School of Computer Sciences, Universiti Sains Malaysia

CST334 NETWORK MONITORING & SECURITY

Semester I, 2020/2021

ASSIGNMENT 1

Lecturer: Assoc. Prof Dr Aman Jantan, PhD

Literature Review on Vulnerabilities in Application Layer of OSI Model

Name: Muhammad Iqrar Amin

Matric: 140659

E-mail: iqraramin@student.usm.my

Table of Contents

List of Figures

List of Tables

Abstract

. In this review I have performed a systematic literature review on vulnerabilities that exists in application layer and its protocols. I have also reviewed some of the ways an individual can utilize to minimize the risk of getting effected by those vulnerabilities. I have complied in this study vulnerabilities in about 10 popular protocols, 7 from regular devices (web, desktop and mobile) and 3 from Internet of Things devices. I have also researched and complied few of the solutions to overcome these vulnerabilities. The results of this review justify the need for new security tools that will assists personal and enterprise organization to defend against popular attacks and vulnerabilities.

1 Introduction

. Application layer is the closest layer to the end user in Open Systems Interconnection (OSI) model, hence covering the most of the threat surface for hackers to attack [1]. Due to this reason, application layer security becomes very important which refers to the different ways of protecting applications at the application layer (layer 7 of OSI model) from malicious attacks. Having a poor application layer security can lead to performance and stability issues along with data lose and nonoperational networks.

According to [2], there is rise of about 40 % in vulnerabilities in application layer alone from August 2019 to September 2019, in only one mere month. Moreover, about 47% vulnerabilities in 2019 have publicly available exploit which any hacker or script kiddie can use to exploit into a system. In addition to this, about 40.2% vulnerabilities do not have a solid solution available at the moment, such as software upgrades, workaround or software patch, and these number are rising as we speak. In this review we will be looking at few of these vulnerabilities that exist in application layer, more specifically the services and protocol that run on application layer, since all the applications running on a system has to use these services in order to communicate over the stack.

This specific paper or rather literature review aims to answer the question: What are the vulnerabilities that exists in application layer of OSI model and its protocols? and What are some counter measure that can be taken against the vulnerabilities in application layer? A literature review is carried out to identify, analyze, and interpret available information (relevant papers) by using appropriate methods.

In this paper, I will perform a systematic literature review on the vulnerabilities in Open Systems Interconnection (OSI) model. This study includes the analysis performed on information gathered from 20 different publications for the whole review and most of the publications answers the research questions. In Section 2, I have briefly introduced or rather overviewed Open Systems Interconnection (OSI) by International Standards Organization (ISO) and Layer 7 of OSI model: Application Layer. In Section 3, I have discussed about the systematic review performed on the topic and the methodology/process used to carry out the successful review. Moving on to Section 4, I have analyzed the findings from all the publications and summarized them in tabular manner and discussed the vulnerabilities in detail. Then in Section 5, I have suggested few possible solutions to counter the vulnerabilities discussed in Section 4. Finally, I have concluded my study/review in Section 6.

2 Background

This section of the review contains an overview of Open Systems Interconnection (OSI) model and overview of Application Layer or Layer 7 of OSI model.

2.1 Overview Open Systems Interconnection (OSI) Model

In 1977, the International Standards Organization (ISO) gathered a team to work on a new architectural design to breakdown the task of communication between variety of system of different sub-tasks. The manageable open system protocol is the result of this "divide and conquer approach". Later in 1984, this model was used for development of an open communication system standard known as Open System Interconnect (OSI) model [3].

OSI model describes seven layers, that explain how network-aware device applications will connect with each other. The model is general and extends to all kinds of networks, not only TCP/IP, not just Ethernet, and all media types. The layer in the OSI layer are as follow: Physical Layer, Data Link Layer, Network Layer, Transport Layer, Session Layer, Presentation Layer and Application Layer, as depicted in Figure 2.1 along with the respective protocols used in each layer.

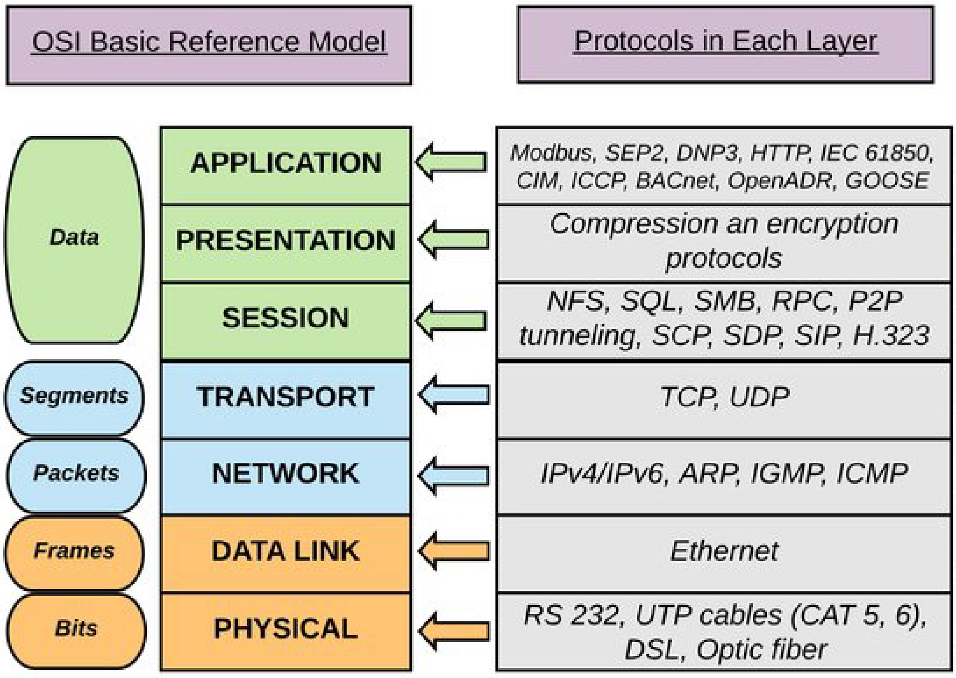
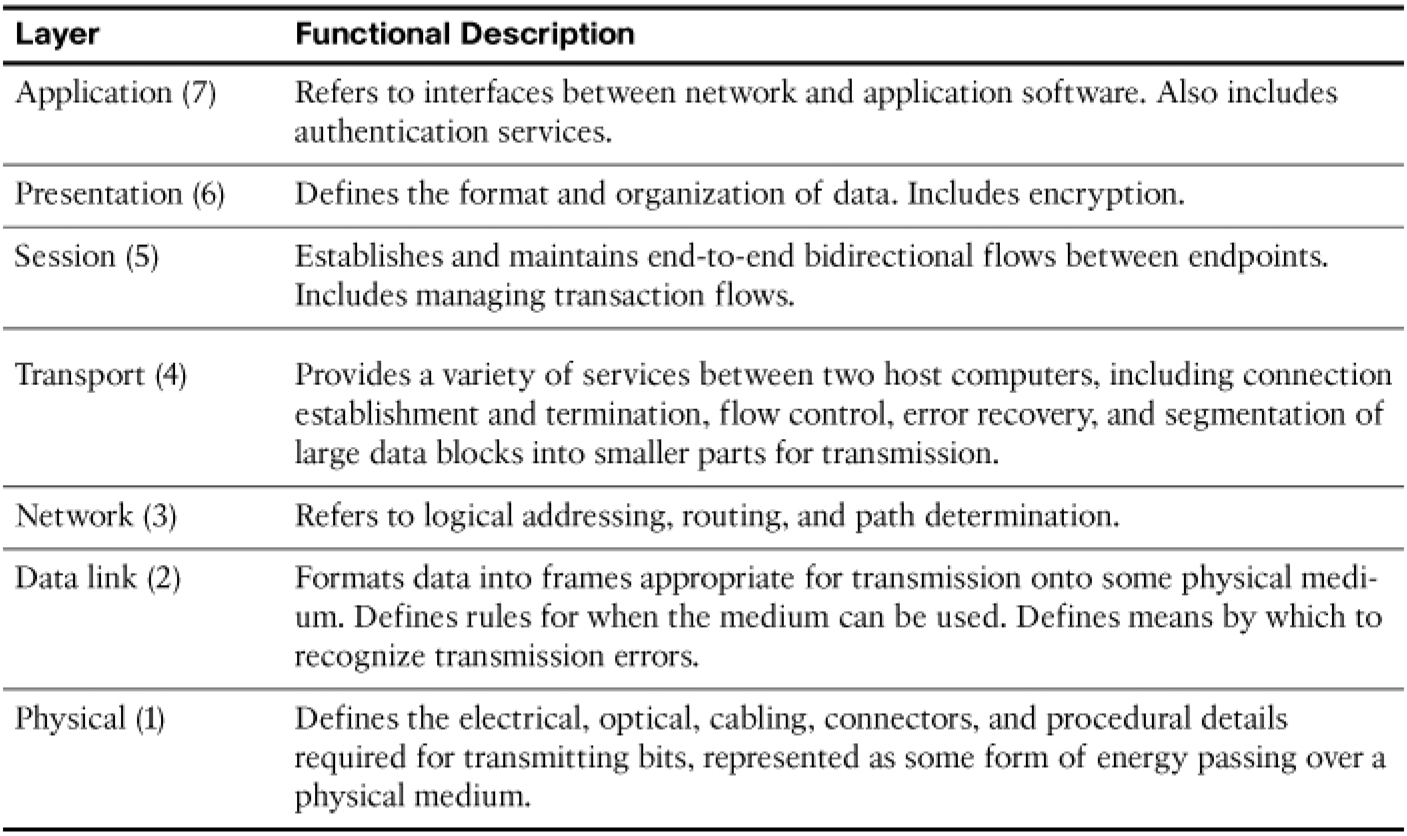

OSI Basic Reference Model
Protocols in Each Layer
Data
Segments
Packets
Frames
Bits
APPLICATION
PRESENTATION
SESSION
TRANSPORT
NETWORK
DATA LINK
PHYSICAL
Modbus, SEP2, DNP3, HTTP, IEC 61850, CIM, ICCP, BACnet, OpenADR, GOOSE
Compression an encryption protocols
NFS, SQL, SMB, RPC, P2P tunneling, SCP, SDP, SIP, H.323
TCP, UDP
IPv4/IPv6, ARP, IGMP, ICMP
Ethernet
RS 232, UTP cables (CAT 5, 6), DSL, Optic fiber

Figure 2.1: OSI model seven layers

Each layer of OSI model has its own functions, which can be seen in Table 1, in transmitting data from one location to another. When an application sends a data to be transmitted to other application, Application layer first transform the information to raw data and send it to Presentation layer where it got compressed and encrypted. Then Session layer establish the flow to the other session layer of receiving end. Then the data get transformed into segments in Transport layer for it to be transmitted easily. Network layer access these segments and convert them into packets to transmit over router, which may travel through several routers before reaching destination. After all packets are ready to be transmitted, Data Link layer make frames out of packets, so they are appropriate for transmission over physical network. Lastly, Physical layer transmits the frames in bits over some medium. When these bits are received at receivers end all the mentioned process happens in reverse.

Table 2.1: The functional description of OSI model layers



Layer	Functional Description
Application (7)	Refers to interfaces between network and application software. Also includes authentication services.
Presentation (6)	Defines the format and organization of data. Includes encryption.
Session (5)	Establishes and maintains end-to-end bidirectional flows between endpoints. Includes managing transaction flows.
Transport (4)	Provides a variety of services between two host computers, including connection establishment and termination, flow control, error recovery, and segmentation of large data blocks into smaller parts for transmission.
Network (3)	Refers to logical addressing, routing, and path determination.
Data link (2)	Formats data into frames appropriate for transmission onto some physical medium. Defines rules for when the medium can be used. Defines means by which to recognize transmission errors.
Physical (1)	Defines the electrical, optical, cabling, connectors, and procedural details required for transmitting bits, represented as some form of energy passing over a physical medium.

2.2 Overview of Application Layer (Layer 7)

The seventh layer of Open Systems Interconnection (OSI) is known as Application Layer [4]. This layer operates as a window for application services. There is a lot of confusion that application layer get mixed with being an application but that is not the case, it is a channel through which other application can communicate.

The application layer sits at the top of the seven-layer architecture of Open Systems Interconnect (OSI), providing applications and clients with network connectivity. Because users generally communicate with this layer the most, protocols for the application layer and the software that implements them frequently concentrate on functionality rather than security. Many of these protocols were developed well before the security of the network was deemed a significant concern, and the application layer protocols and applications are vulnerable to a number of attacks as a result [5].

Application layer exists between users and presentation layer as shown in Figure 2.2. When one user need to send data to other user, data passes through one of the services/protocols running at application layer (Telnet in this case) which then gets encapsulated with header and get passed down to presentation layer. At receiving end this data gets decapsulated after being received from presentation layer and passed through the same service/protocol as sender (Telnet) after getting received by receiver.

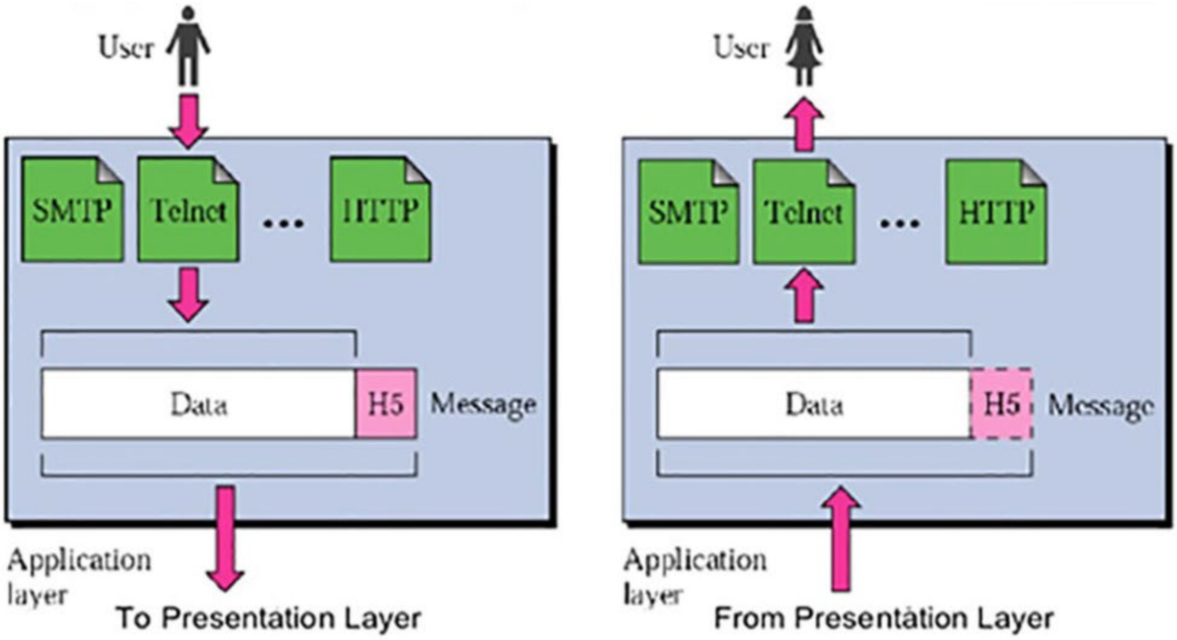


Figure 2.2: Application layer basic architecture

3 Review and Analysis Methodology

I carried out a systematic literal review to identify the vulnerabilities that exists in application layer of OSI model in different platforms and mediums. The details description of literature review and the methodology used for evaluation is provided below.

3.1 Systematic Literature Review (SLR)

I used the methods and strategies for the review proposed by Frâncila Weidt Neiva and Rodrigo Luis de Souza da Silva [6], and Kitchenham et al. [7]. Th following are the descriptions of the activities performed by me.

3.1.1 Systematic Literature Review Question

There are in total 2 research questions around which the whole SLR will circulate, the objective of this review is to identify the vulnerabilities that exists in application layer of OSI model. Hence, I have derived the following research questions to support my research:

RQ1: What are the vulnerabilities that exists in application layer of OSI model and its protocols?

RQ2: What are some counter measure that can be taken against the vulnerabilities in application layer?

3.1.2 Search Strategy

A Systematic Literature Review focuses on searching the scientific databases to get the desired information. The database utilized in this SLR search is listed below:

- IEEE Xplore
- ResearchGate
- Academy of Computing Machinery Digital Library (ACM)
- Science Direct
- Springer Link
- Others

After identifying the scientific databases, I processed with the keywords that are related to the research questions and will help to get research materials from the databases. I used Boolean operation such as “OR” and “AND” to widen the search and get most accurate studies to further do the review. The list of all the keywords are listed in the Table 3.1: Keywords used in search. There are two classification of keywords, Class 1, Class 2, and Class 3, I search all the possible combination of Class 1 AND Class 2 AND Class 3.

Table 3.1: Keywords used in search

Class	Keywords
1	"Application Layer" "Layer 7 of OSI" "Seventh Layer of OSI" "Application Layer Protocols"
2	"vulnerabilities" "risks" "attacks"
3	"Internet of Things" "IoT" "Web Applications" "Mobile Applications"

Based on the above list the following search string was generated to execute in the search engines:

(("Application Layer" OR "Layer 7 of OSI" OR "Seventh Layer of OSI" OR "Application Layer Protocols" OR) AND ("vulnerabilities" OR "attacks" OR "risks") AND ("Internet of Things" OR "IoT" OR "Web Applications" OR "Mobile Applications"))

3.1.3 Reviewing the Search Results and Inclusion Decision

The studies were conducted on selected studies and research based on the inclusion and exclusion criteria. The inclusion criteria as the name suggests includes the research and studies on which the data extraction will be performed, while exclusion criteria includes the criteria which will be used to exclude the studies that are not related from the review. Table 3.2 shows the inclusion and exclusion criteria for selecting the appropriate studies.

Table 3.2: Inclusion and Exclusion Criteria

Type	Criteria
Inclusion Criteria	• Studies that implement application layer protocols in their applications • Studies that discuss about the vulnerabilities and risks in application layer of different applications
Exclusion Criteria	• Studies in languages other than English • Duplicated studies • Studies that restricts from accessing them, in other word full text is not available for them.

The search for the study was performed in four different stages depicted in Figure 3.1: Stages of the search performed in the review. In the very first stage there were around 950 search results from all the database together, on stage 2 the duplicate results were eliminated resulting in 400 studies. In the third stage, the studies were selected based on appropriate title and abstract which reduced the count of studies from 400 to 120. Finally, a comprehensive study was carried out on the leftover studies and 20 studies were selected from them.

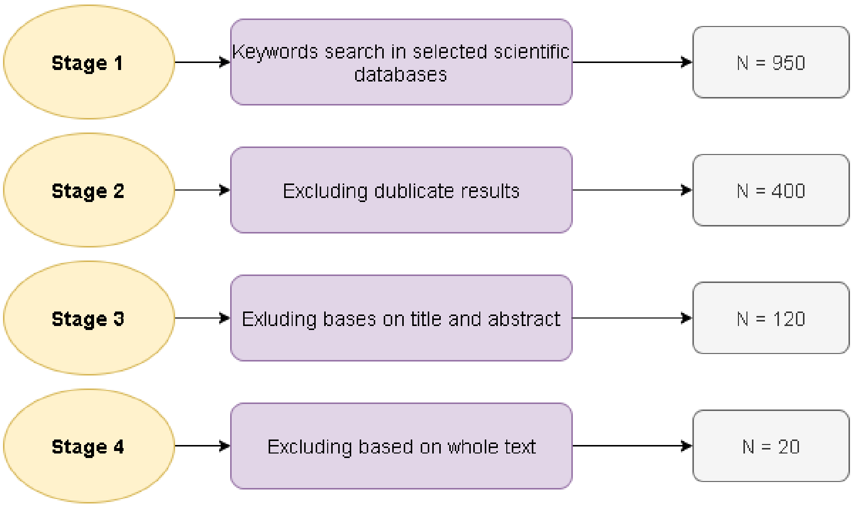

Stage 1
Keywords search in selected scientific databases
N = 950
Stage 2
Excluding dublicate results
N = 400
Stage 3
Exluding bases on title and abstract
N = 120
Stage 4
Excluding based on whole text
N = 20

Figure 3.1: Stages of the search performed in the review.

Once the studies were handpicked after executing inclusion and exclusion criteria, a comprehensive Quality Assessment was conducted to ensure the qualitative property of the search/researched data. In this specific SLR, I have considered the following four QA questions and a marking scheme was implemented on them which was Complete (C) = 1, Partial (P) = 0.5, None (N) = 0. The following is the list of QA questions:

1. Does the study reflect the implementation of application layer protocols in software development?
2. Does the study address the vulnerabilities and risks in application layer of different applications (IoT, Web, Mobile)?
3. Does the study discuss on research process?

After performing QA on the studies, out of 223 studies only 35 were selected to perform comprehensive review on them and find out the factors that contribute to the lack of security in agile methods.

3.1.4 Data Extraction

After carefully selecting the studies for the review that data extraction strategy was applied to each study to gather required information from them. Table 3.3 shows the description of the components in the extraction form. After the data is extracted the comprehensive analysis was done on the gather data.

Table 3.3: Data Extraction Form

Data Extracted	Description
ID	Unique id for each preferred study
Author	Name of the author of the study
Title	Title of the respective study
Year	Year of Publication
Publisher	Publisher's name
Type	Study type (Journal. Research Report, Conference Proceedings, Book Section)
Domain	Research domain (for example Artificial Intelligence, Software Engineering)
Methodology	The technique used to write the study e.g. case study, systematic review, empirical study, interview to get data, observation
Vulnerabilities	Information about the vulnerabilities that are responsible for the lack of security in Application Layer.
Attacks	Different attacks used in the study
Layers	Application Layer used in the study

3.1.5 Results

In the decisive review, 20 studies have been selected and the distribution of the studies selected are shown in Table 3.4, most are from IEEE Xplore (30%), Springer Link (25%), and ResearchGate (20%).

Table 3.4: Distribution of studies from various databases

Scientific Database	Stage 1	Stage 2	Stage 3	Stage 4	Weight
IEEE Xplore	450	175	40	6	30%
ResearchGate	169	90	10	4	20%
ACM	43	30	5	1	5%
Science Direct	38	10	5	1	5%
Springer Link	120	50	30	5	25%
Others	130	45	30	3	15%
Total	950	400	120	20	100%

4 Vulnerabilities in Application Layer

In this section of the review, I will try to answer the first research question **RQ1,** which seek to vulnerabilities in application layer of OSI model while serving different types of applications such as web-based, desktop-based, mobile applications and IoT devices. The Table 4.1 shows the finding for vulnerabilities for several application layer protocols. It also includes some of the Common Vulnerabilities and Exposures (CVE) for corresponding protocol.

Table 4.1: Application layer protocols vulnerabilities

Sr No.	**Application Layer Protocol**	**CVE Reference Code(s)**	**Potential Attack(s)**	**Reference**
Application Layer of Standard (Web, Desktop, Mobile) devices				
1	File Transport Protocol (FTP)	CVE-2019-9760, CVE-2018-18861, CVE-2018-15516	Remote Code Execution, Buffer Overflow, PORT Bouncing	[8] [9]
2	Teletype network (Telnet)	CVE-2020-8797, CVE-2018-10698	Command Line Injection, Man-in-the-Middle	[10]
3	Simple Mail Transfer Protocol (SMTP)	CVE-2019-19660, CVE-2019-19215, CVE-2018-0203	Cross-Site Request Forgery, Buffer Overflow, Denial of Service (DoS)	[11] [12]
4	Doman Name Service (DNS)	CVE-2006-3441, CVE-2019-0811, CVE-2008-1447	Buffer Overflow, Denial of Service (DoS), Cache Poisoning	[13]
5	Trivial File Transfer Protocol (TFTP)	CVE-2018-10389, CVE-2019-5482	Denial of Service (DoS), Buffer Overflow	[14]
6	Hypertext Transfer Protocol (HTTP)	CVE-2020-8427, CVE-2019-16935, CVE-2011-4404, CVE-2019-1010094	SQL Injection, Cross Site Scripting (XSS), Directory Traversal Attacks, Cross Site Request Forgery	[15] [16] [17]
7	Simple Network Management Protocol (SNMP)	CVE-2018-19132, CVE-2006-5538, CVE-2020-6060	Denial of Service (DoS), Format String Attacks, Buffer Overflow	[18] [19]
Application Layer of Internet of Things (IoT) devices				
8	Message Queue Telemetry Transport (MQTT)	CVE-2020-10071, CVE-2019-10244	Buffer Overflow, XML External Entity (XEE)	[20] [21]
9	Constrained Application Protocol (CoAP)	CVE-2020-3162, CVE-2019-17211	Denial of Service (DoS), Integer Overflow	[20]
10	Extensible Messaging and Presence Protocol (XMPP)	CVE-2020-3495, CVE-2017-14486	Remote Code Execution, Sniffing	[21] [22] [23]

All the application layer protocols in this review are discussed below with more details, including several vulnerabilities in each one of them:

1. **File Transport Protocol (FTP):**

   File Transfer Protocol was first introduced in year 1975 [24] and was the choice of protocol for file transfers between hosts and to the internet. It operates at port 20 and 21. The most concerning issue in FTP is that then data shared in FTP is in clear text, including all the passwords, username, and personal information and attackers can easily intercept it and use it to login to several other systems [9]. PORT bouncing is another security issue/vulnerability found in FTP operated servers. By using PORT bouncing an attacker can request access to server ports indirectly by using victims' machine [8].

2. **Teletype network (Telnet):**

   The telnet protocol is a TCP shell service that runs on port 23, it enables the connection by establishing the session with a host on another site. Although Telnet may be designed to allow anonymous connections, usernames and passwords should also be configured to include them. Even then, alas, Telnet sends them in plain text. Attacker may take advantage of that and use attacks such as CL Injection and MIMT to get the shell where they will execute any authorized task while a user is logged in [10].

3. **Simple Mail Transfer Protocol (SMTP):**

   Operating on port 25, SMTP is used for as a mean to transport electronic mail (e-mail) among different hosts with transmission protocols such as TCP/IP [11]. Due to the lack of authentication in SMTP, it is very easy to spoof email addresses. Another issue with SMTP is the weak server configurations for email receiving, this could allow attacker to not only include malicious code in emails and cause buffer overflow upon execution, but it also make is host for spamming resulting in denial of service [12].

4. **Doman Name Service (DNS):**

   DNS is application layer service that operates on port 53 and is commonly used for translating website addresses into corresponding IP addresses and vice versa. It uses both TCP and UDP for transmission but for IP translation purposes it only uses UDP since it is utilizing smaller headers. Buffer overflow attacks can result in gaining command level access to the DNS server or changing files in the region. DoS attacks may have a huge effect on the DNS database and its customers worldwide. Typically, they are geared into root servers. Cache Poisoning can be carried out with the use of spoofed query which typically follows by a spoofed answer which could data in cache including some compromised keys [13].

5. **Trivial File Transfer Protocol (TFTP):**

TFTP is a connectionless version of FTP which uses UDP to decrease overhead and reliability on port 69. Since it does not run on TCP, it does lack session management and authentication overhead which possess a great security issue. It is vulnerable to a buffer overflow attack which occurs when a longer file name is inserted at the time of request for read/write. TFTP is also vulnerable to format string attacks where attacker can perform denial of service attack via format string sequences in TFTP error packet [14].

6. **Hypertext Transfer Protocol (HTTP):**

HTTP is a TCP service which runs on port 80. HTTP contributed to making the Internet the famous tool it is today. As a stateless connection, the HTTP connection paradigm is established. Injection vulnerabilities, such as injection of SQL, OS, and LDAP, occur when untrusted data is submitted as part of a command or query to an interpreter [16]. Whenever a program takes untrusted data and sends it to a web browser without sufficient authentication or escape, XSS vulnerabilities exist [15]. A CSRF attack causes a logged-on victim's browser to send a forged HTTP request to a vulnerable web application, containing the victim's session cookie and all other authentication information [17].

7. **Simple Network Management Protocol (SNMP):**

SNMP is a UDP service that runs on ports 161 and 162 and has been developed to track networks effectively and inexpensively. The buffer overflow vulnerability in SNMP occurs in while handling multiple connections, when a specially timed sequence of SNMP connection can trigger a stack overflow which results in Denial of Service (DoS) [18]. Just like in the case of buffer overflow, denial of service can also be triggered using a format string attack [19].

8. **Message Queue Telemetry Transport (MQTT):**

MQTT is an application layer protocol that is most widely used in IoT devices for message passing. It is a light weight many to many protocols developed for TCP communication. The natural MQTT is a plain protocol and hence is vulnerable to buffer overflow attacks which can be triggered by executing code through open ports which could potential trigger remote code execution or denial of service [20]. It is also vulnerable to XEE attacks attack due to an improper factory and parser initialization [21].

9. **Constrained Application Protocol (CoAP):**

   CoAP operates on UDP instead of TCP where clients and servers can communicate via connectionless datagrams. Retries and recording is also implemented on CoAP in application stack. It also serves as a web transfer protocol for IoT devices. A vulnerability in CoAP could allow an unauthorized remote access to an attacker to cause denial of service attack on IoT device, the reason for this vulnerability is the insufficient input validation of incoming traffic [20]. Another vulnerability exists in CoAP message block, when two unit16_t type of variable are added up, they return a result that wrap around maximum unit16_t value causing insufficient buffer space to be allocated for CoAP message where attacker can place any code to be executed on system or server [20].

10. **Extensible Messaging and Presence Protocol (XMPP):**

    XMPP is an application layer protocol operating on TCP, which is intended to be used for real time communication along with streaming of XML data between different networks. Few of its applications include gaming, voice/video streaming, messaging, and data syndication [21]. XMPP is vulnerable to remote code execution which is due to invalidation of message body. An attacker can easily exploit it by sending XMPP message with malicious code to be executed on the IoT device [22]. Some version of XMPP operated devices uses clear text to send messages, which introduces another vulnerability where attacker can sniff the XMPP network to obtain user credentials, messages and other sensitive information [23].

5 Possible Solutions

In this section of the review, I will try to answer the second research question **RQ2** and look into some possible solutions for each application layer protocol that I presented in last section. There are multiple solution of each protocol, but due to the limit of this review, only few of them will be discussed in a short and concise manner. The solution for each application layer protocol are discussed in detail below:

1. **Solutions for File Transfer Protocol (FTP):**

   As mentioned before, FTP send data in plain text, hence using an encrypted protocol to send files can greatly increase the security of files. By simply using Secure File Transfer Protocol (SFTP), one can encrypt the transferred data, hence increasing the security. Moreover, we can also implement detection system which can detect and parse the commands before executing so if there is PORT command (for PORT Scanning attacks) or "…/" (for Directory Traversal attacks), it can automatically block those requests [8].

2. **Solutions for Teletype Network (Telnet):**

   In order to secure telnet protocol, user need to configure the telnet server with Secure Sockets Layer (SSL) to encrypt and secure communication over telnet sessions. A simple way could be installing Digital Certificate Manager (DCM) to configure the certificate for Telnet server to utilize. Another solution is to use Secure Shell (SSH) to establish session with another host. It is a cryptographic network protocol that securely connects over unsecure networks, and greatly increasing the security [10].

3. **Solutions for Simple Mail Transfer Protocol (SMTP):**

   The first solution is to install digital certificates for SMTP make the connection necessary to use SSL certificate which is quite simple to achieve, this simple change will add high security to mailing server. The other solutions for attack specific, for instance, for format string attacks, users can utilize tools such as Starce, Pscan and Itrace to remove format string vulnerability by helping us to identify the interaction of programs with the system. They also help consumer analyze the code to check format string and alert the user. Where as buffer overflows attacks can be blocked by restricting the argument of certain commands, so if a length of argument increases past certain limit, the connection will be dropped [12].

4. **Solutions for Domain Name Services (DNS):**

   According to [13], Doman Name System Security Extension (DNSSEC) can be used to add extra security in DNS protocol by providing origin authentication, data integrity and authenticated denial of existence to DNS data provided by a name server. DNSSEC utilizes a digital certificate system where user can check the signatures on those certificates to identify if the traffic is coming from a legitimate user.

5. **Solutions for Trivial File Transfer Protocol (TFTP):**

   Trivial File Transfer Protocol is mostly vulnerable to two type of attacks, buffer overflow and format string attacks. The best solution to fight against the buffer overflow attacks is to set a detection device to restrict argument length of commands to 513 bytes, and if increase this limit the detection device should drop the connection. Now, to prevent against format string vulnerability, the detection device should be capable of detecting a pattern having %s, %n, %d, %u, %x, %g, %i, %c, %e, %E, %X, %p, in read/write commands, and if there are such pattern it should raise alert and drop the connection [14].

6. **Solutions for Hypertext Transfer Protocol (HTTP):**

   HTTP is probably the mostly used and most vulnerable application layer protocol due to its connection with internet. The Cross-Site Scripting (XSS) are one of the most common attacks in HTTP, and they need the most amount of attention in order to patch them. To prevent XSS attacks, the user has to be very careful in creating the application to make sure it has proper validation for headers, cookies, query strings, form fields and hidden fields. To prevent from injection attacks, the simplest way is to avoid access of any external interpreters as much as possible. As for CSRF, applications must ensure that they do not rely on browsers' automatically submitted credentials or tokens. Lastly, all the request containing malicious code has to be blocked especially the once with "…/" that might be an attempt to carry out Directory Traversal Attack [17].

7. **Solutions for Simple Network Management Protocol (SNMP):**

   Most of the vulnerabilities in SNMP are due to format string or buffer overflow which later results into devastating denial of service attack. Hence a detection device should be maintained at the server side which could detect the potential attack by checking the presence of bulk request. Once the request is identified, the signature should be written as per guidelines to prevent this vulnerability [25].

8. **Solutions for Message Queue Telemetry Transport (MQTT):**

   The vulnerabilities (Buffer Overflow and XML External Entity) that exist in MQTT protocol can be secured by using MQTT over SLL, this is achieved by first creating a private key, then creating a X509 certificate that will use the private key created in first step and finally, the MQTT server certificate will be created which will authenticate any new user over the network [20]. The creation of certificates can be achieved by utilizing **openssl** command in Linux.

9. **Solutions for Constrained Application Protocol (CoAP):**

   The most viable solution for securing Constrained Application Protocol (CoAP) is to utilize Datagram Transport Layer Security (DTLS) which will secure CoAP over UDP. It is to be noticed that TLS is being used to secure HTTP over TCP, while DTLS is used for securing CoAP over UDP. DTLS can be integrated with CoAP to provide end to end security which will secure CoAP from some common vulnerabilities such as denial of service and integer overflow attacks [20].

10. **Solutions for Extensible Messaging and Presence Protocol (XMPP):**

    There are several ways which user can utilize to secure XMPP. Firstly, User has to make suer that the server is running with a valid and up to date server certificate. Secondly, the clear text communication over XMPP should also be disabled and encryption in communication should be enforced, both for client to server (C2S) communication and for server to server communication (S2S) [23].

6 Conclusion

As more and more people are using web, desktop, and mobile applications along with IoT devices, the vulnerabilities associated with it (application layer) is increasing proportionally. With the same pace, hacker are trying to come up with new methods to hack and attack application layer protocols to gain advantage over system. After browsing though literature, I was able to review some of the most popular vulnerabilities in application layer protocol and few methods on how to overcome them.

The results of this study are an evidence that application layer, specifically application layer protocols, lack security and it calls for research to develop more efficient tools and methods introduce more security in application layer.

7 Acknowledgment

I would like of thank Assoc. Prof Dr Aman Jantan, for giving me such an amazing opportunity to work on this literature review study and enhance my Affective Domain skills along with Cognitive skills.

8 References

[1] F. Security, "What is Application Layer Security?," F5 Security, [Online]. Available: https://www.f5.com/services/resources/glossary/application-layer-security. [Accessed 10 11 2020].

[2] S. Magazine, "Cyberattacks, Application Vulnerabilities Increase by 40 Percent in September 2019," Security Magazine, 25 10 2019. [Online]. Available: https://www.securitymagazine.com/articles/91140-cyber-attacks-application-vulnerabilities-increase-by-40-percent-in-september-2019. [Accessed 25 11 2020].

[3] O. S. I. (. Model., *Journal for the Assocication for Laboratory Automation,* vol. 3, no. 1, pp. 28-35, 1998.

[4] F. Halsall, "A review of the ISO application layer protocols," in *IEE Colloquium on Software in Computer Networks*, London, UK, 1988.

[5] M. Gregg and S. Watkins, Hack the Stack, Rockland, Canda: Syngress Publishing, Inc., 2006.

[6] F. W. N. Silva and R. L. d. S. da, "Systematic Literature Review in Computer Science - A Practical Guide," Federal University of Juiz de Fora, November 2016.

[7] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software.," University of Durham Tech. Rep. EBSE-2007-01, Durham, UK, July 2007.

[8] D. Springall, Z. Durumeric and J. A. Halderman, "FTP: The Forgotten Cloud," 2013.

[9] S. Tobias, "Case Study In Secure File Transfer: Implementing Secure FTP with SSL in a Healthcare Organization," 2020.

[10] B. K. Kumar, N. Raj, D. JP and D. Muralidharan, "Fixing Network Security Vulnerabilities in Local Area Network," in *Proceedings of the Third International Conference on Trends in Electronics and Informatics*, 2019.

[11] V. V. Riabov and R. College, "SSMTP ((SSiimpllee Maaiill Trraanssffeerr Prrottoccoll))," 2005.

[12] S. Alarcon, "Port 25 (SMTP) - Remote Sendmail Header Processing Vulnerability: Exploiting the Internet's Second Most Popular Pasttime," 2003.

[13] C. J. Mitchell and S. Ariyapperuma, "Security vulnerabilities in DNS and DNSSEC," Research Gate, London, 2007.

[14] Q. &. Z. Y. Liu, "TFTP vulnerability finding technique based on fuzzing," *Computer Communications,* vol. 31, no. 1, pp. 3420-3426, 2008.

[15] F. Holik and S. Neradova, "Vulnerabilities of modern web applications," in *40th International Convention on Information and Communication Technology, Electronics and Microelectronics*, Opatija, 2017.

[16] H. Atashzar, A. Torkaman, M. Bahrololum and M. H. Tadayon, "A Survey on Web Application Vulnerabilities and Countermeasures," 2011.

[17] OWASP, "The Ten Most Critical Web Application Security Risks," OWASP, 2017.

[18] P. Chatzimisios, "Security issues and vulnerabilities of the SNMP protocol," 2015.

[19] L. Y. Fazal, "Simple Network Management Protocol(SNMP) Vulnerabilities," GIAC, 2003.

[20] A. Burange, H. Misalkar and U. Nikam, "Security in MQTT and CoAP Protocols of IOT’s Application Layer," 2019.

[21] S. N. Swamy, P. D. Jadhav and P. N. Kulkarni, "Security Threats in the Application layer in IOT," in *International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)*, 2017.

[22] S. Narayanaswamy and A. V. Kumar*, "Application Layer Security Authentication Protocols for the Internet of Things: A Survey," *Advances in Science, Technology and Engineering Systems Journal,* vol. 4, no. 1, pp. 317-328, 2018.

[23] M. I. Malik, I. N. McAteer and S. N. Firdous, "XMPP architecture and security challenges in an IoT ecosystem," in *Australian Information Security Management Conference*, Edith, 2018.

[24] A. Bhushan, "File Transfer Protocol," 1971.

[25] S. Jajodia, Vulnerability Analysis and Defense for the Internet, Springer Science, 2008.