



Captis Shell Commands

20/12/2022



Contents

1	Scope.....	2
2	Overview.....	2
3	Shell Commands.....	2
3.1	Functionality	3
3.2	Available Commands.....	4
3.2.1	Device Interrogation	4
3.2.2	Sensor Interrogation.....	5
3.2.3	Live Mode.....	6
3.2.4	Bootstrap	6
3.2.5	Sleep.....	8
3.2.6	Reset Items in Flash.....	8
3.2.7	GPS	8
3.2.8	Sensor Calibration.....	9
3.2.9	Server Connections	9
3.2.10	Sensor Power.....	9
3.2.11	Digital Output.....	9
4	Conclusion.....	9



1 Scope

The purpose of this document is to detail the functionality of shell commands in Captis Firmware. It allows a customer to have full control over the suite of commands and their Captis device fleet. The shell commands available are limited to the commands contained within this document at the time of publishing.

2 Overview

Shell commands are a native feature in Cumulocity and support for them was introduced in Captis firmware 1.19.00. Shell commands allow you to instruct the device to perform an action (such as bootstrap or sleep) or to interrogate the device or sensors.

Shell commands are queued from the “>_ Shell” tab of the device menu in the Captis Management App. That tab only appears when the device tells Cumulocity (on setup) that it supports shell commands.

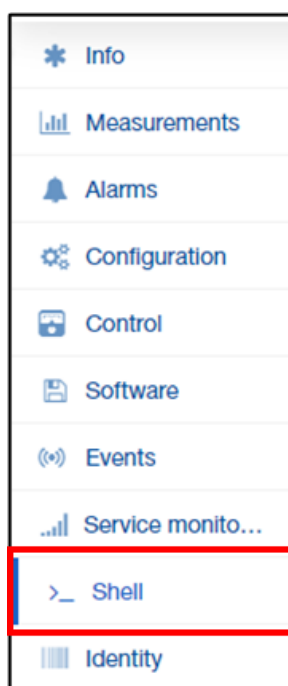


Figure 1

3 Shell Commands

The following section will detail both the shell command functionality and a comprehensive list of available shell commands.




3.1 Functionality


The command structure is comma separated and always starts with the device serial number, then command, then arguments (if applicable). For example:

`<DEVICE SERIAL NUMBER>,<SHELL COMMAND>,<ARGUMENT>`

`89610180002841893752,get_pulse_total,1`

For commands that support 'bulk apply' you can replace the serial number with a wildcard (Asterix). For example, both of the following commands will do the same thing:

 `89610180002841893752,rf`

 `*,rf`

When applying a shell command as a bulk operation you must use a wildcard otherwise the command will be rejected by all but one device.

All commands will return a success or fail status when processed. Some commands will also provide a text response. This response can be viewed by expanding the command item in the 'Shell' tab and viewing the 'Response' field or on the 'Control' tab under the 'Result' field as seen in the figures below:



Figure 2

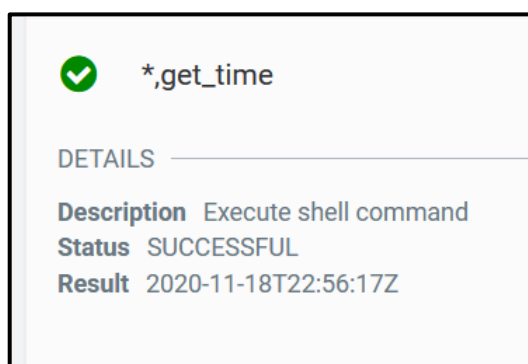


Figure 3



3.2 Available Commands

The following sub-sections will describe each of the command topics below in further detail:

- 📶 Device Interrogation
- 📶 Sensor Interrogation
- 📶 Device Actions
 - Live mode
 - Bootstrap
 - Sleep
 - Reset Items in Flash
 - GPS
 - Sensor Calibration
 - Server Connections
 - Sensor Power
 - Digital Output

3.2.1 Device Interrogation

Command	Description	Arguments	Response	Supports bulk?
rf	Fetch signal strength values	None	"RSSI: -58 dB, RSRP: -91 dB, RSRQ: -16.0 dB, SINR: 6.6 dB"	Yes
tamper	Fetch tamper switch state	None	"SET" or "CLEAR"	Yes
battery	Fetch battery values	None	"3.47 V, 102.000 mA, 4361.383 mAhr, 25.2 degC"	Yes
log_storage	Fetch log storage info	None	"2113 stored, 2112 sent, 0.0 % storage used"	Yes
get_time	Gets the current time in the device	None	"2020-11-18T22:56:16Z"	Yes



3.2.2 Sensor Interrogation

Command	Description	Arguments	Response	Supports bulk?
get_pulse_total	Fetch pulse total volume	DI channel. If not specified then DI1 returned	"P2: 0 L"	Yes
get_switch	Fetch switch state	DI channel. If not specified then DI1 returned	"SW1: OPEN"	Yes
get_analog	Get analog voltage and scaled value (from scaling in config)	None	e.g. "AI1: 1.050 V, 4.199 m"	Yes
get_power_out	Get the external 5V output state	None	ON or OFF	Yes
get_digital_out	Get the digital output state	None	OPEN, CLOSED, or ERROR	Yes
setup_serial	Setup the serial modem for the duration of this operations connection	<protocol>,<baud>,<parity > Protocol: RS232 or RS485 Baud: 9600, 19200, 38400, 57600, 115200	None	Yes
raw_serial_cmd	Set a serial command and return the response	<type>,<message> type is 'HEX' or 'ASCII'	'RX: No response', or 'RX: <serial response>'	Yes
modbus_packet	Send a modbus packet and have the CRC calculated for you and appended and return the response	<type>,<message> type is 'RTU' only	'RX: No response', or 'RX: <serial response>'	Yes
modbus_register	Send a modbus request and return the value	<slave addr>,<register number>,<register type>,<data type>,[<byte order>] register type is "HOLDING_REG", "INPUT_REG" data type is "UINT16", "FLOAT", "UINT32", "SINT16", or "SINT32" byte order is optional and is "ABCD", "DCBA", "CDAB", or "BADC". Will default to ABCD	"Response: <value>"	Yes
ow_search	Search both one wire channels and return the list of device addresses	None	Format is "Channel: address list". For no devices it will show "1: None 2: None". e.g. with devices: "1: 28274E4B07000068 2: 28B21B1B03000033"	Yes
ow_read	Read the values from a single one wire probe	Probe address	"<probe address> - <part number>: <values>". e.g. "28274E4B07000068 - DS18B20: 23.56 degC"	Yes



3.2.3 Live Mode

Live mode is a mode where the Captis device will wait, listen for operations, and action any that are received. This is useful for scenarios such as site installation or connecting a new sensor, where you may wish to try things like antenna position, sensor scaling factors, different modbus registers etc. and do not want to go through the unnecessary delay of back-to-back connections.

Live mode will last for 2 minutes from the time the command is taken. This can be extended at any time while in live mode by sending the command again. The 'stop_live' command will stop waiting and proceed with the rest of the connection.

Command	Description	Arguments	Response	Supports bulk?
live_mode	Listen for operations for a longer time (2 min)	None	None	Yes
stop_live	Stop listening for operations	None	None	Yes




3.2.4 Bootstrap

Bootstrap requires a bootstrap server to be defined in the configuration first and then this command will execute the bootstrap process. The bootstrap process will be attempted upon every connection until it concludes or is cleared with the clear bootstrap command. Note that bootstrap will now be attempted in addition to the main MQTT connection, unlike previous firmware versions where it was either bootstrap or the main MQTT connection, but not both.

If you are migrating from one server to another then it is best to delete the existing server from config and let bootstrap be automatically triggered (bootstrap is attempted when there is no server with a config role defined). If you use the shell command then it will result in 2 MQTT servers in the configuration.

Command	Description	Arguments	Response	Supports bulk?
bootstrap	Tell device to attempt bootstrap	<p><server>,<mode></p> <p>If server is not present, '0' or '1' then the first bootstrap server in config is used.</p> <p>If server is '2' the second defined bootstrap server is used and so on.</p> <p>If server is 's3' then the server with name 's3' will be used.</p> <p>If the server referenced is not defined or does not have a bootstrap role then it will fail the operation.</p> <p>Both server and mode are optional but you will need to specify server if you want to use mode. The default for server is still the first defined bootstrap server in config. The default for mode is 'regular'.</p>	None	Yes



Command	Description	Arguments	Response	Supports bulk?
Bootstrap (Continued)	(Continued)	<p>Mode Descriptions:</p> <p>'regular' - if the server already exists then no roles will be changed. If it is a new server then the previous logic is applied where roles are assigned in the following order (stopping when the selection won't cause a config validation error:</p> <ul style="list-style-type: none">  "MEAS", "CONFIG", "EVENTS", "INFO", "FOTA_OPS", "SHELL"  "MEAS", "EVENTS", "INFO"  "INFO" <p>'greedy' - the new server will take the roles "MEAS", "CONFIG", "EVENTS", "INFO", "FOTA_OPS", "SHELL" with "MEAS" being removed from this list if there is a data server already and "CONFIG" being removed if there is a config server already. No roles from other servers will be modified</p> <p>'data' - the server will have the roles "MEAS", "EVENTS", "INFO" added to its current roles and if "MEAS" exists on another server then it will be removed from that server</p> <p>'config' - the server will have the roles "INFO", "CONFIG", "FOTA_OPS", "SHELL" added to its current roles and if "CONFIG" exists on another server then it will be removed from that server</p> <p>'full' - the server will have the roles "MEAS", "CONFIG", "EVENTS", "INFO", "FOTA_OPS", "SHELL" added to its current roles and if "CONFIG" or "MEAS" exists on another server/s then they will be removed from their respective server/s</p>	(Continued)	(Continued)
clear_bootstrap	Stop bootstrap attempts	None	None	Yes



3.2.5 Sleep

Sleep will place the device into a sleep mode where it will not make connections and will not measure or record sensor data. The only way to exit this mode is to physically wake the device with a magnet. It is highly recommended that this is not used for devices in the field. This is intended for devices in storage, or otherwise unused, in order to save battery and data usage.

Command	Description	Arguments	Response	Supports bulk?
<code>sleep</code>	Puts device in permasleep	None	None	No

3.2.6 Reset Items in Flash

These commands will wipe their respective sections of NVM flash memory. These actions are **not reversible**. For reset settings, the default config will be saved after the previous config was wiped.

Command	Description	Arguments	Response	Supports bulk?
<code>reset_settings</code>	Reset settings to default	None	None	No
<code>reset_logs</code>	Wipe measurement storage	None	None	No
<code>reset_events</code>	Wipe Event Storage	None	None	No

3.2.7 GPS

The 'get_GPS_fix' command instructs the device to get a GPS fix on the next connection. This will only succeed if the device has been enabled for GPS, with an external SMA connection available and an appropriate GPS antenna fitted. The device will continue trying to get a fix every connection until it succeeds. To stop these attempts, send the 'clear_GPS_fix' command.

Command	Description	Arguments	Response	Supports bulk?
<code>get_gps_fix</code>	Tell device to get GPS fix on next connection	None	None	Yes
<code>clear_gps_fix</code>	Tell device to stop attempting GPS fix	None	None	Yes



3.2.8 Sensor Calibration

Command	Description	Arguments	Response	Supports bulk?
<code>pulse_cal</code>	Generate a pulse offset based on meter reading and store in config	Format is <channel>,<volume_total>,<units> e.g. for DI1: pulse_cal,1,12345,L Units must be present and match config	None	No

3.2.9 Server Connections

Command	Description	Arguments	Response	Supports bulk?
<code>nth_conn_override</code>	Override the nth connection logic for this connection and the one after. i.e connect to all servers	None	None	Yes
<code>refresh_info</code>	Tells device to do server setup again	'0' = all servers, '1' to '9' = s1 to s9, 's1' to 's9' = s1 to s9	None	Yes

3.2.10 Sensor Power

This command will turn on the sensor power output until the device finishes with shell commands and configuration processing. This time will be affected by live mode.

Command	Description	Arguments	Response	Supports bulk?
<code>set_power_out</code>	Set the external 5V output state	ON or OFF	None	Yes

3.2.11 Digital Output

Setting the digital output from a shell command is not dependant on setting the digital output to any mode in configuration. It may conflict with other digital output logic (such as process alarms) if configured, therefore use with caution. Command will fail if there is no digital output on the device.

Command	Description	Arguments	Response	Supports bulk?
<code>set_digital_out</code>	Set the digital output state	OPEN or CLOSED	None	Yes

4 Conclusion

Shell commands have been introduced in FW 1.19.00 to provide additional functionality for customers. For any additional information, please refer to our website or contact our support team.