HCA Tech Note 304: Working with Z-Wave Locks

What it offers

Support for Z-Wave was made available in HCA starting with version 12. However, this support is not comprehensive, and is intended to augment, rather than replace, HCA's support for UPB, Insteon, and X-10. There are certain types of Z-wave devices that are not available in UPB, Insteon, or X-10 versions; for example, door locks. Z-Wave door locks are currently available from a number of companies, including Yale, Kwikset, Baldwin, and Schlage. Z-Wave is well suited to door locks because it is a wireless control technology, and can operate from battery power.

What is Z-Wave

Z-Wave is a proprietary wireless mesh networking technology originally developed by the Danish company Zensys, now part of Sigma Designs. Because Sigma Designs considers documentation about Z-Wave technology proprietary, and make such documentation only available under license, it can be a bit challenging to figure out how it works and how to control Z-Wave devices. Fortunately, there is a good deal of information on the Internet, and this document will give examples of commands for controlling Yale locks, and provides a reference to a document by Leviton that shows how to control Kwikset locks.

Mesh networks, in order to cover a distance greater than the range of a single device's transmitter, are designed to relay commands over multiple nodes of the "mesh". If you have a small house, you may only need the Leviton Z-Wave interface device and the target device, e.g. a door lock. However, for greater reliability, and to span greater distances, you may want to add some Z-Wave modules as relay nodes for the mesh. Leviton LEVVRP03 Z-wave appliance modules work well for this. These appliance modules can be plugged into out of the way outlets at strategic locations to improve network reliability, if necessary.

Just as a UPB network is initially configured using a tool external to HCA (UPStart), your Z-Wave network will be configured using an external tool. In the case of the Leviton Vizia VRC0P serial-to-Z-Wave interface that HCA supports, a Leviton software tool is used to configure the Z-wave network (Vizia RF+ Installer Tool). This tool is a free download from: www.leviton.com/rfinstaller, but requires a Leviton LEVVRUSB-1US USB stick for configuration. This USB stick will be the "primary" controller in your network. The Leviton VRC0P will function as a "secondary" controller, in Z-Wave parlance, even though it will be the way HCA controls your Z-Wave devices. Note that the VRC0P you use must have the latest firmware in it in order to support locks, as locks require the use of secure

(encrypted) Z-Wave commands, and only the newest firmware supports this ability. Your VRC0P must also be a "+3", and this version is labeled "Vizia RF+ 3".

Also note that another critical feature of both the Vizia RF +3 VRC0P and Leviton LEVVRP03 appliance modules previously mentioned is that they support "beaming". Because locks are battery powered, they need to conserve power, and only "wake up" periodically to listen for a command targeted at them. Devices that support beaming send out a short beacon that the locks listen for, and when they "hear" this beacon, they keep their receiver powered on to receive the command that might be addressed to them. There are only a few devices in the market that support beaming, and if you have Z-wave devices in your network that do not support beaming, they will not be able to relay commands for locks across the mesh network, hence the recommendation above to use Leviton LEVVRP03 appliance modules as relay nodes. (Other Leviton Z-Wave devices that support beaming are: LEVVRCZ4-M0 Vizia RF+ 4 button zone controllers and LEVVRCS4-M0 Vizia RF+ 4 button scene controllers.)

Instructions for creating a Z-Wave network using the Leviton USB stick and installation software, along with locks and the VRC0P serial interface, can be found at:

http://www2.worthingtondistribution.com/downloads/HAI_Z-Wave_Lock_Integration_1-4-2012.pdf

While these instructions were written for use with another automation system, they provide the information you will need to set up you network and locks. Once set up, your VRC0P interface will be plugged into the computer running HCA, rather than to the automation system referenced in the document. Note that when you do the final network configuration with the Leviton tool, all Z-Wave devices need to be in their final location. Moving Z-Wave devices affects the network topology, and can adversely affect network routing if the network is not re-optimized.

NOTE: Pay particular attention to pages 16 and 17 of the Worthington document: "Add the Leviton Serial Interface" and "Set the Serial Interface For 2-Way". It is important that the Leviton serial interface be added last. If you don't add it last, you will need to repeat the steps on pages 16-17 again. Also, note that if at a later time you add another Z-wave device, such as another lock, you must repeat the steps on pages 16-17 after adding the new devices. If you don't do this, you won't be able to communicate with them using the VRC0P serial interface.

A word about security

With the ability of home automation systems, such as HCA, to be connected to the Internet, security is very important. When you connect devices like door locks to your automation system, security becomes critical. You don't want hackers

unlocking your doors via the internet! A strong remote access password is a necessity, and the use of VPNs for remote access if possible, is recommended.

Z-Wave Command Classes

Z-Wave devices support "Command Classes". These command classes are related to a type of function, such as configuring devices, setting a schedule on a device, etc. A list of command classes by name and number can be found at:

http://wiki.micasaverde.com/index.php/ZWave_Command_Classes

Within a command class, there are numeric commands. An example of commands includes Get and Set commands, to read and write data from a Z-wave device. Read data comes back as a Report. Z-Wave commands sent to the Leviton interface are structured as: command_class, command, and parameters 1-N, specific to the command class and command. (More about this below). Note that the actual numeric value for a command will vary from command class to command class, i.e. the Set command for one command class may be "4", while for another command class it might be "12".

Leviton provides two very useful documents related to the VRC0P interface. One Application Note describes the interface and its commands:

http://www.leviton.com/OA_HTML/ibcGetAttachment.jsp?cItemId=36xtvRg6OgYdYTNKoRtIHw&label=IBE&appName=IBE&minisite=10251

The other gives examples of how to control a Kwikset Lock:

http://s7d5.scene7.com/is/content/BDHHI/ApplicationNote-UsingASCII-Z-Wave-Locks

Locks, and the command classes they support

From manufacturer's brochures and other promotional material, you may be able to determine the command classes that a Z-Wave device supports. For example, from Assa Abloy's documentation, you can find that their Yale Z-Wave door locks support the following command classes:

COMMAND_CLASS_MANUFACTURER_SPECIFIC
COMMAND_CLASS_VERSION
COMMAND_CLASS_SECURITY
COMMAND_CLASS_BATTERY
COMMAND_CLASS_ASSOCIATION
COMMAND_CLASS_CONFIGURATION

COMMAND_CLASS_PROTECTION (V2)
COMMAND_CLASS_DOOR_LOCK
COMMAND_CLASS_USER_CODE
COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (V3)
COMMAND_CLASS_DOOR_LOCK_LOGGING
COMMAND_CLASS_ALARM
COMMAND_CLASS_TIME_PARAMETERS
COMMAND_CLASS_BASIC
COMMAND_CLASS_TIME

From Kwikset's lock brochures, you can find that they support:

COMMAND_CLASS_MANUFACTURER_SPECIFIC
COMMAND_CLASS_VERSION
COMMAND_CLASS_SECURITY
COMMAND_CLASS_BATTERY
COMMAND_CLASS_ASSOCIATION
COMMAND_CLASS_CONFIGURATION
COMMAND_CLASS_PROTECTION
COMMAND_CLASS_DOOR_LOCK
COMMAND_CLASS_USER_CODE
COMMAND_CLASS_SCHEDULE_ENTRY_LOCK (V2)
COMMAND_CLASS_DOOR_LOCK_LOGGING
COMMAND_CLASS_ALARM
COMMAND_CLASS_TIME_PARAMETERS
COMMAND_CLASS_BASIC

The Leviton interface is capable of sending any Z-Wave command class and command, plus parameter data. The commands for doing this are the SE (send) and SS (send secure, i.e. encrypted) commands. Most lock commands must use SS. The ASCII format of these commands is:

>Node #,[SE or SS],command_class,command,parameters 1-n.

An example of an unencrypted exchange you could have between a Z-wave sensor, such as a thermostat, and the Leviton VRC0P, using a terminal emulation program such as Hyperterminal, would be:

>N6,SE,49,4	<- Send multilevel sensor GET
<E000	<- The interface processed the input
<X000	<- Appropriate message has been sent correctly
<N006:049,005,001,009,077	<- REPORT from the sensor:

049 – Command class
005 – Command REPORT
001 – The value sent is temperature

009 – The value represented by 1 byte and in deg. F
077 – Temperature 77F

An example of an encrypted exchange you could have between a Z-wave lock and the Leviton VRC0P, using a terminal emulation program, would be:

```
>N4,SS,98,2                                          <- Command for Lock Status
<E000                                                    <- Request Processed
<N004:152,128,120,002,035,240,079,104,236,241            <- Security nonce message
<X000                                                    <- Request has been sent
<N005:152,064                                            <- Security command class
<n004:000,098,003,255,000,000,254,254                    <- Lock Status Response
```

098 – Command Class
003 – Command REPORT
255 – Lock Mode: Door Secure, i.e. "Locked"
000 – Inside and outside door handles inactive
000 – Door open, bolt locked, latch open
254 – Lock timeout is not supported
254 – Lock timeout is not supported

In the two examples above, the "<E000" indicates that the sent command was correct and the device had enough resources to implement the command, i.e. no error.

In the two examples above, the "<X000" indicates that the transmission was successful.

Other "E" and "X" response codes can be found in the Leviton documentation previously mentioned, and HCA provides the "E" and "X" values returned in flags that are specified in the HCA Z-Wave command.

NOTE: HCA inserts the ">" for you at the beginning of the command, and strips the corresponding < from responses received, so an example of a command in HCA to set the re-lock time on node 4, a Yale Z-wave lock, to 90 seconds, would be:

N4,SS,112,4,3,1,90

N4 – Node 4
SS – Secure send
112 – Command Class
4 – Command SET, and there are three parameters:
3 – Yale parameter number for re-lock time,
1 – Parameter for the number of values that follow (i.e. length), and
90 – Re-lock time in seconds.

Some Things You Can Do

Here are a series of example commands for Yale's Z-Wave lock:

Set A User Code:

N4,SS,99,1,user_code_num,1,xxx,xxx,xxx,xxx,[xxx,xxx,xxx,xxx]

N4 – Node 4
SS – Secure Send
99 – Command Class = COMMAND_CLASS_USER_CODE
1 – Command SET
user_code_num = User code being set (Yale supports 250 codes)
1 – User ID Status field, 1 = Occupied
xxx,xxx,xxx,xxx – user code digits in ASCII decimal per the following list
(Yale lock supports 4 and 8 digit codes, last four fields only used if 8 digits code is being set)

0 (048)
1 (049)
2 (050)
3 (051)
4 (052)
5 (053)
6 (054)
7 (055)
8 (056)
9 (057)

Request Battery Level:

N4,SS,128,2

N4 – Node 4
SS – Secure Send
128 – Command Class
2 – Command GET

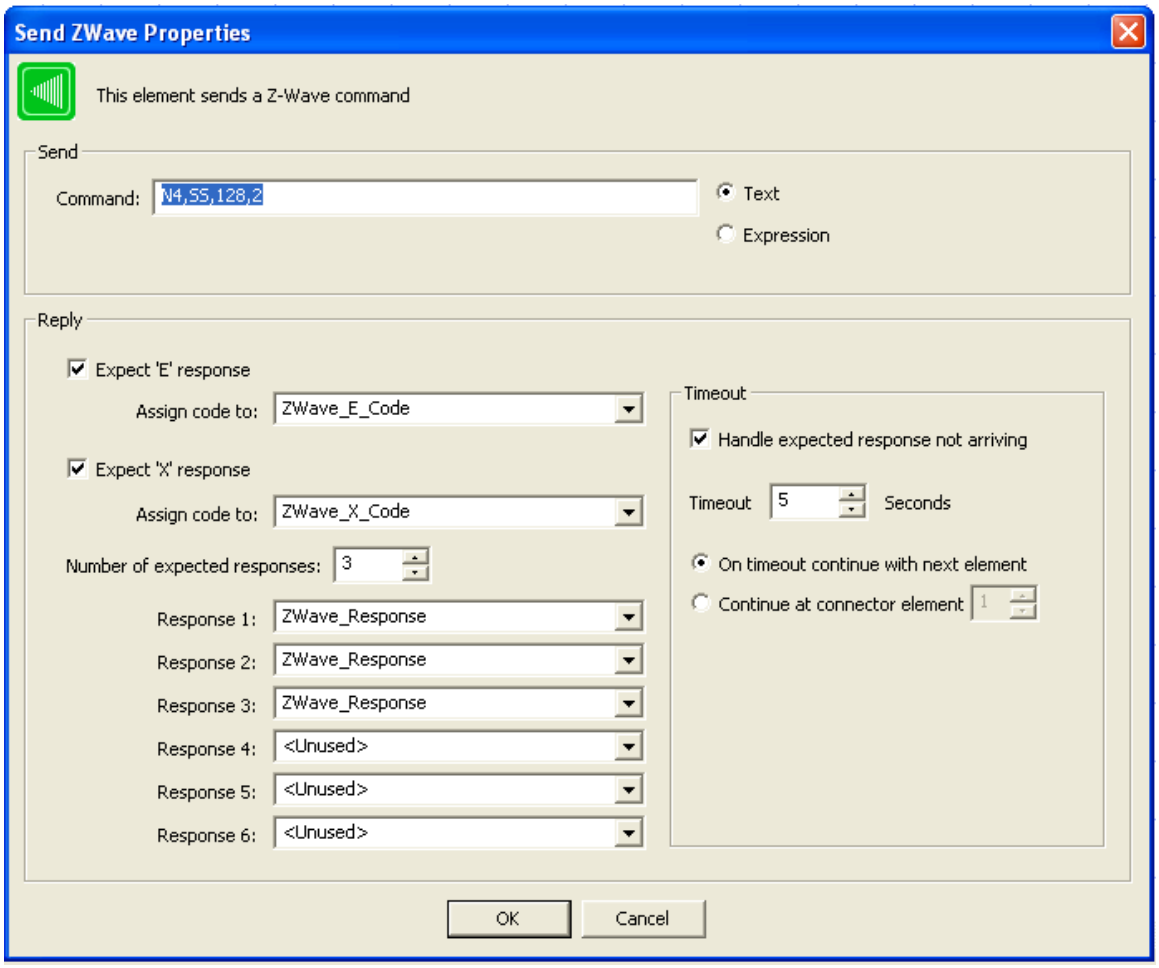
Response:
N004:152,128,119,058,171,059,157,137,129,245
N004:152,064
n004:000,128,003,088

The first two lines of the response are due to this being a secure (encrypted) command, and are not of interest.

The third line contains useful data. “004” is the node number. The lowercase “n”, rather than an uppercase “N”, is due to this being a response to a secure command. A non-secure command would have an “N” rather than an “n”.

000 – Unimportant
128 – Command Class
003 – Command REPORT
088 – Battery level (88%)

Here’s an example of the HCA dialog box for this command (a timeout of at least 5 seconds is recommended):



Set Lock’s Real Time Clock:

N4,SS,139,1,Year_upper_byte,Year_lower_byte,Month,Day,Hour,Minute,Second

139 – Command Class = COMMAND_CLASS_TIME_PARAMETERS
1 – Command (SET)

Year sent as two bytes. Example 2013 = 11111011101, upper byte = 111 = 7, lower byte = 11011101 = 221
Month (1 – 12)
Day (1 – 31)
Hour (0 – 23)
Minute (0 – 59)
Second (0 – 59)

Read Lock's Real Time Clock:

N4,SS,139,2

N4 – Node 4
SS – Secure send
139 – Command Class = COMMAND_CLASS_TIME_PARAMETERS
2 – Command (GET)

Lock reports:

n004:000,139,003,007,221,007,004,016,014,047

which can be parsed as:

n004 – Node 4
000 – Unimportant
139 – Command Class = COMMAND_CLASS_TIME_PARAMETERS
003 – Command (REPORT)
221 – Year sent as two bytes. Example: 2013 = 11111011101, upper byte = 111 = 7, lower byte = 11011101 = 221
7 – Month (July)
4 – Day
16 – Hour (4 PM)
14 – Minute
47 – Second

Unlock Door:

N4,SS,98,1,0

N4 – Node 4
SS – Secure Send
98 – Command Class = COMMAND_CLASS_DOOR_LOCK
1 – Command SET
0 – Unsecure lock

Lock Door:

N4,SS,98,1,255

N4 – Node 4
SS – Secure Send
98 – Command Class = COMMAND_CLASS_DOOR_LOCK
1 – Command (SET)
255 – Secure lock

Limit a user code to certain times/days of the week (Yale lock)

Example: You want to allow access via user code 4, from 9A to 10:30A, on Monday, to allow your cleaning person access. The command would be:

N4,SS,78,16,1,4,1,2,9,0,1,30

N4 – Node 4
SS – Secure send
78 – Command Class = COMMAND_CLASS_SCHEDULE_ENTRY_LOCK
16 – Command (SCHEDULE_ENTRY_LOCK_DAILY_REPEATING_SET)
1 – Set Action [0 = erase schedule or 1 = set schedule]
1 – User Identifier [1 to number of users lock supports (Yale supports 250 codes)]
1 – Schedule Slot ID [1 to number of slots lock supports (Yale supports 1 Slot ID/user)]
2 – Week Day Bitmask [low 7 bits represent each day of the week, with Saturday as the highest order bit, Friday the next lowest bit, and Sunday the lowest order bit.]
Example: All days would be 127, M-F would 62, M is 2.
9 – Start Hour
0 – Start Minute
1 – Duration Hour
30 – Duration Minute

Creating Triggers

User Unlocks Door:

When an event, such as a user entering an access code to unlock the lock occurs, HCA will receive a string asynchronously. This string can be used as a trigger for an HCA program. For example, when user code #2 is entered on the lock keypad, HCA receives the following two strings, (the first is security related, and can be ignored):

N004:152,064
n004:000,113,005,019,002

This can be parsed as:

n004 – Node 4
000 – Unimportant
113 – Command Class = COMMAND_CLASS_ALARM
005 – Command (REPORT)
019 – Alarm Type = 019 indicates keypad unlock
002 – Alarm Level = 002 indicates the user 2 access code was used to perform the action

An HCA Z-wave trigger that would respond to n004:000,113,005,019,002 would be:

n004:.....019,...

Note that periods in the above string are wild cards for a given character position, i.e. they match any character, so the exact number is critical, since it represents a character received from the interface. If you turn on logging for the Z-Wave interface, you can see the full string in the log when an asynchronous event happens.

The triggered program could then use the HCA _mid() string function to extract the 002 and determine who unlocked the lock. A good use of this might be to send you a text message when your child arrived home from school. All you need to do is assign each family member his or her own access code.

Lock auto-relocks:

When the lock auto-relocks (assuming you have auto-relock enabled) after the relock time has passed, HCA receives this string:

N004:152,064 (can be ignored)
n004:000,113,005,027,001

This can be parsed as:

n004 – Node 4
000 – Unimportant
113 – Command Class = COMMAND_CLASS_ALARM
005 – Command (REPORT)
027 – Alarm Type = 027 indicates auto re-lock cycle complete, locked
001 – Alarm Level = 001 indicates auto re-lock cycle complete, locked

Lock sends low battery alarm:

Create these triggers for the Z-wave interface (example is for node 4):

n004:.....167,... (this is for low battery)
n004:.....168,... (this is for critically low battery)
n004:.....169,... (this is for battery too low to operate lock)

Batteries replaced, set clock:

After replacing the batteries in the lock, the real time clock in the lock should be set. Create this trigger from the Z-wave interface (example is for node 4) to trigger your program for setting the real time clock (see example above for command to set clock):

n004:.....130,...

Wrong code entry limit has been exceeded:

In default configuration, the Yale lock will disallow further code entries for a period of time after the maximum number of wrong code entries is reached (default is five incorrect entries). You may want HCA to take an action if this happens, for example, alert you by text message, or turn on all your outdoor lights if it is dark. The trigger for doing this would be (example is for node 4):

n004:.....161,...

Yale Lock Configuration Table

COMMAND_CLASS_CONFIGURATION (see configurable parameters chart below; defaults in **bold**). Note that hex values shown in the table must be changed to decimal for the Leviton interface.

An example of a configuration command based on the table below to turn on the LED on the lock (node 4) so that it periodically flashes would be:

Set Lock Status LED:

N4,SS,112,4,13,1,255

N4 – Node 4
SS – Secure Send
112 – Command Class = COMMAND_CLASS_CONFIGURATION

4 – Command (SET)
13 – Lock Status LED
1 – Size (1 byte)
255 – LED ON = 255 = 0xFF

Set Auto Relock Time:

N4,SS,112,4,3,1,90

N4 – Node 4
SS – Secure Send
112 – Command Class = COMMAND_CLASS_CONFIGURATION
4 – Command (SET)
3 – Parameter # (3 = relock time, see chart below)
1 – Size, in bytes, of what follows
90 – Relock in 90 seconds (can be 5-180 seconds)

Set Lock Audio Mode to Low:

N4,SS,112,4,1,1,2

N4 – Node 4
SS – Secure Send
112 – Command Class = COMMAND_CLASS_CONFIGURATION
4 – Command (SET)
1 – Parameter # (1 = Audio Mode, see chart below)
1 – Size, in bytes, of what follows
2 – Audio level (2 = Low)

Name	Parameter Number	Size	Value	Description
Audio Mode	1	1 byte	1, **2** or 3	3-level control; 1=Silent **2=Low** 3=High (for Product IDs 0x01, 0x02 only) For Product IDs 0x03,0x04: **0x03 = ON** 0x01 = OFF
Auto Re-lock	2	1 byte	**0x00** 0xFF	**0x00 = OFF** 0xFF = ON

Re-lock Time	3	1 byte	5-180 Unsigned Integer	seconds; after successful code entry and unit unlocks, it will automatically re-lock after specified time (**30** = default value)
Wrong Code Entry Limit	4	1 byte	1-7	The number of invalid code entries lock will accept before sending TAMPER Alarm. When number of wrong code entries is exceeded, lock will disable keypad for the time specified by Shutdown Time parameter. (**5** = default)
Language	5	1 byte	**1**,2,or 3	**1=English** 2=Spanish 3=French (for Product IDs 0x01, 0x02 only)
Shutdown Time	7	1 byte	1-255 Unsigned Integer	number of seconds unit will be inoperable after number of wrong code entries is ex-ceeded (**60** = default)
Operating Mode	8	1 byte	**00** 01 02	**Normal Mode** Vacation Mode - all user codes disabled - except for Master Code. Privacy Mode - all user codes disabled and RF Lock/Unlock disabled.
One Touch Locking*	11	1 byte	0x00 **0xFF**	0x00 = OFF **0xFF = ON**
Privacy Button*	12	1 byte	**0x00** 0xFF	**0x00 = OFF** 0xFF = ON

Lock Status LED*	13	1 byte	**0x00** 0xFF	**0x00 = OFF** 0xFF = ON

Yale Lock Alarm Table

COMMAND_CLASS_ALARM

For an example of alarm reports, see example above for a lock reporting low battery.

Note: values are hexadecimal; they must be changed to decimal for use with the Leviton Z-Wave interface.

Alarm Chart

Alarm Reports	Alarm Type	Alarm Level	Description
Master Code Changed or User Added	0x70	0x(00 - F9)	Master code was changed at keypad. Alarm level indicates user slot # where slot #0 is Master Code location. Additional users occupy slots 1-249.
Tamper Alarm	0xA1	0x01 0x02	Keypad attempts exceed code entry limit Front escutcheon removed from main
Manual Unlock	0x16	0x01	By key cylinder or inside thumb turn
RF Operate Unlock	0x19	0x01	by RF module
Manual Lock	0x15	0x01 0x02	By key cylinder or inside thumb turn By touch function (lock and leave)
RF Operate Lock	0x18	0x01	By RF module
Keypad Lock	0x12	0x(00 - F9)	Where Alarm level represents user slot number
Keypad Unlock	0x13	0x(00 - F9)	Where Alarm level represents

			user slot number
Deadbolt Jammed	0x09	0x00	Deadbolt motor jammed
Low Battery Alarms	0xA9 0xA8 0xA7	0x01	Too low to operate Critical Battery Level Low Battery
Auto Lock Operate Locked	0x1B	0x01	Auto re-lock cycle complete, locked
Duplicate Pin-code error	0x71	0x(00 - F9)	Where Alarm level represents user slot number Alarm is generated if code specified in User_Code_Set command already exists in the lock's list of codes.
RF Module Power Cycled	0x82	0x01	Power to RFM was restored, sent by RF module
User Deleted	0x21	0x(01 - F9)	Alarm Level refers to user number
Lock Handing Completed	0x81	0x01	Lock has completed Handing Cycle