMENT, Sync,06h*

*1-byte M value 01h,*

*2-byte N value 0000h,*

*1-byte DID-FID-SN\_0,*

*1-byte Mask\_0,*

*1-byte entry\_ctrl\_byte\_0)*

* 1. The administrator requesting 5 TIDs for PRC:
     1. Approach A:

*PHONE (Administrator tap) DEVICE*

*NDEF\_UPDATE*

*(AUTH, SendRequest, 46h,*

*10h-byte FID,*

*10h-byte Challenge2,*

*04h-byte Time,*

*10h-byte DID\_0,*

*1-byte DID-FID-SN\_0,*

*10h-byte DID\_1,*

*1-byte DID-FID-SN\_1)* 🡪

*NDEF\_READ () 🡪*

*🡨 (00h for OK, AUTH, SendRequest, 51h,*

*10h-byte DID data*

*10h-byte response2 data,*

*10h-byte challenge1 data,*

*10h-byte device name,*

*10h-byte FW\_version,*

*1-byte Battery\_ADC\_Value)*

***OR***

*🡨 (Non-zero if NG, AUTH, SendRequest)*

*NDEF\_UPDATE*

*(PRC, RequestTID, 11h,*

*Response1, 05h)* 🡪

*NDEF\_READ () 🡪*

*🡨 (00h for OK, PRC, RequestTID, 65h,*

*10h-byte Random*

*10h-byte S2(DID-TID-Key\_0, Random),*

*1-byte DID-TID-SN\_0*

*10h-byte S2(DID-TID-Key\_1, Random)*

*1-byte DID-TID-SN\_1,*

*10h-byte S2(DID-TID-Key\_3, Random),*

*1-byte DID-TID-SN\_3*

*10h-byte S2(DID-TID-Key\_4, Random)*

*1-byte DID-TID-SN\_4,*

*10h-byte S2(DID-TID-Key\_5, Random)*

*1-byte DID-TID-SN\_5)*

***OR***

*🡨 (Non-zero if NG, PRC, RequestTID)*

*NDEF\_UPDATE (PRC, AdminFinish) 🡪*

* + 1. Approach B: AUTH.SendRequest -> MANAGEMENT.Sync -> PRC.RequestTID -> MANAGEMENT.AckSync

*PHONE (Administrator tap) DEVICE*

*NDEF\_UPDATE*

*(AUTH, SendRequest, 46h,*

*10h-byte FID,*

*10h-byte Challenge2,*

*04h-byte Time,*

*10h-byte DID\_0,*

*1-byte DID-FID-SN\_0,*

*10h-byte DID\_1,*

*1-byte DID-FID-SN\_1)* 🡪

*NDEF\_READ () 🡪*

*🡨 (00h for OK, AUTH, SendRequest, 51h,*

*10h-byte DID data*

*10h-byte response2 data,*

*10h-byte challenge1 data,*

*10h-byte device name,*

*10h-byte FW\_version,*

*1-byte Battery\_ADC\_Value)*

***OR***

*🡨 (Non-zero if NG, AUTH, SendRequest)*

*NDEF\_UPDATE (MANAGEMENT, Sync, 1Ah,*

*10h-byte Response1,*

*1-byte ActCode value 00h,*

*4-byte START\_LOG\_SN,*

*1-byte MAX\_READ\_SYNC\_NBR,*

*4-byte LAST\_SYNC\_LOG\_SN)* 🡪

*NDEF\_READ 🡪*

*🡨 (00h for OK, MANAGEMENT, Sync,09h*

*1-byte M value 00h,*

*2-byte N value 0000h,*

*1-byte DID-FID-SN\_0,*

*1-byte LogEvent,*

*4-byte TIME)*

*NDEF\_UPDATE*

*(PRC, RequestTID, 11h,*

*10h-byte Response1, 05h)* 🡪

*NDEF\_READ () 🡪*

*🡨 (00h for OK, PRC, RequestTID, 65h,*

*10h-byte Random,*

*10h-byte S2(DID-TID-Key\_0, Random),*

*1-byte DID-TID-SN\_0,*

*10h-byte S2(DID-TID-Key\_1, Random)*

*1-byte DID-TID-SN\_1,*

*10h-byte S2(DID-TID-Key\_2, Random),*

*1-byte DID-TID-SN\_2,*

*10h-byte S2(DID-TID-Key\_3, Random)*

*1-byte DID-TID-SN\_3,*

*10h-byte S2(DID-TID-Key\_4, Random)*

*1-byte DID-TID-SN\_4)*

***OR***

*🡨 (Non-zero if NG, PRC, RequestTID)*

*NDEF\_UPDATE (PRC, AdminFinish) 🡪*

* 1. The TID client taps for the first time to setup its real FID:

*PHONE (Client tap) DEVICE*

*NDEF\_UPDATE*

*(AUTH, SendRequest, 35h,*

*10h-byte FID,*

*10h-byte Challenge2,*

*04h-byte Time,*

*10h-byte DID\_0,*

*1-byte DID-TID-SN\_0)* 🡪

*NDEF\_READ () 🡪*

*🡨 (00h for OK, AUTH, SendRequest, 51h,*

*10h-byte DID data*

*10h-byte response2 data,*

*10h-byte challenge1 data,*

*10h-byte device name,*

*10h-byte FW\_version,*

*1-byte Battery\_ADC\_Value)*

***OR***

*🡨 (Non-zero if NG, AUTH, SendRequest)*

*NDEF\_UPDATE (PRC, RegisterClient, 30h,*

*10h-byte Response1 data,*

*10h-byte USERNAME,*

*10h-byte AES\_ECB(Access\_Right, DID-TID-Key) )* 🡪

*NDEF\_READ 🡪*

*🡨 (OK, PRC, RegisterClient, 21h,*

*10h-byte SC2(DID-FID-Key, Random)*

*10h-byte Random,*

*1-byte DID-FID-SN)*

***OR***

*🡨 (Non-zero if NG, PRC, RegisterClient)*

*NDEF\_UPDATE (PRC, ClientFinish) 🡪*

* 1. Client set/add PIN:

*PHONE (Client tap) DEVICE*

*NDEF\_UPDATE*

*(AUTH, SendRequest, 35h,*

*10h-byte FID,*

*10h-byte Challenge2,*

*04h-byte Time,*

*10h-byte DID\_0,*

*1-byte DID-FID-SN\_0)* 🡪

*NDEF\_READ () 🡪*

*🡨 (OK, AUTH, SendRequest, 51h,*

*10h-byte DID data*

*10h-byte response2 data,*

*10h-byte challenge1 data,*

*10h-byte device name,*

*10h-byte FW\_version,*

*1-byte Battery\_ADC\_Value)*

***OR***

*🡨 (Non-zero if NG, AUTH, SendRequest)*

*NDEF\_UPDATE (MANAGEMENT, SetPIN,*

*30h,*

*10h-byte Response1,*

*10h-byte SC(DID-HPIN, Random),*

*10h-byte Random)* 🡪

* 1. Client remove PIN:

*PHONE (Client tap) DEVICE*

*NDEF\_UPDATE*

*(AUTH, SendRequest, 35h,*

*10h-byte FID,*

*10h-byte Challenge2,*

*04h-byte Time,*

*10h-byte DID\_0,*

*1-byte DID-FID-SN\_0)* 🡪

*NDEF\_READ () 🡪*

*🡨 (OK, AUTH, SendRequest, 51h,*

*10h-byte DID data*

*10h-byte response2 data,*

*10h-byte challenge1 data,*

*10h-byte device name,*

*10h-byte FW\_version,*

*1-byte Battery\_ADC\_Value)*

***OR***

*🡨 (Non-zero if NG, AUTH, SendRequest)*

*NDEF\_UPDATE (AUTH, ValidatePIN, 21h,*

*10h-byte Response1 data,*

*10h-byte Scramble3(Challenge1,DID-HPIN) ,*

*02h)* 🡪

* 1. Administrator UNLOCK door and synchronize client list with DEVICE. In this case, there are 2 client list entries need to reflect changes from PHONE to DEVICE while there are 1 client list entry needs to reflect changes from DEVICE to PHONE.

*PHONE DEVICE*

*NDEF\_UPDATE*

*(AUTH, SendRequest, 46h,*

*10h byte FID,*

*10h byte Challenge2,*

*04h byte Time,*

*10h byte DID\_0,*

*1 byte DID-FID-SN\_0,*

*10h byte DID\_1,*

*1 byte DID-FID-SN\_1)* 🡪

*NDEF\_READ () 🡪*

*🡨 (10h for OK but SYNC required, AUTH, SendRequest, 51h,*

*10h-byte DID data*

*10h-byte response2 data,*

*10h-byte challenge1 data,*

*10h-byte device name,*

*10h-byte FW\_version,*

*1-byte Battery\_ADC\_Value)*

***OR***

*🡨 (Non-zero if NG, AUTH, SendRequest)*

*NDEF\_UPDATE (MANAGEMENT, Sync, 17h,*

*10h-byte Response1, 00h,*

*1-byte DID-FID-SN\_0,*

*1-byte Mask\_0,*

*1-byte entry\_ctrl\_byte\_0,*

*1-byte DID-FID-SN\_1,*

*1-byte Mask\_1,*

*1-byte entry\_ctrl\_byte\_1 )* 🡪

*NDEF\_READ 🡪*

*🡨 (00h for OK, MANAGEMENT, Sync, 06h*

*01h,*

*0000h,*

*1-byte DID-FID-SN\_0,*

*1-byte Mask\_0,*

*1-byte entry\_ctrl\_byte\_0)*

***OR***

*🡨 (01h if Response1 is incorrect, MANAGEMENT, Sync)*

***OR***

*🡨 (02h if requiring further PIN validation, AUTH, Finish)*

***OR***

*🡨 (03h if requiring PIN setup for the client, AUTH, Finish)*

* 1. Administrator set/add GIN:

*PHONE DEVICE*

*NDEF\_UPDATE*

*(AUTH, SendRequest, 35h,*

*10h byte FID,*

*10h byte Challenge2,*

*04h-byte Time,*

*10h byte DID\_0,*

*1 byte DID-FID-SN\_0)* 🡪

*NDEF\_READ () 🡪*

*🡨 (OK, AUTH, SendRequest, 51h,*

*10h-byte DID data*

*10h-byte response2 data,*

*10h-byte challenge1 data,*

*10h-byte device name,*

*10h-byte FW\_version,*

*1-byte Battery\_ADC\_Value)*

***OR***

*🡨 (Non-zero if NG, AUTH, SendRequest)*

*NDEF\_UPDATE (MANAGEMENT, SetGIN,*

*30h,*

*10h-byte Response1,*

*10h-byte SC(DID-HGIN, Random),*

*10h-byte Random)* 🡪

* 1. Administrator remove GIN:

P*HONE (Administrator tap) DEVICE*

*NDEF\_UPDATE*

*(AUTH, SendRequest, 35h,*

*10h byte FID,*

*10h byte Challenge2,*

*04h-byte Time,*

*10h byte DID\_0,*

*1 byte DID-FID-SN\_0)* 🡪

*NDEF\_READ () 🡪*

*🡨 (OK, AUTH, SendRequest, 51h,*

*10h-byte DID data*

*10h-byte response2 data,*

*10h-byte challenge1 data,*

*10h-byte device name,*

*10h-byte FW\_version,*

*1-byte Battery\_ADC\_Value)*

***OR***

*🡨 (Non-zero if NG, AUTH, SendRequest)*

*NDEF\_UPDATE (AUTH, ValidatePIN, 21h,*

*10h-byte Response1 data,*

*10h-byte Scramble3(Challenge1,DID-HPIN) ,*

*03h)* 🡪

* 1. A registered client UNLOCK the door, with GIN validation:

*PHONE DEVICE*

*NDEF\_UPDATE*

*(AUTH, SendRequest, 46h,*

*10h-byte FID,*

*10h-byte Challenge2,*

*04h-byte Time,*

*10h-byte DID\_0,*

*1-byte DID-FID-SN\_0,*

*10h-byte DID\_1,*

*1-byte DID-FID-SN\_1)* 🡪

*NDEF\_READ () 🡪*

*🡨 (OK, AUTH, SendRequest, 51h,*

*10h-byte DID data*

*10h-byte response2 data,*

*10h-byte challenge1 data,*

*10h-byte device name,*

*10h-byte FW\_version,*

*1-byte Battery\_ADC\_Value)*

***OR***

*🡨 (Non-zero if NG, AUTH, SendRequest)*

*NDEF\_UPDATE (AUTH, ValidateGIN, 21h,*

*10h-byte Scramble3(Challenge1,DID-HGIN) ,*

*00h) 🡪*

1. The ISO 14443-4 APDU sequences in Odin command transmission flow:

Here are the APDU sequences happen during NFC forum Type 4 Tag operation in Android.

* 1. Automatic discovery:

When PHONE detects DEVICE, Android system will detect NFC tag by issuing C-APDU sequence as Table 1. Totally 8 APDU is required.

*Select NDEF TAG APPLICATION*

*Select CC FILE*

*ReadBinary CC FILE*

*Select NDEF FILE*

*ReadBinary NLEN*

*Select NDEF FILE*

***ReadBinary UNTIL NDEF’s PAYLOAD part***

***ReadBinary FROM NDEF’s PAYLOAD part***

***Table 1***

* 1. NDEF message read by using android.nfc.tech.Ndef.getNdefMessage

The APDU sequence is the same as the sequence in automatic detection.

* 1. NDEF message write by using android.nfc.tech.Ndef.writeNdefMessage. The APDU sequence is as follows.

*Select NDEF TAG APPLICATION*

*Select CC FILE*

*ReadBinary CC FILE*

*Select NDEF FILE*

*ReadBinary NLEN*

*Selcet NDEF FILE*

***UpdateBinary ZERO NLEN and NDEF file content***

***UpdateBinary real NLEN***

***Table 2***

* 1. Refinement of NDEF message read procedure:

Since our protocol data is encapsulated in NDEF and during a tap, there are multiple NDEF read and NDEF write to complete any Odin tap scenario. It implies there might be hundreds of APDU transmissions which increase the response time felt by end users. Therefore, we would like to reduce the number of APDU transmission in NDEF read and NDEF write.

We notice that android.nfc.tech.Ndef.readNdefMessage needs to first issue 6 C-APDU: select NDEF tag application, select CC file, read CC file, select NDEF file, read NDEF file length, and select NDEF file again (normal face part in **Table 1**) before really update the NDEF content. Then by ReadBinary from offset 0 and ReadBinary from offset of NDEF payload part (bold face in **Table 1**), readNdefMessage completes NDEF read. However, our DEVICE will not change CC file and NDEF file during any tap scenario. Thus the transmission of the first 6 C-APDU can be eliminated and we only the 2 ReadBinary command.

***ReadBinary UNTIL NDEF’s PAYLOAD part***

***ReadBinary FROM NDEF’s PAYLOAD part***

***Table 3***

Of course, then we cannot rely on android.nfc.tech.Ndef.readNdefMessage. This refined version should be implemented by utilizing android.nfc.tech.IsoDep.transceive

* 1. Refinement of NDEF message write procedure:

Similar to NDEF read procedure, the first 6 C-APDU (normal face part in **Table 2**) can be eliminated. Thus we derive a refined version of NDEF message write:

***UpdateBinary ZERO NLEN and NDEF file content***

***UpdateBinary real NLEN***

**Table 4**

Also this refined version should be implemented by utilizing android.nfc.tech.IsoDep.transceive.

Appendix:

1. ISO 14443-4 Packet

The command is of the following format.

CLA INS P1 P2 Lc Data Le

The command used in NFC Forum Type 4 Tag Operation.

* 1. NDEF tag application select:

The C-APDU is as follows.

CLA INS P1 P2 Lc Data Le

00h A4h 04h 00h 07h D2760000850101h 00h

The R-APDU is as follows.

Data SW1 SW2

-- 90h 00h

-- 6Ah 82h

* 1. Select CC file:

The C-APDU is as follows.

CLA INS P1 P2 Lc Data

00h A4h 00h 0Ch 02h E103h

The R-APDU is as follows.

Data SW1 SW2

-- 90h 00h

-- 6Ah 82h

* 1. Select NDEF file:

The C-APDU is as follows.

CLA INS P1 P2 Lc Data

00h A4h 00h 0Ch 02h E104h

The R-APDU is as follows.

Data SW1 SW2

-- 90h 00h

-- 6Ah 82h

* 1. ReadBinary:

The C-APDU is as follows.

CLA INS P1 P2 Lc Data Le

00h B0h Offset -- -- Length Le

The R-APDU is as follows.

Data SW1 SW2

Content read 90h 00h

-- 6Ah 82h

* 1. UpdateBinary:

The C-APDU is as follows.

CLA INS P1 P2 Lc Data Le

00h D6h Offset Length Lc Data to be written in NDEF file --

The R-APDU is as follows.

Data SW1 SW2

-- 90h 00h

-- 6Ah 82h

1. NDEF record, message and storage:

The following figure depicts NDEF short record layout. In Odin protocol we do not use ID field so the 1-byte ID length field and variable length ID field are absent, and IL is 1. In Odin there is only a single NDEF record transmitted in one NDEF message, so MB and ME are both 1, and CF are 0. TNF is 02h which means MIME type.



The NDEF message is stored in an EF file called NDEF file using the following data structure. NLEN is the NDEF message size.

Offset Size Field Remarks

00h 2 NLEN Size of NDEF message stored in the NDEF file

02h NLEN NDEF NDEF message

message

1. Example of complete Odin packet bytes:

The first UpdateBinary command C-APDU in Odin command (Pairing, SendRequest):

CLA INS P1 P2 Lc Data

2-byte NLEN NDEF message

T&F TLEN PLEN TYPE PAYLOAD

Cmd\_mode

00h D6h 00h 00h 49h 0000h D2h 0Bh 39h ‘pkinno/odin’ 01h

PAYLOAD (cont.)

Cmd\_code Data\_len Data

01h 36h ‘odinodinodinodin’ ‘0123456789abcdef’

Le

PAYLOAD (cont.)

Data (cont.)

54h 98h 41h 90h 02h 54h 98h 02h 54h 98h 41h 90h 02h 54h 98h 02h ‘Steven’ --

The second UpdateBinary command C-APDU in Odin command (Pairing, SendRequest):

CLA INS P1 P2 Lc Data Le

2-byte NLEN

00h D6h 00h 00h 02h 0047h --