Active working mode

The reader/writer can be configured in active working mode. Configurable parameters include: antenna used for inventory, upload mode, data sorting mode, event subscription, GPI trigger work, additional bank data, tag filtering settings, etc. When the reader is configured in active working mode, the reader needs to be restarted. After the reader is restarted, it enters active working mode. The reader collects various events that occur, and then actively uploads them to the server based on the event type subscribed by the user.

▼ Event upload mode

The communication methods for event uploading include Http and Socket modes. In the server-side reply, instructions can be issued for the reader and writer to execute. Each reply can only issue one remote instruction at most, or no instructions can be issued.

HTTP mode

Use the post method to submit reader-side events to the server. The Content-Type header is application/json; charset=UTF-8, and the text part is a complete Json object with the following format:

member name	type	describe
reader_name	String	The reader/writer name format is: reader/writer logical name/reader/writer IP address. The logical name of the reader can be set to a string of less than 255 characters
event_type	String	event type
event_data	Event data object (different event types have different structures of event data objects)	Event details

The server-side reply text can contain remote instructions that need to be executed by the reader-writer side. The instructions are encapsulated in Json format and the format is as follows:

member name	type	describe
command_type	String	Remote command type

member name	type	describe
command_data	Server command data object (different remote command types have different structures of server command data objects)	Detailed operating data for remote commands

Socket mode

Reserved for now

event type

The following table explains in detail each event type that may be generated and uploaded on the reader-writer side. Some of the event types are automatically uploaded without subscribing when configuring the reader-writer active working mode, while others must be subscribed. Will upload

Event type name	event_type value	event_data type	Subscribe to upload	Optional remote command	describe
tag read event	tag_read	Tag read event data object	no	set_gpo reboot	After the tags are inventoried, the read tag data will be uploaded periodically or on demand according to the data sorting mode.
Reader/writer exception event	reader_exception	Reader/writer exception event data object	no	reboot	When an exception occurs to the reader/writer, the exception information data will be uploaded immediately.
tag entry event	tag_coming	tag entry event data object	yes	set_gpo reboot	Although the tag data can be obtained through the tag read event server, the reader may first sort out the

Introduction to 18000-6C protocol

Introduction to API interface

code example

Introduction to API public objects

Reader parameters

Synchronize inventory tags

Asynchronous inventory tags

Tag access operations

Active working mode

Peripheral function operation

Event type name	event_type value	event_data type	Subscribe to upload	Optional remote command	describe
					tag data and then upload the read tag, so for applications that need to know immediately when the tag enters the antenna field area type, you need to subscribe to this event. After subscribing to this event, the reader will immediately upload the tag data after the tag is inventoried. Currently, this event function has not been implemented.
heartbeat event	heart_beat	Heartbeat event data object	yes	reboot	Used to monitor whether the reader/writer is still online and working normally. After subscribing to this event, the reader/writer regularly uploads heartbeat data to the server according to the configured heartbeat cycle.
Synchronization time request event	sync_time_req	Synchronization time request event data object	yes	sync_time	The event data object generally contains a timestamp member (a 64-bit integer, representing the

Event type name	event_type value	event_data type	Subscribe to upload	Optional remote command	describe	
						number of milliseconds that have elapsed relative to 1970-01-01 00:00:00), which is used to represent the time when the event occurs, but the After power-on, the time will always be reset to 1970-01-01 00:00:00. If the application needs to use the correct event stamp, it should subscribe to this event. After subscribing to this event, the reader and writer will first send a message to the server before starting active work. When the server uploads a synchronization time event request, the server side must include the synchronization time remote command in the body of the Http reply. The reader-writer side immediately modifies the system time synchronously after receiving this remote command.

Remote command type

Remote command name	command_type value	command_data type	describe
synchronised time	sync_time	Synchronize time data objects	The data carried in the reply to the sync_time_req event instructs the reader to synchronize the system time
Set GPO status	set_gpo	GPO status setting object array	If the server-side Http reply text contains this remote command, the reader/writer will immediately set the GPO status according to the command description.
Reboot the device	reboot	empty object	If the server-side Http reply text contains this remote command, the reader will restart immediately.

Tag read event data object

type	describe
Array of TagInfo objects	Label Information

The following example is a post to a server-side tag that reads the complete Json object of the event.

```
{
      reader_name: "Identium_reader/192.168.1.100"
      event_type: "tag_read" ,
  -   event_data:[
        - {
                epc: "E20041453116009820603EC6" ,
                bank_data: "" ,
                antenna: 1 ,
                read_count: 30 ,
                protocol: 5 ,
                rsi: -63 ,
                firstseen_timestamp: 1550734263000 ,
                lastseen_timestamp: 1550734268000
          },
        - {
                epc: "E20041453116011620703E7F" ,
                bank_data: "" ,
                antenna: 2 ,
                read_count: 1 ,
                protocol: 5 ,
                rsi: -77 ,
                firstseen_timestamp: 1550734263000 ,
                lastseen_timestamp: 1550734268000
            }
        ]
}
```

Reader/writer exception event data object

member name	type	describe
err_code	Number	error code
err_string	String	error message
timestamp	Number	The timestamp when the exception occurred

The following example is a complete Json object that is posted to the server-side reader-writer exception event.

```
{
    reader_name: "Identium_reader/192.168.1.100"
    event_type: "reader_exception" ,
  - event_data: {
        err_code: 101285 ,
        err_string: "FAULT_HIGH_RETURN_LOSS-ant_1" ,
        timestamp: 1550734268000
    }
}
```

tag entry event data object

type	describe
Array of TagInfo objects	Label Information

The following example is a complete Json object of a tag entry event posted to the server side.

```
{
      reader_name: "Identium_reader/192.168.1.100",
      event_type: " tag_coming" ,
   -  event_data:[
         -  {
                  epc: "E200414531160098206O3EFF" ,
                  bank_data: "" ,
                  antenna: 3 ,
                  read_count: 30 ,
                  protocol: 5 ,
                  rsi: -63 ,
                  firstseen_timestamp: 1550734256000 ,
                  lastseen_timestamp: 1550734272000
            },
         -  {
                  epc: "E20041453116011620703ECC" ,
                  bank_data: "" ,
                  antenna: 2 ,
                  read_count: 1 ,
                  protocol: 5 ,
                  rsi: -77 ,
                  firstseen_timestamp: 1550734256000 ,
                  lastseen_timestamp: 1550734272000
            }
      ]
}
```

▼ Heartbeat event data object

type	describe
Number	Heartbeat count. Every time a heartbeat event is uploaded, the count increases by one.

The following example is a complete Json object of a heartbeat event posted to the server.

```
{
      reader_name: "Identium_reader/192.168.1.100,"
      event_type: "heart_beat" ,
      event_data: 1234
}
```

Synchronization time request event data object

type	describe
empty object	

The following example is a complete Json object of a heartbeat event posted to the server.

```
{
      reader_name: "Identium_reader/192.168.1.100,"
      event_type: "sync_time_req" ,
      event_data: { }
}
```

Synchronize time data objects

type	describe
Number	The current time minus the number of milliseconds of 1970-01-01 00:00:00. The reader/writer immediately sets the system time to this time after receiving this information.

The following example is the complete Json object of the synchronization time instruction in the server-side reply body.

```
{
      command_type: "sync_time" ,
        command_data: 1550734153000
}
```

GPO status setting object

member name	type	describe
gpo	Number	GPO pin number
state	Number	GPO pin status, legal value is 0 or 1
duration	Number	The duration of the GPO pin state, in seconds, valid values are 0-3600. When the GPO pin is set to the state state, the GPO pin state returns to the 1-state state after duration seconds. If you need the GPO pin state to remain in state without reversal, duration can be set to 0.

The following example is the complete Json object of the set GPO status command in the server-side reply text. The following example sets GPO1 to status 0 for 2 seconds and sets GPO2 to status 1 for 3 seconds.

```
{
        command_type: "set_gpo" ,
    -   command_data:[
            -   {
                        gpo: 1 ,
                        state: 0 ,
                        duration: 2
                },
            -   {
                        gpo: 2 ,
                        state: 1 ,
                        duration: 3
                }
        ]
}
```