



Gateway Firmware Design

1-July-2015



19201 Parthenia Street, Suite J
Northridge, CA 91234

P: 818.701.9831

F: 818.701.1506



pcssales@pcslighting.com
www.pcslighting.com

Introduction

This document supersedes the Firmware Requirements document. This document is being written as the design and implementation progresses. As of this revision, it is incomplete because the project is still in development. The reader should still read the Requirements document for a foundation, but some things were changed since that document was written. This document includes those changes.

Hardware

The design uses the Digi Connect ME –C. One GPIO pin will be wired to a push button for selecting one of three functions, toggle the device between a known static IP address and DHCP, default the users table to no users, and default all settings to factory values. Additional GPIO will be needed to interface to the RTC when it is added later. Some of these hardware features have been implemented, but are not available yet.

Upon startup, the PIM will be placed in Message Mode in case it was in Pulse Mode.

Discovery of the Gateway on the Ethernet Network

When the device's IP address is not known, UPStart or a client app may broadcast a UDP "query" message. The string "PIM-IP QUERY" must be broadcast to address 255.255.255.255, port 2362. The Gateway will reply by broadcasting a UDP message to the same address and port as follows: "PCS PIM-IP\0" followed by the six-byte MAC address, four-byte IP address, two-byte TCP port, and two-byte version number. All values are little-endian to the right. For example:

```
PCS PIM-IP 0x00 0x00 0x40 0x9D 0x74 0xE4 0x8D 0xC0 0xA8 0x00 0x7F
0x08 0x35 0x01 0x00
```

Where:

- PCS PIM-IP = 10 ASCII characters
- 0x00 = null delimiter
- 0x00 0x40 0x9D 0x74 0xE4 0x8D = MAC address 00:40:9D:74:E4:8D
- 0xC0 0xA8 0x00 0x7F = IP address 192.168.0.127
- 0x08 0x35 = port 2101
- 0x01 0x00 = firmware version 1.0

This design allows the Gateway to receive the message no matter its IP address and subnet mask. It can reply back without regard for the requestor's address or subnet mask. Most routers and switches will not forward the broadcast through from one LAN segment to another. So the device and client will likely need to be on the same physical LAN segment for this to work. This scheme is used widely by others in the industry with the same limitations.

Network Connectivity and Authentication

Clients may connect using TCP to the address and port specified in the device's Discovery reply, described above. All connection handshaking messages are ASCII strings and null-terminated. As soon as a client connects, it must send a message identifying itself. This is the "Client Hello" message and is

defined as follows:

```
ClientName/FWVer/ProtocolVers\0
```

```
For Example: "UPSTART/7.0.16/1:2:3"\0
```

ClientName is any string with a reasonable length. It is used to differentiate one client app from another. As of the date of this document, the Gateway ignores this field. In the future, it may be desired to know when a particular app is connecting. This protocol allows for that. Following ClientName is a slash and the client's software version, separated by another slash, and the protocol versions it supports. Each protocol version (if more than one) is separated by a colon. The Gateway TCP protocol will be versioned to allow for future enhancements. The protocol may or may not change regardless of other firmware version changes. If the protocol changes, it will be assigned a new protocol version number. Some reasons to do this are: adding new commands, changing the format of messages, and adding/improving features. The Gateway will select the highest protocol version it supports from the list provided and reply with that version. If the Gateway supports none of the protocols offered, it will reply with protocol version set to zero. The connection will then be terminated. As of the date of this document, only protocol version 1 exists.

If the Gateway is not done with startup initialization, it will reply with "PIM NOT INITIALIZED"\0 and disconnect.

If the message is incomplete or the Gateway is unable to parse it for any reason, it will reply with "INCOMPLETE MESSAGE"\0 and disconnect. Otherwise, it will reply with the "Server Hello" message as follows:

```
PCS PIM-IP2/{FWver}/{ProtocolVer}/{SomeOtherStuff}\0
```

```
For example: PCS PIM-IP2/1.0/1/ . . . \0
```

In this example, the device's firmware version is 1.0, and it supports protocol version 1.

The "SomeOtherStuff" field indicates whether authentication is required or not. If it is, it will include a 64-byte MD-5 challenge, encoded in ASCII. What determines whether authentication is needed is the presence of at least one username/password pair in the users table (more on it later), programmed into the Gateway by UpStart. If there are no username/password pairs, or if the table is empty, we assume authentication is not performed. If authentication is not required, the "SomeOtherStuff" field will be set to "AUTH NOT NEEDED/x CLIENTS", where x is 1-8 and is the number of clients already connected. If authentication is needed, it will be set to "AUTH REQUIRED/" and followed by the 128 characters representing the 64 byte challenge.

```
For example: PCS PIM-IP2/1.0/1/AUTH NOT NEEDED/2 CLIENTS\0
```

This example shows authentication is not needed and two clients are already connected. This connection is the third. At this point, the client is connected and may send/receive data according to

the protocol rules. Another example:

```
PCS PIM-IP2/1.0/1/AUTH REQUIRED/30E2FED3 . . . 783EA827\0
```

This example shows authentication is needed. “30E2FED3 . . . 783EA827” is 128 chars long. The client will convert the ASCII challenge to binary (64 bytes) and compute an MD-5 hash using the password. Details for doing so are below. It will return the MD-5 response as shown in this example:

```
username/46B3D1F8 . . . D93C4505
```

Where “username” is the username, obviously, and the characters following the slash are 32 ASCII characters encoding the 16-byte MD-5 response.

If the response does not match what the Gateway expected, the device will issue an “AUTHENTICATION FAILED\0” message and disconnect. If it succeeds, the device will issue an “AUTH SUCCEEDED/x CLIENTS\0” message, where x is the number of clients already connected.

The Gateway allows up to eight usable TCP connections simultaneously. A ninth connection is allowed just long enough to tell the client the limit of eight has been reached. The message is “MAX CONNECTIONS REACHED\0”. That connection is then terminated.

If a client is already connected and has put the Gateway in Pulse Mode, no new connections will be allowed. Other clients attempting to connect will receive the message “PULSE MODE ACTIVE\0.”

If a connection is dropped, the device will detect it, and close that socket. While a client is in Pulse Mode, the Gateway will maintain an idle timer for it. The time out value is set by the client upon issuing the “Go to Pulse Mode” command. Each message from the Pulse Mode client will reset that timer. If the timer is allowed to expire, the Gateway will send a Disconnect message to the client, terminate the connection, and exit Pulse Mode to Message Mode.

Security/CRAM

Connections may be authenticated using a Challenge Response Authentication Mechanism based on MD-5 hashes (CRAM-MD5). As described above, the Gateway will indicate in its “Server Hello” message whether authentication is required or not. The client’s CRAM algorithm is as follows:

(Note: Most MD-5 implementations provide for a way to create an MD-5 context and a means to perform block transforms or updates in a chain until the final transform or update. In this algorithm, we will perform an initial transform/update and a final transform/update on two separate MD-5 contexts).

1. Convert the 128-character ASCII challenge to a 64-byte array.
2. Create two 64-byte arrays (“ipad” and “opad”) and initialize all elements to zero.
3. Copy the password as an ASCII char array into both arrays, left-justified.
4. XOR every byte in ipad with 0x36
5. XOR every byte in opad with 0x5C

6. Create an MD-5 Digest object. Call it D1.
7. Transform/update D1 with the ipad array.*
8. Do it again to finalize it with the challenge array.*
9. The result is a 16-byte array we'll call D1Output.
10. Create another MD-5 Digest object. Call it D2.
11. Transform/update D2 with the opad array.*
12. Do it again to finalize it with D1Output.*
13. The result is another 16-byte array we'll call D2Output.
14. Encode D2Output as 32 ASCII characters.
15. Copy the username to the transmit buffer. Add a forward slash after it. Then copy the encoded D2Output following the slash.

*In some implementations, it may be possible to perform both block transforms in one step, rather than two. In that case, just call the finalize function.

The response will look like this (for example):

```
username/82B195F3A8CD9E4097F4A97B65E6D44C
```

The Gateway will compute the same transform and compare what it receives from the client against that.

At this point, the client enters "Command Mode" where all further data is encapsulated in packets.

Command Mode

All commands are binary messages, both directions. In general, a message from a client to the Gateway will consist of a command, followed by data length, the data, and a checksum. Commands will be one byte, data length is a two-byte wide count of the number of data bytes to follow. The data itself may be any length and contain any values. The checksum will be one byte at the end of the data. It is a simple 1's compliment of the eight bit sum of all characters or bytes in the message from the first (command byte) to the last data byte, inclusive.

Command	DataLen	x x x . . . x	Checksum
(0-FF)	(0-FFFF)	(up to 65535 bytes)	(0-FF)

A Reply is required for all Commands (except the Disconnect Command, but it is preferred there). Replies will indicate success or failure. A reply of success may or may not include data. There are two replies that indicate failure. One is a NAK, the other is an error message. The first byte of any reply that is not a NAK is the original received command, plus one. This is followed by a two byte data length, a single byte success or error code, optional data, and a checksum. The checksum is calculated similarly as for commands, above. The two byte data length includes the success or error code and all optional data. The success or error code field is zero for success or any other number to indicate a specific error.

A NAK is sent when the message is not understood, such as an incomplete message or a bad checksum.

The reason NAKs are separate from other error replies is that the command received may be invalid. Therefore the Gateway cannot reply with the command +1. NAK replies will include an error code indicating the problem.

In general, a reply looks like the following:

Command	DataLen	Success/ErrorCode	x x x . . . x	Checksum
(CMD+1)	(0-FFFF)	(0-FF)	(up to 65535 bytes)	(0-FF)

For successful replies with no data:

(CMD+1)	(0001)	(00)	(Cksum)
---------	--------	------	---------

For successful replies with data:

(CMD+1)	(xxxx)	(00)	(x x x ... x)	(Cksum)
---------	--------	------	---------------	---------

For error replies:

(CMD+1)	(0001)	(Error Code)	(Cksum)
---------	--------	--------------	---------

NAK replies:

(FF)	(0001)	(NAK Reason)	(Cksum)
------	--------	--------------	---------

Upon receiving a NAK or other error code, the original sender may repeat the message a limited number of times.

Commands

“DL” is the two-byte data length, “X” is a single byte or array of bytes or characters, and “CS” is the Checksum

CMD 0x10 Keep-alive

- (Pulse Mode Client Only)
- 0x10 0x0000 CS
- Replies:
 - Success: 0x11 0x0001 00 CS
 - NAK

CMD 0x20 Set Date and Time

- 0x20 0x000A Y M D h m s W T TZ CS
 - Y = year (14-99)
 - M = month (1-12)
 - D = day of month (1-31)
 - h = hour in 24-hour time (0-23)
 - m = minute (0-59)
 - s = seconds (0-59)

- W = weekday (1-7)
- T = DST currently active (0=no, 1=yes)
- TZ = signed 16-bit offset from GMT in minutes (high byte first)
- Replies:
 - Success: 0x21 0x0001 0x00 CS
 - Invalid parameter(s): 0x21 0x0001 0x21 CS
 - NAK

CMD 0x22 Get Date and Time

- 0x22 0x0000 CS
 - Replies:
 - Success: 0x23 0x000B 0x00 Y M D h m s W T TZ CS
 - Y = year (14-99)
 - M = month (1-12)
 - D = day of month (1-31)
 - h = hour in 24-hour time (0-23)
 - m = minute (0-59)
 - s = seconds (0-59)
 - W = weekday (1-7)
 - T = DST currently active (0=no, 1=yes)
 - TZ = signed 16-bit offset from GMT in minutes (high byte first)
 - NAK

CMD 0x28 Set Current Schedule

- 0x28 0x0001 X CS
 - X = 0-3
 - Replies:
 - Success: 0x29 0x0001 0x00 CS
 - Invalid parameter(s): 0x29 0x0001 0x21 CS
 - NAK

CMD 0x2A Get Current Schedule

- 0x2A 0x0000 CS
- Replies:
 - Success: 0x2B 0x0002 0x00 X CS
 - X = 0-3
 - NAK

CMD 0x30 Send UPB Network message through PIM

- 0x30 DL X X X . . . X CS
- Replies:
 - Success: 0x31 0x0001 0x00 CS
 - NAK

CMD 0x50 Open file for writing

- If the file already exists, it will be erased when this command executes. The data is a filename string following DOS naming rules, max 8-char name + "." + max 3-char extension. The reply returns a 4-byte file handle that is to be used for subsequent operations on the file. Up to four files may be open simultaneously (for reading or writing) by all connected clients. That's four total, not four each.
- 0x50 DL filename.ext CS
- Replies:
 - Success: 0x51 0x0005 0x00 X X X X CS
 - X = file handle
 - Error: 0x51 0x0001 EC CS
 - EC = (see list in "Protocol Summary" below)
 - NAK

CMD 0x52 Write to file

- This command may store up to 1024 bytes at a time. If the file is short enough, only one Append command needs to be used. But if a file is large, multiple Appends are needed. Because this adds to the end of the file, multiples of this command may be used in sequence until all the data is written. No "seek" or file pointer positioning is provided.
- 0x52 DL handle X X X . . . X CS
 - handle is the 4-byte handle returned by the Open command. X is the data to be stored, up to 1024 bytes. DL includes the number of bytes to store plus four for the handle.
- Replies:
 - Success: 0x53 0x0001 0x00 CS
 - Error: 0x53 0x0001 EC CS
 - EC = (see list in "Protocol Summary" below)
 - NAK

CMD 0x54 Get size of file opened for writing

- 0x54 0x0004 handle CS
 - handle is the 4-byte handle returned by the Open command.
- Replies
 - Success: 0x55 0x0005 0x00 filesize CS
 - filesize is a 32-bit integer, little endian.
 - Error: 0x55 0x0001 EC CS
 - EC = (see list in "Protocol Summary" below)
 - NAK

CMD 0x56 Close file opened for writing

- Separate close commands are needed for files read from/written to because the internal clean-up following the close differs for each.
- 0x56 0x0004 handle CS
- Replies:

- Success: 0x57 0x0001 0x00 CS
- Error: 0x57 0x0001 EC CS
 - EC = (see list in “Protocol Summary” below)
- NAK

CMD 0x60 Open file for reading

- The data is a filename string following DOS naming rules, max 8-char name + “.” + max 3-char extension. The reply returns a 4-byte file handle that is to be used for subsequent operations on the file. Up to four files may be open simultaneously (for reading or writing) by all connected clients. That’s four total, not four each.
- 0x60 DL filename.ext CS
- Replies:
 - Success: 0x61 0x0005 0x00 X X X X CS
 - X = file handle
 - Error: 0x61 0x0001 EC CS
 - EC = (see list in “Protocol Summary” below)
 - NAK

CMD 0x62 Read from file

- This command may read up to 1024 bytes at a time or up to the end of file, whichever is shorter. If the file is short enough, only one Read command needs to be used. But if a file is large, multiple Reads are needed. The Gateway will maintain a read position pointer so there is no need for the client to keep up with that. The pointer is moved with each read, so subsequent reads will continue where the last left off.
- 0x62 0x0004 handle CS
 - handle is the 4-byte handle returned by the Open command.
- Replies:
 - Success: 0x63 DL 0x00 X X X . . . X CS
 - DL is the number of bytes read (+1 for 0 byte)
 - Error: 0x63 0x0001 EC CS
 - EC = (see list in “Protocol Summary” below)
 - NAK

CMD 0x64 Get size of file opened for reading

- 0x64 0x0004 handle CS
 - handle is the 4-byte handle returned by the Open command.
- Replies
 - Success: 0x65 0x0005 0x00 filesize CS
 - filesize is a 32-bit integer, little endian.
 - Error: 0x65 0x0001 EC CS
 - EC = (see list in “Protocol Summary” below)
 - NAK

CMD 0x66 Close file opened for reading

- Separate close commands are needed for files read from/written to because the internal clean-up following the close differs for both.
- 0x66 0x0004 handle CS
- Replies:
 - Success: 0x67 0x0001 0x00 CS
 - Error: 0x57 0x0001 EC CS
 - EC = (see list in "Protocol Summary" below)
 - NAK

CMD 0x70 Delete file

- The data is a filename string following DOS naming rules, max 8-char name + "." + max 3-char extension.
- 0x70 DL filename.ext CS
- Replies:
 - Success: 0x71 0x0001 0x00 CS
 - Error: 0x71 0x0001 EC CS
 - EC = (see list in "Protocol Summary" below)
 - NAK

CMD 0x80 Get File Listing

- 0x80 0x0000 CS
- Replies:
 - Success: 0x81 DL 0x00 NumFiles FileNames CS
 - NumFiles is a one-byte value of the number of file names returned.
 - FileNames is a variable length block containing the ASCII file names. Each name is null-terminated.
 - DL includes the number of bytes in FileNames plus one for the NumFiles byte and plus one for the success byte.
 - Example: 0x81 0x0016 0x00 0x02 "File1.txt\0File2.txt\0" CS
 - Error: 0x81 0x0001 EC CS
 - EC = (see list in "Protocol Summary" below)
 - NAK

CMD 0x90 Start Pulse Mode

- When this command is received, the Gateway will send a message to all other connected clients and then disconnect from them. If the other clients are in the middle of connection handshaking, they will receive the message, "PULSE MODE ACTIVE\0." If they have completed handshaking, they will receive a command F2 (see below). The Gateway will also refuse new connections using the "PULSE MODE ACTIVE\0" message. The data byte is the number of seconds to wait before disconnecting this client should they not send further commands during that time. The range is 0, 20-255. 0 will never disconnect (recommended only for use during development). Any value from 1-19 will default to 20.

- 0x90 0x0001 TimeoutByte CS
- Replies:
 - Success: 0x91 0x0000 0x00 CS
 - NAK

CMD 0x92 Exit Pulse Mode

- When Pulse Mode is exited, the Gateway will allow new connections again.
- 0x92 0x0000 CS
- Replies:
 - Success: 0x93 0x0000 0x00 CS
 - NAK

CMD 0xF0 Disconnect

- 0xF0 0x0000 CS
- Replies:
 - Success: 0xF1 0x0001 0x00 CS
 - NAK
 - The original sender can ignore a NAK and disconnect anyway, because the original sender may be disconnecting due to too many errors to begin with.

Unsolicited messages from Gateway to clients

- MSG 0xE0 - UPB Network message from PIM
 - 0xE0 DL X X X . . . X CS
- MSG 0xE2 – Device state update
 - 0xE2 DL X X X . . . X CS
 - This message always contains 10 bytes. The first byte is the UPB module id of the device being affected. The next 9 bytes are the levels of each “channel” of the device. This is documented in the Gateway Dev-Kit document
- MSG 0xF0 – Disconnect, idle timer expired (to PulseMode clients only)
 - 0xF0 0x0000 CS
- MSG 0xF2 – Disconnect, Pulse Mode just started
 - 0xF2 0x0000 CS

Tables

UpStart and other clients may store or read tables (files) in the Gateway. Some of the tables have specific names and purposes. An application is free to store its own tables in the Gateway file system as long as the filenames don't conflict with these pre-defined ones.

The pre-defined tables have their access checked by the firmware and an error returns if the access is denied. The following list defines those tables, their purposes, and access restrictions.

<u>File name</u>	<u>Purpose</u>
users.dat	Username/passwords/permissions
network.dat	Ethernet configuration
export.upe	UPStart export file
Inkact.dat	Link Activate rules
Inkdact.dat	Link Deactivate rules
devtype.dat	Device Type definitions
devstate.dat	Device States
Inkstate.dat	Link States
schedule.dat	Schedules
suntime.dat	Sunrise and Sunset for the install location
dst.dat	Daylight Saving time start and end info
location.dat	Location of the installation
calendar.dat	Scheduling calendar

The user table contains from one to four users – name and password – and their permissions. The following pseudo-code shows how the firmware handles access to these tables.

```

BOOL CanUserReadThisTable (string tableName)
{
    if tableName is users.dat, network.dat
    {
        if user has user-table-read-write permission
            return (TRUE);
        else
            return (FALSE);
    }

    // All clients can always can read any other tables
    return (TRUE);
}

BOOL CanUserWriteThisTable (string tableName)
{
    if tableName is users.dat, network.dat
        if user has user-table-read-write permission
            return (TRUE);
        else
            return (FALSE);

    if tablename is export.upe, devastate.dat, lnkstate.dat, lnkact.dat, lnkdeact.dat,
        devtype.dat

        if user has table-write-permission
            Return (TRUE);

    if tablename is schedule.dat, suntime.dat, dst.dat, location.dat, calendar.dat

        if user has table-write-permission or schedule-table-write permission
            return (TRUE);
        else
            return (FALSE);

    // Other tables not listed can always be written.
    return (TRUE);
}

```

The idea is to allow only the most sophisticated users to update the configuration that allows other users access to the Gateway (the users.dat and network.dat tables).

Other users, slightly less sophisticated, can update the network definition (export.upe and other tables).

Lower permission level users can't change the UPB network definition but can update the schedule.

Finally, users with no permissions can still connect, view the UPB network, and control devices but not make any changes to the configuration.

Firmware Updates

To manually update the firmware:

1. Make sure the firmware file is named "image.bin" for application files and "rom.bin" for bootloader files.

2. Start a DOS command prompt.
3. "CD" to the folder where the new image is located.
4. Enter: ftp {device IP address}
5. When prompted for username, enter: upstart
6. When prompted for password, enter: firmware
7. After successfully logging in, enter: put image.bin or put rom.bin
8. If rom.bin was "put," wait for it to complete, then put image.bin also.
9. Wait for it to complete then enter: quit.

Note that the image will not be burned to flash until the FTP session is ended with "quit." Watch the yellow and green lights on the Connect ME. Both will be on solid for a few seconds, then the green one will turn off. After a few more seconds, the yellow LED will blink off, then back on, and the green LED will soon begin flashing for network activity as it normally does. At this point the re-flashing is complete.

Bootloader

The bootloader implements the procedure for decoding the push button "taps" to default various settings of the Gateway. As of the current version of this document, this has been implemented, but due to a lack of support in the current hardware, not fully tested. Therefore this feature has been disabled for now.

Press and hold the button in during power up until the green LED on the RJ-45 jack begins flashing rapidly. Release the button. At this point, the Gateway is running in "Bootloader" mode. The Application firmware has not been loaded. In this mode, the FTP server is operational and a new application firmware image may be flashed. From this mode, the button may be "tapped" in a special sequence to default certain things.

Tapping 5-10-2 will cause all factory values to be reloaded. The device will default to obtaining an IP address via DHCP, with TCP Port 2101 used for connections. All tables will be deleted or restored to factory values (empty). Following this, the green LED will repeatedly blink four times to confirm success.

Tapping 5-5-2 will erase the users table only, effectively resetting passwords to blank. This will allow anyone to connect until the table is restored. Following this, the green LED will repeatedly blink three times to confirm success.

Tapping 5-4-2 will toggle the Ethernet configuration between DHCP and a static IP address of 192.168.1.120. If the Gateway is currently using DHCP, it will be set to the static address, and repeatedly blink the green LED two times. If it is currently using any static address, it will be set to DHCP, and repeatedly blink the green LED one time.

After executing any sequence above, any other sequence may then be executed. At any point, the firmware may be re-flashed using FTP also.

For the changes effected by tapping to take effect, the Gateway must be power-cycled.

Troubleshooting

The Gateway implements a feature to assist in debugging/troubleshooting. It may be useful to developers writing apps to connect to it. It may also be useful to technical support personnel trying to troubleshoot a customer's installation problems. Basically, this feature allows the Gateway to send ASCII messages indicating what internal software functions are being executed and error information when a problem is encountered. It does this using a UDP connection. Any application may send a "dummy" byte to the Gateway's IP address on port 12345 using UDP. The Gateway will send back one or more ASCII strings, terminated by "newline" characters. The dummy messages to the Gateway may be sent every two seconds or longer. The Gateway will queue up messages until it receives the dummy byte. If it has no messages to send, it will not reply at all.

Protocol Summary

This is a brief summary of the Ethernet network communication protocols. For details, see above.

Discovery Protocol

- Client broadcasts PIM-IP QUERY to IP address 255.255.255.255, port 2362.
- Client waits 3-4 seconds for any replies, then may repeat the process, ignoring duplicates.
- Client may list the results obtained for the user to select the correct device.
- Gateway replies to the client's broadcast with PCS PIM-IP\0{6-byte MAC address}{4-byte IP Address}{2-byte TCP port}{2-byte firmware version} Only the ID string is in ASCII. All else is binary, little-endian to the right.

ClientHello/ServerHello

- Client sends {ClientName}/{FWVer}/{ProtocolVers}\0 all in ASCII.
- Gateway responds with PCS PIM-IP2/{FWver}/{ProtocolVer}/ . . . \0 all in ASCII.
- The portion following {ProtocolVer}/ may be either:
 - AUTH NOT NEEDED/x CLIENTS
 - AUTH REQUIRED/{challenge}
- If the response is AUTH NOT NEEDED, the client is connected and switches to the Command Mode protocol.
- If the response is AUTH REQUIRED, the client replies with {username}/{challenge response}\0
- If the client's response is correct, the Gateway will send:
 - AUTH SUCCEEDED/x CLIENTS and the client is connected and switches to the Command Mode protocol.
- If it is not correct, the Gateway will send:
 - AUTHENTICATION FAILED and disconnect.

- Other messages from Gateway to client:
 - PIM NOT INITIALIZED - PIM has not completed startup initialization.
 - INCOMPLETE MESSAGE – ClientHello was not understood
 - MAX CONNECTIONS REACHED – Eight connections already exist.
 - PULSE MODE ACTIVE – Sent when the PIM is in Pulse Mode.
 - PCS PIM-IP2/{FWver}/0/ . . . \0 – The Gateway does not support any protocol offered.

Packetized Protocol

(Version 1)

Client to Gateway:

{CMD} {DataLen} {Data} {Checksum}

Gateway Replies:

{CMD+1} 0x0001 0x00 {Checksum}

{CMD+1} {DataLen(+1)} 0x00 {Data} {Checksum}

{CMD+1} 0x0001 {ErrorCode} {Checksum}

0xFF 0x0001 {NAK Reason} {Checksum}

Unsolicited Gateway Messages:

0xE0 {DataLen} {PIM Message} {Checksum}

0xF0 0x0000 {Checksum}

0xF2 0x0000 {Checksum}

NAK Reasons

0x01 BAD CHECKSUM

0x02 INCOMPLETE MESSAGE

0x03 UNKNOWN COMMAND

Error Codes

(File I/O errors)

0x10 TOO MANY OPEN FILES

0x11 FILE DOES NOT EXIST

0x12 OPEN FAILED

0x13 INVALID FILE HANDLE

0x14 WRITE FAILED

0x15 END OF FILE

0x16 SEEK FAILED

0x17 CLOSE FAILED

0x18 INTERNAL ERROR – IO_REQ_CB

0x19 INTERNAL ERROR – DIR_COUNT

0x1A INTERNAL ERROR – REQ_CB_STAT

0x1B INTERNAL ERROR – LIST_DIR

0x1C INVALID FILE NAME
0x1D USER NOT AUTHORIZED

(Miscellaneous errors)

0x20 OUT OF MEMORY
0x21 INVALID PARAMETER(S)