Layered Networking Protocols for Secure Communications in the Internet of Things

Zhangxiang Hu*

February 17, 2021

1 Introduction

The Internet of Things (IoT) which was first introduced by Kevin Ashton in 1999 refers to the interconnected physical objects such as sensors, RFID tags, and smart objects through the Internet. Throughout sensing and actuating, IoT enhances many aspects of our lives and analysts predict that the number of IoT devices will be increased from 7 billion in 2018 to more than 20 billion by 2020 [97, 46]. The increment of connectivity of these smart devices contributes to building digitalized world such as smart homes, smart agriculture and smart cities across the globe.

The massive scale of deployed IoT devices and IoT applications raise many new security concerns which are different from the traditional networks [145, 92, 21]. For instance, attacks against smart grid network in industry could increase the chances for instabilities of power supply or blackouts of cities [85]. The situation could be worse in medical-related area. Attacks against implantable medical devices such as pacemaker can not only steal sensitive patient information, but also reduce the entire life cycle of the device, and even endanger the life of a patient [135]. The differences in IoT networks are due to the heterogeneous nature of IoT devices and IoT devices are very different in terms of size, functionality, computational capability, power usage, etc [120]. For example, an IoT device in healthcare may have limited memory size, processor clock speed and short of power supply [135]. Therefore, major research effort are still required to address the privacy and security for IoT environments.

In general, a secure IoT system needs to provide solutions to against attacks of stealing private information (confidentiality), injecting false information (integrity), and denial of services and functionalities (availability) [39]. One of the main fundamental challenges faced by the IoT network to achieve confidentiality, integrity and availability is to establish secure communication. Similar to conventional wired or wireless networks,, devices in IoT networks also need to share and exchange sensitive information. To protect end-to-end communication security in IoT environment, a lot of adaptations of standard networking protocols have been proposed in the last few decades such as IKE/IPsec [80], DTLS [134], and CoAP [149]. All of these protocols still follow 5-layer network architecture which ranges from the physical layer to the application layer. Each layer leverages different protocols and algorithms to provide security services. For example, devices leverage symmetric encryption schemes such as AES to encrypt sensitive messages to protect confidentiality at the medium access control layer, and perform DTLS at the transport layer to bootstrapping trust and establish the common secret key.

Unfortunately, many solutions to secure communications in IoT networks are not appropriately deployed due to either human errors or traditional solutions are not suitable for IoT environment. An investigation in 2014 [31] reports that 70 percent of IoT devices did not encrypt their communications and 60 percent IoT devices did not use any encryption scheme when updating softwares. The situation becomes even worse recently. A survey in 2020 by Palo Alto [107] reports that 98% IoT device traffic is unencrypted and 72% of healthcare VLANs allowing malware to spread from users' computers to vulnerable IoT devices on the same network.

In this paper, we explore the state of art networking protocols that are applied at each layer to achieve secure communications in IoT. We aim to give a critical overview of the security challenges and requirements

*University of Oregon. {huz}@cs.uoregon.edu.

in IoT ecosystem and provide a taxonomy of networking protocols for IoT networks. This work will focus on standardized IoT network protocols that are widely accepted by standardization associations and industry alliances. Figure 1 provides a detail of IoT networking protocols that we describe in this report.

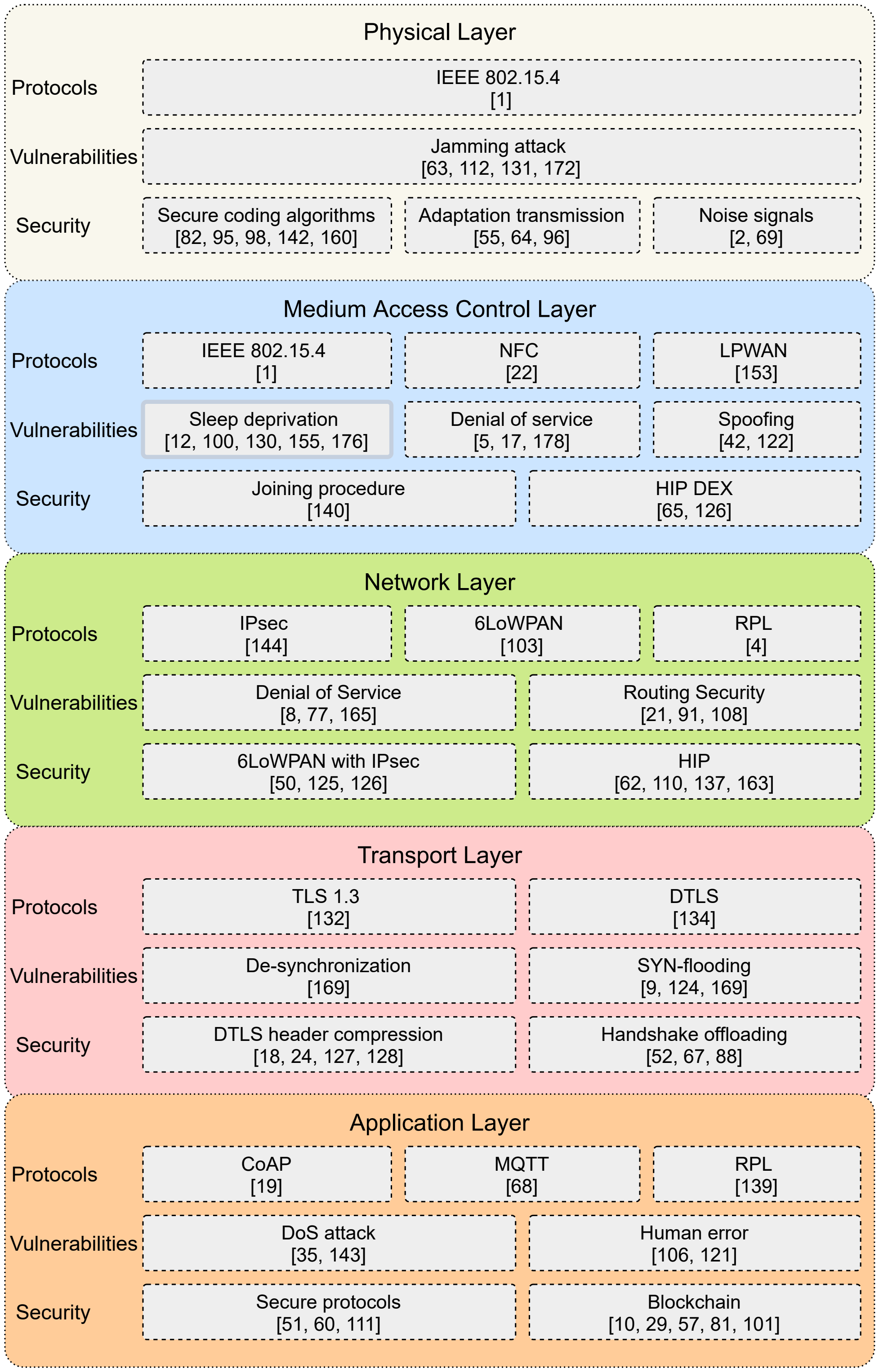

Figure 1: Taxonomy of Standardized IoT networking protocols.

The rest of the paper is organized as follows. Section 2 reviews the security requirements in IoT environments. Section 3 to Section 7 exploit layered communication protocols in IoT from physical layer to application layer and discuss the security vulnerabilities at each networking layer as well as the mitigation schemes. Section 8 investigates the key establishment protocols which is a crucial issue to improve the efficiency of existing protocols. Finally, Section 9 concludes this work and illuminates the newly emerging techniques in IoT communications.

2 Security Requirements, Adversarial Models, and Communication Channels in IoT

In this section, we first explore the security requirements in IoT, we then present different adversarial models and introduce the concept of secure communication channels.

2.1 Security Requirements

Foundations of security requirements in information security usually referred as confidentiality, integrity and availability (CIA triad) [7, 45, 119].

- Confidentiality: Only authorized entities can access the sensitive resource. Security mechanisms such as data encryption, certificate-based authentication are widely used in network systems to provide confidentiality.
- Integrity: Only authorized entities can modify the sensitive resource. To achieve the integrity, authorized entities usually apply some message authentication schemes such as digital signature and message authentication code to guarantee that the original resource is not tampered by unauthorized entities.
- Availability: Any authorized entities should be able to access the sensitive resource. The availability property should not only ensure that authorized entities can access the resource in a normal condition, but also in an extreme condition. For example, when a system is under Denial-of-service (DoS) attack, authorized entities can use firewalls to mitigate the attack or have a failover backup method to provide duplication of the sensitive resource.

However, IoT systems are different from the traditional computer systems in terms of the distinguishing properties of IoT environments. Vasilomanolakis [162] showed that there are four identified unique properties: the uncontrolled environment, the heterogeneity, the need for scalability, and the constrained resources. In particular, IoT devices usually run in uncontrolled environment due to their mobility, physical accessibility, and the lack of trust between each other. Secondly, in an IoT system, devices are highly heterogeneous in terms of size, functionality, manufacturers, power usage, etc [120]. Thus, IoT users need to consider the version compatibility, interoperability and intercommunity in a system. For the scalability, IoT systems usually require highly scalable protocols due to the large number of IoT devices and the volatile nature of IoT systems. Lastly, in contrast to conventional computer devices, IoT devices usually are resource constrained in energy accessibility, memory size, and computational capability. Many protocols or algorithms that are runnable on conventional computer devices may not perform well on IoT devices. Therefore, we need a more subtle definitions to describe the security requirements for IoT environments.

There are various prior work in defining security for IoT systems. For example, [54, 136] recognize the CIA triad and also add privacy to define the security only for implantable medical devices (IMDs) and body area networks (BANs). For the security in a more general IP-Based IoT system, Cirani *et al.* [30] identifies the network security and identity management while [90] focuses more on the privacy and trust. To our best knowledge, the requirements listed in Vasilomannolakis *et al.* [162] have the most extensive coverage to define the security for IoTs. It summarizes the security requirements for IoT and split them into five categories: *Network Security, Identity Management, Privacy, Trust, and Resilience.* However, it does not consider the energy efficiency. Since IoT devices are usually resource-constrained and characterized with low power and less memory size, any mechanisms that are used to achieve the above five security categories should also be able to perform well on most IoT devices.

Therefore, in this report we adopt the five security categories that are defined in [162] and add the extra efficiency property to formalize the security requirements in IoT. We describe them in details below.

1. Network Security. Network security refers to four subcategories: confidentiality, authenticity, integrity, and availability.

   Confidentiality is defined the same as in the CIA triad. In many IoT applications such as eHealthcare, sensitive data may transmitted over the network. The confidentiality guarantees that attackers should not be able to eavesdrop the data or retrieve any useful information from the transmission.

   Authenticity in network security refers to the proof that communication channels are established between authenticated devices. Before transmitting data, an IoT device must ensure that it communicates with the designated devices rather than malicious attackers. Also, authenticity guarantees the data origin authentication that a message originates from a certain entity [39].

   Integrity is also defined the same as in the CIA triad. During the communication, an attacker may intercept the transmission of messages, modify the data or inject false data into the messages. Thus, IoT devices must guarantee that the messages it receives are original and are not modified by attackers.

   The availability ensures that IoT devices and services are still reachable even under link failures.

2. Identity Management. Identity management is another challenge in IoT systems. Due to the complex relationship between devices, services, owners and users, [162] suggests to especially consider authentication, authorization, and accountability.

   Authentication provides the identity verification to ensure that an entity is who it claims to be [39]. Note that authentication is not just limited in authenticity in network security as we described above. It also refers to other applications such as privileged access to services. When a device wants to access to some services, it needs to first prove its identity information and authenticate itself.

   Authorization refers to the access control. It ensures that the authenticated devices have permissions to perform specific operations and should not be able to perform other operations beyond the permissions. One special property in authorization is revocation which allows system administrators to remove permissions from devices.

   Accountability ensures that the operations of devices are bound to their authenticated identities. Thus, when a device becomes malicious such as performing hazardous behaviors or misusing services, it is easy to detect the identity of the malicious device. However, [162] claims that accountability is a particular challenge in IoT systems due to the amount of IoT devices, the magnitude of reuse of devices and services, access delegation, and the heterogeneity of data.

3. Privacy. Privacy in IoT systems includes data privacy, anonymity, pseudonymity, and unlinkability.

   The data privacy extends the definition of confidentiality in network security. In many IoT applications, devices need to collect personal information such as locations in order to provide better services. However, with enough personal information, which is called Personally Identifiable Information (PII), attackers may deduce the identity information of a person. Thus, data privacy must be protected that the data should not reveal any undesired properties.

   Anonymity describes that the identity information of a user or a device should not be leaked to other entities. For example, when a device wants to access some classified resource, the device proves its authorization but without providing its identity or password. Note that anonymity is the opposite of accountability.

   Pseudonymity provides a tradeoff between anonymity and accountability. In pseudonymity, a random pseudonym is associated with a device and operations are bound to the pseudonym rather than the real identity. Solutions such as anonymous credential systems support the pseudonymity in IoT system.

   Unlinkability is considered as indistinguishability of data and operations. In other words, for data and operations that are from the same entity, an attacker should not be able to link such data or operations together.

4. Trust. Trust requirement includes the entity trust and data trust.

   Entity trust refers to the trust of the expected behavior from entities such as devices, users, and services. For example, when a new device joins a system, the system needs to evaluate the risk of the new device. One possible solution to evaluate the trustworthy of an entity is trough trusted computing and reputation systems [76].

   Data trust refers to the trust of data which may be collected from untrusted devices. Therefore, before using the data, the system needs to apply some mechanisms to preprocess it (e.g., data aggregation [123]).

5. Resilience. Resilience refers to the robustness against attacks and failures. Due to the immature security solutions on resource-constrained devices, IoT systems are vulnerable to attacks. Therefore, to enhance the resilience, IoT systems usually leverage intrusion detection systems to provide protection against attacks. In addition, failover and recovery mechanisms are applied to help IoT systems to maintain service even the systems are under failure or attacks.

6. Efficiency. Khan and Salah [81] introduce the efficiency as an extra security requirement for IoT systems.

   Since IoT devices are usually resource-constrained in power and storage, algorithms and protocols must be lightweight such that they can perform well on IoT devices. In addition, attacks on IoT systems may increase in energy consumption and exhausting IoT resources.

In this report, we will focus on the security requirements of *Network Security* since it is the essential part to secure communications in IoT.

2.2 Secure Communication Channels

For communications in IoT systems, to achieve the security requirements that are described in section 2.1, the core component is to establish *secure communication channels* for entities in the systems. In general, a communication channel allows a message **sender** to input a message and a **receiver** to retrieve it [159]. To define the security properties of a communication channel, based on the adversarial model, we leverage the definitions from [159] to describe several different types of communication channels.

1. Insecure channels. An insecure channel leaks all messages that are transmitted between the sender and the receiver. In addition, attackers can modify the messages over the channel and determine the messages that are retrieved by the sender and the receiver.

2. Authenticated channels. An authenticated channel guarantees that all received messages are originally generated by the sender. Note that authenticated channels are different from the authenticity requirement in Network Security that is described in section 2.1. Here authenticated channels are more referred to the integrity requirement. Attackers can only learn the messages but have to forward the same messages correctly. Otherwise, if attackers modify the messages, entities who participate in the communication would detect the malicious behaviors.

3. Confidential channels. An confidential channel does not leak any useful information about the transmitted messages. Confidential channels refer to the confidentiality requirement in the Network Security that is described in section 2.1. Based on the capabilities of adversaries, there are two specific types of confidential channels.

   - Non-malleable confidential channels. In a non-malleable confidential channel, attackers are allowed to either forward the original messages or generate entirely "new" messages that will be retrieved by the receiver. That is, the new messages should be unrelated to the original messages.
   - XOR-malleable confidential channels. In a XOR-malleable confidential channel, attackers can only modify the original messages from the sender but are not allowed to input unrelated messages.

4. Secure channels. A secure channel implements both the authenticated channel and the confidential channel. In other words, a secure communication channel protects both message integrity and Confidentiality. Thus a secure communication channel does not guarantee all the security requirements of communication in IoT environments. For example, a secure channel may not provide authenticity or efficiency property.

Since IoT environments are unique in terms of uncontrolled environment, the heterogeneity, the need for scalability, and the constrained resources, the same security requirement may require different communication protocols on different network layers. Therefore, in the rest of this report, we describe the communication protocols on layered architecture of IoT in a bottom-up manner (from the physical layer to the application layer). In addition, as the core component to achieve secure communication in IoT is to establish secure communication channels, we emphasize our discussion on the transport layers of the protocol stack. In particular, we will focus on the key exchange protocols in IoT since a secure key exchange protocol is the core component to establish secure communication channels.

3 Physical Layer

In this section, we present the standard for communication protocols on the physical layer in IoT environments. In particular, we briefly introduce the IEEE 802.15.4 standard which defines the operations of low-rate wireless personal area networks (LR-WPANs). We also discuss certain security issues and threats towards the physical layer as well as solutions for physical layer protocols to achieve the security requirement as described in Section 2.

3.1 IEEE 802.15.4 Standard

The IEEE 802.15.4 standard [1] was originally introduced by the IEEE 802.15 working group and the latest version was released in 2016. This standard defines the specifications of physical (PHY) layer and medium access control (MAC) sublayer for low-rate wireless personal area networks (LR-WPANs) that demand low rate and low cost applications.

In general, the PHY defined in IEEE 802.15.4 should serve as an interface between the MAC sublayer and the physical radio channel by providing two fundamental services: PHY data service and PHY management service. The PHY data service can be accessed by the MAC sublayer through the physical layer data service access point (PD-SAP) when the MAC sublayer requests services from the PHY layer. The PHY management service can be accessed through physical layer management entity service access point (PLME-SAP) by the physical layer management entity to provide the layer management service.

Notice that in IEEE 802.15.4, the PHY layer itself does not provide any security service, it is only responsible for transforming the physical signal to the corresponding data for MAC sublayer. To offer security properties, the standard requires the PHY layer to cooperate with the MAC sublayer such that the MAC sublayer can provide different level of security such as encryption and message integrity code. We will explore more about the MAC sublayer in the next section.

3.2 Protocols in IEEE 802.15.4 Standard

Since PHY dose not offer any security service for communication, we randomly choose some standardized and widely used PHY layer protocols that are defined in IEEE 802.15.4 standard [1] and simply describe their technical specifications.

Phase-shift keying (PSK) is a fundamental digital modulation technique and it is used to convey signals in many radio communication systems such as wireless LANs, RFID and Bluetooth. It modulates the carrier wave by changing the sine and cosine inputs at a particular time. The simplest version of PSK is Binary Phase-Shift Keying (BPSK). In BPSK, 0 and 1 are represented by sinusoids segments which use two opposite signal phases (0 and 180 degree) to distinguish the data state shift. For example, if the phase of the wave changes by 180 degree, then the data state shifts from 0 to 1, or from 1 to 0. Thus, if phases of two segments are differ as much as possible, it is easier for BPSK to distinguish the data shift. Therefore, it is the most robust digital modulation technique. However, since BPSK can only modulate 1 bit for each segment, it is unsuitable for high data-rate applications. Usually BPSK operates in two frequency bands: 868 MHz band with the data rate of 20 kb/s and 915 MHZ of 40 kb/s.

Another variant of PSK is quadrature phase-shift keying (QPSK). QPSK is very similar to BPSK and widely used in various cellular wireless standards such as GSM, CDMA and LTE. QPSK also uses different phases of carrier wave to modulate the input bits. In contrast, rather than using two phases which shift of 180 degree from each other as in BPSK, QPSK uses four different phases in which phases shift of multiples of 90 degrees: $\frac{\pi}{4}$, $\frac{3\pi}{4}$, $\frac{5\pi}{4}$, $\frac{7\pi}{4}$. Therefore, QPSK usually has a higher data-rate than BPSK since each carrier wave in QPSK can represent two binary bits. However, more different phases in QPSK could also cause a higher bit error since two bits are transmitted simultaneously. Thus, the standard QPSK is less robust than BPSK. To this end, offset quadrature phase-shift keying (O-QPSK) is proposed to address the undesirable quality issue in communication systems. O-QPSK works the same as QPSK except that in O-QPSK, the timing for some input bits (odd bits or even bits) is made offset by one bit period. Consequently, the carrier wave in O-QPSK has much lower fluctuations than the original QPSK. In practice, O-QPSK usually operates in the 2450 MHz, 915 MHz, 780 MHz or 2380 MHz frequency bands with data rate of 250 kb/s and 100 kb/s in the 868 MHz band.

Amplitude-shift keying (ASK) is another prominent digital modulation technique in which only the amplitude of the carrier signal varies according to the digital signal changes. The simplest form of ASK is called on-off keying (OOK). In OOK, during a time period, the presence of a carrier wave indicates a binary one is transmitted and the absence of a carrier wave indicates a binary zero is transmitted. Similar to BPSK, in the standard OOK, it uses two amplitude levels to indicate one-bit binary value. However, if there are multiple possible amplitude levels, then more binary values can be represented. For example, in ASK of four possible amplitude levels, the carrier wave can represent a two-bit binary value: 00, 01, 10 or 11. Therefore, ASK usually has a higher transmission rate than BPSK. In general, ASK operates in the 868.0–868.6 MHz frequency band and in the 902–928 MHz frequency band with the date rate of 250 kb/s. Note that, if more amplitude levels are applied in ASK, the carrier wave can represent more binary values, depending upon the requirement. However, for the same reason in QPSK, more amplitude levels in ASK will also reduce the robustness of the communication systems. In other words, more amplitude levels can cause more bit error in transmission.

3.3 Vulnerabilities and Security

PHY security faces many challenges and it is difficult to employ traditional wireless physical layer security schemes in IoT environments. First of all, IoT devices usually have low data rate requirements, periodic data traffic arrivals, limited hardware and signal processing capabilities etc [104]. Secondly, at the sensor level, various factors has to be considered such as multi-path effects, fading, randomness, spatially distributed nature of the sensors and heterogeneity [152]. Moreover, fundamental assumptions for PHY such as adversarial model, the nature of the wireless channel, and the practical matters during implementations.

Thus, as IEEE 802.15.4 standard suggests, the PHY dose not provide any security service. PHY protocols should only responsible for specifying the interfaces between the MAC layer and physical radio channels. These interfaces include hardware specifications, PHY protocol data unit (PPDU) format, digits modulation technique, the radio frequency requirements, topology and physical network design, etc. If security services are desired, PHY should cooperate with upper layers to provide different security levels. For example, IEEE 802.15.4 standard describes different security levels for the MAC layers to add confidentiality (with encryption) and message integrity (with message authentication code). We will discuss more about the security service in the MAC layer in the next section.

3.3.1 Vulnerabilities on PHY Layer

Since PHY does not offer security services, most PHY protocols are vulnerable to attacks against confidentiality, integrity and availability. Besides the trivial attacks of eavesdropping and modifying messages that are transmitted on communication channels, another common attack on PHY is the jamming attack [112]. Unlike the most Denial of Service (DoS) attacks that happen on upper layers (e.g., application layer and transport layer), the jamming attack is a special DoS attack on lower layer of the IoT network protocol stack. In particular, a malicious device may deviate from the designated protocol and emit jamming radio frequency signals to the wireless medium to add noise in the carrier [63, 172, 131]. The noise would significantly reduce the signal-to-noise ratio to affect the transmitting signals such that legitimate IoT devices in the system may fail to receive the correct data. Note that the jamming attack can be conducted continuously or temporarily with random time intervals [21].

To mitigate the jamming attack, many detection and recovery mechanism are proposed. For detection, Young and Boutaba [175] suggested to measure the signal strength and compare it with a customized threshold. Xu *et al.* [172] presented an analysis of successful packet delivery ratio. They combine the signal strength measurements and location information to enhance the jamming detection. For recovery, error correcting codes is a common solution to control errors in data transmission over noisy communication channels. Some PHY protocols such as O-QPSK reduces the bit error by offsetting the timing of odd or even bits for one bit period. Noubir and Lin [112] also proposed a flexible concatenated code which combines the Reed-Solomon code and a short block code to maintain a low Frame Error Rate.

3.3.2 Security on PHY Layer

For PHY to provide security services, using upper layers has its limits. For example, cryptography-based solutions usually rely on mathematical hard problems (e.g., , integer factorization, discrete logarithm). These solutions also assume that adversaries should not have enough computational power to solve those mathematical problems which may not always true in the real world. In addition, the security services provided by upper layers usually require that communication devices share a common secret key. Traditional key establishment protocols such as Diffie-Hellman (DH) and Rivest–Shamir–Adleman (RSA) are based on the asymmetric cryptography which is considered too costly in both computation and energy. An alternative solution is to let each device in the system to have a pre-shared secret, installed a priori in the devices. Whenever two devices want to start a communication session, they derive the session key from the pr-shared secret. This solution limits the scalability of the system. Adding and removing devices from the system could require to update the firmware in all the devices [104]. In addition, a single compromised device would expose the pre-shared secret and threat the whole system.

Therefore, researches in recent years start to design security solutions on the PHY without a secret key. Rather than using encryption-based approaches, the emerged PHY security schemes generally rely on the information theory and signal coding strategies to support confidentiality and integrity on PHY. Thus, these schemes can supplement or even replace the cryptographic-based communication protocols. The basic assumption behind the key-less solutions is that, compared to communication devices, attackers always have a degraded channel. As depict in figure 2, the alternative security solution on PHY is based on the the wiretap channel which is introduce by Wyner [171] in 1975. In a wiretap channel, the honest sender Alice takes a plain input data M and wants to send it to the honest receiver Bob. Alice first encodes M into a formatted message X and then transmit it via the channel C1 which is called the main channel. After the transmission, Bob receives a message Y and decodes it. Note that Y could be different from X. At the meantime, an eavesdropper Eve is listening the communication via channel C2 which is called the wiretap channel or eavesdropper channel. The message Eve receives via C2 is denoted as Z. To achieve secure communication, the transmission needs to guarantee that Z leaks no information about M and Y can be decoded to correct M with a high probability. Therefore, if the signals transmitted on channels C1 and C2 are different in terms of some conditions, Alice and Bob could achieve a secure communication.

The existing physical layer security approaches without a secret key can be categorized into three different methodologies [56] as follows.

1. Secure coding algorithms. Two popular coding schemes used in PHY security are low-density parity-check (LDPC) codes and polar codes.

    - LDPC is a linear error correcting code that is used to transmit messages over a noisy channel. Thangaraj *et al.* [160] showed that it is possible to construct linear-time decodable secrecy codes by using LDPC codes that achieve secrecy. Liu *et al.* [95] improved the work in [160] and presented a new nested code structure with a new achievable secrecy rates for the type II Gaussian wiretap channel where the main channel is noiseless and the eavesdropper channel is a general binary-input symmetric-output memoryless channel. For the Gaussian wiretap channel, Klinc *et al.* [82] also proposed a new coding scheme that is encodable in linear time and applicable at

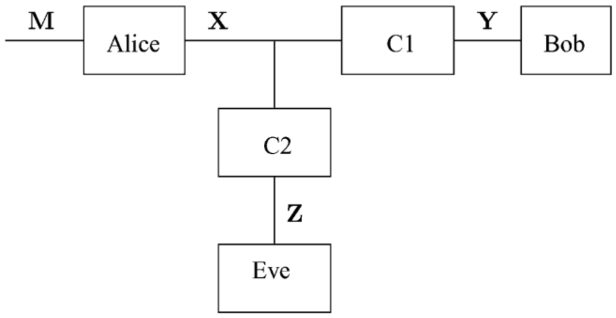

Figure 2: The number of messages drops when t is close to n and p_c is large.

finite block lengths. More importantly, they showed that their scheme is compatible with existing cryptographic solutions to provide improved data security by taking advantage of the statistical nature of communication channels

- The second widely used coding scheme is polar codes. It is also a linear block error correcting code and introduced by Erdal Arikan. Mahdavifar and Vardy [98] constructed a new coding scheme based on polar codes for a wide range of wiretap channels. Their construction works for any instantiation of the wiretap channel model, as long as both both main and wiretap channels are symmetric and binary-input, and the wiretap channel is degraded with respect to the main channel. However, the coding scheme in [98] cannot achieve both strong security and reliability at rates approaching the secrecy capacity. Thus, Sasoglu and Vardy [142] presented a multi-block polar coding scheme for symmetric degraded wiretap channels. Their coding scheme has polynomial time in encoding and decoding while guarantees both strong security and reliability.

2. Channel-based adaptation transmission. This solution is based on the difference of the channel fading rates between the legitimate message receiver and the eavesdropper. If the transmitted signal for the receiver is assumed to be better than the signal for the eavesdropper, then by designing an adaptive transmission scheme, the communication channel can achieve the data security at a certain rate. For example, Hamamreh and Arslan [55] presented a secure waveform design for future 5G wireless system. In their design, signal is transmitted and received in *time domain*. In particular, the new waveform degrades eavesdropper's reception by using secure orthogonal transform division multiplexing to carry and extract data. Other solutions are based on *frequency domain* in which data is transmitted and received over multiple sub-carrier frequencies [96], and *space domain* in which data is transmitted via several spatial transceiver sources such as antennas [64].

   Channel-Based Adaptation Transmission solution has its advantage and disadvantage for resource-constrained devices. On one hand, it does not employee extra computational operations. Thus, it saves resources which makes it suitable for IoT devices. On the other hand, this solution cannot guarantee the perfect secrecy. Partial information, if not whole, could be leaked to adversaries.

3. Noise signals along with the transmitted signals. In this solution, legitimate devices add artificial noise to the original signal such that eavesdropper's channel would be too noisy to recover the original signal. At the mean time, legitimate devices can still send and receive data correctly with a high rate. For example, Hussain *et al.* [69] suggested to have the sender generate some random artificial noise that appeared to be zero at the receiver side at some specific time while degrading correctness at eavesdropper side. Similar idea was also proposed by Akitaya *et al.* [2]. However, rather than removing noise at some specific time, the noise generation scheme in [2] let the receiver remove the noise by analyzing the frequency.

   The solution of adding noise to the original signal also has its advantages and disadvantages Contradict to the Channel-based adaptation transmission solution, adding noise can achieve perfect secrecy. In addition, we can combine this solution with cryptography-based solutions to provide a higher security level. However, adding noise increases the power consumption on the sender side since the sender needs to add artificial noises. Moreover, it could reduce the performance on the receiver side since the noise also affects the main channel. If the noise causes too many errors, the sender may need to retransmit the data multiple times.

4 Medium Access Control Layer

4.1 IEEE 802.15.4

As for the PHY, the IEEE 802.15.4 standard [1] also describes the specifications for the Medium Access Control (MAC) layer. In particular, the MAC layer implements an interface between the PHY and the network layer to support the MAC management service and the MAC data service.

Unlike the PHY which dose not provide any security service for communication in IoT, the MAC layer offers different levels of security to support optional data confidentiality, data authenticity, and replay protection in order to minimize the overhead. For each frame on the MAC layer, if any security service is provided, the Security Enabled field in the frame should be set to one to indicate the existence of the Auxiliary Security Header. In the Auxiliary Security Header, the 3-bit Security Level field further indicates the specific security level for the communication.

As depict in Table 1, there are 8 different levels of security. For level 0, frames are sent without any protection of data confidentiality or data authenticity. Level 1 to 3 and level 5 to 7 leverage Message Integrity Code (MIC), also called Message Authentication Code (MAC) or authentication tag, to protect data authenticity. The numbers after MIC in the table (i.e., 32, 64, 128) denote the length of MIC. Level 4 applies AES counter mode (AES-CTR) to provide data confidentiality service. Only level 5 to 7 provide encryption (ENC) and MIC services for the payload part of the MAC layer to ensure both the data confidentiality and data integrity. Note that for a MAC layer frame, the value 0 in the Security Enabled field or the security level 0 indicates that the security services are optional for IoT devices. In other words, if an IoT device does not implement security services, it should not apply any cryptographic operations on the MAC layer for incoming and outgoing frames.

Table 1: Security levels in IEEE 802.15.4 standard

Security level	Attributes	Data confidentiality	Data authentication
0	None	✗	✗
1	MIC-32	✗	✓
2	MIC-64	✗	✓
3	MIC-128	✗	✓
4	ENC	✓	✗
5	ENC-MIC-32	✓	✓
6	ENC-MIC-64	✓	✓
7	ENC-MIC-128	✓	✓

To protect data authenticity and confidentiality, the IEEE 802.15.4 standard specifies the MAC layer apply the extension of counter mode (CTR) encryption and cipher block chaining message authentication code (CCM*) which is a variant of the Counter with Cipher Block Chaining - Message Authentication Code (CCM) mode.

For data confidentiality, data are encrypted with advanced encryption standard (AES)-128 [41]. The advantage to use AES-128 on the MAC layer is that most IEEE 802.15.4 compatible hardware platforms implement the hardware acceleration for AES-128 to improve the performance and reduce the energy cost [39]. Even if the MAC layer does not employ any security services, the higher layer can still benefit from the hardware acceleration of AES-128 to support confidentiality.

For data authenticity, CCM* is similar to the standard cipher block chaining message authentication code (CBC-MAC). Specifically, CCM* first encrypts the data with AES-128 in CBC mode to create a chain of blocks. However, rather than having the whole last block as the authentication tag, CCM* only uses the leftmost M octets where M has the possible values of $0, 4, 8, or 16$. In other words, M is the size of the MIC code following each message. It is easy to see that smaller value of M can save the size of transmitted messages but reduce the security level as well. Note that to achieve both data authenticity and confidentiality, CCM* generates the authentication tag first and then encrypts both the input data and the authentication tag.

A prerequisite to apply CCM* on the MAC layer is to have the communication IoT devices to share on a common *secret key*. The communication devices use the shared key to generate and verify the authentication tag as well as encrypt and decrypt data. However, even though the MAC layer protocols have a KeyDescriptor lookup procedure to search for the corresponding key, it does not specify how to conduct key exchange for IoT devices in order to share on a common secret key. Instead, the key exchange task is left for the upper layers and we will discuss it in details in Section 8.

The IEEE 802.15.4 standard also provides access control service and protection against replay attack. Devices follow IEEE 802.15.4 standard store an access control lists (ACL) in which contains specific information of destination devices. Thus, only designated devices can join the communication. Also, IEEE 802.15.4 standard defines the *The Frame Counter* filed to specify the unique message ID. In order to support replay protection, the value in Frame Counter is increased for each transmitted message until it reaches a pre-defined maximum value.

4.2 Near-Field Communication Protocol

The Near-Field Communication (NFC) [22] protocol is a special type of Radio-frequency Identification (RFID) protocols. An RFID protocol contains two components: tags and readers. An RFID tag transmits its identity and other digital data to a nearby reader (e.g., a few inches), which is a device that emits radio waves and reads signals back from the RFID tag.

NFC is a short-range wireless technology that allow two IoT devices to communicate with each other by using low-power transmission links. NCF protocol is based on a radio frequency field and it operates at a frequency of 13.56 MHz band. IoT devices wish to establish communication using NFC protocol must be either physically touching or physically close within a few inches of each other to start communication. NFC-enabled devices transmit data through electromagnetic fields and operate either passively or actively. In passive operation, a device stores data and transmits data to other devices which actively generate the electromagnetic field. In active operation, NFC devices generate their own electromagnetic fields and interacts with other NFC-enabled devices.

The standard NFC protocol does not provide any cryptographic solutions to protect security requirements. Devices using NFC protocol are vulnerable to attacks that against confidentiality and integrity in building secure communication channels. Attackers can easily eavesdrop and modify data that are transmitted over the network since all data are in plaintext. Therefore, higher layers are responsible to provide security solutions in order to achieve secure communication. For example, the application layer can encrypt the data before moving it to the lower layers.

On the other hand, the proximity nature of NCF protocol prevent some attacks from happening. Since the communication devices have to be either physically touching or physically close to each other (e.g., 10 centimeters in the real scenarios), a passive external adversary must also be close enough to reliably eavesdrop the transmitted data. Thus, communication devices would be easily to detect the attacker.

A recent technical specification of NFC protocol from NFC Forum in 2015 announced Signature Record Type Definition (RTD) which specifies how to protect the integrity and authenticity of NDEF (NFC Data Exchange Format) Messages. RTD leverages the certificate techniques and the digital signature schemes to verify the identity information and the integrity of transmitted messages. Currently, RTD 2.0 supports two certificate formats of X.509 certificate format and the Machine to Machine (M2M) Certificate formats. Also, the standard digital signature algorithms used in NFC include RSA, DSA, and ECDSA. All three algorithms use SHA-256 for hashes and the security strength ranges from 80 bits to 128 bits.

4.3 Low Power Wide Area Networks Protocols

Low Power Wide Area Networks Protocols (LPWAN) is a type of wireless telecommunication wide area network for battery-powered IoT devices to send small amounts of data over great distances (e.g., several miles) for a long time (e.g., years).

Long Range Wide-Area Network (LoRaWAN) [153] protocol is a special low power, wide area communication protocol designed for LPWAN. LoRaWAN was introduced and maintained by LoRa Alliance which is a non-profit association committed to enabling large scale deployment of LPWAN. LoRaWAN operates on radio frequency bands of 433 MHz, 868 MHz for Europe, 915 MHz for Australia and North America, and 430 MHz for Asia. The data rates range from 0.3 kbp/s to 50 kbp/s. In particular, the LoRa network can control the data rate and radio frequency output for each device individually [153].

LoRaWAN usually has a star-of-stars topology [153]. In LoRaWAN, gateways are responsible to relay data from end-devices to a central network server which then forwards data packets from each device to other devices or application servers. Gateways connect and communicate with the central server via standard IP connections while end-devices connect with gateways use single-hop LoRa or Frequency-shift keying communication.

To secure the communication, LoRaWAN defines two security layers. The first layer is between the end-devices and network server. An end-device and the network server share a unique 128-bit Network Session Key (NwkSEncKey) which is derived from a specific *root key*. The second layer is between the end-to-end at the application level. Each device and an application server share a unique 128-bit Application Session Key (AppSEncKey) which is also derived from the root key.

On the MAC layer, if a data frame carries a payload, before calculating the MIC, LoRaWAN must encrypt the payload part to protect data confidentiality. The encryption scheme is based on AES with a key size of 128 bits as described in IEEE 802.15.4. Specifically, LoRaWAN encrypts each block of data with a secret key using AES-128 scheme and then perform the XOR operation between encrypted blocks and the payload. Note that the key used for encryption depends on the Port field (FPort) in the payload. The value 0 of FPort indicates the payload part only contains MAC commands and NwkSEncKey is applied. Otherwise, if FPort has value between 1 to 255, the payload part contains application-specific data and AppSEncKey is applied.

For data integrity, LoRaWAN calculates the authentication tag over all the fields in a message. The message is a concatenation of MAC Header (MHDR), frame header (FHDR), port field (FPort), and frame payload field (FRMPayload). LoRaWAN then applies the AES Cipher-based Message Authentication Code (AES-CMAC) algorithm on the message with the key NwkSEncKey. AES-CMAC is similar to the Hash-based message authentication code (HMAC). The difference is that AES-CMAC is based on a symmetric key block cipher AES while HMAC is based on a hash function such as SHA-256. At the end, LoRaWAN truncates the first four octets as the final MIC to ensure the data integrity.

4.4 Vulnerabilities and Security

Different from the PHY, the MAC layer starts to provide different level of security services. However, even MAC layer protocols may support confidentiality and integrity, communications on MAC layer still face many attacks and challenges. For example, Ouyang *et al.* [114] show that even data is encrypted, attackers can still retrieve private information (e.g., user location) by analyzing the traffic pattern. If the location is associated with a specific user, the personal identity information would also be leaked to users. Here we present three main attacks that against MAC layer communication.

4.4.1 Vulnerabilities on MAC Layer

We identify the the first type of attacks that against MAC layer communication cause the energy consumption issue. Since IoT devices are usually battery-powered and energy-constrained, the sleep deprivation torture [100, 130] attack for wireless sensor networks can result in depletion of battery of IoT devices. Another attack in [12] keeps devices to stay awake and execute unnecessary tasks to drain the battery energy in the IPv6 over Low power Wireless Personal Area Networks (6LoWPAN). Stajano and Anderson [155] also show that by repeatedly sending handshaking messages, a device is forced to stay in activation mode until deplete its battery. To address the energy consumption issue, Liu *et al.* [176] showed how to use the MAC layer traffic to identify abnormal nodes. In particular, they generate a set of features from MAC layer and apply cross-feature analysis on feature vectors to effectively detect the sleep deprivation attack.

The second type of attacks happen on the MAC layer is denial of service. A DoS attack can prevent legitimate IoT devices from communicating with each other or accessing resources. A common DoS attack technique is to simply flood the MAC layer. For example, a malicious device can send overwhelming MAC data packets and MAC control packets to its neighbor devices such that victim nodes fail to process benign packets and also deplete its battery. Other DoS attack techniques are collision and link layer jamming. Attackers try to analyze the packet sending and arrival times [178] of a benign device, and then transmit packets simultaneously on the same frequency channel as the benign device [17, 5]. In the end, due to the collision of packets, the benign device will discard the received packets and ask for retransmission. The defense strategy against denial of service is to deploy Intrusion Detection System (IDS) or an Intrusion Prevention System (IPS). An IDS/IPS usually deploys at the gateway of IoT systems to monitor malicious activities and policy violations. When an intrusion is detected, a mitigation mechanism would be invoked next to reduce the impact of malicious behaviors. The IDS we described above [176] can also be applied to prevent DoS attacks. OConnor and Reeves [113] present a misuse detection-based Bluetooth IDS against DoS attacks. Their Bluetooth IDS can check the Bluetooth traffic through pattern matching and a set of plug-in modules to detect malicious behaviors.

The last type of attacks is the spoofing attack. To lunch a spoofing attack, attackers can spoof the MAC addresses of other devices and create fake identities in the system in order to access sensitive information [122]. Eldefrawy *et al.* [42] present a new model Scyther to formally analyze the security of LoRaWAN. They show that LoRaWAN is vulnerable to the spoofing attacks since the lack of synchronization between communicating devices. To prevent spoofing attack, Demirbas and Song [34] suggest to use sender location to identify fake MAC addresses. If messages are from the same location but with different identities, then it indicates spoofing attack is happening. Li *et al.* [116] introduce a key-changed mutual authentication protocol for WSN and RFID systems. The protocol integrates a random number generator in the tag and reader to reduce the risk of spoofing. Chen *et al.* [27] propose a K-means cluster analysis to measure the signal strength and use the result to built a real-time localization system to locate the position of attackers. The experimental result in WiFi network and ZigBee network show that the proposed K-means cluster analysis can identity spoofing with high detection rates of 95% and low false positive rate of 5%.

Other attacks toward the MAC layer communication such as de-synchronization, exhaustion, and unfairness are also identified in [21]. Most attacks on the MAC layer protocols target to the availability, resilience, and efficiency requirements in IoT systems.

4.4.2 Security on MAC Layer

As described in IEEE 802.15.4, most MAC layer protocols should leverage the encryption scheme to ensure data confidentiality and the MIC algorithms to protect data integrity. In particular, AES is the widely used algorithm on MAC layer to encrypt data (e.g., AEC-128-CTR) and generate MIC (e.g., AEC-CBC-MAC). There are three reasons to choose the AES algorithm in IoT. First of all, AES is proven to be secure and robust. For now, there is none algorithm can efficiently break AES. Secondly, AES itself is considered as efficient for IoT. AES is a symmetric key algorithm and most operations in AES are bit operations rather than group operations as in public-key algorithm. Lastly, many IoT device manufactures implement the AES algorithm at the hardware level to accelerate the execution speed of AES for their devices. Altolini *et al.* [6] showed that hardware acceleration of AES saves the memory usage for up to six times and reduce the delay for up to two orders of magnitude compared to their software implementation.

For protocols that do not support AES at the MAC layer, they may have other methodologies to ensure security. For example, the proximity nature in NFC prevent attackers from eavesdropping messages in a long distance. Some wireless network technologies such as Wireless-Hart and Sigfox try to keep their security solutions hidden from the public and limit access to their techniques details.

Even most MAC layer protocols provide good security services, there are still some crucial aspects are not addressed well on the MAC layer. The most significant one is the key establishment. AES algorithm requires two communication devices to share a common secret key. However, MAC layer protocols do not define the standard about the key generation procedure or how to perform key exchange between devices. Usually the key establishment task is left to the upper layers. One common solution on MAC layer is to use a pre-shared secret which is installed a priori among devices. The key for each communication session will be derived from the secret. This solution suffers from many drawbacks. For example, if one device in the system is compromised, all communications will be no longer secure. Other key establishment schemes will be discussed later. Another noticeable issue on MAC layer is the joining procedure for new nodes without security capabilities [140]. When a new node wants to join a existing secure IoT system, if the node does not support AES or other security algorithms, the MAC layer protocols do not explain how to achieve security for the new node or the system. Moreover, a message leaves the IEEE 802.15.4 network and continuing transmitted over IP network may not be protected by the link layer security mechanisms [126]. Therefore, extra security mechanisms such as ArchRock PhyNET and HIP DEX [65] should be specified to protect the data security between IEEE 802.15.4 network and IP network. Lastly, security is optional in IEEE 802.15.4 standard, users may fail to apply any security service because of human errors.

Security on the MAC layer provides a strong guarantee for secure communications. Security at upper layers is not required if security is enabled on the MAC layer. Indeed, security on the MAC layer is usually considered more secure than upper layers since it can protect the MAC layer data and upper layer headers as well.

5 Network Layer

The network layer protocols provide end-to-end communications and is responsible for packet forwarding and packet routing. In this section, we examine three standardized protocols that can help achieve secure communication on the network layer.

5.1 Internet Protocol Security

The Internet Protocol Security (IPsec) [144] is a network layer protocol to provide secure data transmission over untrusted networks. IPsec was originally developed for IPv6 to provide a set of security services for communication on the network layer, it is also compatible with IPv4 networks.

IPsec can provide confidentiality, integrity, data-origin authentication, and replay protection for IP data packets. In particular, IPsec includes two protocols to secure the communication: Authentication Header (AH) and Encapsulated Security Payload (ESP). AH protocol provides integrity, data-origin authentication and replay protection, while ESP may provide confidentiality, integrity, data-origin authentication and replay protection. Note that the confidentiality in ESP is optional and can be disabled to reduce the power consumption.

IPsec can be implemented in either transport mode or tunnel mode. In transport mode, only the payload part of the packet is protected (encrypted and/or authenticated) and the IP header is left intact. If AH is applied in transport mode, the IP information and payload cannot be modified since any modification would invalidate the hash value of the packet. Thus, integrity and data-origin authentication are guaranteed. In tunnel mode, the entire IP packet is protected and encapsulated into a new IP packet with a new IP header. Tunnel mode is widely used to create Virtual Private Networks (VPN).

IPsec uses cryptographic security services to protect communications on network layer. It supports many different cryptographic primitives and allows devices choose the most appropriate ones based on their requirements. For integrity and data-origin authentication, IPsec offers both hash-based and encryption-based schemes. Five algorithms are suggested in RFC 4301 [144] standard: HMAC-MD5-96, HMAC-SHA1-96, DES-MAC, KPDK-MD5 and AES-XCBC-96. For confidentiality, IPsec provides many encryption schemes such as AES-CBC, AES-CTR, BLOWFISH, etc. For combined encryption and authentication, IPsec uses AES-CCM and AES-GCM. Note that some algorithms are considered as not secure anymore even they are supported by IPsec. For example, DES-CBC have been shown to be easily broken due to its short key length. In general, AES based algorithms are preferred for both security and efficiency considerations.

Unfortunately, secret key establishment is still the challenge in IPsec to provide security services. All cryptographic primitives used in IPsec require two communication devices to share a common secret key. The key exchange information and the selected cryptographic primitives used by IPSec are contained in IPsec Security Association (SA). SA can be pre-configured or created by the The Internet Key Exchange (IKE) protocols. However, pre-configuration solution limits the scalability while IKE protocols require public key cryptography which is not suitable for IoT devices. Therefore, efficient key exchange protocols are on demand for IPsec.

Moreover, IP-based protocols may not suitable for IoT network. This is usually due to the IoT-specific limitations in packet overhead. Especially for IPv6 related protocols, the packet header is much larger than

IPv4 packet header. It may not be possible to encapsulate IPv6 packets into the MAC layer frames in IoT network. Therefore, some adaption protocols are required for IoT networks to enable IPv6 protocols operate on IoT networks. One such protocol is the IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) which we will describe next.

5.2 IPv6 over Low-Power Wireless Personal Area Networks

6LoWPAN [103] is an intermediary protocol sit between the network layer and MAC layer. One fundamental characteristic of network layer protocols is to enable packets traverse through interconnected networks using different MAC layer protocols. However, one problem for IPv6-based IoT networks is the overhead of the IP header. Since IPv6 header is much larger than IPv4 header, it maybe be difficult to fit an IPv6 header into a single MAC layer frame for Low-power Wireless Personal Area Network (LoWPAN) devices that operate on LoWPAN MAC layer protocol such as IEEE 802.15.4. In particular, the data payload for protocols from higher layers to MAC layer in IEEE 802.15.4 standard is limited to 102 bytes which is not enough for IPv6 packets. Thus, 6LoWPAN is designed to optimize the usage of the limited payload space such that LoWPAN MAC layer protocols can operate with IPv6. The initial 6LoWPAN specification defines the frame format and describes the packet header compression scheme to encapsulate IPv6 packets into the 802.15.4 frames.

However, the official 6LoWPAN standard does not specify any security mechanisms. The specification RFC 4944 [102] only describes the interest of applying security mechanisms in the the 6LoWPAN protocol. A general way to achieve security for 6LoWPAN is to cooperate with MAC layer protocols (e.g., IEEE 802.15.4) or upper layer protocols (e.g., DTLS). Another research trend in recent years is to extend the 6LoWPAN protocol to enable IPsec communication on LoWPAN with IPv6 devices. We will describe more later in this section.

5.3 IPv6 Routing Protocol for Low-Power and Lossy Networks

As the network layer protocols should be responsible for packet forwarding and packet routing, traditional routing protocols cannot be directly applied in IoT network. This is usually due to the unique properties in IoT network such as frequent topological changes and constrained resources available in IoT. Traditional routing protocols provide strict routing that may waste both network and device resources. Therefore, secure end-to-end communications in IoT require new routing strategies that can make use of efficient optimization techniques to save resources.

Many IoT oriented routing protocols have been introduced in last decades. For example, in early flood-based routing protocols such as Dynamic Source Routing [75] and Ad-hoc On-demand Distance Vector Routing [117], routing information is propagated over the whole network until it reaches the destination. However, these flood-based solutions cause too much overhead for IoT networks and do not provide security services.

To this end, IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [4] was proposed to address the overhead and security issues in routing for resource constrained networks. RPL topology maintains a Destination Oriented Directed Acyclic Graph (DODAG) tree with a single root which is also called the sink node. The DODAG uses ranks to describe the quality of route to the sink node. The rank value for each node is calculated with respect to its parents rank value and other relevant metrics such as latency, bandwidth, energy, etc. Finally, the DODAG contains all the routing information and determines the routing tables. Each node in the network can use DODAG to determine whether forward packets upwards to their parents or downwards to their children.

The RPL specification supports three security modes to protect RPL messages: unsecured, preinstalled, and authenticated.

Unsecured mode indicates that all security properties are optional in RPL. Without security properties, RPL implementation can save the total overhead for IoT devices. Note that unsecured mode does not imply RPL messages are not protected. Indeed, RPL could use the MAC layer protocols (e.g., IEEE 201.15.4) to achieve required security properties. However, if neither MAC layer protocols or PHY layer protocols provide any security services, there are many possible attacks against RPL protocol.

The second mode is the preinstalled mode. In this mode, a new device must have a preinstalled key to join an RPL network as either *a host or a router.* By using the preinstaled key, RPL provides different levels of security to protect confidentiality, integrity and authenticity. As shown in Table 2, level 0 and level 2 protect data integrity and authenticity while level 1 and level 3 protect data confidentiality, integrity and authenticity. For confidentiality, RPL supports the CCM with AES-128 as the encryption scheme. For integrity and authenticity, RPL supports both MIC solution and digital signature solution. In particular, RPL uses CBC-MIC (AES-128-CBC) to generate the authentication tag of size 4 or 8 bytes (Table 2a). If digital signature is applied, RPL signs the message with RSASSA-PSS and the signature length could be 384 or 256 bytes (Table 2b). RPL also provides protection against replay attack by maintaining *counters* among RPL nodes. RPL nodes can use Consistency Check (CC) messages which contains randomly generated CC nonces to validate and synchronize counters. Therefore, attackers cannot replay messages since they cannot guess the correct CC nonce.

Table 2: Security levels for RPL with MIC and signature

Security level	Attributes	MIC length
0	MIC-32	4
1	ENC-MIC-32	4
2	MIC-64	8
3	ENC-MIC-64	8
4-7	Unassigned	N/A

(a) RPL seurity with MIC

Security level	Attributes	MIC length
0	Sign-3072	384
1	ENC-Sign-3072	384
2	Sign-2048	256
3	ENC-Sign-2048	256
4-7	Unassigned	N/A

(b) RPL seurity with signature

Authenticated mode also requires a new device to have a preinstalled key before joining an RPL network. Different from the preinstalled mode, the new device can only join as a host with the preinstalled key. If it wants to join as a router, the key authority must authenticate the device first and then derives a second key for the new device. However, RPL does not specify how to perform authentication for the key authority. In addition, authenticated mode cannot be supported by symmetric key cryptography. Thus, authenticated mode is only designed for potential future cryptographic primitives.

5.4 Vulnerabilities and Security

The fundamental functionality of the network layer protocols for IoT is to allow packets to traverse interconnected networks using heterogeneous MAC layer protocols. Similar to IEEE 802.15.4 standard, most standardized network layer protocols provide security services of confidentiality and integrity. Moreover, protocols such as IPsec and RPL also specify key exchange schemes (e.g., pre-installed key, IKE protocols) and provides authenticity. Therefore, with an efficient secure key management scheme, authentication process, and proper implementations of network layer protocols, IoT networks achieve secure communications that can protect data confidentiality, integrity and authenticity. Despite message security with existing solutions such as IPsec, IoT devices are still vulnerable to network disruptions. Most effective attacks on network layer are targeting the availability and efficiency properties.

5.4.1 Vulnerabilities on Network Layer

We investigate two types of attack against the availability on network layer and also discuss the corresponding countermeasures for those attacks.

- The most common attack against availability on network layer is the DoS attack which can prevent legitimate devices from accessing network resources. For example, in some routing protocols, an attacker broadcasts HELLO messages to the whole network with high enough signal power in order to pretend to be the neighbor of every node in the network [77]. When updating the routing information, legitimate nodes would send their packets to the attacker instead of their real neighbors. In the end, packets may simply dropped by the attacker or get lost because the attacker might be out of range. On the other hand, this attack usually cannot exist for long time in networks. Wallgren *et al.* [165] showed that the HELLO-flooding attack in RPL protocol can affect almost all nodes in the network for 10 minutes before the attack is automatically mitigated. Another common DoS attack is the distributed

DoS (DDoS) attack. One of the most famous IoT botnet used for DDoS attack is the Mirai botnet which is composed primarily of embedded and IoT devices. Antonakakis *et al.* [8] analyzed the Mirai botnet and showed how Mirai emerged, what classes of devices were affected and how Mirai variants evolved. According to their study, the Mirai botnet launched tens of thousands of DDoS attacks in seven months against different targets such as online games, political sites, etc.

A popular solution that against DoS attack is to introduce IDS which monitors network states and identify the attacks and attackers. However, traditional IDS usually cannot be directly deployed due to the resource constrained devices. Instead, Kasinathan *et al.* [78] presented an hybrid IDS framework for IoT in 6LoWPAN networks. Their framework added additional detection modules that contain frequency agility manager and security incident and event management system to monitor the attack events and alerts. Raza *et al.* [129] introduced a similar hybrid IDS SVELTE for IP-based IoT devices that use RPL as a routing protocol in 6LoWPAN networks. . SVELTE balances the storage cost for signature based detection and the computing cost for anomaly based detections, thereby applies different detection strategies to target various DoS attacks.

- The second type of attack towards the routing security. As the current approach to routing in 6LoWPAN environment is defined by RPL, an attacker can launch a number of attacks against the RPL to interrupt the network. Le *et al.* [91] presented Rank attacks by changing the rank value which is used by each node to select its parent. A compromised node can intentionally modify the rank value in DODAG to attract child nodes for selecting it as parent, and thereby attract large traffic going toward the root. A similar attack to affect the routing path is the Sybil attacks [108]. In Sybil attack, a malicious node could claim arbitrary number of logical identities to other nodes in the network. Consequently, an attacker may fool other nodes and control large parts of a network without deploying physical nodes. This attack could reduce the effectiveness of fault-tolerance schemes and poses a significant threat to geographic routing protocols [21].

  To mitigate the security threats in RPL, the authentication of rank value and verification of identities should be considered. Dvir *et al.* [40] presented the *Version Number and Rank Authentication* (VeRA) security scheme to protect the rank values in DODAG. Specifically, VeRA allows the root node in DODAG to use hash chains, MIC, and digital signature to authenticate rank values and intermediate nodes can check if the Version Number modified by the root node or by malicious nodes. When a node receives routing information from the DODAG root, it verifies the corresponding Version Number hash chains, MIC, and digital signature and then broadcast the information to all neighbors. When updating the DODAG, an intermediate node can verify if the rank value of its parent is monotonically increased by checking if the current MIC value is from the previous update. While VeRA prevents malicious nodes from forging the relative topological distance to the root, it is vulnerable to rank replay attack. Attackers can simply reuse the version number hash and its rank value during the DODAG update, thereby pass the verification phase and make other nodes select it as the parent.

  To fix this issue, Glissa *et al.* [48] introduced Secure-RPL (SRPL) which is an extension of the RPL protocol. SRPL is also based on hash values to provide authentication in DODAG. Different from VeRA, SRPL sets a threshold to limit the rate of increasing and decreasing the rank value. With the threshold, SRPL monitors the number of times a RPL node is increasing its rank value. If a node changes its rank value that exceeds the threshold during the update, other nodes will discard this node and consider it as an attacker.

  The countermeasures for the Sybil attack is to validate the legitimate identities of every node and limit Sybil identities. To detect Sybil nodes, Zhang *et al.* [177] presented the social graph-based Sybil detection (SGSD) scheme in which legitimate nodes can traverse social graph through random walks. If a walk reaches a known honest node, the honest node verifies this walk and also the nodes on this walk. When there are multiple walks (the number is a pre-defined threshold) pass through a suspicious node and reach the known honest node, the suspicious node would be accepted as an honest node. Otherwise, the suspicious node is considered as a Sybil node. Another idea is to use community detection algorithms. Each suspicious node is ranked according to its social connections with the known trusted nodes [164] or determined by global vote aggregation to estimate the probability of being a Sybil node [173].

5.4.2 Security on Network Layer

Compared to MAC layer protocols, network layer protocols provide a more general end-to-end security. This is because the existing end-points on the Internet can secure their communications without worrying about the underlying network architecture. Especially for IP-based IoT devices, packets can traverse the Internet using heterogeneous MAC layer protocols. Even lower layer protocols do not support any security services, devices can still secure their communications on the network layer. On the other hand, securing communication on network layer usually is more expensive than directly securing the communication on MAC layer. In IPsec, extra headers of IPsec AH and ESP are encapsulated into the packet and consequently increases the total packet size. Moreover, the cryptographic algorithms used in network layer protocols might run on general purpose hardwares, thereby cause more processing time to implement those cryptographic algorithms.

The alternative solution RPL is more efficient than IPsec in IoT environment. RPL provides flexible security levels for different security requirements. It allows users to specify the effective cryptographic algorithms for encryption and authentication. However, RPL does not support IP-based IoT devices. To provide end-to-end security across RPL network and the Internet, a trustworthy gateway is required for transformation between RPL packets and IP packets.

IPsec is a more generic approach to provide security services on the network layer. Since IPsec was originally developed for IPv6 to provide end-to-end security, a better solution is to enable IPsec in resource-constrained IP-based IoT devices. First of all, it provides key exchange mechanisms and provides authentication in addition to confidentiality and integrity. In addition, IPsec is compatible with any transport protocol and ensures the security for transport layer headers and IP headers. Lastly, Granjal *et al.* [53] evaluated the performance of several well known cryptographic algorithms on real sensor nodes and the results have shown that IPsec is a viable option for wireless sensor network as long as as platforms support efficient hardware security optimizations. Therefore, the research community and 6LoWPAN standardizations groups consider IPsec a potential security solution for the IoT [126].

As 6LoWPAN protocol describes the packet header compression scheme to encapsulate IPv6 packets, it is desirable to combine 6LoWPAN and IPsec to ensure secure communication on network layer. However, standard 6LoWPAN protocol does not specify the header encodings schemes for AH and ESP extension headers. The challenge to apply the solution of standard IPsec in 6LoWPAN environments is related to the resource constraints of IoT devices. Thus, IPsec needs header compression schemes to reduce its packet size in order to work in 6LoWPAN. Raza *et al.* [125] introduced the first 6LoWPAN extension that supports both AH and ESP for IPsec. The header encoding scheme for AH can reduce the packet overhead from 24 bytes to 16 bytes for HMAC-SHA1-96. Also, the ESP encoding reduces the packet overhead from 18 bytes to 12 bytes for AES-CBC (without authentication), and 30 bytes to 24 bytes for HMAC-SHA1-96 (with authentication). Their evaluation results demonstrated that it is possible and feasible for IP-based IoT devices to use compressed IPsec to secure communications in the Internet with IPv6 mechanisms. In their following work [126], Raza *et al.* detailed their encoding scheme for ESP in 6LoWPAN. Moreover, they implemented and evaluated the compressed IPsec for 6LoWPAN and also IEEE 802.15.4 MAC layer security. The evaluation results showed that IEEE 802.15.4 security is more efficient for the end node while IPsec is more efficient for forwarding nodes. Moreover, IPsec scales better than IEEE 802.15.4 security in their experiments, especially for large IP packets and large number of hops. However, while the compressed IPsec reduces the packet size, the evaluation results also showed the increasing overhead in terms of energy consumption and average response time. Another drawback is that the proposed encoding scheme only supports the transport mode and tunnel mode is not addressed.

Similar to Raza *et al.* [125], Granjal *et al.* [50] proposed a new compressed 6LoWPAN security headers that allow the protection of IPv6 communications with IPsec on IEEE 802.15.4 networks. In contrast to [125], the compressed security header supports both the tunnel mode and transport mode. They experimentally evaluated the execution times and memory requirements of cryptographic algorithms AES-CCM that they proposed for their integration of IPsec and 6LoWPAN. The results showed that the new header is compatible with existing wireless sensor nodes and IPsec. However, they did not specify the required 6LoWPAN headers or the function of each header field. Moreover, their experiments did not implement the new compression scheme and communication performance is also not given.

Besides IPsec, the host identity protocol (HIP) has been brought into attention in the recent years to provide end-to-end security. HIP is situated above the network layer and can be used with ESP of IPsec. HIP decouples the two functions of location information and identification information that currently performed by IP addresses. In HIP, the host identity is associated with a pair of public/private keys, where the public key is used as the host identifier. Thus, it is suitable to uniquely identify smart devices in the IoT systems. However, HIP cannot directly be applied in IoT due to the extra header overhead and expensive cryptographic operations such as Diffie-Hellman key exchange. In order to facilitate the specific needs for IoT, Vidal Meca *et al.* [163] introduced a lightweight HIP architecture that leverages the Multimedia Internet KEYing (MIKey) protocols [110] for key management. Their HIP architecture enhanced the key management using polynomials and introduced a central authority to perform the key exchange.

Similar to IPsec, in order to reduce the overhead of HIP header, Sahraoui and Bilami [137] presented a compression mechanism for HIP in 6LoWPAN environments. Their compression removed unnecessary field such as header length filed, version field, and checksum field. Also, it computed a shorter hash value (from 128 bits to 96 bits) for the host identifier in HIP. After the compression, the HIP header length for 6LoWPAN is reduced from 40 bytes up to 13 bytes. Furthermore, the authors proposed a distribution scheme to reduce the handshake load and computational load of key exchange in HIP.

Hossain and Hasan [62] also tried to reduce the authentication overhead in HIP. The proposed authentication scheme P-HIP is based on Elliptic Curve Qu-Vanstone (ECQV) cryptography rather than the standard X.509 certificates. During the authentication, HIP devices do not need to verify the corresponding signatures or hash values, thereby minimize the computational cost and save the energy consumption. However, P-HIP still requires public key cryptography to perform key exchange and a trusted Certificate Authority (CA) is needed to distribute ECQV credentials.

In conclusion, to support end-to-end security for IoT on network layer, security mechanisms should consider heterogeneous network layer protocols. We can either design security solutions with the help of a security gateway, or introduce compressed security headers and related security mechanisms in existing solutions such as IPsec and HIP.

6 Transport Layer

Transport layer protocols offer alternative solutions to provide end-to-end communication. Note that network layer protocols provide logical end-to-end communication between end hosts while transport layer protocols provide logical end-to-end communication between processes running on different end hosts. Two main transport layer protocols are Transmission Control Protocol (TCP) which provides a reliable and connection-oriented communication, and User Datagram Protocol (UDP) which provides an unreliable and connectionless communication. However, neither TCP or UDP offers any security services to secure communications on the transport layer. In this section, we investigate two secure communication protocols, the Transport Layer Security (TLS) and the Datagram Transport Layer Security (DTLS), that run on top of TCP and UDP respectively.

6.1 Transmission Layer Security

TLS or its predecessor Secure Sockets Layer (SSL) is the most widely used protocol that runs on top of TCP to secure communication on transport layer. Internet Engineering Task Force (IETF) specifies the latest version of TLS 1.3 [132]. TLS provides authenticity, includes both data origin authentication and identity authentication, integrity, and confidentiality.

Furthermore, TLS specifies three key exchange algorithms in the TLS handshake phase which we will describe later.

The full TLS protocol contains two main components, a handshake phase and a record phase.

- In handshake phase, communication parties perform the key exchange protocol and authenticate each other to provide authenticity service. TLS 1.3 specifies three different key exchange modes: DH Ephemeral (DHE) or Elliptic Curve DHE (ECDHE) mode, Pre-Shared Key (PSK) mode, and PSK with (EC)DHE mode.

  1. DHE and ECDHE allow communication parties to dynamically negotiate the key and provide *forward secrecy.* Forward secrecy means that the compromise of a long-term secret (e.g., private

key) does not leak information about session keys that are previously derived from the long-term secret. DHE supports five different groups: secp256r1, secp384r1, secp521r1, x25519, and x448. ECDHE also supports five different groups: ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, and ffdhe8192.

2. PSK mode requires communication parties pre-share a common secret before the current handshake starts. This secret can be established out of band or in a previous connection and then used in the current handshake phase. Instead of running a full handshake, communication parties could use the secret to directly derive the key for the current communication session. However, PSK mode does not provide forward secrecy.
3. PSK with (EC)DHE mode combines PSK and (EC)DHE to provide forward secrecy. The key is derived from a concatenation of the pre-shared secret and the secret from (EC)DHE computation.

The handshake phase also allows communication parties to authenticate each other by verifying certificates. TLS 1.3 supports the certificate type of X.509v3 and each certificate is signed by a trusted CA. The digital signature schemes used in X.509 and TLS 1.3 handshake phase to provide authenticity include RSASSA-PKCS1-v1_5, ECDSA, RSASSA-PSS RSAE, EdDSA algorithms, and RSASSA-PSS PSS.

- In record phase, TLS fragments the data to be transmitted into manageable blocks and apply Authenticated Encryption with Associated Data (AEAD) scheme protect the data with confidentiality and integrity. Rather than encrypting the data and generating MIC separately with independent keys, AEAD algorithms provide both services with a single procedure. TLS 1.3 supports five AEAD cipher suits: AES_128_GCM_SHA256, AES_256_GCM_SHA384, CHACHA20_POLY1305_SHA256, AES_128_CCM_SHA256, and AES_128_CCM_8_SHA256.

6.2 Datagram Transport Layer Security

TLS provides complete secure communication in untrusted networks, but it may not be a wise choice for IoT environments. This is because TLS runs on top of reliable transport protocols such as TCP which is based on congestion control algorithm and consumes too much network resources during the TCP connection setup. Instead, for security on transport layer in resource-constrained environments, DTLS [134] has been proposed recently to provide the same level of security service as TLS but operates on top of the unreliable transport protocol UDP.

A full handshake in DTLS is described in Figure 3. Flight 1 to flight 3 provide the protection against DoS attack. In flight 2, the server sends back a HelloVerifyRequest message which contains a stateless cookie. Then the client respond with another ClientHello with the cookie added in flight 3. The server verifies the cookie and continue the handshake only if the cookie is valid. Flight 4 to flight 6 are used for communication parties to exchange certificate and key information. Indeed, DTLS uses the same cryptographic algorithms to provide security services as in TLS 1.2 [133].

6.3 Vulnerabilities and Security

As we mentioned above, TLS/DTLS protect end-to-end communication on transport layer to provide security services such as confidentiality (using symmetric encryption schemes and AEAD), integrity (using MIC and AEAD), authenticity (using signature and certificate), and key exchange (using PSK and public key cryptography). Thus similar to the network layer, attacks on transport layer are usually target on the availability property.

6.3.1 Vulnerabilities on Transport Layer

In this report, we investigate two common DoS attacks that against availability and energy efficiency, and we also discuss the corresponding countermeasures.

- De-synchronization. In a de-synchronization attack, an attacker can disrupt the existing connection between two end devices by repeatedly sending fabricated messages with fault sequence numbers or

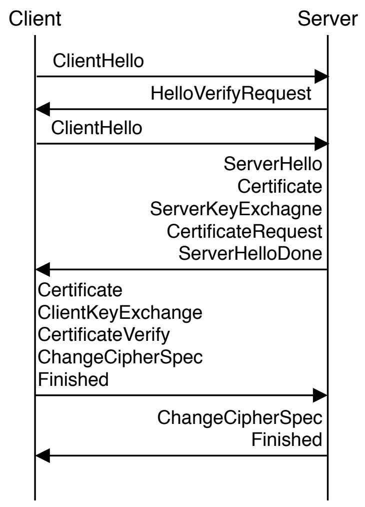

Figure 3: Message flights for full handshake in DTLS.

control flags [169]. As a result, communication devices lose their synchronization and are forced to retransmit data which can lead to the waste of network resource and energy on end devices.

A defense strategy to against de-synchronization attack is to authenticate all packets to guarantee the integrity, including the transport protocol header. As long as attackers cannot forge authentication of packets, end devices can detect forged sequence number and control flags [169].

- SYN-flooding. In a SYN-flooding attack, an attacker sends multiple connection establishment requests to end node but without ever completing the connection [124]. However, the end node have to allocate resources to maintain state for each connection request, thus exhausting energy and memory of the node and finally causing the node to be dead. Note that only connection-oriented transport protocol (i.e., TCP) is vulnerable to this type of attack.

  A naive solution that against SYN-flooding attack is to limit the number of connections for an end node. However, legitimate users may also be prevented from connecting to the node [169]. One alternative defense strategy is to use SYN cookies which is stored at the communication initiator side to avoid maintaining connection state. However, the computational and message overhead makes SYN cookies undesirable for resource-constraint networks [124]. Another solution is the usage of client puzzles. Aura *et al.* [9] suggested to let a server distribute puzzles to clients who which to connect. A client must solve the puzzles before initializing a connection. When there are heavy connection load on the server, the server could scale the puzzles and increase the computational overhead for the puzzles. Attackers would require more computational resources to solve the puzzles, thereby reduce the rate of sending malicious connection requests. On the other hand, legitimate clients would also require more resources to establish a connection.

6.3.2 Security on Transport Layer

Both network layer and transport layer provide the same security features for end-to-end communications. However, network layer protocols provide secure communication on the host level (i.e., host-to-host), while transport layer protocols provide security on a more precise level of process (i.e., process-to-process). Transport layer protocols usually cooperate with application layer protocols to provide security for different applications. Thus, transport layer protocols allow applications to directly and easily choose the required security services rather than specify the security services for a specific machine. Furthermore, the implementations of transport layer protocols have a better compatibility than network layer protocols. For example, Bonetto *et al.* [16] pointed out that most IPsec solutions in VPN require third-party hardware and/or software. An end device wants to access the VPN must install and properly configure a compatible IPsec client application. The installation and maintenance of the client application could be expensive and require human intervention. In contrast, TLS has a mature implementation in nearly all Web browsers. The main drawbacks of TLS are that it only works for web-based applications and consumes too much network resources during set up phase since it runs on top of TCP.

To reduce the overhead of TLS, DTLS is designed to provide secure services with UDP and it has recently received significant attention. DTLS now is encouraged by Internet Engineering Task Force (IETF) for the usage in resource-constrained environments. Kothmayr *et al.* [87] implemented the standard DTLS and evaluated the performance on IoT devices. The results showed that DTLS Handshake with RSA algorithm is feasible for key exchange in the Internet of Things Unfortunately, their solutions require hardware accelerator of RSA cryptographic operations. In their follow-up work [89], Kothmayr *et al.* presented the first fully implemented two-way authentication scheme for IP-based IoT in 6LoWPAN. The authentication scheme is based on standard DTLS protocol with RSA and X.509 certificates. The experiments on real IoT systems demonstrated that DTLS provides complete security services with affordable overhead in terms of energy, latency and memory.

However, DTLS still faces some challenges to become more friendly for resource-constrained devices.

- One major issue is the extra overhead of DTLS header. Similar to IPsec, a DTLS header could be too long to fit in LoWPAN MAC layer protocols such as IEEE 802.15.4. Furthermore, the 6LoWPAN standard does not specify the compression mechanisms for DTLS security header.

  To this end, Raza *et al.* [128] introduced the 6LoWPAN header compression for DTLS. The authors integrates DTLS with 6LoWPAN-GHC [18] which can be used to compress the UDP payload. The proposed approach encodes the header bits separately for DTLS record and handshake, ClientHello, and ServerHello messages. As the result, the overhead of security headers in DTLS can be reduced up to 75% such that the headers can fit within a single Maximum Transmission Unit for IEEE 802.15.4.

  In their following work [127], Raze *et al.* proposed the protocol Lithe which integrates DTLS and the application layer protocol the Constrained Application Protocol (CoAP). Lithe leverages the 6LoWPAN compressed compression mechanisms to compress the DTLS headers and DTLS handshake protocols (i.e., ClientHello, ServerHello, and Handshake messages) while maintain the same security properties as DTLS. The evaluation results showed that Lithe significantly reduces the energy consumption, processing time, and networking overhead. However, Lithe modified the original DTLS protocol and does not compatible with the DTLS standard. A similar work by Chavan and Nighot [24] also proposed an enhanced DTLS to compress DTLS protocol with 6LoWPAN header compression mechanisms.

- Another main challenge for DTLS in IoT environment is the expensive overhead in DTLS handshake. The full handshake in DTLS requires two communication entities to exchange 15 messages. Moreover, the validation of certificates and key establishment procedure during handshake result in a extremely high computational burden and processing time due to the inevitable public key operations.

  One possible solution to mitigate the cost of running DTLS on IoT devices is to offload costly operations to rich-resource entities. Hummen *et al.* [67] suggested to delegate the handshake procedure to the owner of devices (e.g., a security gateway). In particular, all certificate related operations between a IoT device and a server are performed on a device's owner and then the owner sends a message of the communication session state to the device after the handshake. Finally, the device can use the session state message to resume the communication session with the server without extra public key operations.

  Granjal *et al.* [52] proposed a similar architecture to offload Elliptic Curve Cryptography (ECC) based certificate operations to a 6LoWPAN Border Router (6LBR). The 6LBR uses standard DTLS protocol to establish communication session with a server while using pre-shared key security model to communicate with an IoT device. Thus, the 6LBR acts as transparent to both the IoT device and the server. It only intercepts and forwards packets that are transmitted between the device and the server. Kothmayr *et al.* [88] also employed a security gateway to assist mutual authentication. The proposed architecture uses specialized trusted-platform modules (TPM) with hardware acceleration to support RSA cryptography on IoT devices instead of ECC.

  The delegation based solutions significantly reduce the computation and communication overhead on IoT devices. However, these solutions require a trusted third party to perform costly operations and thus become a single point of failure in the protocol. When the trusted third party becomes malicious or gets compromised, all security properties would be no longer exist.

7 Application Layer

The last approach to provide end-to-end security for IoT is to enable security directly at the application layer. The application layer protocols specify how to exchange messages between applications. In this section, we introduce several widely used application layer protocols, discuss their security, and explore the challenges to provide security at the application layer for IoT.

7.1 Constrained Application Protocol

One of the most popular application layer protocol in IoT is the Constrained Application Protocol (CoAP) [19]. CoAP is proposed by the working group of Constrained RESTful Environments (CoRE) within the Internet Engineering Task Force (IETF). Similar to the Hypertext Transfer Protocol (HTTP), CoAP is a web transfer protocol based on REpresentational State Transfer (REST), but optimized for resource constrained devices. CoAP also uses GET, PUT, POST, and DELETE methods to provide message exchange and follow a simple request/response interaction in client/server model. However, unlike HTTP, requests and responses in CoAP are exchanged in asynchronous manner since CoAP operates on top of UDP.

Since UDP is unreliable and connectionless, CoAP provides a lightweight reliability mechanism to guarantee the message transfer. In particular, a CoAP message can be marked as *Confirmable* such that the message receiver must acknowledge the Confirmable message with a corresponding *Acknowledge* message. On the other hand, CoAP also supports unreliable transfer if a message is marked as *Non-Confirmable*, in which case the message receiver does not need to acknowledge the message.

CoAP itself does not provide any security service, it combines with DTLS to secure CoAP messages in terms of confidentiality, integrity, authentication and non-repudiation. CoAP defines four security modes in which differ on how authentication and key negotiation is achieved.

- NoSec mode. DTLS is disabled in this mode and CoAP messages are transmitted directly using UDP without security. However, it is recommended that security should be provided at the network layer (e.g., IPsec).

- PreSharedKey mode. DTLS is enabled in this mode and PSK authentication in DTLS is applied. Devices are pre-programmed with a list of pre-shared symmetric keys and each key is used for communication with other devices or a group of devices. This mode is appropriate for devices that are unable to run public key cryptography. RFC 7252 [149] suggests that PSK mode must at least support the cipher suite of TLS_PSK_WITH_AES_128_CCM_8. If more than two devices share the same key, CoAP allows communication for a group.

- RawPublicKey mode. DTLS is enabled in this mode and public key based authentication scheme in DTLS is applied. A device has a pre-distributed pair of asymmetric keys and an identity that is derived from its public key, while without a certificate. The asymmetric key pair can be validated using an out-of-band mechanism. For example, the manufacturer of the device generates a key pair and installed the key pair on the device before deployment. This mode is defined as mandatory in RFC specification [149] and must implement at least TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8 cipher suite. The public keys should support the ECDSA signature scheme and secp256r1 elliptic curve.

- Certificate mode. DTLS is enabled in this mode and public key based authentication scheme in DTLS is applied. Similar to RawPublicKey mode, a device has a pre-distributed pair of asymmetric keys but also with an X.509 certificate which is signed by a trusted root. This mode must implement at least TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8 cipher suite. The public keys should support the ECDSA signature scheme and secp256r1 elliptic curve. Note that the use of X.509 certificates is not suitable for IoT devices due to the large size of certificates. There are still many challenges for the tradeoff between the security and the efficiency in this mode for IoT network.

One aspect of combining CoAP security with DTLS is that it supports ECC in RawPublicKey mode and Certificate mode. ECC is usually considered as affordable for IoT devices. However, CoAP does not specify any key distribution or management schemes. It assumes that keys are obtained from the DTLS authentication handshake or a pre-distributed process.

7.2 Message Queue Telemetry Transport

Message Queue Telemetry Transport (MQTT) is another application layer protocol for IoT environments. MQTT was first designed by IBM in 1999 [68] and standardized by OASIS in 2013. While CoAP offers both machine-to-machine (M2M) and group communication, MQTT only provides a reliable M2M communication in which connections do not need human intervention. Moreover, MQTT runs on top of TCP/IP instead of UDP to provide reliable communication.

Rather than using request/response model, MQTT is an asynchronous publish/subscribe protocol that uses a broker to transfer messages. In MQTT, a publisher sends notice to the broker, which then forwards updated messages to subscribers automatically. The publish/subscribe model and MQTT are used in many IoT systems such as health, energy, and social media.

Even though MQTT runs over TCP, it is designed to have low overhead and suitable for resource-constrained devices to communicate over unreliable, low bandwidth networks. Especially, the publish/subscribe model usually requires lower network bandwidth and less message processing compared to request/response model since messages do need to be responded. A performance study of MQTT by Thangavel *et al.* [161] showed that MQTT has lower delays than CoAP for low packet losses. On the other hand, CoAP creates less extra traffic to ensure reliability.

To ensure security, MQTT requires a username an a password to provide authentication. However, both username and password are sent in plaintext. To protect these values, MQTT has to rely on the TLS encryption. Another security version of MQTT is SMQTT which addresses some security issues, but it is still under research [150].

7.3 Extensible Messaging and Presence Protocol

Another popular application layer protocol for IoT is the Extensible Messaging and Presence Protocol (XMPP) which standardized by IETF [139] in 2011. Different from CoAP and MQTT, XMPP supports both request/response and publish/subscribe models. It is designed for near real-time communications and thus, supports low latency message exchange such as instant messaging and video call.

XMPP uses XML (Extensible Markup Language) structure as the main data format to transfer messages,. Unfortunately, XML requires additional computational ability to parse XML message and also increases the network traffic overhead due to unnecessary tags. Al-Fuqaha *et al.* [3] showed that the compression scheme for XML called Efficient XML Interchange (EXI) can be used to optimize XML for resource-constrained devices. By applying EXI scheme, XMPP reduces the network overhead and minimize the required storage size.

For security properties, IETF recommends to authentication via Simple Authentication and Security Layer (SASL) profile. SASL provides a set of authentication methods and a device can choose the most fit one based on its specific needs. In order to authenticate a device, XMPP suggests to verify usernames, fully qualified domain names, identifiers and passphrases. In addition, XMPP provides secure communication to guarantee the confidentiality by combining with TLS encryption. Thereby, a device may employ both SASL and TLS to provide fully secure communication in XMPP. Indeed, XMPP is considered as one of the most secure communication protocols for IoT devices.

7.4 Vulnerabilities and Security

7.4.1 Vulnerabilities on Application Layer

IoT has a wide variety of applications and suffer from different security risks even security services are applied. Since most application layer protocols coordinate with transport layer protocols, they face the same security risks as the transport layer protocols such as De-synchronization and SYN-flooding. In addition, application layer protocols can also be exploited by DoS attacks. The path-based DoS attack [143] flood fabricated or replayed packets along multi-hop, end-to-end routing paths from a long distance, thereby overwhelms sensor nodes and affect all communications on the path from the attacker to the destination. To defend the path-based DoS (PDoS) attack, Deng *et al.* [35] proposed a lightweight solution against PDoS based on one-way hash chains (OHC). Each node stores a OHC which is configured for intermediary nodes to detect PDoS attack. When an intermediary node cannot verify the correctness of a OHC, the node would filter the suspicious packets.

Besides acting as victims for IoT devices in DoS attacks, attackers can also use them to create botnets and launch DDoS attacks. Mirai is a Linux malware botnet which can be used to obtain shell access of IoT devices. After Mirai controls enough IoT devices, the whole Mirai botnet is waiting orders to launch a DDoS attack. Antonakakis *et al.* [8] studied the Mirai botnet and showed the DDoS attack that against different major Domain Name Service providers.

Other security threats for application layer protocols are mainly from human errors. Since most application layer protocols do not support any built-in security service, they have to coordinate with other layers to provide security services. Developers may neglect to secure the applications or have incorrect configurations of security services during the implementations. Nebbione and Calzarossa [106, 121] exploited different vulnerabilities in IoT application layer protocols due to the lack of appropriate security services or to their incorrect configurations.

7.4.2 Security on Application Layer

Application layer protocols usually have to coordinate with other layer protocols in order to provide secure communications in IoT environment. For example, as we mentioned above, CoAP, MQTT, and XMPP have to leverage the transport layer protocol such as DTLS or TLS to protect confidentiality.

The main advantage for application layer protocols to provide security services on other layers is that it simplifies the development of applications [30]. All applications can run on the same standardized security protocols. Therefore, developers can focus on the application without dealing with the details of implementations of security protocols on other layers. In other words, applications may reuse the implementations (i.e., the same code) of these security protocols and significantly reduce the code size. On the other hand, security services on other layers are usually considered as optional for application layer protocols. Thus, developers must explicitly specify what security services must be applied. However, due to the diversity of application layer protocols, developers may neglect or misconfigure these services in the implementation of their applications. Nebbione and Calzarossa [106] has shown that vulnerabilities for application layer protocols are usually refer to improper message validation/parsing (e.g., , buffer overflow) and weak authentication/authorization mechanisms. These vulnerabilities are appearing with an increased frequency in recent years. Moreover, providing security services on transport layer or network layer may also lead to some setup issues in IoT networks [30, 59]. For example, when communication parties introduce an intermediate node such as a proxy to replay or filter application messages, the communication parties must fully trust the intermediate node. Moreover, the message retransmission mechanism in DTLS may require large buffers for a message receiver to hold data for retransmission.

Another approach to provide secure end-to-end communication is to apply security services at the application layer directly. Granjal *et al.* [51] proposed the first secure communication protocol to address security services at application layer for IoT. The proposed extended CoAP defines three security options that integrated with the standard CoAP protocol. The three options specify that if at the application layer, a CoAP message is protected, identity and authorization mechanisms are applied (e.g., X.509 certificate or PKC), and security-related CoAP messages are processed by cryptography algorithms (e.g., AES/CCM). This approach provides granular security on a per-message basis instead of per-packet basis.

The main challenge to provide security services at the application layer is the lacking of well defined and standardized secure mechanism for application layer protocols. This is usually due to the heterogeneity of IoT applications. Some existing work were considered as potential security standards for application layer protocols in IoT environment. Although Secure/Multipurpose Internet Mail Extensions (S/MIME) [60] was first designed to secure MIME data for mail users, it can also be applied to protect any application data in terms of confidentiality, integrity and non-repudiation of origin, and authentication. Another possible standard is the Secure Real-time Transport Protocol (SRTP) [111]. SRTP defines a framework to provide confidentiality, message authentication, and replay protection for RTP messages (e.g., instant messaging and video call). Similar to S/MIME, SRTP can also be applied to protect any application data.

A more recent solution is the blockchain technology which was originally proposed for cryptocurrencies such as Bitcoin [105] and Ethereum [33]. Blockchain provides a distributed ledger to store transaction records and achieve consensus in a decentralized fashion. Due to the decentralized nature of blockchain, it has recently received a lot of attentions in the field of IoT to secure communication. Researchers believe that blockchain can help smart devices to securely interact and transact with each other autonomously without human interventions [101]. Christidis and Devetsikiotis [29] reviewed how blockchain can benefit the IoT domain and be applied in different IoT applications. For example, blockchain can facilitate the energy trading, billing, and supply chain management among IoT devices. Khan and Salah [81] also believe that blockchain can help secure IoT communications. With blockchain, each IoT device would have a unique ID and a symmetric key pair once installed and joined the blockchain network, thereby eliminated key management and distribution process which is required in other secure protocols such as DTLS and IPsec. Moreover, blockchain can provide data integrity and authentication since data in the blockchain network are cryptographically proofed and signed by participants. Hammi *et al.* [57] proposed a real blockchain based decentralized system to ensure authentication of devices and protect the data integrity and availability. When an IoT device is compromised, blockchain could detect the compromise and the device may even self-healing with the blockchain [10].

Blockchain solution also has its drawbacks. The main disadvantage of blockchain in IoT environment is the significant energy, delay and computational overhead. For instance, the proof of work (mining) may overwhelm the limited computing resources for an IoT devices since mining may require to compute a large number of hash values. Another drawback is that traditional blockchains do not guarantee the privacy since transactions in the blockchain are stored in plaintext and exposed to the public. Whenever the property of privacy is needed, IoT devices should apply a privacy-preserving blockchain [86] instead to ensure transaction records are hidden from both the public and the participants. However, a privacy-preserving blockchain would introduce more computational overhead that resource-constraint devices may not be able to afford. More investigation and research are still required to make blockchain technology more friendly in IoT environments.

8 Key Exchange Protocols

A crucial issue to adopt security mechanisms on different networking layers is the key distribution and management. In IoT environments, symmetric key cryptography (SKC) such as AES and MAC are efficient and easy to implement in hardware platforms, thereby usually preferred to resource-intensive Public Key Cryptography (PKC) to provide security services. However, SKC schemes require each entity who participate in the communication should share a common cryptographic key with other entities prior the communication. Indeed, SKC based solutions suffer from scalability and key management issues. Therefore, designing an efficient key establishment scheme/protocol becomes a serious research problem in IoT.

Some standardized networking protocols propose a suggestion of key establishment schemes in their specifications. For example, IEEE 802.15.4 standard at MAC layer state that a pre-shared secret could be installed priori among devices and the key for communication is derived from the secret. The network layer protocol IPsec suggests to apply the IKE protocols. An lightweight version IKEv2 [79] is proposed to address the resource concerns. The transport layer protocols TLS supports DHE/ECDHE, PSK mode, and PSK with DHE/ECDHE. However, each of them suffer from different drawbacks which could significantly affect the resource usage and security in IoT environment. Therefore, we believe a full literature study of key establishment schemes is necessary to complete the discussion of secure communication protocols in IoT.

In this section, we explore the applicability and limitations of existing key establishment schemes which are potentially suitable in the context of IoT. The taxonomy is based on two main classes used to establish a secret key for communication, namely symmetric and public cryptographic solutions. Note that in this report we only consider the pair-wise key establishment, while the group-wise key establishment is left for future work. See Figure 4 for a breakdown of IoT key establishment protocols.

8.1 Symmetric Key Solution

In symmetric key solution, we distinguish two key establishment approaches as follows.

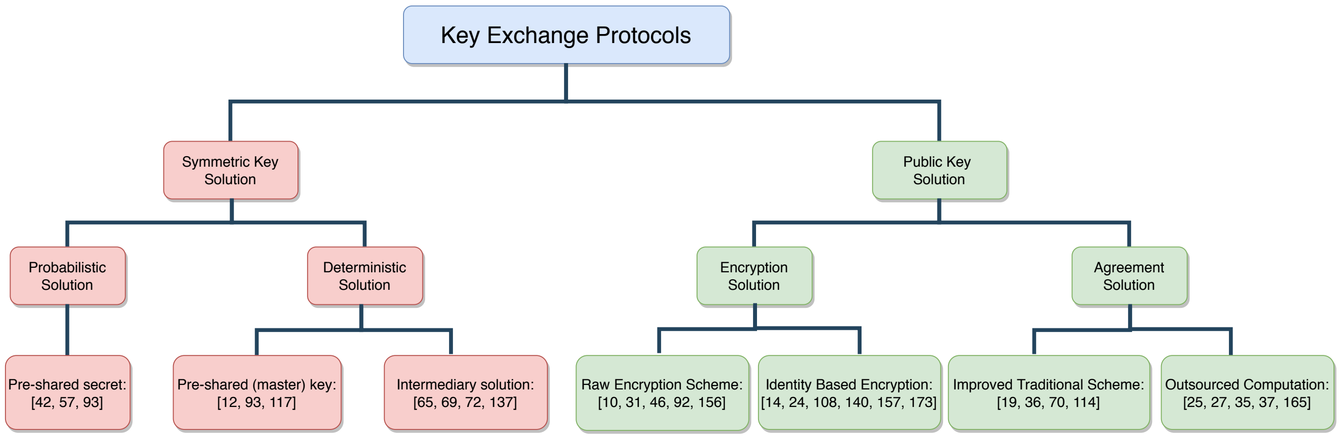
Key Exchange Protocols
Symmetric Key Solution
Public Key Solution
Probabilistic Solution
Deterministic Solution
Encryption Solution
Agreement Solution
Pre-shared secret: [42, 57, 93]
Pre-shared (master) key: [12, 93, 117]
Intermediary solution: [65, 69, 72, 137]
Raw Encryption Scheme: [10, 31, 46, 92, 156]
Identity Based Encryption: [14, 24, 108, 140, 157, 173]
Improved Traditional Scheme: [19, 36, 70, 114]
Outsourced Computation: [25, 27, 35, 37, 165]

Figure 4: Taxonomy of IoT key establishment protocols.

8.1.1 Deterministic Solution

Deterministic key establishment schemes guarantee that the two communication entities will agree on the same session key (for the current communication session) after running the protocol. The most common approach in deterministic key establishment schemes is to pre-install the secret key in the two nodes. This approach does not require any cryptographic computations or exchange any messages to establish the session key. Therefore, it is efficient in terms of energy and network overhead. However, when a node is compromised, the attacker can obtain all sensitive data in the node and access all messages from the nodes who share the same key with the compromised node, including all messages from the communication history.

Instead of pre-install the secret key directly, some works suggest to pre-share a secret and then derive the session key from the secret. One example of the pre-shared secret is bivariate polynomials [94]. Two parties Alice and Bob share a bivariate polynomial $f(x, y)$ such that $f(a, b) = f(b, a)$. To build the key, Alice computes $f(I_a, I_b)$ and Bob computes $f(I_b, I_a)$ where I_a and I_b are identities for Alice and Bob respectively. Since $f(I_a, I_b) = f(I_b, I_a)$, Alice and Bob will use it as the session key for communication. Also, rather than using just one polynomial, the authors suggest to use a set of bivariate polynomials and leverage a real-time discovery scheme to protect them from being compromised. Even an attacker compromise a node, the attacker cannot distinguish which polynomials are overlapped with other nodes. Another instance of the pre-shared secret is a long master secret key. Perrig *et al.* [118] suggested to calculate the session key from the pre-shared master key. The two communication parties will generate two random nonces and then hash the two nonces with the pre-shared master key to obtain the session key. After the communication, the pre-shared master key will be removed from parties' memory. In the next communication session, the new session key will then be generated from the previous session key. When a node is compromised by an attacker, the attacker will not be able obtain the message history. However, if the pre-shared master key is leaked to the attacker, the attacker can recover all messages and the confidentiality will be destroyed. Rabiah *et al.* resolved this vulnerability in their recent work [13] to provide *Perfect Forward Secrecy* (PFS) such that even if the master key is leaked somehow at some point, all messages that are transferred in past sessions remain secure. Their protocol also generates two random nonces and encrypts them with the master key. Then both communication parties use the two nonces to derive the session key. However, the session key update phase will use a random set of sequence numbers of session frames to protect all session keys in the past.

Another approach in deterministic solutions is to introduce a third party (or parties) to help resource-constrained nodes to perform key exchange. Hummen *et al.* [66] proposed a delegation-based key exchange protocol for IoTs. Each node in the system maintains a key associated with an external trusted server. When a node wants to initialize a communication session with another node, it leverages the trusted server to derive a new secret key for the communication. A similar idea is to offload computational-expensive operations to a proxy located within the WSN. Hussen *et al.* [70] suggested that a sensor node could delegate the DH key exchange protocol to a trusted proxy. The proxy performs the protocol on behalf of the node and sends the session key to the node at the end. Although this approach drastically reduces the resource usage of IoT devices, the trusted server or proxy is a major point of failure.

Rather than placing trust into a single third party, a better way is to use multiple intermediary parties [138, 70, 73]. The main idea in this approach is to leverage the *secret sharing scheme* [147]. A node can initiate a key exchange with another node by splitting a secret into multiple *secret shares*. Each share leaks no information about the original secret. Then the node sends each share to a different intermediary party who forwards the share to the other node. The other node can reconstruct the secret by assembling all the shares and finally derive the session key from the secret. Even if some intermediary parties are compromised, the attacker still cannot recover the original secret.

The intermediary party-based solutions can significantly reduce the computational overhead on resource-constrained devices. In these solutions, devices must trust the third party/parties to honestly follow the protocols. If the third party/parties are compromised, there is no guarantee of the key agreement and useful information could also be leaked to adversaries. However, to our best knowledge, there is no secure key exchange protocols that against malicious intermediary parties.

8.1.2 Probabilistic Solution

In contrast to deterministic solution, probabilistic key exchange protocol does not guarantee the session key establishment at the end. Two node may fail to agree on the same key with a certain probability. Eschenauer and Gligor [43] proposed the first probabilistic key exchange protocol. In their protocol, each node randomly select a set of keys from a universal key pool. When a node wants to communicate with another node, it broadcasts its keys' ID to the other node. If the key sets in the two nodes overlap, the session key will then be derived from the intersection of the two key sets. If an attacker compromise a node, the attacker can only learn the key set of this node, while the session key and key sets in other nodes remains secure. A similar work is proposed in [94] but using random bivariate polynomials instead of keys. Chan *et al.* [58] improved Eschenauer's work and discussed that the size of a key set and the number of overlapped keys would affect the performance of the protocol. Their improvement could reduce the memory space for the key set and enhance the resilience against compromised nodes. Du *et al.* [167] also proposed a scheme to reduce the memory usage by only establishing necessary session keys based on the deployment knowledge of nodes.

As we discussed above, in all symmetric key exchange protocols, entities who participate in the key establishment protocols are required to pre-share some secrets such as keys and bivariate polynomials before the key exchange starts. Indeed, for cryptographic key exchange solutions, Impagliazzo and Rudich [72] showed that SKC alone is not sufficient to establish a key exchange protocol via public channels. On the other hand, non-cryptographic key exchange solutions do not involve any cryptographic operations, but they usually rely on strong assumptions such as proximity. We will discuss some non-cryptographic key exchange solutions at the end of this section.

8.2 Public Key Solution

In contrast to symmetric key solutions, public key solutions do not require a pre-shared secret among entities. Thus, PKC based solutions usually can achieve better scalability and are preferred in large IoT systems (e.g., smart city). The main drawback to apply PKC in IoT environments is that PKC usually involves expensive cryptographic operations which may not suitable for resource-constrained devices. In this section, we explore different PKC-based key establishment schemes and show that PKC are still necessary and adaptable for IoT systems.

We divide our discussion into two classes, namely PKC encryption solutions and PKC agreement solutions.

8.2.1 PKC Encryption Solution

Traditional PKC encryption schemes are widely deployed in IoT systems. A node Alice first generates a session key for the communication session. Then she encrypts the session key with Bob's public key and sends the encrypted session key to Bob. Bob decrypts it with his private key and finally obtains the session key. Some famous PKC encryption schemes are RSA, DSA, ElGammal, etc. These solutions are widely adopted in the conventional Internet due to their flexibility, scalability and key management efficiency. However, PKC encryption schemes involves expensive cryptographic operations in message encryption and decryption, thereby they may not suitable for IoT networks.

One exception is the ECC encryption scheme. Indeed, ECC is considered as very efficient in some IoT systems [157]. ECC provides the same security level as RSA but with much smaller size of the key. For example, to ensure a security level of 128-bit, RSA requires a key of 3072 bits while ECC only needs 256 bits. Researchers also try to improve the efficiency of ECC and make it more friendly to IoT. Works in [11] and [32] introduced different elliptic curve variants and showed that an elaborately picked curve can reduce the computational overhead. Liu and Ning [93] implemented the TinyECC which is a configurable library for ECC operations in wireless sensor networks. The experimental evaluations showed that an optimized configuration of TinyECC can significantly reduce the execution time and resource consumptions.

Another exception is the NTRU algorithm [47]. NTRU is a lattice-based encryption scheme alternative to RSA and ECC. It relies on the Shortest Vector Problem (SVP) rather than the integer factorization or discrete logarithm problem (DLP). NtruEncrypt is highly efficient and suitable for some IoT systems such as smart cards or RFID tags. Gaubatz *et al.* [47] compared Rabin's scheme (similar to RSA), NTRU and ECC, and the results showed that NTRU consists of the smallest computations and power consumptions. However, NTRU requires more memory to store keys and transfer large size of messages.

Note that PKC encryption solution usually requires the verification of the authenticity of public keys. A user must verify that a public key is indeed associated with a correct entity. Otherwise the PKC encryption solution is vulnerable to the man-in-the-middle attack. The verification of public keys are usually achieved by certificates. A Certificate Authority (CA) may participate to distribute certificates to different users in the system. However, the verification of certificate is expensive for IoT devices, especially when traditional PKC schemes (e.g., RSA) are involved to generate certificates. Some communication protocols such as TLS and DTLS are using certificates to build the trust relationship between two entities.

An alternative solution to ensure the authenticity of public keys is the Identity-Based Encryption (IBE) [148]. IBE uses an unique string (e.g., email, phone number, ssn) which could represent a user's identity as the user's public key. A trusted party called Private Key Generator (PKG) is responsible to generate the private key for each entity from the entity's public key. Since the public key is directly associate with the identity, certificate is no longer needed to ensure the identity. However, the PKG now becomes the major point of failure. It knows all the private keys and can easily access all transferred messages in the system.

The implementation of IBE schemes relies on PKC encryption schemes. Thus, it will incur heavy resource consumptions. In the IoT context, IBE is usually implemented by ECC and many works have been investigated to design new lightweight IBE schemes. Chen [25] showed that IBE scheme with ECC implementation can be used to secure communications for RFID tags. Yang *et al.* [174] proposed a lightweight Elliptic Curve Diffie-Hellman (ECDH) key exchange protocol IBAKA. IBAKA tailors the existing Boneh and Franklin IBE (BF-IBE) [15] into ECDH protocol by encrypting the parameters exchanged in the ECDH with the BF-IBE approach. Their protocol requires 2 bilinear pairings and 3 scalar point multiplications for each key establishment [109]. Szczechowiak and Collier [158] presented the TinyIBE to reduce the number of bilinear pairings. TinyIBE is based on Sakai and Kasahara IBE (SK-IBE) [141] which eliminates the pairing calculation in encryption and only needs 2 messages exchange to have a key agreement.

8.2.2 PKC Agreement Solution

The PKC agreement solution allows two communication parties to derive the secret session key from some public information. Two communication parties generate their own public information respectively, and then exchange the public information. Finally, only the two parties can derive the session key from the public information while other parties will learn no information about the session key from the public information. A classical example of PKC agreement solution is the DH protocol [37]. DH protocol relies on the hardness of DLP and two communication parties only need to exchange two messages to establish the session key. However, DH protocol requires expensive group operation which is considered unsuitable for IoT system.

Elliptic curve DH (ECDH) [71] is a variant of. It is based on ECC and relies on the hardness of elliptic curve DLP (ECDLP). ECDH can provide the same level of security as DH but with much smaller security parameters. Practical measurements [84, 146, 154] have shown that ECDH performs well on different IoT platforms and now is widely accepted in IoT environments.

Some recent works improves the standard ECDH and reduces the computational overhead on resource-constrained devices. One possible solution is to speed up the scalar multiplication in elliptic curve. Boyko *et al.* [20] presented a scheme to reduce the computational cost of a scalar multiplication operation to a few EC additions, but with extra storage that is linear to the store intermediate results. Ozmen and Yavuz [115] improved Boyko's work by performing some pre-computations in an offline phase to eliminate online scalar multiplications. In addition, their algorithm only needs a small constant size of storage overhead rather than linear overhead. Another solution is to outsource expensive cryptographic to a third party such as a cloud server. Chen *et al.* [26] proposed a scheme to securely outsource modular exponentiations to two untrusted cloud servers. In addition, Chen's solution can verify the results from the two untrusted servers with probability of 2/3. However, the outsourced two servers should not collude with each other. Following works in [166, 38] allow the outsourcing to a single untrusted servers. Nevertheless, Chevalier and Laguillaumie [28] showed that these solutions are secure only if the server is semi-honest and did not achieve the claimed security guarantees when the server is malicious. They presented polynomial-time attacks that can recover partial or even the whole secret input from clients. Crescenzo *et al.* [36] proposed a malicious secure outsourcing mechanism. They introduce a rigorous definition of security for outsourced computation protocols and formally proved that the proposed mechanism is secure under the definition even if the server is malicious. In addition, their algorithm can detect the cheating server with overwhelming probability (e.g., $1-2^{-128}$).

9 Conclusion and Future Work

As the development of IoT, an essential need for IoT systems is the secure communication between IoT devices. A standardized set of existing networking protocols provide fundamental security services in communications. Unfortunately, these protocols may not suitable to directly adopted in IoT networks due to the severe resource constraints of IoT devices. In addition, these protocols may not be able to achieve the expected security and rise unforeseen challenges due to the heterogeneity of IoT applications. In this report, we reviewed the widely accepted secure communication protocols in IoT and explored each layer of the IoT network protocol stack. Moreover, we investigated common security vulnerabilities at each layer that would threat the IoT communications even secure protocols are applied. We also showed existing mitigation techniques to against attacks that are derived from such vulnerabilities. Then we gave a full analysis of security of each networking layer, identified some missing gaps and presented current research trend in IoT communication protocol security. Finally, we surveyed the key exchange protocols in IoT which is a core component to efficiently adopt security mechanisms on different networking layers.

While we delivered a taxonomy of protocols to secure IoT communications at each networking layer, some newly emerging cryptographic primitives may enhance the security and efficiency to improve the existing protocols.

9.1 Lightweight Cryptography

Lightweight cryptography is aiming at designing new ciphers to meet requirements for some special IoT devices. In these devices, resources are extremely limited such that they cannot even perform symmetric ciphers. Lightweight cryptography should have smaller footprint, low energy consumption, and low computational power [30], but without weakening the security (i.e., leak useful information). Usually lightweight cryptography refers to the trade-offs between security level, cost, and performance. For example, the Scalable Encryption Algorithm (SEA) [156] is designed for small embedded applications. The main advantage of SEA is its key size could be as small as 6 times the processor size and the "on-the-fly" key derivation. Therefore, SEA scalable and adaptable to different hardware platforms. TWINE [83] is a lightweight block cipher with block length of 64 bits and key sizes of 80 and 128 bits. In the hardware implementation, TWINE has the circuit size of 2K gates while AES has the circuit size of 15K gates. Evaluations showed that the efficiency of TWINE is more than twice that of AES and now TWINE is considered as "to-class" performance in both hardware and software implementations. Some other lightweight cryptography includes the Tiny Encryption Algorithm [168], PRESENT Cipher [14], and HIGHT cipher [61]. Unfortunately, lightweight cryptography does not rise too much interest in the research community and thus is not fully discussed in this report.

9.2 Blockchain

As we discussed in section 7, blockchain provides an alternative solution to secure IoT communications. While blockchain is first designed to provide a decentralized digital currency system, it can also facilitate many new applications for IoT. In fact, many Blockchain-based solutions have already been proposed for IoT to share data, exchange sensitive information, and distribute keys [81, 44, 74, 99, 151]. For example, Fakhri and Mutijarsa [44] studied the usage of Ethereum along with a smart contract to Secure IoT Communication. Their results showed that compared to MQTT, IoT communication with blockchain technology has a higher level of security. Manzoor *et al.* [99] also use Ethereum blockchain to secure IoT data sharing. The data will be re-encrypted by a proxy and then stored in the blockchain. Thus, the data is only visible to the owner and the person who presents in the smart contract. Although block-based protocol could improve the security of IoT communications, it also has many limitations in IoT environments. The main drawback is its efficiency. The mining work in blockchain may require high computational capacity and consume a lot of energy. Thus a more efficient ad-hoc blockchain should be proposed for IoT. Another drawback is the privacy issue. Since traditional blockchains store data in plaintext and all records are available to the public, attacker can easily access all data and conclude useful information about users. Thus we need a special variant privacy-preserving blockchain to address the privacy issue in some scenarios. In fact, blockchain-based solution will be our research focus in the future.

9.3 Zero-Knowledge Proof

Zero-knowledge proof (ZKP) [49] is a fundamental cryptographic primitive that allows a prover to convince a verifier that some statement is true. During the proof, the verifier should not learn any useful information except the statement is true. For example, in the context of IoT, students in a class would like to anonymously provide feedbacks about a course using resource-constrained devices (e.g., clickers) in a lecture. However, only authorized students who register the course are allowed to participate in the evaluation. Thus, a student must prove its authorization but without leaking any identity information [23]. Indeed, ZKP technique usually combines with the blockchain to protect the privacy of users' identity information [170]. Further investigation into Blockchain-based protocols and ZKP protocols could improve the security of IoT communications and address many privacy issues in IoT.

References

[1] Ieee standard for low-rate wireless networks. *IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011)*, pages 1–709, 2016.

[2] T. Akitaya, S. Asano, and T. Saba. Time-domain artificial noise generation technique using time-domain and frequency-domain processing for physical layer security in mimo-ofdm systems. In *2014 IEEE International Conference on Communications Workshops (ICC)*, pages 807–812, 2014.

[3] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys Tutorials*, 17(4):2347–2376, 2015.

[4] R. Alexander, A. Brandt, J. Vasseur, J. Hui, K. Pister, P. Thubert, P. Levis, R. Struik, R. Kelsey, and T. Winter. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks, Mar. 2012.

[5] A. A. A. Alkhatib and G. S. Baicher. Wireless sensor network architecture. In *2012 International Conference on Computer Networks and Communication Systems (CNCS 2012)*, 2012.

[6] D. Altolini, V. Lakkundi, N. Bui, C. Tapparello, and M. Rossi. Low power link layer security for iot: Implementation and performance analysis. In *2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 919–925, 2013.

[7] J. P. Anderson. Information security in a multi-user computer environment. volume 12 of *Advances in Computers*, pages 1 – 36. Elsevier, 1972.

[8] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou. Understanding the mirai botnet. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 1093–1110, August 2017.

[9] T. Aura, P. Nikander, and J. Leiwo. Dos-resistant authentication with client puzzles. In B. Christianson, J. A. Malcolm, B. Crispo, and M. Roe, editors, *Security Protocols*, pages 170–177, Berlin, Heidelberg, 2001.

[10] M. Banerjee, J. Lee, and K.-K. R. Choo. A blockchain future for internet of things security: a position paper. *Digital Communications and Networks*, 4(3):149 – 160, 2018.

[11] D. J. Bernstein. Curve25519: New diffie-hellman speed records. In M. Yung, Y. Dodis, A. Kiayias, and T. Malkin, editors, *Public Key Cryptography - PKC 2006*, pages 207–228, 2006.

[12] T. Bhattasali and R. Chaki. A survey of recent intrusion detection systems for wireless sensor network. volume 196, pages 268–280, 07 2011.

[13] A. Bin Rabiah, K. Ramakrishnan, E. Liri, and K. Kar. A lightweight authentication and key exchange protocol for iot. 01 2018.

[14] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoe. Present: An ultra-lightweight block cipher. In P. Paillier and I. Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems - CHES 2007*, pages 450–466, 2007.

[15] D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *Advances in Cryptology — CRYPTO 2001*, pages 213–229, 2001.

[16] R. Bonetto, N. Bui, V. Lakkundi, A. Olivereau, A. Serbanati, and M. Rossi. Secure communication for smart iot objects: Protocol stacks, use cases and practical examples. In *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–7, 2012.

[17] T. Borgohain, U. Kumar, and S. Sanyal. Survey of security and privacy issues of internet of things. *CoRR*, abs/1501.02211, 2015.

[18] C. Bormann. 6LoWPAN-GHC: Generic Header Compression for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs), Nov. 2014.

[19] C. Bormann, A. P. Castellani, and Z. Shelby. Coap: An application protocol for billions of tiny internet nodes. *IEEE Internet Computing*, 16(2):62–67, 2012.

[20] V. Boyko, M. Peinado, and R. Venkatesan. Speeding up discrete log and factoring based schemes via precomputations. In K. Nyberg, editor, *Advances in Cryptology — EUROCRYPT'98*, pages 221–235, 1998.

[21] I. Butun, P. Osterberg, and H. Song. Security of the internet of things: Vulnerabilities, attacks, and countermeasures, 2020.

[22] I. J. S. . Cards and security devices for personal identification. Iso/iec 14443-1:2018 cards and security devices for personal identification — contactless proximity objects — part 1: Physical characteristics. ISO/IEC Standard, ISO/IEC, 2018.

[23] I. Chatzigiannakis, A. Pyrgelis, P. G. Spirakis, and Y. C. Stamatiou. Elliptic curve based zero knowledge proofs and their applicability on resource constrained devices. In *2011 IEEE Eighth International Conference on Mobile Ad-Hoc and Sensor Systems*, pages 715–720, 2011.

[24] A. Chavan and M. Nighot. Secure coap using enhanced dtls forinternet of things. *International Journal of Innovative Research in Computer and Communication Engineering*, 2:7602–7608, 2014.

[25] W. Chen. An ibe-based security scheme on internet of things. In *2012 IEEE 2nd International Conference on Cloud Computing and Intelligence Systems*, volume 03, pages 1046–1049, 2012.

[26] X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou. New algorithms for secure outsourcing of modular exponentiations. *IEEE Transactions on Parallel and Distributed Systems*, 25(9):2386–2396, 2014.

[27] Y. Chen, W. Trappe, and R. P. Martin. Detecting and localizing wireless spoofing attacks. In *2007 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, pages 193–202, 2007.

[28] C. Chevalier, F. Laguillaumie, and D. Vergnaud. Privately outsourcing exponentiation to a single server: Cryptanalysis and optimal constructions. In *Computer Security – ESORICS 2016*, pages 261–278, 2016.

[29] K. Christidis and M. Devetsikiotis. Blockchains and smart contracts for the internet of things. *IEEE Access*, 4:2292–2303, 2016.

[30] S. Cirani, G. Ferrari, and L. Veltri. Enforcing security mechanisms in the ip-based internet of things: An algorithmic overview. *Algorithms*, 6:197–226, 04 2013.

[31] H.-P. D. Company. Hp study reveals 70 percent of internet of things devices vulnerable to attack. 2014.

[32] C. Costello and P. Longa. Fourq: four-dimensional decompositions on a q-curve over the mersenne prime. 06 2015.

[33] C. Dannen. *Introducing Ethereum and Solidity: Foundations of Cryptocurrency and Blockchain Programming for Beginners*. USA, 1st edition, 2017.

[34] M. Demirbas and Youngwhan Song. An rssi-based scheme for sybil attack detection in wireless sensor networks. In *2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks(WoWMoM'06)*, pages 5 pp.–570, 2006.

[35] J. Deng, R. Han, and S. Mishra. Defending against path-based dos attacks in wireless sensor networks. In *Proceedings of the 3rd ACM Workshop on Security of Ad Hoc and Sensor Networks*, page 89–96, New York, NY, USA, 2005.

[36] G. Di Crescenzo, M. Khodjaeva, D. Kahrobaei, and V. Shpilrain. Practical and secure outsourcing of discrete log group exponentiation to a single malicious server. In *Proceedings of the 2017 on Cloud Computing Security Workshop*, CCSW '17, page 17–28, 2017.

[37] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Trans. Inf. Theor.*, 22(6):644–654, Sept. 2006.

[38] Y. Ding, Z. Xu, J. Ye, and K.-K. R. Choo. Secure outsourcing of modular exponentiations under single untrusted programme model. *Journal of Computer and System Sciences*, 90:1 – 13, 2017.

[39] D. Dragomir, L. Gheorghe, S. Costea, and A. Radovici. A survey on secure communication protocols for iot systems. In *2016 International Workshop on Secure Internet of Things (SIoT)*, pages 47–62, 2016.

[40] A. Dvir, T. Holczer, and L. Buttyan. Vera - version number and rank authentication in rpl. In *2011 IEEE Eighth International Conference on Mobile Ad-Hoc and Sensor Systems*, pages 709–714, 2011.

[41] M. J. Dworkin, E. B. Barker, J. R. Nechvatal, J. Foti, L. E. Bassham, E. Roback, and J. F. D. Jr. Advanced encryption standard (aes). https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf, 2001.

[42] M. Eldefrawy, I. Butun, N. Pereira, and M. Gidlund. Formal security analysis of lorawan. *Computer Networks*, 148:328 – 339, 2019.

[43] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, CCS '02, page 41–47, 2002.

[44] D. Fakhri and K. Mutijarsa. Secure iot communication using blockchain technology. In *2018 International Symposium on Electronics and Smart Devices (ISESD)*, pages 1–6, 2018.

[45] M. U. Farooq, M. Waseem, A. Khairi, and S. Mazhar. A critical analysis on the security concerns of internet of things (iot). *International Journal of Computer Applications*, 111:1–6, 2015.

[46] Gartner. Gartner says 8.4 billion connected "things" will be in use in 2017, up 31 percent from 2016. https://www.gartner.com/en/newsroom/press-releases/2015-11-10-gartner-says-6-billion-connected-things-will-be-in-use-in-2016-up-30-percent-from-2015, 2015.

[47] G. Gaubatz, J. . Kaps, E. Ozturk, and B. Sunar. State of the art in ultra-low power public key cryptography for wireless sensor networks. In *Third IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 146–150, 2005.

[48] G. Glissa, A. Rachedi, and A. Meddeb. A secure routing protocol based on rpl for internet of things. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7, 2016.

[49] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, page 291–304, 1985.

[50] J. Granjal, E. Monteiro, and J. S. Sa Silva. Enabling network-layer security on ipv6 wireless sensor networks. In *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, pages 1–6, 2010.

[51] J. Granjal, E. Monteiro, and J. S. Silva. Application-layer security for the wot: Extending coap to support end-to-end message security for internet-integrated sensing applications. In V. Tsaoussidis, A. J. Kassler, Y. Koucheryavy, and A. Mellouk, editors, *Wired/Wireless Internet Communication*, pages 140–153, Berlin, Heidelberg, 2013.

[52] J. Granjal, E. Monteiro, and J. S. Silva. End-to-end transport-layer security for internet-integrated sensing applications with mutual and delegated ecc public-key authentication. In *2013 IFIP Networking Conference*, pages 1–9, 2013.

[53] J. Granjal, R. Silva, E. Monteiro, J. Sa Silva, and F. Boavida. Why is ipsec a viable option for wireless sensor networks. In *2008 5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems*, pages 802–807, 2008.

[54] D. Halperin, T. S. Heydt-Benjamin, K. Fu, T. Kohno, and W. H. Maisel. Security and privacy for implantable medical devices. *IEEE Pervasive Computing*, 7(1):30–39, 2008.

[55] J. M. Hamamreh and H. Arslan. Secure orthogonal transform division multiplexing (otdm) waveform for 5g and beyond. *IEEE Communications Letters*, 21(5):1191–1194, 2017.

[56] J. M. Hamamreh, H. M. Furqan, and H. Arslan. Classifications and applications of physical layer security techniques for confidentiality: A comprehensive survey. *IEEE Communications Surveys Tutorials*, 21(2):1773–1828, 2019.

[57] M. T. Hammi, B. Hammi, P. Bellot, and A. Serhrouchni. Bubbles of trust: A decentralized blockchain-based authentication system for iot. *Computers & Security*, 78:126 – 142, 2018.

[58] Haowen Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *2003 Symposium on Security and Privacy, 2003.*, pages 197–213, 2003.

[59] K. Hartke. Practical Issues with Datagram Transport Layer Security in Constrained Environments. Internet-Draft draft-hartke-dice-practical-issues-01, Internet Engineering Task Force, Apr. 2014. Work in Progress.

[60] P. E. Hoffman. Enhanced Security Services for S/MIME. RFC 2634, June 1999.

[61] D. Hong, J. Sung, S. Hong, J. Lim, S. Lee, B.-S. Koo, C. Lee, D. Chang, J. Lee, K. Jeong, H. Kim, J. Kim, and S. Chee. Hight: A new block cipher suitable for low-resource device. In *Cryptographic Hardware and Embedded Systems - CHES 2006*, pages 46–59, 2006.

[62] M. Hossain and R. Hasan. P-hip: A lightweight and privacy-aware host identity protocol for internet of things. *IEEE Internet of Things Journal*, pages 1–1, 2020.

[63] F. Hu and N. K. Sharma. Security considerations in ad hoc sensor networks. *Ad Hoc Networks*, 3(1):69 – 89, 2005.

[64] H. Hui, A. L. Swindlehurst, G. Li, and J. Liang. Secure relay and jammer selection for physical layer security. *IEEE Signal Processing Letters*, 22(8):1147–1151, 2015.

[65] R. Hummen, M. Komu, and R. Moskowitz. Hip diet exchange (dex). 2020.

[66] R. Hummen, H. Shafagh, S. Raza, T. Voig, and K. Wehrle. Delegation-based authentication and authorization for the ip-based internet of things. In *2014 Eleventh Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pages 284–292, 2014.

[67] R. Hummen, J. H. Ziegeldorf, H. Shafagh, S. Raza, and K. Wehrle. Towards viable certificate-based authentication for the internet of things. In *Proceedings of the 2nd ACM Workshop on Hot Topics on Wireless Network Security and Privacy*, HotWiSec '13, page 37–42, New York, NY, USA, 2013.

[68] U. Hunkeler, H. L. Truong, and A. Stanford-Clark. Mqtt-s — a publish/subscribe protocol for wireless sensor networks. In *2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE '08)*, pages 791–798, 2008.

[69] M. Hussain, Q. Du, L. Sun, and P. Ren. Security protection over wireless fading channels by exploiting frequency selectivity. In *2016 8th International Conference on Wireless Communications Signal Processing (WCSP)*, pages 1–5, 2016.

[70] H. R. Hussen, G. A. Tizazu, Miao Ting, Taekkyeun Lee, Youngjun Choi, and Ki-Hyung Kim. Sakes: Secure authentication and key establishment scheme for m2m communication in the ip-based wireless sensor network (6l0wpan). In *2013 Fifth International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 246–251, 2013.

[71] K. Igoe, D. McGrew, and M. Salter. Fundamental Elliptic Curve Cryptography Algorithms. RFC 6090, Feb. 2011.

[72] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, STOC '89, page 44–61, 1989.

[73] M. A. Iqbal and M. Bayoumi. Secure end-to-end key establishment protocol for resource-constrained healthcare sensors in the context of iot. In *2016 International Conference on High Performance Computing Simulation (HPCS)*, pages 523–530, 2016.

[74] U. Javaid, M. N. Aman, and B. Sikdar. Blockpro: Blockchain based data provenance and integrity for secure iot environments. In *Proceedings of the 1st Workshop on Blockchain-Enabled Networked Sensor Systems*, BlockSys'18, page 13–18, 2018.

[75] D. B. Johnson, D. A. Maltz, and J. Broch. *DSR: The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks*, page 139–172. USA, 2001.

[76] A. Josang, R. Ismail, and C. Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618 – 644, 2007. Emerging Issues in Collaborative Commerce.

[77] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: attacks and countermeasures. In *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications, 2003.*, pages 113–127, 2003.

[78] P. Kasinathan, G. Costamagna, H. Khaleel, C. Pastrone, and M. A. Spirito. Demo: An ids framework for internet of things empowered by 6lowpan. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, page 1337–1340, 2013.

[79] C. Kaufman, P. E. Hoffman, Y. Nir, P. Eronen, and T. Kivinen. Internet Key Exchange Protocol Version 2 (IKEv2). RFC 7296, Oct. 2014.

[80] S. Kent. IP Encapsulating Security Payload (ESP). RFC 4303, 2005.

[81] M. A. Khan and K. Salah. Iot security: Review, blockchain solutions, and open challenges. *Future Generation Computer Systems*, 82:395 – 411, 2018.

[82] D. Klinc, J. Ha, S. W. McLaughlin, J. Barros, and B. Kwak. Ldpc codes for the gaussian wiretap channel. *IEEE Transactions on Information Forensics and Security*, 6(3):532–540, 2011.

[83] E. Kobayashi, T. Suzaki, K. Minematsu, and S. Morioka. Twine: A lightweight block cipher for multiple platforms. volume 7707, 03 2012.

[84] R. K. Kodali and A. Naikoti. Ecdh based security model for iot using esp8266. In *2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, pages 629–633, 2016.

[85] N. Komninos, E. Philippou, and A. Pitsillides. Survey in smart grid and smart home security: Issues, challenges and countermeasures. *IEEE Communications Surveys Tutorials*, 16(4):1933–1954, 2014.

[86] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 839–858, 2016.

[87] T. Kothmayr, W. Hu, C. Schmitt, M. Bruenig, and G. Carle. Poster: Securing the internet of things with dtls. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, SenSys '11, page 345–346, 2011.

[88] T. Kothmayr, C. Schmitt, W. Hu, M. Brünig, and G. Carle. A dtls based end-to-end security architecture for the internet of things with two-way authentication. In *37th Annual IEEE Conference on Local Computer Networks - Workshops*, pages 956–963, 2012.

[89] T. Kothmayr, C. Schmitt, W. Hu, M. Brünig, and G. Carle. Dtls based security and two-way authentication for the internet of things. *Ad Hoc Networks*, 11(8):2710 – 2723, 2013.

[90] D. Kozlov, J. Veijalainen, and Y. Ali. Security and privacy threats in iot architectures. In *Proceedings of the 7th International Conference on Body Area Networks*, page 256–262, Brussels, BEL, 2012.

[91] A. Le, J. Loo, A. Lasebae, A. Vinel, Y. Chen, and M. Chai. The impact of rank attack on network topology of routing protocol for low-power and lossy networks. *IEEE Sensors Journal*, 13(10):3685–3692, 2013.

[92] S.-G. Lee, S.-Y. Lee, and J.-C. Kim. A study on security vulnerability management in electric power industry iot. *Journal of Digital Contents Society*, 17(6):499–507, 2016.

[93] A. Liu and P. Ning. Tinyecc: A configurable library for elliptic curve cryptography in wireless sensor networks. In *2008 International Conference on Information Processing in Sensor Networks (ipsn 2008)*, pages 245–256, 2008.

[94] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security*, CCS '03, page 52–61, 2003.

[95] R. Liu, Y. Liang, H. V. Poor, and P. Spasojevic. Secure nested codes for type ii wiretap channels. In *2007 IEEE Information Theory Workshop*, pages 337–342, 2007.

[96] R. Liu and W. Trappe. 2010.

[97] K. L. Lueth. State of the iot 2018: Number of iot devices now at 7b – market accelerating. `https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/`, 2018.

[98] H. Mahdavifar and A. Vardy. Achieving the secrecy capacity of wiretap channels using polar codes. *IEEE Transactions on Information Theory*, 57(10):6428–6443, 2011.

[99] A. Manzoor, M. Liyanage, A. Braeke, S. S. Kanhere, and M. Ylianttila. Blockchain based proxy re-encryption scheme for secure iot data sharing. In *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 99–103, 2019.

[100] D. Martins and H. Guyennet. Wireless sensor network attacks and security mechanisms: A short survey. In *2010 13th International Conference on Network-Based Information Systems*, pages 313–320, 2010.

[101] D. M. Mendez, I. Papapanagiotou, and B. Yang. Internet of things: Survey on security and privacy. *CoRR*, abs/1707.01879, 2017.

[102] G. Montenegro, J. Hui, D. Culler, and N. Kushalnagar. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. RFC 4944, Sept. 2007.

[103] G. Montenegro, C. Schumacher, and N. Kushalnagar. IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals, 2007.

[104] A. Mukherjee. Physical-layer security in the internet of things: Sensing and communication confidentiality under resource constraints. *Proceedings of the IEEE*, 103(10):1747–1761, 2015.

[105] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Technical report, Manubot, 2019.

[106] G. Nebbione and M. C. Calzarossa. Security of iot application layer protocols: challenges and findings. *Future Internet*, 12(3):55, 2020.

[107] P. A. Networks. 2020 unit 42 iot threat report. 2020.

[108] J. Newsome, E. Shi, D. Song, and A. Perrig. The sybil attack in sensor networks: Analysis & defenses. In *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*, page 259–268, 2004.

[109] K. T. Nguyen, M. Laurent, and N. Oualha. Survey on secure communication protocols for the internet of things. *Ad Hoc Networks*, 32:17 – 31, 2015. Internet of Things security and privacy: design methods and optimization.

[110] K. Norrman, F. Lindholm, E. Carrara, J. Arkko, and M. Naslund. MIKEY: Multimedia Internet KEYing, Aug. 2004.

[111] K. Norrman, D. McGrew, M. Naslund, E. Carrara, and M. Baugher. The Secure Real-time Transport Protocol (SRTP). RFC 3711, Mar. 2004.

[112] G. Noubir and G. Lin. Low-power dos attacks in data wireless lans and countermeasures. *SIGMOBILE Mob. Comput. Commun. Rev.*, 7(3):29–30, July 2003.

[113] T. OConnor and D. Reeves. Bluetooth network-based misuse detection. In *2008 Annual Computer Security Applications Conference (ACSAC)*, pages 377–391, 2008.

[114] Y. Ouyang, Z. Le, D. Liu, J. Ford, and F. Makedon. Source location privacy against laptop-class attacks in sensor networks. In *Proceedings of the 4th International Conference on Security and Privacy in Communication Netowrks*, SecureComm '08, 2008.

[115] M. O. Ozmen and A. A. Yavuz. Low-cost standard public key cryptography services for wireless iot systems. In *Proceedings of the 2017 Workshop on Internet of Things Security and Privacy*, IoTS&amp;P '17, page 65–70, 2017.

[116] L. Peng, W. Ru-chuan, S. Xiao-yu, and C. Long. Privacy protection based on key-changed mutual authentication protocol in internet of things. In L. Sun, H. Ma, and F. Hong, editors, *Advances in Wireless Sensor Networks*, pages 345–355, 2014.

[117] C. E. Perkins and E. M. Royer. Ad-hoc on-demand distance vector routing. In *Proceedings WMCSA'99. Second IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, 1999.

[118] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar. Spins: Security protocols for sensor networks. In *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, MobiCom '01, page 189–199, 2001.

[119] C. P. Pfleeger, S. L. Pfleeger, and J. Margulies. *Security in Computing (5th Edition)*. Prentice Hall Press, USA, 5th edition, 2015.

[120] T. Qiu, N. Chen, K. Li, M. Atiquzzaman, and W. Zhao. How can heterogeneous internet of things build our future: A survey. *IEEE Communications Surveys Tutorials*, 20(3):2011–2027, 2018.

[121] R. A. Rahman and B. Shah. Security analysis of iot protocols: A focus in coap. In *2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC)*, pages 1–7, 2016.

[122] W. Raja, M. Anwar, A. Bakhtiari, A. Zainal, K. Abdullah, and K. Qureshi. Security issues and attacks in wireless sensor network. 30:1224–1227, 06 2014.

[123] S. Randhawa and S. Jain. Data aggregation in wireless sensor networks: Previous research, current status and future directions. *Wireless Personal Communications*, 97(3):3355–3425, 2017.

[124] D. R. Raymond and S. F. Midkiff. Denial-of-service in wireless sensor networks: Attacks and defenses. *IEEE Pervasive Computing*, 7(1):74–81, 2008.

[125] S. Raza, S. Duquennoy, T. Chung, D. Yazar, T. Voigt, and U. Roedig. Securing communication in 6lowpan with compressed ipsec. In *2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*, pages 1–8, 2011.

[126] S. Raza, S. Duquennoy, J. Höglund, U. Roedig, and T. Voigt. Secure communication for the internet of things—a comparison of link-layer security and ipsec for 6lowpan. *Security and Communication Networks*, 7, 12 2014.

[127] S. Raza, H. Shafagh, K. Hewage, R. Hummen, and T. Voigt. Lithe: Lightweight secure coap for the internet of things. *IEEE Sensors Journal*, 13(10):3711–3720, 2013.

[128] S. Raza, D. Trabalza, and T. Voigt. 6lowpan compressed dtls for coap. In *2012 IEEE 8th International Conference on Distributed Computing in Sensor Systems*, pages 287–289, 2012.

[129] S. Raza, L. Wallgren, and T. Voigt. Svelte: Real-time intrusion detection in the internet of things. *Ad Hoc Netw.*, 11(8):2661–2674, Nov. 2013.

[130] A. S. Reegan and E. Baburaj. Key management schemes in wireless sensor networks: A survey. In *2013 International Conference on Circuits, Power and Computing Technologies (ICCPCT)*, pages 813–820, 2013.

[131] J. R. R. Renofio, M. E. Pellenz, E. Jamhour, A. Santin, M. C. Penna, and R. D. Souza. On the dynamics of the rpl protocol in ami networks under jamming attacks. In *2016 IEEE International Conference on Communications (ICC)*, pages 1–6, 2016.

[132] E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3, Aug. 2018.

[133] E. Rescorla and T. Dierks. The Transport Layer Security (TLS) Protocol Version 1.2, Aug. 2008.

[134] E. Rescorla and N. Modadugu. Datagram Transport Layer Security Version 1.2. RFC 6347, 2012.

[135] M. Rushanan, A. D. Rubin, D. F. Kune, and C. M. Swanson. Sok: Security and privacy in implantable medical devices and body area networks. In *2014 IEEE Symposium on Security and Privacy*, pages 524–539, 2014.

[136] M. Rushanan, A. D. Rubin, D. F. Kune, and C. M. Swanson. Sok: Security and privacy in implantable medical devices and body area networks. In *2014 IEEE Symposium on Security and Privacy*, pages 524–539, 2014.

[137] S. Sahraoui and A. Bilami. Compressed and distributed host identity protocol for end-to-end security in the iot. In *2014 International Conference on Next Generation Networks and Services (NGNS)*, pages 295–301, 2014.

[138] Y. B. Saied and A. Olivereau. D-hip: A distributed key exchange scheme for hip-based internet of things. In *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–7, 2012.

[139] P. Saint-Andre. Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence, Mar. 2011.

[140] S. M. Sajjad and M. Yousaf. Security analysis of ieee 802.15.4 mac in the context of internet of things (iot). In *2014 Conference on Information Assurance and Cyber Security (CIACS)*, pages 9–14, 2014.

[141] R. Sakai and M. Kasahara. Id based cryptosystems with pairing on elliptic curve. *IACR Cryptology ePrint Archive*, 2003, 04 2003.

[142] E. Sasoglu and A. Vardy. A new polar coding scheme for strong security on wiretap channels. In *2013 IEEE International Symposium on Information Theory*, pages 1117–1121, 2013.

[143] Sencun Zhu, S. Setia, S. Jajodia, and Peng Ning. An interleaved hop-by-hop authentication scheme for filtering of injected false data in sensor networks. In *IEEE Symposium on Security and Privacy, 2004. Proceedings. 2004*, pages 259–271, 2004.

[144] K. Seo and S. Kent. Security Architecture for the Internet Protocol. RFC 4301, 2005.

[145] Y. Seralathan, T. T. Oh, S. Jadhav, J. Myers, J. P. Jeong, Y. H. Kim, and J. N. Kim. Iot security vulnerability: A case study of a web camera. In *2018 20th International Conference on Advanced Communication Technology (ICACT)*, pages 172–177. IEEE, 2018.

[146] D. P. Shah and P. G. Shah. Revisting of elliptical curve cryptography for securing internet of things (iot). In *2018 Advances in Science and Engineering Technology International Conferences (ASET)*, pages 1–3, 2018.

[147] A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, Nov. 1979.

[148] A. Shamir. Identity-based cryptosystems and signature schemes. In *Proceedings of CRYPTO 84 on Advances in Cryptology*, page 47–53, Berlin, Heidelberg, 1985.

[149] Z. Shelby, K. Hartke, and C. Bormann. The Constrained Application Protocol (CoAP), June 2014.

[150] M. Singh, M. A. Rajan, V. L. Shivraj, and P. Balamuralidhar. Secure mqtt for internet of things (iot). In *2015 Fifth International Conference on Communication Systems and Network Technologies*, pages 746–751, 2015.

[151] M. Singh, A. Singh, and S. Kim. Blockchain: A game changer for securing iot data. In *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, pages 51–55, 2018.

[152] A. Soni, R. Upadhyay, and A. Jain. Internet of things and wireless physical layer security: A survey. In *Computer Communication, Networking and Internet Security*, pages 115–123, Singapore, 2017.

[153] N. Sornin, M. Luis, T. Eirich, T. Kramp, and O. Hersent. Lorawan specification. LoRa Standard, LoRa alliance, 2015.

[154] K. Sowjanya, M. Dasgupta, S. Ray, and M. S. Obaidat. An efficient elliptic curve cryptography-based without pairing kpabe for internet of things. *IEEE Systems Journal*, 14(2):2154–2163, 2020.

[155] F. Stajano and R. Anderson. The resurrecting duckling: security issues for ubiquitous computing. *Computer*, 35(4):supl22–supl26, 2002.

[156] F.-X. Standaert, G. Piret, N. Gershenfeld, and J.-J. Quisquater. Sea: A scalable encryption algorithm for small embedded applications. In *Smart Card Research and Advanced Applications*, pages 222–236, 2006.

[157] M. Suarez-Albela, P. Fraga-Lamas, and T. M. Fernandez-Carames. A practical evaluation on rsa and ecc-based cipher suites for iot high-security energy-efficient fog and mist computing devices. *Sensors (Basel, Switzerland)*, 18(11), November 2018.

[158] P. Szczechowiak and M. Collier. Tinyibe: Identity-based encryption for heterogeneous sensor networks. In *2009 International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, pages 319–354, 2009.

[159] B. Tackmann. *A Theory of Secure Communication*. PhD thesis, 2014.

[160] A. Thangaraj, S. Dihidar, A. R. Calderbank, S. W. McLaughlin, and J. Merolla. Applications of ldpc codes to the wiretap channel. *IEEE Transactions on Information Theory*, 53(8):2933–2945, 2007.

[161] D. Thangavel, X. Ma, A. Valera, H. Tan, and C. K. Tan. Performance evaluation of mqtt and coap via a common middleware. In *2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, pages 1–6, 2014.

[162] E. Vasilomanolakis, J. Daubert, M. Luthra, V. Gazis, A. Wiesmaier, and P. Kikiras. On the security and privacy of internet of things architectures and systems. In *International Workshop on Secure Internet of Things (SIoT)*, pages 49–57, 2015.

[163] F. Vidal Meca, J. H. Ziegeldorf, P. M. Sanchez, O. G. Morchon, S. S. Kumar, and S. L. Keoh. Hip security architecture for the ip-based internet of things. In *2013 27th International Conference on Advanced Information Networking and Applications Workshops*, pages 1331–1336, 2013.

[164] B. Viswanath, A. Post, K. P. Gummadi, and A. Mislove. An analysis of social network-based sybil defenses. In *Proceedings of the ACM SIGCOMM 2010 Conference*, SIGCOMM '10, page 363–374, 2010.

[165] L. Wallgren, S. Raza, and T. Voigt. Routing attacks and countermeasures in the rpl-based internet of things. *International Journal of Distributed Sensor Networks*, 9(8):794326, 2013.

[166] Y. Wang, Q. Wu, D. S. Wong, B. Qin, S. S. M. Chow, Z. Liu, and X. Tan. Securely outsourcing exponentiations with single untrusted program for cloud storage. In *Computer Security - ESORICS 2014*, pages 326–343, 2014.

[167] Wenliang Du, Jing Deng, Y. S. Han, and P. K. Varshney. A key predistribution scheme for sensor networks using deployment knowledge. *IEEE Transactions on Dependable and Secure Computing*, 3(1):62–77, 2006.

[168] D. J. Wheeler and R. M. Needham. Tea, a tiny encryption algorithm. In *Fast Software Encryption*, pages 363–366, Berlin, Heidelberg, 1995.

[169] A. D. Wood and J. A. Stankovic. Denial of service in sensor networks. *Computer*, 35(10):54–62, 2002.

[170] W. Wu, E. Liu, X. Gong, and R. Wang. Blockchain based zero-knowledge proof of location in iot. *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pages 1–7, 2020.

[171] A. D. Wyner. The wire-tap channel. *The Bell System Technical Journal*, 54(8):1355–1387, 1975.

[172] W. Xu, W. Trappe, Y. Zhang, and T. Wood. The feasibility of launching and detecting jamming attacks in wireless networks. In *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '05, page 46–57, 2005.

[173] J. Xue, Z. Yang, X. Yang, X. Wang, L. Chen, and Y. Dai. Votetrust: Leveraging friend invitation graph to defend against social network sybils. In *2013 Proceedings IEEE INFOCOM*, pages 2400–2408, 2013.

[174] L. Yang, Chao Ding, and Meng Wu. Establishing authenticated pairwise key using pairing-based cryptography for sensor networks. In *2013 8th International Conference on Communications and Networking in China (CHINACOM)*, pages 517–522, 2013.

[175] M. Young and R. Boutaba. Overcoming adversaries in sensor networks: A survey of theoretical models and algorithmic approaches for tolerating malicious interference. *IEEE Communications Surveys Tutorials*, 13(4):617–641, 2011.

[176] Yu Liu, Yang Li, and Hong Man. Mac layer anomaly detection in ad hoc networks. In *Proceedings from the Sixth Annual IEEE SMC Information Assurance Workshop*, pages 402–409, 2005.

[177] K. Zhang, X. Liang, R. Lu, and X. Shen. Sybil attacks and their defenses in the internet of things. *IEEE Internet of Things Journal*, 1(5):372–383, 2014.

[178] W. Znaidi, M. Minier, and J.-P. Babau. An ontology for attacks in wireless sensor networks. 01 2008.