IoT Application Layer: Communication protocols, Security requirements and Latest Cyber Attacks.

[1]Amos K. Kibet [2] Rosanna A. Esquivel

[1]Angeles University Foundation, [2]Angeles University Foundation
[1]Graduate School,
[1]Angeles University Foundation, Angeles City, Philippines

Abstract: In this day and age, Internet use is growing, which has led to development of the Internet of Things (IoT), IoT facilitates the communication, computation and coordination of machines and objects. IoT is classified in two types, first is inside of IoT and second is outside of IoT. Inside of IoT is consider as protocols. Outside IoT is consider as sensor, actuators and other physically components. In inside of IoT there are protocol stack which have different layers like Application layer, Transport layer, Internet layer and Physical/Link layer. IoT's layers role is to ensure open and effective communication and transaction between two objects and use of different applications to build a continuous bond between them. IoT has many obstacles and security/privacy are some of the core issues associated with IoT's broad application.

Index Terms: **Internet of Things (IoT), layered architectures, Application layer protocols, cyber attacks**

I. INTRODUCTION

About 58% of the world's population is estimated to use the internet **[1]**.The increase of internet usage is because of smartphones, In contrast to a laptop or desktop, a smartphone has improved features. Smartphones are more personal and it improves the experience of the user while using it. Again, reduced internet cost has led to high internet usage, large networks around the world now offer low-cost 4G/5G high-speed Internet. Another aspect of growing Internet users is that people can connect and synchronize through the Internet with other people across the world. The internet usage has allowed objects and machines to connect, communicate transact with each other and has brought rise to what is called Internet of Things (IoT) **[2].**

IoT transforms "things" from passively computing and making individual decisions to actively and omni presently communicating and collaborating to make a single crucial decision. The underlying technologies of ubiquitous computing, embedded sensors, and communication protocols allow IoT to achieve its aims, however, these communication protocols impose lots of threats to the entire IoT ecosystem **[3].**

In this paper, we highlight IoT communication protocols that are operating at different layers of the networking. We specifically narrow it down to Application layer communication protocols, security requirement and the recent cyber threats associated with Application layer, the reason why we concentrated in application layer is because this is final host layer which is closest to the end user and it presents potential intruders with the biggest attack surface. The application layer includes the user interface and various other critical functions, and if successfully exploited entire networks maybe controlled, user data may be stolen, and individual applications may fall under an intruder’s control.

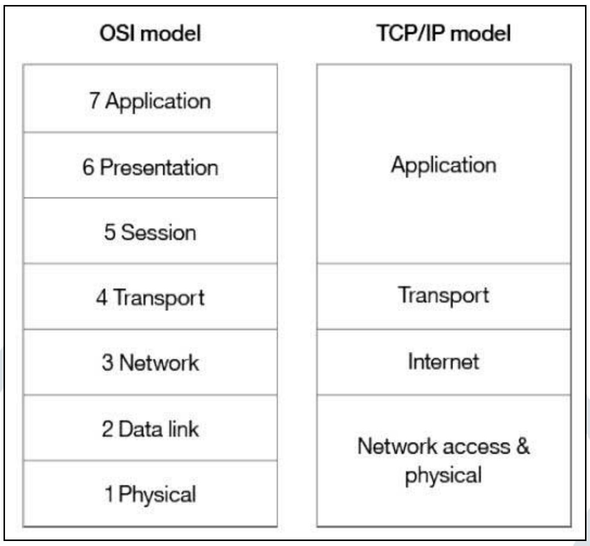
There are many benefits provided by IoT, but, on the other side, it also has some challenges such as poor management, energy efficiency, identity management, security and privacy **[4].** Security and privacy are the most critical issues facing in the development of IoT. In IoT, all devices are connected to the Internet because, without the Internet, they cannot perform their tasks. There are many attackers on the Internet that steal the confidential information of objects. The attackers can use information of users in any illegal way according to their needs, which can result in a great loss for user’s trust **[5].** Privacy has also become an issue for IoT. This means that the information of users must be in secure hands and not be accessible to anyone except authentic users **[6].** Therefore, security and privacy should be ensured by preventing unauthorized access of IoT devices.

The primary objectives of this article to provide an overview of all layered IoT architecture, present Application Layer communication technologies protocols, its security requirement, highlight the latest trending security attacks currently occurring on the application layer and how they affect IoT applications and users. The remaining part of this paper is organized as follows: Section 2 describes the IoT model layers and narrows down to application layer communication protocols, Section 3 handles the IoT issues and security requirement while Section 4 discuss Security Issues associated with application, Section 5 summarize our discussion and highlights the main points presented and future research.

II. IoT MODEL LAYERS

Open Systems Interconnection (OSI) model is the standard ISO model for the internet. The OSI model architecture has seven Layers - Physical, Data Link, Network, Transport, Session, Presentation and Application. Though the actual implementation of OSI model is done through TCP-IP model which simplifies the seven-layer OSI model to four-layer internet protocol suite. In TCP-IP model (realistic implementation of OSI Model), the physical and data link layer are merged to form physical and network access layer and the session, presentation and application layers of OSI model are merged to a single application layer **[7].**

Figure 1: The ISO model.



2.1 Application Layer

Application layer refers to OSI Layer 5, 6 and 7. In IOT architecture, this layer lies above the transport layer. It is highest layer in the architecture extending from the client side. It is the interface between the end devices and the network. This layer is implemented at the end of the device through a dedicated application. Application layer is implemented by the browser. It is the browser that enforces application layer protocols such as Hyper Text Transfer Protocols (HTTP), Hyper Text Transfer Protocol Secure (HTTPS), Simple Mail Transfer Protocol (SMTP), and File Transfer Protocol (FTP). In the same way, IOT also has application layer protocols specified in context.

This layer is accountable for formatting and presenting data. The internet application layer is typically based on the HTTP protocol. However, in resource-constrained environments, HTTP is not suitable because it is extremely heavyweight and therefore entails a high overhead parsing and IoT is heavy weight processing. There are many alternative protocols that have been developed for IOT situations. Common Communication protocols in all the layer include are as shown below:

Table 1: Layers and Protocols.

Layer	Protocols
Application Layer	CoAP, MQTT,XMPP, AMQP, RESTFUL,
Transport Layer	UDP, DTLS
Internet Layer	RPL, 6LoWPAN
Physical/Link Layer	IEEE 80

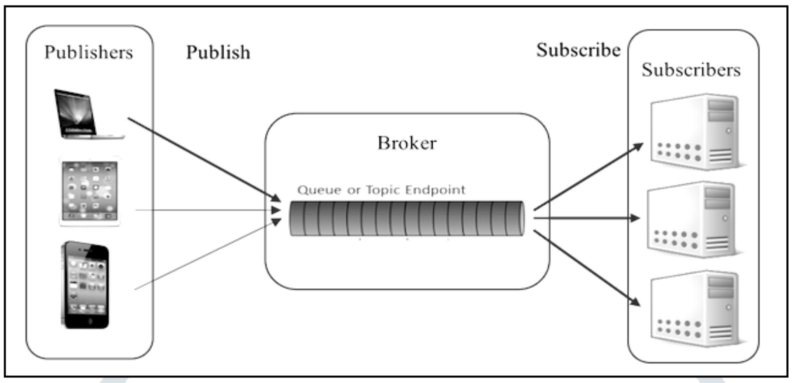
2.1.1 Message Queue Telemetry Transport. (MQTT).

MQTT was introduced by IBM in 1999 and standardized by OASIS in 2013 **[8].** It is intended to provide integrated connectivity between applications and middleware on one side and networks and communications on the other side, MQTT system is made up of three main units: publisher, subscriber, and broker. It is intended as a lightweight messaging protocol that uses publish / subscribe operations to exchange data between clients and servers. In addition, its small size, low energy usage, reduced data packets and ease of implementation make the protocol ideal for the "machine-to-machine" or "Internet of Things" world.

Components of MQTT:

- Broker is the server that handles the data transmission between the clients.
- A topic is the place a device want to put or retrieve a message to/from.
- The message is the data that a device receives
- Publish is the process a device does to send its message to the broker.
- Subscribe is where a device does to retrieve a message from the broker.

Figure 2: MQTT Structure.



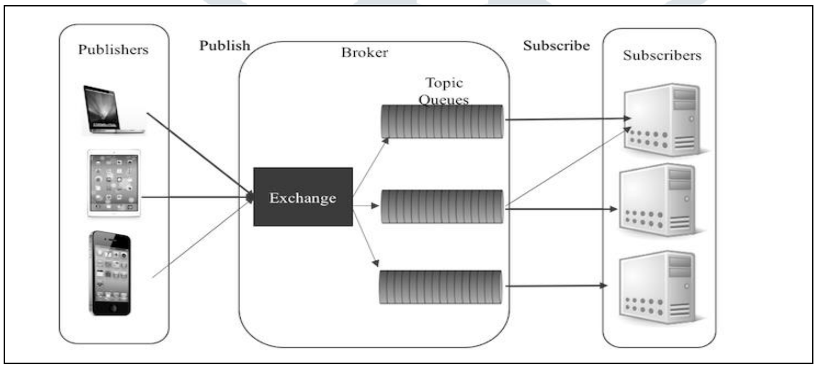
2.1.2 The Advanced Message Queuing Protocol. (AMQP).

AMQP is also another session communication protocol designed for the financial services industry. It operates over TCP and provides publish / subscribe structure similar to that of MQTT. The difference is that the broker is divided into two key components: exchange and queue **[9].**

Components of AMPQP:

- **Broker** is the server that handles the data transmission between the clients.
- **A topic** is the place a device want to put or retrieve a message to/from.
- **The message** is the data that a device receives “when subscribing” from a topic or send “when publishing” to a topic.
- **Publish** is the process a device does to send its message to the broker.
- **Subscribe** is where a device does to retrieve a message from the broker.
- **Exchange** is part of the broker (i.e. server) which receives messages and routes them to queues
- **Queue** (message queue) is a place where messages are received with and from where consumers.
- **Binding** gives rules for distributing messages from exchanges to queues

Figure 3: AMQP Structure.



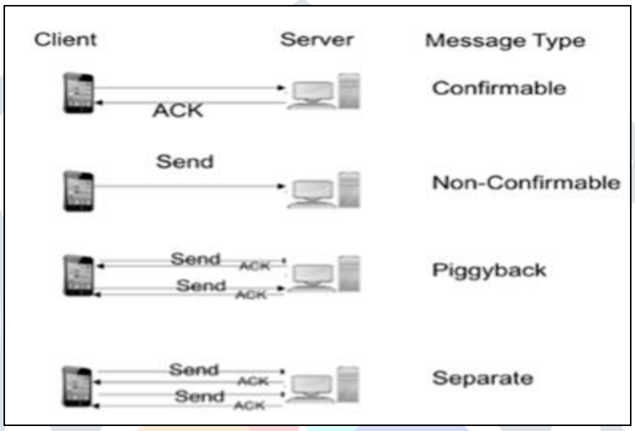
2.1.3 Constrained Application Protocol (CoAP)

CoAP is another session communication protocol designed for Constrained Application Protocol is specifically designed for constrained (limited) Hardware.The hardware that doesn't support HTTP or TCP/IP can use CoAP Protocol. It is a lightweight protocol that needs low power IOT application like for communication between battery powered IOT devices. Like HTTP, it also follows client-server model. The clients can GET, PUT, DELETE or POST informational resources over the network. CoAP makes use of two message types - requests and responses. The messages contains a base header followed by message body and length is specified by the datagram. The messages or data packets are small in size, so that they can communicate among constraint devices without data losses. The messages can be confirmable or non-confirmable. In confirmable messages, the client need to respond with an acknowledgement after receiving the data packet. The request messages can contain query strings to implement additional functionalities like search, sort or paging **[9].**

Components of CoAP:

CoAP architecture is divided into two main sublayers: messaging and request/response. The messaging sublayer is responsible for reliability and duplication of messages while the request/response sublayer is responsible for communication. As shown in Figure 4, CoAP has four messaging modes: confirmable, non- confirmable, piggyback and separate.

Figure 4: CoAP Structure.



Confirmable Message (CON): Some messages require an acknowledgement. These messages are called "Confirmable". When no packets are lost, each confirmable message elicits exactly one return message of type Acknowledgement. Confirmable message always carries either a request or response and MUST NOT be empty.

Non-Confirmable Message (NON): Some other messages do not require an acknowledgement. This is particularly true for messages that are repeated regularly for application requirements, such as repeated readings from a sensor where eventual arrival is sufficient. A non-confirmable message always carries either a request or response, as well, and MUST NOT be empty.

Piggy-Backed Message: It occurs when any response is sent immediately by the server after receiving a confirmable or non-confirmable message also termed as an acknowledge message.

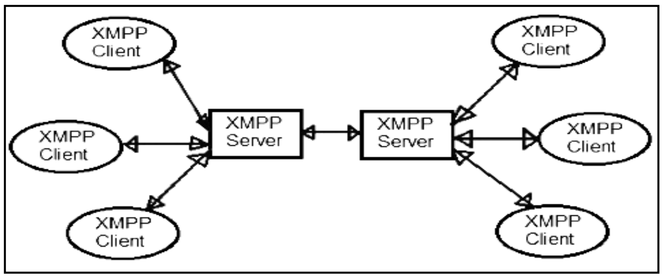
Separate Message: it occurs when the server sends an empty message rather than an acknowledgment. The purpose is to stop the client from resending the message. This message generally takes some time for delivery. The server will send a confirmable message when it is ready.

2.1.4 Extensible Messaging and Presence Protocol. (XMPP)

XMPP is a messaging protocol designed for message transfer applications. It is well known and has known to be highly effective on the internet. It has recently been reused for IoT applications as well as a protocol for Software Defined Networks. This reuse is to implement its use of XML, which makes it easy to extend. XMPP favors both publish / subscribe architecture and the request / response architecture. It's intended for near real-time applications and, thus, efficiently supports low-latency small messages.

XMPP does not provide any quality of service guarantees and, hence, is not practical for M2M communications. Moreover, XML messages create additional overhead due to lots of headers and tag formats which increase the power consumption that is critical for IoT application. Hence, XMPP is rarely used in IoT but has gained some interest for enhancing its architecture in order to support IoT applications **[10].**

Figure 5: XMPP Structure.

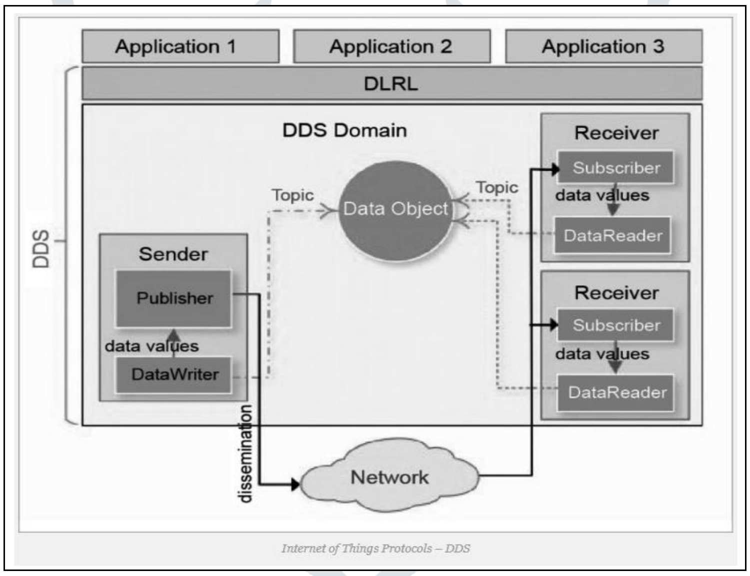


2.1.5 Data Distribution Service. (DDS).

DDS is another publish/subscribe protocol that is designed by the Object Management Group (OMG) for M2M communications. The basic benefit of this protocol is the excellent quality of service levels and reliability guarantees as it relies on a broker-less architecture, which suits IoT and M2M communication.

It defines two sublayers: data-centric publish - The first takes the responsibility of message delivery to the subscribers and data-local reconstruction sublayers allows a simple integration of DDS in the application layer.
Publisher layer is responsible for sensory data distribution.
Data writer interacts with the publishers to agree about the data and changes to be sent to the subscribers.
Subscribers are the receivers of sensory data to be delivered to the IoT application.
Data readers basically read the published data and deliver it to the subscribers and the topics are basically the data that are being published **[11].**

Figure 5: DDS Structure.



III. IOT SECURITY REQUIREMENTS

The IoT is growing very rapidly. In the coming years, vehicles, manufacturing, supply chain, agriculture, healthcare and energy industries is expected to witness the wide spread adoption of the IoT and flourish through new IoT devices and applications. These industries devices and applications are expected to deal with vital private information such as personal data. In addition to this, such smart devices will be connected to global information networks for their access anytime, anywhere. Therefore, the IoT domain may be a target of attackers. To facilitate the full adoption of the IoT in various industries, it is critical to identify and analyze distinct features of IoT security and privacy, including security requirements, vulnerabilities, threat models and countermeasures, from the IoT perspective **[12].**

3.1 IoT security requirements:

Confidentiality: It is essential to ensure data protection and availability for only authorized users. The term user in IoT includes human as well as machines and services and also includes internal objects and external objects. It is important for IoT users to be mindful of the data management mechanisms to be applied, the process or the person responsible for the management and to ensure data privacy across the process.

Integrity: it ensures that data obtained are not changed by an opponent in transit. The IoT is centered on data sharing and information between different types of devices, which is why it is important to make sure data accuracy; that data is received from the right recipient as well as to ensure that the data is not recorded during the transmission process

Availability: it ensures the survival of authorized parties to IoT services (whether private/public or cloud services) even under denial-of-service attacks when needed. IoT users should have all the data available with them so that they can reach the IoT services whenever they need it.

Authentication: it enables an IoT device to guarantee the identity of the peer it communicates. Each IoT object must be able to accurately identify and authenticate other objects and requires a mechanism to authenticate entities each time IoT interacts.

Authorization: it takes place when identity is successfully verified by the system, allowing complete access to the information and resources. However, rights to access resources only are verified with authorization after the system determined and in what extent users are able to access the system. Permission is the process to determine if the authenticated user has access to the specific resources.

Fault Tolerance: it is security scheme that enables a system to continue operating properly in the event of the failure of (or one or more faults within) some of its components.

IV. LATEST APPLICATION LAYER ATTACKS

IoT will be extensively used throughout our lives, and the safety hazards will be greater in these vital areas, including in the business and social environment at home, at work and in different areas. Several kinds of attacks may jeopardize IoT. These attacks might be at the physical or software level. IoT devices have different memory amounts and various computer resource abilities when they are connected. Protecting these 'things' from different types of attacks, security personnel need know the latest attacks that affects application layer so that appropriate security structure and services should be provided.

4.1 Latest security threats in application layer:

HTTP flood: It is a type of layer 7 application attack that affects web servers that collect data using the GET requests. Cybercriminals send requests from GET or POST to a web server target. The specific purpose of these requests is to consume considerable resources. Bots begin with a given HTTP link and repetitively follow every link on the provided website. This is how HTTP flood attacks are launched **[13].**

Main-in-The-Middle (MiTM) Attack: It is an attack in which the attacker sneakily intercepts the sender and recipient who believe that they communicate directly. Since an attacker controls communication, he or she can therefore modify messages as necessary. It poses a serious security threat because it enables the attacker to capture and manipulate information in real time **[14].**

Denial-of-Service attack (DoS): Most IOT devices have small memory and limited calculation resources and are therefore vulnerable to attacks on resource and network comprehension. Attackers may send numerous requests for specific things to be processed in order to drain their resources. DoS attacks may cause belting in the communication channel and thus break the communication channel between nodes. Network availability can also be disrupted by flooding the network with a large number of packets, consumption of computational resources like bandwidth, memory, disk space, or processor time

Injection: it allow attackers to modify a command back end statement by means of unaccounted user input as the all-time favorite category of application attacks. Injections are of the following kind: SQL, Code Injection, OS command, LDAP injection, XML Injection, XPath Injection, SSL Injection IMAP / SMTP Injection, Buffer Overflow All of this includes untrustworthy and handling Web-enabled requests, commands and queries.

SQL injection continues to be the most leading Injection attack, this is because Organizations don’t expeditiously apply security patches to their applications or databases. Running an old code, agencies are vulnerable in their website applications and databases to attack. Currently there is rapid development and developers of still creating vulnerable applications. A lack of awareness of safe coding methods, coupled with the perception of the long and costly building of secure software were the first steps SQL injection.

Broken Authentication and Session Management: Several types of programming flaws allow attackers to bypass the authentication procedures that are used by an application, it includes all aspects of handling user authentication and managing active sessions. Authentication is a critical aspect of this process, but even solid authentication mechanisms can be undermined by flawed credential management functions, including password change, forgot my password, remember my password, account update, and other related functions **[15].** A wide array of account and session management flaws can result in the compromise of user or system administration accounts. Development teams frequently underestimate the complexity of designing an authentication and session management scheme that adequately protects credentials in all aspects of the site.

Cross site scripting (XSS): This is a common attack vector that injects malicious code into a vulnerable web application. XSS differs from other web attack vectors in that it does not directly target the application itself. Instead, the users of the web application are the ones at risk **[16].** A successful cross site scripting attack can have devastating consequences for an online business's reputation and its relationship with its clients, depending on the severity of the attack, user accounts may be compromised, trojan horse programs activated and page content modified misleading users into willingly surrendering their private data.

Cross site scripting attacks can be broken down into two types: *Stored XSS*, also known as persistent XSS, is the more damaging of the two. It occurs when a malicious script is injected directly into a vulnerable web application. *Reflected XSS* involves the reflecting of a malicious script off of a web application, onto a user's browser. The script is embedded into a link, and is only activated once that link is clicked on.

Insecure Direct Object References: The insecure direct object references vulnerability allows an attacker to steal other users' data of a specific type. Outside just the data in a database, an attacker can exploit it to access restricted files or directories on the server. An insecure direct object references vulnerability is commonplace and easy to exploit **[17].** It is caused when a direct reference to a restricted object is exposed to users as part of the URL parameter. In addition, the application fails to verify whether the user is authorized to access the requested object for which the reference is present in the request URL. The name parameter in this URL specifies the exact filename to retrieve. Attackers can modify this predictable parameter to access reports for other months.

Security Misconfiguration: Poorly configured app-security can come in many different forms. Misconfigurations can occur in a developer's own code, in the code of pre-made features and functions, or through the API. They can appear in the app itself, in the servers and databases used by the app, or in resources used during the development process. Any level of an organization's application stack can manifest a configuration flaw, and the more layers there are the greater the chances for a mistake leading to a vulnerability. Test systems may not be properly firewalled from production systems; and the basic principal of least privilege is not always enforced. As an application grows in scope, it becomes more difficult to keep security configurations tight and effective **[18].**

Security vulnerabilities can be exposed from unexpected places. Error messages may contain clues for attackers if they're improperly handled. Leftover code and sample applications from the development process may contain known vulnerabilities, allowing attackers to gain access to the application server. When not properly configured, debugging information like these error messages and detailed stack traces, vital to the developer, can become weapons in an attacker's hand. This misconfiguration issue can also occur on private servers that use third-party software.

Sensitive Data Exposure: The confidence that the server protects this sensitive data is the foundation for any information entered by the user on a web application. However, the breaches of data lead to skepticism over and over again. The vulnerability of sensitive user data exposure has also increased with the multitude of today's web applications **[19]** and vulnerable unstructured data like files, documents, and sensitive information were involved. Hackers take advantage of inadequate security and unencrypted data stored, transmitted or processed. Sensitive data exposure, vulnerability occurs when a web application fails to adequately protect sensitive information from being revealed to illegitimate users.

Both external and internal attacks may result in the exposure of sensitive data. A disgruntled employee presents more risk than any external employee because the employee already has access to and the power to abuse the company's information. Cloud storage is a convenient option in order to store your data, but it provides an open platform for attacks when it is not correctly secured. Bad attacks may also occur if a suspicious link is accidentally clicked, unsustainable websites visited or malicious malware downloaded.

Missing Function Level Access Control: If the authentication check of sensitive request processors is not adequate or if the vulnerability is missing, the Access Control level function may be categorized as missing **[20].** Accessing a URL that contains any sensitive information or exposes functionality intended only for authorized users indicates missing access control.

Cross-Site Request Forgery (CSRF): This an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. CSRF attacks specifically target state-changing requests, not theft of data, since the attacker has no way to see the response to the forged request, with a little help of social engineering (such as sending a link via email or chat), an attacker may trick the users of a web application into executing actions of the attacker's choice. If the victim is a normal user, a successful CSRF attack can force the user to perform state changing requests like transferring funds, changing their email address, and so forth. If the victim is an administrative account, CSRF can compromise the entire web application **[21].**

Using Components with Known Vulnerabilities: This category is about using unpatched third-party components. Attackers can easily exploit old third-party components because their vulnerabilities have been publicized, and tools and proof of concepts often allow cybercriminals to take advantage of these flaws with ease. Any script can conduct an exploit.

V. CONCLUSION AND RECOMMENDATION

The emerging idea of Internet of Things (IoT) is quickly finding its path throughout our modern life, aiming to enhance the quality of life by connecting various smart devices, technologies and applications. Generally, the IoT would allow for the automation of everything around us. This paper presented an IoT overview and its security requirement. We have articulated different research about layered architectures of IoT, we narrowed down to Application Layer and also described security attacks based on Application layers that can affect the performance of IoT. We have surveyed the literature on the existing attacks and we in future will work on developing security structure and services based on Blockchain for IoT security.

REFERENCES

[1] Internet Users. (2019). http:// http://www.internetlivestats.com//

[2] Oppitz M., Tomsu P. Inventing the Cloud Century. Springer; Cham, Switzerland: (2018). Internet of Things; pp. 435–469.

[3] Hwaiyu Geng (2017). Internet of Things and Data Analytics Handbook.

[4] Yaqoob I., Ahmed E., Hashem I.A.T., Ahmed A.I.A., Gani A., Imran M., Guizani M. (2017). Internet of Things Architecture: Recent Advances, Taxonomy, Requirements, and Open Challenges. IEEE Wireless. Communication.

[5] Jing Q., Vasilakos A.V., Wan J., Lu J., Qiu D. (2014). Security of the Internet of things: Perspectives and challenges. Wireless. Network.

[6] Sicari S., Rizzardi A., Grieco L.A., Coen-Porisini A. (2015). Security, privacy and trust in Internet of Things: The road ahead. Computer. Network.

[7] Priya Khachane. (2016.) IOT-Architecture-Standards-Protocols

[8] Locke D. Locke. (2010). MQ telemetry transport (MQTT) v3. 1 protocol specification.

[9] Tara Salman, Raj Jain. (2017). A_Survey_of_Protocols_and_Standards_for_Internet_of_Things.

[10] Tara Salman, Raj Jain. (2017). A_Survey_of_Protocols_and_Standards_for_Internet_of_Things.

[11] Hwaiyu Geng. (2017). Internet of Things and Data Analytics Handbook.

[12] Anil Chacko1, Thaier Hayajneh1, (2018). Security and Privacy Issues with IoT in Healthcare.

[13] Ali B., Awad A.I. (2018). Cyber and Physical Security Vulnerability Assessment for IoT-Based Smart Homes. Sensors.

[14] Muhammad Burhan, 1 Rana Asif Rehman, 1 Bilal Khan,1 and Byung-Seo Kim2,* (2018). IoT Elements, Layered Architectures and Security Issues: A Comprehensive Survey.

[15] Daniel Blazquez. (2019). OWASP - broken-authentication-and-session-management.

[16] Imperva inc (2019). Application-security/cross-site-scripting-xss-attacks/.

[17] Chetan Karande. (2019). https://www.oreilly.com/library/view/securing-node applications/9781491982426/ch04.html.

[18] Open Web Application Security Project - OWASP 2018 https://www.immuniweb.com/blog/OWASP-security-misconfiguration.html.

[19] Breeze Telecom. (2016). sensitive-data-exposure-vulnerability-causes-and-prevention

[20] Open Web Application Security Project - OWASP 2018 https://www.owasp.org/index.php/Top_10_2013-A7 Missing_Function_Level_Access_Control.

[21] Open Web Application Security Project - OWASP 2018 https://www.owasp.org/index.php/Cross-Site_Request_Forgery (CSRF).