Z-wave Faulty Encryption Detection

Jeffrey Garip (jwg327), Vishal Tyagi (vt890), Joseph Henersersky (jh5363), Touhid Chowdhury (mtc405), Derrick Du (dcd310)

Abstract

The Z-wave network protocol has been developed in order to implement in-home mesh networks for appliance and control devices. Z-wave networks are more than just online services, instead they interact with a users physical place. The importance of security in such a protocol is directly linked to the importance of security in the location and services implemented with it. One early detected vulnerability lies in the potential to downgrade encryption security in a node allowing attackers to send arbitrary commands to supposedly secure devices. We will present a solution to this problem in form of a standard protocol implemented in the controlling hub of the system.

Introduction

Z-Wave technology has given many the ability to self automate their home appliances such as doors, thermostats, lights, and many other accessories. The Z-wave protocol is used to implement a mesh network that permits many smart appliances to communicate with one another via radio frequencies. These signals are sent to a central hub which grants the homeowner access to view each individual device on the network.

Not all home automation devices require the same level of security, as such some devices require secure communication via encryption whereas others don't. This can lead to vulnerabilities in the encryption protocol that once exploited can render the device obsolete or disable all security features. Our main focus will be on the downgrade attack from s0 to s2 level security and how required backwards compatibility with s0 is putting the consumer at risk to a potential attack. Additionally, we will consider the implications of a compromised system and provide a solution for detecting a compromised node before a malicious entity can cause harm to the home in question.

Background

Z -Wave protocol was developed by Zensy in 2001 and later purchased by Sigma Designs. The PHY and MAC specifications are available publically and have been published in the ITU G.9959 [1]. The routing protocol has not been publicly disclosed but has been largely reconstructed [Figure 1] via network captures header field fuzzing by Badenhop et al in their paper "The Z-Wave routing protocol and its security implications" [2]. The application layer has been released to the public as of 2016 in order to allow tech companies to incorporate Z-wave functionality into their future consumer appliances [3].

Each device connected to a Z-wave home automation network has a list of Command Classes associated to it. For example, a Z-wave light switch can be compliant with the "Switch" command class which allows a controlling hub to send a "toggleOff" or "toggleOn" message to implement functionality. Of course different types of devices will utilize vastly different command classes so upon initial pairing into the network the new device will need to send to the network controller a list of command classes it utilizes.

A special command class called the Security command class can be used for secure communication between high importance devices. This is where s0 and s2 security is implemented. The earlier version, s0, used a preset 16-byte initialization vector of all zeros. An intruder, if present during the pairing session, could easily sniff the key exchange and thus learn the shared network key. S2 security has removed this preset key to instead work with a Diffie-Hellman key exchange method as well as providing unique keys to all device groups on the network instead of just one network key [4]. That being said, Z-wave devices still provide backwards compatibility in order to S0 devices allowing for an attacker to downgrade the improved security measures of S2. The security evaluation by Behrang Fouladi and Sahand Ghanoun [5] uncovers the vulnerability where the Z-wave does not check for the state validation. This can lead to a successful attack as the shared network key can be overwritten and

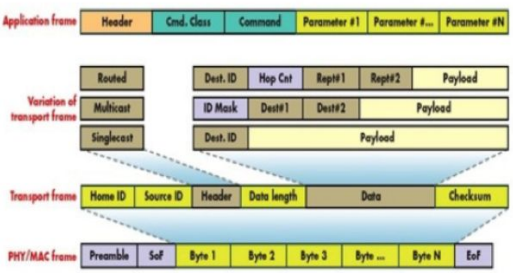

Figure 1: Z-wave Routing Protocol [11]

thus the attacker can have access control of the door lock.

The primary solution to the problem is to configure and deploy all the devices with the higher and secured S2 standard. This will remove the S0 downgrading vulnerability completely and make the devices secured. But in terms of practical approach, this solution comes at a very high expense as it requires to replace all the already deployed devices which will be too costly for the users and hence, it is not feasible. Therefore, there needs to be a solution, which we discuss in this paper, that removes the vulnerability without the same level of overhead.

Threat Model

In the previous paper, we provided a attack tree (modified to be more aligned to realistic attacks) [Figure 2] that highlighted the potential ways that an attacker can misuse z-wave to gain control of users devices and confidential information. Fuller et al. (2016) proposed a 'Misuse-Based Intrusion Detection System' (MBIDS) that monitors network traffic in an attempt to detect malicious packets [6]. Such a design is important because attackers will always be able to inject packets into network that uses radiowaves as its medium of transport. There must be certain enhancements made to the current security model to ensure that the unwanted attacks are dealt with. The updated security framework (S2) was created after a DDos attack flooded servers that originated from unprotected devices. The software is backwards compatible which is where the vulnerabilities come in [7].

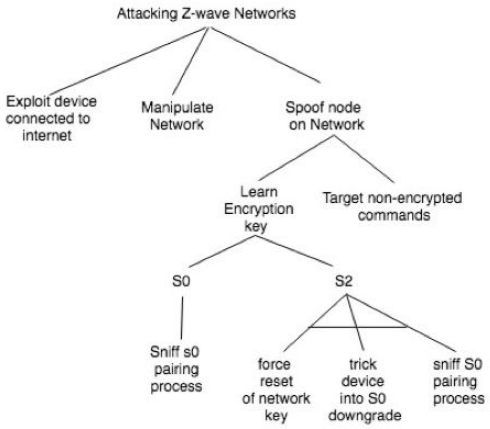

Figure 2: Z-wave Attack Tree

Below is detailed view into how threats are being mitigated by designers currently as well as possible solutions that may be implemented based on the threats on the attack tree in Figure 2.

Attacking the Z-wave network can be categorized into three sections. The first is to send exploits to devices that are connected to the internet. This would be possible by learning the public IP address of a Z-wave network gateway device. The second option would be to manipulate the network itself. This can be accomplished by sending crafted routing messages to alter the perceived network view of devices in the network [2]. The third section and the method that this paper is focused on is spoofing a network node to inject commands into the system. This can be achieved in different approaches. If the device is still using S0 for encryption, the encryption key can be sniffed during the pairing process. If the device is using S2 security, sniffing can also be used granted the attacker can downgrade the device to S0 during pairing. While an attacker could hope to be in range during the initial pairing, they may also spoof a reset command from the controller. Upon resetting most deployed devices don't check for an originally stored network key and revert back to defaults allowing an attacker to simulate the whole pairing process with them and the device [5].

Design

The solution should have the following characteristics to ensure that the system is secure:

- Be able to prevent any attacks from occurring when the system is in a vulnerable state
- The solution must use existing system to ensure that the cost is minimized
- Ensure that the system maintains all assumptions regarding performances

A Z-wave system has two types of nodes: control nodes and slave nodes. The control nodes are the ones responsible for sending commands and the slave nodes respond to commands. Slaves nodes also occasionally send commands to other slave nodes that otherwise would not be able to communicate with the control node. Each individual z-wave network has a unique HomeID (also known as network ID) that is the primary identifier for nodes in a specific network [7]. Nodes with different HomeID are unable to communicate with each other to ensure the security of each system. The Z-wave protocol supports a 128-bit key

advanced encryption standard. One of the biggest design issues with the Z-wave system is the possibility of attacker masquerading as the user and injecting false packets into the network [8]. The proposed solution will ensure that certain measures are taken to prevent such attacks as well as take action if attackers attempt to compromise the system's integrity. This includes creating a data structure to keep track of nodes that are utilizing the command class that is discussed in detail in the following section.

Implementation

While the most ideal solution to the problem is a universal upgrade of all Z-wave devices to support S2 security as mentioned before, the infeasibility comes from the implementation of this method. Such a solution would require updates of all already deployed devices and discontinue compatibility with the weaker S0 security. Embedded devices may have no simple way of firmware upgrades, especially if it exceeds the memory capacity required of its hardware, are the reason that this issue is so pertinent. Therefore it is of utmost concern that our proposed solution provides easier installation and deployment options than this original idealistic solution.

Due to our Faulty Encryption Detection scheme requiring only the support of the home's controller hub, improving the network's security need only rely on the single update of the controllers software, something that is already supported by most, if not all, controller hubs on the market. After such an update the network would continue to function in essentially the same exact fashion as before, the only difference being in the actions taken by the controller upon receiving a packet from a known node ID with unrecognized encryption. The original functionality of the controller hub is to decrypt the message with the network key and subsequently check the message authentication field to ensure correctness. In a scenario where a controller has had its encryption key maliciously modified the authentication field would of course not appear valid to the controller and the packet would be immediately dropped without further processing. This allows, say, a door lock to be hijacked without the system ever noticing [5].

Our implementation would have the controller node take action instead of staying passive upon receiving an invalid frame as described. This could be in the form of alerting the homeowner or directly triggering an alarm and potentially alerting authorities of unauthorised access to the home. Because these actions require human intervention it seems necessary to include a form of prior check for tampering with the home device in question before having the controller take further action.

In order to accomplish this we will need the controller to store an additional data structure keeping track of which nodes utilize the security command class. The security command class encapsulates other command classes to provide encryption, authentication, and integrity [2]. This proposed data structure would need 1 byte for the node ID and an additional byte for the boolean value indicating if it utilizes the additional security layer [2]. With 255 possible node IDs (excluding the controller node ID) this will in total take around half a kilobyte, something very likely possible for any controller hub on the market.

Upon receiving a seemingly faulty message for a specific node ID, the controller will check this node ID against the proposed data structure. If indeed the node ID is known to use the security layer the controller will then send a message encapsulated by the security layer to this node and monitor its response. For the choice of which message to send, it is preferable to send a message that simply warrants a response but does not change anything within the device. Additionally, it is preferable to have one message type that will be sent to whichever node is in question, regardless of what type of device that is. Fortunately, all command classes are built on the superclass named "CommandClass" which has built in functions that all nodes must be configured to handle appropriately [9]. For example, in the OpenZwave specification the CommandClass class includes a GetNodeID function which is very suitable for our implementation.

So now the steps are as follows: controller receives potentially suspicious message from a known node ID, controller checks newly defined table to determine if node ID is associated with using the Security Class, if so the controller sends an encrypted "GetNodeID" message encapsulated in the Security Layer. Upon the node in question receiving this message from the controller, one of two things will happen. Either the node will not have been compromised and it will correctly decrypt the message and generate the appropriate response. The home controller will receive the

expected response and take no further action. Alternatively, the compromised node will receive the message from the controller and decrypt the application data with its faulty network key supplied by the attacker. Of course this will lead to nonsense and the authentication field will indicate that the received data is faulty. The compromised node will subsequently drop the packet. Upon the home controller receiving no response it will now know the device is compromised and can take appropriate action such as triggering the home alarm system, contacting the homeowner, or even contacting local authorities.

Evaluation

A thorough evaluation of our proposed solution would require a full working Z-wave network setup which we could manually implement and perform various attacks on, leading to bug findings and resulting tweaks. Due to time limitations and lack of resources we will instead evaluate our solution with the assumption that our current implementation will work as expected. This still leaves us with many additional areas to consider.

Our Faulty Encryption Detection scheme will succeed in identifying nodes that are hijacked via resetting their network key. Additionally, we have circumvented false positives by including a checking mechanism to ensure a node has been hijacked before taking additional action that requires human intervention. This is important due to potential close proximity of multiple Z-wave networks and unintended traffic. Additionally this can all be implemented through updating only the controller hub's software which is extremely easy due to its connection to the internet and existing support for software updates. This is a huge improvement in both security and ease of implementation. As soon as a node is compromised and an attacker attempts to send a message to that node the homeowner will know and action can be taken quickly.

Unfortunately, this does not stop the actual take over of a node. This means an attacker can still compromise a door lock and open the front door of the house. While this is still a huge problem, if the house alarm starts ringing as soon as the door lock opens up the attacker will likely be deterred and flee before causing any harm. Additionally, there are increasingly more attacks on the Z-wave protocol that continue to be discovered by security researchers. Many of these allow attackers to take over the controlling hub themselves which allows them complete control over the Z-wave home network [8]. Our proposed solution has no defense against other attacks other than the one it is meant to deter. For further hardening of Z-wave networks we propose the routing protocols officially be made public in order to allow security researchers to do proper testing on Z-wave network attacks as well as focusing research on securing any devices connected to the internet as these provide the most accessible attack vector.

Related Solutions

A. The research work by Ramsey, Rice, Pecarina and Fuller [8] proposes use of Misuse-Based Intrusion Detection System (MBIDS) to detect the rate of injected packets, compare these packets with gateway log file entries and inform the user via an alarm or other action. This is performed by evaluation of Z-Wave command classes and payload parameters.
B. As proposed by J.D. Fuller and B.W. Ramsey[10], there are few measures that a user can take to prevent packet injections by an attacker. Few of those measures are: Reverse Proxy Server (RPS); End-Device Encryption.

Conclusion

Z-wave technology provides a solid basic protocol upon which home automation systems can be developed. That being said, Z-wave is yet to have an explicit standard, especially as new device types created by different manufacturers can be incorporated into Z-wave networks at any time. While this leaves a vast attack surface that is yet to be fully enumerated by the security community, the s2 to s0 security downgrade attack has attracted much attention due to its heavy potential impact of being able to to breach physical security devices such as door locks. In this paper we have designed and presented a serious counter measure to such attacks or any attacks attempting to compromise the encryption key stored by a Z-wave network device exempt the controller. Our main goal in our proposed solution is to enable the controller node to take action when the system encounters a security breach by triggering a pre-set alarm. Upon receiving a message from a known node ID but that doesn't decrypt correctly the controller node

"pings" that node id. If the controller doesn't receive an appropriate response it triggers an alarm. While our implementation succeeds in its goal of not needing to update all previously deployed devices on the market, it does require the update of the controller node and additionally does not prevent the takeover of the device but only detects its compromise. Due to this, it is still advised that the switch from s0 to s2 security without backwards compatibility be made as quickly as possible. Lastly, many other attacks on Z-wave networks continue to be published ranging from network topology manipulation to gateway device exploitation. We see the future of the Z-wave protocol security being dependent on open source development to allow for proper enumeration of design flaws as well as strongly designed cryptographic systems to provide proven secure communication methods.

References

1. G.9959 : Short range narrow-band digital radiocommunication transceivers - PHY, MAC, SAR and LLC layer specifications
https://www.itu.int/rec/T-REC-G.9959
2. The Z-Wave routing protocol and its security implications
https://www.sciencedirect.com/science/article/pii/S0167404817300792?openDownloadIssueModal=true&fbclid=IwAR0SyyCb1XS3uwloYrv49E2ydXm1ouOdVYFhF_sUU_3wpoqvY9UMEMoNe-8
3. Z-wave Specifications Go Open-Source
https://www.electronicdesign.com/iot/z-wave-specifications-go-open-source
4. Q&A: The Lowdown on Z-Wave's S2 Security Support
https://www.electronicdesign.com/embedded-revolution/qa-lowdown-z-wave-s-s2-security-support
5. Security Evaluation of the Z-Wave Wireless Protocol
https://sensepost.com/cms/resources/conferences/2013/bh_zwave/Security%20Evaluation%20of%20Z-Wave_WP.pdf
6. Fuller J, Ramsey B, Pecarina J, Rice M, Wireless intrusion detection of covert channel attacks
https://books.google.com/books?hl=en&lr=&id=XD7QCwAAQBAJ&oi=fnd&pg=PA137&dq=wireless+intrusion+detection+of+covert+channel+attacks+fuller+j&ots=myr2O1_QzB&sig=uSAPjAKrog3P4slIpbzQp44vlSg#v=onepage&q&f=false
7. Z-wave
https://en.wikipedia.org/wiki/Z-Wave
8. Misuse-based detection of Z-Wave network attacks
https://reader.elsevier.com/reader/sd/pii/S0167404816301341?token=7CB89653944CA0ED038032D8855BEEB6615A48D381C1154B9643999FDD7F693D04A745967634BD3D2391A7B2E3589656
9. OpenZWave::CommandClass Class Reference
http://www.openzwave.com/dev/classOpenZWave_1_1CommandClass.html
10. Rogue Z-Wave Controllers: A Persistent Attack Channel
https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7365922
11. Catching the Z-wave [diagram]
https://www.embedded.com/design/connectivity/4025721/Catching-the-Z-Wave