The Internet of Things: A Bridge between Virtual Attacks and Tangible Consequences

Danielle Zelin

Tufts University, Department of Computer Science
Mentor: Ming Chow
Comp 116: Introduction to Computer Security

Abstract

From smart cement to smart streetlights, the Internet of Things (IoT) has become an increasingly popular field. IoT is both machine-to-machine and machine-to-sensor communication via automated devices over the Internet. It is estimated that there will be over 50 billion connected devices by 2020. Some of these smart devices include cars, bridges, planes, medical devices, etc. However, breaking into a persons car is different than breaking into a website. As people become dependent on these smart devices, the risks behind them can become life threatening. Breaches can sometimes cause physical harm to someone, especially in transportation devices and medical devices. Also, does getting access to a smart device open a door to personal information and local computers? This is why the security behind IoT is extremely important in protecting both a persons personal information and their physical well-being. Security must be a concern throughout the entire process of development. It is important to define the protocols used, the potential vulnerabilities, and what the dangers are if something is compromised. In analyzing popular protocols used to develop IoT devices and applications, security vulnerabilities can be located and methods can be determined to ideally fix the vulnerabilities. The two protocols analyzed were Message Queuing Telemetry Transport Protocol (MQTT) and Constrained Application Protocol (CoAP) because of their popularity in the IoT field. The methods in which developers use these protocols, the recommended security approaches, the current security flaws, and potential methods to fix these flaws are addressed further in the paper.

Contents

Introduction

People always wonder what the next big thing will be in technology. Just a few decades ago, the Internet was unimaginable. Even before that, devices like a television and cellphones transformed the way people lived. Little do they know that the next big thing is already among us. The Internet of Things (IoT) has already become an important part of society, yet the extent of what it can be used for is not well known. IoT is machine-to-machine or machine-to-sensor communication via automated devices over the Internet. IoT is becoming more and more popular every day with new and innovative ideas. As the popularity increases, IoT is starting to be used in more in medical devices, transportation technology, smart city development, and within homes. All of which lead to people becoming completely dependent on technology. Therefore, the protocols used for the device communication must be secure in order to ensure complete safety for the consumers. While there are a number of recommended methods for establishing security in MQTT and CoAP, they are not sufficiently capable of doing so. This paper exposes the vulnerabilities in the recommended security features and provides the basis for sound alternatives.

The Internet of Things is an important part of a new generation of technology that every person, household item, and city will eventually use. So many wireless protocols exist; however, given the scope of IoT and simplicity of the devices being used those existing protocols may not be the best fit. A lightweight protocol must be implemented in order to help devices communicate with the constrained resources that they have access to. Two of the more popular lightweight protocols used for the Internet of Things are Message Queuing Telemetry Transport Protocol (MQTT) and Constrained Application Protocol (CoAP).

MQTT is a machine-to-machine, publish-subscribe protocol. It can reach over far distances. Clients and servers communicate with each other by means of control packets as seen in Figure A. In order to reduce the amount of data sent over the network, each bit in the control packets is carefully crafted. A control packet contains up to three parts: a fixed header, a variable header, and a payload.[1] The fixed header identifies the control packet type, any related flags, and the remaining number of bytes in that packet. In Figure B below, the list of packets types and their purpose are displayed. The variable header contains the packet identifier if needed depending on the control packet type. The packet identifier is a unique identifier used to identify a message. Lastly, the payload contains the actual data that needs to be transmitted in bytes.

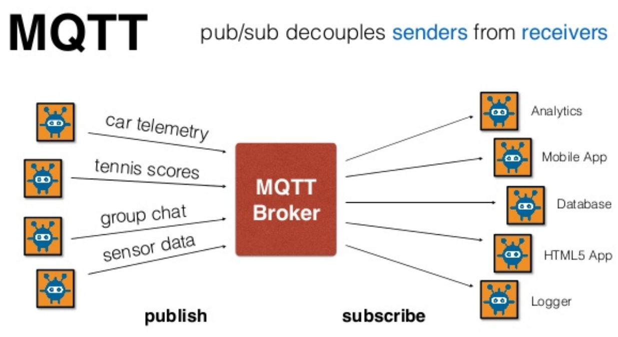
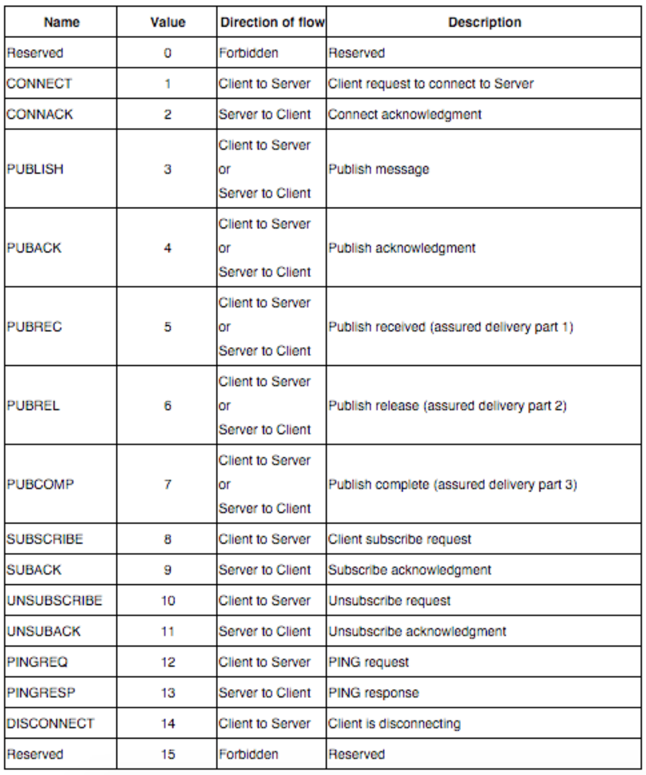


Figure A: The general publish subscribe process for MQTT protocol is displayed in this figure. As seen on the left, the clients can publish messages reporting information about themselves and all of the subscribers can view the messages through the MQTT broker.[2]

[1]SolaceSystems. *MQTT Control Packet Format*. URL: https://sftp.solacesystems.com/Portal_Docs/index.html#page/MQTT_Specification_Conformance/2_MQTT_Control_Packet_format.html.

[2]Bryan Boyd. *MQTT: A Practical Protocol for the Internet of Things*. Aug. 2014. URL: http://www.slideshare.net/BryanBoyd/mqtt-austin-api.



Name	Value	Direction of flow	Description
Reserved	0	Forbidden	Reserved
CONNECT	1	Client to Server	Client request to connect to Server
CONNACK	2	Server to Client	Connect acknowledgment
PUBLISH	3	Client to Server or Server to Client	Publish message
PUBACK	4	Client to Server or Server to Client	Publish acknowledgment
PUBREC	5	Client to Server or Server to Client	Publish received (assured delivery part 1)
PUBREL	6	Client to Server or Server to Client	Publish release (assured delivery part 2)
PUBCOMP	7	Client to Server or Server to Client	Publish complete (assured delivery part 3)
SUBSCRIBE	8	Client to Server	Client subscribe request
SUBACK	9	Server to Client	Subscribe acknowledgment
UNSUBSCRIBE	10	Client to Server	Unsubscribe request
UNSUBACK	11	Server to Client	Unsubscribe acknowledgment
PINGREQ	12	Client to Server	PING request
PINGRESP	13	Server to Client	PING response
DISCONNECT	14	Client to Server	Client is disconnecting
Reserved	15	Forbidden	Reserved

Figure B: The control packets types for messages sent using MQTT protocol are displayed above. The types are specified in the first four bits of the control packet.[3]

In MQTT, the control packets also contain a Quality of Service (QoS) level in which the sender and receiver can agree on the importance of a control packet. If the QoS is 0, the message will only be sent once and wont require an acknowledgment. If QoS is 1, the sender will continuously send the same message until an acknowledgment is received. Lastly, if QoS is 2, a message will only be sent once and will ensure that the same message will not be processed a second time. It is the safest of the three QoS levels.[4] QoS is important because on unreliable networks its easier to prioritize information in order to ensure the necessary data is received.

Similarly, CoAP is an IoT application layer protocol that is meant to be used as a replacement for HTTP. CoAP specializes in web transfer protocol for constrained nodes that may not have access to enough resources to use HTTP as seen in Figure C. CoAP operates over UDP

[3]SolaceSystems, *MQTT Control Packet Format*, op. cit.
[4]Ibid.

rather than TCP and provides GET, POST, PUT, and DELETE requests. It supports 4 types of messages: confirmable (CON), non-confirmable (NON), acknowledgment (ACK), and reset (RST). Unlike NON messages, CON messages require an acknowledgment from the receiver to confirm the message was received and processed. An ACK message is a response to a CON message. Lastly, a RST message is only sent when a CON or NON message has been received but it cannot be processed due to an incomplete or unclear message.[5]

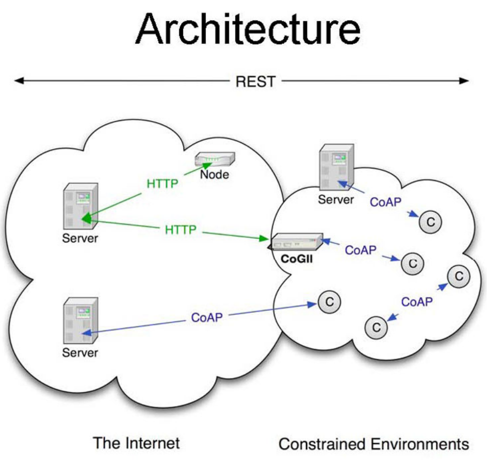


Figure C: The HTTP-like protocol for constrained nodes in constrained environments is demonstrated in this diagram. As displayed above, smaller IoT devices communicate with servers and proxies via CoAP which then use HTTP to communicate with other servers.[6]

Although CoAP is a single protocol, it is easier to think of it as a two layer approach. One deals with sending asynchronous messages on the message layer using UDP and the other is on the request/response layer using method and response codes. On the message layer, in reliable message transport, when the client sends a CON type message to the server, the server must respond ACK if message was successfully processed, otherwise RST is sent. In unreliable message transport, when the client sends NON to the server, the server must not respond unless the message could not be processed and in that case RST is sent. On the request/response

[5]C. Bormann Z. Shelby K. Hartke. "The Constrained Application Protocol (CoAP)". in: *Internet Engineering Task Force (IETF)* (2014).

[6]Jack Mannino. *Implementing CoAP The Secure Way, Part I: Fundamentals*. May 2015. URL: `https://nvisium.com/blog/2015/05/27/implementing-coap-secure-way-part-i/`.

layer, requests can be either confirmable or non confirmable. When the client sends a CON message, the server will respond with an ACK message if the message was received. If the ACK message is empty, then the client will wait for a CON message from the server because the response wasnt immediately available. The client will respond with an ACK message to acknowledge that they received a separate response from the server. Lastly, if a NON message is sent, the server will respond with a NON message as well.[7] This is important to understand because it is clear that data is transferred to and from clients and servers on a regular basis. The security concerns of the data sent on the constrained networks are addressed further in the paper. However, in order to understand security vulnerabilities of specific protocols, it is essential to have a grasp for the methods in which devices using the protocols can communicate.

To the Community

The general population currently is not aware of how much IoT technology is already in their lives. Everything from a car to an apple watch can now fall under the general branch of IoT because they involve small computers that communicate with other machines without a human proxy. With the rate in which IoT is growing (over 50 billion connected devices by 2020), every person will not only have cell phones and iPads, but they will also have smart watches, smart homes, smart cars, and more. Almost anything imaginable will be embedded with small devices that can communicate with other machines. Not only will these devices perform functions without commands, but they will also be able to observe human habits and cater their actions based on the observations. While that in and of itself is terrifying to some people, the scary thing is what will happen if these devices arent secure.

The protocols MQTT and CoAP provide developers with the means to create secure devices, but that doesnt mean all developers build their devices and technology in the most secure way.

[7]Z. Shelby, "The Constrained Application Protocol (CoAP)", op. cit.

There are several attacks that have already occurred due to overlooking security vulnerabilities. Over the last few years, there have been serious IoT breaches that may not have resulted in casualties, but could have been depending on the intentions of the attacker. For example, recently several people hacked into a massive hospital network and had access to 4.5 million sensitive records. Also, they were able to control the devices that were used in the hospitals to change the dosage a pump delivers. Although IoT is clearly an innovative and amazing new technology, if implemented insecurely, can result in catastrophe. Similarly, there is new technology that allows a user to digitally tag a target when shooting a rifle. If the technology is not secure, cybercriminals are able to change someones target and ultimately cause them to shoot somewhere that they did not intend. This can greatly affect a police sniper who needs to be precise in their targets.[8] These are two of a large collection of possible breaches with just the technology available today. If 50 billion connected devices will be available by 2020, then the amount of security vulnerabilities will also increase exponentially.

With the amount of devices connected today (12 billion), there is on average 1.7 devices for each person on the planet. If the number of devices connected increases to 50 billion, that is about 7 devices per person on the planet. That is about 5 more devices that a cybercriminal can hack into and 5 more devices for a cybercriminal to access private information on your network.

As the demand for smart devices increase, the more developers will be challenged to create innovative new technology that will change how people interact with their devices. Because security is not needed to create a device with its intended functions, when tight for deadlines, it is often the first thing to be dropped. However, that cannot be the case. If a device is not secure, the product can be dangerous and as seen from the above examples, can even be life threatening. Although meeting deadlines may seem extreme during development, taking those

[8]Bill Montgomery. *The 10 Most Terrifying IoT Security Breaches you aren't aware of (so far).* Sept. 2015. URL: `https://www.linkedin.com/pulse/10-most-terrifying-iot-security-breaches-so-far-you-arent-montgomery`.

few extra weeks to make the device secure will save the company from years of backlash if an attack occurs on their product.

Defense

As referenced in the introduction, there are several security concerns when it comes to IoT. Some potential vulnerabilities in developing IoT devices and applications include poorly designed web interfaces, unsuccessfully attempting authorization and authentication, data transported without being encrypted, insecure network services, and communication between nodes that cannot support the same protocols. These are just some of the many vulnerabilities that all protocols can easily contain.

MQTT provides some methods of security including encryption, authentication, authorization, and server dependability.[9] MQTT relies on Transport Layer Security (TLS) to encrypt the data sent between clients and servers. TLS is used to prevent third parties from viewing data sent over the internet and therefore will provide a more secure environment for devices using MQTT protocol. Secondly, MQTT provides the users with the opportunity to enter a username and password as authentication. This ensures that the correct users are sending the data and allows the clients and servers to recognize potentially malicious nodes. MQTT uses mutual authentication in which both the clients and servers must provide their identity to each other. This way both nodes are able to confirm that they are communicating with the correct IP address. MQTT also takes into account what each user should have access to and what actions they should be performing. In other words, MQTT can authorize information sent and verify that it is both expected and safe to process. Lastly, due to the constrained environments that these devices exist, the server can easily run out of resources.[10] Therefore, developers using

[9]Andrew Schofield. *Security and MQTT*. Dec. 2014. URL: https://www-304.ibm.com/connections/blogs/aim/entry/security_and_mqtt?lang=en_us.

[10]Ibid.

MQTT are given the option to monitor and limit the number of connections that both client and servers have to ensure that it will not exceed the server limits.

Similarly CoAP provides a list of recommendations for developers in order to encourage security for the IoT devices. CoAP suggests developers use either Transport Layer Security or IPsec.[11] As explained above, TLS will ensure that no third party will intercept data sent between two nodes. Similarly, IPsec authenticates and encrypts messages sent between nodes and therefore, has a similar purpose to TLS.

However, it is proven that these methods of securing IoT devices isn't actually secure. First off, both TLS and IPsec were not developed to operate in constrained nodes. This can make certain procedures more challenging and therefore, not completely reliable. Secondly, it requires extra messaging between clients and servers and is more costly.[12] Therefore, more server resources are needed to send messages between clients and servers which can be an issue in constrained networks. Clients and servers must also set up the Security Association which is an establishment of security. If either of the communicating nodes changes IP addresses it must reconfigure and get a new Security Association. Not all nodes can support IPsec which will cause compatibility issues and ultimately safe security measures such as authentication and encryption would not be able to be implemented. Both protocols rely on other protocols to operate which are not ideal for constrained nodes.[13] Therefore, it is clear that although both MQTT and CoAP provides basic and sometimes successful security options, it is not always reliable depending on the availability and complexity of the resources communicating on the network.

As a developer, the ideal method of defending against these attacks is ensuring the devices and applications are secure throughout the development process. If security is a main con-

[11]Mahdi Aiash Thamer A. Alghamdi Aboubaker Lasebae. "Security Analysis of the Constrained Application Protocol in the Internet of Things". In: *School of Science and Technology, Middlesex University, London, UK* (Jan. 2014).

[12]Schofield, *Security and MQTT*, op. cit.

[13]Thamer A. Alghamdi, "Security Analysis of the Constrained Application Protocol in the Internet of Things", op. cit.

cern, then it will not fall through the cracks. Also, as discussed in ”Security Analysis of the Constrained Application Protocol in the Internet of Things”, new measures are being taken to develop new protocols in which all of the encryption and authentication capabilities are built in and designed specifically for constrained environments. The current protocol in progress is called Secured CoAP (S-CoAP).[14] The developers should take these new developments into consideration when choosing a protocol to use depending on the purpose of the devices, their environment, and the access to resources. It is crucial that the developers are aware of the potential vulnerabilities of CoAP and MQTT before and after deployment to ensure the resources used in their IoT technology are all compatible and support necessary protocols.

On the contrary, as a client, it is very important to understand the protocols used behind devices that may be purchased. Asking questions to the developers to ensure that the devices are secure especially if they can affect the well being of people. This includes any clients in the medical field, any mechanical or civil engineers that may be building structures for smart cities, anyone operating government devices that, if exposed, can result in private information being leaked, and more. A product should never be purchased and used if prior research has not been conducted to ensure no prior attacks have occurred.

Conclusion

IoT devices and applications are becoming more and more popular every day. These devices can be used in innovated and beneficial ways. For example: smart heart monitors that will text a doctor if something unusual is detected, smart pumps that will administer the correct amount of medication to patients in hospitals, smart thermostats that will reduce the amount of energy used to heat a building or house, and smart cars that can drive people around even if these people are not capable of driving themselves. However, these devices are only beneficial

[14]Ibid.

and usable if they are fully secure. If someone can hack into an application and change the readings of the smart heart monitors, or change the temperature detected for smart thermostats, then they cannot perform the given tasks. After analyzing CoAP and MQTT, two very popular protocols used in IoT, it is clear that there are still many security vulnerabilities that must be addressed before using these smart devices. Both developers and clients must be aware of these vulnerabilities and be confident that their product can protect against potential breaches.

References

Boyd, Bryan. *MQTT: A Practical Protocol for the Internet of Things*. Aug. 2014. URL: `http://www.slideshare.net/BryanBoyd/mqtt-austin-api`.

Mannino, Jack. *Implementing CoAP The Secure Way, Part I: Fundamentals*. May 2015. URL: `https://nvisium.com/blog/2015/05/27/implementing-coap-secure-way-part-i/`.

Montgomery, Bill. *The 10 Most Terrifying IoT Security Breaches you aren't aware of (so far)*. Sept. 2015. URL: `https://www.linkedin.com/pulse/10-most-terrifying-iot-security-breaches-so-far-you-arent-montgomery`.

Schofield, Andrew. *Security and MQTT*. Dec. 2014. URL: `https://www-304.ibm.com/connections/blogs/aim/entry/security_and_mqtt?lang=en_us`.

SolaceSystems. *MQTT Control Packet Format*. URL: `https://sftp.solacesystems.com/Portal_Docs/index.html#page/MQTT_Specification_Conformance/2_MQTT_Control_Packet_format.html`.

Thamer A. Alghamdi Aboubaker Lasebae, Mahdi Aiash. "Security Analysis of the Constrained Application Protocol in the Internet of Things". In: *School of Science and Technology, Middlesex University, London, UK* (Jan. 2014).

Z. Shelby K. Hartke, C. Bormann. "The Constrained Application Protocol (CoAP)". In: *Internet Engineering Task Force (IETF)* (2014).