Whisker.IO Commercial Gateway™

Technical User's Manual

Digital Six Laboratories LLC
414 Ash Avenue * Yukon, OK * 73099
Phone: (888) 551-0112
www.d6labs.com

Patent Pending

1 Table of Contents

2 Revision History

Revision	Date	Revisor	Description
0.A	11/18/2015	SJM	Initial Revision

3 Introduction

Whisker.IO™ is a comprehensive wireless IoT solution platform that consist of the following components:

1. **Whisker.IO Engine™ –** long range, battery powered wireless IoT module. Uses **LoRa™** radio technology.
2. **SensorBlock™** – Whisker.IO Engine based, pre-packaged, battery powered wireless I/O devices that can measure and monitor anything at a distance of up to 4 miles outdoors and 2500 feet indoors
3. **Whisker.IO Gateway** – Connects Whisker.IO Engine enabled devices to the cloud using a variety of standard connectivity options including Ethernet, WiFi and Cellular. The gateway can push data to any cloud storage/app server that supports the MQTT protocol
4. **Cloud Storage and Web Portal –** Storage of historical and diagnostic data, provisioning, visualization, and business logic
5. **Mobile App** – a mobile tool used for installation and maintenance of a Whisker.IO IoT network.

The primary job of the gateway is to act as a bridge between the Internet and the various Whisker.IO enabled devices. Devices can send readings and information to the gateway, which will in turn push that data to clients in the cloud. Likewise, clients can send commands and requests to the gateway, which will in turn communicate those to the devices.

The commercial gateway is designed for indoor use and is small (3x4x1.5"), self-contained, and very easy to use.

It communicates to the Whisker.IO devices using an integrated ½ wave articulated di-pole antenna.

Internet connectivity options include Ethernet, WiFi, and Cellular. The gateway communicates to the cloud using MQTT and HTTP protocols.

Each gateway is shipped with a 2A, 5V wall transformer and 3 meter micro-USB power cable.

The commercial gateway has three (3) status lights on the top of the enclosure that indicate power, cloud connectivity status and activity, and Whisker.IO wireless status and activity. These provide a quick indication of operating status and are very useful in debugging problems in the field.

Every gateway is assigned a unique serial number at the factory. This serial number reflects the MAC address of the internal Whisker.IO Engine and is used to uniquely identify the gateway globally.

4 Ordering Information

Ordering Code	Antenna Configuration
GW-900-CM1	902-928MHz, Ethernet
GW-900-CM2	902-928MHz, WiFi
GW-900-CM3	902-928MHz, Cellular
GW-868-CM1	868MHz (European), Ethernet
GW-868-CM2	868MHz (European), WiFi
GW-868-CM3	868MHz (European), Cellular

5 Specifications

- 900MHz Quad Core ARM Cortex A7 CPU
- 1GB RAM
- Linux OS
- Application software written in C#, runs under Mono
- 4 USB ports – supports external mass data storage
- Ethernet, WiFi, and Cellular connectivity
- MicroSD card – OS and data storage
- Auto software update – configurable
- MQTT and HTTP protocol support
- Integrates with Whisker.IO cloud storage and web portal
- Works with all Whisker.IO devices
- Full API via MQTT – allows clients to remotely configure gateway and interact with devices
- Whisker.IO radio communications uses **LoRa™** radio technologies.

5.1 Electrical Specifications

Parameter	Minimum	Typical	Maximum	Units
Supply voltage		5.0		VDC
Supply current – Ethernet version			650	mA
Supply current – WiFi version			750	mA
Supply current – Cellular version			1750	mA
Transmit power – programmable	13		20	dBm
Receiver sensitivity		-120		dBm
Link budget	132		139	dB
Demodulation SNR	-10	-12.5	-20	dBm (2)
RF data rate	7.031	7.031	37.5	kBaud (1)

Notes:

(1) The spreading factor and data rate varies based on signal quality. Modules closer together will communicate using a higher data rate. Modules further apart will use a lower data rate to compensate for the higher path loss. Typical Whisker engine data rate is 7kBaud.

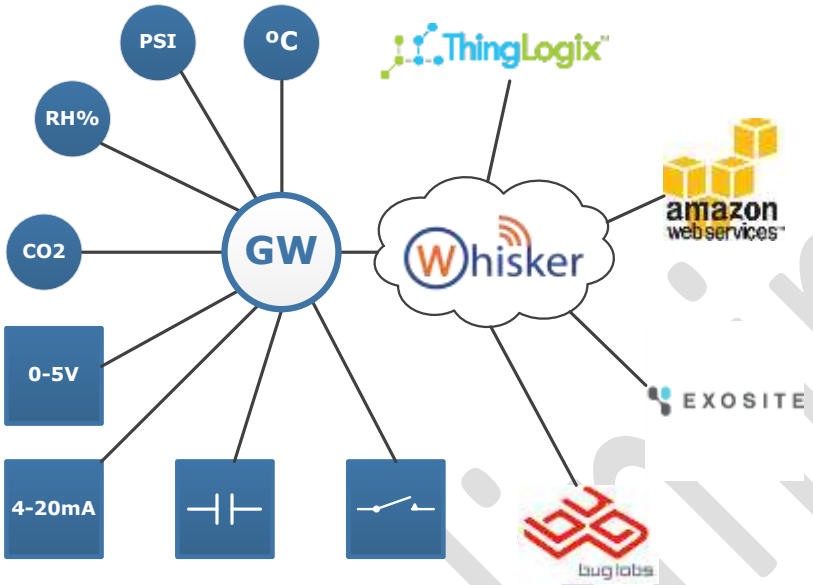
(2) Typical GFSK radio (e.g. Bluetooth or ZWave) require at least +9dB SNR.

6 Concepts and Definitions

6.1 Whisker.IO Solution Overview

The following diagram shows the general relationship between the components of the Whisker.IO solution platform.

Our SensorBlocks™ are capable of monitoring environmental conditions (indicated by blue circles). They can also monitor and control equipment and processes using a combination of digital and analog inputs and outputs (indicated by blue circles).



OEM products can also be connected to a Whisker.IO gateway using our Whisker.IO Engine. This provides a fast and inexpensive way to connect legacy devices to the Internet of Things.

Any Whisker.IO enabled device can monitor its environment using sensors and report measured data to the gateway. These reports can be scheduled periodically or can be configured to trigger when specific conditions are met.

Each Whisker.IO device must be provisioned to operate with the gateway. The link between the gateway and the device is based on the unique serial number (MAC address) programmed into each at the factory. The provisioning process configures the gateway with information about the device and the device with the serial number of the gateway.

When a device sends a report (or update) to the gateway, the gateway in turn publishes the data to the cloud using MQTT.

MQTT is a publish/subscribe protocol that allows a client to publish data to a topic and other (multiple) clients to subscribe to the topic so that they automatically get notified when new messages are available. Our topics are structured to create a map that ties the message to a gateway/device/channel relationship (this will be explained in detail in later sections).

The gateway software also subscribes to certain MQTT topics that allow clients in the cloud to push messages to the gateway. The gateway provides a complete API with the following capabilities:

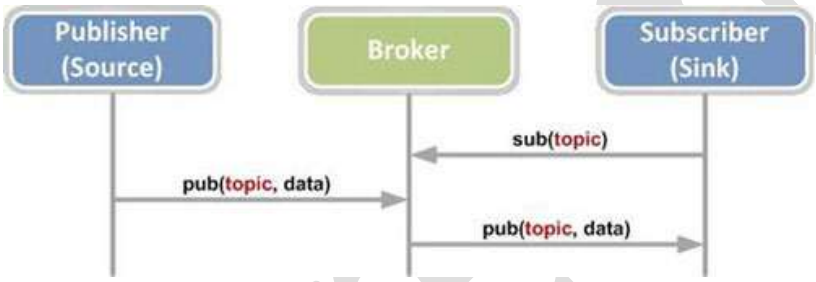
- Change gateway configuration
- Request information from the gateway (such as a list of connected devices)
- Send commands to devices
    - Change configuration of device
    - Request reading immediately
    - Change state of analog or digital output

Our Whisker.IO cloud server can be configured to receive and store messages from gateways (and devices). Customers can then use our portal to view and analyze data from the devices and configure behaviors that can perform automatic functions based on device data. Behaviors are very flexible and can be used to send text messages and emails, send commands to devices, post to REST API urls and more.

Additionally, the gateway can be configured to communicate directly with 3rd party cloud services and applications. We already have integrations for ThingLogix, Amazon, and BugLabs and we are working on integrations for ThingWorx, Exosite, and more.

6.2 MQTT

MQTT is a protocol ideally suited to IoT applications. It uses a publish/subscribe mechanism that is far more efficient than the polling mechanism required by HTTP based applications.



6.2.1 Broker

The Broker is a software application running on a computer (server) anywhere in the cloud. It must have a static IP address so that both publishers and subscribers can connect to it.

Subscribers connect to the broker and subscribe to specific topics. Publishers connect to the broker and publish messages to specific topics. When a new message is published on a topic, the broker pushes the message to all subscribers that have subscribed to that topic.

Digital Six Laboratories LLC offers a free-to-use broker on our cloud server running in Microsoft Azure.

6.2.2 Topic

A topic is a string that works very similar to a url. Topics are comprised of segments concatenated by '/'. Here is an example:

whiskerio/gw/DE1A0F78/pub/devices/DE1A0F11/AI03

Topics are hierarchal, so subscriptions can listen to groups of topics using wild cards. For example, a subscriber could subscribe to the above topic, or if the subscriber wanted to receive messages from all devices on that gateway, it could subscribe to 'whiskerio/gw/DE1A0F78/#'. For more information on topics and wildcards, visit www.mqtt.org/documentation.

6.2.3 Client

A client is a software application that connects to a broker and can both publish and subscribe to topics.

The gateway software is a client. It publishes updates on 'whiskerio/gw/gw_mac/pub/devices/dev_mac/chan_id', where:

'gw_mac'	is the gateway MAC address
'dev_mac'	is the device MAC address
'chan_id'	is the channel id

The gateway also subscribes to the topic 'whiskerio/gw/gw_mac/sub'. Other clients can publish to this topic to send API commands to the gateway.

Our mobile application is also a client. It uses the gateway API to request and display a list of connected devices. When a device is selected, the mobile application subscribes to the topics for each channel so that it will receive any updates. When a device reports a new update to the gateway, the gateway publishes to the proper topic which in turn causes the broker to push the message to the mobile app. The mobile app will then update the channel displays to show the current reading and a historical trend.

Our cloud server also acts as a client. It subscribes to all messages from all gateways. When a new message is received, the cloud server checks the database to see if the gateway is configured for cloud data storage. If it is, the message is parsed and the data is stored in the data base.

The cloud server also checks to see if there are any routing rules for a particular gateway, device, and channel. The rules allow the messages to be pushed to other cloud servers and applications.

6.3 Provisioning

As mentioned earlier, provisioning is the process of configuring a gateway and device to operate together. Whisker.IO devices are very easy to provision using our mobile application. Each device is shipped with a serial number label that includes a QR code that can be scanned by any mobile device with a camera. The "Whisker.IO Solution Quick Start Guide" details the procedure.

The process is very simple. The mobile application's main screen shows a list of available gateways. The process for adding a gateway to the list is also detailed in the aforementioned document.

To provision a new device, the user would first select a gateway by clicking or tapping it in the list. This will bring up the device listing for the gateway.

New devices are provisioned from this screen by clicking the '+' icon on the tool bar. This action will bring up a new screen for provisioning a device.

All that is required to complete the provisioning process is to scan the QR code on the device's serial number label and give it a name.

The new device is provisioned once the save icon on the toolbar is clicked or tapped.

The device must be powered on and in provisioning mode during this process, but that too is easy. Every Whisker.IO device stays in provisioning mode for several minutes when power is applied and most devices can also be placed in provisioning mode by swiping a magnet over the serial number label of the device. Other methods for entering provisioning mode may exist for other devices; the particular device manual should be consulted.

6.4 Provision Codes

The Whisker.IO Engine that is the heart of every device is a very powerful and flexible IoT device with more than 100 possible inputs and outputs. There are many configuration parameters related to sample rate, power mode, etc. that can also be set differently from device to device based on its purpose.

We have devised a simple method for identifying the configuration of a device: the ***provision code***. The provision code is a 4 digit number that identifies the unique configuration of a device.

We maintain a list of standard provision codes. New codes are added as new device types are released. One of the functions of our Whisker.IO cloud server is to maintain this list in JSON format. Gateways regularly update their local copy of the list from the server. If a device is provisioned with an unknown provisioning code, the gateway will query the server to determine the correct configuration.

The following is an abbreviated list of provision codes and with a description of the configuration they represent. It is presented for informational purposes only; it is not comprehensive and it is likely to change.

Provision Code	Device Type	Description
0001	Temp/RH SensorBlock	Battery power mode. Temperature, humidity, battery voltage, digital input 0 and digital input 1 are measured every minute and reported to the provisioned gateway.
0005	Air Quality SensorBlock	Battery power mode. CO2 equiv PPM, battery voltage, internal temperature, digital input 0 and digital input 1 are measured every minute and reported to the provisioned gateway
1000	Metering SensorBlock	Battery power mode. Digital input 0 is configured as a counter. Digital counter 0, digital input 1, battery voltage, and internal temperature are reported every minute to the provisioned gateway.
2000	ModBus SensorBlock	Line power mode. The local UART is configured as a Modbus RTU master. Remote commands can be used to read and write ModBus registers.

Provision codes in the 5000 to 9999 range are reserved for custom OEM products. Customers integrating Whisker.IO Engines into their products can request a custom provision code by contact Digital Six Laboratories.

6.5 Devices and Channels

Each Whisker.IO device can be equipped with multiple inputs and outputs. Each input or output is referred to as a channel.

A device can periodically monitor up to four (4) analog and/or digital input channels in total. This represents a physical limitation on the amount of data that can be sent by a device at one time.

However, devices can be equipped with more than four (4) channels. Additional input channels can be read on demand of a command from the gateway or they can be configured to report to the gateway if their state meets some condition. Output channels, of course, do not count in the periodic limitations and a device can have up to 31 digital output and 31 analog output channels, for a total of 62 possible output channels.

Additionally, Whisker.IO devices can operate as Modbus RTU masters. Periodic sampling can be set for up to four (4) Modbus registers (or a combination of registers and internal channels), but remote commands can access the entire range of Modbus tables on multiple Modbus RTU slave devices.

A device is uniquely identified by its MAC address.

A device channel is uniquely identified by its channel ID. A channel ID is composed of two letters and two numbers. The two letters identify the channel type. The two numbers identify the channel number. Together, they identify a unique channel. For example, 'AI1D' is analog input #27, which happens to be the internal temperature sensor built into the Whisker.IO engine.

6.6 MQTT topic mapping

When channel data is sent to the gateway by a device (either periodically or because of a trigger), the gateway will publish the data via MQTT to a topic that uniquely identifies the channel; this means that the topic must include the gateway's MAC address, the device's MAC address, and the channel ID.

whiskerio/gw/DE1A0E88/pub/devices/DE1A0F9F/AI1D

whiskerio/gw – this is the base topic. All gateways publish to this base topic. So, to subscribe to all gateway messages, the subscriber would specify 'whiskerio/gw/#'.

DE1A0E88 – this is the MAC address of the gateway. All topics that start with 'whiskerio/gw/DE1A0E88' are specific to this gateway

pub – there are several types of messages a gateway can send including logging, diagnostic, and publications.

devices – there are actually several types of publications a gateway can make. This segment identifies this topic as a device publication which means it includes an update for a specific channel.

DE1A0E9F – this is the MAC address of the device

AI1D – this is the channel ID. Again, it is the internal temperature sensor of the Whisker.IO engine embedded into the device.

The relationship between the gateway, device, and channel and the topic are unique to Whisker.IO and one of the patent pending features that make Whisker.IO such a flexible IoT solution platform.

The gateway publishes to the following topics:

whiskerio/gw/*gw_mac*/log – if configured, logging messages are published here
whiskerio/gw/*gw_mac*/diag – if configured, diagnostic messages are published here
whiskerio/gw/*gw_mac*/avail – the gateway will announce availability on this topic when it boots
whiskerio/gw/*gw_mac*/pub/resp/*resp_id* –the gateway publishes API responses to this topic
whiskerio/gw/*gw_mac*/pub/devices/*dev_mac*/*chanid* – this has already been explained

The gateway also subscribes to the following topic

whiskerio/gw/*gw_mac*/sub – API requests are published to this topic

All messages for all topics other than diagnostics and logging are encrypted using a key specific to the gateway. For more information, see the Security section.

7 Operation

The gateway is designed to operate as a stand-alone device or as a component of a distributed IoT solution.

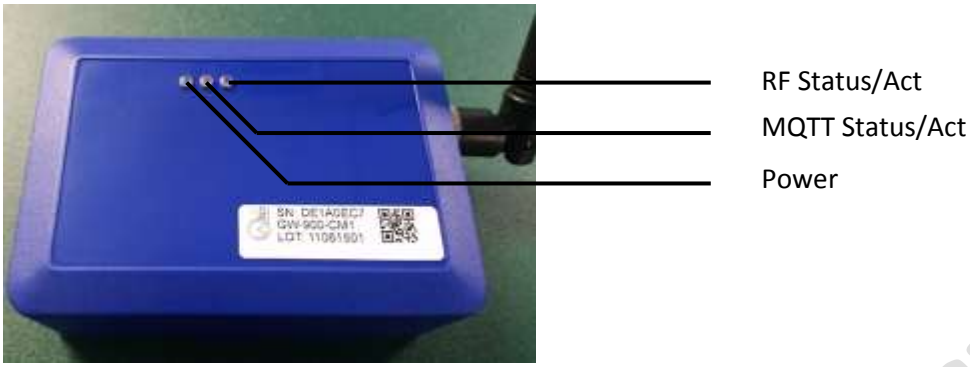
In a basic application, the gateway can operate without the need for any cloud server or application. Using our public MQTT broker, the gateway can communicate directly with our mobile application, allowing users to configure the gateway, provision new devices, configure existing devices, and monitor the values being monitored by the devices, and even send commands to devices to change the state of outputs. No intermediary server is required.

Enterprise applications require a more complicated solution where the gateway communicates with a cloud server for the purposes of recording device data, persisting configuration information, and providing users with analytic tools and rules/function based behavior intelligence. Digital Six Laboratories offers cloud storage and a web app for enterprise applications, but our gateway works with other cloud solutions from vendors such as Amazon, ThingWorx, BugLabs, and more.

The Whisker.IO commercial gateway works equally well in both applications and requires no additional options or cost to support either one.

7.1 Status Lights

The commercial gateway has three (3) status lights on top of the plastic enclosure as shown below:



The Power light will be on anytime power is applied to the gateway. If this light is not lit, then there is no power to the gateway.

The MQTT light indicates that the gateway is connected to the Internet and has been able to establish a persistent connection to our MQTT broker. This light will normally be on when connectivity is established and will blink quickly whenever messages are sent back and forth between the gateway and the broker.

The RF light indicates that the gateway is connected to the Whisker.IO network and is able to transmit and receive to and from Whisker.IO devices. This light will normally be on when the Whisker.IO radio is working properly and will blink quickly whenever messages are sent back and forth between the gateway and Whisker.IO devices.

7.2 Operation with Whisker.IO Mobile Application

The interaction between the gateway and the mobile application was introduced in Provisioning earlier. The functions of the mobile application are:

1. Maintain a list of authorized gateways. The user must have the security key for a gateway to add it to the list.
2. Display a list of devices for a selected gateway. Each device in the list will indicate the signal quality and battery voltage, providing an overall view of the condition of a location.
3. Display a list of channels for a selected device. The channel display will automatically configure itself based on the channel type; digital inputs will show a binary state where analog inputs will show a reading and historical graph.
4. Provide controls for changing state of output channels on selected device. For example, in the case of the MIU SensorBlock, three buttons will be displayed that allow the user to open a valve, close a valve, and partially open a valve.
5. Provide a user interface for changing the configuration of a gateway.
6. Provide a user interface for provisioning new devices.
7. Provide a user interface for changing the configuration of a device.

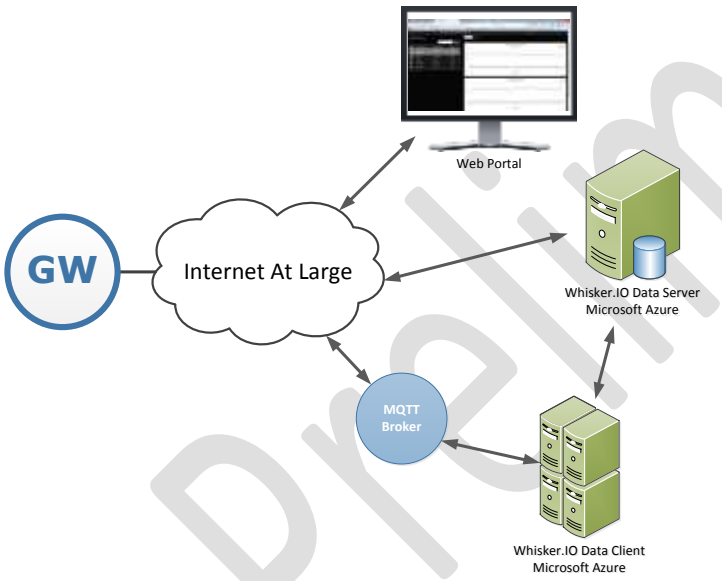
The mobile application does not work through the Whisker.IO Cloud Platform: it communicates directly with the gateway. It is intended as an installation and maintenance tool. For more information, refer to "Whisker.IO Mobile Application User's Manual".

7.3 Operation with Whisker.IO Cloud Platform

Interaction with the Whisker.IO Cloud Platform was introduced in Whisker.IO Solution Overview. The functions of the cloud platform are:

1. Store channel readings from devices in the field for future analysis and reporting.
2. Generate actions based on the values of channel readings. These can include notification, configuration, action, and calling other cloud services through MQTT or HTTP APIs. For example, the cloud platform can be used to initiate a trouble ticket when an pump failure occurs followed by notification to the person or group of people responsible for the maintenance.
3. Provide a user portal where the user can:
    a. Quickly determine the state of their devices and locations and identify any problems
    b. Analyze historical trends using graphing and data tables
    c. Export historical information for use in 3rd party applications
    d. Generate reports such as weekly production, maintenance rates, etc.
    e. Change configuration of cloud portal
    f. Add/remove users and set their privileges
4. Forward channel readings to other MQTT and HTTP based services. For example, a forwarding rule would allow channel readings to be pushed to Amazon's new IoT platform.

The following diagram shows the relationship between the various components of the Whisker.IO Cloud Platform.



Our cloud platform software runs in Microsoft's Azure Cloud Computing Platform. We can replicate and scale our platform at will.

The Data Client's primary job is to communicate directly with deployed gateways. It subscribes to the 'whiskerio/gw/#' topic so that it receives all publications from all Whisker.IO Gateways. As messages come in, the data client queries the database server to determine if the source gateway is configured for cloud operation. If so, the data client parses the topic to determine the source and nature of the message and acts accordingly. For example, if the topic indicates that the message is a channel reading, the data client will generate a new record in the database to store the reading for future use.

For every channel reading message, the data client also checks to see if a forwarding rule is in place. If so, the data client will connect with the specified server using the specified protocol (MQTT or HTTP) and forward the message. This allows the Whisker.IO Cloud Platform to act like a bridge between Whisker.IO gateways and other cloud services and applications.

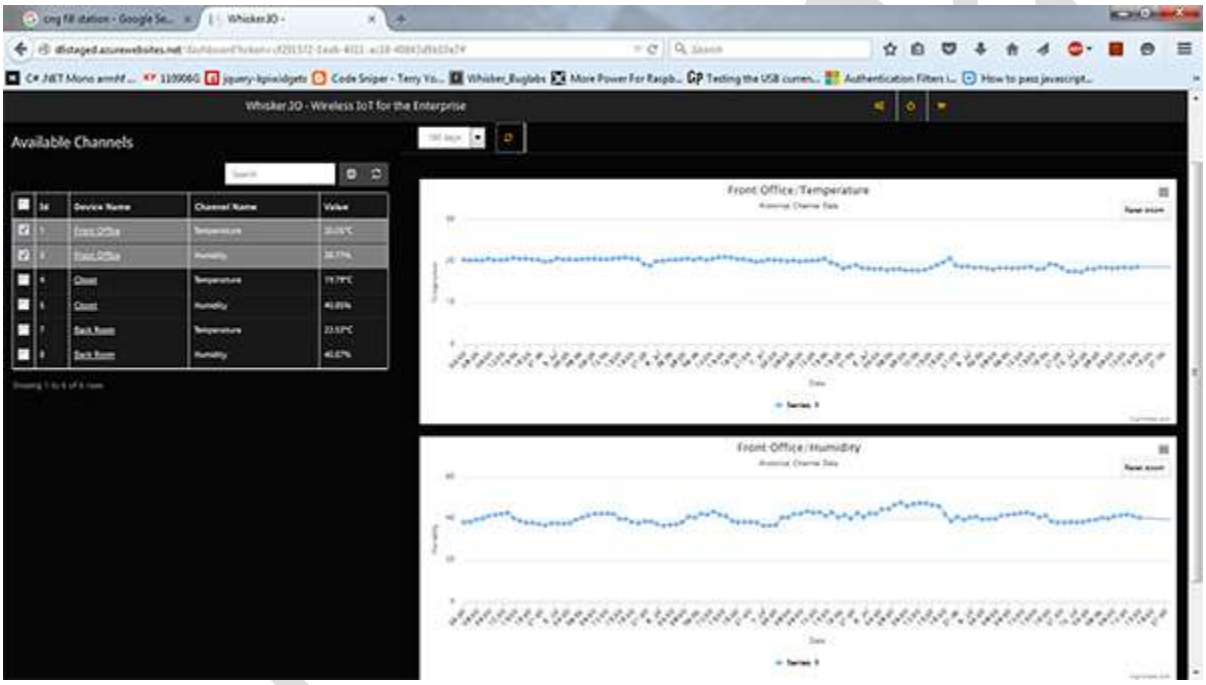
In addition, the data client checks the database to determine if a behavior rule applies to a particular channel or gateway. If so, it executes that rule as well.

Behavior rules can operate in two scopes. Gateway rules operate whenever a message from the specified gateway is received. Channel rules operate whenever a message from the specified channel is received.

A behavior rule is actually a C# script that runs when the message is received.

The Data Server's primary job is to provide a RESTful API that allows Whisker.IO data to be stored and retrieved. It is comprised of a Microsoft SQL database and a MVC Web API running in Microsoft Azure.

The Web Portal provides a web based GUI for accessing, analyzing, and reporting on data stored in the database.



7.4 Operation with Other Cloud Platforms

The gateway can be configured to directly communicate with other cloud platforms. As of this revision, the gateway software supports 2Lemetry (Amazon), BugLabs, and ThingSpeak. The gateway can directly communicate with these platforms; the Data Client is not involved. For more information, see "Configuring a Whisker.IO Gateway for 3rd Party Connectivity".

Other platforms can be supported through our Data Client via routing rules. For more information, see "Whisker.IO Routing Rules App Note".

8 Security

The commercial gateway encrypts every message except logging and diagnostic messages using AES-256 encryption.

A unique encryption key is programmed into each gateway at the factory and provided to the customer on the provisioning card included with the gateway. This key should be kept secret and only provided to those people who need to have access to the gateway directly.

The encryption key can be changed by authorized users, so it is possible to re-secure a gateway whose current encryption key has been compromised.

9 MQTT API

The MQTT API is encrypted, so the encryption key for a gateway must be known in order to make API calls on that gateway.

The API is available on the 'whiskerio/gw/gw_mac/sub' topic, where gw_mac is the MAC address of the target gateway.

An API call is packaged in JSON as follows:

```
{
  "command" : "command",
  "messageGuid" : "mguid",
  "clientGuid" : "cguid",
  "data" :
  {
  …
  }
}
```

Where:

command – is the API command to execute
mguid – is GUID that identifies the message. This changes each message
cguid – is a GUID that identifies the client. This is unique to the client
data – contains additional variables that depend on the command

Valid API commands are:

ChangeConfig	Changes a configuration parameter
ChangeKey	Changes the gateway encryption key. Additional security measures are required for this command.
ChangeName	Changes the gateway name
ChangeLocation	Set's the gateway's lat,long
ReqDeviceState	Requests the state of a particular device

GetDeviceList	Retrieves a list of devices provisioned on a gateway
DeleteDevice	Deletes a device from a gateway
AddDevice	Adds a new device to the gateway
UpdateDevice	Changes device configuration (name, position, etc)
QueueWhiskerCommand	Queues a command for a device. Used to change output state or configuration of device.
Reboot	Remotely reboots a gateway. Additional security measures are imposed on this function.
ResetWhisker	Remotely resets the Whisker.IO Engine that is built into the gateway.

For detailed documentation regarding the MQTT API, please see "Whisker.IO Commercial Gateway MQTT API Documentation".

10 Optimizing Range Performance

The communication range between a gateway and its devices is affected by how and where the gateway is mounted.

If possible, the gateway should not be mounted on a metal surface. If this is required, the antenna should be remotely mounted using an extension cable, which can be purchased separately.

The antenna is articulated so that it can be correctly positioned regardless of the orientation of the gateway. The antenna should be oriented so that it points up.

The height of the gateway is critically important. In general, it should be mounted as high as possible. Actual performance is impossible to accurately predict for any given application because there are many factors that must be accounted for. The following table should be used only for reference purposes. For new deployments, actual range testing should be accomplished on-site to ensure that the performance is adequate.

Height	Estimated Indoor Range	Estimated Outside Range
2 ft	1000 feet	2000 feet
4 ft	2000 feet	4500 feet
10ft	2500 feet	4 miles
30ft	NA	40 miles

11 Firmware Upgrade

Under normal operation, there should be no need for firmware upgrades. However, from time to time, we may need to release upgrades to fix bugs or add new features.

When new firmware is released, there are two possible upgrade paths.

First, if the gateway is so configured and has access to the Internet on ports 21 and 22, it will automatically download new firmware from our server at midnight if new firmware is available. It will then shut down the gateway application, copy the new firmware into place, and restart the gateway application.

The second method is by swapping out SD cards. If the gateway is under a service agreement, we will automatically provide a new SD card for each gateway when new firmware is released. The new SD card will be configured exactly like the deployed gateway, with the same devices and settings. The upgrade process is as simple as removing power, changing the SD card, and restoring power.

12 Required Ports

Several ports must be open for the gateway to operate properly. Others are optional.

Port	Description	Optional or Required?
21	FTP port – for firmware upgrades. Outgoing only.	Optional
22	SFTP port – for firmware upgrades. Outgoing only.	Optional
80	HTTP port – for various REST services	Required
443	HTTPS port – for various REST services	Required
1883	MQTT port	Required

13 External url's

13.1 MQTT

The gateway must connect to at least one MQTT broker in the cloud: the D6 broker. This broker is located at a fixed IP address:

D6 Labs MQTT Broker – **98.172.177.69**

Additional brokers may be configured based on how the gateway is setup to operate. In many cases, the gateway is configured to connect to Amazon AWS. If so configured, the url for that broker is:

AWS Broker - **data.iot.us-east-1.amazonaws.com**

The gateway can connect to other MQTT services such as ThingWorx, Meshify, and Exosite. Please refer to each company's technical documentation for a listing of their broker IP and urls.

13.2 HTTP

The gateway also consumes several REST APIs via port HTTP. The following is a list of those APIs and their corresponding URLs.

API	URL	Description
Twilio	https://api.twilio.com	Used for text message and automatic voice calls
Whisker.IO	https://www.d6labs.com/api	Provides access to WhiskerIO database. Used to update config, etc.

14 Agency Certifications

14.1 United States FCC

The Whisker.IO Engine module complies with Part 15 of the FCC rules and regulations. Compliance with the labeling requirements, FCC notices and antenna usage guidelines is required.

To fulfill FCC Certification, the OEM must comply with the following regulations:

1. The system integrator must ensure that the text on the external label provided with this device is placed on the outside of the final product.
2. Whisker.IO Engine module may only be used with antennas that have been tested and approved for use with this module.

14.1.1 OEM Labeling Requirements

The Original Equipment Manufacturer (OEM) must ensure that FCC labeling requirements are met. This includes a clearly visible label on the outside of the final product enclosure that displays the contents shown in the figure below.

> Contains FCC ID: 2AFTI-WEN9*
>
> The enclosed device complies with part 15 of the FCC rules. Operation is subject to the following 2 conditions: (i). this device may not cause harmful interference and (ii). This device must accept any interference received, including interference that may cause undesired operation.

14.1.2 FCC Notices

Important: The Whisker.IO Engine module has been certified by the FCC for use with other products without any further certification required (as per FCC section 2.1091). Modifications not expressly approved by Digital Six Laboratories could void the user's authority to operate the equipment.

Important: OEMs must test final product to comply with unintentional radiators (FCC section 15.107 and 15.109) before declaring compliance of their final product to Part 15 of the FCC Rules.

Important: The RF module has been certified for remote and base radio applications. If the module will be used for portable applications, the device may require additional SAR testing. Consult Part 15 of FCC rules for more information.

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures: Re-orient or relocate the receiving antenna, Increase the separation between the equipment and receiver, Connect equipment and receiver to outlets on different circuits, or Consult the dealer or an experienced radio/TV technician for help.

14.1.3 RF Exposure

CAUTION: To satisfy FCC RF exposure requirements for mobile transmitting devices, a separation distance of 20 cm or more should be maintained between the antenna of this device and persons during device operation. To ensure compliance, operations at closer than this distance are not recommended. The antenna used for this transmitter must not be co-located in conjunction with any other antenna or transmitter.

The preceding statement must be included as a CAUTION statement in OEM product manuals in order to alert users of FCC RF Exposure compliance.

14.2 Additional Countries

Additional certifications can be provided on request. Please contact sales@d6labs.com.