

# Rifidi Edge Server User's Guide

---

*Version 2.2 (Rifidi Edge Server version 2.2)*

*July 2013*

## Table of Contents

Table of Contents .....	2
Documentation .....	4
Overview of Rifidi Edge Server Directory Structure.....	4
Important Folders within the Edge Server Directory.....	4
Important Files within the Edge Server Directory .....	5
Starting the Edge Server .....	5
Using the Console to Determine the Rifidi Edge Server State .....	5
Creating Readers Configurations .....	6
Reader Adapter Guides.....	7
Alien Reader Guide .....	7
Properties.....	8
Connection properties: .....	8
General properties: .....	8
Commands .....	8
Alien-Poll: .....	8
Alien-Push-Start: .....	9
Alien-Push-Stop:.....	9
Common Usage Scenario: .....	9
For a Push:.....	9
For a Poll: .....	9
LLRP Reader Guide .....	10
Properties.....	10
Connection properties: .....	10
General properties: .....	11
Commands .....	11
LLRP-Configure .....	11
LLRP-Push-Stop .....	11
LLRP-Poll.....	11
LLRP-ADD_ROSPEC-File.....	11
Common Usage Scenario .....	12

For a Push:.....	12
For a Poll: .....	12
Generic Reader Guide .....	13
Properties.....	13
Commands .....	13
Awid Reader Guide .....	14
Properties.....	14
Commands .....	14
Awid-Read-Block-Data .....	14
Awid-Mask-Poll .....	14
Awid3014-Poll/Awid2010-Poll .....	14
Awid3014-Push-Stop/Awid2010-Push-Stop .....	14
Common Usage Scenario: .....	15
For push: .....	15
For poll: .....	15
Opticon Reader Guide.....	15
Properties.....	15
Commands .....	16
ThingMagic Reader Guide.....	16
Properties.....	16
Commands .....	16
ThingMagic-Poll: .....	16
Thinkify Reader Guide.....	16
Properties.....	17
Reference .....	17
System Properties .....	17
Configuration Commands .....	18
Diagnostic Commands .....	19

## Documentation

There are several places to look for specific documentation needs

- The User Documentation (this document) is a PDF that explains how to run and control the Rifidi Edge Server
- The Developer Documentation explains how to develop reader adapters and plugins for our Edge Server
- The Javadoc for the Rifidi API can be found online at <http://www.transcends.co/www/javadoc/edge2.0.0/>
- The wiki has some helpful pages to answer some common questions that users and developers have. It can be found at <http://wiki.rifidi.net>

The wiki has also a quick start guide for getting a reader setup with a basic configuration:

[http://www.transcends.co/www/docs/Rifidi\\_Edge\\_Server\\_Quick\\_Start\\_Guide\\_2.1.pdf](http://www.transcends.co/www/docs/Rifidi_Edge_Server_Quick_Start_Guide_2.1.pdf)

The forums provide a way for users and developers to ask questions. It can be found online at <http://forums.rifidi.net>

## Overview of Rifidi Edge Server Directory Structure

This section explains the directory structure of the Rifidi Edge Server. It points out the important files and folders that users should be aware of, and explains how to modify them in order to meet particular configuration needs.

### Important Folders within the Edge Server Directory

- Applications Folder  
The applications folder is the folder in which custom Rifidi Edge Server applications should be deployed. By default, the applications will not be started when the edge server starts. If this behavior is desired, use the default.ini file.
- Config directory  
This folder contains all the Rifidi Edge configuration files, such as the rifid.xml file, the rifid-amq-external.xml, and logging.properties file.
- Logs Folder  
The logs folder contains all log files generated by the Rifidi Edge Server, which is useful for debugging and troubleshooting problems
- Licenses Folder  
The licenses folder contains all license information for the Rifidi Edge Server and all its dependencies.
- Plugins & Configuration Folders  
These two folders contain the jar files and configuration information for the Rifidi Edge Server. Users should not normally ever need to edit any files in these folders.

## Important Files within the Edge Server Directory

- **rifidiedgeserver executable (linux)**  
This is the executable to start the server. While it is possible to execute this file right from the shell, it's better to use the start up script located in `/etc/init`. You can edit this file to change any system properties for linux.
- **config/logging.properties**  
This file is a log4j properties file that lets the user have fine-grained control over the logging. For more information, see <http://logging.apache.org/log4j/1.2/index.html>
- **config/rifidi.xml**  
This is the persistence file for all reader configurations. When the server stops and restarts, these configurations will be automatically loaded, and they will return to their saved state.
- **config/rifidi-amq-external.xml**  
The ActiveMQ configuration file
- **rifidiserver.ini (windows)**  
This file contains java system properties. They are loaded when the server starts up and affect things like which network interfaces and ports certain processes (such as JMS, RMI, and web services) are started at. In addition, custom rifidi applications may supply their own system properties using this file.
- **applications/default.ini**  
The applications folder contains a file called 'default.ini' which is a list of applications that should be started up when the Rifidi Edge Server starts. The name of the application is simply the name of the folder that contains the application. Each application should be located on a separate line in the default.ini file. If this file doesn't exist, you can create it.

## Starting the Edge Server

To start the edge server run the appropriate executable. In linux:

```
> sudo chmod +x rifidi-server
> ./rifidi-server
```

In windows, just double click on the rifidiserver.exe file.

## Using the Console to Determine the Rifidi Edge Server State

The Rifidi Edge Server console allows users to find out useful, detailed information about the current state of the edge server. Once you are connected, issue the 'help' commands to list out all available commands.

```
osgi> help
```

The commands at the top are all commands provided by the OSGi framework itself to display information and control the bundles. The commands at the bottom (under the Rifidi Edge Server section) are all commands provided by the Rifidi Edge Server. One useful command is the 'readers' command, which lists all available reader configurations and their state:

```
osgi> readers

ID: Awid2010_1
  session (1): IPSession: 192.168.1.91:4000 (CREATED),
               [GPIO Session IPSession: 192.168.1.91:4001 (CREATED)]
               Recurring Command(0): Awid2010_Push_Start_1

ID: Alien_1
  session (1): IPSession: 192.168.1.120:23 (PROCESSING),
               [Autonomous Session IPServerSession: 54321 (PROCESSING)],
               [GPIO Session IPServerSession: 54322 (PROCESSING)]
```

In this case, there are two reader configurations. The first is called `Awid2010_1`. It has two interactive sessions that are both tied to session ID 1. One is for reading tags (at 192.168.1.91:4000), and the other for interacting with GPI/O events (at 192.168.1.91:4001). Neither session is connected, since they both say they are in the `CREATED` state. In addition there is a recurring command scheduled to execute once the session connects.

The second configuration has an ID of `Alien_1`. It has three sessions, all tied to session ID 1. The interactive session (at 192.168.1.120:23) is used for issuing commands to the reader. The autonomous session (at port 54321) simply listens for tags that the Alien reader pushes to it. The GPIO session (at port 54322) is another passive session that listens for the Alien reader to push GPI/O events to it. There are no commands scheduled on the session, and the session is currently active since the states of the sessions are `PROCESSING`.

## Creating Readers Configurations

It is possible to use the OSGi console to create and control reader configurations. To list the available reader adapters use the `'readertypes'` command:

```
osgi> readertypes
ThingMagic
LLRP
Alien
Awid3014
Awid2010
Alien-Autonomous
```

To create a configuration to connect to an Alien reader, use the `'createreader'` command.

```
osgi> createreader Alien IPAddress 192.168.1.8 Port 23
Reader Created with ID Alien 1
```

Each reader adapter has its own set of properties (e.g. `IPAddress`, `Port`) that allow users to specify connection information. For details see

[http://wiki.rifidi.net/index.php?title=Edge\\_Server\\_Sensor\\_Plugins](http://wiki.rifidi.net/index.php?title=Edge_Server_Sensor_Plugins) .

Next, create a session:

```
osgi>createsession Alien 1
```

```
Session created: Alien_1:1
```

For this particular reader, we will schedule a recurring command that polls the reader to ask for tags. To list the available command types, use the 'commandtypes' command:

```
osgi> commandtypes
ID: Awid-Read-Block-Data
ID: Awid-Mask-Push-Start
ID: Awid3014-Push-Stop
ID: LLRP-Configure
ID: Awid3014-Push-Start
ID: LLRP-Push-Stop
ID: LLRP-Poll
ID: Awid2010-Push-Stop
ID: ThingMagic-Poll
ID: Awid2010-Push-Start
ID: Alien-Push-Stop
ID: Alien-Push-Start
ID: Alien-Poll
```

We need to create a new Alien-Poll command and submit it to the session:

```
osgi>createcommand Alien-Poll
Command Configuration Created with ID Alien_Poll_1

osgi>executecommand Alien_1 1 Alien_Poll_1 1000
Command submitted
```

Finally, start the session and save the configuration so that it will automatically start if the edge server is stopped and restarted.

```
osgi> startsession Alien_1 1
Session Alien_1:1 started

osgi> save
Configuration Saved!
```

## Reader Adapter Guides

This section is a guide to all of the various reader guides offered out-of-the-box. It explains how to use each of the reader adapters to connect to the specific kind of reader and collect tags back from that reader.

### Alien Reader Guide

This section represents an updated user's guide for communicating with an Alien reader through RifiDi Edge. The instructions should work with any Alien reader as they all share the same protocol but it has been tested on the 8800, 9800 and 9900 series of readers.

The general outline to create a session can be found in the 'Rifidi Getting Started Guide'. The only differences are the properties set which are outlined below.

## Properties

These are the properties of the Alien reader which can be adjusted (except for read-only properties) by a "setproperties" command.

### *Connection properties:*

- MaxNumConnectionAttempts: The number of connections that the Edge Server will try to connect to the reader before giving up. -1 is equivalent to infinite.
- Username: The username used to connect to the reader.
- Password: The password used to connect to a reader.
- IpAddress: The IP of the physical reader.
- Port: The port that the physical reader uses to connect with the Edge Server.
- ReconnectionInterval: The amount of time in milliseconds that the Edge Server will wait after failing to connect before attempting to reconnect to a reader.

### *General properties:*

- Uptime: The time in milliseconds that the reader has been turned on. (Read only)
- MACAddress: The MAC address of the reader. (Read only)
- ReaderType: The type of reader you are connected to (9800, 9900, etc). (Read only)
- PersistTime: The amount of time a tag will be persisted in memory once it is read. If this value is set to -1 the tags will be persisted indefinitely until they are returned through a command.
- ExternalOutput : The bit mask representation of the current output of the reader. (Read only)
- DisplayName: The name that will be displayed for this reader.
- IOStreamPort: The port to open up to listen to IOEvents from the reader
- MaxAntennas: The maximum number of antennas this reader can support. (Read only)
- ReaderVersion: The firmware version that this reader has. (Read only)
- NotifyPort: The port that the Edge Server will use to receive Autonomous connections.
- InvertExternalOutput: Flip the values of the external output.
- ExternalInput: The bit mask representation of the current input of the reader. (Read only)
- InvertExternalInput: Flip the values of the external input.

## Commands

This is a list of the commands that can be used when communicating with the Alien reader.

### *Alien-Poll:*

This command will poll the alien reader with "get taglist" commands for a given duration, returning any tags seen by the reader. It has one property:



- **TagType:** The type of tag that will be searched for. “0” will search for only Gen1 tags. “1” will search for only Gen2 tags. “2” will search for all kinds of tags. Some readers, such as the 9900, can only read Gen2 tags. Make sure this value is set to “1” with such readers.

#### *Alien-Push-Start:*

This command will start the Alien reader pushing tag reads back to a given IP and port within a given time.

- **AutoStopTimer:** The amount of time between reports returning, in milliseconds.
- **NotifyAddressPort:** Set this to the same value as the “NotifyPort” in the reader properties.
- **NotifyAddressHost:** The IP of the machine the Edge Server is running on.

Set the NotifyAddress to the IP that Edge is running on, and set NotifyPort to the port that is set in the properties of the reader. The execute interval is set to -1 because we only want the command to run once (all the command does is turn on the AutoMode and NotifyMode properties of the reader). After running these commands, assuming the properties are set correctly, you should have tags coming back to the reader.

After this the final piece is to set up the Reader to go into Autonomous mode and send messages to the port that Rifidi Edge is listening on. This is best done through the web interface or admin tool of the reader.

#### *Alien-Push-Stop:*

This command will turn off the Autonomous and Notify modes for the reader. It has no properties.

### **Common Usage Scenario:**

#### *For a Push:*

```
//Create an Alien Push command
> createcommand Alien-Push-Start

//Create the autonomous command
> setproperties Alien_Push_Start_1 NotifyAddressPort 54321
NotifyAddressHost 192.168.1.201

//Executes the stop command
> executecommand Alien_1 1 Alien_Push_Start_1 -1

//Create the stop command
createcommand Alien-Push-Stop

//Execute the stop command once
> executecommand Alien_1 1 Alien_Push_Stop_1 -1
```

#### *For a Poll:*

```
//Create the Alien Poll command
```

```

> createcommand Alien-Poll

//Starts polling the reader every 1000 milliseconds
> executecommand Alien_1 1 Alien_Poll_1 1000

//Stops and deletes the command
> deletecommand Alien_Poll 1

```

## LLRP Reader Guide

This section will give an overview of the commands and properties associated with the LLRP module in Rifidi Edge.

LLRP is an EPCGlobal specification for communication in a common language with LLRP Compliant readers. The Rifidi Edge Server supports all LLRP commands as outlined by the specification. However, many readers also provide custom extensions so you should be sure to test your reader prior to use.

If you want more help using an LLRP reader, try LLRP Commander:

<http://www.fosstrak.org/llrp/index.html>

LLRP Commander can help you visualize LLRP messages very easily, and it can also help you build XML messages to use with the “LLRP-ADD\_ROSPEC-File” command.

We have tested our LLRP Adapter with the following Readers:

Note: In our experience the LLRP Adapter should work with any LLRP compliant reader

- Impinj
- Motorola (XR Series) – any LLRP compliant readers
- Sirit
- Thingmagic

Note on the Motorola: A setting is enabled by default which caused the reader to not send back any tags. With version 2.2 we have fixed the default configuration in such a way that all Motorolas should now behave correctly.

## Properties

### *Connection properties:*

These properties are used when connecting to the reader.

- ReconnectionInterval: The amount of time in milliseconds that the plugin will wait when a connection attempt fails before attempting to reconnect.
- Port: The port that will be used to connect to the physical reader.
- IpAddress: The IP that will be used to connect to the physical reader.
- MaxNumConnectionAttempts: The maximum number of attempts to connect that the plugin will make before giving up.

### *General properties:*

- **DisplayName:** The name of the reader that will be displayed.
- **ReaderConfigPath:** The path of an optional .llrp file which can be used to configure the reader when the session starts. Simply create a “SET\_READER\_CONFIG” command in LLRP Commander (URL), and export the resulting XML into the desired file.

## **Commands**

### *LLRP-Configure*

This command will configure a reader to send tags back to Rifidi Edge. It has 4 properties:

- **ROSpecID:** The ID of the ROSpec you want to create.
- **AntennaID:** The IDs of the antennas that you want to look for tags on (0 means all antennas).
- **Duration:** (For PUSH trigger only) How often the ROSpec will push tag lists back to the reader (in milliseconds).
- **TriggerType:** (PUSH or POLL) Which kind of trigger will be used to generate ROResults. PUSH means that the reader will push ROResults back to the reader asynchronously. POLL means that a LLRP-Poll command must be created that will synchronously poll the LLRP reader and send tags back. If you aren't sure which kind you want to use, use PUSH and set the duration to the value you want.

### *LLRP-Push-Stop*

This command will delete the ROSpec on the reader that is pushing reports back to Edge. It has 1 property:

- **ROSpecID:** The ID of the ROSpec that you want to delete. “0” will delete all ROSpecs.

### *LLRP-Poll*

This command will constantly poll a ROSpec on the reader for reports. To use this command, first set up a ROSpec with the “POLL” option with the LLRP-Configure command. After that, create an execute an LLRP-Poll command on that reader with a desired interval, and the reader will constantly poll the ROSpec created for tags. It has 1 property:

- **ROSpecID:** The ID of the ROSpec that you wish to receive tag reports from.

### *LLRP-ADD\_ROSPEC-File*

This command allows you to add a ROSpec from a file. You can configure the file using the excellent LLRP Commander then you can export that file and submit it to the reader using this command. This command will delete all ROSpecs that are on the reader at the moment, and it will add and enable the ROSpec that you submit (it is assumed to have a start trigger that isn't NULL, if it is NULL you will have to edit the file and re-submit or manually start the ROSpec; if you aren't sure what trigger to use use an IMMEDIATE trigger). This command has 1 property:

**Filename:** The name of the file that you wish to submit to the reader. The location of the file should be in the /rifidiserver/sensorconfig directory.

- createcommand LLRP-ADD\_ROSPEC-File
- setproperties LLRP\_ADD\_ROSPEC\_File\_1 Filename myrospec.llrp
- executecommand LLRP\_1 1 LLRP\_ADD\_ROSPEC\_File\_1 -1

This sequence of commands will look at the file “myrospec.llrp” in your /rifidiserver/sensorconfig directory and craft a ROSpec out of it. All existing ROSpecs will be deleted, and the new ROSpec will be sent, enabled and started. The parameters of how the ROSpec is configured are completely up to you. If you need help crafting a ROSpec, try LLRP Commander.

### Common Usage Scenario

#### *For a Push:*

This is a sequence of commands that will create, execute, and stop reading tags from the LLRP in a “Push” fashion.

```
//Create and execute an LLRP-Configure command
> createcommand LLRP-Configure

//This sets the ID of the ROSpec we will create to 1 and the amount
of time between reports to 1000
> setproperties LLRP_Configure_1 TriggerType PUSH ROSpecID 1
Duration 1000
> executecommand LLRP_1 1 LLRP_Configure_1 -1

//Stop the command
> createcommand LLRP-Push-Stop

//ROSpec 1 is the ROSpec that will be stopped and deleted
> setproperties LLRP_Push_Stop_1 ROSpecID 1
> executecommand LLRP_1 1 LLRP_Push_Stop_1 -1
```

#### *For a Poll:*

This is a sequence of commands that will create, execute, and stop reading tags from the LLRP in a “Poll” fashion.

```
//Create an LLRP Poll command
> createcommand LLRP-Poll

//Sets the ROSpecID to 1
> setproperties LLRP_Poll_1 ROSpecID 1

//Executes the command every 1000 milliseconds
> executecommand LLRP_1 1 LLRP_Poll_1 1000

//Stop the poll command (by deleting it)
> deletecommand LLRP_Poll_1
```

## Generic Reader Guide

The Generic Reader Adapter is provided for any device where you are looking for a simple way to connect and pass data (such as tag data) to the Rifidi Edge Server. Ways in which we have used the Generic adapter is with handheld and smartphone applications.

This reader will open up a ServerSocket at the port specified, and it will receive data in according to a certain format that is outlined below. It has no properties except for the port, and it has no commands. You can either send the data directly, if the reader supports pushing and the data can be formatted as you choose, or you can write a small module that will receive data from the reader and send it to the Generic reader from there.

If you need to get a handheld reader to work with Rifidi Edge, use this as a base. The general outline to create a Generic Reader Session can be found in the 'Rifidi Getting Started Guide'. The only differences are the properties set which are outlined below.

### Properties

There is only one property for the Generic reader: the port. When you start the reader, it will create and start a TCP ServerSocket at this port.

- Port: The port that the ServerSocket will listen to.

```
> setproperties Generic 1 Port 2121
```

### Commands

There are no commands for the Generic reader. Instead, it will listen for data coming in to the specified ServerSocket in this format:

```
ID:(tag ID)|Antenna:(antenna)|Timestamp:(millis since epoch)
```

It is assumed the tags being read in are Gen2 tags. All data will be formed into a TagReadEvent and put into Esper.

If you like, you can specify other information which can be added into the TagReadEvent. Simply add more key:value pairs, separated by pipes. For instance, if you wanted to give the Speed and RSSI values as well as the regular values, you could format the incoming messages like this:

```
ID:(tag ID)|Antenna:(antenna)|Timestamp:(millis since epoch)|Speed:(speed value)|RSSI:(rssi value)
```

So an incoming tag might look like this:

```
ID:350009080001|Antenna:1|Timestamp:123456789|Speed:.03|RSSI:885.7
```

Any values besides the ID, Antenna, and Timestamp will go in the ExtraInformation, which is a String:String hashmap. The value will be inserted as a value for the corresponding key (so, "Speed" would be inserted with the key "Speed" and the value "0.3").

## Awid Reader Guide

This section represents the user's guide for communicating with an AWID reader through Rifidi Edge. There are separate readers for the 2010 and the 3014, however the steps are almost exactly the same unless otherwise specified. All examples given here will use the 2010 reader.

### Properties

- **ReconnectionInterval:** The amount of time in milliseconds that the Edge Server will wait after failing to connect before attempting to reconnect to a reader.
- **Port:** The port that the physical reader uses to connect with the Edge Server.
- **Host:** The IP of the physical reader.
- **MaxNumConnectionAttempts:** The number of connections that the Edge Server will try to connect to the reader before giving up. -1 is equivalent to infinite.
- **DisplayName:** The name that will be displayed for this reader.

### Commands

#### *Awid-Read-Block-Data*

This command will read a specific memory bank of tags in view of the reader.

- **MemoryBank:** The memory bank to read from.
- **createcommand** Awid-Read-Block-Data
- **setproperties** Awid\_Read\_Block\_Data\_1 MemoryBank 2

```
> executecommand Awid2010 1 1 Awid_Read_Block_Data_1 -1
```

#### *Awid-Mask-Poll*

This command configures the Awid reader to send back tags using the Gen 2 Portal ID With Mask command. For more on this command, refer to the Awid documentation for a detail listing of the properties. The general usage of properties is to first list the property name followed by the value.

- **Timeout:** Execute this command for timeout\*100 ms. If set to 0x00, execute until stop command is sent.
- **Repeat:** Return results every repeat\*100ms. If set to 0, continuously return tags.

Refer to the Awid documentation for the properties not specified. See below for an example of how to use the Awid-Mask-Poll command to “push” data back to the edge server.

#### *Awid3014-Poll/Awid2010-Poll*

This command polls the Awid reader for any tags on its field. It has no properties. See below for an example.

#### *Awid3014-Push-Stop/Awid2010-Push-Stop*

This command should be sent if you wish to stop an Awid-Mask-Poll command that is pushing tags back (this will only happen if the “Timeout” value is set to 0).

## Common Usage Scenario:

### For push:

```
//Create the poll command
> createcommand Awid-Mask-Poll

//Set the Timeout to 0. Now the Awid will push back tags.
> setproperties Awid_Mask_Poll_1 MemoryBank 3 Timeout 0

//Execute the command as a one-time
> executecommand Awid2010_1 1 Awid_Mask_Poll_1 -1

//Create the stop command
> createcommand Awid2010-Push-Stop

//Stop the Awid from pushing
> executecommand Awid2010 1 1 Awid2010 Push Stop 1
```

### For poll:

```
//Create the poll command
> createcommand Awid2010-Poll

//Execute the poll command every 1000 milliseconds
> executecommand Awid2010_1 1 Awid2010_Poll_1 1000

//Delete the poll command to stop it
> deletecommand Awid2010_Poll_1
```

## Opticon Reader Guide

This document represents an updated user's guide for communicating with an Opticon Barcode reader. In order to get an Opticon reader to work, you will have to connect it to your computer and set it to USB-to-Serial bridge mode. This will create a virtual serial port which you may receive barcode information from.

You will also need to configure the barcode reader to send the data back in the correct format. Here is a sample configuration which will configure the options correctly. The first 7 options, starting with USB VCP and ending with ^C (ETX) are mandatory, but the last 5 are optional and can be changed or eliminated.

The default port in Linux that the Opticon will connect to is /dev/ttyUSB0. If that port is in use, it will instead default to /dev/ttyUSB1, and so on. If the Opticon is using /dev/ttyUSB0 and it becomes unplugged while it is running, once it is plugged back in it would go to /dev/ttyUSB1. This means that your SerialPort property will be incorrect, and the Opticon won't work. To solve this, stop the reader session first if you ever want to unplug the Opticon. This will release the /dev/ttyUSB0 port, and once the Opticon is plugged back in it will use /dev/ttyUSB0 again.

### Properties

- SerialPort : This is the virtual serial port that the Opticon will connect to.

- DisplayName : The name that will be displayed for this reader.

## Commands

There are no commands for the Opticon reader, as you cannot send data back to it directly.

## ThingMagic Reader Guide

This section represents an updated user's guide for communicating with a Thingmagic reader. This plugin should work with both Mercury 4 and Mercury 5 readers; it may work with other Thingmagic readers as well.

## Properties

- ReconnectionInterval : The amount of time in milliseconds that the Edge Server will wait after failing to connect before attempting to reconnect to a reader.
- Port : The port that the physical reader uses to connect with the Edge Server.
- IPAddress : The IP of the physical reader.
- MaxNumConnectionAttempts : The number of connections that the Edge Server will try to connect to the reader before giving up. -1 is equivalent to infinite.
- DisplayName : The name that will be displayed for this reader.

## Commands

### ThingMagic-Poll:

This command will poll the reader for tags at a rate you specify. You may specify to search for certain types of tags, however if you set all of the values to 0, all tags will be returned.

- Epc0 : Should the reader look for EPC Class 0 tags? 0 if no, 1 if yes.
- Epc1 : Should the reader look for EPC Class 1 tags? 0 if no, 1 if yes.
- Gen2 : Should the reader look for EPC Gen2 tags? 0 if no, 1 if yes.
- Iso860006b : Should the reader look for ISO860006b tags? 0 if no, 1 if yes.
- Timeout : The amount of time in milliseconds that must pass before the reader stops looking for tags and returns its results.

```
> createcommand ThingMagic-Poll
> setproperties ThingMagic_Poll_1 Ep0 0 Epc1 0 Gen2 0 Iso860006b 0
Timeout 1000
> executecommand ThingMagic_1 1 ThingMagic_Poll_1 -1
```

This sequence of commands will create a ThingMagic-Poll command and search for all tags, returning every 1000 milliseconds.

## Thinkify Reader Guide

This section represents a guide for connecting to a Thinkify TR-50 or TR-200 reader over USB. This reader has no commands: rather it simply starts reading as soon as you connect to it.

```
osgi> createreader Thinkify50 Port COM4
Reader Created with ID Thinkify50_1
```



After creating the reader you need to create and start the session. Note: the readertype “Thinkify50” is also valid for the Thinkify TR-200 reader.

## Properties

- **ReconnectionInterval** : The amount of time in milliseconds that the Edge Server will wait after failing to connect before attempting to reconnect to a reader.
- **Port** : The port that the physical reader uses to connect with the Edge Server. This will be the address of a serial port that will link to the USB device. On windows it shows on my machine as COM6. On linux it shows as /dev/ttyACM0. Please refer to Thinkify's documentation to figure out what the name of the port is for your device.
- **MaxNumConnectionAttempts** : The number of connections that the Edge Server will try to connect to the reader before giving up. -1 is equivalent to infinite.
- **DisplayName** : The name that will be displayed for this reader.
- **Duration**: The amount of time in milliseconds between fetching the tag list. Default is 1000 ms.

## Reference

This section contains a reference that documents the many system properties and command that can be used to configure the Rifidi Edge Server.

## System Properties

All properties can be found and edited in the rifidiserver.ini file (or the rifidi-server file in linux).

- **-Dorg.rifidi.edge.configuration**  
This is the path to the persistence file (i.e. rifidi.xml)
- **-Dcom.sun.management.jmxremote.port**  
The port to open up JMX on. This is important if you are using a program like VisualVM to profile the server
- **-Djava.rmi.server.hostname**  
The hostname to open up the RMI registry at. Default is 127.0.0.1
- **-Dorg.rifidi.edge.core.rmi.port**  
The port of the RMI registry Default is 1101
- **-Dorg.rifidi.edge.autostart**  
If set to true, will start saved sessions automatically if they were in the PROCESSING state when saved. If set to false, session will not automatically start, even if they were in the PROCESSING state. It might be useful to set to false for debugging purposes
- **-Dorg.rifidi.edge.logging**  
The path to the Log4J properties file to control logging
- **-Dosgi.console**  
The port to start the OSGi console at. If no argument is supplied, the console will not be bound to a port and will be available only to the process that executed the rifidiserver. If this argument is changed to another port besides 2020, the management console will be available to any computer on the same network as the Rifidi Box, which makes the application insecure.

- `-Dorg.rifidi.ui.notify`  
If set to true, reader adapters will automatically place tag read events on the JMS queue for the Workbench to consume. This comes at some performance cost, so set to false if you are concerned about performance
- `-Dorg.rifidi.home`  
The path to the RifidiHome directory, which includes the applications folder.

## Configuration Commands

The following commands allow you to configure readers and query the edge server for its current state. They can be issued from the OSGi console:

- `loadApp <Application Name>`  
Load an application that is deployed to the Rifidi Edge Server. The application will be started.
- `save`  
Save the configuration to a file (config.xml in the config directory)
- `setloglevel <loggerName> <logLevel>`  
Set the log level for a given logger.
- `readertypes`  
Get the list of available reader types
- `readers`  
Get the list of configured readers
- `createreader <readertype> [<propName> <propValue>]*`  
Create a new reader
- `deletereaders <readerid>`  
Delete a reader
- `commandtypes`  
Get the list of available command types
- `commands`  
Get the list of configured commands
- `createcommand <commandtype> [<propName> <propValue>]*`  
Create a new command
- `deletecommand <commandid>`  
Delete a command
- `configurations`  
Get the list of configurations (both reader and command configurations)
- `getproperties <id>`  
Get the properties of a configuration
- `setproperties <id> [<propName> <propValue>]*`  
Set the properties on a configuration
- `applypropchanges <readerid>`  
Apply the property changes on the configuration to the reader

- `createsession <readerid>`  
Create a new session on the given reader
- `deletesession <readerid> <sessionid>`  
Delete a session on the given reader
- `startsession <readerid> <sessionid>`  
Start a session on the given reader
- `stopsession <readerid> <sessionid>`  
Stop a session on the given reader
- `executecommand <readerid> <sessionid> <commandid> <interval>`  
Execute a command in a session
- `killcommand <readerid> <sessionid> <commandid>`  
Execute a command in a session

## Diagnostic Commands

These commands help you to figure out if your application is working properly. They are also issued from the OSGi console:

- `testGPI <readerID> <port>`  
Returns the GPI value of the given reader's port
- `setGPO <readerID> [<port>]*` - Sets the given ports to high. Any ports not mentioned that are currently high will be set to low.
- `flashGPO <readerID> <port>`  
Flashes the given GPO port high for 4 seconds.
- `simGPIHigh <readerID> <port1> [port2]*`  
Simulates setting the given GPI ports to high for the given reader
- `simGPILow <readerID> <port1> [port2]*`  
Simulates setting the given GPI ports to low for the given reader
- `simGPIFlashHigh <readerID> <seconds> <port1> [port2]*`  
Simulates setting the given GPI ports to high for the given reader, then setting them back to high after the given duration has passed
- `simGPIFlashLow <readerID> <seconds> <port1> [port2]*`  
Simulates setting the given GPI ports to low for the given reader, then setting them back to high after the given duration has passed
- `recenttags <readerID>`  
Prints out a list of tags seen on the given reader within the last 5 minutes.
- `currenttags <readerID>`  
Prints out a list of tags currently seen by the given reader.