



**UNIVERSITY OF  
ALBERTA**

**Detection and Prevention of Cybersecurity Risks of  
Cloud-connected Machinery in Industrial Internet of  
Things system (IIoT)**

**Capstone Project**

Presented by  
Shing Kit Lao

**University of Alberta  
Master of Science in Internetworking  
Edmonton, Canada**

Supervisor  
Sandeep Kaur

---

## Table of Contents

Glossary.....	3
Abstract.....	6
<b>1. Introduction to IIoT .....</b>	<b>7</b>
<b>1.1. IoT and IIoT .....</b>	<b>7</b>
<b>1.2. IT and OT .....</b>	<b>7</b>
<b>1.3. IIoT Architecture Model.....</b>	<b>9</b>
<b>1.4. Cloud Computing in IIoT .....</b>	<b>10</b>
<b>1.5. 5G in IIoT .....</b>	<b>10</b>
<b>2. Introduction to IoT Protocol.....</b>	<b>13</b>
<b>2.1. Cloud Computing and 5G Expose OT Risk to the Internet .....</b>	<b>13</b>
<b>2.2. IoT protocol.....</b>	<b>14</b>
<b>2.3. The Importance of MQTT protocol .....</b>	<b>18</b>
<b>3. Overview of MQTT .....</b>	<b>21</b>
<b>3.1. Publish-Subscribe Model .....</b>	<b>21</b>
<b>3.2. Message Format and Type.....</b>	<b>22</b>
<b>3.3. Topic .....</b>	<b>23</b>
<b>3.4. Quality of Service .....</b>	<b>24</b>
<b>3.5. Security Feature .....</b>	<b>24</b>
<b>4. Literature Review of Securing MQTT .....</b>	<b>25</b>
<b>4.1. Enhanced Authentication and Authorization .....</b>	<b>25</b>
<b>4.2. Enhanced Protocol .....</b>	<b>26</b>
<b>4.3. Enhanced Network Infrastructure .....</b>	<b>27</b>
<b>4.4. Additional Intrusion Detection System (IDS) .....</b>	<b>27</b>
<b>5. Analysis of MQTT Vulnerabilities .....</b>	<b>28</b>
<b>5.1. Reported Vulnerabilities in NVD database.....</b>	<b>28</b>
<b>5.2. Potential Attack Surface according to Shodan Report .....</b>	<b>31</b>
<b>6. Proposed Mechanism .....</b>	<b>33</b>
<b>7. Results Analysis and Elaboration.....</b>	<b>36</b>
<b>7.1. Testing Environment Setup .....</b>	<b>36</b>
<b>7.2. Areas of the Proposed Mechanism Not Being Tested.....</b>	<b>37</b>
<b>7.3. Experiment 1: Testing Normal Case as Control Test.....</b>	<b>37</b>

<b>7.4.</b>	<b>Experiment 2: Testing Field Value Validation .....</b>	<b>38</b>
<b>7.5.</b>	<b>Experiment 3: Testing Required Field Validation.....</b>	<b>38</b>
<b>7.6.</b>	<b>Experiment 4: Testing Logical Error Validation .....</b>	<b>38</b>
<b>7.7.</b>	<b>Experiment 5: Testing Wildcard Topic Subscription .....</b>	<b>39</b>
<b>7.8.</b>	<b>Experiment 6: Testing Invalid Sequence Number .....</b>	<b>39</b>
<b>7.9.</b>	<b>Testing Result Summary .....</b>	<b>40</b>
<b>8.</b>	<b>Conclusion and future work.....</b>	<b>41</b>
<b>A.</b>	<b>Reference.....</b>	<b>42</b>

## Glossary

Term	Definition
Internet of Things (IoT)	The interconnection of physical devices, which are equipped with sensor to collect data, and management platforms to perform analysis on the data collected. <sup>[1][2]</sup>
Industrial Internet of Things (IIoT)	Also known as Industry 4.0. It is the application of IoT in industrial usage including the manufacturing industry, gas, and oil plant industry and utilities industry. <sup>[5]</sup>
Information Technology (IT)	One of the two components of IIoT. IT networks deal with the flow of data or information across an organization. <sup>[6]</sup>
Operational technology (OT)	One of the two components of IIoT. OT networks manage the operation and control of physical processes and machinery. <sup>[6]</sup>
Industrial Control Systems (ICSs)	A major component of OT. The systems are used to monitor and control industrial processes. ICSs are mission-critical with high availability. <sup>[7]</sup>
Supervisory Control and Data Acquisition (SCADA) systems	The systems are used to manage ICSs. They provide graphical interface for operators to observe system status, receive alarms and adjust the process. <sup>[7]</sup>
Human Machine Interfaces (HMIs)	The graphical interface between human and machines used in SCADAs. <sup>[7]</sup>
Distributed Control Systems (DCSs)	A specially designed automated control system that consists of geographically distributed control elements over the plant or control area. <sup>[43]</sup>
Purdue Enterprise Reference Architecture (PERA)	Also known as the Purdue Model. It models the best practice in segregating industrial control system (ICS) in OT network from business system in IT networks. <sup>[5][9]</sup>
Industrial Demilitarized Zones (DMZs)	The segment and network used to segregate the IT and OT networks in the PERA model. <sup>[5][9]</sup>
Cloud Computing	The delivery of computing services – including servers, storage, databases, networking, software, analytics and intelligence – over the Internet (“the cloud”) <sup>[41]</sup>
Big Data	A collection of large and complex data sets that are difficult to store and process using traditional database management applications. <sup>[42]</sup>
Machine Learning	An automated data processing and decision-making algorithms designed to improve at every stage of their assigned task based on their experience. <sup>[42]</sup>

Artificial Intelligence (AI)	Artificial intelligence leverages computers and machines to mimic the problem-solving and decision-making capabilities of the human mind. [44]
5G	It is the 5th generation mobile network. It is designed to connect virtually everyone and everything together including machines, objects, and devices. [45]
Enhanced ultra-reliable low-latency communication (eURLLC)	One of the three major 5G services supports low-latency transmissions of small payloads with very high reliability from a limited set of terminals. [46]
Massive machine type communications (mMTC)	One of the three major 5G services supports a massive number of Internet of Things (IoT) devices, which are only sporadically active and send small data payloads. [46]
Enhanced mobile broadband (eMBB)	One of the three major 5G services supports stable connections with very high peak data rates, as well as moderate rates for cell-edge users. [46]
IoT Protocol	It is designed for creating intelligent environment that will automate the manufacturing process and change the way human interacting with machines. [14][15]
Transport Layer Security (TLS)	It is an Internet Engineering Task Force (IETF) standard protocol to provide privacy and data integrity between two communicating applications. It is run on top of reliable transport protocol such as TCP. [47]
Datagram Transport Layer Security (DTLS)	It is an Internet Engineering Task Force (IETF) standard protocol to provides communications privacy for datagram protocols. It is based on the TLS protocol and provides equivalent security guarantees. [48]
Message Queuing Telemetry Transport Protocol (MQTT)	An open protocol designed to support messaging transport from remote locations/devices involving small code footprints, low power, low bandwidth, high-cost connections, high latency, variable availability, and negotiated delivery guarantees. [20]
Constrained Application Protocol (CoAP)	It is designed for resource-constrained node and resource-constrained network running over User Data Protocol (UDP). [20][21][22]
Advanced Message Queuing Protocol (AMQP)	An open standard protocol for message-oriented middleware. It provides flow controlled, message-oriented communication with message-delivery guarantees, and authentication / encryption based on TLS. [23]
Extensible Messaging	An open-source protocol designed for human-to-

and Presence Protocol (XMPP)	human instance messaging among multiple parties in a structured but extensible XML data format. [25]
Bluetooth Low Energy (BLE)	A new version of Bluetooth optimized for IoT connections. The power consumption of BLE is even less than standard Bluetooth. [16]
LoRa	A wireless radio technology for wireless LAN network patented by Semtech. LoRa is the main non-cellular physical layer protocol operating in the unlicensed spectrum. [26]
LoRaWAN	An open-source media access control layer protocol leveraging the physical LoRa protocol. [26]
Zigbee	A suite of high-level communication protocols based on IEEE 802.15.4 specification. It operates in low power, low bandwidth, and short-range wireless network designed for IoT. [16][27]
National Vulnerability Database (NVD)	A renowned platform for cybersecurity vulnerability reporting.
Common Vulnerabilities and Exposures (CVE)	A renowned platform for cybersecurity vulnerability reporting.
Rivest–Shamir–Adleman (RSA)	It is an asymmetric public-key cryptosystem. Its security lies with the integer factorization problem. [49]
Lightweight Elliptic Curve (ECC)	It is an asymmetric public-key cryptosystem. Its security lies with the elliptic curve discrete logarithm problem. It requires comparatively less or smaller parameters for encryption and decryption than RSA, but with equivalent levels of security. [49]
Intrusion Detection System (IDS)	It is a system that monitors network traffic for suspicious activity and alerts when such activity is discovered. [50]
National Cyber Security Centre (NCSC)	An UK government body provide effective incident response to minimize harm to the UK, help with recovery, and learn lessons for the future. [40]

## Abstract

Internet of Things (IoT) is the interconnection of physical devices, which are equipped with sensor to collect data, and management platforms to perform analysis on the data collected. IoT helps people to make decision or further control on other devices automatically. <sup>[1][2]</sup> Industrial Internet of Things (IIoT) is the application of IoT in various industries such as manufacturing, energy plant or water supply.

Nowadays, 5G network allows massive physical devices to interconnect effectively. Cloud computing provides reliable and cost-effective computing power. Big data/Artificial intelligence (AI) can interpret the massive data collected efficiently. With all these evolving technologies, IIoT is widely invested and adopted by various industries in recent years to reduce operation cost, improve reliability and increase profit. However, the interoperability also raises new cybersecurity concerns. Unlike IT network, the consequence of cyberattack on IIoT systems can be disastrous and fatal.

By leveraging cloud computing and 5G in the IIoT applications, there are many protocols design for machine communication only are now exposed to the internet. Some of the machine-to-machine network protocols are prevalent as IoT protocols also. But those protocols are generally designed with few or even no security features. It relies on the underlying infrastructure for protection.

This project will discuss those IoT protocols in general first. Then we will research the potential cybersecurity risks, detection techniques, and preventive measures of the most widely used IoT protocol - Message Queuing Telemetry Transport Protocol (MQTT).

## 1. Introduction to IIoT

### 1.1. IoT and IIoT

The Internet of Things (**IoT**) refers to the physical devices embedded with sensors and computing ability connecting to the Internet to collect and exchange data. According to forecast from Statista, the number of connected IoT devices will dramatically increase up to 75 billion by 2025, which is three times more than 2019. <sup>[3]</sup> Owing to the advancement in tiny low-cost computer chips and prevalence of wireless network, almost all physical devices can be turned into IoT nowadays. It can be as common as a light bulb in a house or as complicated as driverless aeroplane. <sup>[4]</sup>

The industrial Internet of Things (**IIoT**), also known as **Industry 4.0**, is the application of IoT in industrial usage including the manufacturing industry, gas, and oil plant industry and utilities industry. Traditional industrial machineries like factory's machines, aircraft's engines, oil and gas plant's drilling rigs can all be examples of IIoT nowadays. IIoT, as distinct from IoT, converges information technology (**IT**) and operational technology (**OT**). <sup>[5]</sup>

By leveraging the big data to analyze numerous data collected from IIoT devices, IIoT helps industry improving the manufacturing efficiency, streamlining production line, minimizing human risks in monitoring gas and oil fields, and many other numerous benefits. It is another major industrial revolution into the digital world and hence called Industry 4.0.

However, connecting IIoT devices to communication networks also exposes industrial devices to significant cybersecurity risks. Besides, in order to get the product time-to-market, many IoT manufacturers tend to think little of the security. There is a vast amount of insecure IoT devices flooding in the market. <sup>[2]</sup>

### 1.2. IT and OT

It is crucial to distinguish between IT and OT before understanding the IIoT cybersecurity risk in-depth as different attack surfaces are exposed from them.

- **Information Technology (IT):** IT networks deal with the flow of data or information across an organization. <sup>[6]</sup>
- **Operational Technology (OT):** OT networks manage the operation and control of physical processes and machinery. <sup>[6]</sup>

There are endless arguments in the integration of IT and OT. Bridging the gap between them will improve efficiency, especially after the rise of technology like cloud computing



and big data analysis. On the other hand, the integration will expose OT to the cyberattack from the internet as OT is not expected for internetworking originally by design. <sup>[6]</sup>

OT systems are used to manage industrial operations. They are usually built with specific industrial purpose. OT includes industrial control systems (ICSs), for example, human machine interfaces (HMIs), supervisory control and data acquisition (SCADA) systems, and distributed control systems (DCSs). <sup>[7]</sup>

The operating environment, security priority and resource requirement are completely different between IT and OT systems. Industrial machines and systems typically have a long lifespan which can run for more than 20-50 years. The operating processes are quite stable that the owners are more reluctant to change them even during digital transformation journey. The result of failure in OT networks can be fatal. Unlike IT networks that there is normally financial or personal data loss if any failure, the malfunction of OT networks could shut down the industrial process or even oil and gas plant. It can affect the stability of the city and arise threat to human lives. Traditionally, OT is designed with system reliability and human safety in priority while IT concerns the data and information security the most. <sup>[8]</sup>

Besides, IT systems are usually maintained by IT professionals, but OT systems are usually operated by the workers without IT background. OT systems are supposed to operate in closed private networks. OT systems can also run-in extreme conditions like high and low temperature, with nuclear exposure or deep into water that may not always be human reachable. So, the OT hardware and software are less frequent to be upgraded and patched until deficiency is found in current system. In general, the communication protocols used in OT network are lightweight and ready to be run in low battery, low computing power and high resource constrained conditions. So, they are lack of security feature like data encryption, connection authentication or instruction detection that are commonly and widely used in IT networks. <sup>[8]</sup>

OT networks are protected by segregation with IT network and without direct internet connection. But this OT design assumption is no longer valid after integration with cloud with IoT aware devices. More and more OT systems expose to internet for IIoT and increase their chance of being cyberattacked. <sup>[8]</sup>

IIoT inherits all cybersecurity risks of IT. So, the protections of ordinary IT networks also apply to IT in IIoT. OT are those industrial systems not designed for internet connection. What the most important when considering cybersecurity related to IIoT is to find a way to transfer information between IT and OT networks. At the same time, segregating control and access to OT networks needs to be maintained to keep secure control of the technology controlled by those networks. <sup>[8]</sup>

### 1.3. IIoT Architecture Model

The Purdue Enterprise Reference Architecture (PERA), also known as the Purdue Model, was created at the Purdue University in the early 1990's. It models the best practice in segregating industrial control system (ICS) in OT network from business system in IT networks. Six "Levels" are defined with each representing a subset of systems. In general, going down from level 5 to level 0, the devices have more control and access to critical process but fewer security features. The lower-level systems rely on upper-level network and architecture for protection. <sup>[5][9]</sup>

The PERA is not designed as cybersecurity or IIoT reference model. It is originally designed to separate the systems in IT and OT networks. It is still treated as common conceptual framework nowadays to allow others to understand how security components can be introduced to the IT/OT network. <sup>[5]</sup>

ICS/OT systems are defined in Levels 0-3. Enterprise/IT systems are defined in Levels 4-5. The communication between the OT and IT boundaries are via the firewall defined in the demilitarized zones (DMZs). To protect the ICS system and OT networks, the connection to the IT is minimal or even eliminated traditionally. <sup>[5]</sup>

But in IIoT usage nowadays, the data flows are no longer in the same manner as Purdue Model. The smart sensors and controllers in Level 0-1 can now bypass firewall and communicate with cloud via edge computing. It exposes ICS system in OT network to unexpected cyber risks. <sup>[9]</sup>

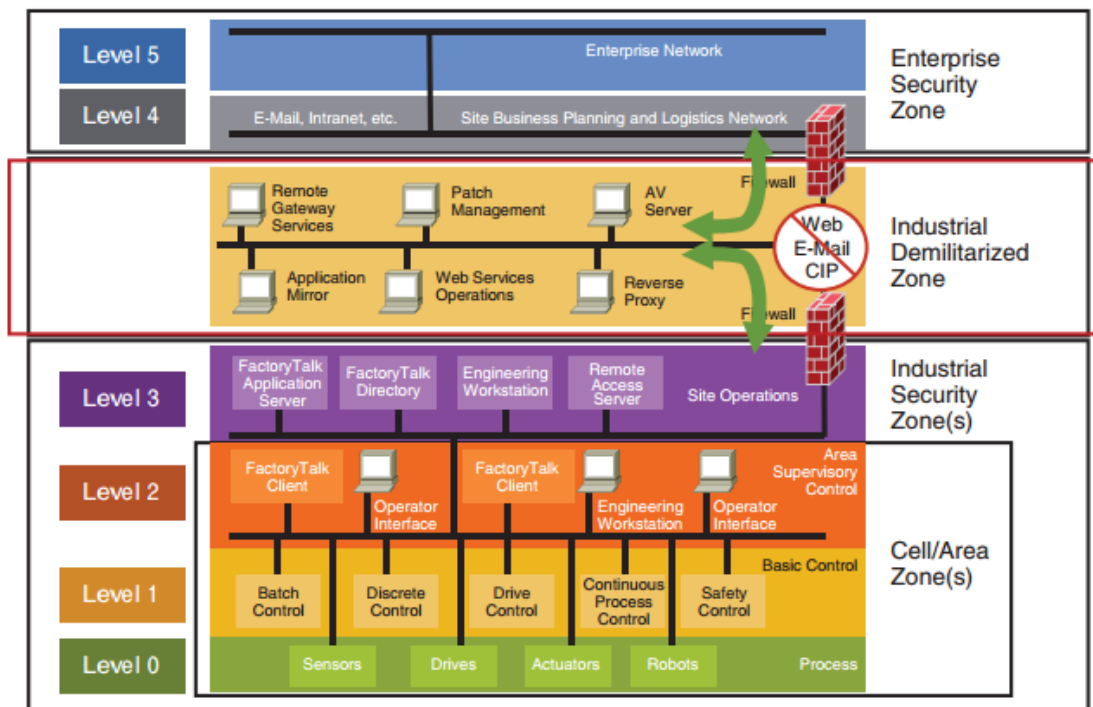


Figure 1. The Purdue Enterprise Reference Architecture (PERA) <sup>[9][10]</sup>

## 1.4. Cloud Computing in IIoT

With the improvement in network bandwidth and better chip design, every machine can be equipped as IIoT devices. IIoT needs more storage and computing power to analyze the data. Cloud computing complements the IIoT advancement with unlimited computing resources. It collaborates with all IoT devices to provide real-time control and data monitoring. <sup>[11]</sup>

Another modern technology often bundled with cloud computing and IIoT is big data and machine learning. IIoT device is to collect data. Cloud computing provides storage and resource. It is the big data and machine learning technologies to analyze the vast amount of data in real time and provide corresponding decision making. Big data leverages the IIoT input to show hidden correlations, unidentified patterns and expose novel solutions from the data set. It drives the efficiency improvement, process automation and production line streamlining in the industry. <sup>[11]</sup>

## 1.5. 5G in IIoT

5G is an indispensable to the future of IIoT. It is not simply a network advancement with faster bandwidth. The requirement of IIoT and cyber-physical system are well incorporated in the design of 5G since early specification design phase. Figure 2 shows some of the 5G design specific for IIoT requirements. <sup>[12]</sup>




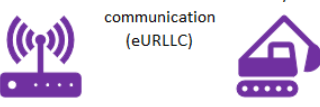


	Design	Feature
<b>5G</b> 	 Private 5G Network	<ul style="list-style-type: none"> <li>• Unique network ID</li> <li>• Integrated and independent architectures</li> <li>• Seamless fallback to public network</li> </ul>
	 Spectrum	<ul style="list-style-type: none"> <li>• Licensed spectrum</li> <li>• Shared spectrum</li> <li>• Unlicensed spectrum</li> </ul>
	 Enhanced ultra-reliable low-latency communication (eURLLC)	<ul style="list-style-type: none"> <li>• Low latency</li> <li>• Ultra-reliability</li> <li>• CoMP (coordinated multipoint transmission) multi-TRP (Transmission and Reception Point)</li> <li>• Service Multiplexing</li> <li>• Enhanced Mobility</li> </ul>
	 Time Sensitive Networking TSN	<ul style="list-style-type: none"> <li>• Ethernet over 5G</li> <li>• Deterministic networking</li> <li>• Device time sync</li> </ul>
	 Positioning	<ul style="list-style-type: none"> <li>• Network and device based</li> <li>• Industrial IoT requirements</li> </ul>

Figure 2. 5G design for IIoT requirements <sup>[12]</sup>

With the rapid adoption of IIoT solutions in various industries, there are different requirements on the network capabilities, not just the latency. For example, the self-driving vehicles require large coverage area and high reliability to monitor real time traffic conditions. The utilities plant requires low power consumption and large capacities to connects vast amount of IIoT devices working in extreme conditions. There are three key 5G network protocols with distinct characteristics designed for different IoT applications. They are: <sup>[13]</sup>

- Enhanced ultra-reliable low-latency communication (eURLLC)
- Massive machine type communications (mMTC)
- Enhanced mobile broadband (eMBB)

The summary of their features and applications are shown in Figure 3.

Protocol	Key Features	Applications
Enhanced ultra-reliable low-latency communication (eURLLC)	<ul style="list-style-type: none"> <li>• High reliability</li> <li>• Low latency</li> <li>• High coverage</li> </ul>	<ul style="list-style-type: none"> <li>• Self driving cars</li> <li>• Unmanned aerial vehicle network</li> <li>• Mission critical applications</li> </ul>
Massive machine type communications (mMTC)	<ul style="list-style-type: none"> <li>• Massive number of connected device</li> <li>• Long device battery life</li> <li>• High coverage</li> </ul>	<ul style="list-style-type: none"> <li>• Industrial Automation</li> <li>• Smart Cities</li> <li>• Smart Homes</li> </ul>
Enhanced mobile broadband (eMBB)	<ul style="list-style-type: none"> <li>• High bandwidth &amp; data rate</li> <li>• Low latency</li> <li>• High spectral efficiency and throughput</li> </ul>	<ul style="list-style-type: none"> <li>• 4K HD Videos</li> <li>• Augmented / Virtual Reality</li> <li>• Distance learning and education</li> </ul>

Figure 3. 5G network protocols for different applications <sup>[13]</sup>

5G is the foundation for the industry successful transformation to IIoT. It provides much greater data transfer rate, extremely low latency, and superior reliability that previous network generations, like 3G and 4G, cannot provide. With the prevalence of Industry 4.0, there will be exponential growth in IIoT device adoption in all industries and incredibly increase in amount of data generated. Industry will require more rapid and capable analytical tools with machine learning to drive their business growth and system stability. Here is the illustration of 5G technologies usage for Industry 4.0 in a manufacturing factory.

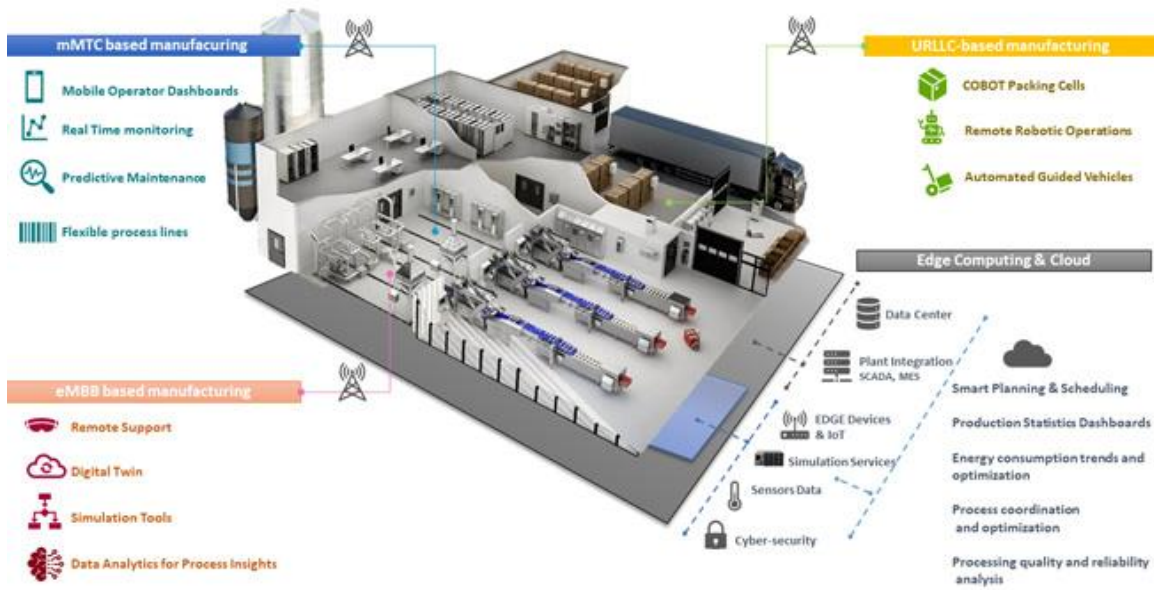


Figure 4: Diagram of the architecture of 5G-based IIoT for Industry 4.0 [13]

This paper is organized as follows. An overview of IoT protocol is described in section 2. An overview of Message Queuing Telemetry Transport Protocol (MQTT) protocol is described in Section 3. Literature review in securing MQTT is provided in section 4. The existing MQTT vulnerabilities is analyzed in section 5. Proposed mechanism to secure MQTT protocol is described in Section 6. The testing results are discussed in Section 7. The paper is concluded in Section 8.

## 2. Introduction to IoT Protocol

When talking about IoT protocol, some may use the term M2M interchangeably. M2M stands for machine-to-machine communication. Both protocols allow machines to communicate, collect and exchange data. They enable machines to perform tasks without human intervention. <sup>[14]</sup>

But M2M and IoT are not synonymous. M2M is designed purely for machine communications well before emergence of IoT. IoT is designed for creating intelligent environment that will automate the manufacturing process and change the way human interacting with machines. IoT usually involves communications among sensors, cloud server, artificial intelligence, machine learning as well as user interface. M2M can be treated as subset of IoT as some M2M will help the creation of intelligence environment. At the same time, M2M protocols are more vulnerable to cyber-attack as they are not designed for internet communications originally. <sup>[14][15]</sup>

### 2.1. Cloud Computing and 5G Expose OT Risk to the Internet

As discussed in section 1.3, the conventional Purdue Model separates IT and OT network clearly. The OT network are supposedly well protected within private network. So, many devices within the OT network are not built with security feature in design. However, with the advance in cloud computing as well as development of 5G in recent years, more and more IIoT devices originally in the OT network are now connected to the internet directly. It is in the sake of more efficient big data and artificial intelligence analysis. They are usually connected via the internet with larger computing capacity, faster network and supporting vast number of IIoT devices connected simultaneously.

On the other hand, the security of IIoT device does not advance at the same pace. Most of them are industrial machines embedded with sensors or controllers connecting to the internet. They are more reluctant to be upgraded as there is much larger operational risk. There can be physical constraints that the modern security feature like TLS encryption cannot be applied. It can be the extreme condition the devices operating that the device cannot be changed or upgraded easily. It can also be low-battery requirement limiting the computation power that cannot support complicated encryption.

As shown in Figure 5, in last decade, there are increasing number of state-sponsored cyberattacks against OT network and devices. These state-sponsored activities do not attack for commercial intellectual property (IP) or financial benefits. The attacks usually target at those utilities factory or oil and gas plant to disrupt the operation and daily lives of another country. The result can be disastrous rather than only financial lost like normal IT network attacks.

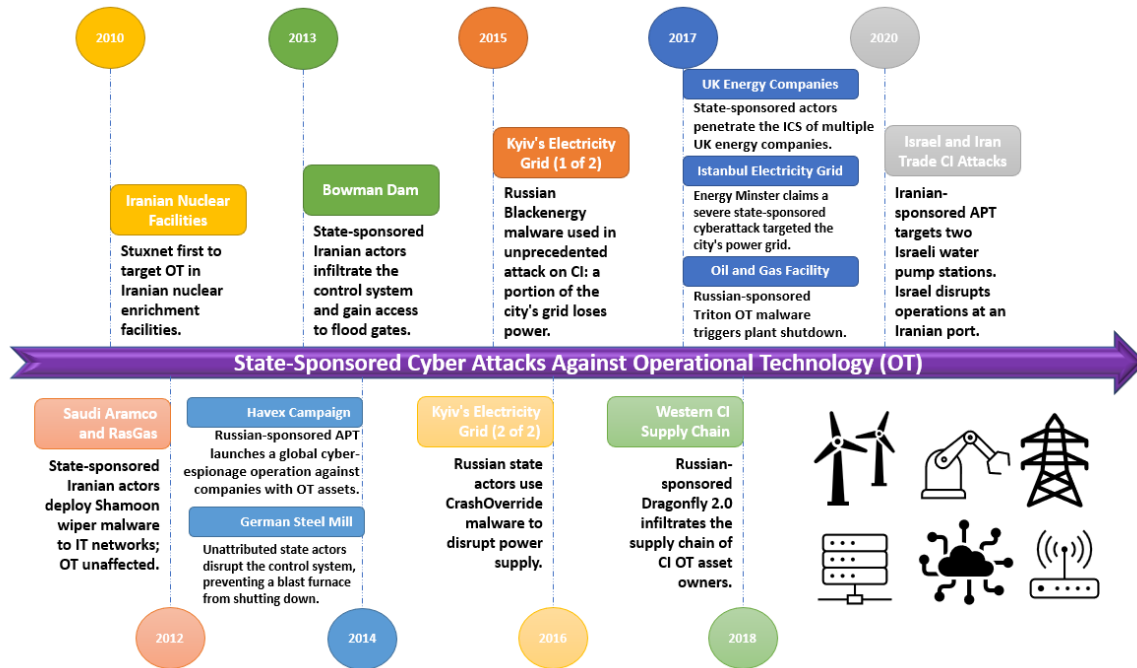


Figure 5. State-Sponsored Cyber Attacks Against Operational Technology (OT) in last decade [8]

There is much research in protecting IT network, cloud network and 5G network as they have wide usage in non IIoT aspect. Comparatively, the research in protecting OT network is not adequate when they are specific to IIoT concerns. Besides, many OT machines communicated in specific protocols with specific purpose. There is lack of standard protocol in IIoT. The heterogenous nature of OT network is also a major reason why it is harder to be protected than IT network. The consequence of cyber-attack on OT is more fatal than IT network but OT network is weaker in security protection in nature. More research in protecting OT is required in priority.

## 2.2. IIoT protocol

There are various IIoT protocols enabling device-to-device, machine-to-machine, device/machine-to-gateway, device/machine-to-cloud communication, or their combinations. [16] Each protocol has its own usage and capability. Some are suitable for low power consumption environment. Some are more scalable into vast amount of IIoT devices communication. With reference to a survey, the IIoT protocols can be classified into four broad categories, namely: application protocols, service discovery protocols, infrastructure protocols and other influential protocols. [17][18] In Figure 6, some major IIoT protocols are shown with reference to the TCP/IP model that we are familiar in IT network. They include both open-source and proprietary protocols. In this section, we will go through major application protocols and infrastructure protocols.

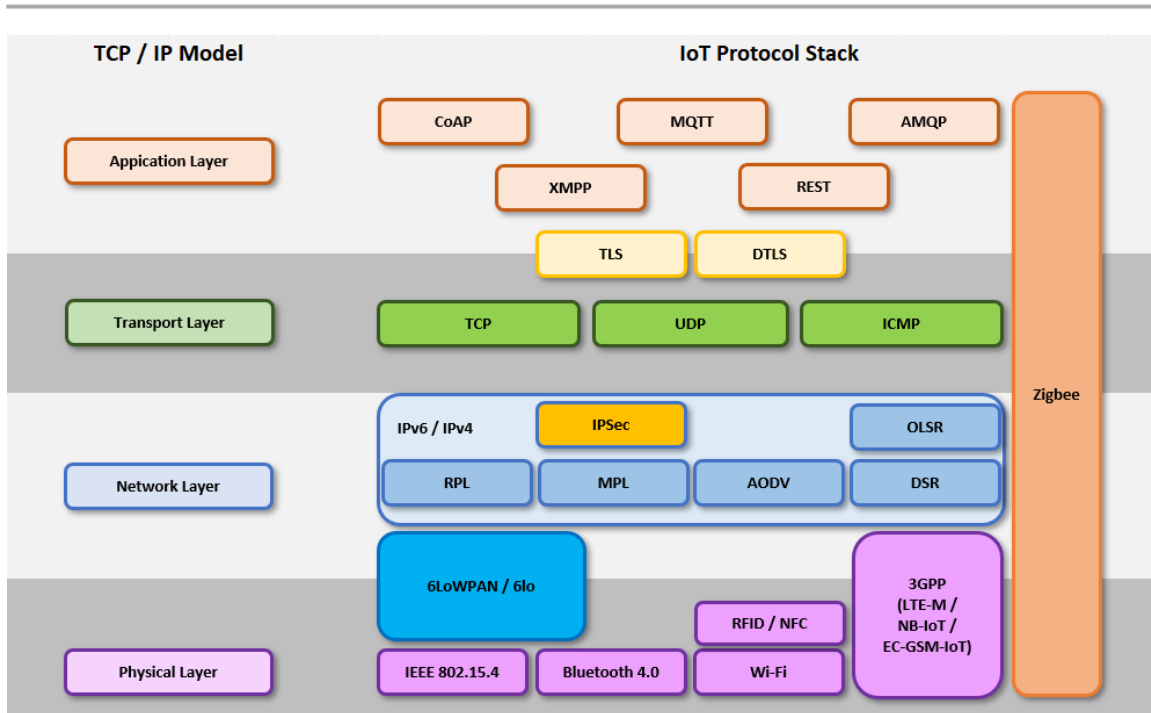


Figure 6. IoT Protocol Stack <sup>[18][19]</sup>

### i) Application Protocols

**MQTT** – Message Queuing Telemetry Transport Protocol is an open protocol governed by OASIS. MQTT was originally a proprietary protocol developed by IBM in 1999. MQTT is designed to support messaging transport from remote locations/devices involving small code footprints (e.g., 8-bit, 256KB ram controllers), low power, low bandwidth, high-cost connections, high latency, variable availability, and negotiated delivery guarantees. Its lightweight feature running in publish/subscription architecture on Transmission Control Protocol (TCP) / Internet Protocol (IP) protocol make it one of the leading M2M/IoT protocol nowadays. <sup>[20]</sup>

To make MQTT lightweight and suitable to run in resource-constraint devices, MQTT communicates in plaintext by default. The encryption depends on the underlying transport layer by running Transport Layer Security (TLS) over TCP protocol or on the application layer with payload encryption implemented by the client application themselves if required. By doing so, more computational power is required. These optional security features depend on other layer as well as user’s decision is the cause of the highly vulnerable to cyberattacks against MQTT.

**CoAP** – Constrained Application Protocol was first published by The IETF Constrained RESTful Environments working group in 2013. It is specially designed for resource-



constrained node and resource-constrained network running over User Datagram Protocol (UDP). As TLS cannot be run on UDP, to secure the communication, Datagram Transport Layer Security (DTLS) is used instead. <sup>[20][21][22]</sup>

CoAP operates in request/response architecture between application nodes. It is designed to easily integrate with HTTP following REST-architecture while meeting specialized requirements such as multicast support, very low overhead, and simplicity. CoAP is another leading M2M/IoT protocol used in the industry. <sup>[22]</sup>

**AMQP** – Advanced Message Queuing Protocol is an open standard protocol for message-oriented middleware. The specification defines features of AMQP as message oriented, queuing, routing (including point-to-point and publish-and-subscribe), reliability and security. As a protocol originated by JPMorgan Chase in 2003 and later governed by OASIS, the protocol provides flow controlled, message-oriented communication with message-delivery guarantees such as at-most-once, at-least-once, and exactly-once and authentication / encryption based on TLS. It assumes running on reliable transport layer protocol such as TCP. <sup>[23]</sup>

Although AMQP features are powerful, the protocol is heavy to implement. It may not be suitable for some IIoT devices with resource-constraint and limited network bandwidth. Besides, those complete guaranteed delivery feature may not be required in the IIoT applications. This make AMQP not as widely adopted as MQTT or CoAP in IIoT usage. <sup>[24]</sup>

**XMPP** – Extensible Messaging and Presence Protocol originated from Jabber open-source community in 1999. This open-source protocol is currently supported by the XMPP Standards Foundation. It is designed for human-to-human instance messaging among multiple parties in a structured but extensible XML data format. It is designed for lightweight middleware and is widely adopted as a communication protocol. <sup>[25]</sup>

Its strength in addressing and scalability capabilities make it stand out for consumer oriented IoT applications such as smart appliances. It is now used for M2M communications for routing XML data. However, its deficiencies in Quality of Service and end-to-end encryption make it less prevalence in IIoT applications. <sup>[23]</sup>

## ii) Infrastructure Protocols

**Bluetooth and BLE** – Originally developed by Ericsson in 1990's, Bluetooth is now commonly adopted as “Personal Area Network (PAN)” protocol. It covers a small geographical region (<10M) transmitted in ultra-high frequency (2.4GHz) radio waves. This low power connectivity protocol makes it suitable for IoT applications. <sup>[16]</sup>

Bluetooth Low Energy, also known as Bluetooth LE or BLE, is a new version of Bluetooth optimized for IoT connections. The power consumption of BLE is even less than standard Bluetooth. It is therefore suitable in various IoT usages like smart home control and in-store tracking. <sup>[16]</sup>

**Wi-Fi** – Apart from its widely usage at home, commercial buildings, shopping malls or even different hotspots in the city, Wi-Fi is also a commonly used IoT protocol. It is strong in providing fast and large data transmission within short to medium range geographic area. It also provides different standards working on different frequencies and providing different deployment options depends on situation.

The major shortcoming in IIoT application is that Wi-Fi is very power consuming. Besides, it is not scalable and short-range coverage. All these drawbacks limit Wi-Fi pervasiveness in IIoT usage which usually requires low power consumption, high scalability to connect thousands of IoT device and wide coverage more than just a building.

**LoRa and LoRaWAN** – LoRa is a wireless radio technology for wireless LAN network. It is patented by Semtech. LoRa is the main non-cellular physical layer protocol operating in the unlicensed spectrum. It is commonly used in Low Power Wide Area Network (LAWAN) with operating frequencies lower than that of cellular network. LoRaWAN is a media access control layer protocol leveraging the physical LoRa protocol. Unlike LoRa, LoRaWAN is an open-source protocol developed by the LoRa Alliance with Semtech as one of the driving forces behind. <sup>[26]</sup>

The long range, low power and low bandwidth features of these protocols drive their rapid growth of adoption in the IIoT wireless connectivity. They are best fit for applications that are non-critical, low traffic, mainly uplink traffic involved, battery powered, and low-cost sensor required. It can be applied in IIoT that requires longer distance connectivity like mining, farming, and manufacturing industry. It is an alternative option for shorter distance coverage solution like Bluetooth and Wi-Fi. It is also an alternative for more expensive solution like cellular. <sup>[26]</sup>

**Cellular** – Cellular is one of the most prevalence protocols for IoT application for its strong in signal communication over a very large area. More and more fast and reliable standards are implemented by the telecommunication service providers from legacy 2G, 3G to now 4G/LTE and 5G. Especially for 5G, the standard is developed with IoT/IIoT usage in design as discussed in section 1.5. <sup>[16]</sup>

The major drawbacks are that it is costly to build the network infrastructure as well as higher power consumptions. But its long-distance capability is replaceable that there are still many organizations investing in its development and deployment. <sup>[16]</sup>

**Zigbee** – Zigbee is an open standard governed by the Zigbee Alliance. It has a suite of high-level communication protocols based on IEEE 802.15.4 specification. It operates in low power, low bandwidth, and short-range wireless network. It is designed for IoT network with long battery life devices and up to 65,000 nodes. It works in unlicensed bands including 2.4GHz, 900MHz and 868MHz. Zigbee supports multiple network topologies including point-to-point, point-to-multipoint and mesh networks. It has a longer range but lower data rate than BLE. Its IoT applications includes traffic management systems or industrial equipment transferring in low wireless data rate. <sup>[16][27]</sup>

### 2.3. The Importance of MQTT protocol

Although there are different control and measures in protecting the OT network, there are still many OT networks exposed to the public internet with minimal protection. The success of cybersecurity implementation depends on people, process, and technology. Many cybersecurity issues are due to misconfiguration or the implementation faults of different vendors rather than the protocol or system design itself.

Shodan (<https://www.shodan.io>) is a famous search engine to navigate all internet-connected devices. According to Shodan searching results, MQTT and CoAP are the most prevalent IoT protocols used. The default MQTT broker port is 1883 which is an unsecure channel without the use of TLS. The Shodan search result can be further filtered by inputting “MQTT” “Port:1883” in the search criteria. As shown in Figure 7, there are 312 thousand of unsecured MQTT brokers connecting to the internet around the world as of 5 February 2022. Similarly, searching for the default unsecured port 5683 for CoAP protocol in Shodan search engine, there are 332 thousand of unsecured CoAP internet-connected devices as shown in Figure 8. These devices are more vulnerable to be attacked. The number of internet-connected devices for other IoT protocols like AMQP are significantly fewer and is not shown in this report.

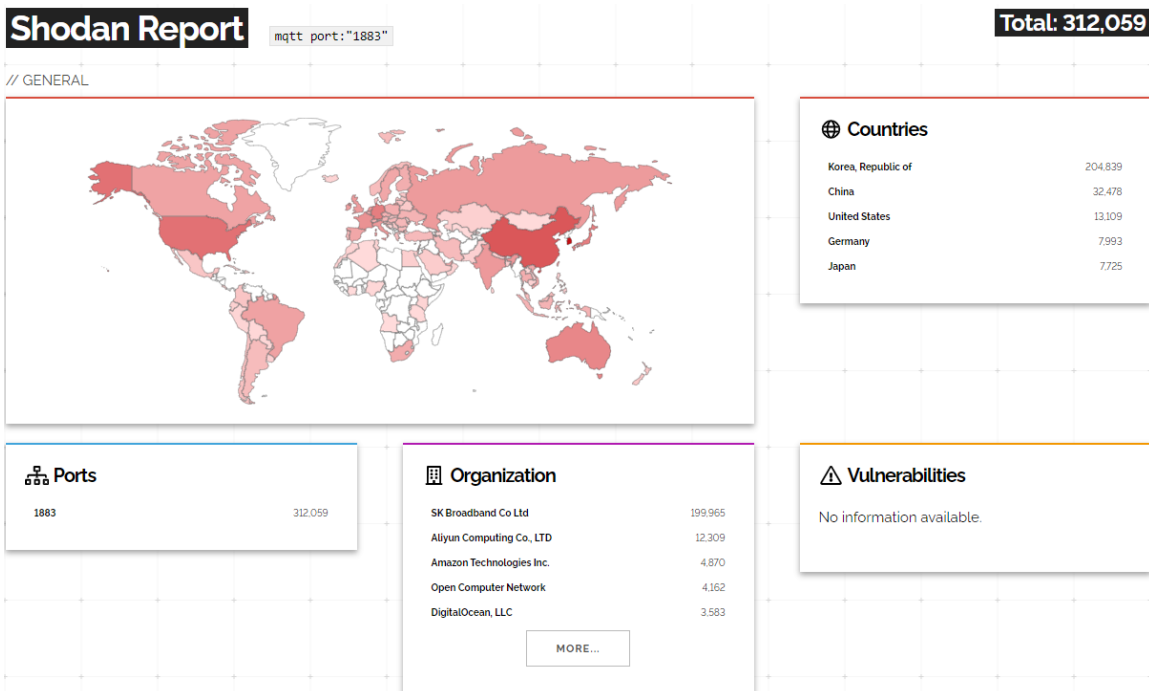


Figure 7. Shodan report of internet devices using MQTT protocol on default port 1883 taken on 5 February 2022

<https://www.shodan.io/search/report?query=MQTT+port%3A%221883%22>

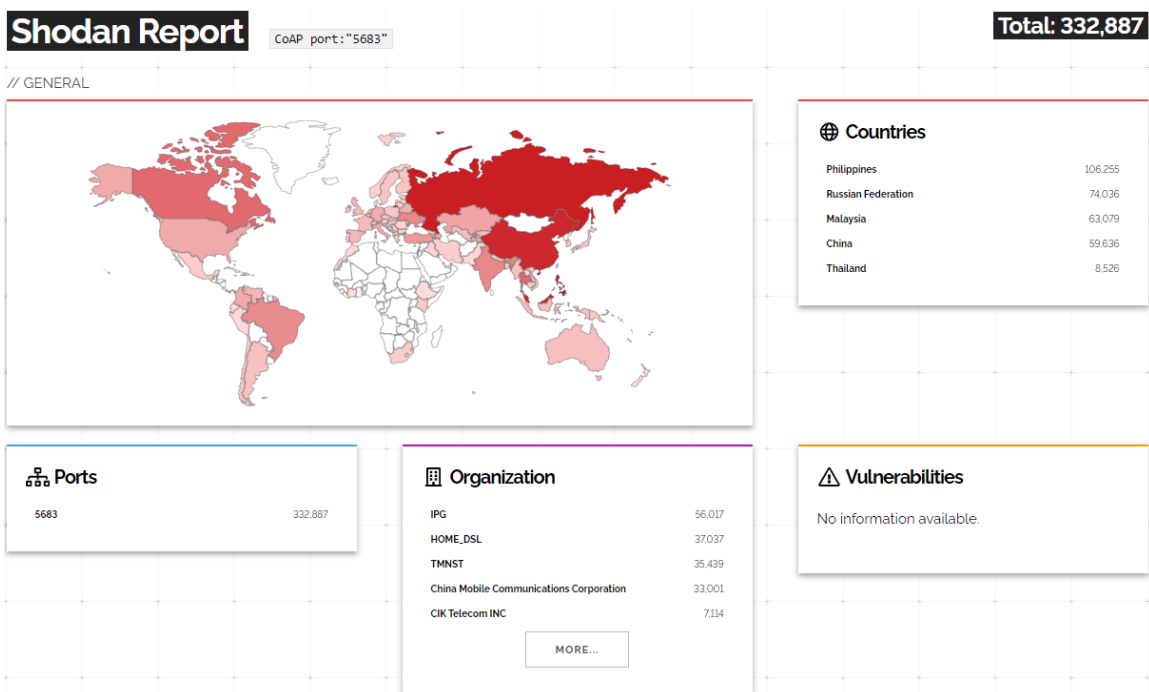


Figure 8. Shodan report of internet devices using CoAP protocol on default port 5683 taken on 5 February 2022

<https://www.shodan.io/search/report?query=CoAP+port%3A%225683%22>

Further analysis in these two M2M protocols was performed by accessing the reported protocol vulnerabilities in the National Vulnerability Database (NVD, <https://nvd.nist.gov>), and Common Vulnerabilities and Exposures (CVE, <https://cve.mitre.org>) databases accessed on 5 February 2022. Both databases are renowned platforms for cybersecurity vulnerability reporting.

There are only 26 CoAP related reported vulnerabilities but 94 MQTT related reported vulnerabilities. Diving deeper into the MQTT vulnerabilities, there has been a general increasing trend since 2015 in terms of number of vulnerabilities and percent of MQTT related against all reported vulnerabilities as shown in Figure 9 and Figure 10. Notice that the percentage was calculated from NVD database only as CVE does not support search all function. The result coincides the upwards trend of cloud computing and IIoT usage in recent years. In the remaining sections of the report, MQTT will be further researched.

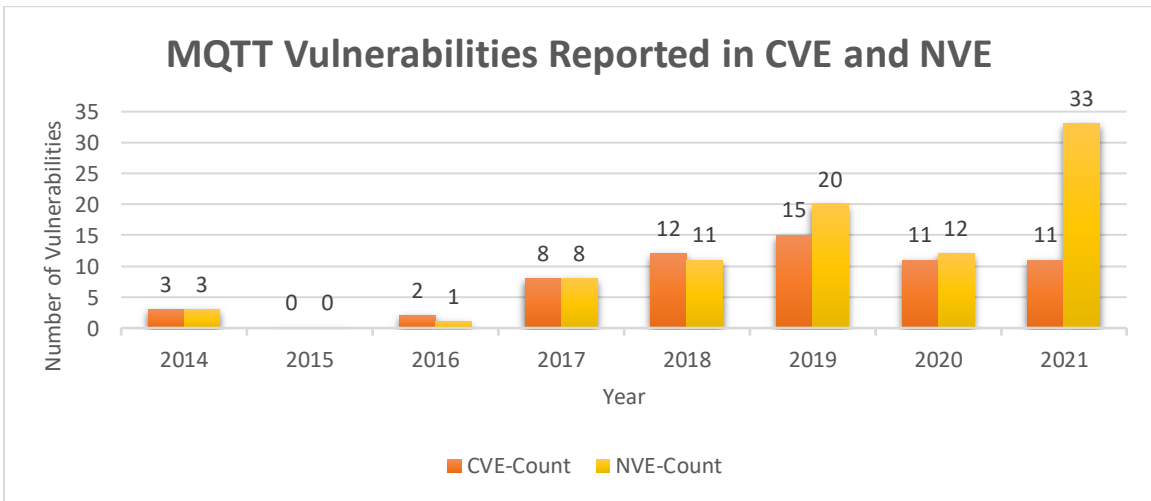


Figure 9. Number of MQTT reported vulnerabilities in CVE and NVE databases during 2014 - 2021

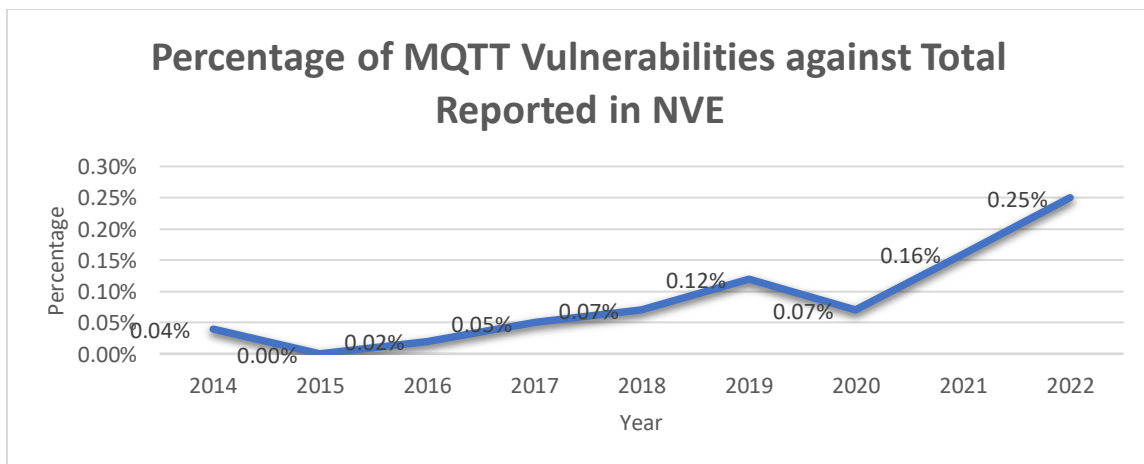


Figure 10. Percentage of MQTT reported vulnerabilities against all vulnerabilities in NVE databases during 2014 – 2022

### 3. Overview of MQTT

Before we deep dive into the research in securing MQTT, we need to have an overview of MQTT protocol and understand its current security features. In this section, various aspects of the MQTT protocol will be discussed.

#### 3.1. Publish-Subscribe Model

There are three components in the MQTT architecture: i) MQTT Broker Server, ii) MQTT client as publisher and iii) MQTT client as subscriber. There is only 1 MQTT broker server to centralize the messages and coordinate the workflow among clients. There are two types of MQTT clients. Publisher will send the data to the broker as topic. Subscriber will register the topic interested to the broker. Whenever there is new message sent from publisher, broker will broadcast to the subscribers who are interesting to the topic. The broker may store messages to the database optionally if the clients request to retain message. <sup>[28]</sup>

MQTT operates in publish-subscribe model. The publisher and subscriber can connect at different times. Different publisher can publish message in the same topic. Multiple subscribers can get the messages at the same time. The asynchronous connectivity and multiple-to-multiple connections features are particularly suitable for IIoT use cases. Besides the centralized broker design can help to reduce the computing complexity and power consumption from the clients. Clients do not need to handle multiple connections as in point-to-point model if there are multiple clients listening to the same topic. The broker can have more computing power while keeping the clients lightweight. That is why MQTT stands out in M2M/IoT protocol usage.

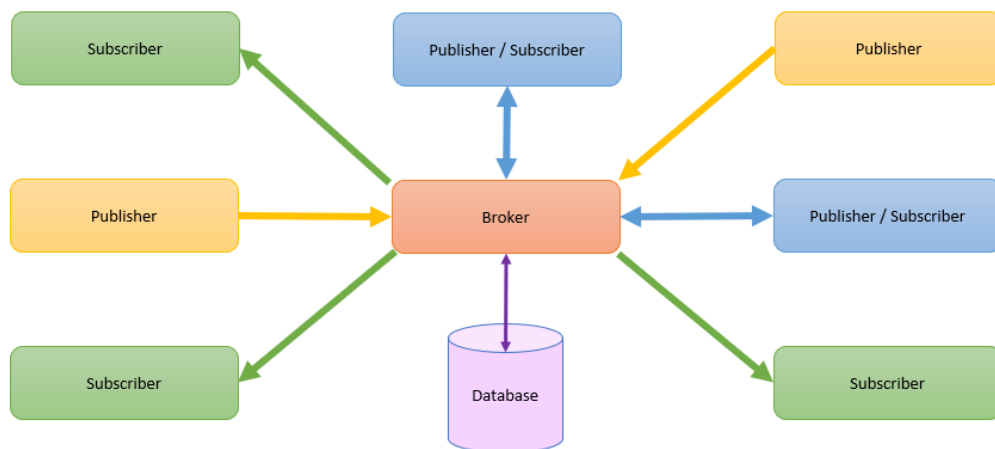


Figure 11. Publish-Subscribe model of MQTT protocol <sup>[29]</sup>

### 3.2. Message Format and Type

A typical MQTT package consists of four main components. They are i) fixed 1 byte of control header, ii) fixed 1 byte of package length, iii) variable length of header options and iv) variable length of payload. The 2 fixed length components are mandatory in all MQTT message while the 2 variable length components are optional. <sup>[30]</sup>

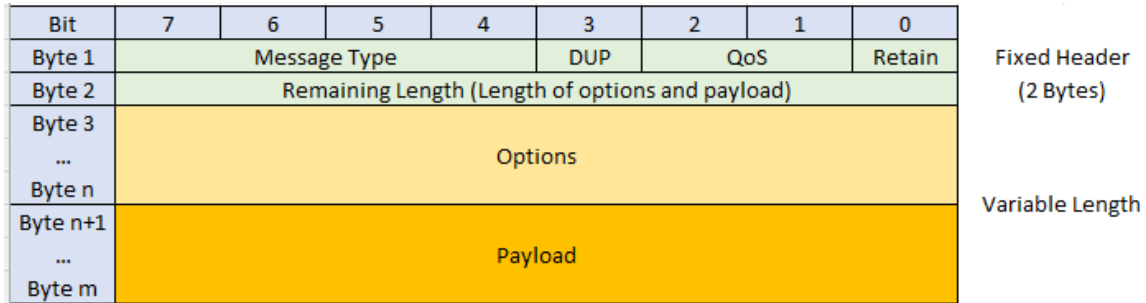


Figure 12. General MQTT packet format <sup>[34]</sup>

There are 16 (4-bit) message types in MQTT protocol. Only fourteen message types are defined by the specification with the remaining two reserved for future use. Figure 13 shows a typical message interaction among broker, publishers, and subscribers. CONNECT message will wait for the connection to be established. DISCONNECT message will wait for the MQTT clients to complete all necessary works and for the TCP/IP session to disconnect. PUBLISH message will return immediately to the application thread after passing the request to the MQTT Broker/Clients. SUBSCRIBE will register the client's interested topic to the broker. There are some message types for quality of service (QoS) purpose including PUBREC, PUBREL and PUBCOMP and some for heartbeat purpose including PINGREQ and PINGRESP. <sup>[28]</sup>

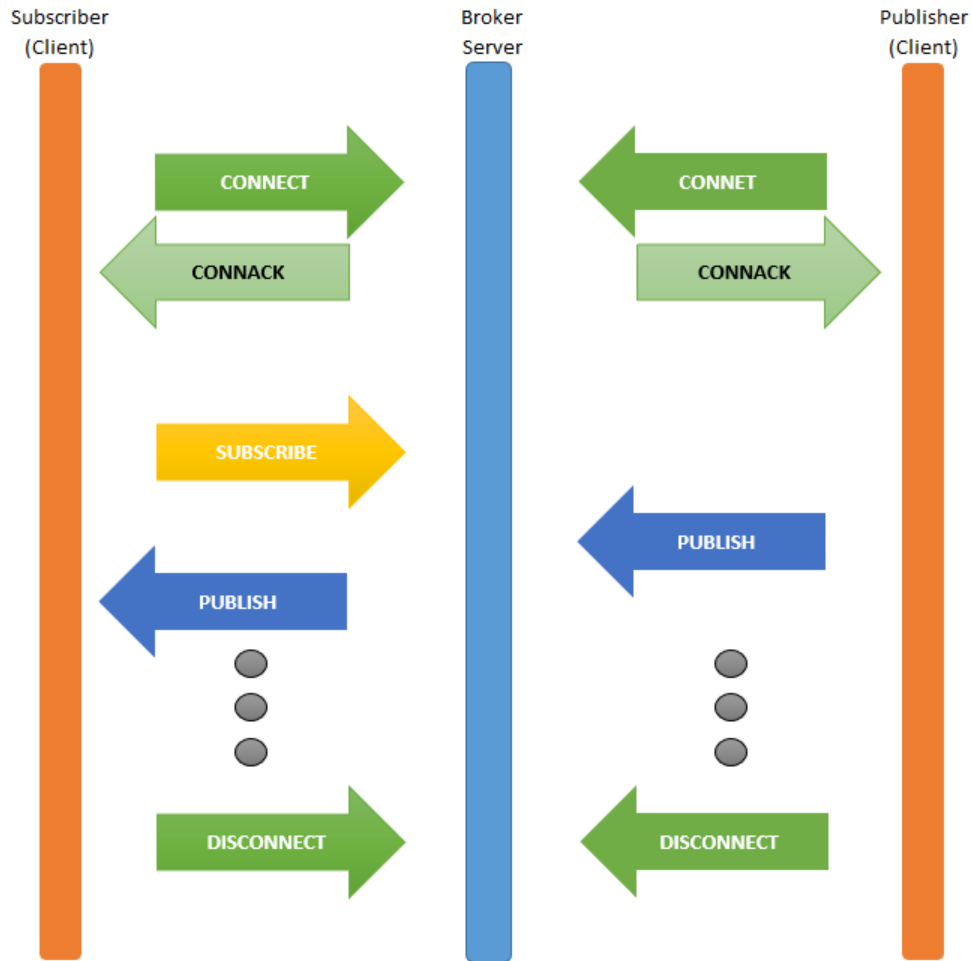


Figure 13. A typical message interaction of MQTT protocol communication (QoS 0)

### 3.3. Topic

The topic of MQTT protocol is in hierarchical format with one or more topic levels. The topic levels are separated by using forward slash (/), for instance, plant\_1/room\_2/machine3/temperature. MQTT protocol supports two type of wildcard topic subscription. Single-level wildcard is denoted by plus character (+). For example, the topic plant\_1/+/machine3/temperature covers plant\_1/room\_1/machine3/temperature, plant\_1/room\_2/machine3/temperature, plant\_1/room\_3/machine3/temperature, etc. Multi-level wildcard is denoted by hash character (#). For example, the topic plant\_1/# covers plant\_1/room\_1/machine3/temperature, plant\_1/room\_2/machine4/humidity, etc. The wildcard feature is also one of the vulnerabilities as attackers can listen to all topics by listening to wildcard top level topic (#) if the client connection is not secured. It will be discussed in detail in section 5. <sup>[28]</sup>



### 3.4. Quality of Service

Quality of Service (QoS) in MQTT is specified when client connects to broker. It defines the message assurance feature between client and broker, not client-to-client end to end. There are three QoS level: 0 for at most one, 1 for a least 1 and 2 for exactly once.

- QoS level 0 means that the sender (client or broker) will not wait for acknowledgement. So, QoS 0 PUBLISH message will be received at most one only.
- QoS level 1 means that the sender will keep retry until acknowledgement is received. So, QoS 1 PUBLISH message will be received at least one.
- QoS level 2 involves two-level handshake between sender and receiver to ensure exactly one message will be received. The sender needs to know when to resend and receiver needs to know how to identify and discard duplicated message.

MQTT is one of a few IoT protocol supporting QoS service with lightweight packet size. This make MQTT prevalence among the IoT protocols. <sup>[28]</sup>

### 3.5. Security Feature

Authentication of MQTT protocol is achieved by sending username and password from client to broker in CONNECT message. Broker will validate the profile and authorize the allowed topic and resource after authentication. However, the authentication feature is optional and the profile in CONNECT message is sent in plain text. The intermediate nodes or attackers can see the message content if they are sent in plaintext TCP. The encryption relies on the underlying transport layer like implementing MQTT over Transport Layer Security (TLS) instead of plain TCP.

The payload of the MQTT protocol is sent in plaintext also by design. The end-to-end encryption is left to the client to design which algorithm to use and how to implement them. This kind of default setting without security features provide a major vulnerability on the node or even network running MQTT protocol. Many organizations just use the default setting without any security feature implemented. More details will be discussed in section 5.

## 4. Literature Review of Securing MQTT

MQTT is a lightweight and relatively simple protocol that makes it dominate the IoT protocol. On the other hand, the protocol provides minimal security feature. It leaves to the implementer to determine what security approach is suitable for its usage. TLS is the common way to secure MQTT. But the certificate generation and session key management with high computation complexity is inappropriate for IoT devices with limiting computing resource. In summary, there are four directions in securing MQTT: i) securing the key for authentication and payload encryption, ii) implement authorization at broker like using access control list (ACL), iii) enhance the protocol for better security capability and iv) enhance the network infrastructure or components to detect and prevent the cyberattack. MQTT was not designed for IoT/IIoT initially. Apart from exposing to internet connection, IIoT applications are generally operated in low power, limited computing resource, low bandwidth, and long-lasting constraints. In this section, only the research related to securing MQTT for IoT/IIoT applications will be discussed. Other research in securing MQTT for general purpose is not in scope.

### 4.1. Enhanced Authentication and Authorization

In [29], it pointed out that TLS protocol consumes more than 100KB of memory and a lot of computing resource. TLS may not be suitable for IIoT devices usage with low power and low bandwidth constraint. So, the research proposed to use Cryptographic Smart Card technology. It is a hardware secure and well tested technology to execute cryptographic functions with a public key repository. Without modifying the protocol, it used asymmetric cryptography algorithm (RSA\_NOPAD, RSA\_PKCS1, ECC, etc.) for the authentication and block cipher algorithm for payload encryption. The key feature of this proposal is that the key-pair and random number generation are generated directly in smartcard device. The private keys never leave the device and random number only leaves the device in encrypted format. It protects the MQTT applications from Spoofing, Man-in-the-middle, reply attack, statistical disclosure attack and denial of service attack.

In [31], it used lightweight elliptic curve (ECC) instead of Rivest–Shamir–Adleman (RSA) cryptography as the secure asymmetric key for encrypting data. As proven in the research, ECC provides the same cryptographic strength with much smaller keys than RSA-based algorithm. The ratio ranges from 7:1 to 30:1. Stronger the strength with larger the key-size, the higher the compact ratio is. For example, a 2048-bit RSA key is at the same strength as a 224-bit ECC key. ECC uses less computing power and memory resources which is important to IIoT applications. The only drawback is that ECC requires more agreement between the devices such as what type of curve to use and the curve parameters.

In [32], a block cipher approach for securing MQTT (BS-MQTT) was proposed. A chaotic algorithm was used for authentication. Firstly, the algorithm ensured the diversity among consecutive security keys to avoid attackers from observing pattern among keystreams. Secondly, BS-MQTT refreshed the secret key used between client and broker regularly to avoid the standing key being stolen and used. Thirdly, the algorithm with topic based self-key agreement and block cipher was proposed to diffuse the relationship of plaintext and ciphertext. That means the key generation depends on the previous used topic.

In [33], a token-based approach was suggested. It based on OAuth 2.0 authorization framework for authentication and authorization. A protected authorization server (AS) is responsible to generate key defining the scope, lifetime, and other access attribute whenever the IoT device need to access the server. It used a challenge-response mechanism combined with elliptic curve cryptography running on-chip physically unclonable functions (PUFs) to generate the secret key required. It can avoid the key from being stored in the IoT device memory or exchanged over the line. PUFs will be assessed to get new key if needed. Besides, PUFs are used to authenticate and authorize the IoT device. This kind of hardware authentication is well protected against physical and cloning attack on the physical IoT device to get the secured key.

## 4.2. Enhanced Protocol

In [34], a Secure MQTT (SMQTT) with enhanced protocol feature was proposed. The encryption was based on the lightweight Attribute Based Encryption (ABE) over elliptic curves. The key feature was that ABE supports broadcast encryption in which single publisher's encrypted message can be decrypted by all subscribers. This made the solution suitable for IIoT applications. The proposal enhanced the protocol by using the reserved MQTT message type "0000" as new MQTT publish message Spublish. The proposed protocol was secure under chosen plain text attack (CPA), chosen ciphertext attack (CCA), man-in-the-middle and collusion attacks.

In [35], an enhanced version MQTTSec was proposed. It focused on payload encryption based on Context-Aware Cryptographic Selection Algorithm (CASA). This selection algorithm was specifically designed for IoT applications. It selected the encryption algorithm dynamically based on the battery power, data size, supported encryption algorithm on the IoT client and pre-configured throughput context. It used a cryptographically strong pseudo-random number generator (CSPRNG) instead of Linear Congruential Generator (LCG) for encryption key generation. CSPRNG was chosen because LCG generation number is predictable. With enhanced CONNECT, CONNACK and PUBLISH message, the protocol can protect the IoT applications from chosen

plaintext attack (CPA), chosen ciphertext attack (CCA), man in the middle (MMA) attack and the cryptanalysis attack.

### **4.3. Enhanced Network Infrastructure**

In [36], it presented a new infrastructure to make the MQTT protocol scalable and secure for IIoT applications. It introduced multi-level brokers for data aggregation purpose. The aggregation can be achieved by production line, machine manufacturers or production site. Maintaining configuration for thousands of IoT devices in industrial plant can be very complex. The aggregation helped to reduce the rule-based setting in the firewall or Intrusion Detection System (IDS). It also introduced single trusted authentication and authorization server (AS) for client authentication and token generation for authorization. It offloaded the more and more complex encryption algorithm from resource limited IoT device to separate AS server. AS server generally has more computing power and network resources.

### **4.4. Additional Intrusion Detection System (IDS)**

In [37], a lightweight Intrusion Detection System (IDS) called Secure-MQTT was introduced. The rule-based engine used fuzzy logic to selected network traffic features. Besides, it also employed fuzzy rule interpolation to dynamically add new rules according to past traffic patterns. It protected the IoT application from denial-of-service attack efficiently.

In [38], it introduced an IDS to detect and protect protocol-based vulnerabilities. It based on the reported vulnerabilities in common databases like NVD and CVE to define parsing rules. It can help to prevent additional attacks arisen from protocol-based vulnerabilities, such as packet crafting attacks and flooding attacks.

## 5. Analysis of MQTT Vulnerabilities

As review in section 2, MQTT is the most popular IIoT protocol. There is plenty of research done to secure MQTT from all aspects in section 4. There is end-to-end network protection by public/private key encryption. There is protocol level research to enhance MQTT to secure MQTT. There is intrusion detection system-based research to safeguard MQTT-enabled devices. But there are still numerous cyber vulnerabilities of MQTT usage in IIoT. In this section, the known MQTT vulnerabilities will be further analyzed in this section.

### 5.1. Reported Vulnerabilities in NVD database

To figure out the root cause of reported vulnerabilities related to MQTT protocol, the vulnerabilities from 2014-2021 in the NVD is investigated. CVE database has similar reported vulnerabilities and so it is not further investigated to avoid duplication.

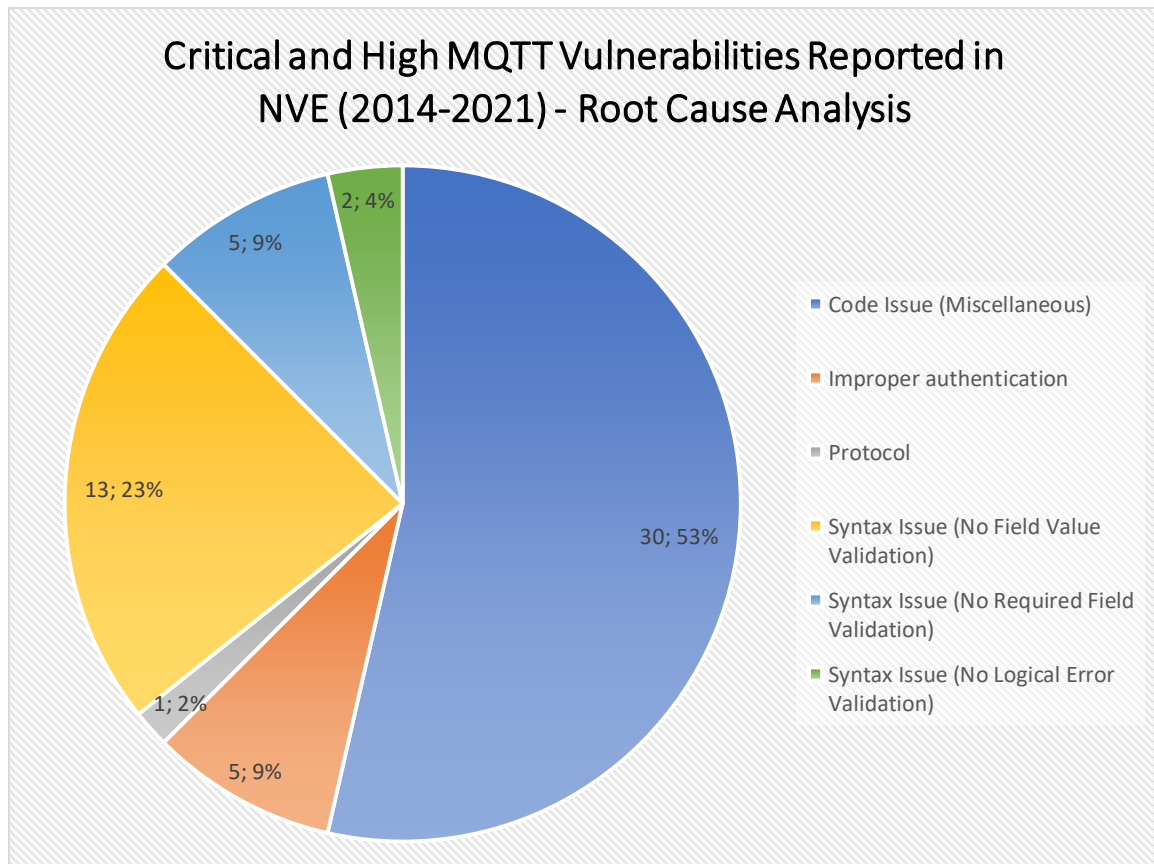


Figure 14. Root cause analysis of reported MQTT Vulnerabilities in NVE database from 2014 to 2021

There are 94 reported MQTT protocol related vulnerabilities from 2014 to 2021. Among the 94 issues, 56 issues are critical or high severities. Surprisingly, there is only 1 case related to deficiency of protocol itself. More than half of the issues are due to logical coding issue implemented by specific vendor. There are 9% issues due to improper implementation or configuration in authentication by specific vendor. These kind of vendor’s coding issues cannot be detected or prevented efficiently from cybersecurity perspective. The only effective way is to apply the software patch frequently. The remaining 36% are due to lack of strict syntax checking conforming to the specification. Although these kinds of issues are also vendor specific, it can be detected in advance by non-vendor specific solution like next generation firewall (NGFW) or intrusion detection system (IDS) for syntax validation. These syntax issues can be further categorized into 3 main types: i) no field value validation (23%), ii) no required field validation (9%) and iii) no logical error validation (4%). In this section, we will closely look into these syntax related issues.

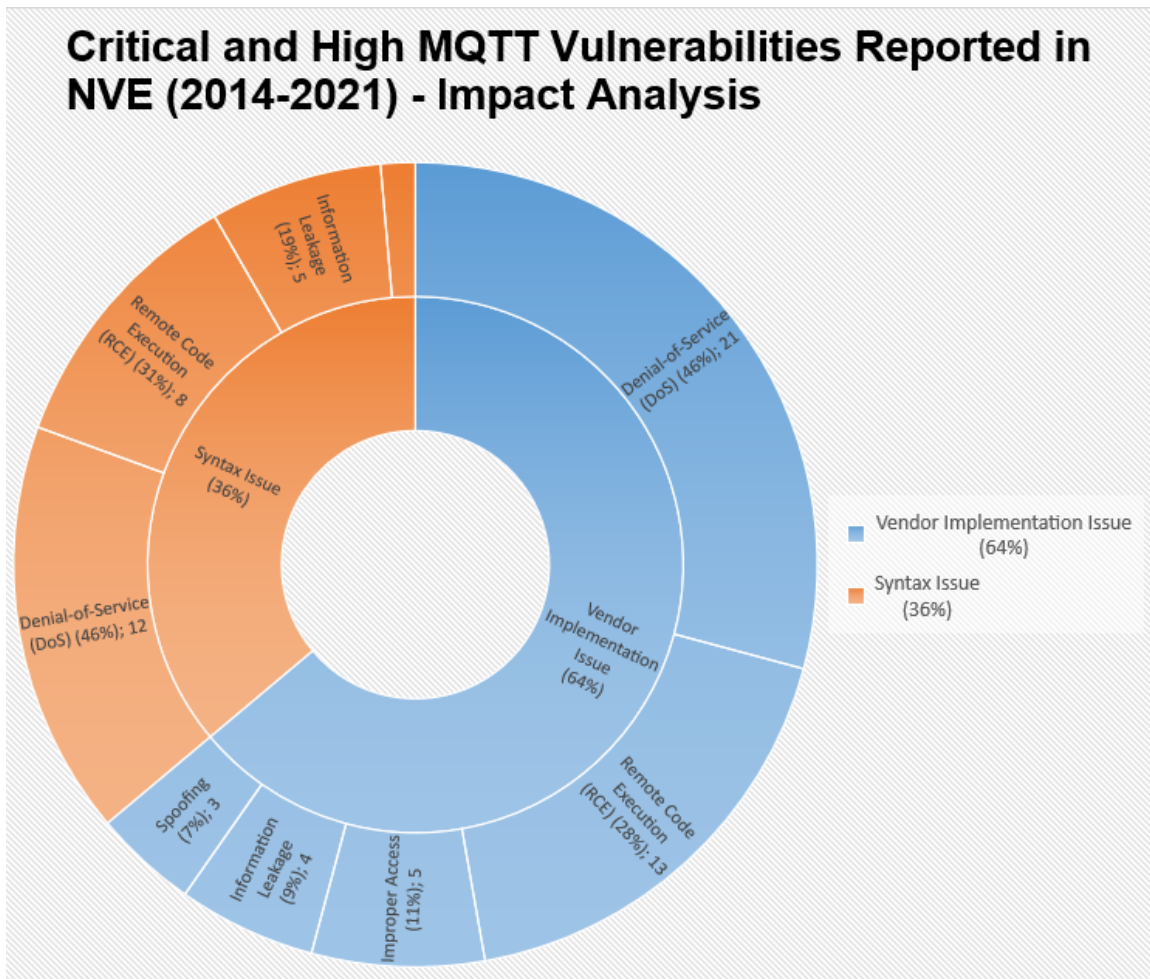


Figure 15. Impact analysis of reported MQTT Vulnerabilities in NVE database from 2014 to 2021

### **i) No Field Value Validation**

The most typical field exploited by attacker is the unmatched length field. Without proper validation of the length field with the actual packet size, the message can be parsed unexpectedly. This is also the major root cause of syntax related issues in reported MQTT protocol vulnerabilities. Here are some critical vulnerabilities found.

In CVE-2021-41036, the Paho MQTT C Client does not verify the remaining length against the actual packet size causing out-of-bounds write. In CVE-2020-10071, Zephyr MQTT parsing code does not validate the length field of publish message properly allowing buffer overflow attack. Typical consequences of such overflow attack can be server crash, DoS, and remote unauthorized code execution. In CVE-2020-10062, the Zephyr project, an off-by-one error results in the MQTT package header with 1-4 bytes being misinterpreted as 5 bytes. The memory can be corrupted, and remote code is possibly executed. <sup>[38]</sup>

Some vulnerabilities will impact MQTT broker server. In CVE-2018-19417, the MQTT server in Contiki-NG OS will parse a publish message with variable length header into fixed size buffer in implementation. The buffer can only fit maximum 64 bytes. The attacker can send publish message with more than 64 bytes even violating the rules. As there is no length checking, remote code execution is viable by stack-smashing attack (overwriting the function return address).

In CVE-2018-18765, CVE-2018-18764, CVE-2017-2894, CVE-2017-2892, Cesanta Mongoose library does not verify the length field when decode the UTF-8 content in the MQTT Subscribe message. Some specially crafted packets can lead to out-of-bounds memory access resulting in information disclosure and denial of service. <sup>[38]</sup>

Similarly, some other high severity reported MQTT vulnerabilities are also due to improper field length check, including CVE-2019-17210, CVE-2019-13120, CVE-2018-17614, CVE-2016-10523 and CVE-2017-2895. All these vulnerabilities can be avoided by proper field length validation.

### **ii) No Required Field Validation**

The second major root cause of syntax related MQTT vulnerabilities is missing required field validation, which accounts for 9% of total critical and high reported MQTT vulnerabilities.

In CVE-2016-9877, an issue is found in Pivotal RabbitMQ. For MQTT broker server using username/password authentication, if the username is validated but password is omitted, the connection can still be made successfully. This allows attackers to connect to MQTT server, subscribing to all topics as well as publish harmful topic messages to other

MQTT subscribers in the network. In CVE-2019-9749, the MQTT input plugin in Fluent Bit fails to handle a malformed crafted packet correctly. The consequence is that the server crashes due to the exception case.

Similarly, some other high severity reported MQTT vulnerabilities are also due to missing required field validation, including CVE-2018-11993, CVE-2018-8531 and CVE-2017-2893. All these vulnerabilities can be avoided by required field validation. <sup>[38]</sup>

### iii) No Logical Error Validation

The last common MQTT syntax related vulnerability is missing logical error validation. It includes the logical relationship among fields, say field A must be smaller than field B. But it is surprising that not all vendor implementing these validations correctly.

In CVE-2020-13849, an arbitrary large value of Keep-Alive value specified by a client can occupy all the MQTT broker server connections resulting in DoS attack. Per the definition of MQTT protocol 3.1.1, it requires MQTT to set the timeout value of a client connection to be 1.5 times of the Keep-Alive value. For a 16-bit field, the maximum Keep-Alive value is 65,535. So, one CONNECT message can occupy the server connection for maximum 27 hours and 18 mins. Even worse, the Keep-Alive value is specified by client rather than any objective measurement like the round-trip time between server and client. Attacker exploits a small bandwidth but can hold the server connects for a pro-long period of time.

Another instance in CVE-2019-11778, the Eclipse Mosquitto server may possibly crash if a MQTT client send a packet with the “will delay interval” value larger than the “session expiry interval”. According to the specification, the relationship should be reversed that “session expiry interval” is larger than “will delay interval”. <sup>[38]</sup>

## 5.2. Potential Attack Surface according to Shodan Report

The security feature of MQTT is by default optional. The password authentication is optional. The TLS security channel setup relies on underlying TCP layer and is not mandatory. The encryption of payload is not defined and leaves to application to determine how the end-to-end encryption is done. According to the Shodan report in section 2, there are more than 300 thousand open MQTT brokers assessable on the internet without any security feature implemented. More seriously, the subscription topic supports wildcard “#” filter. The subscriber can subscribe all topics without knowing any specific topic name and path.

Per the findings from the Trend Micro research, many important information can be exposed in the MQTT message. Figure 16 shows the broadcasting telemetry data of an



open MQTT broker. It is from a machine operating in Europe. The blurred name reveals the underlying brand of the machine. Figure 17 shows the over the air (OTA) firmware upgrade message. <sup>[39]</sup>

```

{
  "telemetryDataList": [{
    "date": "Mar 14, 2018 7:09:50 PM",
    "category": "main",
    "description": "InfoAxes Position Y",
    "deviceDescription": "CNC [REDACTED]",
    "dataType": "NUMERIC",
    "devId": 7,
    "varId": 40,
    "value": 0.0,
    "quality": true
  }, {
    "date": "Mar 14, 2018 7:09:50 PM",
    "category": "main",
    "description": "InfoAxes Position Z",
    "deviceDescription": "CNC [REDACTED]",
    "dataType": "NUMERIC",
    "devId": 7,
    "varId": 42,
    "value": 0.0,
    "quality": true
  }, {
    "date": "Mar 14, 2018 7:09:50 PM",
    "category": "main",
    "description": "InfoAxes Position X",
    "deviceDescription": "CNC [REDACTED]",
    "dataType": "NUMERIC",
    "devId": 7,
    "varId": 38,
    "value": 0.0,
    "quality": true
  }, {
    "date": "Mar 14, 2018 7:09:50 PM",
    "category": "main",
    "description": "InfoAxes Position Y",
    "deviceDescription": "CNC [REDACTED]",
    "dataType": "NUMERIC",
    "devId": 7,
    "varId": 40,
    "value": 0.0,
    "quality": true
  }, {
    "date": "Mar 14, 2018 7:09:50 PM",
    "category": "main",
    "description": "InfoAxes Position Z",
    "deviceDescription": "CNC [REDACTED]",
    "dataType": "NUMERIC",
    "devId": 7,
    "varId": 42,
    "value": 0.0,
    "quality": true
  }, {
    "date": "Mar 14, 2018 7:09:50 PM",
    "category": "main",
    "description": "InfoAxes Position X",
    "deviceDescription": "CNC [REDACTED]",
    "dataType": "NUMERIC",
    "devId": 7,
    "varId": 38,
    "value": 0.0,
    "quality": true
  }
  ]
}

```

Figure 16. Example telemetry record of a machine operating in Europe <sup>[39]</sup>

```

topic: [REDACTED]-a15464/Simplmentation/ota/firmware/73db98a223cfca494b6ab6f09d141aba timestamp: March 13th 2018, 18:22:22.965
broker: [REDACTED] topic_b64: - payload: 6QEAIJzyEEA888BAAlUAABAQAABQ9RBAHESaQMkAED///8///8PQP9/EED///9F8P//PwAQAAAc4gBAAEo
AQExKAEAAABnBgAAAAGDgrAEDwMABAoCRAQLcdwQQAEG8gABAA63wSAGASwdAJsfmh/QEpTzhPIeb/KiNLpwxEAeb/wAAAJFIMEoYHAAAAIEH/KQ8oDygCKS8oDygSKT84H4sv
Ad7/wAAADAIdDwi.x+KESwTANBAAAABLBwAnx+eh9ASmPKIBi0hApDzLPECgPQqAIAAdH/wAAAJEIMEkYuAAoD4siKQ8MAikfBIUADAIPbwwCKXByzXgoDwyEAcf/wAAAJDIMI
sYjACgPiYIpDyhwKT8MAiJPCCG/Mch/JzMEDBIIITwgoPzG//yezDCg/Mb7/JzMEDBIIITwgoPzG8/yezB5KgASJPCDIPCAwSICMwICB0jJIoFzgpKlMp08YGADg/SH8o0wGs/8

```

Figure 17. OTA firmware upgrade message via MQTT <sup>[39]</sup>

The typical IoT attack starts with finding weakest node from the IT network. Then by installing malware via the IT network, it can further hop to different servers in the network. Attackers can then penetrate into OT network via the vulnerable nodes in the IT network. The wildcard topics subscribed from the unsecured open MQTT broker provide abundant information like the firmware version and the machine brand to the attackers. Attackers can launch specific firmware or machine attack to penetrate into the OT network. It will be much easier than brute-force attack or arbitrary malware attack from the network.

## 6. Proposed Mechanism

There is various research in securing MQTT as reviewed in section 4. But there are still many vulnerabilities not yet addressed as discussed in section 5. A vast amount of MQTT brokers simply use the default configurations without adopting any security feature. From the reported vulnerabilities, 50% of them are vendor code implementation issue that can only be resolved by applying update patches. Besides, the research usually resolves a particular problem at a time. There is a lack of a holistic review for how to secure the MQTT protocol in IIoT application usage. In this section, zero trust architecture will be adopted to implement a holistic solution for securing MQTT protocol in IIoT applications.

There is no single technology that can protect the applications from all kinds of cyberattacks. Defense in cybersecurity needs to be implemented in multiple layers. Zero trust concept draws a lot of attention in recent years. With the advancement in technology, the computer network is vulnerable anytime anywhere. So, the network design cannot assume any inherited trust, even within internal network. Assuming comprised is the fundamental concept behind zero trust architecture. Instead of persistence permission, we need to keep evaluating and validating the trustworthiness of the connections. National Cyber Security Centre (NCSC) from UK government provides eight principles in implementing the zero-trust architecture. <sup>[40]</sup>

Firstly, the principles advise that all devices should be uniquely identifiable. Device identity tightly bound to the device on a secure hardware processor gives highest confidence. Software-based key stores in a well-managed device gives lower confidence. Software-based key stores in an unmanaged device gives the lowest confidence. So, hardware token-based authentication is preferred. Besides, it is mature technology in the market that there are couple of RSA or ECC enabled chips available. The chips can be run in low power consumption and are suitable to IIoT application. They can help to avoid man-in-the-middle attack.

Secondly, the principles suggest using policies to authorize requests. So, instead of simply authenticating based on profile in CONNECT message of MQTT, the broker server, or even an additional policy engine per NCSC suggested, should depend on a basket of information to make decision. My proposal is to add a protocol-based rule validation at the policy engine also. The policy engine is typically run separately with more computing resources and bandwidth than the IoT sensor devices. So, more complex logic can be built with not much resource constraint concern.

As in the findings in section 5, around 40% of high-risk vulnerability can be avoided by strict syntax validation. Although different vendors will also implement similar validation in some degree, we cannot 100% trust the vendor implementation as shown in the reports. The implementation vulnerabilities are from various vendors with some leaders in the market. So, the best solution is to implement a vendor independent validation policy engine. Besides, protocol validation only needs to be implemented once only unless the specification is changed. By independent implementation, system stability improved by avoiding any new implementation error arisen from vendor software upgrade. This proposal can avoid the denial-of-service attack and remote code execution scenarios as stated in section 7.

Thirdly, as shown in section 5, there are many MQTT brokers connecting to internet with default unsecure configuration only. The attackers can listen to all topics using wildcard at root level (#) to get all messages and see if any valuable information for them to further attack, say firmware version or machine brand used. In actual case, the topics used within the organization are well-known beforehand. The subscribers within the organization must know the exact topic path in order to interpret the message correctly. It is required to use wildcard only at sub-level but not at the top level. Without knowing the top-level topic path, the difficulties to listen to the broker messages will be largely increased. So, my proposal is to disable the top-level wildcard in SUBSCRIBE message. It can be implemented in the policy engine also. It can help to avoid information leakage and denial of service attack.

Fourthly, the principles recommend keeping authentication on every message request. I would propose various changes to enhance the MQTT protocol: i) adding sequence number to payload of all message types and ii) allowing DISCONNECT message to be sent from broker to client to disconnect suspicious session. Sequence number is not the standard MQTT protocol field. By adding it to payload, only those internal subscribers know how to interpret it. The attackers will not be able to continue the sequence number correctly. The workflow is that sequence number will be reset for each client and broker session during CONNECT messages interchanged. The sequence number as well as the hash of the message is stored at the broker side. As long as there is duplicated sequence number with different hash, large gap in sequence number, etc, it indicates the device may be compromised and follow-up action is required, say sending alert to a dashboard. The broker should enhance to send DISCONNECT message to client for high-risk scenario and the client needs to be able to handle the disconnection and reconnect with new key and sequence number again. It can help to avoid reply-attack and man-in-the-middle attack.

Finally, the principles propose to keep monitoring for the device and network healthiness. I propose to add an auto connection refresh mechanism for say every 30 mins. The current MQTT protocol session between client and server has keep alive checking by PING message. But there is no hard connection refresh if the connection keeps healthy. Regular refresh is essential as the clients may somehow compromised. Resetting connection can force the client and server regenerate the keys for encryption/decryption to minimize the impact of compromised device and the impact of information leakage.

Notice that zero trust principles assume that the network is comprised, even within the internal network. So, the proposed changes are applicable to internal or external connections. The MQTT client can be essentially any IoT sensors within the internal networks. It can also be a cloud server or AI enabled analytical device in the external networks. As the IIoT inherits all IT, OT, cloud and 5G cyber threat, the existing measures, like firewall, are still applied. They are not in scope of this proposal discussion. In Figure 18, the summary of the proposed changes to protect MQTT-enabled IIoT applications is illustrated.

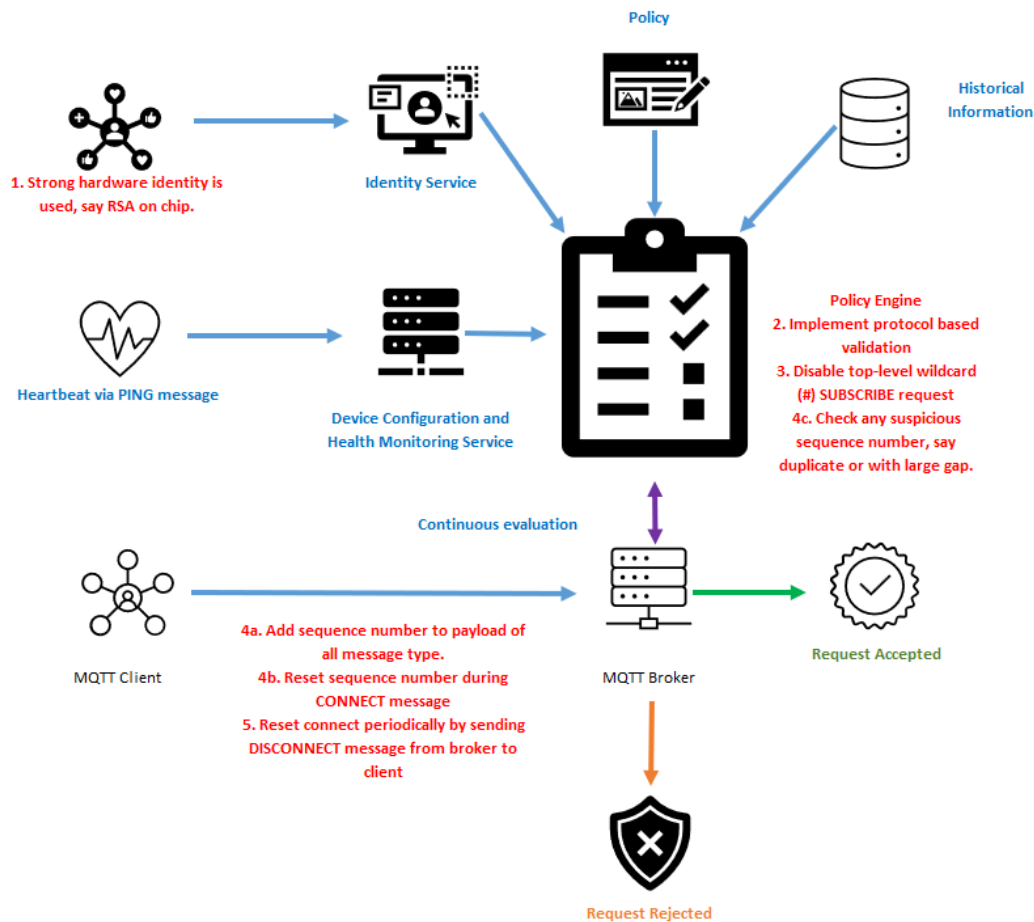


Figure 18. Proposed changes to protect MQTT enabled applications within the organization

## 7. Results Analysis and Elaboration

### 7.1. Testing Environment Setup

A set of experiments according to the proposed changes in section 6 were performed. They will be discussed in the following sections. The testing environment setup and specification are shown in Figure 19 and Figure 20. There were 3 machines with the same hardware configuration - Window 11 Home with 16GB RAM. One of the machines ran all normal MQTT clients. I simply pick a typical scenario with more publishers (10) than subscribers (2) for testing. Another machine ran purely the attackers with 1 subscriber trying to listen the message illegally and 1 publisher trying to attack the system. The attackers are separated from the machines of clients and server to avoid disturbing the performance measurement as performance of attackers is not my concern. The third machine ran the MQTT broker and the policy engine that will block the invalid traffic.

For all the experiments, two sets of testing were performed. One set enabled the policy engine, and another set did not. Besides, to focus on testing the proposed solution only, the broker ran at default port 1883 without TLS enabled as it was not part of the proposal. TLS can be applied on top the proposal. But it was not in our testing scope.

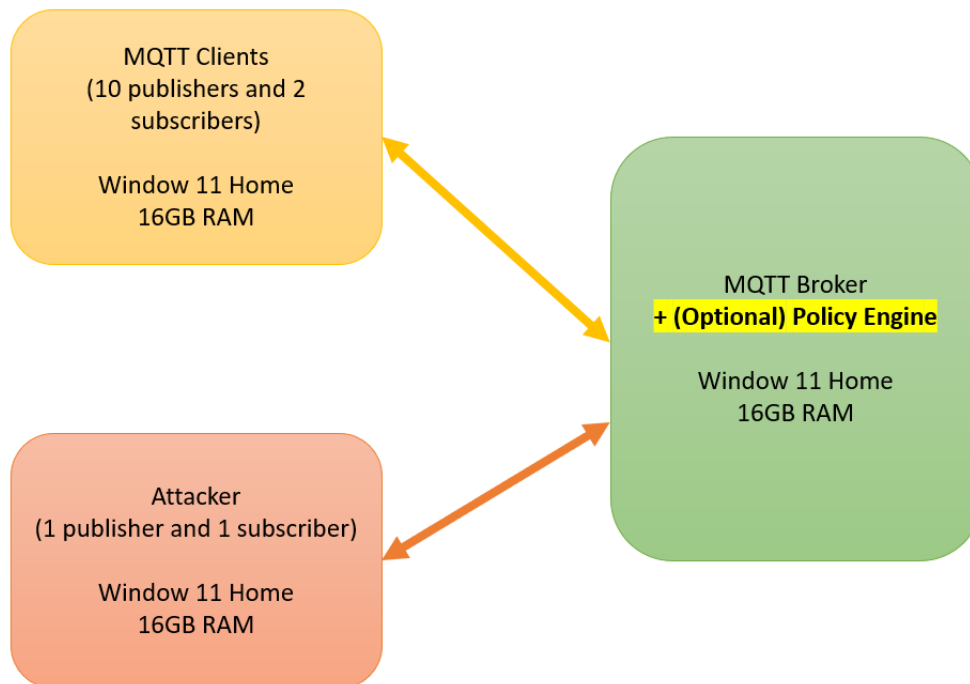


Figure 19. Testing Environment Setup

	<b>MQTT Broker</b>	<b>MQTT Publisher and Subscriber</b>	<b>Attacker</b>
<b>Hardware</b>	Intel® Core™ i7-10510U CPU @ 1.80GHz 2.30 GHz	Intel® Core™ i7-10510U CPU @ 1.80GHz 2.30 GHz	Intel® Core™ i7-10510U CPU @ 1.80GHz 2.30 GHz
<b>Primary Memory capacity</b>	16GB	16GB	16GB
<b>Operating System</b>	Windows 11 Home (64bit)	Windows 11 Home (64bit)	Windows 11 Home (64bit)
<b>Java version</b>	Java jdk-16	Java jdk-16	Java jdk-16
<b>MQTT version</b>	3.1.1	3.1.1	3.1.1
<b>MQTT Software version</b>	mosquitto-2.0.14	Eclipse Paho Client v1.2.5	Eclipse Paho Client v1.2.5

Figure 20. Server specification of the testing environment

## 7.2. Areas of the Proposed Mechanism Not Being Tested

One proposed change in section 6 about using the hardware token generation on chip was not tested in this project. As in the research in [33], key generation from chip is viable and is the preferred solution to avoid key exchange over the network and being stolen. However, this project does not have the required hardware chip for testing. It is shown in section 6 for the sake of completeness of the solution. In this section, the experiments were based on some hardcode keys in client and server program to perform the encryption instead of getting it dynamically from chip. But it did not affect the validity of the remaining architecture and testing results.

## 7.3. Experiment 1: Testing Normal Case as Control Test

Experiment	Description	Message Blocked by Policy Engine	CPU usage of Clients	CPU usage of Policy Engine	CPU usage of Broker with Policy Engine Enabled	CPU usage of Broker with Policy Engine Disabled
1	Normal case (Control test)	NA	60-70%	20-30%	45-55%	45-55%

Figure 21. Testing result of experiment 1 – Normal case

In this experiment, only normal messages were tested. All the messages were processed successfully no matter enabling policy engine at broker or not. But it is worth to note that the CPU usages of broker with policy engine enabled and disabled were the same. It was because all messages were processed in both cases. So, the CPU usage of policy engine was the additional resource required for normal case. This was the drawback of implementing policy engine.

## 7.4. Experiment 2: Testing Field Value Validation

Experiment	Description	Message Blocked by Policy Engine	CPU usage of Clients	CPU usage of Policy Engine	CPU usage of Broker with Policy Engine Enabled	CPU usage of Broker with Policy Engine Disabled
2	Field Value validation	Yes	60-70%	20-30%	5-10%	50-60%

Figure 22. Testing result of experiment 2 – Field Value Validation

In this experiment, different malformed message with arbitrary invalid fields values were tested. The cases included putting non-numeric values in expiry time field, message field length not matching actual payload length, message not in MQTT format, etc. The policy engine could discard all malformed messages successfully. It is worth to highlight that the total CPU usage policy engine and broker was less than broker with policy engine disabled. So, the lightweight policy engine is more efficient than letting the broker to discard them even the broker can handle them. It is because the logic of policy engine is more straight forward than broker and less intermediate processes required. Besides, the CPU usage of broker after using policy engine was significant less because the messages were discarded by policy engine and did not reach the broker.

## 7.5. Experiment 3: Testing Required Field Validation

Experiment	Description	Message Blocked by Policy Engine	CPU usage of Clients	CPU usage of Policy Engine	CPU usage of Broker with Policy Engine Enabled	CPU usage of Broker with Policy Engine Disabled
3	Required Field validation	Yes	60-70%	20-30%	5-10%	50-60%

Figure 23. Testing result of experiment 3 – Required Field Validation

In this experiment, message with missing required fields were tested. They included missing client ID, message type or message length. The policy engine could discard all malformed messages successfully. The CPU usage pattern was the same as experiment#2 that policy engine was more effective to discard malformed message in total.

## 7.6. Experiment 4: Testing Logical Error Validation

Experiment	Description	Message Blocked by Policy Engine	CPU usage of Clients	CPU usage of Policy Engine	CPU usage of Broker with Policy Engine Enabled	CPU usage of Broker with Policy Engine Disabled
4	Logical Error validation	Yes	60-70%	20-30%	5-10%	50-60%

Figure 24. Testing result of experiment 4 – Logical Error Validation

In this experiment, message with unmatched field values among fields were tested. They included “will delay interval” value larger than the “session expiry interval”. The policy engine could discard all malformed messages successfully. The CPU usage pattern was the same as experiment#2 that policy engine was more effective to discard malformed message in total.

### 7.7. Experiment 5: Testing Wildcard Topic Subscription

Experiment	Description	Message Blocked by Policy Engine	CPU usage of Clients	CPU usage of Policy Engine	CPU usage of Broker with Policy Engine Enabled	CPU usage of Broker with Policy Engine Disabled
5	Wildcard topic subscription	Yes	60-70%	5-10%	5-10%	10-15%

Figure 25. Testing result of experiment 5 – Wildcard Topic Subscription

This experiment was to send some SUBSCRIBE message to the broker with some having wildcard at topic level (#) and some having wildcard at sub-level (i.e., plant\_1/#). Notice that they were valid topic according to specification. So, they were processed as normal message by broker with policy engine disabled. With policy engine enabled, the SUBSCRIBE message with root level wildcard (#) were discarded while others can be processed normally. There is not much difference in CPU usage as there were only few subscription messages the same as actual cases. The policy engine could parse these messages effectively.

### 7.8. Experiment 6: Testing Invalid Sequence Number

Experiment	Description	Message Blocked by Policy Engine	CPU usage of Clients	CPU usage of Policy Engine	CPU usage of Broker with Policy Engine Enabled	CPU usage of Broker with Policy Engine Disabled
6	Invalid sequence number	Yes	60-70%	5-10%	5-10%	10-15%

Figure 26. Testing result of experiment 6 – Testing Invalid Sequence Number

In this experiment, for simplicity, all messages with out of sequence number or duplicated sequence number were discarded. More complex rule, say tolerating sequence number within a range, can be implemented but it is not in our testing scenario. There was not much difference in CPU usage as the policy engine can parse these messages effectively.



On the other hand, the CPU usage of broker with policy engine enabled is smaller than that with policy engine disabled. It was because the messages were discarded by policy engine and did not reach broker. The total CPU usage of policy engine and broker was almost the same as broker without policy engine. So, implementing policy engine did not increase resource usage in this case.

## 7.9. Testing Result Summary

Experiment	Description	Message Blocked by Policy Engine	CPU usage of Clients	CPU usage of Policy Engine	CPU usage of Broker with Policy Engine Enabled	CPU usage of Broker with Policy Engine Disabled
1	Normal case (Control test)	NA	Normal	Normal	Normal	Normal
2	Field Value validation	Yes	Normal	Normal	Low	High for invalid packets
3	Required Field validation	Yes	Normal	Normal	Low	High for invalid packets
4	Logical Error validation	Yes	Normal	Normal	Low	High for invalid packets
5	Wildcard topic subscription	Yes	Normal	Normal	Low	Normal
6	Invalid sequence number	Yes	Normal	Normal	Low	Normal

Figure 27. Summary of the experiments' results

In summary, the expected vulnerabilities discussed in section 6 were 100% detected and blocked successfully as expected. With policy engine disabled, the broker CPU usage fluctuated a bit when handling the invalid packets. After enabling policy engine, the broker CPU usage was lower for invalid packets scenario as the traffic is already blocked at policy engine.

On the other hand, the CPU usage of policy engine was relatively steady no matter for normal or invalid packets scenario. This was because policy engine only parsed the message and determined to block or forward. There was not much business logic or special handling for both cases. So, the CPU usage kept constant. According to experiment 1, the only drawback of implementing policy engine was that there was additional CPU resource required for normal case as the message needed to be parsed two times by both policy engine and broker. Considering server running policy engine generally having more resource than those IIoT client device, this drawback should not be the constraint in actual case.

For experiment 5 and 6, the packets were valid under current MQTT v3.1.1 protocol specification. They are blocked due to the additional security enforcement as discussed in section 6. So, they were just treated as normal packets for broker with policy engine disabled and no abnormal CPU was observed.

## 8. Conclusion and future work

In this project, both the research work on securing MQTT protocol and the actual reported vulnerabilities are reviewed. There is no single solution that can tackle on the cyber threat. MQTT, as the most popular IoT protocol, does not have much security features defined in the specification. It relies on the implementors and the underlying infrastructure to protect it. A holistic solution based on the zero-trust principles was mostly implemented and tested successfully. There is additional CPU resource required for implementing the policy engine. But there is no performance degrade observed from other testing IoT devices.

In this project, only MQTT protocol is reviewed. The same analysis and zero-trust principles can also be applied to other IoT protocols. As shown in the Shodan report in section 3.3, CoAP protocol is another prevalent IoT protocol. But many CoAP servers connecting to the internet do not enable the security feature using the default setting. It is similar to the case of MQTT protocol. We can extend the policy engine and implement CoAP validation rules as future work. Besides, the key generation from hardware on chip is just in proposed solution and not yet tested due to hardware limitation. This can be further test with different token-based algorithm say RSA and ECC on various hardware and chips for best performance and lowest power consumption suitable for IIoT applications.

## A. Reference

- [1] C. Panchal, V. M. Khadse, and P. N. Mahalle, “Security Issues in IIoT: A Comprehensive Survey of Attacks on IIoT and Its Countermeasures”, 2018 IEEE Global Conference on Wireless Computing and Networking (GCWCN), 2018, pp. 124-130, doi: 10.1109/GCWCN.2018.8668630.
- [2] “What Are the Risks Associated With Industrial IoT (Industrial Internet Of Things)?” Archon, [www.archonsecure.com](http://www.archonsecure.com),  
<https://www.archonsecure.com/blog/risks-with-industrial-iiot>
- [3] “Number Of IoT Devices 2015-2025.”, Statista Research Department, Statista, [www.statista.com](http://www.statista.com), 27 November 2016,  
<https://www.statista.com/statistics/471264/iiot-number-of-connected-devices-worldwide>
- [4] Steve Ranger, “What Is the IoT? Everything You Need To Know About the Internet Of Things Right Now.”, ZDNet, [www.zdnet.com](http://www.zdnet.com), 3 February 2020,  
<https://www.zdnet.com/article/what-is-the-internet-of-things-everything-you-need-to-know-about-the-iiot-right-now>
- [5] “MES Software Basics| Manufacturing Execution Systems”, PLEX Smart Manufacturing Platform, [www.plex.com](http://www.plex.com), <https://www.plex.com/products/analytics-and-industrial-iiot/what-is-iiot>
- [6] “What is IoT”, Attila Security, [www.attilasec.com](http://www.attilasec.com),  
<https://www.attilasec.com/iiot-security-guide#what-is-iiot>
- [7] Graham Williamson, “OT, ICS, And SCADA – Cybersecurity Essentials.”, KuppingerCole, [www.kuppingercole.com](http://www.kuppingercole.com), 25 May 2021,  
<https://www.kuppingercole.com/insights/ot-ics-scada>
- [8] “Cyber Threat Bulletin: Cyber Threat To Operational”, Canadian Centre for Cyber Security, [cyber.gc.ca](http://cyber.gc.ca), 16 December 2021,  
<https://cyber.gc.ca/en/guidance/cyber-threat-bulletin-cyber-threat-operational-technology>
- [9] Jonathan Trull, “Addressing Cybersecurity Risk In Industrial IoT And OT”, Microsoft Security Blog, [www.microsoft.com](http://www.microsoft.com), 21 October 2020,  
<https://www.microsoft.com/security/blog/2020/10/21/addressing-cybersecurity-risk-in-industrial-iiot-and-ot>
- [10] P. Ackerman, “The Purdue Model for Industrial Control Systems.”, October 2017, [packtpub.com](http://packtpub.com),  
[https://subscription.packtpub.com/book/networking\\_and\\_servers/9781788395151/1/ch011v11sec10/the-purdue-model-for-industrial-control-systems](https://subscription.packtpub.com/book/networking_and_servers/9781788395151/1/ch011v11sec10/the-purdue-model-for-industrial-control-systems)

- [11] Nick McKenna, “The Relationship Between IoT, Big Data And the Cloud.” McKenna Consultants, [www.mckennaconsultants.com](http://www.mckennaconsultants.com), 5 May 2021, <https://www.mckennaconsultants.com/relationship-between-iot-big-data-and-the-cloud/#:~:text=Cloud%20Computing%20in%20IoT%20works,the%20IoT%20through%20the%20Internet.>
- [12] Brown Gabriel, “Private 5G Mobile Networks for Industrial IoT” Qualcomm, [www.qualcomm.com](http://www.qualcomm.com), July 2019, <https://www.qualcomm.com/media/documents/files/private-5g-networks-for-industrial-iot.pdf>
- [13] Brendon McHugh, “5G for Industrial Internet Of Things (IIoT) And the Role Of SDRs” Microwave Journal, [www.microwavejournal.com](http://www.microwavejournal.com), 1 November 2021, <https://www.microwavejournal.com/articles/37049-g-for-industrial-internet-of-things-iiot-and-the-role-of-sdrs>
- [14] Sharon Shea, “What Is Machine-to-Machine (M2M)?” IoT Agenda, [internetofthingsagenda.techtarget.com](http://internetofthingsagenda.techtarget.com), 1 August 2019, <https://internetofthingsagenda.techtarget.com/definition/machine-to-machine-M2M>
- [15] Ann Marie Johlle and McDonough Kyri, “Difference Between IoT And M2M - InterviewBit.” InterviewBit, [www.interviewbit.com](http://www.interviewbit.com), 12 December 2021, <https://www.interviewbit.com/blog/difference-between-iot-and-m2m>
- [16] Mary K. Pratt, “Top 12 Most Commonly Used IoT Protocols And Standards.”, IoT Agenda, [internetofthingsagenda.techtarget.com](http://internetofthingsagenda.techtarget.com), 1 April 2021, <https://internetofthingsagenda.techtarget.com/tip/Top-12-most-commonly-used-IoT-protocols-and-standards>
- [17] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari and M. Ayyash, “Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications” in IEEE Communications Surveys & Tutorials, vol. 17, no. 4, pp. 2347-2376, Fourthquarter 2015, doi: 10.1109/COMST.2015.2444095.
- [18] A. Rullo, E. Bertino and D. Saccá, “PAST: Protocol-Adaptable Security Tool for Heterogeneous IoT Ecosystems” 2018 IEEE Conference on Dependable and Secure Computing (DSC), 2018, pp. 1-8, doi: 10.1109/DESEC.2018.8625143.
- [19] Bhagya Nathali Silva, Murad Khan and Kijun Han (2018), “Internet of Things: A Comprehensive Review of Enabling Technologies, Architecture, and Challenges”, IETE Technical Review, 35:2, 205-220, DOI: 10.1080/02564602.2016.1276416
- [20] “OASIS Message Queuing Telemetry Transport (MQTT) TC.” OASIS, [www.oasis-open.org](http://www.oasis-open.org), [https://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=mqtt](https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=mqtt)

- [21] S. U. A. Laghari, S. Manickam, A. K. Al-Ani, S. U. Rehman and S. Karuppayah, “SECS/GEMsec: A Mechanism for Detection and Prevention of Cyber-Attacks on SECS/GEM Communications in Industry 4.0 Landscape” in IEEE Access, vol. 9, pp. 154380-154394, 2021, doi: 10.1109/ACCESS.2021.3127515.
- [22] Z. Shelby, ARM, K. Hartke, C. Bormann, “RFC 7252 - The Constrained Application Protocol (CoAP).” IETF, ietf.org, June. 2014, <https://datatracker.ietf.org/doc/html/rfc7252>
- [23] “Advanced Message Queuing Protocol” Wikipedia, en.wikipedia.org, 1 May 2018, [https://en.wikipedia.org/wiki/Advanced\\_Message\\_Queueing\\_Protocol](https://en.wikipedia.org/wiki/Advanced_Message_Queueing_Protocol)
- [24] “IoT Protocols & Standards Guide - Protocols Of the Internet Of Things.”, AVSystem, www.avsystem.com, 4 March 2020, <https://www.avsystem.com/blog/iot-protocols-and-standards/#:~:text=Due%20to%20its%20heaviness%2C%20AMQP,devices%20and%20the%20network%20are>
- [25] “XMPP | An Overview Of XMPP.”, xmpp.org, <https://xmpp.org/about/technology-overview>
- [26] “LoRa And LoRaWAN: the Technologies, Ecosystems, Use Cases And Market.”, I-SCOOP, www.i-scoop.eu, <https://www.i-scoop.eu/internet-of-things-iot/lpwan/iot-network-lora-lorawan>
- [27] “What Is Zigbee Wireless Mesh Networking?.”, Digi International, www.digi.com, <https://www.digi.com/solutions/by-technology/zigbee-wireless-standard>
- [28] Hasan, Hamid & Alhusainy, Bahaa, “Evaluation of MQTT Protocol for IoT Based Industrial Automation.”, 8. 19364-19369, 8 Dec 2018, [https://www.researchgate.net/publication/329505570\\_Evaluation\\_of\\_MQTT\\_Protocol\\_for\\_IoT\\_Based\\_Industrial\\_Automation/citation/download](https://www.researchgate.net/publication/329505570_Evaluation_of_MQTT_Protocol_for_IoT_Based_Industrial_Automation/citation/download)
- [29] E. B. Sanjuan, I. A. Cardiel, J. A. Cerrada and C. Cerrada, “Message Queuing Telemetry Transport (MQTT) Security: A Cryptographic Smart Card Approach”, in IEEE Access, vol. 8, pp. 115051-115062, 2020, doi: 10.1109/ACCESS.2020.3003998.
- [30] “MQTT Specification.”, OASIS, mqtt.org, <https://mqtt.org/mqtt-specification>
- [31] A. Rahman, S. Roy, M. S. Kaiser, and M. S. Islam, “A Lightweight Multi-Tier S-MQTT Framework to Secure Communication between low-end IoT Nodes”, 2018 5th International Conference on Networking, Systems and Security (NSysS), 2018, pp. 1-6, doi: 10.1109/NSysS.2018.8631379.
- [32] Ranbir Singh Bali, Fehmi Jaafar, and Pavol Zavarasky, “Lightweight authentication for MQTT to improve the security of IoT communication.”, In Proceedings of the 3rd International Conference on Cryptography, Security and Privacy (ICCSP '19), Association for Computing Machinery, New York, NY, USA, 6–12, 2019

- [33] M. Naveed Aman, S. Taneja, B. Sikdar, K. C. Chua, and M. Alioto, “Token-Based Security for the Internet of Things With Dynamic Energy-Quality Tradeoff” in IEEE Internet of Things Journal, vol. 6, no. 2, pp. 2843-2859, April 2019, doi: 10.1109/JIOT.2018.2875472.
- [34] M. Singh, M. A. Rajan, V. L. Shivraj and P. Balamuralidhar, “Secure MQTT for Internet of Things (IoT)”, 2015 Fifth International Conference on Communication Systems and Network Technologies, 2015, pp. 746-751, doi: 10.1109/CSNT.2015.16.
- [35] M. A. Massad and B. A. Alsaify, “MQTTSec Based on Context-Aware Cryptographic Selection Algorithm (CASA) for Resource-Constrained IoT Devices”, 2020 11th International Conference on Information and Communication Systems (ICICS), 2020, pp. 349-354, doi: 10.1109/ICICS49469.2020.239541.
- [36] M. Amoretti, R. Pecori, Y. Protskaya, L. Veltri and F. Zanichelli, “A Scalable and Secure Publish/Subscribe-Based Framework for Industrial IoT” in IEEE Transactions on Industrial Informatics, vol. 17, no. 6, pp. 3815-3825, June 2021, doi: 10.1109/TII.2020.3017227.
- [37] A. P. Haripriya, and K. Kulothungan, “Secure-MQTT: An Efficient Fuzzy Logic-Based Approach to Detect DoS Attack in MQTT Protocol for Internet of Things.”, EURASIP Journal on Wireless Communications and Networking 2019, no. 1 (April 1, 2019): 1–15. doi:10.1186/s13638-019-1402-8
- [38] M. Husnain, K. Hayat, E. Cambiaso; U. U. Fayyaz, M. Mongelli, H. Akram, Abbas S. Ghazanfar, G.A. Shah, “Preventing MQTT Vulnerabilities Using IoT-Enabled Intrusion Detection System.”, Sensors 2022, 22, 567. <https://doi.org/10.3390/s22020567>
- [39] M. Federico, V. Rainer, Q. Davide, “The Fragility of Industrial IoT’s Data Backbone Security and Privacy Issues in MQTT and CoAP Protocols”, Trend Micro Research, trendmicro.com, [https://documents.trendmicro.com/assets/white\\_papers/wp-the-fragility-of-industrial-IoTs-data-backbone.pdf?v1](https://documents.trendmicro.com/assets/white_papers/wp-the-fragility-of-industrial-IoTs-data-backbone.pdf?v1)
- [40] “Zero Trust Architecture Design Principles.”, National Cyber Security Center NCSC, nsc.gov.uk, 23 July 2021, <https://www.ncsc.gov.uk/collection/zero-trust-architecture>
- [41] “What Is Cloud Computing? A Beginner’s Guide”, Microsoft Azure, azure.microsoft.com, <https://azure.microsoft.com/en-ca/overview/what-is-cloud-computing>
- [42] “Machine Learning and Big Data | Is It the Future?”, Edureka. www.edureka.co, 16 December 2021, <https://www.edureka.co/blog/machine-learning-and-big-data>
- [43] “What Is Distributed Control System (DCS)?”, Electrical Technology, www.electricaltechnology.org, 24 July 2016, <https://www.electricaltechnology.org/2016/08/distributed-control-system-dcs.html>

- [44] “What Is Artificial Intelligence (AI)?”, IBM, [www.ibm.com](http://www.ibm.com), 3 May 2020, <https://www.ibm.com/cloud/learn/what-is-artificial-intelligence>
- [45] “What Is 5G | Everything You Need to Know About 5G | 5G FAQ.”, Qualcomm, [www.qualcomm.com](http://www.qualcomm.com), <https://www.qualcomm.com/5g/what-is-5g>
- [46] P. Popovski, K. F. Trillingsgaard, O. Simeone and G. Durisi, “5G Wireless Network Slicing for eMBB, URLLC, and mMTC: A Communication-Theoretic View”, in *IEEE Access*, vol. 6, pp. 55765-55779, 2018, doi: 10.1109/ACCESS.2018.2872781.
- [47] T. Dierks, and E. Rescorla, “RFC 5246 - The Transport Layer Security (TLS) Protocol Version 1.2.”, IETF, [datatracker.ietf.org](http://datatracker.ietf.org), August 2008, <https://datatracker.ietf.org/doc/html/rfc5246>
- [48] E. Rescorla, and N. Modadugu, “RFC 6347 - Datagram Transport Layer Security Version 1.2.”, IETF, [datatracker.ietf.org](http://datatracker.ietf.org), January 2012, <https://datatracker.ietf.org/doc/html/rfc6347>
- [49] Dindayal Mahto and Dilip Yadav, “RSA and ECC: A comparative analysis.”, *International Journal of Applied Engineering Research*, 12. 9053-9061, January 2017, [https://www.researchgate.net/publication/322558426\\_RSA\\_and\\_ECC\\_A\\_comparative\\_analysis](https://www.researchgate.net/publication/322558426_RSA_and_ECC_A_comparative_analysis)
- [50] Ben Lutkevich, “What Is an Intrusion Detection System (IDS)?”, TechTarget, [www.techtarget.com](http://www.techtarget.com), October 2021, <https://www.techtarget.com/searchsecurity/definition/intrusion-detection-system>