Cliff Roth
212-777-1180
cliff@techeditorial.net
4000 words
3/11/22

Introduction to Z-Wave Security

Encrypted communications within a Z-Wave Network for secure locks, sensors and other wireless devices

TK - Sigma Designs

Abstract:
The Z-Wave Security Layer provides a robust and highly secure level of protected communication for signals travelling between wireless security devices such as door locks, window sensors and motion detectors and a security or home automation system controller. Based on a true random number generator and industry-standard encryption techniques, in combination with strict time-outs and ultra-low-power technology that can limit signal range to just a few feet (making eavesdropping of password codes impossible), the Z-Wave Security Layer can be easily implemented by product designers. This article provides an overview of how the Z-Wave Security Layer works, and explains which aspects of security are handled by Z-Wave, and which aspects must be implemented by an OEM using Z-Wave communications for security-related products.

***

Demand for wireless security systems and home automation is on the fast track, with many cable-TV and telephone companies jumping in to offer these services. Most new home security and business security systems use wireless communications, for installation ease, but the use of wireless technology does raise some security concerns of its own. Fortunately, Z-Wave wireless technology incorporates security features that make Z-Wave a solid choice for applications in home automation, security, and building control. The Z-Wave Security Layer provides Z-Wave communications with robust, reliable encryption and decryption. With secure communications, devices such as smart door locks, sensors and alarm systems are trustworthy components for providing home security, or security for office buildings, hotels, and other commercial premises. This paper explains how the Z-Wave Security Layer works, and how designers can take advantage of it for secure, low power communications.

What is Z-Wave?

Z-Wave is a wireless mesh networking protocol and set of radio and application layer interoperability specifications that is widely used in home connectivity, access, security, and sensor applications. The range of Z-Wave is up to about 100-meters (330-feet), depending on terrain and indoor obstructions. Mesh network repeaters are built into most AC-powered Z-Wave devices, extending range and providing coverage to difficult-to-reach rooms and outdoor areas. This flexibility combined with extremely low power consumption have made Z-Wave the leading choice for battery-powered wireless security devices such as window sensors, motion sensors, and door locks, as well as for the central controllers that these devices communicate with.

Created initially for command and control of devices such as lights and home entertainment in the home, the Z-Wave specifications have developed over the years to meet the evolving needs of consumers and service providers. Throughout its growth the Z-Wave Alliance, Z-Wave developers, and the Z-Wave team have kept true to three key principles:

- To develop and maintain a single open standard for application layer interoperability across all devices.
- To develop and maintain a lightweight, power-efficient, and robust mesh-networking protocol.
- Maintain backwards compatibility with previous generations of Z-Wave devices across all layers.

The development of the Z-Wave Security Layer was consistent with all these principles.

Z-Wave Security in the Connected Home

Originally conceived in 2007 to meet the needs of the emerging market of wireless door locks for the connected home, the Z-Wave Security Layer was developed in conjunction with Z-Wave developers and several outside security consultant firms. The result of this work was the establishment of an application layer security specification that:

- Provides secure end-to-end communication between devices
- Allows secure and non-secure nodes to coexist in the same network
- Allows non-secure nodes to act as repeaters for secure communications

The Z-Wave Security Specification has enabled the successful deployment of secure door locks and other secure devices to *millions* of homes!

With the Z-Wave Security Specification, mixed networks combining secure and non-secure devices are perfectly acceptable, since non-secure nodes may act as repeaters for secure (encrypted) communications.

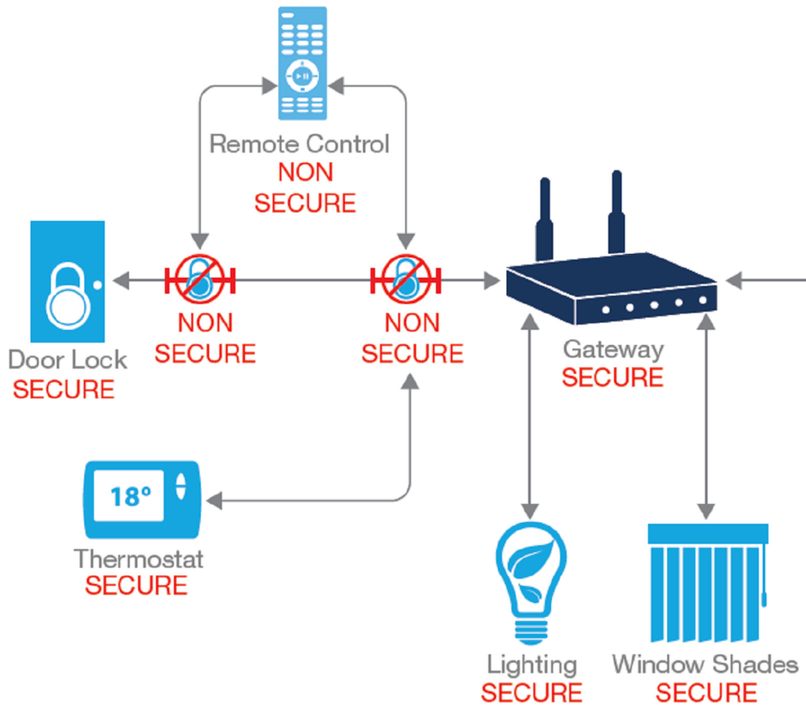


Figure 1 - A Z-Wave network may combine secure and non-secure devices, with encrypted secure messages safely passing through non-secure repeater devices.

The Z-Wave Security Layer

The Z-Wave Security Layer is an application layer specification. It defines the mechanism for enabling secure 128-bit Advanced Encryption Standard (AES-128) application communication between nodes in a Z-Wave network. The Security Layer provides the necessary mechanisms and requirements to ensure integrity, confidentiality, and data freshness of encrypted messages. The Security Layer provides protection and detection and/or reaction against several types of security attacks.

The Z-Wave Security Layer defines the commands and process used to facilitate:

- Message Encapsulation, the task of taking a plain text frame and encapsulating the frame into an encrypted security message

- Network Key Management, the task of initial key distribution

- Secure Command Handling, the task of identifying and handling what commands are securely supported when communicating with a security enabled device

These are each explained more fully in the next few sections.

Message Encapsulation

Message encapsulation -- the sending of an encrypted message -- requires the exchange of three commands as illustrated in the figure below.

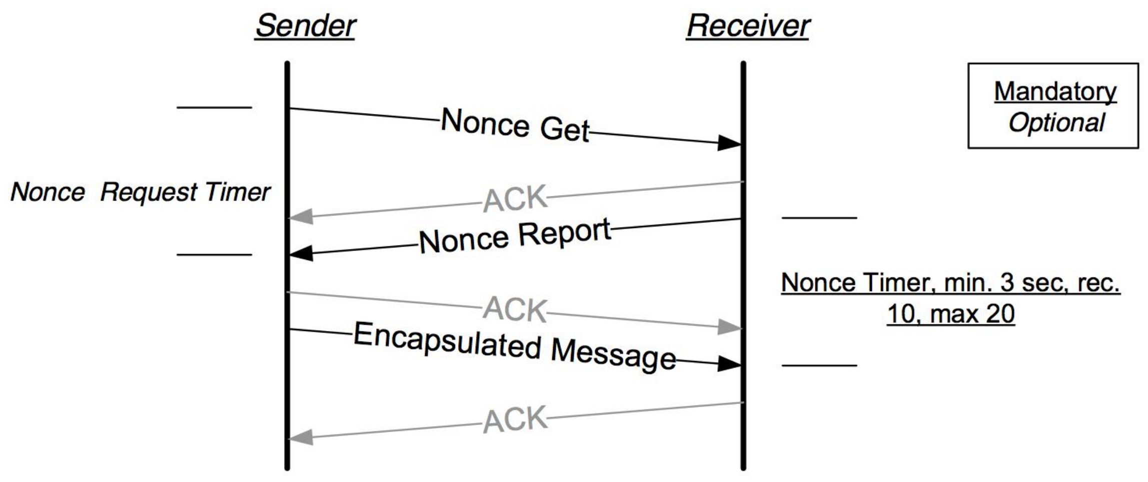


Figure 2 - Transmission of a secure message

The details of this communication protocol are summarized in the exchange below, where Node A, the Sender, wishes to send a secure message to node B, the Receiver.

1. Node A sends a request for a nonce (i.e., a ***n***umber used ***once***) to node B via the Nonce Get command unencrypted.

2. Node B uses its Pseudo Random Number Generator (PRNG) to generate a random 8-byte nonce and then sends it in a Nonce Report command to node A. Node B also stores the nonce in his internal nonce table and initiates a timer for how long the nonce is valid (adjustable from 3 to 20 seconds).

3. Upon reception of the Nonce Report Node A then:
    a. stores that value as receiver's nonce for node B and the associated Receiver's nonce Identifier (RI).

    b. uses its PRNG to generate an 8-byte sender's nonce. This new nonce is concatenated with the receiver's nonce to form the 16-byte Initialization Vector (IV)
    c. Node A then encrypts its intended payload using the encryption key KE (explained below) and the IV.
    d. Node A the computes a Message Authentication Code (MAC) using the authentication key KA (explained below), the encrypted payload, and the IV.
    e. Node A then composes and sends the Security Message Encapsulation frame to node B, comprised of the Node A's generated nonce, Node B's RI, the encrypted payload, and the MAC.

4. Upon receiving the Security Message Encapsulation frame, Node B
    a. uses the Receiver's nonce Identifier (RI) in the package to identify the correct receiver's nonce in its "internal nonce" table
    b. if the timer for the RI has expired the frame MUST be discarded
    c. If the timer has not expired then Node B reconstructs the IV and verifies the correctness of the MAC
    d. if the MAC is incorrect, the frame MUST be discarded
    e. if the MAC is correct the original payload is decrypted and passed to higher layer application logic for handling.

As the secure message encapsulation process requires three frames (not including protocol acknowledgments) to send one message, the overhead for encrypted communication is high. To help optimize performance the Z-Wave Security Specification allows for embedding multiple nonce requests into one Encapsulated Message frame, reducing the frame overhead and also enabling streaming data operations.

Network Key Management

A node that has been securely included into the Z-Wave network is characterized by being in possession of the AES Shared Network Key. This Network Key (KN) is 128 bits long and is shared by all secure nodes in the network.

The Network Key (KN) is initially generated by the controller. (The controller is the hub of the Z-Wave network.) The key is generated when the system is set up by taking 16 random bytes from the PRNG.

The Network Key (KN) is stored in NVRAM. It is never used directly for encryption or authentication purposes. Instead the Authentication Key (KA) and the Encryption Key (KE) are derived from KN, as follows:

$$KA = AES(KN; V1) \qquad KE = AES(KN; V2)$$

Where V1 and V2 are defined as 0x55 (85 in hexadecimal) and 0xAA (170 in hexadecimal) respectively, and repeated 16 times. The derived keys KA and KN are then stored in SRAM.

(AES represents the 128-bit Advanced Encryption Standard, essentially a function generator that receives two inputs consisting of a key and a value, and creates one encrypted output. The key is 128-bits long.)

Distributing KN During Z-Wave Device Inclusion

With the Network Key (KN) established, the controller is now capable of distributing KN to Z-Wave Security Enabled devices immediately after successful inclusion of the node into the controller's network. At no other time is KN to be distributed, nor should a device accept KN. This process is illustrated in the figure below. (The initial Scheme Get is for the controller to learn the node's supported security scheme.)

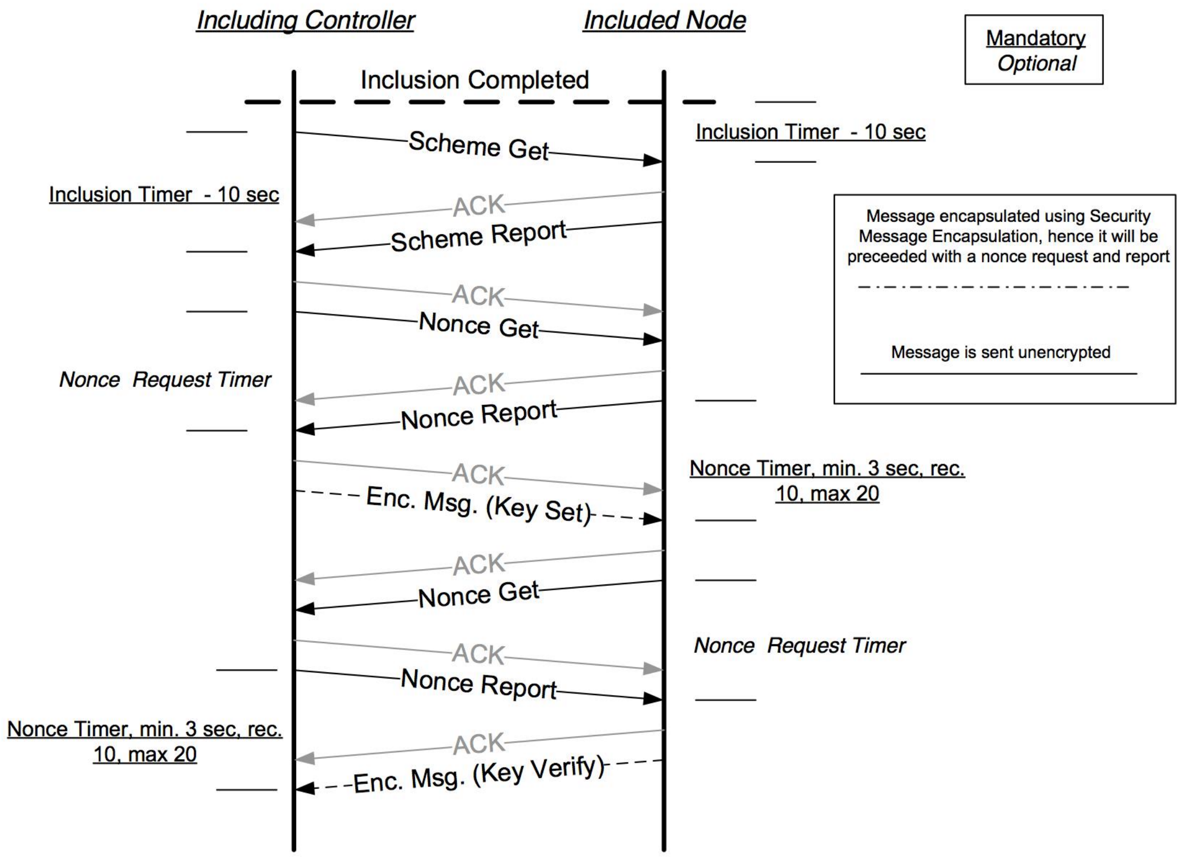


Figure 3 - Network key distribution

If the Network Key Verify fails or if any of the diagramed timeouts are reached, the Z-Wave Security Specification requires that the device be listed as non-secure in the network's trust center. Once listed as non-secure the specification prohibits any future attempts at secure communication with the device and prohibits the device from responding to any secure communications while it remains in the network. However the node may communicate with other non-secure nodes in the network.

To establish the non-secure node as a secure node in the network the Z-Wave Security Specification requires the node to be removed from the network, and the node inclusion process to be repeated as previously illustrated.

Secure Command Handling

A node that has been securely included into the network is capable of interacting with both secure and non-secure nodes in the network.

- Non-secure interactions are available to any nodes in the network via published commands supported by the node.
- Secure interactions are only available to other secure nodes in the network. The secure commands supported by a device can only be queried via the Z-Wave Security Layer.

A Z-Wave device that implements the Z-Wave Security specification will often support a few commands non-securely for device and manufacturer identification, and support the primary commands for device operation only via encrypted communications. The Z-Wave Security Specification requires all devices to strictly adhere to their published list of commands supported securely and non-securely. A device is prohibited from acting upon or sending commands in a non-secure fashion if they are advertised as supported securely.

It is up to the implementation of each application to decide which commands should be supported using security encapsulation. For example, a device may choose to support all its command classes non-securely if it is being included in a non-secure network, but use security for all command classes when included in a secure network.

Implementation of the Z-Wave Security Specification in Devices

The Z-Wave SDK provides devices with a rich set of APIs for both Z-Wave network interaction, such as the transmission and reception of frames with other devices, and for interacting with the various hardware subsystems utilized by embedded devices, such as I/O peripheral control.

As an application layer specification it is important to note that the implementation of the Z-Wave Security Specification is comprised of elements from both the Z-Wave SDK, offered by Sigma Designs, and elements implemented by application of the device provided by the OEM. Here is an overview of the key security components required to implement the security specification and where they reside.

Security Elements Implemented in the Z-Wave SDK

Hardware Random Number Generator - RFRNG

The Z-Wave hardware utilized in all Z-Wave devices is capable of producing true random numbers using internal random RF noise found in the transceiver. The use of this feature is both slow and requires that all radio communications be temporarily disabled. As such this feature is only used for algorithms depending on true randomness, e.g. as a seed generator, and for establishing the shared Network Key (KN).

Pseudo Random Number Generator – PRNG

To supplement the RFRNG the Z-Wave SDK also features a pseudo-random number generator (PRNG) that is seeded by the RFRNG. This generator has a 128-bit inner state (stored in SRAM) and three functions: state generation, state update, and output generation. The solution is based on the Barak-Halevi PRNG proposal [1] and the Davies-Meyer hash function [2], both of which are provably secure. The PRNG is used for the generation of all nonces and the associated Initialization Vector.

Timers

The Security Command Class is governed by a number of timers that MUST be adhered to. These timers can be categorized into two groups:

- Message Timers: When sending and receiving secure messages there are two timers in effect, if any of the two timers are exceeded the message MUST be discarded.
    - Nonce Request Timer that is optional for the application to implement. This timer governs the duration of time permitted from sending a nonce request until receiving a nonce report.
    - Nonce Timer that is mandatory and MUST be implemented by the application. This timer states the duration of time permitted for the receiving node to wait for the Security Message Encapsulation frame to be received after it has sent the

nonce report.

- Inclusion Timers: During inclusion each individual step of the inclusion process has a specific timer that must not be exceeded. If the timer is exceeded that secure inclusion MUST be aborted and the newly included node MUST become node secure.

The Z-Wave SDK provides OEM applications with several software timers for use in tracking the above specified security timers.

Advanced Encryption Standard Electronic Code Book - Core AES

The security layer uses 128-bit Advanced Encryption Standard (AES-128) for both message encryption and message authentication. This algorithm encrypts a single 128-bit block of plaintext into a single 128-bit ciphertext using the plaintext and the previously defined Authentication or Encryption Key (KA or KE) as input.

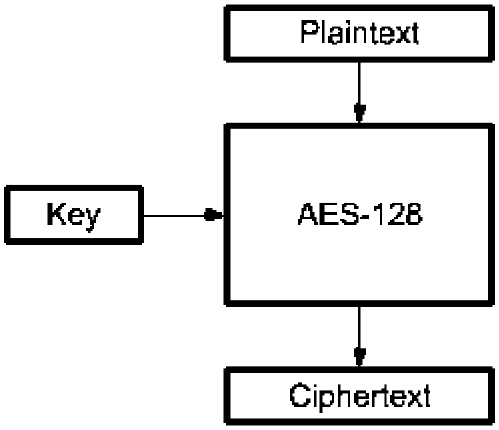


Figure 4 - AES-ECB (Electronic Code Book)

Beginning with the introduction of the 400 series Z-Wave Hardware platform all Z-Wave radios contain a built-in NIST standardized AES 128 block cipher hardware engine (NIST is the U.S. federal standards agency, like a national version of the ISO.) The cipher engine is used by embedded Z-Wave devices to perform all encrypt and decrypt processes. (Z-Wave security enabled devices built on the 300 series hardware platform perform this cipher process in software at the application layer, utilizing verified third party libraries.)

Security Elements Implemented in the OEM Application

Initialization Vector - Nonce

The Initialization Vector is a publicly known value (as previously illustrated) constructed of both the sender’s nonce and the receiver's nonce through simple concatenation:

IV = (sender's nonce || receiver's nonce)

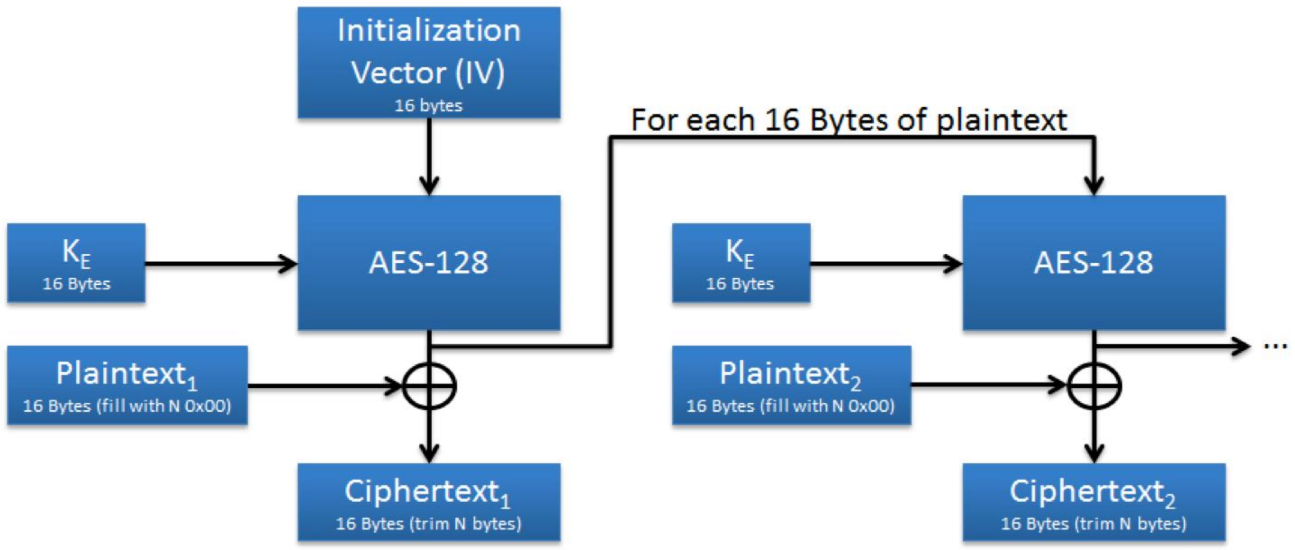
In the security layer the nonces have two properties:

- They are fresh, i.e., an attacker cannot predict them before they were generated, and they remain valid only a short time. Thus, they can be used to check whether a message associated with the nonce is also fresh.

- They are used only once, i.e., an attacker does not gain anything from storing the message and nonce in the hope that the same nonce is used again.

Payload Encryption / Decryption – AES OFB

The payload (commands) within a Security Message Encapsulation frame is encrypted using an algorithm known as AES Output Feedback (AES-OFB). In AES-OFB the IV is used as input for the AES-128 function along with the Encryption Key KE to generate a ciphertext. This ciphertext is XOR’d with the first 16 Bytes of the payload plaintext. If the plaintext is less than 16 bytes it is padded with 0x00). If the payload of the plain text is greater than 16 bytes the resulting XOR’d value is used as an input into the ciphertext block along with KE. The produced ciphertext is then XOR’d with the next block of plaintext. This process is repeated until the entire contents of the plaintext payload are XOR’d with the produced ciphertext. Due to the nature of the algorithm this process is used for both payload encryption and payload decryption, with the ciphertext being substituted with the plaintext for decryption.



Initialization Vector (IV) 16 bytes
For each 16 Bytes of plaintext
K_E 16 Bytes
AES-128
K_E 16 Bytes
AES-128
$Plaintext_1$ 16 Bytes (fill with N 0x00)
$Plaintext_2$ 16 Bytes (fill with N 0x00)
...
$Ciphertext_1$ 16 Bytes (trim N bytes)
$Ciphertext_2$ 16 Bytes (trim N bytes)

Figure 5 - AES-OFB Algorithm

Message Authentication Code - AES CBC-MAC

To verify the authenticity of a Security Message Encapsulation frame an 8-bit Message Authentication Code (MAC) is embedded as part of the frame. The MAC is created by encrypting key authentication data using the AES CBC-MAC algorithm with the previously described Authentication Key KA as input into the AES function.

The Authentication Data structure consist of the following, in order:

- Initialization Vector
- Security Header
- Sender's Node ID
- Receiver's Node ID
- Payload Length
- The encrypted payload data (all inclusive, in sequence)

As the input authentication data is greater than 16 bytes the output of the AES function is XOR'd with the next 16 bytes of authentication data and repeated until all the authentication data has been passed through. The resulting output of the last iteration is trimmed down to contain the first 8 bytes. This 8-byte value is the MAC.

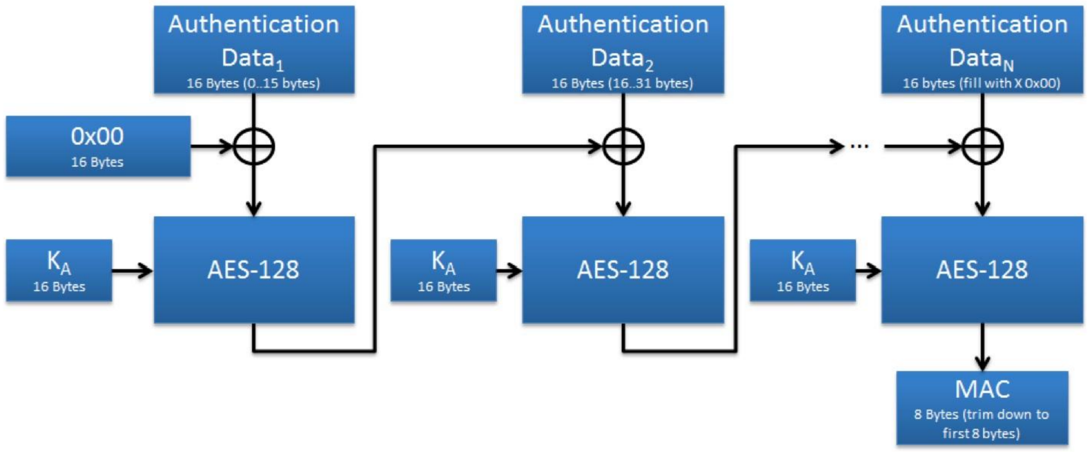


Figure 6 - AES-CBC-MAC Algorithm

Security Attacks and Error Handling

Improper handling of errors can introduce a variety of security problems. Not only do they weaken a system, but they can also allow an attacker to extract protected information about the system or a targeted device. To that end the Z-Wave Security Specification endeavors to ensure that devices not only comply with the mechanisms described, but that secure devices also properly respond to error events.

Potential Error Events

While some potential error situations have already been previously discussed in this paper we have not addressed the full spectrum of errors that may occur. Here are some more errors that can occur:

Problems with incoming Security Message Encapsulation frame

- If the Receiver's nonce Identifier (RI) found within a Security Message Encapsulation frame does not correspond to one of the nonce IDs in the receiver's internal nonce table, then the Security Message Encapsulation frame MUST be discarded.

- If the MAC in a Security Message Encapsulation frame is not correct, then the Security Message Encapsulation frame MUST be discarded

Problems with Outgoing Security Message Frames

- If an error occurs at network level when attempting to send a secure frame with payload, this error should be forwarded to the application for analysis.

- If an error occurs at network level when attempting to send a secure frame without payload, the following steps should be taken:

    - If the frame was a nonce request, then the application should be notified. At which point the application may choose to attempt to transmit the nonce request frame.

    - If the frame was a nonce report, then the nonce must be deleted from the internal nonce table and no further action is taken.

- If an outgoing encrypted frame cannot be delivered before the receiver's nonce expires, then the frame MUST be discarded. In addition, the application is notified.

- If too many internal nonces are generated in a short spell of time (i.e., before they expire or get used by the communication partner), the table of challenges will overflow. New nonces will then push the oldest ones out. In an extreme situation (e.g., a massive alarm by all sensors in a sensor network), this can lead to a deadlock situation where due to constant demands for new nonces, existing internal nonces get overwritten by new ones before the responses corresponding to the previous nonces are received. The table should thus be dimensioned in such a way that it can handle the worst-case communication situation for the application at hand.

Network Problems

- The application layer has to be aware that Security Message Encapsulation frames can get lost in transmission just as non-secure frames can.

- The application layer has to be aware that an Ack or a Routed Ack for a Security Message Encapsulation frame does not mean that the package reached the receiver's application layer. It could have been accepted by the network or routing layer (which sends the Ack or Routed Ack, respectively) and then have been discarded by the security layer (due to the wrong Receiver's nonce identifier or MAC, for example).

Failure Recovery

- If a device loses power, as may occur when A/C power is lost or when the battery is depleted, the SRAM memory contents of a device will be lost. This could affect a number of security-relevant data values. When power is restored these values MUST be reset properly:

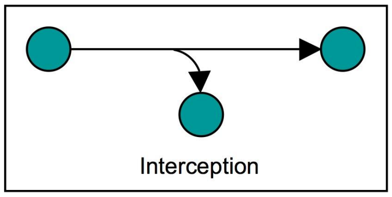
    - The PRNG has to be re-initialized.
    - The encryption and authentication keys KE and KA have to be re-computed.
    - The internal nonce table has to be reset.
    - The buffers for external nonces and waiting payloads have to be reset.

The same steps must also be taken if the controller application crashes or has to be restarted.

Security Attacks

Interception Prevention

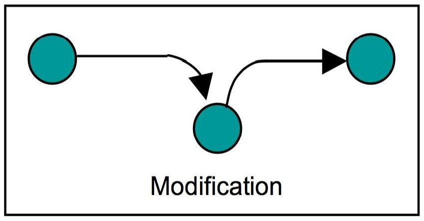
Interception is a passive attack in which an attacker analyzes traffic in an effort to reveal important knowledge about the state of the system. Traffic analysis can be performed even when the messages are encrypted and cannot be decrypted.



Interception and traffic analysis is difficult to guard against and is beyond the scope of the Z-Wave Security Specification. However it is strongly encouraged that the device's application logic exhibit the same traffic profiles regardless of the state of the system (armed, unarmed etc.). This independence of the traffic profile will make traffic analysis difficult for an attacker.

Modification Attack Prevention

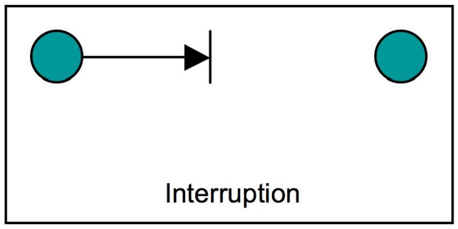
A Modification attack is an active attack in which a device intercepts a frame and modifies its content before delivering it to the intended receiver.



For an attacker to perform this action on an encrypted frame the attacked would have to be in possession of the Network Key KN. The only realistic way for an attacker to obtain KN would be to intercept (eavesdrop) and capture all RF transmission during inclusion of a secure node into the network. This is a difficult process as the attacker would have to be within RF range of a network, most likely within the premises of a homeowner, at a very specific time. To increase the difficulty of eavesdropping and make it nearly impossible, during the inclusion process the radios of the controller and the device being included can be put into a low output power mode. This typically reduces the RF range between the two devices to below 1 meter.

Interruption Attack Prevention

An interruption attack is any attack in which one or more frames are prevented from being delivered to the intended recipient.

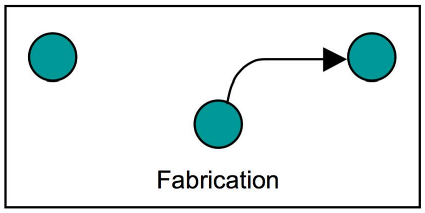


It is impossible to protect 100% against any and all kinds of interruption attacks. For example a very simple attack could be jamming the radio frequency used, or the tampering of the device in question, or a denial-of-service attack performed by saturating the target node with external communication requests.

Detection of interruption attacks is especially important in the destination nodes because they typically receive crucial alarm information, such as from window sensors. This can be achieved by sending keep-alive messages to the destination at a given rate. This enables detection of interruption of the keep-alive messages in the destination. This functionality is not provided by Sigma Designs and must therefore be implemented at the application level of the device.

Fabrication and Replay Attack Prevention

A fabrication attack is an active attack in which an attacker fabricates an encrypted packet in an attempt to fool the device into performing an action, such as returning data or activating a mechanism.



The Z-Wave Security Specification has been designed to make fabrication attacks extremely difficult to perform, as long as the intruder doesn’t know the Network Key (KN).

A modification of a fabrication attack is the Replay (and/or Preplay) attack. This is an active attack in which an attacker retransmits a previous valid data transmission either maliciously or fraudulently repeated and/or delayed.

In a Replay attack the attacker intercepts a message in transit and stores it. At some later point in time, the attacker sends this message to the intended receiver, making it believe that the message was sent just recently.

In a Preplay attack the attacker requests a response from a node before it is requested by a legitimate network node. Once the request from the legitimate node comes, the attacker sends the old, recorded response.

The Z-Wave Security Specification has been designed to protect against such Replay and Preplay attacks through required use of the nonce challenge/response mechanism with timers to catch delayed signals.

Reordering Attack Prevention

In a Reordering attack, the attacker delays a message until another message that was sent later has arrived at the destination. The Sigma Designs solution limits the interval in which reordering can occur by imposing a number of timers that dictate for how long nonces and their associated encrypted message are valid.

Network Management Protection

In addition to utilizing the reduced-power operating mode that limits range to a few feet during setup, functions such as inclusion and exclusion can also be protected by a password to avoid unauthorized operation. The OEM must implement this kind of access control at the application level.

Conclusion

Sigma Designs offers a robust portfolio of security-enabled Z-Wave solutions for OEMs incorporating the Z-Wave Security Layer. As the Z-Wave Alliance's primary silicon vendor Sigma Designs provides Z-Wave solutions featuring highly secure, reliable communications while maintaining extremely low power consumption for battery-powered devices. The Z-Wave Security Layer is crucial to achieving these objectives. Using a true random number generator operating locally and in conjunction with a reduced-range high security system setup mode, pseudo random numbers and 128-bit industry-standard encryption, the Z-Wave Security Layer delivers state-of-the-art signal protection. Full implementation of Z-Wave Security Specification requires utilizing Z-Wave Security Layer in combination with OEM implementation of other security features. For more information about Z-Wave security and obtaining a Z-Wave SDK contact info@sigmadesigns.com.

References:
1 - "A model and Architecture for Pseudo-Random Generation and Applications to dev/random," by Boaz Barzk and Shai Halevi.
http://www.cs.princeton.edu/~boaz/Papers/devrand.pdf

2 - Encyclopedia of Cryptography and Security, Volume 1, edited by Henk C.A. van Tilborg, Sushil Jajodia

Author Bio:
Jonathan Adams is _____ with Sigma Designs. He received his ___ from ____ and is currently_____.