

BB4-8422 SNMP Trap Receiver

SNMP Trap Receiver Rule Status

Dashboard / Protocol Status / SNMP Trap Receiver

Rules

Show 10 entries

Search:

Rule ↑↓	Device ↑↓	Trap OID ↑↓	Var OID ↑↓	Seconds since receive ↑↓	Trap receive count ↑↓
1	192.168.1.20	1.3.6.1.4.1.318.0.5	1.3.6.1.4.1.318.2.3.3.0	212906.19	0
2	192.168.1.20	1.3.6.1.4.1.318.0.9	1.3.6.1.4.1.318.2.3.3.0	212906.23	0
Rule	Device	Trap OID	Var OID	Seconds since receive	Trap receive count

Showing 1 to 2 of 2 entries

Previous 1 Next

[✕ Clear Counts](#)





The trap receiver status page simply provides a list of all currently configured rules and some diagnostic information about them. Trap receive count is the number of times this trap has been received since restart.

Configuration

SNMP Trap Receiver Rules

Rules **Config File** Snmptrapd.conf

Show entries Search:

Rule Number ↑↓	Peer Name ↑↓	Trap OID ↑↓	Var OID ↑↓	Local Object ↑↓
1  	192.168.1.20	1.3.6.1.4.1.318.0.5	1.3.6.1.4.1.318.2.3.3.0	23 : Object 23
2  	192.168.1.20	1.3.6.1.4.1.318.0.9	1.3.6.1.4.1.318.2.3.3.0	23 : Object 23
Rule Number	Peer Name	Trap OID	Var OID	Local Object

Showing 1 to 2 of 2 entries

Previous **1** Next

Add new trap rule(s) before rule number: **Insert rule(s)**

Set Source Info

The Trap Receiver Rules are where you configure this IoTServer to receive SNMP traps sent by other agents. The data received from or derived from those traps is stored in local data objects.

Rule Number ↑↓	Peer Name ↑↓	Trap OID ↑↓	Var OID ↑↓	Local Object ↑↓
1  	192.168.1.20	1.3.6.1.4.1.318.0.5	1.3.6.1.4.1.318.2.3.3.0	23 : Object 23

- Click on the rule number or pencil icon to jump to the rule editing page for this rule.
- Click on the trash can icon to delete this rule.

Add new trap rule(s) before rule number: **Insert rule(s)**

To add or insert new rules, enter a number of rules to add, and select a starting point. Then click the Insert button.

Set Source Info

The Set Source Info button is a diagnostic tool. You do not want multiple client functions (from all protocols combined) trying to store data into the same local object. Doing so would result in erroneous data (data would jump back and forth showing the most recently received data from any source). If you only have one source of data, then this isn't an issue. But if there will be multiple sources of data, then use the Set Source Info button to "claim" objects for this protocol function (e.g. SNMP Trap Receive).

If objects have already been claimed in another protocol and you try to claim the same objects, you will see one or more error messages displayed at the top of the screen. Go to the Data Objects -> Object Status page to see where objects have been claimed. The Clear Source Info button is also found on the Data Objects -> Object Status page. To clear the claims and try again, use that button. Clicking the Set Source Info button more than once will also result in error messages since objects have already been claimed once.

SNMP Trap Receiver Rule Edit

Dashboard / Protocol Configuration / SNMP Trap Receiver / Receive Rule #1

SNMP Trap Receive Rule #1

Rule Number – Used as a reference in the rule list for ordering the rules.

When a Trap is received from this peer:

Peername – Specifies the peername that the trap is expected to be received from. If an IP address is given, then the IP address of the incoming trap message is checked. If a host name is given and the incoming trap message contains a host that could be looked up (via DNS lookup), then the host name field is checked instead of IP address.

IMPORTANT: The peer name specified here is for filtering purposes in processing of the received trap. Actually receiving the trap requires that the sender be recognized as permitted in settings in the snmptrapd.conf file. Permission to receive the trap is established in snmptrapd.conf. If received, then the peer name provided here is used to filter the results for purposes of determining if this rule applies.

Matching these OID values:

Trap OID: 1.3.6.1.4.1.318.0.5 Var OID: 1.3.6.1.4.1.318.2.3.3.0

TrapOID – Specifies the Trap OID that is expected. Traps and Informs always contain a Trap OID identifying the trap or inform, along with additional varbinds with a data payload. The Trap OID must match the OID given here before action will be taken.

VarOID – Specifies the varbind OID to look for in the message if the Peername and Trap OID match.

Applying this hint to octet strings:

Hint: 0 - Character

Hint – Provides direction on how to interpret data in the event the value type is Octet String. All ASN data types are interpreted according to their type; however, Octet String will default to being interpreted as a character string unless this hint says otherwise, and the length is exactly 4 or 8 bytes (octets).

Hint Label	Description
None	No special interpretation
Float	Treat 4-byte Octet String as RFC 6340 IEEE 754 32-bit floating point
Double	Treat 8-byte Octet String as RFC 6340 IEEE 754 64-bit floating point

Save result in Local object: 23 named Object 23

ResultObj – Specifies the local object where data retrieved from the varbind called out by VarOID will be placed (or fixed value will be placed) assuming the VarOID is found, and Peername and Trap OID match.

Set fixed value upon trap received

Set fixed value upon trap received Fixed Value: 1

Fixed – Check to specify that a fixed value should be placed in the local object when this trap is received, rather than try to derive a data value from the trap message. Uncheck to parse data

from the trap varbind to be placed in the local object. (Use "Y" to check and "N" to uncheck in CSV or XML files.)

FixedVal – The fixed value to be placed in the local object any time this trap is received, assuming the fixed option has been enabled. This feature is most often used in conjunction with the Timeout feature.

Reset local object to Value: 0 after 0 Seconds

Timeout – Integer number of seconds for trap receive timeout. Note that this is not a session timeout, it provides for an automatic reset of the local object value. If zero, then the timeout feature is disabled.

TimeoutVal – Specifies the fixed value that should be placed in the local object after the timeout period has expired. The typical scenario is that either the value gleaned from the trap message itself, or the fixed value, would be placed in the local object upon receiving the trap. Then ‘timeout’ period later, the timeout value is placed in the local object, replacing the trap value.

The combination of fixed value and timeout can be useful in causing a local object to reflect the state of an alarm in an RFC 1628 UPS system. Alarm traps in the UPS will be sent periodically when the alarm condition exists, and simply stop being sent when the alarm clears. If the trap sets a fixed value, and if the timeout is set to a period longer than the alarm traps repeat, then this creates a means of having a static indication of alarm presence.

SNMP Trap Receiver Config File


Dashboard / Protocol Configuration / SNMP Trap Receiver



Rules Config File Snmpttrapd.conf

All of your configuration information is stored in an internal database each time you click the Save button on any page where configuration entries may be made. To make configuration portable from one device to another, and for purposes of retaining a backup copy, the configuration information may be exported and imported as XML or CSV files. This page is where your configuration file management takes place.

It is important to note that the XML file saved within any one client/server function will contain the configuration information for only that function. Depending on overall system configuration, a complete backup may involve more than one XML or CSV file.

Configuration File Load/Save

Configuration file: 


 



XML Files: When an XML file has been selected, click the Load button to clear the configuration database and reload configuration from the given XML file.

Select an existing name to overwrite or enter a new file name, and then click Save to write the current configuration to the file in XML format.

You may type in a new name in the file name window for purposes of saving a new file. If you click the Refresh button, the file name will be restored to the name currently loaded into the client. The name could have been changed by selecting a file from the list below, or by typing in a new name. If the displayed name has not yet been used, then Refresh will restore the file name to what was most recently loaded.

Configuration File Load/Save

Configuration file: 

CSV Files: If a CSV file is selected, the Load and Save buttons will change into load/save CSV buttons. When a CSV file has been selected, click the Load button to clear the configuration database and reload configuration from the given CSV file.

Select an existing name to overwrite or enter a new file name, and then click Save to write the current configuration to the file in CSV format.

Configuration File Load/Save

Configuration file: [Refresh](#)

[Load Conf file into Engine](#) [Save Current Config into .conf file](#)

Snmpttrapd.conf Files: Any time the snmptrapd.conf file is regenerated on the Snmptrapd.conf page, you will need to come here to transfer that generated file into the SNMP engine. Select the generated .conf file from the drop-down list below, and then click the Load button. You can also retrieve a copy of the snmptrapd.conf file actually in use by clicking the Save button. The content of the currently in-use snmptrapd.conf file will be transferred to the file name you have entered.

NOTE: Any time you reload the snmptrapd.conf file here, you also need to click the Restart SNMP button at the bottom of this page.

Configuration File Management

Available Configuration Files: [View](#) [Delete File](#)

[Add files...](#) [Start upload](#) [Download](#)


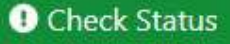
The drop-down list will show a list of all configuration files currently found in the device's configuration folder. When you select an XML or CSV file from this list, the name will be copied to the Load/Save section of this page for pending load or save.

You may view the selected file by simply clicking View. You can delete the file by clicking Delete.

You may upload files to the IoTServer from your PC. Start by clicking Browse, and then use the browser's file dialog to locate the file on your PC. Once a file is selected on your PC, click the "Start upload" button to initiate the transfer.

You may also download files from the IoTServer to your PC. Click the Download button to transfer the selected file to your PC.

Configuration Error Logs

Configuration Error Logs:  View  Check Status

Any time an XML or CSV file is loaded, an error log file is generated. The error log file will be given the same name as the loaded file, but with ".err" as the suffix instead of ".xml" or ".csv". You may view the error log by selecting it from the list and clicking View.

Status is normally displayed in a message box at the top of the screen when the load or save operation is complete. But if you want to double check the status of the previous file operation, click Check Status.

SNMP Logging File

Logging enabled:  Set logging options

Select Yes to enable logging, or No to disable. When selecting Yes, provide a log name in the /home/customer/logs/ directory. All accesses to the MIB by external managers are logged here. It is recommended that you enable logging only temporarily for diagnostic purposes to avoid eventually running out of file space. NOTE: The log file enabled here is not the same log file as enabled for the SNMP Agent.

Because the snmptrapd.service does not log SNMPv3 traps to the log file, additional logging has been added to the csiTrapHandler application. If the file csiTrapReceiver_info_log is found in the /home/customer/logs/ directory, the SNMPv3 traps will be logged there instead of snmptrapd.log.

The log files can be viewed on the System -> Logs page.

 Restart SNMP  Suspend **Current Status: Running** 

The SNMP Trap Receiver task needs to be suspended while a file load operation is in progress to prevent acting on any partial configurations. This suspend/resume operation will normally happen automatically as part of the sequence invoked by the Load button when loading an XML file. The task must be explicitly suspended here for importing a CSV file. The Suspend button will become a Resume button when the task is suspended. Click Resume to continue operation. The current status is always displayed here.

The SNMP Trap Receive task suspended via the Suspend button is the API task that provides the interface between the web UI and the internal task management. The SNMP Trap Engine itself is another process. Any time the trap receive rules are altered, it is necessary to restart the SNMP trap receiver (snmptrapd service). Click Restart SNMP to reload Snmptrapd with the new definition of receive rules.

Additional SNMP Trap Receiver Tips and Techniques

Trap Filters

The minimum snmptrapd.conf file for receiving SNMPv1 or SNMPv2 traps is as follows:

Preview of generated SNMP Trap Receiver configuration

```
snmpTrapdAddr 162

#Users for agent defined here

#Trap handler - Do not change this next line
traphandle default /usr/bin/csiTrapHandler

#SNMP V1/V2 Specific settings
disableAuthorization yes
authCommunity execute oakpark
```

Save config to file: snmptrapdGenerated.conf

✓ Save to file

This example assumes an SNMP community of “oakpark”. Yours will be different and set to what you have chosen. The “traphandle” tells the snmptrapd service what to do with the trap. In this case, it will be passed to the csiTrapHandler for further processing against the trap receive rules.

The traphandle includes the trap filter "default". This is equivalent to **.1.3.6.1.4.1*** as defined in the snmptrapd.conf manual available at net-snmp.org. An example that requires additional filters is the UPS MIB defined by RFC 1628. Its traps have a trap OID such as 1.3.6.1.2.1.33.2.0.1. To include these in our filter, it is necessary to manually edit the snmptrapd.conf file and add one more filter line as illustrated below. After clicking the Generate Config button on the Snmptrapd.conf page, you can freely edit the content of the window where its generated content is displayed. When you click the Save button at the bottom of that window, all content including your edited content is saved. This can then be loaded into the trap receiver.

Preview of generated SNMP Trap Receiver configuration

```
snmpTrapdAddr 162

#Users for agent defined here

#Trap handler - Do not change this next line
traphandle .1.3.6.1.2.1* /usr/bin/csiTrapHandler
traphandle default /usr/bin/csiTrapHandler

#SNMP V1/V2 Specific settings
disableAuthorization yes
authCommunity execute oakpark
```

Save config to file:

The comment says "do not change this next line". We didn't change it. We inserted a line before it. You don't want to omit the default line. But you can add additional lines.

We should also note that SNMP v1/v2 was illustrated here for simplicity. The same approach would be used for SNMP v3 with the difference being the authentication and privacy parameters listed per user following the #Users... line. These are pulled in automatically from the User List when the Generate button is clicked.

Receiving SNMPv1 Generic Traps

There are a number of generic traps which do not have traditional SNMP trap OIDs. These are encoded by snmptrapd per RFC 2576 and then forwarded to csiTrapHandler. In particular, the traps are as follows:

generic-trap parameter	snmpTrapOID.0
0	1.3.6.1.6.3.1.1.5.1 (coldStart)
1	1.3.6.1.6.3.1.1.5.2 (warmStart)
2	1.3.6.1.6.3.1.1.5.3 (linkDown)
3	1.3.6.1.6.3.1.1.5.4 (linkUp)
4	1.3.6.1.6.3.1.1.5.5 (authenticationFailure)
5	1.3.6.1.6.3.1.1.5.6 (egpNeighborLoss)

If you wish to receive these traps, you may need to add a filter line as illustrated above. Then use the OIDs indicated here as the Trap OID. For the Var OID, simply use the sender community OID 1.3.6.1.6.3.18.1.4.0 to give the trap receiver something to look for. There is no real data payload in this type of trap. You would configure the receive rule to set a fixed value upon receipt of this trap since the trap itself is really the value you are interested in.

Two Level Trap Logging

SNMP Logging File

Logging enabled: Yes ▾ Log file: /home/customer/logs/snmp 

When you enable trap logging, there are actually two levels of logging available. Traps received by snmptrapd are logged as a minimum. With additional steps, a second level will be the logging of traps handled by the csiTrapHandler. The snmptrapd service is responsible for receiving traps at the protocol stack level. It then hands off received traps to the trap handler for further processing that includes matching OIDs to trap receive rules and potentially placing received data in local data objects.

The trap handler logging is enabled by the existence of a log file named “csiTrapReceiver_info_log” in the /home/customer/logs/ directory. The snmptrapd service logging is enabled by the “enable logging” selection at the bottom of the SNMP Trap Receiver’s Config File page. Any filename can be used for snmptrapd logging.

There is one catch to the dual logging. If you use only `csiTrapReceiver_info_log` to enable logging, then both `snmptrapd` and `csiTrapHandler` will be writing to the same file, and they will overwrite each other. To avoid this, do the following at the bottom of the Trap Receiver's Config File page:

- 1) Enter `csiTrapReceiver_info_log` in the Log file window, select Yes, and click Select Logging Options.
- 2) Without disabling logging, enter a different filename in the Log file window, keep Yes selected, and click Select Logging Options again.

You now have two logging files active. The `csiTrapReceiver_info_log` file will be used by `csiTrapHandler`, and the second file, whatever it is named, will be used by the `snmptrapd` service.

Selecting No and clicking Select Logging Options will disable logging to both files. The `csiTrapReceiver_info_log` will be renamed `expiredTrap_info_log` when logging is disabled (since the mere existence of `csiTrapReceiver_info_log` will trigger logging by `csiTrapHandler`).

Both of these log files can be very useful diagnostic tools to help get your trap receiver rules figured out. If traps do not appear in the log file for `snmptrapd`, check your authentication settings. The traps will not appear in the `csiTrapHandler` file if they do not first appear in the `snmptrapd` log file. If traps appear in the `snmptrapd` log file but not the `csiTrapHandler` log file, you may need to add additional filters to the `snmptrapd.conf` file. The logging done by `csiTrapHandler` goes one step further and inserts comments such as "Found IP address", "Found trap OID", and "Found var OID" indicating its progress and success (or not) in matching rules to the trap.

SNMP Trap Receiver Snmptrapd.conf File

Dashboard / Protocol Configuration / SNMP Trap Receiver

Rules

Config File

Snmpttrapd.conf

Snmpttrapd.conf File Generator

The SNMP Trap engine that responds to Trap messages requires a configuration file named `snmptrapd.conf` to direct its functionality, primarily in terms of authorization. This page allows automated generation of that file, but with the option to manually edit it.

The SNMP engine in this IoTServer is the widely used open source Net-SNMP package. If you are concerned about manually editing the snmptrapd.conf file, simple search the Internet for Net-SNMP documentation - there is much to be found.

Trap receiver settings

Listening port:	<input type="text" value="162"/>
Auth Community (SNMP v1/v2):	<input type="text" value="oakpark"/>

The port on which SNMP listens for Trap messages defaults to the standard port 162. You may change it here if you wish.

Authorization for access in SNMPv1 and SNMPv2 is very simply. You only need to match the community names which are treated sort of like a password. If using SNMP v1 or v2 (v2c), enter your community strings here. Trap messages received as v1 or v2 will be expected to match this community.

[Generate Config](#)

Note: Generating the configuration here will not rewrite SNMP Trap Receiver Configuration. It will generate a preview of the configuration file. User authentication information is taken from user settings found on the Users page. After generating the preview, you can then save the preview as a file below. Then go to the Config File page to import that file into the actual trap receiver.

Click the Generate Config button to auto-generate an snmptrapd.conf file. Parameters included in the file will be taken from two places: (1) The entries showing above on this page. (2) User settings from the System -> Users page (if SNMPv3).

SNMPv3 enforces access only by known users. These users may be a person or may be another machine, but must be defined as a "user" either way. To permit SNMPv3 Traps (or Informs) to be received, go to the System -> Users page and create a "user" for each person or machine that will send Traps to this IoTServer. Once the users are created, and they have been selected for SNMPv3 Trap access, they will be automatically included in the snmptrapd.conf file generated here.

Preview of generated SNMP Trap Receiver configuration

```
snmpTrapdAddr 162

#Users for agent defined here
createUser -e 0x800000000300c0b7eb4e10 jimhogenson MD5 [REDACTED]
authUser execute jimhogenson

#Trap handler - Do not change this next line
traphandle default /usr/bin/csiTrapHandler

#SNMP V1/V2 Specific settings
disableAuthorization yes
authCommunity execute oakpark
```

Save config to file:

✓ Save to file

Once the interim snmptrapd.conf file is generated, it may be viewed and optionally edited here. The file content displayed here is not actually saved to a file until you click the Save button at the bottom. When Save is clicked, the file name given will be created or overwritten with the content currently displayed.

There are two more steps required to complete the reconfiguration of the SNMP Trap Receiver. Go to the [SNMP Trap Receiver Config File](#) page, select the newly created interim snmptrapd.conf file, and click the Load button (as discussed under “SNMP Trap Receiver Config File”). Finally, after loading the new snmptrapd.conf file, click Restart SNMP at the bottom of the Config File page.

SNMP Trap Receiver XML Files

Example XML file

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!-- IoT Server SNMP Trap Receive configuration file -->

<configuration>

<trap_recv_rules>
<rule peername="192.168.1.67" resultobj="22" trapoid="1.3.6.1.4.1.3815.1.3.1.2.0"
varoid="1.3.6.1.4.1.3815.1.3.1.1.1.1.2.1.0"/>
</trap_recv_rules>
```

</configuration>

<trap_recv_rules> section

Rule Number is implied by order in XML, primarily used for database lookup of a specific rule. The rule number has functional significance in some applications, but not when it comes to trap receive rules.

peername="xxxx" – Specifies the peername that the trap is expected to be received from. If an IP address is given, then the IP address of the incoming trap message is checked. If a host name is given and the incoming trap message contains a host that could be looked up (via DNS lookup), then the host name field is checked instead of IP address.

trapoid="x.x.x.x..." – Specifies the Trap OID that is expected. Traps and Informs always contain a Trap OID identifying the trap or inform, along with additional varbinds with a data payload. The Trap OID must match the OID given here before action will be taken.

varoid="x.x.x.x..." – Specifies the varbind OID to look for in the message if the peername and Trap OID match.

resultobj="n" – Specifies the local object where data retrieved from the varbind called out by varOid will be placed assuming the varOid is found, and peername and trapOid match.

hint="n" – Provides direction on how to interpret data in the event the value type is Octet String. All ASN data types are interpreted according to their type; however, Octet String will default to being interpreted as a character string unless this hint says otherwise. The exceptions are hint="1" to interpret 4-octet Octet Strings as 32-bit floating point per RFC 6340, and hint="2" to interpret 8-octet Octet Strings as 64-bit floating point per RFC 6340. If the hint is non-zero but the Octet String length is not 4 or 8, then it will still be treated as a character string.

fixed="n" – When non-zero, specifies that this fixed value should be placed in the local object when this trap is received, rather than try to derive a data value from the trap message.

timeout="n" – Integer number of seconds for trap receive timeout. Note that this is not a session timeout, it provides for an automatic reset of the local object value. If zero, then the timeout feature is disabled.

tval="n.nn" – Specifies the fixed value that should be placed in the local object after the timeout period has expired. The typical scenario is that either the value gleaned from the trap message itself, or the fixed value, would be placed in the local object upon receiving the trap. Then 'timeout' period later, the timeout value is placed in the local object, replacing the trap value. Since traps sometimes indicate only when a condition is present, and not when the condition ceases to exist, this provides a means for automatic resetting of the condition. In particular, UPS systems with RFC 1628 will automatically resend traps indicating alarm conditions, but simply stop sending them when the alarm goes away. The only way to have an alarm state indicator is to

use a combination of fixed value and timeout value, and have the timeout period set to just longer than the trap repeat time.

SNMP Trap Receiver CSV Files

A CSV file may be imported to configure various aspects of the IoTServer (or Babel Buster gateway). A single CSV file may contain multiple sections. When a file including an “Objects” section is imported by the Data Engine, local objects will be configured. When a file including one or more “Modbus” sections is imported by an instance of the Modbus Engine, Modbus gateway functionality will be configured. The same Modbus file may be imported by a Modbus Client or Modbus Server, and either RTU or TCP, and only those sections of interest to that Modbus function will be imported. The CSV file may also contain one or more SNMP sections, and so forth.

A section begins when the word “Begin” appears in the first column of a line. All lines up to and including a line that begins with the word “End” will be taken to be part of that section.

The line immediately following the “Begin” line must be a header line. A Header line is one which labels the columns of data that will follow the Header line.

All lines following the Header line are data lines that are expected to contain the same number of columns as the Header line, and whose contents are defined by the labels found in each column of the Header line.

Labels in the section Begin and End lines, and labels in the Header line are NOT case sensitive and will be interpreted equally whether upper case, lower case, or some combination of both (for readability).

Labels may NOT contain embedded spaces. A label is terminated by a comma, line-end, or space. Labels may not be encapsulated in quote characters; however, data content in data lines may be encapsulated in quote characters and may contain embedded spaces or blanks if quoted.

Some labels in the Header line may be considered optional. The minimum required columns are indicated in the definition of each data section.

Columns in the Header line do not have to follow any particular order. They may be rearranged to the user’s liking. The only restriction is that data in subsequent data lines must match up with the labels placed in the Header line. Data lines may contain fewer columns than the Header line, but may not contain more. Data columns that the user wishes to deliberately omit, but omit between included columns, should be indicated by place holder commas (which will simply appear as blank cells in a spread sheet program).

A Begin line will contain three columns:

Column 1: BEGIN

Column 2: Function as noted below

Column 3: Sub-function as noted in definition of the Function.

Functions may be any of the following (with this listed expanded from time to time):

- LOCALDATA
- MODBUS
- SNMP

Sub-functions:
SNMP (client)

- DEVICES
- READMAPS
- WRITEMAPS
- WALKRULES

SNMP (agent)

- MIBVARS
- DEVICES
- TRAPSENDERULES

SNMP (trap receiver)

- TRAPRECVRULES

NOTE: The same SNMP CSV file may NOT contain both client and server sections as DEVICES becomes ambiguous. SNMP requires different applications for different purposes (csiSnmpClient, csiSnmpAgent, csiSnmpTrapRecv). A fourth application, csiTrapHandler, is also associated with csiSnmpAgent.

SNMP Trap Receiver Example

The trap receiver runs as a separate task independent of both the SNMP server (agent) and SNMP client. This allows the trap receiver to have its own “new data” flag which allows the IoTServer to function as a trap forwarder if desired. The following illustrates a trap receive CSV configuration file.

```
BEGIN,SNMP,TRAPRECVRULES
PEERNAME,TRAPOID,VAROID,HINT,FIXED,FIXEDVAL,TIMEOUT,TIMEOUTVAL,RES
ULTOBJ
192.168.1.67,1.3.6.1.4.1.3815.1.3.1.1.1,1.3.6.1.4.1.3815.1.3.1.1.1.2.1.0,NONE,N,0.000000,0,0.
000000,22
END
```

SNMP (agent/server) TRAPRECVRULES Section

Peername – Specifies the peername that the trap is expected to be received from. If an IP address is given, then the IP address of the incoming trap message is checked. If a host name is given and the incoming trap message contains a host that could be looked up (via DNS lookup), then the host name field is checked instead of IP address.

TrapOID – Specifies the Trap OID that is expected. Traps and Informs always contain a Trap OID identifying the trap or inform, along with additional varbinds with a data payload. The Trap OID must match the OID given here before action will be taken.

VarOID – Specifies the varbind OID to look for in the message if the Peername and Trap OID match.

Hint – Provides direction on how to interpret data in the event the value type is Octet String. All ASN data types are interpreted according to their type; however, Octet String will default to being interpreted as a character string unless this hint says otherwise, and the length is exactly 4 or 8 bytes (octets).

Hint Label	Description
None	No special interpretation
Float	Treat 4-byte Octet String as RFC 6340 IEEE 754 32-bit floating point
Double	Treat 8-byte Octet String as RFC 6340 IEEE 754 64-bit floating point

Fixed – Enter “Y” to specify that a fixed value should be placed in the local object when this trap is received, rather than try to derive a data value from the trap message. Enter “N” to parse data from the trap varbind to be placed in the local object.

FixedVal – The fixed value to be placed in the local object any time this trap is received, assuming the Fixed column indicates “Y”. This feature is most often used in conjunction with the Timeout feature.

Timeout – Integer number of seconds for trap receive timeout. Note that this is not a session timeout, it provides for an automatic reset of the local object value. If zero, then the timeout feature is disabled.

TimeoutVal – Specifies the fixed value that should be placed in the local object after the timeout period has expired. The typical scenario is that either the value gleaned from the trap message itself, or the fixed value, would be placed in the local object upon receiving the trap. Then ‘timeout’ period later, the timeout value is placed in the local object, replacing the trap value.

ResultObj – Specifies the local object where data retrieved from the varbind called out by VarOID will be placed (or fixed value will be placed) assuming the VarOID is found, and Peername and Trap OID match.