TARVOS PRO

User Guide

STAR SYSTEMS INTERNATIONAL ™

Version 2.0

Attention

The Tarvos Pro RFID Integrated reader described in this User Guide is a commercial product and must be installed by professional installer.

FCC Radiation Exposure Statement

The antennas used for this transmitter must be installed to provide a minimum separation distance of at least 1 meter from any person and must not be co-located with any other transmitter.

Site License Disclaimer

Users of the Tarvos Pro RFID Integrated reader acknowledge that a site license is required when the device is configured for FCC Part 90 regulations. It is the user's responsibility to file for the site license and submit the appropriate fees and payments to the regulating authority. United States filings require submission of FCC Form 601 with Schedule D and H. Canadian filings require submission of Industry Canada forms IC2365BB and IC2430BB.

Licence d'Etat-client Avertissement

Client (utilisateur final) reconnaît que le site d'une licence est requise pour chaque lecteur emplacement du système. Il incombe au client de déposer pour la licence d'exploitation et soumettre le paiement du dépôt approprié. Unis dépôts États exigent l'achèvement et la soumission du formulaire FCC 601 à l'annexe D et H. dépôts canadiennes exigent l'achèvement et la soumission de Industrie Canada Formulaires IC2365BB et IC2430BB.

Changes or Modifications

Changes or modifications to the RFID reader not expressly approved by STAR Systems International Ltd. could void the user's authority to operate the Tarvos Pro RFID Integrated reader.

WARNING

This equipment complies with FCC Part 90 and Industry Canada.RSS-137 rules. This device complies with FCC Part 15 and Industry Canada license exempt RSS standard(s). Operation is subject to the following two conditions: (1) this device may not cause interference, and (2) this device must accept any interference, including interference that may cause undesired operation of the device.

Cet appareil est conforme à FCC Partie15 de Industrie Canada RSS standard exempts de licence (s). Son tilisation est soumise à Les deux conditions suivantes: (1) cet appareil ne peut pas provoquer 'interférences et (2) cet appareil doit accepter Toute interférence, y compris les interférences qui peuvent causer un mauvais fonctionnement du dispositif.

WARNING: Class A Devices

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio-frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case users will be required to correct the interference at their own expense.

Antennas

This radio transmitter (Tarvos Pro RFID Integrated reader **IC ID 20068-TVSPRO31000**) has been approved by Industry Canada to operate with the antenna types listed below with the maximum permissible gain indicated. Antenna types not included in this list, having a gain greater than the maximum gain indicated for that type, are strictly prohibited for use with this device.

Model:	Gain	Horizontal	Beamwidth
1)	Avior	15 dBi	30°
2)	Avior-36°	14 dBi	36°
3)	Avalon	13 dBi	36°
4)	Cheetah	12 dBi	42°
5)	Kuma	10 dBi	55°

Professional Installation

The Tarvos Pro RFID Integrated Reader has integrated a 13 dBi gain, 36° beamwidth antenna and one external antenna port. Professional installation or authorized service personnel is required to configure radio parameters of the transmitter using the software for adjusting total EIRP (30W) power at local installation to ensure compliance with FCC Rules.

RF Cable Option

For the applications in Parking and Secure Access control, cable length is limited. Cable with attenuation of maximum 4dB will typically be used.

For the applications of Electronic Toll Collection (ETC), cable length would be site orientated that could lead to extended cable length. Cable with attenuation of maximum 12dB will typically be used.

Please read this before connecting the TARVOS PRO reader

Download the Electrical Precaution and Wiring Connection Guide:

Contents

User Guide

Version 2.0

14th June 2024

Specifications are subjected to changes without prior notice.

Disclaimer and limitation of liability

STAR Systems International Ltd. shall not be liable for technical or editorial errors or omissions contained herein or for incidental or consequential damages about the furnishing, performance, or use of this material. The information in this document is provided “as is” without warranty of any kind - including but not limited to, the implied warranties of merchantability and fitness for a purpose and is subjected to change without notice. The warranties for STAR Systems products are set forth in the express limited warranty statements accompanying such products. Nothing herein should be construed as constituting an additional warranty.

Users of the Tarvos Pro RFID reader acknowledge that a site license is required when the device is configured for FCC Part 90 regulations. It is the user’s responsibility to file for the site license and submit the appropriate fees and payments to the regulating authority. United States filings require submission of FCC Form 601 with Schedule D and H. Canadian filings require submission of Industry Canada forms IC2365BB and IC2430BB.

Your safety is extremely important. Read and follow all warnings and cautions in this document before handling and operating RFID equipment. You can be seriously injured, and equipment and data can be damaged if you do not follow the safety warnings and cautions.

A caution alerts you to an operating procedure, practice, condition, or statement that must be strictly observed to prevent equipment damage or destruction, or corruption or loss of data.

Note: Notes either provide extra information about a topic or contain special instructions for handling a condition or set of circumstances.

I. Introduction

1.1 Purpose and Scope

This purpose of this document is to educate developers designing reader control and data applications for the Tarvos Pro reader. It describes the Tarvos Pro Reader interfaces and protocols as well as providing references and relevant UHF RFID information.

The users of the Tarvos Pro reader are assumed to have basic knowledge of software development and network connectivity. In addition, a high-level understanding of reader protocols would be beneficial. Please refer to the documents in the next section for more details.

1.2 Reference Documents

The documents below will describe UHF RFID and the protocols in more detail.

- EPC Global, *EPC™ Radio-Frequency Identity Protocols Generation-2 UHF*, v2.0.1
- EPC Global, *Low Level Reader Protocol (LLRP)*, Version 1.1

1.3 Document Conventions

Item	Convention	Explanation / Example
Acronyms/Initialisms	All uppercase; usually spelled out on first use.	Radio Frequency Identification (RFID)
Book titles	Title caps, italic.	See the *Low Level Reader Protocol (LLRP)*
Chapter titles	Title caps, in quotation marks.	See Chapter 2, "Tarvos Pro External Interfaces"
Code samples and examples	Monospaced font.	antennas.mux_sequence=1 2 2 1
URLs	Lowercase	http://www.star-int.net/

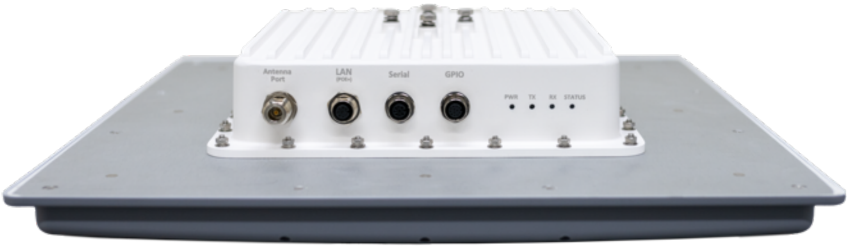
2. Tarvos Pro External Interfaces

The Tarvos Pro Integrated Reader is a multi-protocol Automatic Vehicle Identification (AVI) and tolling reader. The Tarvos Pro reader provides network and serial connectivity with the controlling system software as well as general purpose inputs and outputs.

2.1 Reader Hardware Overview

Font side

Back side



Front Panel View

2.2 Reader Physical Interface

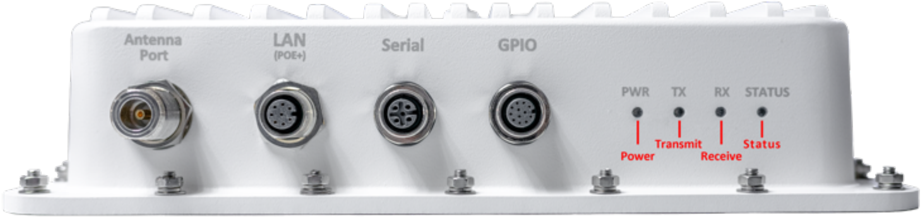


FIGURE 1 - Connector and LED indicators

LED	Operational description
Power [PWR]	Green indicates power has been applied to the reader. Any other colors indicate the reader is booting or powered off.
Transmit [TX]	Green when transmitting. Amber indicates one more radio warning have occurred (e.g. missing antenna). Red indicates one more radio error have occurred. When Red, the “Transmit” LED follows the value of the “diag.radio_error_status” CLI variable.
Receive [RX]	Blinks green when tag signals are being decoded. Faster blink indicates higher read rate.
Status	Green indicates no error. Amber indicates one or more general warnings have occurred. Red indicates one or more general errors have occurred. The “Status” LED follows the value of the “diag.error_status” CLI variable.

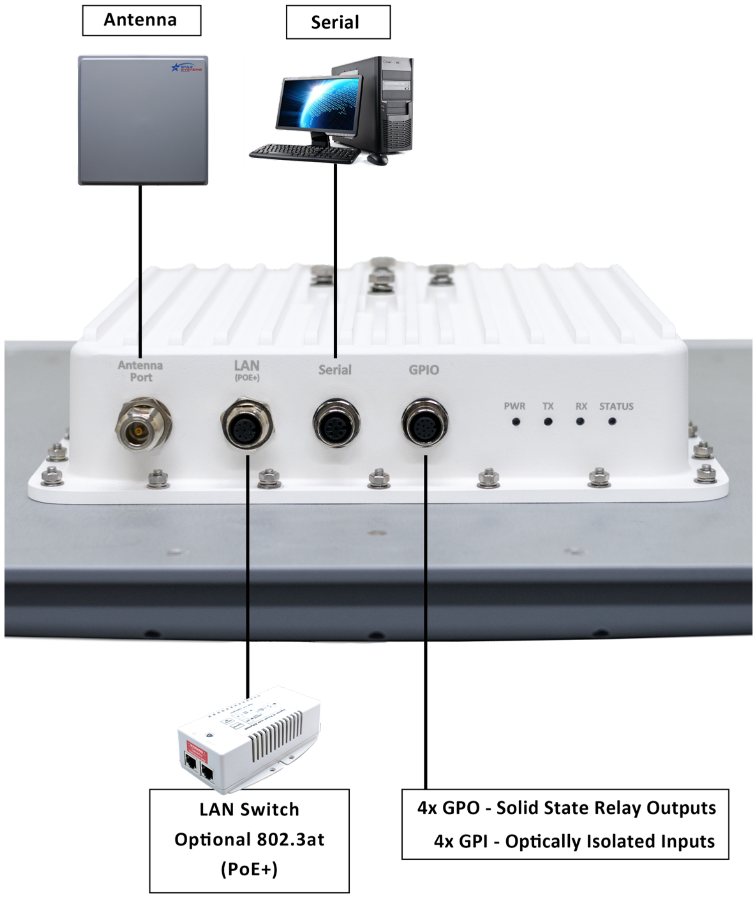
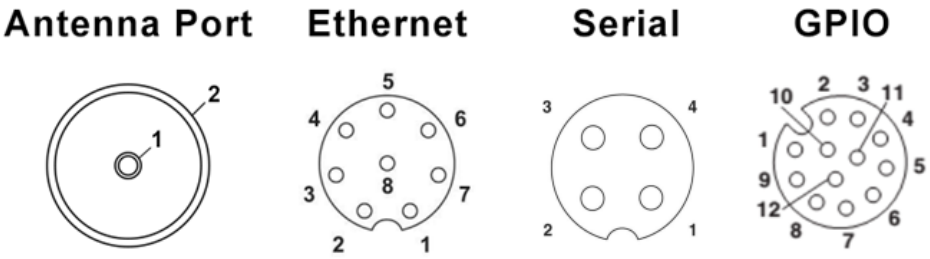


FIGURE 2 – Tarvos Pro Connection Diagram

Interface	Description
PoE+	25.5W IEEE 802.3at Compliant (Type 2)
Ethernet	IEEE 802.3 10BASE-T/100BASE-TX IEEE 802.3 compliant Ethernet transceiver through an RJ-45 connector that has PoE+ magnetics.
Serial	RS232 serial port (default setting 115200, 8-N-1)
GPIO	Four opto-isolated inputs and four opto-isolated open collector outputs

Connector pin out details

The following diagram provides specific details regarding each connector type:



Antenna Port:

Pin	Signal	Description
1	RF	Center Pin RF output
2	GND	Ground

Ethernet - LAN(PoE+):

Pin	Mode A	Mode B	Description
1	Rx+, DC+	Rx+	LAN Rx+, DC+ for Mode A POE Spec
2	Rx-, DC+	Rx-	LAN Rx-, DC+ for Mode A POE Spec
3	Tx+, DC-	Tx+	LAN Tx+, DC- for Mode A POE Spec
4	Unused	DC+	DC+ for Mode B POE Spec
5	Unused	DC+	DC+ for Mode B POE Spec
6	Tx-, DC-	Tx-	LAN Tx-, DC- for Mode A POE Spec
7	Unused	DC-	DC- for Mode B POE Spec
8	Unused	DC-	DC- for Mode B POE Spec

Serial:

Pin	Signal	Description	Color
1	TX	RS232 Transmit	Brown
2	N/A	N/A	N/A
3	GND	Signal Return	Blue
4	RX	RS232 Receive	Black

GPIO:

Pin	Signal	Description	Color
1	GPO3_ISO	Optically Isolated, Open Collector General Purpose Output #3	Brown
2	GP3_Return	Return path for General Purpose Input/Output #3	Blue
3	GPI2_ISO	Optically Isolated, General Purpose Input #2	White
4	GPI3_ISO	Optically Isolated, General Purpose Input #3	Green
5	GP2_Return	Return path for General Purpose Input/Output #2	Pink
6	GPI1_ISO	Optically Isolated, General Purpose Input #1	Yellow
7	GPO2_ISO	Optically Isolated, Open Collector General Purpose Output #2	Black
8	GP1_Return	Return path for General Purpose Input/Output #1	Gray
9	GPI0_ISO	Optically Isolated, General Purpose Input #0	Red
10	GPO1_ISO	Optically Isolated, Open Collector General Purpose Output #1	Purple
11	GP0_Return	Return path for General Purpose Input/Output #0	Red/Gray
12	GPO0_ISO	Optically Isolated, Open Collector General Purpose Output #0	Red/Blue

2.3 Reader Software Interface

Interface	Description
TCP Port 50007	TSI Command Port
TCP Port 50008	TSI Event Port
TCP Port 5084	LLRP Port
Serial	TSI Command and Event Port
TCP Port 80	Web Interface
Syslog	Outputs to a remote UDP Port 514
TCP Port 3334	Firmware Update Port • Automatically used by Web and GUI Updates
UDP Port 3333	Discovery port (Multicast IP addresses 239.192.7.1 and 239.192.7.2)
SNTP	Simple Network Time Protocol Client • RFC 2030
SNMP	Simple Network Management Protocol SNMPv1 Agent Conforms to: • RFC 1155, RFC 1157, RFC 1212, RFC 1213 SNMPv2c Agent based on: • RFC 1905, RFC 1906
DHCP	DHCP enabled by default. Fallback IP: 169.254.0.20

3. High Level Reader Connectivity

The Tarvos Pro reader “Text Stream Interface” (TSI) protocol contains two main channels. The first channel (port 50007) provides for a command and response protocol that will be described in more detail later. The second channel (port 50008) provides for an event channel stream for asynchronous information transmitted by the Tarvos Pro reader.

Note, there is a maximum of 8 actively connected TCP channels to the Tarvos Pro reader. This includes the TSI ports (50007 and 50008) as well as the LLRP port (5084).

3.1 Command Channel (port 50007)

The reader port 50007 is a raw TCP port. Meaning, it is not a shell, there is no echo, and there is no interpretation of special ASCII characters. It accepts ASCII characters as commands and responds with ASCII characters.

The TSI protocol requires that each command terminates with “\r\n” and each response terminates with “\r\n\r\n”. As an example, the command to get the operating mode of the reader is “setup.operating_mode”. Here is some Python code that would implement this command and process the response:

```
import socket

host_ip = "169.254.0.20"
command_channel_port = 50007
command_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
command_socket.connect((host_ip, command_channel_port))
command = "setup.operating_mode\r\n"
command_socket.sendall(command)
expected_response = "ok standby\r\n\r\n"
response = command_socket.recv(256)
if response == expected_response:
    print "Success"
```

3.2 Event Channel (port 50008)

The reader port 50008 is a raw TCP port. It is a one-way channel in that it does not process data coming into the Tarvos Pro reader. It only sends asynchronous event data out. Each asynchronous event is terminated with “\r\n\r\n”.

Upon first connecting to port 50008, the Tarvos Pro reader will output an event connection string followed by the identification of the reader channel (e.g. “event.connection id=536961760). This ID field will be used in determining which events the channel is subscribed to (please see the TSI documentation on the “reader.events.register()” function.

Please note that the “event.connection id” value will remain constant while the event channel is connected. However, this value is not guaranteed to be the same on subsequent connections to the event channel. Each time a connection to the event channel is established the value of the “event.connection id” may be different.

Here is an example of event channel processing in Python:

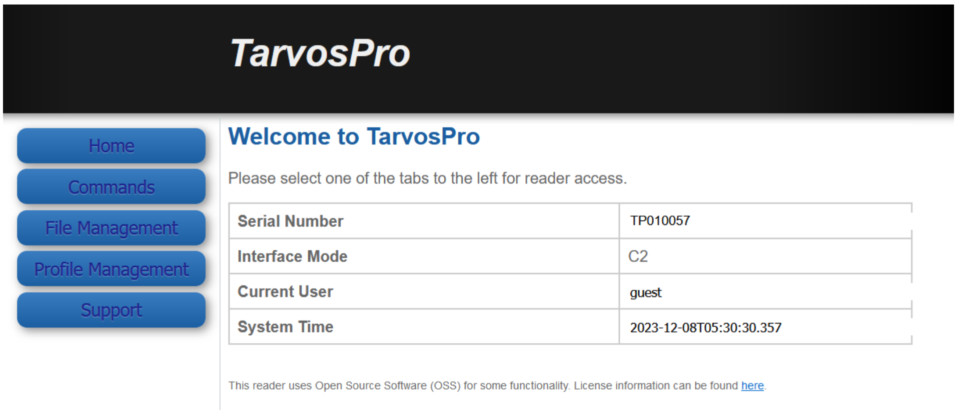
```
import socket

host_ip = "169.254.0.20"
event_channel_port = 50008
event_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
event_socket.connect((host_ip, event_channel_port))
response = event_socket.recv(256)
i1 = response.find('event.connection id =') + len('event.connection id =')
i2 = response.find('\r\n\r\n')
id = int(response[i1:i2])
print "Channel ID: ", id
while 1:
    response = event_socket.recv(256)
    print response
```

4. Web Description

4.1 Welcome Page

On the “Welcome Page”, click on one of the tabs on the left to navigate through the interface.



Welcome to TarvosPro

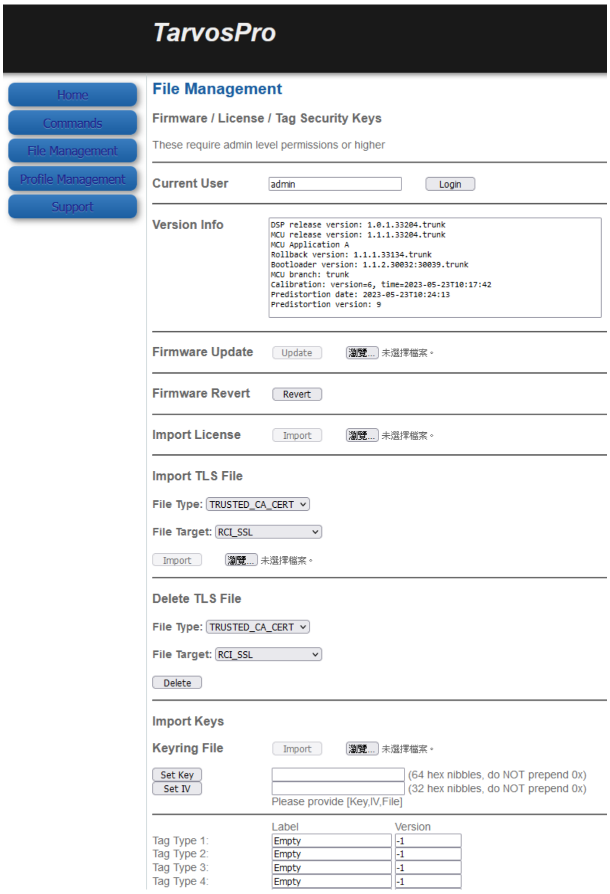
Please select one of the tabs to the left for reader access.

Serial Number	TP010057
Interface Mode	C2
Current User	guest
System Time	2023-12-08T05:30:30.357

This reader uses Open Source Software (OSS) for some functionality. License information can be found here

4.2 File Management Page

From the “File Management” page, you can perform a firmware update, firmware reverts, import a license or import a key ring.



File Management

Firmware / License / Tag Security Keys

These require admin level permissions or higher

Current User: admin

Version Info

```
DSP release version: 1.0.1.33204.trunk
MCU release version: 1.1.1.33204.trunk
MCU Application A
Rollback version: 1.1.1.33134.trunk
Bootloader version: 1.1.2.30032:30039.trunk
MCU branch: trunk
Calibration: version=6, time=2023-05-23T10:17:42
Predistortion date: 2023-05-23T10:24:13
Predistortion version: 9
```

Firmware Update 未選擇檔案。

Firmware Revert

Import License 未選擇檔案。

Import TLS File

File Type: TRUSTED_CA_CERT

File Target: RCI_SSL

未選擇檔案。

Delete TLS File

File Type: TRUSTED_CA_CERT

File Target: RCI_SSL

Import Keys

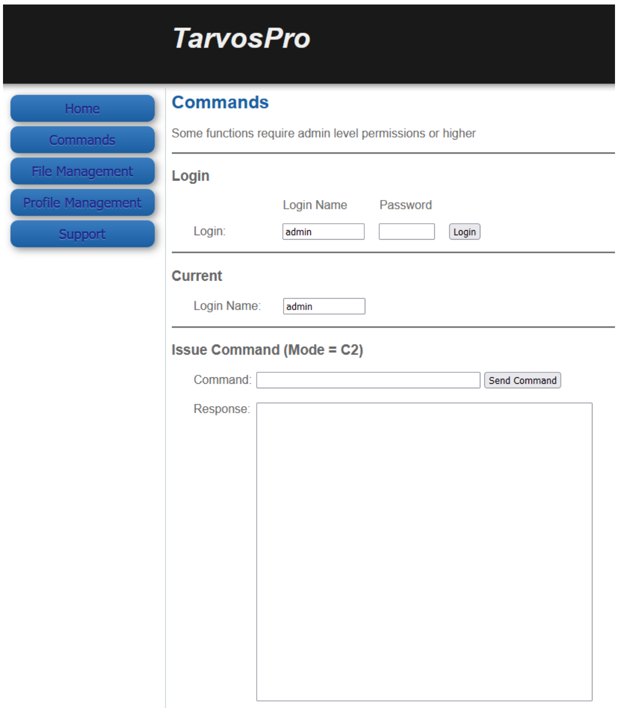
Keyring File 未選擇檔案。

(64 hex nibbles, do NOT prepend 0x)
(32 hex nibbles, do NOT prepend 0x)
Please provide [Key,IV,File]

	Label	Version
Tag Type 1:	Empty	-1
Tag Type 2:	Empty	-1
Tag Type 3:	Empty	-1
Tag Type 4:	Empty	-1

4.3 Commands

From the “Commands” page, you can enter CLI commands or type ‘help’ to get more information.



Commands

Some functions require admin level permissions or higher

Login

	Login Name	Password
Login:	admin	

Current

Login Name: admin

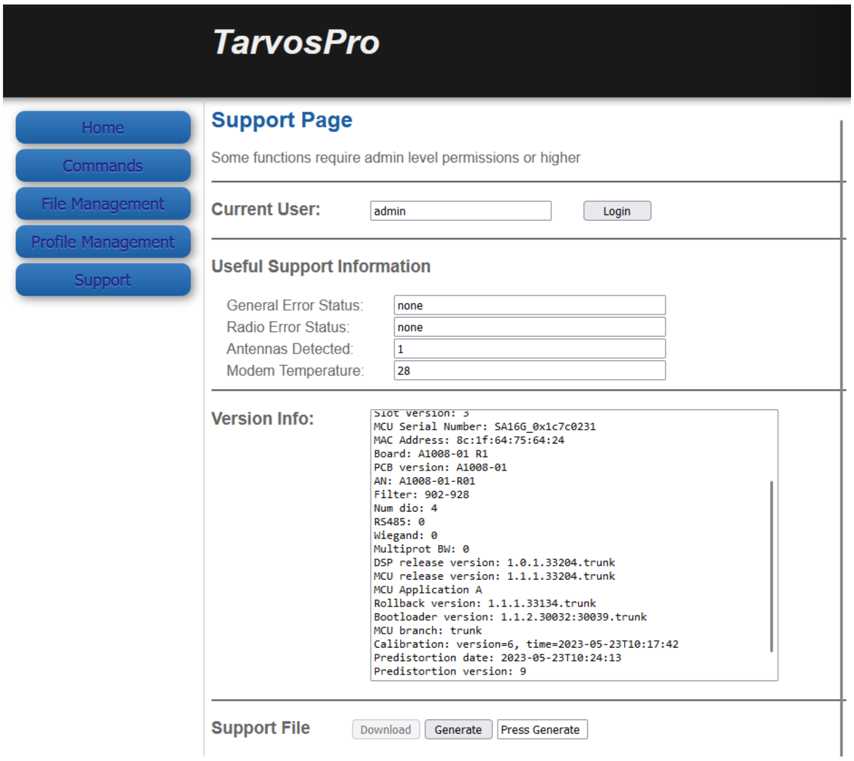
Issue Command (Mode = C2)

Command:

Response:

4.4 Support

From the “Support” page, you can download the support file and look for useful diagnostics.



Support Page

Some functions require admin level permissions or higher

Current User: admin

Useful Support Information

General Error Status:	none
Radio Error Status:	none
Antennas Detected:	1
Modem Temperature:	28

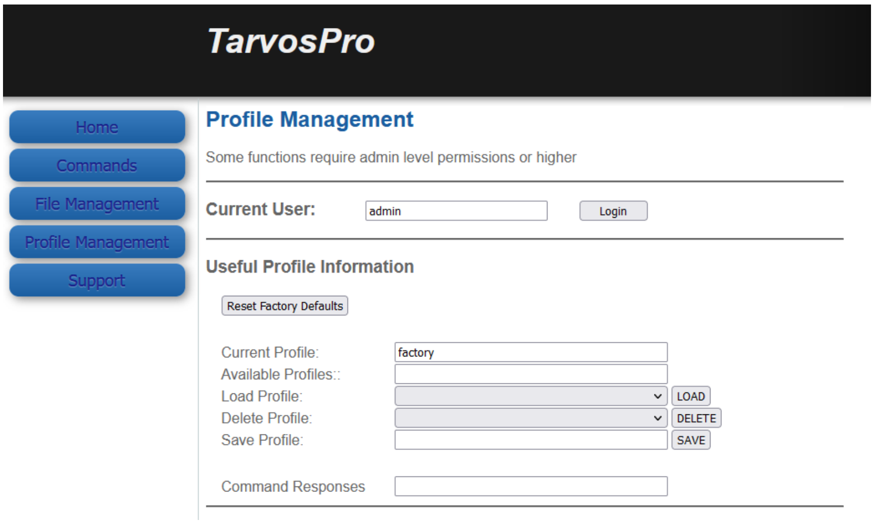
Version Info:

```
Slot version: 3
MCU Serial Number: SA16G_0x1c7c0231
MAC Address: 8c:1f:64:75:64:24
Board: A1008-01 R1
PCB version: A1008-01
AN: A1008-01-R01
Filter: 902-928
Num dio: 4
RS485: 0
Wiegand: 0
Multiprot BW: 0
DSP release version: 1.0.1.33204.trunk
MCU release version: 1.1.1.33204.trunk
MCU Application A
Rollback version: 1.1.1.33134.trunk
Bootloader version: 1.1.2.30032:30039.trunk
MCU branch: trunk
Calibration: version=6, time=2023-05-23T10:17:42
Predistortion date: 2023-05-23T10:24:13
Predistortion version: 9
```

Support File

Press Generate

4.5 Profile Management



Profile Management

Some functions require admin level permissions or higher

Current User: admin Login

Useful Profile Information

Reset Factory Defaults

Current Profile: factory

Available Profiles::

Load Profile: LOAD

Delete Profile: DELETE

Save Profile: SAVE

Command Responses

From the “Profile Management” page, you can view the current reader profile information as well as export and import configurations from and to different readers.

Please see section 5.4 for additional details on profiles.

5. TSI Interface

The Tarvos Pro reader implements the “Text Stream Interface” (TSI). The TSI is an ASCII, text-based communications protocol used to send and receive commands to the Tarvos Pro.

5.1 Permissions

TSI has two main user login names (‘guest’ and ‘admin’). Each login has permissions to perform a different set of TSI commands. Each command and the required permissions will be described later.

The default passwords for each login are the same as the login type. So, the password for ‘admin’ is ‘admin’ and the password for ‘guest’ is ‘guest’. The table below contains a list of commands that are useful for logging in and changing login information. These commands are described in more detail later in this document.

reader.login()	reader.logout()	reader.who_am_i()	reader.set_pwd()

Below is a quick example of logging in as ‘admin’ with the known default password:

```
“reader.login(admin, admin)\r\n”
```

5.2 Resetting to Factory Defaults

- TSI command on CMD port 50007
  - “`reader.profile.reset_factory_default()\r\n`”
  - Resets all variables to factory defaults and sets the profile to the Default variables.
- “ct1” repeatedly typed on serial port during boot sequence
  - Performs same function as TSI command above for resetting profile to Default.
- “ct2” repeatedly typed on serial port during boot sequence
  - Performs all the functions of a “ct1” reset
  - Resets network to DHCP
- “ct3” repeatedly typed on serial port during boot sequence
  - Performs all of the functions of a “ct2” reset
  - Forces a firmware rollback to the previously installed version

5.3 Datatypes

- Boolean
  - true or false, 0 or 1
- String
  - Variable length character array (e.g. “10.172.0.1”). Its value depends upon the specific variable of function parameters.
- Integer
  - A numeric value. Its range depends upon the specific variable or function parameters.
- Integer Array
  - An array of Integer’s (e.g. “1 2” in “antennas.mux_sequence=1 2”).
- Enum
  - A set of possible String values that a variable or function parameter must be set to (e.g. “tag_id”, “tid”, “type”, …).
- Enum Array
  - An array of Enum’s (e.g. “tag_id tid” in “tag.reporting.report_fields=tag_id tid”).
- Hex Array
  - A hex string starting with ‘0x’ (e.g. ‘0x12345678’).

5.4 Profiles

There are several TSI commands that facilitate the ability to save and load profiles on the reader (in addition to the reset_factory_default() function discussed earlier). Here are the commands:

reader.profile.delete()	reader.profile.list()	reader.profile.load()	reader.profile.save ()
reader.profile.active			

These commands allow the user to save current configurations of variables. After saving the configuration, the name given to that configuration will be active the next time the reader reboots. A total of 5 unique profiles can be saved by the Tarvos Pro.

Please see the additional information on these commands in the section that documents the TSI commands.

5.5 Reader Modes

TSI has two main operating modes ('standby' and 'active'). When in 'active' mode, the reader is actively reading tags from any configured antennas (see 'antennas.mux_sequence'). Any tags read during 'active' mode will be sent as events on the event channel (see "tag.reporting" namespace). In 'standby' mode, the reader is no longer trying to read tags.

setup.operating_mode			

To turn the reader into active mode, you can perform the following TSI command on port 50007 of the reader:

```
"setup.operating_mode=active\r\n"
```

5.6 GPIO

The Tarvos Pro reader has four digital inputs and four digital outputs. They can be used in multiple ways that are controlled through the readers TSI "dio" namespace. And they are monitored through the event channels via "event.dio.*" events.

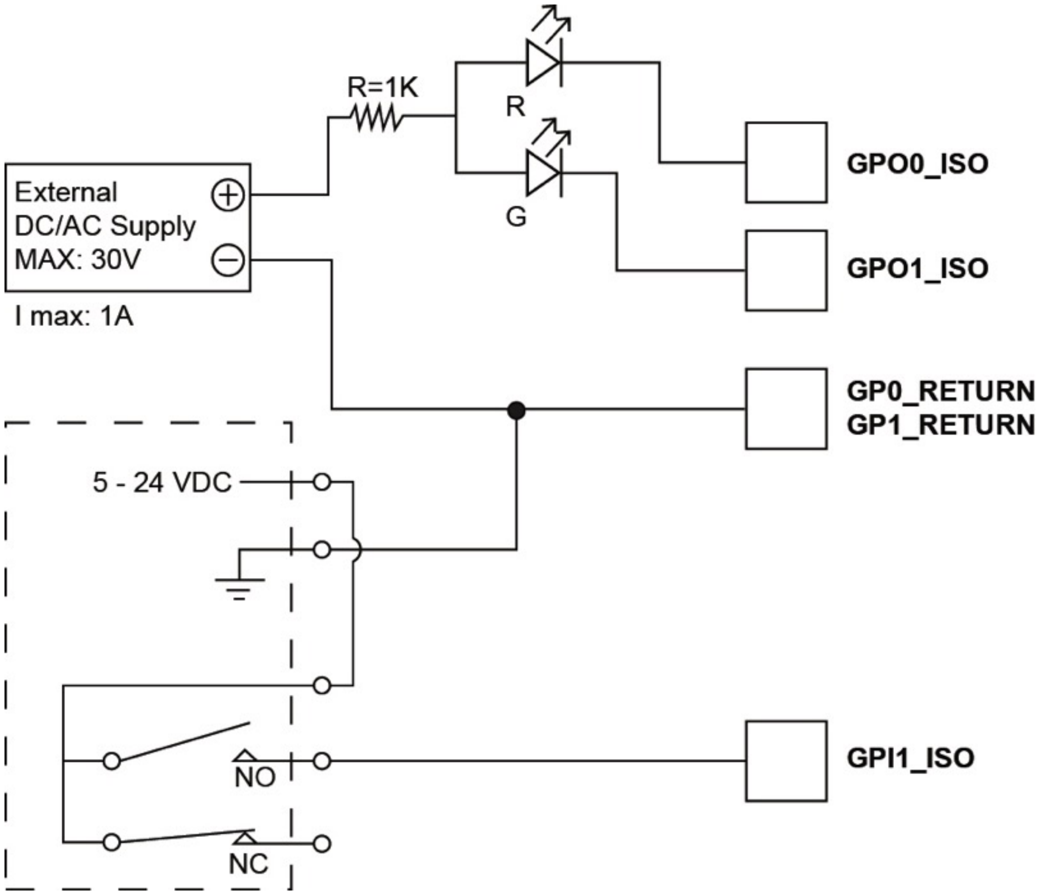


FIGURE 3 – DIO connection diagram

a. DIO Control

Here are some key control “dio” namespace variables in the TSI CLI:

dio.in.[1|2|3|4]	dio.out.[1|2|3|4]	dio.trigger_high.[1|2]	dio.trigger_low.[1|2]

The dio.in commands allow clients to monitor the inputs by polling for their values. The dio.out commands allow the state of the digital outputs to be set. All with TSI CLI commands via the command channel.

The dio.trigger commands allow clients to setup automatic operating modes of the reader based upon a change of state of an input. A change of an input from high to low or from low to high can be used to automatically put the reader into ‘standby’ or ‘active’ mode.

b. DIO Monitor

event.dio.[1|2|3|4] value=[0|1] time=<time>
event.configuration.change name=dio.out.[1|2|3|4] newvalue=[0|1] oldvalue=[0|1|None]

Whenever a DIO input value changes state, the reader will send out an event with information as to the new value of the input as well as the exact timestamp from the reader.

```
“event.dio.1 value=0 time=2017-11-10T10:06:57.710\r\n\r\n”
```

The reader also sends out configuration change events any time a variable changes state. This can be used to also monitor when a DIO output event occurs from the reader.

```
“event.configuration.change name=dio.out.2 newvalue=1 oldvalue=None time=2017-11-
10T10:05:23\r\n\r\n”
```

5.7 Tag Operations

Beyond putting the reader into ‘active’ mode and monitoring the event channel for interrogated tags, the reader can perform the tag operations suggested by the commands below.

tag.read_access_pwd()	tag.read_id()	tag.read_kill_pwd()	tag.read_tid()
tag.read_user_data()	tag.unlock()	tag.lock()	tag.write_access_pwd()
tag.write_id()	tag.write_kill_pwd()	tag.write_tid()	

Please see the documentation on the TSI commands. For a quick example:

```
“tag.write_kill_pwd(tag_id=0x294315325E9DCB8F0871F7B3, kill_pwd=0x12345678)\r\n”
```

5.8 Tag Database

The reader has a non-volatile tag database that can store up to the most recent 2^{20} (1,048,576) tags read by the reader. The TSI commands available for the tag database are shown below.

tag.db.get ()	tag.reporting.taglist_fields	tag.db.enable	tag.db.purge()
tag.db.set_acknowledged()	tag.db.next_audit_record	tag.db.get_and_purge()	

Please see the documentation on the TSI commands.

The tag database can be used to synchronously collect tags from the reader (as opposed to asynchronously via the event channel). This can be useful in situations where communications to the reader are interrupted for any reason (e.g: loss of network). Each tag has a unique audit record value that can be used to acknowledge the tag. If the client uses this mechanism to acknowledge every tag it sees, it should do the following for every tag arrival in the event channel:

```
“tag.db.set_acknowledged(true, <audit_record>)\r\n”
```

If this has been done for every tag, then if a loss of communication occurs, the following command can be used to get all unacknowledged tags in the database after reconnecting to the reader.

```
“tag.db.get(acknowledged=false)\r\n”
```

5.9 Events

The following are prefixes of events that are generated by the event channel.

- "event.dio"
  - DIO events triggered by digital input state changes
- "event.status.channel_transition"
  - Frequency change events (see setup.event_notifications)
- "event.connection.attempt"
  - Indicates an LLRP connection attempt
- "event.connection.close"
  - Indicates the close of an LLRP connection
- "event.end_of_aispec"
  - Indicates the end of an LLRP AISpec on the modem
- "event.rospec"
  - Indicates an LLRP ROSpec change
- "event.antenna"
  - Indicates an antenna mux change
- "event.tag.report"
  - Data from a tag that has been interrogated
- "event.tag.raw_arrive"
  - Data from a tag that has been interrogated (the first time)
- "event.tag.arrive"
  - Data from a tag that has been interrogated (the first time – see tag.reporting.arrive_generation)
- "event.tag.depart"
  - After a tag has no longer been interrogated for tag.reporting.depart_time, this event will be generated.
- "event.status.inventory_start"
  - The start of an inventory round
- "event.status.inventory_end"
  - The end of an inventory round
- "event.error.mcu"
  - Error events from the MCU processor
- "event.error.dsp"
  - Error events from the DSP processor
- "event.warning.mcu"
  - Warning events from the MCU processor
- "event.warning.dsp"
  - Warning events from the DSP processor
- "event.info.mcu"
  - Info events from the MCU processor
- "event.info.dsp"
  - Info events from the DSP processor

5.10 Filtering

The TSI interface provides for filtering using the Gen 2 select command. Below are the main commands for filtering.

modem.protocol.isoc.filter.f.action	modem.protocol.isoc.filter.f.enabled
modem.protocol.isoc.filter.f.length	modem.protocol.isoc.filter.f.mask
modem.protocol.isoc.filter.f.mem_bank	modem.protocol.isoc.filter.f.offset
modem.protocol.isoc.filter.f.session	modem.protocol.isoc.filtering.enabled

Where “f” is a value from 1 to 8.

Please see the documentation on the TSI commands.

For a quick example of creating a filter to read a specific 96 bit EPC identifier (“E2C06F921122338899112233”, 12 bytes or 6 words), see below:

```
# Filter 1 matches the EPC length in the PC word
modem.protocol.isoc.filter.1.action=assert_deassert
modem.protocol.isoc.filter.1.length=5
modem.protocol.isoc.filter.1.mask=0x06
modem.protocol.isoc.filter.1.mem_bank=membank_epc
modem.protocol.isoc.filter.1.offset=16
modem.protocol.isoc.filter.1.enabled=true

# Filter 2 matches the actual EPC
modem.protocol.isoc.filter.2.action=nothing_deassert
modem.protocol.isoc.filter.2.length=96
modem.protocol.isoc.filter.2.mask=E2C06F921122338899112233
modem.protocol.isoc.filter.2.mem_bank=membank_epc
modem.protocol.isoc.filter.2.offset=32
modem.protocol.isoc.filter.2.enabled=true

modem.protocol.isoc.filtering.enabled=true
```

5.11 Namespaces and Commands

The TSI interface divides the command sets into namespaces as described in the table below.

Namespace	Description
Antennas	Configures antenna powers, sequences, and displays antenna detected info.
Com	Configures and displays communication channels (serial and Ethernet)
Diag	Displays and clears diagnostics and error conditions
DIO	Displays and sets digital inputs and outputs.
Info	Model, serial number, time, and other general information.
Modem	Low level modem configuration and operations (advanced).
Reader	Reader configuration and interfaces including login and profiles.
Setup	High level reader configuration and operation control.
Tag	Tag operations and configuration including filters and event information.
Version	Reader software and hardware version information.

Each namespace has a set of commands associated with them that will be described later in this section. Each command is either a variable or a function:

- Variables

  Sets or gets a variable value. Variables are often, but not always, stored in the profiles that can be saved (see section on profiles).

  The following attributes are defined for all Variables:

  - Read Permission
    Defines permissions required to see the current state of the variable.
  - Write Permission
    Defines permissions required to modify the state of the variable.
  - Default
    Defines the initial state of the variable as well as the state set when resetting to factory defaults. If this value is NULL, it means the variable does not have a default and will not get reset when going to factory defaults.
  - Priority
    The priority determines the order that a variable gets restored during profile or factory default restoration. The higher the priority number, the earlier the variable will get restored in the restore sequence. If this value is “-1”, it means the variable does not get restored.

- Functions

  Performs an operation. Functions do not affect the state of the profile; however, they can affect the state of the reader.

  The following attributes are defined for all Functions:

  - Permission
    Defines permissions required to execute the function.
  - Min parameters
    The minimum number of parameters that need to be specified for the function to execute.
  - Max parameters
    The maximum number of parameters that can be specified for the function to execute.

a. Antennas

Variable List Table

antennas.mux_sequence	antennas.port_count
antennas.detected	antennas.diag
antennas.a.conducted_power	antennas.p.a.conducted_power
antennas.conducted_power_units	antennas.a.advanced.attenuation
antennas.a.advanced.cable_loss	antennas.a.advanced.computed_conducted_power
antennas.a.advanced.gain	antennas.a.advanced.gain_units
antennas.a.label	antennas.a.position
antennas.a.height	

antennas.mux_sequence

Read Permission	admin	Write Permission	admin
Default	0	Priority	5

```
(Integer Array) Specify a list of antenna ports to use while scanning the field
                for tags. A value of 0 means any connected antenna will be used
                while scanning the field. A non-zero value indicates both the
                antenna port numbers and the order of scanning for the antennas.
                The same antenna can be specified more than once to get more
                than one scan per sequence through all the ports.

Example:
    antennas.mux_sequence=1 2
    ok
```

antennas.port_count

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

(Integer) Number of antenna ports on the reader.

```
Example:
    antennas.port_count
    ok 2
```

antennas.detected

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

(Integer Array) List of detected antennas connected to the reader.

```
Example:
    antennas.detected
    ok 1 2
```

antennas.diag

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

(String) Detailed information about the antennas.

antennas.a.conducted_power

Read Permission	admin	Write Permission	admin
Default	270	Priority	4

(Integer) Transmit power for antenna port 'a'. Where 'a' can be between 1 and antennas.port_count.

If a protocol specific antenna power has been set then this variable will return 'PROTOCOL_SPECIFIC'. For example, if antennas.isoc.1.conducted_power is set to a different value than antennas.isob.1.conducted_power, this variable will return 'PROTOCOL_SPECIFIC'. If you set this variable in that situation, both protocol specific powers will be set with this value.

```
Example:
    antennas.1.conducted_power=250
    ok
```

antennas.p.a.conducted_power

Read Permission	admin	Write Permission	admin
Default	270	Priority	3

```
(Integer) Protocol specific transmit power for antenna port 'a'. Where 'a' can
          be between 1 and antennas.port_count. And where 'p' is determined by
          the supported protocols of the reader (see setup.valid_protocols).

          If antennas.a.conducted_power is used, the protocol specify power
          will be replaced.

Example:

    antennas.isoc.1.conducted_power=250
    ok
```

antennas.conducted_power_units

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1

```
(Enum) The displayed conducted power units (or advanced computed conducted
power).  Its in
       either 'cBm' (centibel-milliwatt) or 'mBm' (millibel-milliwatt).  On
entry, the conducted power
       level is automatically determined.  Those above 1000 are considered
milliBels, those below
       are considered in centiBels.

       2300 mBm == 230 cBm

Example:

    antennas.conducted_power_units=cBm
    ok

    antennas.1.conducted_power
    ok 230

    antennas.conducted_power_units=mBm
    ok

    antennas.1.conducted_power
    ok 2300
```

antennas.a.advanced.attenuation

Read Permission	admin	Write Permission	installer
Default	0	Priority	5

```
(Integer) Antenna attenuation (0 to 400 centiBels)

Example:
    antennas.1.cable_loss=50
    ok
```

antennas.a.advanced.cable_loss

Read Permission	admin	Write Permission	installer
Default	18	Priority	5

```
(Integer) Antenna cable loss (0 to 100 centiBels)

Example:
    antennas.1.cable_loss=50
    ok
```

antennas.a.advanced.computed_conducted_power

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

```
(Integer) Antenna computed conducted power (if antenna conducted power is 0 for
ALL protocols)

Example:
    antennas.1.computed_conducted_power
    ok 290
```

antennas.a.advanced.gain

Read Permission	admin	Write Permission	installer
Default	130	Priority	5

```
(Integer) Antenna gain (-200 to 200).  See gain_units.
          Either centi-dBi or centi-dBd.  200 -> 20.0
          dBd.

Example:
    antennas.1.gain=100
    ok
```

antennas.a.advanced.gain_units

Read Permission	admin	Write Permission	installer
Default	dBi	Priority	5

```
(Integer) Antenna gain units ('dBd', 'dBi').

Example:
    antennas.1.gain_units=dBd
    ok
```

antennas.a.label

Read Permission	admin	Write Permission	admin
Default	""	Priority	5

```
(String) Antenna label for reports (see report fields)
```

antennas.a.position

Read Permission	admin	Write Permission	admin
Default	-1	Priority	5

```
(Integer) Position of the antenna from the base point.  This
          value is in centimeters. -1 means disabled for this
          antenna.
```

antennas.a.height

Read Permission	admin	Write Permission	admin
Default	-1	Priority	5

```
(Integer) Height of the antenna from the ground.  This
          value is in centimeters. -1 means disabled for this
          antenna.
```

b. Com

Function List Table

com.network.1.set	com.network.force_ntp_sync
com.serial.set	com.network.connection_table
com.network.close_llrp	com.wiegand.debug_cmd

Variable List Table

com.modem.timeout	com.web.timeout
com.network.tcpkeepalive	com.network.1.default_gateway
com.network.1.ip_address	com.network.1.mac_address
com.network.1.method	com.network.1.settings
com.network.1.subnet_mask	com.network.ntp_servers
com.network.last_ntp_error	com.network.ntp_sync_period
com.serial.baudrate	com.serial.databits
com.serial.echo	com.serial.parity
com.serial.rawmode	com.serial.settings
com.serial.stopbits	com.network.hostname
com.network.syslog.remote.1.ip_address	com.network.syslog.remote.1.log_level
com.network.syslog.remote.1.port	com.modem.mcu_time_sync_period
com.serial.amtech.enabled	com.serial.amtech.din_ll_low
com.serial.amtech.eot_mode	com.serial.amtech.ignore_dio
com.serial.amtech.sdelay	com.network.snmp.enable
com.wiegand.interpacket_delay	com.wiegand.retransmission_interval
com.wiegand.events_only	com.network.auto_reset_link.timeout
com.network.auto_reset_link.repeat	com.network.auto_reset_link.reboot

com.network.1.set()

Permission	admin	Min parameters	1	Max parameters	4

```
Specifies the ip acquisition method, ip address, subnet mask and default gateway
    Param 1: 'method'          ==> (Enum) 'dhcp', 'static'
    Param 2: 'ip_address'      ==> (String) In 'static' method, specifies IP
                                            address (Example, 10.172.1.52)
    Param 3: 'subnet_mask'     ==> (String) In 'static' method, specifies
                                            subnet mask (e.g., 255.255.255.0)
    Param 4: 'default_gateway' ==> (String) In 'static' method, specifies
                                            default gateway (e.g., 10.172.1.1)

    Note: Use com.network.1.* for determining current values for the parameters
          set here.

Example:
    com.network.1.set(method=dhcp)
    ok
```

com.network.force_ntp_sync()

Permission	admin	Min parameters	0	Max parameters	0

```
Function forces NTP synchronization.
```

com.serial.set()

Permission	admin	Min parameters	4	Max parameters	6

```
Specifies the reader's serial port settings
    Param 1: 'baudrate'   ==> (Enum) '1200', '2400', '4800', '9600', '19200',
                                 '38400', '57600', '115200'
    Param 2: 'databits'   ==> (Enum) '7', '8'
    Param 3: 'stopbits'   ==> (Enum) '1', '2'
    Param 4: 'parity'     ==> (Enum) 'none', 'even', 'odd'
    Param 5: 'echo'       ==> (Enum) 'true', 'false' (only 'false' supported)
    Param 6: 'rawmode'    ==> (Enum) 'true', 'false' (only 'true' supported)

Example:
    com.serial.set(baudrate=57600, databits=8, stopbits=1, parity=none)
    ok
```

com.network.connection_table()

Permission	admin	Min parameters	0	Max parameters	0

```
Dumps statistics about past and present connections.

This function takes no parameters.

Example:
    com.network.connection_table()
    ok
    Cmd/Resp channels: 1
    Event channels: 1
    37 - level 5, rcvd 624, sent 597, ip 10.172.0.240, time 3495, C2_CMD
    42 - events sent 3437, bytes sent 263722, C2_EVENT
```

com.network.close_llrp()

Permission	admin	Min parameters	0	Max parameters	0

```
Close existing LLRP connection.  If a previous connection has not been
released by the client, this function can be used to force the closing
and cleaning up of that connection.  After calling this function, a
new LLRP connection can be made.

This function takes no parameters.

Example:
    com.network.close_llrp()
    ok
```

com.wiegand.debug_cmd()

Permission	admin	Min parameters	1	Max parameters	1
Send a command to wiegand chip and get the response.					

com.modem.timeout

Read Permission	admin	Write Permission	admin
Default	10000	Priority	5
(Integer) Timeout for commands sent to the modem (in milliseconds).			

com.web.timeout

Read Permission	admin	Write Permission	admin
Default	5	Priority	5
(Integer) Login timeout for web (in minutes). Minimum is 5 minutes. After this length of time the web user will be logged out and set to the default login level (see setup.default_login_level).			

com.network.tcpkeepalive

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1
(Boolean) Use TCP keep alive if needed on all TCP network connections. This is not a profile variable (not saved in profiles). It is saved in non volatile memory and requires a reboot to take effect.			

com.network.1.default_gateway

Read Permission	guest	Write Permission	factory
Default	NULL	Priority	-1
(String) Default gateway used for first network interface on reader. Read only, configureable through com.network.1.set()			

com.network.1.ip_address

Read Permission	guest	Write Permission	factory
Default	NULL	Priority	-1
(String) IP address of first network interface on reader. Read only, configurable through com.network.1.set()			

com.network.1.mac_address

Read Permission	guest	Write Permission	factory
Default	NULL	Priority	-1
(String) MAC address of the first network interface on the reader.			

com.network.1.method

Read Permission	guest	Write Permission	factory
Default	NULL	Priority	-1
(Enum) Method used to acquire IP address. Read only, configurable through com.network.1.set() Possible values: 'dhcp', 'static'			

com.network.1.settings

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(String) A list of the settings for all com.network.1.* values. Example: com.network.1.settings ok Method: DHCP IP address: 10.172.0.194 Subnet mask: 255.255.255.0 Default gateway: 10.172.0.1			

com.network.1.subnet_mask

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(String) The subnet mask used for the first network interface on the reader. Read only, configureable through com.network.1.set()			

com.network.ntp_servers

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1
(String) SNTP servers to be used to synchronize time (Reader uses simple network time protocol) Example: com.network.ntp_servers ok 0.0.0.0 0.0.0.0			

com.network.last_ntp_error

Read Permission	admin	Write Permission	none
Default	NULL	Priority	-1

```
(Float) Returns the last determined error (in milliseconds) between
        the reader and the NTP server.

Example:
    com.network.last_ntp_error
    ok 1230
```

com.network.ntp_sync_period

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1

```
(Integer) Set the NTP sync period in seconds.
          Minimum valus is 10 and maximum is 120.

Example:
    com.network.last_ntp_error
    ok 1230
```

com.serial.baudrate

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

```
(Enum) Baudrate for serial communication.
Read only. Set with com.serial.set().
Possible values: '1200', '2400', '4800', '9600', '19200', '38400', '57600',
                 '115200'
```

com.serial.databits

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

```
(Enum) Databits for serial communication.
       Read only. Set with com.serial.set().
       Possible values: '7', '8'
```

com.serial.echo

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

```
(Boolean) Echo for serial communication.
          Read only.  Always 'false'.  Serial communications does not support
          echo.
```

com.serial.parity

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

```
(Enum) Parity for serial communication.
       Read only. Set with com.serial.set().
       Possible values: 'none', 'even', 'odd'
```

com.serial.rawmode

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

```
(Boolean) Mode for serial communication.
          Read only.  Always 'true'.  Serial port does not support shell
          features.
```

com.serial.settings

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

```
(String) A list of the settings for all com.serial values.

Example:
    com.serial.settings
    ok
    baudrate = 115200
    databits = 8
    stopbits = 1
    parity = 1
    local echo = 0
    raw mode = 1
```

com.serial.stopbits

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

```
(Enum) Stopbits for serial communication.
       Read only. Set with com.serial.set().
       Possible values: '0', '1'
```

com.network.hostname

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1

```
(String) Hostname of the reader.

Example:
    com.network.hostname
    ok reader01234
```

com.network.syslog.remote.1.ip_address

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1

```
(String) IP address of remote syslog file server.
         Log messages will get sent to this IP address via syslog ports.

Example:
    com.network.syslog.remote.1.ip_address
    ok 10.172.1.195
```

com.network.syslog.remote.1.log_level

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1

```
(Enum) Maximum level to send to remote syslog server.  Anything of less severe
       level will not get sent.
       Possible values: 'Emergency', 'Alert', 'Critical', 'Error', 'Warning',
                        'Notice', 'Info', 'Debug'

Example:
    com.network.syslog.remote.1.log_level
    ok Info
```

com.network.syslog.remote.1.port

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1

```
(String) Port of remote syslog file server.
         Log messages will get sent to this port via syslog ports.

Example:
    com.network.syslog.remote.1.port
    ok 514
```

com.modem.mcu_time_sync_period

Read Permission	admin	Write Permission	admin
Default	60000	Priority	5
(Integer) Determines how often to synchronize the times between the MCU and the DSP. Its in milliseconds. A value of 0 disables it. The minimum value otherwise is 1000 milliseconds.			

com.serial.amtech.enabled

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) Enable Amtech Interface on the serial port. If disabled, the normal CLI interface will appear on the serial port. If enabled, the Amtech interface will appear on the serial port.			

com.serial.amtech.din_l1_low

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) Disabled: DIO Input 1 HIGH puts reader in active mode. DIO Input 1 LOW puts reader in standby mode. Enabled: DIO Input 1 LOW turns reader in active mode. DIO Input 1 HIGHT puts reader in standby mode.			

com.serial.amtech.eot_mode

Read Permission	admin	Write Permission	admin
Default	CR	Priority	5
(Enum) Possible values (CR, LF, CRLF, ETX) Character termination for incoming commands. CRLF is always the terminator for outgoing responses.			

com.serial.amtech.ignore_dio

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) Disabled: Use DIO to put reader in active and standby mode. In this mode, setup.operating_mode can still be used. Enabled: Ignore DIO. setup.operating_mode must be used to control the reader active/standby mode.			

com.serial.amtech.sdelay

Read Permission	admin	Write Permission	admin
Default	250	Priority	5
(Enum) Delay, in milliseconds, before going into standby mode after using DIO to put reader into standby mode.			

com.network.snmp.enable

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1
(Boolean) Enables or disables SNMP (Simple Network Management Protocol) The operation will take effect only after rebooting the reader.			

com.wiegand.interpacket_delay

Read Permission	admin	Write Permission	admin
Default	10	Priority	5
(Integer) 0 to 5000 milliseconds. Time to delay after finishing the last wiegand transmission, before sending the new transmission.			

com.wiegand.retransmission_interval

Read Permission	admin	Write Permission	admin
Default	10	Priority	5
(Integer) 1 to 255 seconds. If a tag remains in the field, this variable specifies how frequent the wiegand data will get transmitted. For a value of 5, the weigand data will get transmitted every 5 seconds until the tag departs the field.			

com.wiegand.events_only

Read Permission	admin	Write Permission	admin
Default	1	Priority	5
(Boolean) If false, wiegand events will be accompanied by a tranmission of the wiegand bits through the physical wiegand port on the reader. If true, no tranmission will occur into the physical wiegand port.			

com.network.auto_reset_link.timeout

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Timeout, in seconds, where an auto reset of the link will be attempted if no connections are attempted or used in the timeframe. 0 disables.			

com.network.auto_reset_link.repeat

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) The number of times to repeat an auto link reset before either giving up or rebooting (see com.network.auto_reset_link.reboot)			

com.network.auto_reset_link.reboot

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) After exhausting auto link repeat resets, the reader will either stop trying (this variable is 0) or will reboot the reader (this variable is 1)			

c. Diag

Function List Table

diag.netstat	diag.clear_error_status
diag.clear_radio_error_status	diag.port_return_loss
diag.system.memory	

Variable List Table

diag.ata_filter	diag.error_status
diag.radio_error_status	diag.log_llrp
diag.print_to_serial	diag.cpu_temp
diag.sockets_info	diag.error_handler.period
diag.trace_params	diag.temp_check_period
diag.macphy.status	

diag.netstat()

Permission	admin	Min parameters	0	Max parameters	0
Display network diagnostic information.					

diag.clear_error_status()

Permission	admin	Min parameters	0	Max parameters	0
Returns diag.error_status to the 'none' state.					

diag.clear_radio_error_status()

Permission	admin	Min parameters	0	Max parameters	0
Returns diag.radio_error_status to the 'none' state.					

diag.port_return_loss()

Permission	admin	Min parameters	0	Max parameters	0
Display antenna port return loss information for diagnostic purposes.					

diag.system.memory()

Permission	admin	Min parameters	0	Max parameters	0
Return information on the system memory usage.					

diag.ata_filter

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1
(Bool) Enable or disable ATA tag filter.			

diag.error_status

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(Enum) Displays error status Possible values are 'critical', 'error', 'warning', or 'none' Check reader.view_logs() to obtain additional information if 'none' is not returned.			

diag.radio_error_status

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(Enum) Displays radio error status Possible values are: 'none', 'VSWR shutdown', 'Temp shutdown', 'Internal Port shutdown', 'Error shutdown' 'Regulatory shutdown' Check reader.view_logs() to obtain additional information if 'none' is not returned.			

diag.log_llrp

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1
(Boolean) Log upstream LLRP cmd/resp packets (Do NOT set this without help from tech support).			

diag.print_to_serial

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1
(Boolean) Prints syslog and diagnostic information on the RS232 port (default is off).			

diag.cpu_temp

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(double) Displays the CPU internal temperature.			

diag.sockets_info

Read Permission	support	Write Permission	none
Default	NULL	Priority	-1
(String) Displays socket info. max total blocks (sockets), max used at any time, current total sockets, current sockets available.			

diag.error_handler.period

Read Permission	admin	Write Permission	admin
Default	60	Priority	-2
This variable is ignored. Only a placeholder for backward compatibility			

diag.trace_params

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1
(Enum Array) Diagnostic trace information. Possible values: 'macphy', 'atc', 'pwr_ctrl', 'inv_sum', 'metal', 'cog_radio'			

diag.temp_check_period

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) If greater than 0, reader will log the temperature every diag.temp_check_period minutes. The log will appear in the DEBUG logs.			

diag.macphy.status

Read Permission	support	Write Permission	none
Default	NULL	Priority	-1
(Enum) running or stopped			

d. DIO

Variable List Table

dio.in.debounce	dio.num_inputs
dio.num_outputs	dio.in.d
dio.out.d	dio.trigger_high.d
dio.trigger_low.d	

dio.in.debounce

Read Permission	admin	Write Permission	admin
Default	1000	Priority	5
(Integer) DIO debounce time in microseconds			

dio.num_inputs

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(Integer) Number of dio input pins on the reader.			

dio.num_outputs

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(Integer) Number of dio output pins on the reader.			

dio.in.d

Read Permission	admin	Write Permission	support
Default	NULL	Priority	-1
(Integer) Digital input pin value (0 or 1)			

dio.out.d

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1
(Integer) Get or set digital output pin value (0 or 1)			

dio.trigger_high.d

Read Permission	admin	Write Permission	admin
Default	none	Priority	5
(Enum) Mode of operation when digital input goes from low to high Possible values: 'none', 'active_mode', 'standby_mode'			

dio.trigger_low.d

Read Permission	admin	Write Permission	admin
Default	none	Priority	5
(Enum) Mode of operation when digital input goes from high to low Possible values: 'none', 'active_mode', 'standby_mode'			

e. Info

Variable List Table

info.model	info.serial_number
info.dsp_serial_number	info.time
info.name	info.time_reporting
info.time_zone	info.time_zone_offset

info.model

Read Permission	guest	Write Permission	factory
Default	NULL	Priority	-1

(String) Reader model.

info.serial_number

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

(String) Reader serial number.

Example:

```
    info.serial_number
    ok READER_1234567890A
```

info.dsp_serial_number

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

(String) DSP Processor serial number.

Example:

```
    info.dsp_serial_number
    ok 22189D8B1384E0E800007A89EA25AA97
```

info.time

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1

(String) Current reader time.

Example:

```
    info.time=2017-01-16T13:15:00
    ok

    info.time
    ok 2017-01-16T13:15:02.790
```

info.name

Read Permission	guest	Write Permission	admin
Default	NULL	Priority	-1

(String) Reader name (same as com.network.hostname)

info.time_reporting

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1

(String) Reports reader timezone (LOCAL or GMT)

info.time_zone

Read Permission	admin	Write Permission	admin
Default	GMT	Priority	-2

This variable is ignored. Only a placeholder for backward compatibility

info.time_zone_offset

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1

(String) Set/Get reader time zone offset. Values are specified from UTC.

```
         For example, for Eastern Standard Time:
             info.time_zone_offset = UTC-05
             ok

         You can also specify in terms of minute offsets:
             info.time_zone_offset = UTC+05:30
             ok
```

f. Modem

Function List Table

modem.protocol.isoc.read	modem.protocol.isoc.write
modem.protocol.isob.read	modem.protocol.isoc.physical.set

Variable List Table

modem.protocol.ps111.control.min_dwell_time	modem.protocol.iso10374.control.max_reads_per_dwell
modem.protocol.iso10374.control.min_dwell_time	modem.protocol.t21.control.frequency
modem.protocol.ps111.control.frequency	modem.protocol.isoc.filter.num_filters
modem.protocol.isoc.filter.f.action	modem.protocol.isoc.filter.f.enabled
modem.protocol.isoc.filter.f.length	modem.protocol.isoc.filter.f.mask
modem.protocol.isoc.filter.f.mem_bank	modem.protocol.isoc.filter.f.offset
modem.protocol.isoc.filter.f.session	modem.protocol.isoc.filtering.enabled
modem.protocol.isoc.filtering.use_session	modem.protocol.p.physical.sensitivity
modem.protocol.iso10374.control.report_tag_as_raw_hex	modem.protocol.isob.control.report_tag_as_6bitascii
modem.protocol.isob.control.6bitascii_addr	modem.protocol.isob_80k.control.report_tag_as_6bitascii
modem.protocol.isob_80k.control.6bitascii_addr	modem.protocol.ps111.control.duty_cycle_period
modem.protocol.passive.control.duty_cycle_period	modem.protocol.isoc.control.use_block_write
modem.protocol.isoc.control.auto_mac.enable	modem.protocol.isoc.control.number_slots_q
modem.protocol.isoc.control.max_incr_slots_q	modem.protocol.isoc.control.inventory_both_targets
modem.protocol.isoc.control.query_sel	modem.protocol.isoc.control.query_session
modem.protocol.isoc.control.query_target	modem.protocol.isoc.control.session_id
modem.protocol.physical.valid_modes	modem.protocol.p.physical.valid_modes
modem.protocol.p.physical.mode	modem.protocol.p.physical.a.mode
modem.diag.current_temperature	modem.protocol.isob.filter.num_filters
modem.protocol.isob.filter.f.enabled	modem.protocol.isob.filter.f.mask
modem.protocol.isob.filter.f.address	modem.protocol.isob.filter.f.data
modem.protocol.isob.filter.f.opcode	modem.protocol.isoc.physical.pilot_tone
modem.protocol.isoc.control.user_data_offset	modem.protocol.isoc.control.user_data_length
modem.protocol.isoc.physical.sideband	modem.protocol.isob.control.user_data_addr
modem.protocol.isob.control.user_data_length	modem.protocol.isoc.control.tx_atten
modem.protocol.isob_80k.control.tx_atten	modem.protocol.ps111.control.tx_atten
modem.protocol.iso10374.control.tx_atten	modem.protocol.t21.control.tx_atten
modem.control.sync.period	modem.control.sync.mode
modem.protocol.ps111.control.write_occasion	modem.protocol.ps111.control.report_occasion

modem.protocol.ps111.control.new_tag_window	modem.protocol.t21.control.ack_tag
modem.protocol.t21.control.ack_with_noop	modem.protocol.isoc.control.read_retries
modem.protocol.isoc.control.select_flag_delay	modem.control.cognitive.auto_mux
modem.control.cognitive.lbt_max_hops	modem.control.cognitive.lbt_threshold
modem.diag.mxfe_temp	modem.protocol.ps111.control.report_tag_as_raw
modem.protocol.isob_40k.control.report_as_isob	modem.protocol.isob_80k.control.report_as_isob
modem.control.vswr_warning_delay	modem.control.power_save
modem.protocol.isoc.control.wiegand.facility_code_length	modem.protocol.isoc.control.wiegand.facility_code_offset
modem.protocol.isoc.control.wiegand.user_code_length	modem.protocol.isoc.control.wiegand.user_code_offset
modem.protocol.isoc.control.wiegand.enabled	modem.protocol.iso10374.control.wiegand.enabled
modem.protocol.iso10374.control.wiegand.length	modem.protocol.iso10374.control.wiegand.offset
modem.protocol.isob.control.wiegand.enabled	modem.protocol.isob.control.wiegand.length
modem.protocol.isob.control.wiegand.offset	modem.protocol.p.weight

modem.protocol.isoc.read()

Permission	admin	Min parameters	2	Max parameters	6

```
Read information from the first tag to respond in the field.
    Param 1: 'tag_id'       ==>  (Hex Array) Identifier of tag being operated
                                             upon (for ISOC, its the EPC) (e.g.
                                             0x294315325E9DCB8F0871F7B3)
    Param 2: 'pwd'          ==>  (Hex Array) 32 bit password (for ISOC) of the
                                             tag (e.g. 0x12345678)
    Param 3: 'mem_bank'     ==>  (Enum) '0' (reserved), '1' (epc), '2' (tid), or
                                        '3' (user memory)
    Param 4: 'word_ptr'     ==>  (Integer) Address of read operation on tag
                                           (needs to be 16 bit aligned)
    Param 5: 'word_count'   ==>  (Integer) Number of words for read operation
                                           (words are 16 bits).
    Param 6: 'antenna'      ==>  (Integer) Antenna port to work with
                                           (e.g. '1', '2', ...)

    Note: This will go through all antennas and ISOC protocol and respond with
    the first tag seen (or will timeout)

Example:
    modem.protocol.isoc.read(mem_bank=1, word_ptr=2, word_count=1, antenna=2)
    ok data = 0x1234
```

modem.protocol.isoc.write()

Permission	admin	Min parameters	3	Max parameters	6

```
Write information to the to the tag specified by 'tag_id'.
    Param 1: 'tag_id'      ==>  (Hex Array) Identifier of tag being operated
                                             upon (for ISOC, its the EPC)
                                             (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 2: 'pwd'         ==>  (Hex Array) Current access password of targeted
                                             tag.
    Param 3: 'mem_bank'    ==>  (Enum) '0' (reserved), '1' (epc), '2' (tid), or
                                        '3' (user memory)
    Param 4: 'word_ptr'    ==>  (Integer) Address of read operation on tag
                                           (needs to be 16 bit aligned)
    Param 5: 'data'        ==>  (Hex Array) Data to be written to word_ptr
                                             address.  Must be 16 bit aligned in
                                             hex.
    Param 6: 'antenna'     ==>  (Integer) Antenna port to work with
                                           (e.g. '1', '2', ...)

Example:
    modem.protocol.isoc.write(tag_id=0x1234, mem_bank=1, word_ptr=2,
                              data=0x1234, antenna=2)
    ok
```

modem.protocol.isob.read()

Permission	admin	Min parameters	2	Max parameters	4

```
Read block_count blocks of 8 bytes of data at address from the chosen tag (if in
field).
'tag_id' and 'address' are required.  If 'antenna' is not specified, the read
will
attempt to read the tag on all connected ports in the antennas.mux_sequence.
    Param 1: 'tag_id'       ==>  (Hex Array) Identifier of tag being operated
                                              upon .  This must be exactly 64
bits.
                                              (for ISOB, its the unique 64 bit ID)
                                              (e.g. 0x5E9DCB8F0871F7B3)
    Param 2: 'address'      ==>  (Integer) The address in memory to start reading
8 bytes.
    Param 3: 'block_count'  ==>  (Integer) The number of 8 byte blocks to read
(max of 31 blocks at a time)
    Param 4: 'antenna'      ==>  (Integer) Antenna port to work with
                                            (e.g. '1', '2', ...)

Example:
    modem.protocol.isob.read(tag_id=0x0123456789abcdef, address=0)
    ok data = 0x0123456789abcdef
```

modem.protocol.isoc.physical.set()

Permission	admin	Min parameters	2	Max parameters	7

```
Not used.  There for backward compatibility.  Do not use.
```

modem.protocol.ps111.control.min_dwell_time

Read Permission	admin	Write Permission	admin
Default	6	Priority	5
(int) ps111 min dwell time			

modem.protocol.iso10374.control.max_reads_per_dwell

Read Permission	admin	Write Permission	admin
Default	2	Priority	5
(int) Iso10374 max reads per dwell			

modem.protocol.iso10374.control.min_dwell_time

Read Permission	admin	Write Permission	admin
Default	15	Priority	5
(int) Iso10374 min dwell time			

modem.protocol.t21.control.frequency

Read Permission	admin	Write Permission	admin
Default	911750	Priority	5
(Enum) Frequency to use when scanning for T21 tags. Possible values: 911750 - 919750			

modem.protocol.ps111.control.frequency

Read Permission	admin	Write Permission	admin
Default	915750	Priority	5
(Enum) Frequency to use when scanning for PS111 tags. Possible values: '915750'			

modem.protocol.isoc.filter.num_filters

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(Integer) Returns the number of tag filters in the modem.protocol.isoc.filter.f.* space. The 'f' value will start at 1 and go to num_filters. (see other modem.protocol.isoc.filter.f.* variables)			

modem.protocol.isoc.filter.f.action

Read Permission	admin	Write Permission	admin
Default	ASSERT_DEASSERT	Priority	5

```
(Enum) Action tag takes based on filter match (see other
       modem.protocol.isoc.filter.f variables).
       Possible values: 'assert_deassert', 'assert_nothing', 'nothing_deassert',
                        'negate_nothing', 'deassert_assert' 'deassert_nothing',
                        'nothing_assert', 'nothing_negate'
```

modem.protocol.isoc.filter.f.enabled

Read Permission	admin	Write Permission	admin
Default	false	Priority	4

```
(Boolean) Enables the ISOC mask filter.
          (see other modem.protocol.isoc.filter.f variables)
```

modem.protocol.isoc.filter.f.length

Read Permission	admin	Write Permission	admin
Default	16	Priority	5

```
(Integer) Bit length of the ISOC mask filter (0 to 255).
          Corresponds to LLRP's C1G2TagInventoryMask Parameters 'TagMask' length.
```

modem.protocol.isoc.filter.f.mask

Read Permission	admin	Write Permission	admin
Default	0x0000	Priority	5

```
(Hex Array) Bit mask for the ISOC mask filter.
            Corresponds to LLRP's C1G2TagInventoryMask Parameters 'TagMask'
            value (must match the modem.protocol.isoc.filter.f.length value).
```

modem.protocol.isoc.filter.f.mem_bank

Read Permission	admin	Write Permission	admin
Default	NOT_USED	Priority	5

```
(Enum) Memory bank used for the ISOC mask filter.
       Possible values: 'membank_epc', 'membank_tid', 'membank_user', 'not_used'
       Corresponds to LLRP's C1G2TagInventoryMask Parameters 'MB' value (1-3)
```

modem.protocol.isoc.filter.f.offset

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Bit offset for the ISOC mask filter. Corresponds to LLRP's C1G2TagInventoryMask Parameters 'Pointer' value.			

modem.protocol.isoc.filter.f.session

Read Permission	admin	Write Permission	admin
Default	NOT_USED	Priority	5
(Enum) ISOC session to use for tag communication when using the filters. Possible values: 'S0', 'S1', 'S2', 'S3', 'SL', 'not_used' Corresponds to LLRP's C1G2TagInventoryStateAwareFilterAction Parameters 'Target' value 'SL' is used by default if this filter is enabled and all session variables point to a 'not_used' value.			

modem.protocol.isoc.filtering.enabled

Read Permission	admin	Write Permission	admin
Default	false	Priority	3
(Boolean) Enables all ISOC mask filters. This enables (or disables) all of the modem.protocol.isoc.filter.f mechanisms.			

modem.protocol.isoc.filtering.use_session

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) Target the tag session inventoried flag for ISOC filter. If 'true', use the modem.protocol.isoc.control.session_id in the filters. If 'false', use the value in each filter variable (use 'SL' if not otherwise configured) Corresponds to LLRP's C1G2TagInventoryStateAwareFilterAction Parameters 'Target' value			

modem.protocol.p.physical.sensitivity

Read Permission	admin	Write Permission	support
Default		Priority	5
(Integer) Set protocol sensitivity . (only supported for ISOC and ISO10374 protocols) 'p' is one of the readers valid protocols (see setup.valid_protocols)			

modem.protocol.iso10374.control.report_tag_as_raw_hex

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) If false, convert 128 bit ISO10374 tag data into 6 bit ascii format and display, hex data otherwise. (in tag reports)			

modem.protocol.isob.control.report_tag_as_6bitascii

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) If true, convert 128 bit ISOB tag data into 6 bit ascii format and display (in tag reports)			

modem.protocol.isob.control.6bitascii_addr

Read Permission	admin	Write Permission	admin
Default	0x70	Priority	5
(Integer) User data offset for 128 bits of data to convert to 6 bit ascii if modem.protocol.isob.control.report_tag_as_6bitascii set to true.			

modem.protocol.isob_80k.control.report_tag_as_6bitascii

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) If true, convert 128 bit ISOB_80K tag data into 6 bit ascii format and display (in tag reports)			

modem.protocol.isob_80k.control.6bitascii_addr

Read Permission	admin	Write Permission	admin
Default	0x70	Priority	5
(Integer) User data offset for 128 bits of data to convert to 6 bit ascii if modem.protocol.isob_80k.control.report_tag_as_6bitascii set to true.			

modem.protocol.ps111.control.duty_cycle_period

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) If 0, duty cycle is disabled. If non-zero, use this period (in ms) when PS111 is enabled. From 0 to 10000. This is implemented with an LLRP ROSpecStartTrigger thats periodic. With an offset and period equal to this variable. The reader will scans all the ports for PS111 at a period equal to this variable.			

modem.protocol.passive.control.duty_cycle_period

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) If 0, duty cycle is disabled. If non-zero, use this period for all passive protocols. From 0 to 10000. This is implemented with an LLRP ROSpecStartTrigger thats periodic. With an offset and period equal to this variable. The reader will scans all the ports for passive protocols at a period equal to this variable.			

modem.protocol.isoc.control.use_block_write

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) Use block write capability. Corresponds to using the C1G2BlockWrite OpSpec when writing to tags (if supported by the tag).			

modem.protocol.isoc.control.auto_mac.enable

Read Permission	admin	Write Permission	admin
Default	true	Priority	5
(Boolean) Enable or disable auto mac layer. It is recommended that this value remains 'true' unless directed to change it.			

modem.protocol.isoc.control.number_slots_q

Read Permission	admin	Write Permission	admin
Default	4	Priority	5
(Integer) Initial number slots to set q to.			

modem.protocol.isoc.control.max_incr_slots_q

Read Permission	admin	Write Permission	admin
Default	15	Priority	5
(Integer) Max value increment for q (from number_slots_q).			

modem.protocol.isoc.control.inventory_both_targets

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) Inventory both targets. (make sure to disable modem.protocol.isoc.control.auto_mac.enable)			

modem.protocol.isoc.control.query_sel

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Query Select. Advanced parameter. Corresponds to "Sel" field (plus 1) in Query command (see gen2v2 specification of the Query command). 0 means the reader determines the appropriate value. 1 or 2 corresponds to ignoring SL flag (00 or 01 in the "Sel" field). 3 selects tags with the SL flag deasserted (10 of "Sel" field). 4 selects tags with the SL flag asserted (11 of "Sel" field). (make sure to disable modem.protocol.isoc.control.auto_mac.enable)			

modem.protocol.isoc.control.query_session

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Query Session. Advanced parameter. Corresponds to "Session" field (plus 1) in Query command (see gen2v2 specification of the Query command). 0 -> Uses session_id 1 -> SESSION 0 (00 in "Session" field) 2 -> SESSION 1 (01 in "Session" field) 3 -> SESSION 2 (10 in "Session" field) 4 -> SESSION 3 (11 in "Session" field) (make sure to disable modem.protocol.isoc.control.auto_mac.enable)			

modem.protocol.isoc.control.query_target

Read Permission	admin	Write Permission	admin
Default	TGT_A	Priority	5
(Enum) Query Target. Advanced parameter. Corresponds to "Target" field in Query command (see gen2v2 specification of the Query command). Possible values: 'TGT_A' or 'TGT_B' (make sure to disable modem.protocol.isoc.control.auto_mac.enable) (if modem.protocol.isoc.control.inventory_both_targets is set to 1, this field is ignored)			

modem.protocol.isoc.control.session_id

Read Permission	admin	Write Permission	admin
Default	SESSION_1	Priority	5
(Enum) Session ID to use in ISOC modes. Possible values: 'session_0', 'session_1', 'session_2', 'session_3' Used if query_session is 0. (make sure to disable modem.protocol.isoc.control.auto_mac.enable)			

modem.protocol.physical.valid_modes

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(Enum) List of ALL valid RF modes for all protocols.			

modem.protocol.p.physical.valid_modes

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(Enum) List of valid RF modes for protocol 'p'. 'p' is one of the readers valid protocols (see setup.valid_protocols) The return value is protocol specific.			

modem.protocol.p.physical.mode

Read Permission	admin	Write Permission	admin
Default		Priority	5

```
(Enum) Active RF mode for protocol p.
       'p' is one of the readers valid protocols (see setup.valid_protocols)
       Please see modem.protocol.p.physical.valid_modes for accepted values
       (protocol dependent)

       Will return ANTENNA_SPECIFIC if any of the
       modem.protocol.p.physical.a.mode value is set different from this value.

       After setting this value, all modem.protocol.p.physical.a.mode values
       will get updated to this same value.
```

modem.protocol.p.physical.a.mode

Read Permission	admin	Write Permission	admin
Default		Priority	5

```
(Enum) Active RF mode for protocol p on antenna a.
       'p' is one of the readers valid protocols (see setup.valid_protocols)
       'a' is one of the readers valid antenna ports (1 to antennas.port_count)

       Please see modem.protocol.p.physical.valid_modes for accepted values
       (protocol dependent)

       Setting modem.protocol.p.phyical.mode changes this value.
```

modem.diag.current_temperature

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

```
(Integer) Displays current modem temperature (in celsius)

Example:
    modem.diag.current_temperature
    ok 40
```

modem.protocol.isob.filter.num_filters

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

```
(Integer) Returns the number of tag filters in the modem.protocol.isob.filter.f.*
space.
          The 'f' value will start at 1 and go to num_filters.
          (see other modem.protocol.isob.filter.f.* variables)
```

modem.protocol.isob.filter.f.enabled

Read Permission	admin	Write Permission	admin
Default	false	Priority	4
(Boolean) Enables the ISOB group select filter. (see other modem.protocol.isob.filter.f variables)			

modem.protocol.isob.filter.f.mask

Read Permission	admin	Write Permission	admin
Default	0x00	Priority	5
(Hex Array) Byte mask for the ISOB group select filter (exactly one byte). For every bit in this field that is set, the corresponding byte in the data field will be compared with the corresponding data on the tag (starting at address) during the group select.			

modem.protocol.isob.filter.f.address

Read Permission	admin	Write Permission	admin
Default	0x00	Priority	5
(Hex Array) Address for the ISOB group select filter (exactly one byte). The location, on the tag, to do the comparison (of possibly 8 bytes), with the data in the filter.			

modem.protocol.isob.filter.f.data

Read Permission	admin	Write Permission	admin
Default	0x0000000000000000	Priority	5
(Hex Array) Data for the ISOB group select filter (exactly 8 bytes). The value in this field will be compared with the value on the tag (if the corresponding bit in the Byte mask is set) for the group select.			

modem.protocol.isob.filter.f.opcode

Read Permission	admin	Write Permission	admin
Default	SELECT_EQ	Priority	5
(Enum) The group select command to use for this filter. Possible values: SELECT_EQ, SELECT_NE, SELECT_GT, SELECT_LT, UNSELECT_EQ, UNSELECT_NE, UNSELECT_GT, UNSELECT_LT,			

modem.protocol.isoc.physical.pilot_tone

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) If 'false', the use of the pilot_tone(TREXT) will be automatically determined by the reader. If 'true', the pilot tone will be always be used for a higher sensitivity reader.			

modem.protocol.isoc.control.user_data_offset

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) User data offset for tag reports or tag.read_user_data.			

modem.protocol.isoc.control.user_data_length

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) User data length for tag reports or tag.read_user_data.			

modem.protocol.isoc.physical.sideband

Read Permission	admin	Write Permission	admin
Default	upper	Priority	5
(Enum) Sideband selected. Possible values: 'upper', 'lower', 'double' If the high speed license has not been purchased, this variable will always be double and cannot be changed.			

modem.protocol.isob.control.user_data_addr

Read Permission	admin	Write Permission	admin
Default	24	Priority	5
(Integer) User data address for tag reports or tag.read_user_data.			

modem.protocol.isob.control.user_data_length

Read Permission	admin	Write Permission	admin
Default	4	Priority	5
(Integer) User data length for tag reports or tag.read_user_data.			

modem.protocol.isoc.control.tx_atten

Read Permission	admin	Write Permission	admin
Default	0	Priority	-2
This variable is ignored. Only a placeholder for backward compatibility			

modem.protocol.isob_80k.control.tx_atten

Read Permission	admin	Write Permission	admin
Default	0	Priority	-2
This variable is ignored. Only a placeholder for backward compatibility			

modem.protocol.ps111.control.tx_atten

Read Permission	admin	Write Permission	admin
Default	0	Priority	-2
This variable is ignored. Only a placeholder for backward compatibility			

modem.protocol.iso10374.control.tx_atten

Read Permission	admin	Write Permission	admin
Default	0	Priority	-2
This variable is ignored. Only a placeholder for backward compatibility			

modem.protocol.t21.control.tx_atten

Read Permission	admin	Write Permission	admin
Default	0	Priority	-2
This variable is ignored. Only a placeholder for backward compatibility			

modem.control.sync.period

Read Permission	admin	Write Permission	admin
Default	50000	Priority	-2
This variable is ignored. Only a placeholder for backward compatibility			

modem.control.sync.mode

Read Permission	admin	Write Permission	admin
Default	OFF	Priority	-2
This variable is ignored. Only a placeholder for backward compatibility			

modem.protocol.ps111.control.write_occasion

Read Permission	admin	Write Permission	admin
Default	NEVER	Priority	-2
This variable is ignored. Only a placeholder for backward compatibility			

modem.protocol.ps111.control.report_occasion

Read Permission	admin	Write Permission	admin
Default	ALWAYS	Priority	-2
This variable is ignored. Only a placeholder for backward compatibility			

modem.protocol.ps111.control.new_tag_window

Read Permission	admin	Write Permission	admin
Default	60	Priority	-2
This variable is ignored. Only a placeholder for backward compatibility			

modem.protocol.t21.control.ack_tag

Read Permission	admin	Write Permission	admin
Default	1	Priority	-2
This variable is ignored. Only a placeholder for backward compatibility			

modem.protocol.t21.control.ack_with_noop

Read Permission	admin	Write Permission	admin
Default	false	Priority	-2
This variable is ignored. Only a placeholder for backward compatibility			

modem.protocol.isoc.control.read_retries

Read Permission	admin	Write Permission	admin
Default	1	Priority	5
(Integer) Read retries for ISOC read commands as well as report fields the require read commands when reader is in active mode..			

modem.protocol.isoc.control.select_flag_delay

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) If a tag has a delay more than the Gen 2 T4 time to set its flags after performing a select command, put that delay here. The value is in microseconds. The max value is 2000 microseconds and the min is 0.			

modem.control.cognitive.auto_mux

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Used to enable cognitive radio.			

modem.control.cognitive.lbt_max_hops

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Max number of LBT hops. Valid values are between 0 and 10.			

modem.control.cognitive.lbt_threshold

Read Permission	admin	Write Permission	admin
Default	-6400	Priority	5
(Integer) LBT threshold. Value is a 32-bit integer.			

modem.diag.mxfe_temp

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
Returns value of the temp read fom the MxFE			

modem.protocol.ps111.control.report_tag_as_raw

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Boolean) If false, the 24 bit PS111 serial number from the tags 256 bit data (bits 23 thru 46) will be used in the tag reports tag_id field. When false, the raw data can be obtained by registering for prot_data in the report fields. If true, reports all 256 bits in the tag_id field including the CRC.			

modem.protocol.isob_40k.control.report_as_isob

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) If true, report isob_40k tags as ISOB in "type" field of reports. If false, report them as ISOB_40K.			

modem.protocol.isob_80k.control.report_as_isob

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) If true, report isob_80k tags as ISOB in "type" field of reports. If false, report them as ISOB_80K.			

modem.control.vswr_warning_delay

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) The delay, in milliseconds, before reporting VSWR faults as warnings. If 0, any VSWR fault will result in a warning immediately. A maximum of 10 seconds is allowed (10000 milliseconds).			

modem.control.power_save

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Boolean) If set to 1 (true), modem will go to sleep when it can in order to save power. If set to 0 (false), modem will never go to sleep. If in active mode, without duty cycling, the modem will never go to sleep even with this variable. To save power, be sure to use duty cycling in active mode. Power can also be saved if active mode triggered by GPIO or an external controller only when needed.			

modem.protocol.isoc.control.wiegand.facility_code_length

Read Permission	admin	Write Permission	admin
Default	8	Priority	5
(Integer) Length of the facility code, in bits (usually 8 for 26 bit wiegand).			

modem.protocol.isoc.control.wiegand.facility_code_offset

Read Permission	admin	Write Permission	admin
Default	72	Priority	5
(Integer) Offset for the facility code in the EPC.			

modem.protocol.isoc.control.wiegand.user_code_length

Read Permission	admin	Write Permission	admin
Default	16	Priority	5
(Integer) Length of the user code, in bits (usually 16 for 26 bit wiegand).			

modem.protocol.isoc.control.wiegand.user_code_offset

Read Permission	admin	Write Permission	admin
Default	80	Priority	5
(Integer) Offset for the user code in the EPC.			

modem.protocol.isoc.control.wiegand.enabled

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) Enable/Disable wiegand output for isoc.			

modem.protocol.iso10374.control.wiegand.enabled

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) Enable/Disable wiegand output for iso10374.			

modem.protocol.iso10374.control.wiegand.length

Read Permission	admin	Write Permission	admin
Default	24	Priority	5
(Integer) Length of facility/user code, in bits (usually 24 for 26 bit wiegand) for iso10374.			

modem.protocol.iso10374.control.wiegand.offset

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Offset of facility/user code for iso10374.			

modem.protocol.isob.control.wiegand.enabled

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) Enable/Disable wiegand output for isob.			

modem.protocol.isob.control.wiegand.length

Read Permission	admin	Write Permission	admin
Default	24	Priority	5
(Integer) Length of facility/user code, in bits (usually 24 for 26 bit wiegand) for isob.			

modem.protocol.isob.control.wiegand.offset

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Offset of facility/user code for isob.			

modem.protocol.p.weight

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Protocol emphasis factor. Allowed weights are between -10 and +10, inclusive. Positive values force the selected protocol to run more often, and negative values less often.			

g. Reader

Function List Table

reader.flash_led	reader.is_alive
reader.login	reader.logout
reader.who_am_i	reader.set_pwd
reader.rollback_firmware	reader.import_license
reader.check_status	reader.register_event
reader.events.register	reader.unregister_event
reader.events.unregister	reader.events.unregister_all
reader.reboot	reader.events.trigger
reader.events.list_registered	reader.bind
reader.events.bind	reader.events.query_bind
reader.profile.delete	reader.profile.list
reader.profile.load	reader.profile.reset_factory_default
reader.profile.save	reader.profile.show_running_config
reader.view_log	reader.dsp_reset
reader.clear_licenses	reader.import_key
reader.clear_key_rings	reader.get_auth_token
reader.check_service	

Variable List Table

reader.events.utc_precision	reader.events.rssi_precision
reader.dsp_is_inited	reader.timestamp_all_events
reader.profile.active	reader.licenses
reader.events.buffer	reader.operations.o.start_trigger.mode
reader.operations.o.start_trigger.dio	reader.operations.o.start_trigger.level
reader.operations.o.start_trigger.period	reader.operations.o.start_trigger.offset
reader.operations.o.stop_trigger.mode	reader.operations.o.stop_trigger.dio
reader.operations.o.stop_trigger.level	reader.operations.o.stop_trigger.duration
reader.operations.o.protocols	reader.operations.o.mux_sequence
reader.operations.o.priority	reader.operations.num_operations
reader.description	

reader.flash_led()

Permission	admin	Min parameters	0	Max parameters	0
Flashes the reader LEDs for a few seconds to aid in reader locationing.					

reader.is_alive()

Permission	guest	Min parameters	0	Max parameters	0
Pings the reader to ensure its operational.					

reader.login()

Permission	guest	Min parameters	2	Max parameters	2
Performs reader login on current C2 cmd channel. Param 1: 'login' ==> (Enum) User to login ('guest', 'admin', 'support') Param 2: 'pwd' ==> (String) User password Default password for 'admin' is 'admin' and for 'guest', its 'guest'. Example: reader.login(login=admin, pwd=admin) ok					

reader.logout()

Permission	guest	Min parameters	0	Max parameters	0
Performs reader logout on current C2 cmd channel. Login goes back to setup.default_login_level.					

reader.who_am_i()

Permission	guest	Min parameters	0	Max parameters	0
Reports the current login level ('guest', 'admin', 'support')					

reader.set_pwd()

Permission	guest	Min parameters	2	Max parameters	3

```
Changes reader password of specified user.

    Param 1: 'login'    ==>  (Enum) User to login ('guest', 'admin')
    Param 2: 'pwd'      ==>  (String) Current user password
    Param 3: 'new_pwd'  ==>  (String) New user password

Example:
    reader.set_pwd(login=admin, pwd=password, new_pwd=new_password)
    ok
```

reader.rollback_firmware()

Permission	admin	Min parameters	0	Max parameters	0

```
Rolls back the reader firmware.
```

reader.import_license()

Permission	admin	Min parameters	1	Max parameters	1

```
Import reader license
    Param 1: 'filename'  ==>  (String) Location of file to perform license import
with.
```

reader.check_status()

Permission	admin	Min parameters	0	Max parameters	0

```
This always returns the same information and is there for backward compatibility.
Do not use on
new systems.
```

reader.register_event()

Permission	admin	Min parameters	1	Max parameters	2

```
Please use reader.events.register() instead.
```

reader.events.register()

Permission	admin	Min parameters	1	Max parameters	2

```
Registers for events on a specific C2 event channel.  Must specify 'name'.  If
an event channel has not been bound (see reader.events.bind), the 'id' must
also be specified.
    Param 1: 'id'    ==>  (Integer) C2 event channel identifier.
    Param 2: 'name'  ==>  (String) Event mask to register for channel output.

    'name' is a wildcard mask.  If any event matches the mask, it will be sent
    out that event channel.  If you set 'name' to 'event.tag', then all events
    starting with 'event.tag' will come out the event channel.  e.g.,
    'event.tag.arrive', 'event.tag.report', 'event.tag.depart', ....

When connecting to an event channel via port 50008, the first output string you
will see will be something like:
    "event.connection id = 532"

Where '532', is the event id used in this command.  For example:
    reader.events.register(id=532,name=event.tag.arrive)
    ok
```

reader.unregister_event()

Permission	admin	Min parameters	1	Max parameters	2

```
Please use reader.events.unregister() instead.
```

reader.events.unregister()

Permission	admin	Min parameters	1	Max parameters	2

```
Unregisters for events on a specific C2 event channel.  Must specify 'name'.  If
an event channel has not been bound (see reader.events.bind), the 'id' must also
be specified.
    Param 1: 'id'    ==>  (Integer) C2 event channel identifier.
    Param 2: 'name'  ==>  (String) Event mask to unregister for channel output.

For a list of currently registered for events, see
reader.events.list_registered()

When connecting to an event channel via port 50008, the first output string you
will see will be something like:
    "event.connection id = 532"

Where '532', is the event id used in this command.  For example:
    reader.events.unregister(id=532,name=event.tag.arrive)
    ok
```

reader.events.unregister_all()

Permission	admin	Min parameters	1	Max parameters	1

```
Unregisters for all events on a specific C2 event channel.  If an event channel
has not been bound (see reader.events.bind), the 'id' must be specified.
    Param 1: 'id'    ==>  (Integer) C2 event channel identifier.

For a list of currently registered for events, see
reader.events.list_registered().

When connecting to an event channel via port 50008, the first output string you
will see will be something like:
    "event.connection id = 532"

Where '532', is the event id used in this command.  For example:
    reader.events.unregister_all(id=532)
    ok
```

reader.reboot()

Permission	admin	Min parameters	0	Max parameters	0

```
Reboots the reader.
```

reader.events.trigger()

Permission	admin	Min parameters	1	Max parameters	1

```
Distributes event to all event channels with an event mask that matches.
    Param 1: 'name'    ==>  (String) Name of event to distribute

Example:
    reader.events.trigger(event.tag.special 1234)
    ok
```

reader.events.list_registered()

Permission	admin	Min parameters	0	Max parameters	1

```
Lists the events registered for one or all channel ids.
    Param 1: 'id'    ==>  (Integer) C2 event channel identifier.
```

reader.bind()

Permission	admin	Min parameters	1	Max parameters	1

```
Please use reader.events.bind() instead.
```

reader.events.bind()

Permission	admin	Min parameters	1	Max parameters	1

```
Sets the event channel for future calls to reader.events.register function that
don't specify 'id'.
    Param 1: 'id'    ==>  (Integer) C2 event channel identifier.

Example:
    reader.events.bind(id=512)
    ok
```

reader.events.query_bind()

Permission	admin	Min parameters	0	Max parameters	0

```
Query the default event channel ID for the register and unregister event
functions.
```

reader.profile.delete()

Permission	admin	Min parameters	1	Max parameters	1

```
Deletes the specified profile
    Param 1: 'filename'    ==>  (String) Name of profile to delete.
```

reader.profile.list()

Permission	admin	Min parameters	0	Max parameters	0

```
Returns list of all saved profile names.
```

reader.profile.load()

Permission	admin	Min parameters	1	Max parameters	1

```
Loads specified reader profile
    Param 1: 'filename'    ==>  (String) Name of profile to load.
```

reader.profile.reset_factory_default()

Permission	admin	Min parameters	0	Max parameters	0

```
Reset reader profile to the factory default.
```

reader.profile.save()

Permission	admin	Min parameters	1	Max parameters	1

```
Saves current reader configuration to specified profile name.
    Param 1: 'filename'    ==>  (String) Name of profile to save (can be "loaded"
                                          later)
```

reader.profile.show_running_config()

Permission	admin	Min parameters	0	Max parameters	0

```
Shows the current running configuration parameters of the reader.
```

reader.view_log()

Permission	admin	Min parameters	0	Max parameters	2

```
View some or all of the logged messages stored on the reader.
    Param 1: 'max'     ==>  (Integer) View the last 'count' logs stored (defaults
to 100).
    Param 2: 'level'   ==>  (Enum) Type of logs to display from system.  All logs
will be displayed
                                   for the level chosen and higher (i.e. warning
logs include errors).
                                   ('error', 'warning', 'info', 'debug')

Example:
    reader.view_log(max=20, level=info)
    ok
```

reader.dsp_reset()

Permission	admin	Min parameters	0	Max parameters	0

```
Resets the DSP processor.
```

reader.clear_licenses()

Permission	admin	Min parameters	0	Max parameters	0

```
Clears all of the active reader licenses.  This will disable any functionality
used that relates to the cleared licenses.
```

reader.import_key()

Permission	admin	Min parameters	1	Max parameters	2
Import reader keys Param 1: 'filename' ==> (String) Location of file to perform key file import with. Param 2: 'public_key' ==> (Boolean) True if this is targetting the public key					

reader.clear_key_rings()

Permission	admin	Min parameters	0	Max parameters	0
Clear all key rings from reader memory.					

reader.get_auth_token()

Permission	admin	Min parameters	0	Max parameters	0
Provides an authorization token for firmware update. Used automatically by the web page and by the GUI.					

reader.check_service()

Permission	admin	Min parameters	1	Max parameters	1
Not used. There for backward compatibility. Do not use.					

reader.events.utc_precision

Read Permission	admin	Write Permission	admin
Default	millisecond	Priority	5
Set time precision in reader events to "MICROSECOND" or "MILLISECOND"			

reader.events.rssi_precision

Read Permission	admin	Write Permission	admin
Default	CB	Priority	5
Set rssi precision to "CB" or "Q8". CB values are in centibel and Q8 values are 16 bits with 8 bits right of the decimal precision. Divide the Q8 number by 256 in floating point to get a high precision RSSI value in dB.			

reader.dsp_is_inited

Read Permission	guest	Write Permission	factory
Default	NULL	Priority	-1
Returns true if the DSP on the reader has been initialized.			

reader.timestamp_all_events

Read Permission	admin	Write Permission	admin
Default	False	Priority	5
(Boolean) Timestamp all events. Add timestamp to outgoing events that are not already timestamped.			

reader.profile.active

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(String) Returns the current active profile for the reader			

reader.licenses

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(String) Gets the active reader licenses.			

reader.events.buffer

Read Permission	admin	Write Permission	admin
Default	False	Priority	5
Not used. There for backward compatibility. Do not use.			

reader.operations.o.start_trigger.mode

Read Permission	admin	Write Permission	admin
Default	none	Priority	5

```
(Enum) Trigger mode to start reader operations. Values are 'none', 'immediate',
       'operating_mode_only', 'dio_only', 'periodic_only', 'dio', 'periodic'.

    'none'                : Default mode.  Reader start operation disabled.
    'immediate'           : Turn on reader operation immediately, the
start_trigger
                            will go to 'none' after starting (and will continue
                            executing).
    'operating_mode_only' : Reader operation starts when setup.operating_mode
                            is set to 'active'
    'dio_only'            : DIO will cause the operation to execute.
                            setup.operating_mode will have no effect on starting
                            reader operation.
    'periodic_only'       : The reader will immediately execute the periodic
                            operation.  setup.operating_mode will have no
                            effect on starting this reader operation.
    'dio'                 : DIO will cause the operation to execute.
                            setup.operating_mode set to 'active' will also
                            cause this operation to execute.
    'periodic'            : Periodic operation will execute when
                            setup.operating_mode is set to 'active'.

    Note: 'duration' of any operations comes from stop trigger.
```

reader.operations.o.start_trigger.dio

Read Permission	admin	Write Permission	admin
Default	1	Priority	5

(Integer) DIO number to use to start operation in dio modes.

reader.operations.o.start_trigger.level

Read Permission	admin	Write Permission	admin
Default	1	Priority	5

(Enum) DIO level to start operaton in dio modes (1 or 0)

reader.operations.o.start_trigger.period

Read Permission	admin	Write Permission	admin
Default	0	Priority	5

(Integer) Period of operation in ms (periodic modes).

reader.operations.o.start_trigger.offset

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Offset of operation in ms (periodic modes).			

reader.operations.o.stop_trigger.mode

Read Permission	admin	Write Permission	admin
Default	none	Priority	5

```
(Enum) Trigger to stop reader operations. Values are 'none', 'immediate',
       'operating_mode_only', 'dio_only', 'duration_only', 'dio', 'duration'.

    'none'                : Default mode.  Reader operaton stop trigger disabled.
    'immediate'           : Turn off reader operation immediately.  The
stop_trigger
                            will go to 'none' after stopping.
    'operating_mode_only' : Reader operation stops when setup.operating_mode
                            is set to 'standby'
    'dio_only'            : DIO will cause the operation to stop.
                            setup.operating_mode will have no effect on stopping
                            reader operation.
    'duration_only'       : The reader will stop the operation after a 'duration'
                            setup.operating_mode will have no effect on this
                            reader operation.
    'dio'                 : DIO will cause the operation to stop.
                            setup.operating_mode set to 'standby' will also
                            cause this operation to stop.
    'duration'            : The reader will stop the operation after a 'duration'
                            setup.operating_mode will also cause this operation
                            to stop.

    Note: 'period' and 'offset' from start trigger.
```

reader.operations.o.stop_trigger.dio

Read Permission	admin	Write Permission	admin
Default	1	Priority	5
(Integer) DIO number to use to stop operation in dio modes.			

reader.operations.o.stop_trigger.level

Read Permission	admin	Write Permission	admin
Default	1	Priority	5
(Integer) DIO level to stop operaton in dio modes (0 or 1)			

reader.operations.o.stop_trigger.duration

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Duration of operation in ms (duration and dio modes). In dio modes, this is a timeout if dio not received.			

reader.operations.o.protocols

Read Permission	admin	Write Permission	admin
Default	ISOC	Priority	5
(Enum Array) Operation protocols (see setup.protocols).			

reader.operations.o.mux_sequence

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer Array) Operation mux sequence (see antennas.mux_sequence).			

reader.operations.o.priority

Read Permission	admin	Write Permission	admin
Default	5	Priority	5
(Integer) Priority of reader operation. The higher the value, the higher the chance of the reader operation to preempt another reader operation.			

reader.operations.num_operations

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(Integer) Number of operatons in the reader.operatons.* namespace.			

reader.description

Read Permission	admin	Write Permission	admin
Default	""	Priority	5
Reader description that can be used in the antenna events as a report field. (see tag.reporting.report_fields, tag.reporting.arrive_fields, ...)			

h. Setup

Variable List Table

setup.advanced.hop_early	setup.advanced.do_opspecs_once
setup.operating_mode	setup.advanced.preferred_frequencies
setup.advanced.valid_frequencies	setup.sub_region
setup.region	setup.valid_regions
setup.protocols	setup.install_type
setup.default_login_level	setup.event_notifications
setup.tag_volume	setup.valid_protocols
setup.heartbeat.ip_address	setup.heartbeat.port
setup.heartbeat.interval	

setup.advanced.hop_early

Read Permission	admin	Write Permission	admin
Default	0	Priority	0
(Integer)	Setting this value to 1 in fixed frequency regulatory regions that allow multiple 'setup.advanced.preferred_frequencies', will enable the reader to always try and transmit on the first specified frequency when it can. So, if the regulatory region requires a minimum off time of 100 ms, the reader will only be off the first channel for 100 ms and it will then hop early back to the first channel. If this value is set to 0, the reader will hop through all the frequencies in the 'setup.advanced.preferred_frequencies' list one at a time as required by the regulatory region.		

setup.advanced.do_opspecs_once

Read Permission	admin	Write Permission	admin
Default	true	Priority	5
(Integer)	Determines if we continue to perform tag operations repeatedly while its "in the field". If this value is true, after a tag arrives, each user data read, or tid read, or tag security operation happens once. A value of true will provide better performance if a tag only needs to have operations performed upon it once. A value of false will force each tag operations to be performed every time that tag is singulated. A tag is "in the field" from the time it arrives and before it departs. After a tag departs, it is no longer considered "in the field" and future arrivals will cause the operation to occur again.		

setup.operating_mode

Read Permission	admin	Write Permission	admin
Default	standby	Priority	0

```
(Enum) Configures reader operating mode.
       Possible values are 'standby' 'active', or "reader_operations".

       For "reader_operations", see the reader.operations.* namespace.

Example:
    setup.operating_mode=standby
    ok
```

setup.advanced.preferred_frequencies

Read Permission	admin	Write Permission	admin
Default		Priority	5

```
(Integer Array) Sets preferred frequencies to use in regions that allow fixed
                frequencies (see setup.advanced.valid_frequencies)

Example:
    setup.advanced.preferred_frequencies=902750
    ok
```

setup.advanced.valid_frequencies

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

```
(String) Gets hop table frequencies in FHSS regions and available fixed
         frequencies in fixed frequency regions.

Fixed Example:
    setup.advanced.valid_frequencies
    ok Fixed: 902750 903250 910750 911250 911750 912250 912750 913250 913750
    914250 914750 915250 915750 916250 916750 917250 917750 918250 918750 919250
    919750 920250 920750 921250 927250

Hopping Example
    setup.advanced.valid_frequencies
    ok Hopping: 902750 903250 903750 904250 904750 905250 905750 906250 906750
    907250 907750 908250 908750 909250 909750 910250 910750 911250 911750 912250
    912750 913250 913750 914250 914750 915250 915750 916250 916750 917250 917750
    918250 918750 919250 919750 920250 920750 921250 921750 922250 922750 923250
    923750 924250 924750 925250 925750 926250 926750 927250
```

setup.sub_region

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1

```
This does nothing when you set it, and will return region otherwise.
Its there for backward compatibility, do not use for new systems.
```

setup.region

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1

```
(Enum) Display and/or set the regions (see setup.valid_regions for settable
       regions)
       Possible values are enumerated in setup.valid_regions.

Example for display:
    setup.region
    ok FCC_PART_15

Example for setting:
    setup.region=FCC_PART_90
    ok
```

setup.valid_regions

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

```
(String) Return the available regulatory regions.

Example:
    setup.valid_regions
    ok FCC_PART_90 FCC_PART_15 ETSI_EN_302_208
```

setup.protocols

Read Permission	admin	Write Permission	admin
Default	ISOC	Priority	5

```
(Enum) Display or configure tag protocols used when reader is in 'active' mode
       (see setup.valid_protocols)
       Possible values are listed in setup.valid_protocols.

Example for display:
setup.protocols
    ok ISOC

Example for setting:
    setup.protocols=ISOC
    ok
```

setup.install_type

Read Permission	admin	Write Permission	admin
Default		Priority	2

```
(Enum) Sets install type.
       Possible values: ORT, PACS, PORTAL, SIM, SHELF, POS, CONVEYOR, EAS,
             PRINTER, HANDHELD, DESKTOP,

Example for display:
    setup.install_type
    ok ORT

Example for setting:
    setup.install_type=ORT
    ok
```

setup.default_login_level

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1

```
(Enum) Sets the default login level for the reader.
       Possible values are 'admin' or 'guest'

Setting this to 'admin' prevents the need to login and all admin commands will
be available.

Example for display:
    setup.default_login_level
    ok guest

Example for setting:
    setup.default_login_level=admin
    ok
```

setup.event_notifications

Read Permission	admin	Write Permission	admin
Default	none	Priority	5

```
(Enum Array) Event notifications to enable.
             Possible values: 'none', 'channel_transition'
                   'none':                    No event notifications enabled
                   'channel_transition':      Enable
event.status.channel_transition notification.

Example for display:
    setup.event_notifications
    ok none

Example for setting:
    setup.event_notifications=channel_transition
    ok
```

setup.tag_volume

Read Permission	admin	Write Permission	admin
Default	1	Priority	5

(Enum) Configures tag volume for LLRP (goes to a default Q value of Gen2)

```
       Possible values: 1 1_4 4_8 8_16 16_32 32_64 64_128 128_256 256_512
                        512_1024 1024_2048
```

setup.valid_protocols

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

(String List of protocols valid for a reader (see setup.protocols).

Example:

```
    setup.valid_protocols
    ok ISOC ISOB
```

setup.heartbeat.ip_address

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1

IP address of remote host to send heartbeat messages.

Example:

```
    setup.heartbeat.ip_address
    ok 127.0.0.1
```

setup.heartbeat.port

Read Permission	admin	Write Permission	admin
Default	0	Priority	5

Port of remote host to send heartbeat messages.

Example:

```
    setup.heartbeat.port
    ok 8080
```

setup.heartbeat.interval

Read Permission	admin	Write Permission	admin
Default	0	Priority	5

Interval in seconds of heartbeat messages to remote host. 0 disables heartbeat messages.
setup.heartbeat.ip_address must be set before heartbeat system can be enabled

Example:

```
    setup.heartbeat.interval
    ok 30
```

i. Tag

Function List Table

tag.lock	tag.lock_access_pwd
tag.lock_id	tag.lock_kill_pwd
tag.lock_user_data	tag.read
tag.read_access_pwd	tag.read_id
tag.read_kill_pwd	tag.read_tid
tag.read_user_data	tag.unlock
tag.write	tag.write_access_pwd
tag.write_id	tag.write_kill_pwd
tag.write_user_data	tag.db.get
tag.db.set_acknowledged	tag.db.purge
tag.db.get_and_purge	tag.kill
tag.security.tag_type.pubkey.clear	tag.db.scan_tags

Variable List Table

tag.reporting.arrive_generation	tag.reporting.depart_time
tag.reporting.report_fields	tag.reporting.raw_arrive_fields
tag.reporting.arrive_fields	tag.reporting.depart_fields
tag.writeback.isoc.enable	tag.writeback.isoc.use_block_write
tag.writeback.isoc.basic.op.f.enable	tag.writeback.isoc.basic.op.f.action
tag.writeback.isoc.basic.op.f.offset	tag.writeback.isoc.basic.op.f.data
tag.writeback.isoc.basic.op.f.mask	tag.writeback.isoc.basic.op.f.value
tag.writeback.isoc.basic.filter_type	tag.writeback.isoc.basic.filter_mask
tag.writeback.isoc.basic.filter_value	tag.writeback.isob_80k.enable
tag.writeback.isob_80k.basic.op.f.enable	tag.writeback.isob_80k.basic.op.f.action
tag.writeback.isob_80k.basic.op.f.offset	tag.writeback.isob_80k.basic.op.f.data
tag.writeback.isob_80k.basic.op.f.mask	tag.writeback.isob_80k.basic.op.f.value
tag.tagspec.num_tagspecs	tag.tagspec.f.enabled
tag.tagspec.f.match	tag.tagspec.f.tagmask
tag.tagspec.f.tagdata	tag.tagspec.f.mb
tag.tagspec.f.pointer	tag.tagspec.f.name
tag.authenticate.csi_00.tam_1.enable	tag.authenticate.csi_00.tam_1.key_id
tag.authenticate.csi_00.tam_2.enable	tag.authenticate.csi_00.tam_2.key_id
tag.authenticate.csi_00.tam_2.profile	tag.authenticate.csi_00.tam_2.offset

tag.authenticate.csi_00.tam_2.block_count	tag.authenticate.csi_00.tam_2.prot_mode
tag.security.tag_type.tt.label	tag.security.tag_type.tt.version
tag.security.tag_type.tt.type	tag.security.password_authentication_enable
tag.security.tid_authentication_enable	tag.security.v2_authentication_mode
tag.security.artesp.operation	tag.reporting.taglist_fields
tag.db.require_duplicate_acks	tag.db.enable
tag.db.next_audit_record	tag.db.acknowledge_timeout
tag.db.store_tags	tag.db.create_entry_on_arrival
tag.db.max_count	tag.writeback.ps111.use_dynamic_write_data
tag.writeback.ps111.write_type	tag.writeback.ps111.write_data.tc_agency_data
tag.writeback.ps111.write_data.tc_agency_id	tag.writeback.ps111.write_data.tc_date
tag.writeback.ps111.write_data.tc_future	tag.writeback.ps111.write_data.tc_lane_id
tag.writeback.ps111.write_data.tc_plaza_id	tag.writeback.ps111.write_data.tc_seq_num
tag.writeback.ps111.write_data.tc_time	tag.writeback.ps111.write_data.tc_vehicle_class
tag.writeback.ps111.write_data.tm_date	tag.writeback.ps111.write_data.tm_reader_id
tag.writeback.ps111.write_data.tm_time	tag.writeback.ps111.write_occasion
tag.reporting.authenticate_fields	tag.security.pwd_threshold
tag.security.tag_type.pubkey.label	tag.security.tid_mask.enable
tag.reporting.estimate_fields	

tag.lock()

Permission	admin	Min parameters	1	Max parameters	5

```
Lock tag fields (first 3 parameters must be specified)
    Param 1: 'lock_fields'  ==>  (String) Any combination of 'k' (kill), 'a'
                                          (access), 'u' (user data), 'e' (epc),
                                          't' (tid)
    Param 2: 'lock_type'    ==>  (Enum) 'unsecured', 'perma_unsecured',
                                        'secured', 'perma_secured'
    Param 3: 'tag_id'       ==>  (Hex Array) Identifier of tag being operated
                                             upon (for ISOC, its the EPC)
                                             (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 4: 'pwd'          ==>  (Hex Array) 32 bit password (for ISOC) of the
                                             tag (e.g. 0x12345678)
    Param 5: 'antenna'      ==>  (Integer) Antenna port to work with
                                           (e.g. '1', '2', ...)

Note: Only ISOC supported.

Example:
    tag.lock(tag_id=0x294315325E9DCB8F0871F7B3, lock_type=UNSECURED,
             lock_fields=ae)
    ok
```

tag.lock_access_pwd()

Permission	admin	Min parameters	1	Max parameters	4

```
Lock tag access password (first 2 parameters must be specified)
    Param 1: 'lock_type'    ==>  (Enum) 'unsecured', 'perma_unsecured',
                                        'secured', 'perma_secured'
    Param 2: 'tag_id'       ==>  (Hex Array) Identifier of tag being operated
                                             upon (for ISOC, its the EPC)
                                             (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 3: 'pwd'          ==>  (Hex Array) 32 bit password (for ISOC) of the
                                             tag (e.g. 0x12345678)
    Param 4: 'antenna'      ==>  (Integer) Antenna port to work with
                                           (e.g. '1', '2', ...)

Note: Only ISOC supported.
Note: Identical to tag.lock() with lock_fields set to 'a'

Example:
    tag.lock_access_pwd(tag_id=0x294315325E9DCB8F0871F7B3, lock_type=UNSECURED)
    ok
```

tag.lock_id()

Permission	admin	Min parameters	1	Max parameters	4

```
Lock tag id (epc) (first 2 parameters must be specified)
    Param 1: 'lock_type'    ==>  (Enum) 'unsecured', 'perma_unsecured',
                                        'secured', 'perma_secured'
    Param 2: 'tag_id'       ==>  (Hex Array) Identifier of tag being operated
                                             upon (for ISOC, its the EPC)
                                             (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 3: 'pwd'          ==>  (Hex Array) 32 bit password (for ISOC) of the
                                             tag (e.g. 0x12345678)
    Param 4: 'antenna'      ==>  (Integer) Antenna port to work with
                                           (e.g. '1', '2', ...)

Note: Only ISOC supported.
Note: Identical to tag.lock() with lock_fields set to 'e'

Example:
    tag.lock_id(tag_id=0x294315325E9DCB8F0871F7B3, lock_type=UNSECURED)
    ok
```

tag.lock_kill_pwd()

Permission	admin	Min parameters	1	Max parameters	4

```
Lock tag kill pwd (epc) (first 2 parameters must be specified)
    Param 1: 'lock_type'    ==>  (Enum) 'unsecured', 'perma_unsecured',
                                        'secured', 'perma_secured'
    Param 2: 'tag_id'       ==>  (Hex Array) Identifier of tag being operated
                                             upon (for ISOC, its the EPC)
                                             (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 3: 'pwd'          ==>  (Hex Array) 32 bit password (for ISOC) of the
                                             tag (e.g. 0x12345678)
    Param 4: 'antenna'      ==>  (Integer) Antenna port to work with
                                           (e.g. '1', '2', ...)

Note: Only ISOC supported.
Note: Identical to tag.lock() with lock_fields set to 'k'

Example:
tag.lock_kill_pwd(tag_id=0x294315325E9DCB8F0871F7B3, lock_type=UNSECURED)
ok
```

tag.lock_user_data()

Permission	admin	Min parameters	1	Max parameters	4

```
Lock tag user data (epc) (first 2 parameters must be specified)
    Param 1: 'lock_type'    ==>  (Enum) 'unsecured', 'perma_unsecured',
                                        'secured', 'perma_secured'
    Param 2: 'tag_id'       ==>  (Hex Array) Identifier of tag being operated
                                             upon (for ISOC, its the EPC)
                                             (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 3: 'pwd'          ==>  (Hex Array) 32 bit password (for ISOC) of the
                                             tag (e.g. 0x12345678)
    Param 4: 'antenna'      ==>  (Integer) Antenna port to work with
                                           (e.g. '1', '2', ...)

Note: Only ISOC supported.
Note: Identical to tag.lock() with lock_fields set to 'u'

Example:
    tag.lock_user_data(tag_id=0x294315325E9DCB8F0871F7B3, lock_type=UNSECURED)
    ok
```

tag.read()

Permission	admin	Min parameters	1	Max parameters	4

```
Read information from the first tag to respond in the field.
    Param 1: 'report'       ==>  (Enum Array) 'kill_pwd', 'access_pwd',
                                              'tag_id', 'tid', 'user_data'
    Param 2: 'tag_id'       ==>  (Hex Array) Identifier of tag being operated
                                             upon (for ISOC, its the EPC)
                                             (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 3: 'pwd'          ==>  (Hex Array) 32 bit password (for ISOC) of the
                                             tag (e.g. 0x12345678)
    Param 4: 'antenna'      ==>  (Integer) Antenna port to work with
                                           (e.g. '1', '2', ...)

Note: This will go through all antennas and all protocols (see setup.protocols)
      and respond with the first tag seen (or will timeout)

Example:
    tag.read(report=tag_id tid, antenna=1)
    ok tag_id=0xB27AE47FD851275A7D01, tid=0xE2806810200000016011A040
```

tag.read_access_pwd()

Permission	admin	Min parameters	0	Max parameters	3

```
Read access password from the first tag to respond in the field.
    Param 1: 'tag_id'       ==>  (Hex Array) Identifier of tag being operated
                                             upon (for ISOC, its the EPC)
                                             (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 2: 'pwd'          ==>  (Hex Array) 32 bit password (for ISOC) of the
                                             tag (e.g. 0x12345678)
    Param 3: 'antenna'      ==>  (Integer) Antenna port to work with
                                           (e.g. '1', '2', ...)

Note: This will go through all antennas and all protocols (see setup.protocols)
      and respond with the first tag seen (or will timeout)
Note: This is the same as tag.read() with 'report' set to 'access_pwd'

Example:
    tag.read_access_pwd()
    ok access_pwd=0x12345678
```

tag.read_id()

Permission	admin	Min parameters	0	Max parameters	3

```
Read tag id from the first tag to respond in the field.
    Param 1: 'tag_id'       ==>  (Hex Array) Identifier of tag being operated
                                             upon (for ISOC, its the EPC)
                                             (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 2: 'pwd'          ==>  (Hex Array) 32 bit password (for ISOC) of the
                                             tag (e.g. 0x12345678)
    Param 3: 'antenna'      ==>  (Integer) Antenna port to work with
                                           (e.g. '1', '2', ...)

Note: This will go through all antennas and all protocols (see setup.protocols)
      and respond with the first tag seen (or will timeout)
Note: This is the same as tag.read() with 'report' set to 'tag_id'

Example:
    tag.read_id()
    ok tag_id=0xB27AE47FD851275A7D01
```

tag.read_kill_pwd()

Permission	admin	Min parameters	0	Max parameters	3

```
Read kill password from the first tag to respond in the field.
    Param 1: 'tag_id'       ==>  (Hex Array) Identifier of tag being operated
                                             upon (for ISOC, its the EPC)
                                             (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 2: 'pwd'          ==>  (Hex Array) 32 bit password (for ISOC) of the
                                             tag (e.g. 0x12345678)
    Param 3: 'antenna'      ==>  (Integer) Antenna port to work with
                                           (e.g. '1', '2', ...)

Note: This will go through all antennas and all protocols (see setup.protocols)
      and respond with the first tag seen (or will timeout)
Note: This is the same as tag.read() with 'report' set to 'kill_pwd'

Example:
    tag.read_kill_pwd()
    ok kill_pwd=0x12345678
```

tag.read_tid()

Permission	admin	Min parameters	0	Max parameters	3

```
Read tid from the first tag to respond in the field.
    Param 1: 'tag_id'       ==>  (Hex Array) Identifier of tag being operated
                                              upon (for ISOC, its the EPC)
                                              (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 2: 'pwd'          ==>  (Hex Array) 32 bit password (for ISOC) of the
                                              tag (e.g. 0x12345678)
    Param 3: 'antenna'      ==>  (Integer) Antenna port to work with
                                            (e.g. '1', '2', ...)

Note: This will go through all antennas and all protocols (see setup.protocols)
      and respond with the first tag seen (or will timeout)
Note: This is the same as tag.read() with 'report' set to 'tid'

Example:
    tag.read_tid()
    ok tid=0xE2806810200000016011A040
```

tag.read_user_data()

Permission	admin	Min parameters	0	Max parameters	3

```
Read user data from the first tag to respond in the field.
    Param 1: 'tag_id'       ==>  (Hex Array) Identifier of tag being operated
                                              upon (for ISOC, its the EPC)
                                              (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 2: 'pwd'          ==>  (Hex Array) 32 bit password (for ISOC) of the
                                              tag (e.g. 0x12345678)
    Param 3: 'antenna'      ==>  (Integer) Antenna port to work with
                                            (e.g. '1', '2', ...)

Note: This will go through all antennas and all protocols (see setup.protocols)
      and respond with the first tag seen (or will timeout)
Note: This is the same as tag.read() with 'report' set to 'user_data'

Example:
    tag.read_user_data()
    ok user_data=0x01230000000000000000000000000000000000000000000000000000
```

tag.unlock()

Permission	admin	Min parameters	1	Max parameters	4

```
Unlock tag fields (first 2 parameters must be specified)
Param 1: 'unlock_fields'  ==>  (String) Any combination of 'k' (kill),
                                        'a' (access), 'u' (user data),
                                        'e' (epc), 't' (tid)
Param 2: 'tag_id'         ==>  (Hex Array) Identifier of tag being operated upon
                                           (for ISOC, its the EPC)
                                           (e.g. 0x294315325E9DCB8F0871F7B3)
Param 3: 'pwd'            ==>  (Hex Array) 32 bit password (for ISOC) of the
                                           tag (e.g. 0x12345678)
Param 4: 'antenna'        ==>  (Integer) Antenna port to work with
                                         (e.g. '1', '2', ...)

Note: Only ISOC supported.
Note: This is the same thing as calling tag.lock() with 'lock_type' set to
          UNSECURED and 'lock_fields' set to 'unlock_fields'

Example:
    tag.unlock(tag_id=0x294315325E9DCB8F0871F7B3, unlock_fields=ae)
    ok
```

tag.write()

Permission	admin	Min parameters	1	Max parameters	10

```
Write and lock any subset of tag fields (must specify tag_id)
    Param 1:  'new_tag_id'  ==>  (Hex Array) New identifier to write to tag
                                             being operated upon (for ISOC, its
                                             the EPC)
                                             (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 2:  'kill_pwd'    ==>  (Hex Array) New kill password to write to tag
                                             being operated upon
    Param 3:  'access_pwd'  ==>  (Hex Array) New access password to write to
                                             tag being operated upon
    Param 4:  'tid'         ==>  (Hex Array) New tid to write to tag being
                                             operated upon
    Param 5:  'user_data'   ==>  (Hex Array) New user data to write to tag being
                                             operated upon
    Param 6:  'lock_fields' ==>  (Hex Array) Areas on tag to lock if desired
                                             (same as 'lock_fields' for
                                             tag.lock())
    Param 7:  'lock_type'   ==>  (Hex Array) Type of lock (same as 'lock_type'
                                             for tag.lock())
    Param 8:  'tag_id'      ==>  (Hex Array) Identifier of tag being operated
                                             upon (for ISOC, its the EPC)
                                             (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 9:  'pwd'         ==>  (Hex Array) Current access password of targeted
                                             tag.
    Param 10: 'antenna'     ==>  (Integer) Antenna port to work with
                                           (e.g. '1', '2', ...)

Note: Only ISOC supported.

Example:
    tag.write(tag_id=0x294315325E9DCB8F0871F7B3, new_tag_id=0x399DCB8F0871F7B5)
    ok
```

tag.write_access_pwd()

Permission	admin	Min parameters	1	Max parameters	5

```
Write a tag's access password (must specify tag_id)
    Param 1: 'access_pwd'    ==>  (Hex Array) New access password to write to
                                              tag being operated upon
    Param 2: 'lock_type'     ==>  (Hex Array) Type of lock (same as
                                              'lock_type' for tag.lock())
    Param 3: 'tag_id'        ==>  (Hex Array) Identifier of tag being operated
                                              upon (for ISOC, its the EPC)
                                              (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 4: 'pwd'           ==>  (Hex Array) Current access password of
                                              targeted tag.
    Param 5: 'antenna'       ==>  (Integer) Antenna port to work with
                                            (e.g. '1', '2', ...)

Note: Only ISOC supported.
Note: Same as tag.write() when specifying 'access_pwd'

Example:
    tag.write_access_pwd(tag_id=0x29325E9DCB8F0871F7B3, access_pwd=0x12345678)
    ok
```

tag.write_id()

Permission	admin	Min parameters	1	Max parameters	5

```
Write a tag's new identifier (must specify original tag_id)
    Param 1: 'new_tag_id'    ==>  (Hex Array) New tag identifier to write to
                                              tag being operated upon
    Param 2: 'lock_type'     ==>  (Hex Array) Type of lock (same as '
                                              lock_type' for tag.lock())
    Param 3: 'tag_id'        ==>  (Hex Array) Identifier of tag being operated
                                              upon (for ISOC, its the EPC)
                                              (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 4: 'pwd'           ==>  (Hex Array) Current access password of
                                              targeted tag.
    Param 5: 'antenna'       ==>  (Integer) Antenna port to work with
                                            (e.g. '1', '2', ...)

Note: ISOC only supported
Note: Same as tag.write() when specifying 'new_tag_id'

Example:
    tag.write_id(tag_id=0x29431E9DCF0871F7B3, new_tag_id=0x394315CB8F0871F7B5)
    ok
```

tag.write_kill_pwd()

Permission	admin	Min parameters	1	Max parameters	5

```
Write a tag's kill password (must specify tag_id)
    Param 1: 'kill_pwd'     ==>  (Hex Array) New kill password to write to
                                             tag being operated upon
    Param 2: 'lock_type'    ==>  (Hex Array) Type of lock (same as
                                             'lock_type' for tag.lock())
    Param 3: 'tag_id'       ==>  (Hex Array) Identifier of tag being operated
                                             upon (for ISOC, its the EPC)
                                             (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 4: 'pwd'          ==>  (Hex Array) Current access password of
                                             targeted tag.
    Param 5: 'antenna'      ==>  (Integer) Antenna port to work with
                                           (e.g. '1', '2', ...)

Note: Only ISOC supported.
Note: Same as tag.write() when specifying 'kill_pwd'

Example:
    tag.write_kill_pwd(tag_id=0x294315325E9DCB8F0871F7B3, kill_pwd=0x12345678)
    ok
```

tag.write_user_data()

Permission	admin	Min parameters	1	Max parameters	5

```
Write a tag's user data (must specify tag_id)
    Param 1: 'user_data'    ==>  (Hex Array) New user data to write to tag
                                             being operated upon
    Param 2: 'lock_type'    ==>  (Hex Array) Type of lock (same as 'lock_type'
                                             for tag.lock())
    Param 3: 'tag_id'       ==>  (Hex Array) Identifier of tag being operated
                                             upon (for ISOC, its the EPC)
                                             (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 4: 'pwd'          ==>  (Hex Array) Current access password of targeted
                                             tag.
    Param 5: 'antenna'      ==>  (Integer) Antenna port to work with
                                           (e.g. '1', '2', ...)

Note: Only ISOC supported.
Note: Same as tag.write() when specifying 'user_data'

Example:
    tag.write_user_data(tag_id=0x294315325E9DCB8F0871F7B3, user_data=0x12345678)
    ok
```

tag.db.get()

Permission	admin	Min parameters	0	Max parameters	4

```
Get the tags in the database and display them in the cmd channel using
tag.reporting.taglist_fields.
    Param 1: 'tag_id'       ==> (String) Tag ID to be displayed from the database
if it exists.
    Param 2: 'audit_record' ==> (Integer) Audit record of tag to be displayed
from the database.
    Param 3: 'acknowledged' ==> (Bool) Only get tags from the database that have
the acknolwedged
                                       flag set 'true' or 'false'
    Param 4: 'max'          ==> (Integer) Maximum number of tags to retrieve from
the database
                                          (defaults to 100 if not specified).
Limit of 10000.

If 'audit_record' and 'max' are specified, the database will display all valid
tags starting at
'audit_record' and going to 'audit_record' minus 'max'.  For example, specifying
'tag.db.get(audit_record=1000, max=100)' will get all valid tags with audit
records starting at
1000, and going down to 901 (including 901, but not 900).  Another example, if
using
'tag.db.get(audit_record=0, max=100)', tags with audit record starting at 0, then
(2^32 - 1), down
to (2^32 - 99).

Example:
    tag.db.get(max=1)
    ok
```

tag.db.set_acknowledged()

Permission	admin	Min parameters	2	Max parameters	3

```
Set the acknowledgment status of a tag in the database.
    Param 1: 'acknowledged'   ==> (Bool) true or false
    Param 2: 'audit_record'   ==> (Integer)
    Param 3: 'dup_ack_number' ==> (Integer) 0 or 1

    If param 3 is not provided, a 0 is defaulted and used.

    Param 3 is used in conjunction with tag.db.require_duplicate_acks.

Example:
    tag.db.set_acknowledge(true, 1)
    ok
```

tag.db.purge()

Permission	admin	Min parameters	0	Max parameters	0

```
Purge the tag database (all tags stored in nonvolatile and volatile memory)

Example:
    tag.db.purge()
    ok
```

tag.db.get_and_purge()

Permission	admin	Min parameters	0	Max parameters	4

```
Get the tags in the database and display them in the cmd channel using
tag.reporting.taglist_fields then purges the tag database (all tags stored in
nonvolatile and volatile memory)
    Param 1: 'tag_id'        ==> (String) Tag ID to be displayed from the database
if it exists.
    Param 2: 'audit_record' ==> (Integer) Audit record of tag to be displayed
from the database.
    Param 3: 'acknowledged' ==> (Bool) Only get tags from the database that have
the acknolwedged
                                       flag set 'true' or 'false'
    Param 4: 'max'           ==> (Integer) Maximum number of tags to retrieve from
the database.

Example:
    tag.db.get_and_purge(max=1)
    ok
```

tag.kill()

Permission	admin	Min parameters	2	Max parameters	3

```
Lock tag fields (first 3 parameters must be specified)
    Param 1: 'tag_id'        ==>  (Hex Array) Identifier of tag being operated
                                              upon (for ISOC, its the EPC)
                                              (e.g. 0x294315325E9DCB8F0871F7B3)
    Param 2: 'kill_pwd'      ==>  (Hex Array) 32 bit kill password (for ISOC) of
                                              the tag (e.g. 0x12345678)
    Param 3: 'antenna'       ==>  (Integer) Optional antenna port to work with
                                            (e.g. '1', '2', ...)

Note: Only ISOC supported.

Example:
    tag.kill(tag_id=0x294315325E9DCB8F0871F7B3, kill_pwd=0x12345678)
    ok
```

tag.security.tag_type.pubkey.clear()

Permission	admin	Min parameters	0	Max parameters	0

```
Clear public key from reader.
```

tag.db.scan_tags()

Permission	admin	Min parameters	1	Max parameters	3

```
Get the tags in the field for 'ms' time.  If blocking, the
tags will be returned inline using tag.reporting.taglist_fields.
    Param 1: 'ms'      ==> (Integer) Number of milliseconds to keep the reader
active
    Param 2: 'block'   ==> (Boolean) This function wait for completion before
returning
    Param 3: 'antenna' ==> (Integer) The antenna to use for the operations
(defaults
                                     to the antennas.mux_sequence)
```

tag.reporting.arrive_generation

Read Permission	admin	Write Permission	admin
Default	no_wait	Priority	5

```
(Enum) Specify how to wait for data with regard to generating 'event.tag.arrive'
       events.
       Possible Values: 'no_wait', 'wait_for_all', 'wait_for_tid',
                        'wait_for_data'
           no_wait: Generates arrival event on first tag EPC read even if TID or
                    USER_DATA not available.
           wait_for_all: Generates arrival event only after receiving TID and
                         USER_DATA (if requested)
           wait_for_tid: Generates arrival event only after receiving TID
                         (if requested)
           wait_for_data: Generates arrival event only after receiving
                          USER_DATA (if requested)

Example:
    tag.reporting.arrive_generation=wait_for_all
    ok
```

tag.reporting.depart_time

Read Permission	admin	Write Permission	admin
Default	10000	Priority	5

```
(Integer) If a tag has not been seen within this departure time (milliseconds),
          a depart event will be generated ('event.tag.depart'). Max value is
          100000 milliseconds and minimum is 100 ms.
```

tag.reporting.report_fields

Read Permission	admin	Write Permission	admin
Default	tag_id	Priority	5

```
(Enum Array) Tag fields reported in 'event.tag.report' events
             Possible values: 'tag_id', 'tid', 'type', 'antenna', 'rssi',
                              'user_data', 'freq', 'tx_power', 'time',
                              'prot_data', 'description', v2_authentic',
                              'max_sames', 'wiegand_id'
                   'tag_id':  Always required to have this field.  Its the
                              tag_id in the report.
                   'tid':          Add the TID of the tag to the tag report.
                   'type':         Add the type of protocol of the tag read to
                                   the tag report (.e.g. ISOC, ISOB,...)
                   'antenna':      Add the antenna port to the tag report
                   'rssi':         Add the tag rssi (relative signal strength
                                   indicator) value to the tag report
                   'user_data':    Add the tag user data to the tag report
                   'freq':         Add the frequency used to read the tag to the
                                   tag report
                   'tx_power':     Add the transmit power used to read the tag
                                   to the tag report
                   'wiegand_id':   Add the wiegand_id value of the tag to report
                   'time':         Add the microsecond accurate timestamp of
                                   the time the tag was read to the tag report
                   'pc':           Adds the protocol control word to tag report
                   'xpc1':         Adds the extended protocol control word 1 to
                                   tag report
                   'xpc2':         Adds the extended protocol control word 2 to
                                   tag report
                   'description':  Fixed reader description added to report.
                                   (see reader.description variable)
                   'max_sames':    If more than one tag in the field has the same
                                   tag_id (e.g. using untraceable), this field
will
                                   be nonzero.  This field will be one less than
                                   the number of tags in the field with the same
                                   tag_id.  If 2 tags in the field with same
                                   tag_id, this value will be 1.
                   'v2_authentic': Result of V2 authentication. Possible values
are
                                   AUTHENTIC, INAUTHENTIC, UNKNOWN,
                                   GROUP_NOMATCH, GROUP_INAUTHENTIC,
                                   GROUP_AUTHENTIC, TAG_SPECIFIC_INAUTHENTIC,
                                   TAG_SPECIFIC_AUTHENTIC
                   'activate_secure_mode': Add the tag response to the ARTESP
command
                                   ACTIVATE_SECURE_MODE.
                   'authenticate_obu': Add the tag response to the ARTESP command
                                   AUTHENTICATE_OBU

Example:
    tag.reporting.report_fields=tag_id type antenna tx_power
    ok
```

tag.reporting.raw_arrive_fields

Read Permission	admin	Write Permission	admin
Default	tag_id	Priority	5

```
(Enum Array) Tag fields reported in 'event.tag.raw_arrive' events
             Possible values: 'tag_id', 'tid', 'type', 'antenna', 'rssi',
                              'user_data', 'freq', 'tx_power', 'time',
'audit_record'
                              'description'
             'tag_id':       Always required to have this field.  Its the
                             tag_id in the report.
             'tid':          Add the TID of the tag to the tag report.
             'type':         Add the type of protocol of the tag read to
                             the tag report (.e.g. ISOC, ISOB,...)
             'antenna':      Add the antenna port to the tag report
             'rssi':         Add the tag rssi (relative signal strength
                             indicator) value to the tag report
             'user_data':    Add the tag user data to the tag report
             'freq':         Add the frequency used to read the tag to
                             the tag report
             'tx_power':     Add the transmit power used to read the tag
                             to the tag report
             'time':         Add the microsecond accurate timestamp of the
                             time the tag was read to the tag report
             'audit_record': Placeholder for now.
             'description':  Fixed reader description added to report.
                             (see reader.description variable)

Example:
    tag.reporting.raw_arrive_fields=tag_id type antenna tx_power
    ok
```

tag.reporting.arrive_fields

Read Permission	admin	Write Permission	admin
Default	tag_id	Priority	5

```
(Enum Array) Tag fields reported in 'event.tag.arrive' events
             Possible values: 'tag_id', 'tid', 'type', 'antenna', 'rssi',
                              'user_data', 'freq', 'tx_power', 'time',
                              'description', 'max_sames', 'v2_authentic',
                              'wiegand_id'
             'tag_id':       Always required to have this field.  Its the
                             tag_id in the report.
             'tid':          Add the TID of the tag to the tag report.
             'type':         Add the type of protocol of the tag read to
                             the tag report (.e.g. ISOC, ISOB,...)
             'antenna':      Add the antenna port to the tag report
             'rssi':         Add the tag rssi (relative signal strength
                             indicator) value to the tag report
             'user_data':    Add the tag user data to the tag report
             'freq':         Add the frequency used to read the tag to
                             the tag report
             'tx_power':     Add the transmit power used to read the tag
                             to the tag report
             'wiegand_id':   Add the wiegand_id value of the tag to report
             'time':         Add the microsecond accurate timestamp of the
                             time the tag was read to the tag report
             'description':  Fixed reader description added to report.
                             (see reader.description variable)
             'max_sames':    If more than one tag in the field has the same
                             tag_id (e.g. using untraceable), this field will
                             be nonzero.  This field will be one less than
                             the number of tags in the field with the same
                             tag_id.  If 2 tags in the field with same
                             tag_id, this value will be 1.
             'v2_authentic': Result of V2 authentication. Possible values are
                             AUTHENTIC, INAUTHENTIC, UNKNOWN,
                             GROUP_NOMATCH, GROUP_INAUTHENTIC,
                             GROUP_AUTHENTIC, TAG_SPECIFIC_INAUTHENTIC,
                             TAG_SPECIFIC_AUTHENTIC

Example:
    tag.reporting.arrive_fields=tag_id type antenna tx_power
    ok
```

tag.reporting.depart_fields

Read Permission	admin	Write Permission	admin
Default	tag_id	Priority	5

```
(Enum Array) Tag fields reported in 'event.tag.depart' events
             Possible values: 'tag_id', 'tid', 'type', 'antenna', 'rssi',
                              'user_data', 'freq', 'tx_power', 'time',
                              'repeat', 'description', 'max_sames',
                              'v2_authentic', 'wiegand_id'
             'tag_id':       Always required to have this field.  Its the
                             tag_id in the report.
             'tid':          Add the TID of the tag to the tag report.
             'type':         Add the type of protocol of the tag read to
                             the tag report (.e.g. ISOC, ISOB,...)
             'antenna':      Add the antenna port to the tag report
             'user_data':    Add the tag user data to the tag report
             'wiegand_id':   Add the wiegand_id value of the tag to report
             'time':         Add the microsecond accurate timestamp of
                             the time the tag was read to the tag report
             'repeat':    Add the number of times the tag has been read
                             since the arrival event to the tag report.
             'description':  Fixed reader description added to report.
                             (see reader.description variable)
             'max_sames':    If more than one tag in the field has the same
                             tag_id (e.g. using untraceable), this field will
                             be nonzero.  This field will be one less than
                             the number of tags in the field with the same
                             tag_id.  If 2 tags in the field with same
                             tag_id, this value will be 1.
             'v2_authentic': Result of V2 authentication. Possible values are
                             AUTHENTIC, INAUTHENTIC, UNKNOWN,
                             GROUP_NOMATCH, GROUP_INAUTHENTIC,
                             GROUP_AUTHENTIC, TAG_SPECIFIC_INAUTHENTIC,
                             TAG_SPECIFIC_AUTHENTIC

Example:
    tag.reporting.arrive_fields=tag_id type antenna tx_power
    ok
```

tag.writeback.isoc.enable

Read Permission	admin	Write Permission	admin
Default	false	Priority	4

```
(Boolean) Enables a custom LLRP ISOC writeback capability (see
          tag.writeback.isoc.basic.op)

Example:
    tag.writeback.isoc.enable=1
    ok
```

tag.writeback.isoc.use_block_write

Read Permission	admin	Write Permission	admin
Default	false	Priority	5

```
(Boolean) Use block write capability of ISOC tags that support the non mandatory
          block write.
          Note: Be sure you are targeting tags that all support block write.

Example:
    tag.writeback.isoc.use_block_write=1
    ok
```

tag.writeback.isoc.basic.op.f.enable

Read Permission	admin	Write Permission	admin
Default	false	Priority	4

```
(Boolean) Enables a basic custom LLRP ISOC writeback capability (see
          tag.writeback.isoc.basic.op.f)

Note: 'f' wildcard in the name can be from 1-4.  Four writeback operations can
      be chained together to attempt to operate four writebacks on the same tag
      in the same handle session.

Example:
    tag.writeback.isoc.basic.op.1.enable=1
    ok
```

tag.writeback.isoc.basic.op.f.action

Read Permission	admin	Write Permission	admin
Default		Priority	5

```
(Enum) Writeback action to perform
       Possible values: 'write', 'add', 'subtract', 'time'
       'write':    Write a fixed data into a fixed offset (see
                   tag.writeback.isoc.basic.op.f.data and
                   tag.writeback.isoc.basic.op.f.offset)
       'add':      Add a fixed value to a fixed offset (see
                   tag.writeback.isoc.basic.op.f.value,
                   tag.writeback.isoc.basic.op.f.offset, and
                   tag.writeback.isoc.op.f.mask)
       'subtract': Subtract a fixed value from a fixed offset (see
                   tag.writeback.isoc.basic.op.f.value,
                   tag.writeback.isoc.basic.op.f.offset, and
                   tag.writeback.isoc.op.f.mask)
       'time':     Write the current timestamp into a fixed offset on the
                   tag (see tag.writeback.isoc.basic.op.f.offset, and
                   tag.writeback.isoc.basic.op.f.mask)

Note: 'f' wildcard in the name can be from 1-4.  Four writeback operations can
      be chained together to attempt to operate four writebacks on the same tag
      in the same handle session.
```

tag.writeback.isoc.basic.op.f.offset

Read Permission	admin	Write Permission	admin
Default	0	Priority	5

(Integer) Offset, in bytes, into the tag where the tag.writeback.isoc.basic.op.f.action will take place.

Note: 'f' wildcard in the name can be from 1-4. Four writeback operations can be chained together to attempt to operate four writebacks on the same tag in the same handle session.

tag.writeback.isoc.basic.op.f.data

Read Permission	admin	Write Permission	admin
Default	0x0000	Priority	5

(Hex Array) Data to use in for the tag.writeback.isoc.basic.op.f.action ('write' action).

Note: 'f' wildcard in the name can be from 1-4. Four writeback operations can be chained together to attempt to operate four writebacks on the same tag in the same handle session.

tag.writeback.isoc.basic.op.f.mask

Read Permission	admin	Write Permission	admin
Default	0x0000	Priority	5

(Hex Array) Mask to use in for the tag.writeback.isoc.basic.op.f.action ('add', 'subtract', and 'time' actions).

Note: 'f' wildcard in the name can be from 1-4. Four writeback operations can be chained together to attempt to operate four writebacks on the same tag in the same handle session.

tag.writeback.isoc.basic.op.f.value

Read Permission	admin	Write Permission	admin
Default	0x0000	Priority	5

(Hex Array) Value to use in for the tag.writeback.isoc.basic.op.f.action ('add', 'subtract')

Note: 'f' wildcard in the name can be from 1-4. Four writeback operations can be chained together to attempt to operate four writebacks on the same tag in the same handle session.

tag.writeback.isoc.basic.filter_type

Read Permission	admin	Write Permission	admin
Default	all	Priority	5

```
(Enum) Writeback filter type for tag.writeback.isoc.basic actions.
       Possible values: 'all', 'tid', 'uii'
       'all': The basic op actions will happen to all tags in field of view
       'tid': The basic op actions will happen to tags matching the TID as
              specified by tag.writeback.isoc.basic.filter_mask and
              tag.writeback.isoc.basic.filter_value.
       'uii': The basic op actions will happen to tags matching the EPC as
              specified by tag.writeback.isoc.basic.filter_mask and
              tag.writeback.isoc.basic.filter_value.
```

tag.writeback.isoc.basic.filter_mask

Read Permission	admin	Write Permission	admin
Default	0x0000	Priority	5

```
(Hex Array) Writeback filter mask for tag.writeback.isoc.basic.filter_type
            ('tid' and 'uii')
            Used in conjunction with tag.writeback.isoc.basic.filter_value
```

tag.writeback.isoc.basic.filter_value

Read Permission	admin	Write Permission	admin
Default	0x0000	Priority	5

```
(Hex Array) Writeback filter mask for tag.writeback.isoc.basic.filter_type
            ('tid' and 'uii')
            Used in conjunction with tag.writeback.isoc.basic.filter_mask
```

tag.writeback.isob_80k.enable

Read Permission	admin	Write Permission	admin
Default	false	Priority	4

```
(Boolean) Enables a custom LLRP ISOB_80K writeback capability (see
          tag.writeback.isob_80k.basic.op)

Example:
    tag.writeback.isob_80k.enable=1
    ok
```

tag.writeback.isob_80k.basic.op.f.enable

Read Permission	admin	Write Permission	admin
Default	false	Priority	4

```
(Boolean) Enables a basic custom LLRP ISOB_80K writeback capability (see
          tag.writeback.isob_80k.basic.op)

Note: 'f' wildcard in the name can be from 1-4.  Four writeback operations can
      be chained together to attempt to operate four writebacks on the same tag
      in the same handle session.

Example:
    tag.writeback.isob_80k.basic.op.1.enable=1
    ok
```

tag.writeback.isob_80k.basic.op.f.action

Read Permission	admin	Write Permission	admin
Default		Priority	5

```
(Enum) Writeback action to perform
       Possible values: 'write', 'add', 'subtract', 'time'
       'write':    Write a fixed data into a fixed offset (see
                   tag.writeback.isob_80k.basic.op.f.data and
                   tag.writeback.isob_80k.basic.op.f.offset)
       'add':      Add a fixed value to a fixed offset (see
                   tag.writeback.isob_80k.basic.op.f.value,
                   tag.writeback.isob_80k.basic.op.f.offset, and
                   tag.writeback.isob_80k.op.f.mask)
       'subtract': Subtract a fixed value from a fixed offset (see
                   tag.writeback.isob_80k.basic.op.f.value,
                   tag.writeback.isob_80k.basic.op.f.offset, and
                   tag.writeback.isob_80k.op.f.mask)
       'time':     Write the current timestamp into a fixed offset on the
                   tag (see tag.writeback.isob_80k.basic.op.f.offset, and
                   tag.writeback.isob_80k.basic.op.f.mask)

Note: 'f' wildcard in the name can be from 1-4.  Four writeback operations can
      be chained together to attempt to operate four writebacks on the same tag
      in the same handle session.
```

tag.writeback.isob_80k.basic.op.f.offset

Read Permission	admin	Write Permission	admin
Default	0	Priority	5

```
(Integer) Offset, in bytes, into the tag where the
          tag.writeback.isob_80k.basic.op.f.action will take place.
          Should be 8-byte aligned for the tag to support the command.

Note: 'f' wildcard in the name can be from 1-4.  Four writeback operations can
      be chained together to attempt to operate four writebacks on the same tag
      in the same handle session.
```

tag.writeback.isob_80k.basic.op.f.data

Read Permission	admin	Write Permission	admin
Default	0x0000000000000000	Priority	5

(Hex Array) Data to use in for the tag.writeback.isob_80k.basic.op.f.action
('write' action).

Note: 'f' wildcard in the name can be from 1-4. Four writeback operations can
be chained together to attempt to operate four writebacks on the same tag
in the same handle session.

tag.writeback.isob_80k.basic.op.f.mask

Read Permission	admin	Write Permission	admin
Default	0x0000000000000000	Priority	5

(Hex Array) Mask to use in for the tag.writeback.isob_80k.basic.op.f.action
('add',
'subtract', and 'time' actions).

Note: 'f' wildcard in the name can be from 1-4. Four writeback operations can
be chained together to attempt to operate four writebacks on the same tag
in the same handle session.

tag.writeback.isob_80k.basic.op.f.value

Read Permission	admin	Write Permission	admin
Default	0x0000000000000000	Priority	5

(Hex Array) Value to use in for the tag.writeback.isob_80k.basic.op.f.action
('add', 'subtract')

Note: 'f' wildcard in the name can be from 1-4. Four writeback operations can
be chained together to attempt to operate four writebacks on the same tag
in the same handle session.

tag.tagspec.num_tagspecs

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

(Integer) Returns the number of tag specs in the tag.tagspec.f.* space. The 'f' value
will start at 1 and go to num_tagspecs.
(see other tag.tagspec.f variables)

tag.tagspec.f.enabled

Read Permission	admin	Write Permission	admin
Default	false	Priority	4
(Boolean) Enables the ISOC tagspec (see other tag.tagspec.f variables)			

tag.tagspec.f.match

Read Permission	admin	Write Permission	admin
Default	false	Priority	5
(Boolean) Corresponds to LLRP's C1G2TagSpecs / C1G2TargetTag Parameters 'Match' (see other tag.tagspec.f variables)			

tag.tagspec.f.tagmask

Read Permission	admin	Write Permission	admin
Default	0x0000	Priority	5
(Hex Array) Bit mask for the tagspec tag mask. Corresponds to LLRP's C1G2TagSpecs / C1G2TargetTag Parameters 'TagMask' value			

tag.tagspec.f.tagdata

Read Permission	admin	Write Permission	admin
Default	0x0000	Priority	5
(Hex Array) Bit mask for the tagspec pattern. Corresponds to LLRP's C1G2TagSpecs / C1G2TargetTag Parameters 'TagData' value			

tag.tagspec.f.mb

Read Permission	admin	Write Permission	admin
Default	membank_epc	Priority	5
(Enum) Memory bank used for the TagSpec. Possible values: 'membank_rsvd', 'membank_epc', 'membank_tid', 'membank_user', 'not_used' Corresponds to LLRP's C1G2TagSpec / C1G2TargetTag Parameters 'MB' value (0-3)			

tag.tagspec.f.pointer

Read Permission	admin	Write Permission	admin
Default	32	Priority	5

```
(Integer) Address of first bit to match/not match.
          Corresponds to LLRP's C1G2TagSpec / C1G2TargetTag Parameters 'Pointer'
value
```

tag.tagspec.f.name

Read Permission	admin	Write Permission	admin
Default	0x0000	Priority	5

```
(Hex Array) Tag spec name.
```

tag.authenticate.csi_00.tam_1.enable

Read Permission	admin	Write Permission	admin
Default	false	Priority	5

```
(Boolean) Enable TAM1.  If enabled, a tag authentication message of type 1
          will be sent to any inventoried tag using the key id from
          tag.authenticate.csi_00.tam_1.key_id.  The result of this operation
          will appear on the event channel as event.tag.authenticate messages.

          event.tag.authenticate tag_id=0xE2C06F920000003A002E1613, csi=00,
                                 tam=1, error=0,
ichallenge=0xCFBF9A143C294938B93B,
                                 response=0x897CF2356F15DE94952ACCD149009107

          For this command, csi (cryptographic suite index) is always 00, tam is
          always 1, error of 0 meaning success.  If error is 0, then ichallenge
          will be a randomly generated interrogator challenge sent by the reader
          and the response will be the encrypted response value read back from
          the tag.

          Please see the documentation on the the C1G2 tag authenticate command
for
          more information.  As well as information from both the tag vendor and
the
          CSI specific standard.
```

tag.authenticate.csi_00.tam_1.key_id

Read Permission	admin	Write Permission	admin
Default	0	Priority	5

```
(Integer) Key ID value to use for TAM1.  Please see
          tag.authenticate.csi_00.tam_1.enable for more information.
```

tag.authenticate.csi_00.tam_2.enable

Read Permission	admin	Write Permission	admin
Default	false	Priority	5

```
(Boolean) Enable TAM2.  If enabled, a tag authentication message of type 2
          will be sent to any inventoried tag using the key id from
          tag.authenticate.csi_00.tam_2.key_id.   Only key_id 1 is currently
          supported for TAM2.

          The result of this operation will appear on the event channel as
          event.tag.authenticate messages.

          event.tag.authenticate tag_id=0xE2C06F920000003A002E1613, csi=00,
tam=2,
                                 error=0, ichallenge=0x73C08A2914E7DE067B2E,
                                 response=0x938431EA000DD04FE0A662D0CB1CBB...

          For this command, csi (cryptographic suite index) is always 00, tam is
          always 2, error of 0 meaning success.  If error is 0, then ichallenge
          will be a randomly generated interrogator challenge sent by the reader
          and the response will be the encrypted response value read back from
          the tag.

          Please see the documentation on the the C1G2 tag authenticate command
for
          more information.  As well as information from both the tag vendor and
the
          CSI specific standard.
Example:
    tag.authenticate.csi_00.tam_2.enable=1
    ok
```

tag.authenticate.csi_00.tam_2.key_id

Read Permission	admin	Write Permission	admin
Default	1	Priority	5

```
(Integer) Key ID value to use for TAM2.  Must always be 1.  Please see
          tag.authenticate.csi_00.tam_2.enable for more information.
```

tag.authenticate.csi_00.tam_2.profile

Read Permission	admin	Write Permission	admin
Default	0	Priority	5

```
(Integer) Profile value to use for TAM2.  The following profiles
          can be read back encrypted with TAM2:
              0 - EPC bank
              1 - TID
              2 - User Memory

          Up to 128 bits can be read back from tam2_offset in the
          selected profile.

          Please see tag.authenticate.csi_00.tam_2.enable for more
          information.
```

tag.authenticate.csi_00.tam_2.offset

Read Permission	admin	Write Permission	admin
Default	0	Priority	5

```
(Integer) Offset value to use for TAM2.  This is the offset in the
          profile targetted (see tam_2.profile).
          Please see tag.authenticate.csi_00.tam_2.enable for more
          information.
```

tag.authenticate.csi_00.tam_2.block_count

Read Permission	admin	Write Permission	admin
Default	1	Priority	5

```
(Integer) Block count value to use for TAM2.  Must always be 1.
          Please see tag.authenticate.csi_00.tam_2.enable for more
          information.
```

tag.authenticate.csi_00.tam_2.prot_mode

Read Permission	admin	Write Permission	admin
Default	1	Priority	5

```
(Integer) Protocol mode value to use for TAM2.  Must always be 1.
          Please see tag.authenticate.csi_00.tam_2.enable for more
          information.
```

tag.security.tag_type.tt.label

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

```
(String) This is the key ring label. (e.g. Key Ring 0xffffffff)
```

tag.security.tag_type.tt.version

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(String) This is the version of the tag type used.			

tag.security.tag_type.tt.type

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(Enum) This is the key ring type. (AES, V1, or N/A (when key ring tag type is empty))			

tag.security.password_authentication_enable

Read Permission	admin	Write Permission	admin
Default	false	Priority	4
(Bool) Enable password_authentication_enable.			

tag.security.tid_authentication_enable

Read Permission	admin	Write Permission	admin
Default	false	Priority	4
(Bool) Enable tid_authentication_enable.			

tag.security.v2_authentication_mode

Read Permission	admin	Write Permission	admin
Default	DISABLED	Priority	4

```
(Enum) V2 Tag Security Authentication Mode (see v2_authentication report field).
       Possible Values:
        DISABLED:                                Turns this authentication off
        GROUP_AUTHENTICATION_ONLY:               Performs group only
authentication of the tag
                                                 (expects untraceable to be
enabled on tag)
        TAG_SPECIFIC_AUTHENTICATION_ONLY:        Turns this authentication off
                                                 (expects untraceable to be
disabled on tag)
        GROUP_AND_TAG_SPECIFIC_AUTHENTICATION:   Turns on group authentication
retrieval of the
                                                 tag specific EPC as well as tag
specific
                                                 authentication
        GROUP_AUTHENTICATION_AND_GET_FULL_EPC:   Turns on group authentication
retrieval of the
                                                 tag specific EPC
```

tag.security.artesp.operation

Read Permission	admin	Write Permission	admin
Default	NONE	Priority	4

```
(Enum) ARTESP security operation mode. Sends the requested ARTESP command upon
tag singulation.
       Possible Values:
        NONE:                   No additional commands
        ACTIVATE_SECURE_MODE:   Sends the ACTIVATE_SECURE_MODE command
        AUTHENTICATE_OBU:       Sends the AUTHENTICATE_OBU command
```

tag.reporting.taglist_fields

Read Permission	admin	Write Permission	admin
Default	tag_id	Priority	5

```
(Enum Array) Tag fields reported in 'tag.db.get()' cmd response
             Possible values:   'tag_id', 'tid', 'user_data', 'type',
                                'time', 'antenna', 'repeat'
                                'tid_authentic', 'pw_authentic', 'acknowledged',
                                'audit_record', 'tag_type_index'
             'tag_id':          Always required to have this field.  Its the
                                tag_id in the response.
             'tid':             Add the TID of the tag to the tag response.
             'user_data':       Add the User Data of the tag to the tag response.
             'type':            Add the type of protocol of the tag read to
                                the tag response (.e.g. ISOC, ISOB,...)
             'time':            Add the microsecond accurate timestamp of the
                                time the tag was read first and last to the tag
                                response
             'antenna':         Add the antenna port to the tag response
             'repeat':          Add the number of times the tag has been read
                                since the arrival event to the tag report.
             'tid_authentic':   If tag security being used, TID authenticity
             'pw_authentic':    If tag security being used, password authenticity
             'acknowledged':    Tag has been acknowledged by
tag.db.set_acknowledged
             'audit_record':    Placeholder for now.
             'tag_type_index':  Tag security matching index.
             'dup_ack0':        Duplicate ack0 (see
tag.db.require_duplicate_acks)
             'dup_ack1':        Duplicate ack1 (see
tag.db.require_duplicate_acks)

Example:
    tag.reporting.taglist_fields=tag_id type antenna tid
    ok
```

tag.db.require_duplicate_acks

Read Permission	admin	Write Permission	admin
Default	false	Priority	5

```
        Defaults to false.

        If set to true, both "dup_ack0" and "dup_ack1" will need to be
        true in order for "acknowledged" to be true.

        If set to false, only "dup_ack0" will need to be
        true in order for "acknowledged" to be true.
```

tag.db.enable

Read Permission	admin	Write Permission	admin
Default	NULL	Priority	-1

Enable or disable the tag database on reader.

```
Example:
    tag.db.enable=1
    ok
```

tag.db.next_audit_record

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

Returns next audit record (read only)

```
Example:
    tag.db.next_audit_record
```

tag.db.acknowledge_timeout

Read Permission	admin	Write Permission	admin
Default	10	Priority	5

(Integer) Not used. There for backward compatibility.

tag.db.store_tags

Read Permission	admin	Write Permission	admin
Default	false	Priority	5

(Boolean) Not used. There for backward compatibility.

tag.db.create_entry_on_arrival

Read Permission	admin	Write Permission	admin
Default	false	Priority	5

(Boolean) Not used. There for backward compatibility.

tag.db.max_count

Read Permission	admin	Write Permission	admin
Default	50000	Priority	5

(Integer) Not used. There for backward compatibility.

tag.writeback.ps111.use_dynamic_write_data

Read Permission	admin	Write Permission	admin
Default	true	Priority	5
(Bool) If true, time, date, lane, and sequence identifiers are dynamically generated. When false, the values are determined by the variables in the tag.writeback.ps111.* namespace.			

tag.writeback.ps111.write_type

Read Permission	admin	Write Permission	admin
Default	traffic_and_toll	Priority	5
(Enum) Determines which data gets written to the tag. For tag.writeback.ps111.write_data.tc* data, use TOLL_COLLECTION. For tag.writeback.ps111.write_data.tm*, use TRAFFIC_MANAGEMENT. To use both namespaces, use TRAFFIC_AND_TOLL.			

tag.writeback.ps111.write_data.tc_agency_data

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Agency data for TOLL_COLLECTION.			

tag.writeback.ps111.write_data.tc_agency_id

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Write data for TOLL_COLLECTION.			

tag.writeback.ps111.write_data.tc_date

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Date data for TOLL_COLLECTION if use_dynamic_write_data is false.			

tag.writeback.ps111.write_data.tc_future

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Future data for TOLL_COLLECTION.			

tag.writeback.ps111.write_data.tc_lane_id

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(List of Integers) Lane ID data for TOLL_COLLECTION. A space separated list with one entry for each antenna.			

tag.writeback.ps111.write_data.tc_plaza_id

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Plaza data for TOLL_COLLECTION.			

tag.writeback.ps111.write_data.tc_seq_num

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Sequence number data for TOLL_COLLECTION if use_dynamic_write_data is false.			

tag.writeback.ps111.write_data.tc_time

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Time data for TOLL_COLLECTION if use_dynamic_write_data is false.			

tag.writeback.ps111.write_data.tc_vehicle_class

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Vehicle class data for TOLL_COLLECTION.			

tag.writeback.ps111.write_data.tm_date

Read Permission	admin	Write Permission	admin
Default	0	Priority	5
(Integer) Date data for TRAFFIC_MANAGEMENT if use_dynamic_write_data is false.			

tag.writeback.ps111.write_data.tm_reader_id

Read Permission	admin	Write Permission	admin
Default	0	Priority	5

```
(Integer) Reader ID data for TRAFFIC_MANAGEMENT.
```

tag.writeback.ps111.write_data.tm_time

Read Permission	admin	Write Permission	admin
Default	0	Priority	5

```
(Integer) Time data for TRAFFIC_MANAGEMENT if use_dynamic_write_data is false.
```

tag.writeback.ps111.write_occasion

Read Permission	admin	Write Permission	admin
Default	NEVER	Priority	5

```
(Enum) Determines when to write to PS111 tags.
       * NEVER     - Never write to tag automatically
       * ALWAYS    - Always write to tag automatically
       * WHEN_NEW  - Only when tag has first "arrived"
```

tag.reporting.authenticate_fields

Read Permission	admin	Write Permission	admin
Default	tag_id tam	Priority	5

```
(Enum Array) Tag fields reported in 'event.tag.report' events
             Possible values: 'tag_id', 'tid', 'type', 'antenna', 'rssi',
                              'user_data', 'freq', 'tx_power', 'time',
                   'tag_id':  Always required to have this field.  Its the
                              tag_id in the report.
                   'type':      Add the type of protocol of the tag read to
                                the tag report (.e.g. ISOC, ISOB,...)
                   'antenna':   Add the antenna port to the tag report
                   'rssi':      Add the tag rssi (relative signal strength
                                indicator) value to the tag report
                   'freq':      Add the frequency used to read the tag to the
                                tag report
                   'tx_power':  Add the transmit power used to read the tag
                                to the tag report
                   'time':      Add the microsecond accurate timestamp of
                                the time the tag was read to the tag report
                   'tam':       Required for this event and includes the
                                csi, tam, ichallenge, and response fields.

Example:
    tag.reporting.authentic_fields=tag_id tam time antenna
    ok
```

tag.security.pwd_threshold

Read Permission	admin	Write Permission	admin
Default	-600	Priority	5
(Integer) If 0, this variable is disabled. Otherwise its values are from -400 to -800. If enabled along with tag security operations for password or tid authentication, the tag security operations will consider a tag as inauthentic if this threshold is met or greater and the password access does not return from the tag.			

tag.security.tag_type.pubkey.label

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1
(String) This is the public key ring label.			

tag.security.tid_mask.enable

Read Permission	admin	Write Permission	admin
Default	1	Priority	5
(Boolean) If set to false (0), the tag security TID mask will not be used and any tag that has the proper version information and key index in the user data of the tag will be authenticated			

tag.reporting.estimate_fields

Read Permission	admin	Write Permission	admin
Default	tag_id position	Priority	5

```
(Enum Array) Tag fields reported in 'event.tag.estimate' events
             Possible values: 'tag_id', 'tid', 'type', 'antenna', 'rssi',
                              'user_data', 'freq', 'tx_power', 'time',
                              'repeat', 'description', 'position',
                              'position_confidence', 'height',
                              'height_confidence'
             'tag_id':       Always required to have this field.  Its the
                             tag_id in the report.
             'tid':          Add the TID of the tag to the tag report.
             'type':         Add the type of protocol of the tag read to
                             the tag report (.e.g. ISOC, ISOB,...)
             'antenna':      Add the antenna port to the tag report
             'user_data':    Add the tag user data to the tag report
             'time':         Add the microsecond accurate timestamp of
                             the time the tag was read to the tag report
             'repeat':       Add the number of times the tag has been read
                             since the arrival event to the tag report.
             'description':  Fixed reader description added to report.
                             (see reader.description variable)
             'position':     Horizontal position of the tag with respect
                             to antennas in system.
             'height':       Vertical position of the tag with respect to
                             antennas in system.

Example:
    tag.reporting.estimate_fields=tag_id position
    ok
```

j. Version

Variable List Table

version.hw	version.hw_detail
version.detail	version.rollback
version.sw_detail	version.sw

version.hw

Read Permission	guest	Write Permission	factory
Default	NULL	Priority	-1

(String) Reader hardware version.

Example:

```
    version.hw
    ok Board Serial Number: READER_012341234
    DSP Serial Number: 22189D8B1384E0E800007A89EA25AA97
    DSP Chip ID: E5040004
    Model: ReaderModel
    Security Chip Serial: 0123424DD1BEBAB2EE
```

version.hw_detail

Read Permission	guest	Write Permission	factory
Default	NULL	Priority	-1

(String) Reader hardware version (exactly same as version.hw).

version.detail

Read Permission	guest	Write Permission	factory
Default	NULL	Priority	-1

(String) Detailed reader information (combines version.sw and version.hw)

version.rollback

Read Permission	admin	Write Permission	factory
Default	NULL	Priority	-1

(String) Reader rollback software version.

Example:

```
    version.rollback
    ok 0.2.3671.trunk
```

version.sw_detail

Read Permission	guest	Write Permission	factory
Default	NULL	Priority	-1

(String) Reader software version (exactly same as version.sw).

version.sw

Read Permission	guest	Write Permission	factory
Default	NULL	Priority	-1

(String) Reader software version.

Example:

```
    version.sw
    ok DSP version: 0.2.3770.trunk
    DSP Application A
    MCU version: 0.2.3770.trunk
    MCU Application B
    Rollback version: 0.2.3671.trunk
    Bootloader version: 0.2.3563M.trunk.SELF
```

6. LLRP Description

The Tarvos Pro also supports the “Low Level Reader Protocol” (LLRP) specification produced by EPC global. Please see *Low Level Reader Protocol (LLRP)*, Version 1.1.

Extensions to LLRP

```
UploadFileStruct(SubType=0)

    Rsvd[2:0];

    Ver[2:0];

    MessageType[12:0];

    MessageLength[31:0];

    u32 MessageID[31:0];

    VendorID[31:0]

    Subtype[7:0];

    FileTransferCustom Parameter(1);



//=============================================================================


UploadFileResponseStruct(SubType=1)

    Rsvd[2:0];

    Ver[2:0];

    MessageType[12:0];

    MessageLength[31:0];

    u32 MessageID[31:0];

    VendorID[31:0]

    Subtype[7:0];

    LLRPStatusTLV Parameter(1);

    FileDataCustom Parameter(0_1);



//=============================================================================
DownloadFileStruct(SubType=2)

    Rsvd[2:0];

    Ver[2:0];

    MessageType[12:0];

    MessageLength[31:0];

    u32 MessageID[31:0];

    VendorID[31:0]

    Subtype[7:0];

    FileTransferCustom Parameter(1);

    FileDataCustom Parameter(1);



//=============================================================================


DownloadFileResponseStruct(SubType=3)

    Rsvd[2:0];

    Ver[2:0];

    MessageType[12:0];

    MessageLength[31:0];

    u32 MessageID[31:0];

    VendorID[31:0]

    Subtype[7:0];

    LLRPStatusTLV Parameter(1);



//=============================================================================
DebugEventsStruct(SubType=4)
    Rsvd[2:0];
    Ver[2:0];
    MessageType[12:0];
    MessageLength[31:0];
    u32 MessageID[31:0];
    VendorID[31:0]
    Subtype[7:0];
    MacPhyTraceCustom Parameter(0_1);
    AtcTraceCustom Parameter(0_1);
    MtlDetTraceCustom Parameter(0_1);
    CognitiveRadioTraceCustom Parameter(0_1);
    PowerControlTraceCustom Parameter(0_1);
    InventoryTraceCustom Parameter(0_1);


//============================================================================

TagDepartStruct(SubType=5)
    Rsvd[2:0];
    Ver[2:0];
    MessageType[12:0];
    MessageLength[31:0];
    u32 MessageID[31:0];
    VendorID[31:0]
    Subtype[7:0];
    TagReportDataTLV Parameter(0_N);


//============================================================================
McuDspCommandStruct(SubType=6)

    Rsvd[2:0];

    Ver[2:0];

    MessageType[12:0];

    MessageLength[31:0];

    u32 MessageID[31:0];

    VendorID[31:0]

    Subtype[7:0];

    Command[7:0];

    LoginLevel[7:0];

    DataLength[15:0];

    Data[...];



//=============================================================================

McuDspResponseStruct(SubType=7)

    Rsvd[2:0];

    Ver[2:0];

    MessageType[12:0];

    MessageLength[31:0];

    u32 MessageID[31:0];

    VendorID[31:0]

    Subtype[7:0];

    Response[7:0];

    DataLength[15:0];

    Data[...];

    LLRPStatusTLV Parameter(1);



//=============================================================================
SyslogMessageStruct(SubType=8)
    Rsvd[2:0];
    Ver[2:0];
    MessageType[12:0];
    MessageLength[31:0];
    u32 MessageID[31:0];
    VendorID[31:0]
    Subtype[7:0];
    Level[7:0];
    Tag[15:0];
    AID[15:0];
    SyslogMsgLength[15:0];
    SyslogMessage[...];


//==============================================================================

CustomSetRoSpecStruct(SubType=9)
    Rsvd[2:0];
    Ver[2:0];
    MessageType[12:0];
    MessageLength[31:0];
    u32 MessageID[31:0];
    VendorID[31:0]
    Subtype[7:0];
    ROSpecID[31:0];
    CustomRoSpecPower Parameter(0_N);
    CustomRoSpecSensitivity Parameter(0_N);
    CustomRoSpecRFMode Parameter(0_N);
    CustomRoSpecMuxSequence Parameter(0_1);
    CustomRoSpecInvParamOrder Parameter(0_1);
    ROSpecStartTriggerTLV Parameter(0_1);
    CustomRoSpecSingulation Parameter(0_1);
    CustomRoSpecFixedFreq Parameter(0_1);


//==============================================================================
CustomSetRoSpecResponseStruct(SubType=10)

    Rsvd[2:0];

    Ver[2:0];

    MessageType[12:0];

    MessageLength[31:0];

    u32 MessageID[31:0];

    VendorID[31:0]

    Subtype[7:0];

    LLRPStatusTLV Parameter(1);



//==============================================================================


CustomGetRoSpecStruct(SubType=11)

    Rsvd[2:0];

    Ver[2:0];

    MessageType[12:0];

    MessageLength[31:0];

    u32 MessageID[31:0];

    VendorID[31:0]

    Subtype[7:0];

    ROSpecID[31:0];

    AntennaID[15:0];

    ProtocolID[15:0];

    RequestedData[31:0];



//==============================================================================
CustomGetRoSpecResponseStruct(SubType=12)
    Rsvd[2:0];
    Ver[2:0];
    MessageType[12:0];
    MessageLength[31:0];
    u32 MessageID[31:0];
    VendorID[31:0]
    Subtype[7:0];
    LLRPStatusTLV Parameter(1);
    CustomRoSpecPower Parameter(0_1);
    CustomRoSpecSensitivity Parameter(0_1);
    CustomRoSpecRFMode Parameter(0_1);
    CustomRoSpecFixedFreq Parameter(0_1);
    ROSpecTLV Parameter(0_N);


//=============================================================================

TracingSpecCustomParameterStruct(SubType=1)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    enable_macphy[7:0];
    enable_atc[7:0];
    enable_metal_detection[7:0];
    enable_cognitive_radio[7:0];
    enable_power_control[7:0];
    enable_agc[7:0];
    enable_inv_summary[7:0];
    enable_inv_event[7:0];
    capture_criteria[i7:0];
    capture_delay_count[7:0];
    capture_rn16[7:0];
    capture_epc[7:0];
    capture_handle[7:0];
    capture_access[7:0];

    capture_write[7:0];



//===============================================================================


MacPhyTraceCustomParameterStruct(SubType=2)

    Rsvd[5:0];

    ParameterType[12:0];

    ParameterLength[15:0];

    VendorID[31:0];

    Subtype[31:0];

    mac_empty_ratio[i15:0];

    mac_slot_number[i15:0];

    mac_increase_count[i15:0];

    mac_collision_count_since_newq[i15:0];

    mac_q[7:0];

    mac_command[7:0];

    mac_tag_volume_estimate[7:0];

    mac_current_target[7:0];

    mac_session[7:0];

    port[7:0];

    channel[7:0];

    demod_flags[7:0];

    demod_result[7:0];

    current_pilot_rssi[i31:0];

    pilot_threshold[i31:0];

    subcarrier_frequency_state[i31:0];

    symbol_period_estimate[i31:0];

    carrier_phase[i15:0];

    preamble_distance[i31:0];

    max_decode_distance[i31:0];

    total_decode_distance[i31:0];

    ave_decode_distance[i31:0];

    timestamp32[31:0];

    demod_norm[i15:0];

    demod_block_exp[7:0];
    sideband[7:0];

    rx_gain[7:0];

    rx_peak_det[i31:0];


//==============================================================================

AtcTraceCustomParameterStruct(SubType=3)

    Rsvd[5:0];

    ParameterType[12:0];

    ParameterLength[15:0];

    VendorID[31:0];

    Subtype[31:0];

    port[7:0];

    channel[7:0];

    timestamp32[31:0];

    operation[7:0];

    error[31:0];

    dac[i31:0];

    flags[7:0];

    trace_event_cnt[31:0];

    carrier_shift[i7:0];

    error_target[31:0];

    spiral_thres[31:0];


//==============================================================================

MtlDetTraceCustomParameterStruct(SubType=4)

    Rsvd[5:0];

    ParameterType[12:0];

    ParameterLength[15:0];

    VendorID[31:0];

    Subtype[31:0];

    port[7:0];

    freq[7:0];

    state[7:0];
    carrier_estimate[31:0];
    filtered_i[31:0];
    filtered_q[31:0];
    sample_count[7:0];
    time32[31:0];
    energy[31:0];
    phase[31:0];
    delta_phase[31:0];
    delta_energy[31:0];
    flags[31:0];


//==============================================================================

CognitiveRadioTraceCustomParameterStruct(SubType=5)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    port[7:0];
    channel[7:0];
    timestamp32[31:0];
    upper_sideband1_filtered[i31:0];
    lower_sideband1_filtered[i31:0];
    upper_1st_adj_filtered[i31:0];
    lower_1st_adj_filtered[i31:0];
    upper_sideband2_filtered[i31:0];
    lower_sideband2_filtered[i31:0];
    upper_2nd_adj_filtered[i31:0];
    lower_2nd_adj_filtered[i31:0];
    upper_3rd_adj_filtered[i31:0];
    lower_3rd_adj_filtered[i31:0];


//==============================================================================
TagParamReportCustomParameterStruct(SubType=6)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    enable_phase[7:0];
    enable_rssi[7:0];
    enable_speed[7:0];
    enable_range[7:0];
    enable_position[7:0];
    enable_tx_power[7:0];


//=============================================================================

PowerControlTraceCustomParameterStruct(SubType=11)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    tx_power[i15:0];
    set_point[i15:0];
    detector[i31:0];
    integrator[i31:0];
    antenna[i31:0];
    frequency[31:0];
    time[63:0];


//=============================================================================
InventoryTraceCustomParameterStruct(SubType=12)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    ParameterLength[15:0];
    pParameter[...];


//============================================================================

ROSyncCustomParameterStruct(SubType=13)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    reader_id[7:0];
    slot_no[15:0];
    total_slot[7:0];
    duration[15:0];


//============================================================================

WritebackPS111CustomParameterStruct(SubType=14)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    OpSpecID[15:0];
    Enable[7:0];
    Action[7:0];
    Offset[15:0];
    MaskByteCount[15:0];
    Mask[...];

    DataByteCount[15:0];

    Data[...];

    ValueByteCount[15:0];

    Value[...];



//==============================================================================


InstallTypeCustomParameterStruct(SubType=15)

    Rsvd[5:0];

    ParameterType[12:0];

    ParameterLength[15:0];

    VendorID[31:0];

    Subtype[31:0];

    ID[31:0];



//==============================================================================


FileTransferCustomParameterStruct(SubType=16)

    Rsvd[5:0];

    ParameterType[12:0];

    ParameterLength[15:0];

    VendorID[31:0];

    Subtype[31:0];

    ID[7:0];

    Address[31:0];

    DataLength[15:0];



//==============================================================================
FileDataCustomParameterStruct(SubType=17)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    DataLength[15:0];
    pData[...];


//==============================================================================

ReaderInformationParameterStruct(SubType=18)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    SerialNumberLength[15:0];
    pSerialNumber[...];
    DSPSerialNumberLength[15:0];
    pDSPSerialNumber[...];
    DSPChipIDLength[15:0];
    pDSPChipID[...];
    ReaderModelLength[15:0];
    pReaderModel[...];
    HardwareVersionLength[15:0];
    pHardwareVersion[...];
    SoftwareVersionLength[15:0];
    pSoftwareVersion[...];
    BootloaderVersionLength[15:0];
    pBootloaderVersion[...];
    SecurityChipSerialLength[15:0];
    pSecurityChipSerial[...];


//==============================================================================
TxPowerParameterStruct(SubType=19)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    TxPower[15:0];


//==============================================================================

TagTransitParameterStruct(SubType=20)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    TagState[15:0];
    RepeatCount[15:0];
    AuditRecord[31:0];
    firstMicroseconds[63:0];
    lastMicroseconds[63:0];


//==============================================================================

InventoryEventParameterStruct(SubType=21)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    ROSpecID[7:0];
    Port[7:0];
    Protocol[7:0];
    RFMode[7:0];
    State[7:0];
    time[63:0];


//============================================================================

WritebackISOCParameterStruct(SubType=22)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    OpSpecID[15:0];
    Enable[7:0];
    BlockWrite[7:0];
    Action[7:0];
    Offset[15:0];
    MaskWordCount[15:0];
    Mask[...];
    DataWordCount[15:0];
    Data[...];
    ValueWordCount[15:0];
    Value[...];


//============================================================================

RegulatoryRegionsSupportedParameterStruct(SubType=23)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    CommunicationStandardsCount[7:0];
    CommunicationStandards[...];


//============================================================================
RegulatoryCommunicationStandardParameterStruct(SubType=24)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    CountryCode[15:0];
    CommunicationStandard[15:0];


//==============================================================================

WritebackISOCOpSpecResultParameterStruct(SubType=25)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    Result[7:0];
    OpSpecID[15:0];
    WordCount[15:0];
    Data[...];


//==============================================================================

DepartTimeParameterStruct(SubType=26)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    Value[31:0];


//==============================================================================
AdditionalRFTransmitterParameter(SubType=27)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    HOPTableID[15:0];
    ChannelIndex[15:0];
    TransmitPower[15:0];


//=============================================================================

ReaderSerialNumberParameterStruct(SubType=28)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    SerialNumberLength[15:0];
    pSerialNumber[...];


//=============================================================================

AgcTraceCustomParameterStruct(SubType=29)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    port[7:0];
    channel[7:0];
    timestamp32[31:0];
    bb_gain[7:0];
    rf_gain[7:0];
    flags[7:0];
    rf_peak[i31:0];


//==============================================================================

C1G2InventoryControlParameterStruct(SubType=30)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    InventoryBothTargets[7:0];
    QuerySelPredetermined[7:0];
    MaxIncSlotsQ[7:0];
    SelectCmdPeriod[15:0];


//==============================================================================

TagReadCustomParameterStruct(SubType=31)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    MB[7:0];
    DataBitCount[15:0];
    Data[...];


//==============================================================================
C1G2TAM1TLVParameterStruct(SubType=32)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    OpSpecID[15:0];
    AuthMethod[7:0];
    CustomData[7:0];
    RFU[7:0];
    KeyId[7:0];
    IChallengeLength[7:0];
    IChallenge[...];


//==============================================================================

C1G2TAM2TLVParameterStruct(SubType=33)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    OpSpecID[15:0];
    AuthMethod[7:0];
    CustomData[7:0];
    RFU[7:0];
    KeyId[7:0];
    Profile[7:0];
    Offset[15:0];
    BlockCount[7:0];
    ProtMode[7:0];
    IChallengeLength[7:0];
    IChallenge[...];


//==============================================================================
C1G2UntraceableTLVParameterStruct(SubType=34)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    OpSpecID[15:0];
    RFU[7:0];
    U[7:0];
    EPC[7:0];
    TID[7:0];
    User[7:0];
    Range[7:0];
    AccessPassword[31:0];


//============================================================================

C1G2TAM1OpSpecResultTLVParameterStruct(SubType=35)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    Result[7:0];
    OpSpecID[15:0];
    WordCount[15:0];
    ReadData[...];
    IChallengeLength[7:0];
    IChallenge[...];


//============================================================================
C1G2TAM2OpSpecResultTLVParameterStruct(SubType=36)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    Result[7:0];
    OpSpecID[15:0];
    WordCount[15:0];
    ReadData[...];
    IChallengeLength[7:0];
    IChallenge[...];


//============================================================================

C1G2UntraceableOpSpecResultTLVParameterStruct(SubType=37)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    Result[7:0];
    OpSpecID[15:0];


//============================================================================

AccessSpecPS111TLVCustomParameterStruct(SubType=38)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    AccessSpecID[31:0];
    AntennaID[15:0];
    ProtocolID[7:0];
    ROSpecID[31:0];
    WritebackPS111Custom Parameter(1_N);


//==============================================================================

CustomRoSpecPowerParameterStruct(SubType=39)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    AntennaID[15:0];
    ProtocolID[7:0];
    PowerIndex[15:0];
    PowerValue[i31:0];
    AdvancedAntenna Parameter(0_1);


//==============================================================================

CustomRoSpecSensitivityParameterStruct(SubType=40)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    AntennaID[15:0];
    ProtocolID[7:0];
    SensitivityIndex[15:0];
    SensitivityValue[i31:0];


//==============================================================================
CustomRoSpecRFModeParameterStruct(SubType=41)

    Rsvd[5:0];

    ParameterType[12:0];

    ParameterLength[15:0];

    VendorID[31:0];

    Subtype[31:0];

    AntennaID[15:0];

    ProtocolID[7:0];

    ModeID[31:0];



//============================================================================


CustomRoSpecMuxSequenceParameterStruct(SubType=42)

    Rsvd[5:0];

    ParameterType[12:0];

    ParameterLength[15:0];

    VendorID[31:0];

    Subtype[31:0];

    AntennaCount[15:0];

    pAntennaID[...];



//============================================================================


CustomRoSpecInvParamOrderParameterStruct(SubType=43)

    Rsvd[5:0];

    ParameterType[12:0];

    ParameterLength[15:0];

    VendorID[31:0];

    Subtype[31:0];

    ProtocolCount[7:0];

    pProtocolID[...];



//============================================================================
CustomRoSpecSingulationParameterStruct(SubType=44)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    session_id[7:0];
    auto_mac[7:0];
    inventory_both_targets[7:0];
    max_incr_slots_q[7:0];
    number_slots_q[7:0];
    query_sel[7:0];
    query_session[7:0];
    query_target[7:0];
    select_cmd_period[15:0];


//============================================================================

CustomRoSpecFixedFreqParameterStruct(SubType=45)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    AntennaID[15:0];
    ProtocolID[7:0];
    count[7:0];
    frequencies[...];


//============================================================================
ISOBFilterTLVParameterStruct(SubType=46)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    Command[7:0];
    Mask[7:0];
    Address[7:0];
    Data[...];


//=============================================================================

TagSecurityV1TLVParameterStruct(SubType=47)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    OpSpecID[15:0];
    PWAuthEnable[7:0];


//=============================================================================

TagSecurityV1OpSpecResultTLVParameterStruct(SubType=48)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    Result[7:0];
    OpSpecID[15:0];
    OpState[7:0];
    SecurityState[7:0];
    TagTypeIndex[7:0];
```

```
//============================================================================

PrecisionPhaseParameterStruct(SubType=49)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    PrecisionPhase[15:0];


//============================================================================

PrecisionRSSIParameterStruct(SubType=50)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    PrecisionRSSI[i15:0];


//============================================================================

AdvancedAntennaParameterStruct(SubType=51)
    Rsvd[5:0];
    ParameterType[12:0];
    ParameterLength[15:0];
    VendorID[31:0];
    Subtype[31:0];
    GainUnits[7:0];
    Gain[i15:0];
    CableLoss[i15:0];
    Attenuation[i15:0];
    ConductedPowerValue[15:0];
    ComputedConductedPowerValue[15:0];

     //============================================================================
```

Warranty

WARRANTY

STAR Systems warrants all Hardware Products against defects in materials and workmanship under normal use and service for one (1) year from the original purchase date (the "Warranty"). Batteries included in a reader or package are warranted for six (6) months. Any Extended Warranties must be noted on the original invoice. If covered by the Warranty, STAR Systems will repair or replace the product, at its sole discretion, and return it to Customer.

EXCLUSIONS

The Warranty shall not apply in cases of damage caused by abuse, mishandling, acts of God, vandalism, over-voltage, accidents, electrostatic discharge or electrical issues, improper installation or use, unsuitable environments, using the product for unintended purposes, unauthorized modification of the product, modification of the printed circuit board by parties other than STAR Systems, damage that is caused during shipping for warranty service, or to product returned with a broken warranty security seal. In these cases, STAR Systems will provide a repair or replacement estimate before performing any work.

RMA PROCEDURE

Prior to returning a product for RMA, Customer must contact STAR Systems Customer Care at cust.care@star-int.net for a case number and return instructions. Failure to obtain a case number or to adhere to the provided return instructions may lead to delays or refusal of the returned Product, with return shipping at the Customer's expense. The Product should be properly packaged and protected, preferably in the original carton and protective packaging materials. For products under Warranty, Customer is responsible for the shipping cost to STAR Systems' designated repair location, while STAR Systems covers the return shipping using its standard shipment method. Customers seeking expedited shipping options will be responsible for any additional associated costs. For out-of-warranty products, Customer is responsible for the shipping costs to and from STAR Systems' designated repair location.

DISCLAIMER OF WARRANTIES

Other than set forth above, STAR systems hereby disclaims all warranties, expressed or implied, including without limitation, the warranties of merchantability, fitness for a particular purpose and noninfringement.

LIMITATION OF LIABILITY

In no event will STAR Systems be liable for any consequential, indirect, exemplary, special, or punitive damages, whether arising out of contract, tort, negligence, strict liability or otherwise. In no event will star systems international total cumulative, aggregate liability, whether arising out of contract, tort, negligence, strict liability, or otherwise, exceed the price actually paid by customer for the product from which the claim arises.

This warranty gives Customer specific legal rights, and Customer may also have other rights that may vary from local jurisdiction. If Customer has questions concerning the product or warranty, contact the dealer from which it was purchased. Customer may also contact STAR Systems at the following address and ask for warranty assistance.

About Us

Founded in 2013, STAR Systems International is a global leader in Automatic Vehicle Identification (AVI) Technologies. Our primary focus is to deliver best-in-class transponders, readers, and professional consulting services for Smart City Initiatives. Our expertise lies in a wide range of applications, including Electronic Tolling (ETC), Congestion/Road Use Charging, Electronic Vehicle Registration (EVR), Express/HOT Lane, Fleet Management, Parking, and Secure Access Control.

At STAR Systems, we are driven by a customer-centric approach. Our goal is to ensure the success of our customers by leveraging our technical expertise and implementation experience. Guided by three core principles - Outstanding People, Innovative Products, and Service Excellence - we strive to advance Smart City Technologies and contribute to the long-term growth of the AVI industry

STAR Systems International Companies

STAR Systems International Pte. Ltd. (Singapore)
STAR Systems International Ltd. (Hong Kong)
STAR Systems America, LLC (U.S.A.)
STAR RFID & Systems India Pvt Ltd. (India)
STAR RFID Co., Ltd. (Thailand)

Technical Support
https://star-int.net/support/
support@star-int.net

Other Inquiries
salesinquiry@star-int.net
+852 3691-9925 (HK)
+1 888-457-7755 (US Toll Free)

Tarvos Pro User Guide - Version 2.0