ISSN: 2455 - 5630

International Journal of Scientific Research and Modern Education

Impact Factor 7.137, Special Issue, March 2020 – Conference Proceedings

International Conference on Rise of Disembedded Unilateral Economy: InnoVision in the Era of Deglobalization (KRUPACON 2019) On 8th & 9th November 2019 Organized By

Krupanidhi Group of Institutions, Bangalore, Karnataka

SECURE SOFTWARE DEVELOPMENT LIFE CYCLE FOR IOT BASED HEALTH MONITOR SYSTEM

Rajkumar N* & Roopini J**

* Krupanidhi Group of Institutions, Bangalore, Karnataka

** Krupanidhi Degree College, Bangalore, Karnataka

Cite This Article: Rajkumar N & Roopini J, "Secure Software Development Life Cycle for IoT Based Health Monitor System", International Journal of Scientific Research and Modern Education, Special Issue, March, Page Number 28-31, 2020.

Abstract:

The Internet of Things (IoT) has found several uses for healthcare systems with the capacity to connect integrated devices in various systems to the Internet, to provide instant reading, remote supervision and to enable better decision-making. There were other critical weaknesses, such as theft of sensitive health data, hackers eavesdropping to private conversations via the baby monitor, and entering or shutting down a moving Internet-enabled car completely. The understanding of security does not increase at the same rate as the quantity of IoT devices. Consequently, security becomes a major concern with the rise of IoT that links numerous devices and offers new attack vectors. The aim of this essay is to emphasise the benefits of IoT security. The project begins with the design and implementation of a basic health monitor to monitor patients' pulse rates and body temperatures remotely in hospitals. Safety is included into the software development life cycle while building the health monitor (SDLC).The aim of this integration is to show that the key known weaknesses in security are identified and mitigated, providing a highly secure final product. The safety of the health monitor is incorporated in the development of the software (SDLC). This integration is designed to illustrate how important known security defects may be identified and mitigated, leading to a safer end product. The OWASP (Open Web Application Security Project) also examines the ioT Health Monitor for five of the Top 10 IoT Vulnerabilities: (1) Inappropriate authentication/authorisation of the web interface, (3) transmission encryption, (4) privacy concerns and (5) inadequate security configuration. OWASP is a famous non-profit organisation, mostly dedicated to the development of software security tools and standards.

Introduction:

In all aspects of our everyday life IoT is viewed as having an impact. The present 8.4 billion connected items will reach 20.4 billion by the year 2020, according to famous research firm Gartner [1] [2]. Applications have been identified in a wide range of areas, including home automation, medical treatment and production [3] [4] [7] due to rising progress in Iot-based smart systems. The fast growth in IoT applications at large scales has been illustrated by Xu et al. [4]. The transformational implications of the Internet of Things on health cannot be exaggerated. IoT's attractive quality solves significant health cost concerns [7] by allowing embedded devices to be connected through the internet and remote monitoring in real-time, allowing more prudent decision-making. The smooth detection and transfer of vital health sign data without human intervention, in contrast to traditional in-person monitoring systems, lowers errors and cost of transmission. The ability to remotely monitor the vital signs saves time and delay for the doctor who may be deadly when a medical emergency arises.

Health information is very sensitive and must be properly secured in any system that handles it. Certain laws such as the HIPAA (Health Insurance Portability and Accountability Act of 1996) have been created to protect such health information in the US. HIPAA is a US statute which lays forth data security and safety criteria for medical information protection and requires health data processing systems to comply with these requirements. Research has therefore shown that health data processing applications are highly sensitive to harmful attacks by users. [9, 10].

IoT has its own set of security vulnerabilities and its enticing perks. The attack by Malware Mirai, which affected more than 1 million devices, with 96% of IoT devices, has underlined the necessity for IoT security [11]. According to a major study firm IoT is only given around 10 per cent of IT security expenditures to security, which is alarming [12], although it will participate in 25 per cent in assaults by 2020. Targeting physical things through the internet is now possible as IoT [13] is increasing. Even scientists have expressed concern about the apparent lack of security in current IoT applications [14].

In this paper, we suggest that we can fix vulnerability holes in IoT devices with a root-level and granular approach. This means that the OWASP Secure-SDLC standard must be adapted and integrated into the SDLC IoT Application, resulting in a Secure-SDLC for IoT[15]. What better method to demonstrate this than with an IoT-driven health monitor with restrictions on privacy and security. Security demands are incorporated from the original requirements to the final deployment stage. Each level aims at detecting and mitigating the number of known vulnerabilities and preventing them from extending to the next phase of the Secure SDLC. The Open Web Application Security Project (OWASP) gives additional confidence in the IoT Health Monitor for the top 5 IoT vulnerabilities. OWASP provides IoT manufacturers with support in building secure IoT software and regularly categorises the top ten IoT vulnerabilities [16].

IoT based Health Monitor:

The IoT-based health monitoring is feasible with remote surveillance of the heart rate, body temperature and tracking of hospitalised patients. The IoT architecture is separated into three levels, according to Moosavi et al.[17]: (1) Device/Edge Layer, (2) Fog Layer and (3) Internet Layer.

Device / Edge Layer:

The base layer of the device is the physical device such as sensors and microcontrollers. For this health monitor, the pulse and temperature sensors are chosen to collect information on cardiac rate and temperature. By sampling the pulse signals over time, the pulse rate is calculated. The sensors are maintained on the body of the patient. The thermometer is connected as the thermometer is attached to the finger of the patient to the axis. These sensors regularly gather body vitals using the linked microcontroller. The microcontroller does preliminary calculations before transferring the data over Wi-Fi via a Transmission Control Protocol (TCP) link to the Fog layer. Some security features are downloaded to the Fog layer, detailed in the following section, because the resources on the microcontroller are constrained.

Fog Layer:

The Fog layer acts as an intermediate between the Device/Edge Layer and the Internet Layer. This layer has been introduced to provide an additional safety layer. The Fog Layer is a Windows service that takes data from the Device/Edge Layer through a TCP connection, conducts the necessary security processes and transfers the data to the next internet layer. This layer relieves part of the resource-restricted device/edge layer of significant safety demands while maintaining the necessary level of safety. This layer

Internet Layer:

The web layer is the architecture's last tier. This layer consists of the web application and the database/source. The web application is responsible for collecting and entering data in the database from the Fog layer. It is also responsible for presenting the data to end users. End users include patients, guards and physicians. The web programme allows the doctor to watch and make choices on the basis of virtually in real-time medical data without being in contact with the patient physically.

Secure Software Development Life Cycle:

The fundamental SDLC comprises of: (1) the requirements for collection, (2) the development, (3) the development, (4) the testing and (5) its deployment. Secure-SDLC is a modified version of the OWASP standard integrating safety firmly into every phase of the SDLC IoT Application [14]. The procedure starts by defining the security requirements at each SDLC level as described in the following sections.

Requirements Gathering:

This is the first step of the SDLC and involves the compilation of functional and business applications. At these time duties include the collection of safety requirements, the identification of regulatory requirements and conformity requirements and the identification of major safety hazards.

- (1) Authentication, (2) Encryption & Secure Communications, (3) error management, (4) Session management, (5) Logging, (6) Input validation, (7), and (8) storage security demands may be converted into a particular category.
- Recognized and must follow acceptable standards of geographical regulation and compliance; HIPAA and the Payment Card Industry are significant examples in the U.S. (PCI).
- Recognize the inherent hazards of data processed, the functioning of the application and the risks connected with the use of the programme.

Design:

The software and system components are completed during the design phase. The architecture and design of the IoT solution are evaluated throughout the design phase using security threat modelling techniques. Threat players and likely threat situations should be considered for each of the IoT design components. The principal distinguishing feature between classical threat analysis and design assessments is the application of industry-specific threat knowledge. The result of the design review and threat modelling procedures includes only standard and authorised frames, modules, APIs and design requirements. Microsoft's (1) OWASP and (2) threat modelling techniques [18, 19] were the best knowledgeable. This research uses a combination of OWASP and Microsoft methodologies. In all threat modelling methodologies are utilised two basic strategies STRIDE and DREAD. STRIDE is used to identify hazards whereas DREAD is utilised in risk calculations. The danger categories which form the STRIDE acronym allow for the identification and mapping of potential threats:

- Masking: Unauthorized spoofing access.
- Manipulation: manipulation of data without consent.
- Repudiation: a threat by the user to reject his or her behaviour.
- Unauthorized data exposure: Data Exposed Without Authorisation.
- Denial of Service (DoS): If a service for authorised users is not available, the service is termed DoS.
- Privilege Elevation: valid used to obtain excessive privileges to hazardous tactics.

The measurements of the risk calculation for the DREAD acronym are:

- Reproducibility
- Potential damage
- Opportunity to exploit
- Accessibility
- Affected users

Development:

A static security study is carried out throughout the development process. For static security analysis, an automated code review or a manual code review can be employed. The objective of the code review is to identify common coding defects that might lead to security vulnerabilities. Sensitive data like passwords revealed in or hard-coded in the code, vulnerabilities like cross-site scripting, SQL injection, and Cross Site Request Forgery are only a few instances of some frequent mistakes. Sensible information like user input validations that lead to user input validations.

Testing:

The initiative is being pursued to ensure it meets both functional and safety standards. Security penetration tests are dynamic at this stage and include vulnerability assessments and penetration tests. There is a fundamental contrast between risk assessment and penetration tests, namely the vulnerability evaluation, which simply identifies all vulnerabilities. Dynamic testing include OWASP Top 10 vulnerabilities in test scenarios that will provide confidence in the safety position of the product.

Deployment:

Secure configurations are used in addition to the system configurations needed to operate the application. In this step of the SDLC, safety testing is carried out to ensure both that the vulnerabilities identified during the test phase are fixed and that the product delivered is securely configured.

Out Comes:

The main component of the SDLC is then considered: (1) Gathering of Requirements, (2) Design, (3) Development, (4) Testing and (5) Exploitation. In each level of the IoT Application SDLC [14], OWASP Secure-SDLC standard is modified to closely integrate security. The procedure is started by identifying security criteria for each SDLC step as stated in the following sections.

Requirements Gathering:

The following sections explain the various safety criteria for IoT health monitoring.

- Authentication: Mutual authentication between all components of the system must be carried out before any data transfer.
- Authorisation: Access is provided depending on roles of users of the web interface. The idea of least privilege should be observed while providing access to each position.
- Error handling: The software must handle all the error conditions gracefully. The error messages should be simple and should not reveal unnecessary information about the technical system or configuration.
- Login Management: The user' session should be invalidated whenever a user logs out of a web application.
- Logging audits: minimal safety auditing events should be documented. All architectural parts were logged. Registers are a log type.
- Validation of input fields for users: All input fields for the users are required to be validated. Data from the varying confidence limits between distinct architectural components must be validated before any processing and storage.
- Secure and encrypted communication: In transit, all sensitive data must always be encrypted. Communication with SSL (Secure Socket Layer) is a standard data transfer security mechanism.
- Storage: Personally Identifiable Information (PII) must be encrypted during storage, including application credentials for login. Cryption of hard-coded source code credentials.

Design:

We will begin a high-level design test to see whether the present architecture shows obvious safety defects. During this investigation we have recognised the resource-restricted nature of the microcontroller. This restriction makes a safe socket layer (SSL) connection impossible to create a safe transmission of encrypted data between the microcontroller and the web server. At both data transfer points, SSL requires keeping resource-intensive certificates. To circumvent this limitation, the Fog Layer was built. The SSL connection operations were discharged through the Fog layer from the microcontroller. While the risk lessened, data transfers were still encoded between the Fog Layer and the Microcontroller. The transfer of data between the microcontroller and the Fog layer is limited to the local network and cannot be accessed on the internet. This reduces the risk exposure substantially. In order to decrease the residual risk, we have used the Rivest-Shamir-Adleman (RSA) encoder method with enough primary numbers. After the concept has been changed to further analyse risk, the threat model is constructed similar to the explanation above. This was generated using a Microsoft Threat Modelling Tool. The model helps to see the many components. Trust boundary (in red) and data transfer in one-way architecture This enables evaluations of threats on the ground. detail. Every danger is based on the Health Monitor Threat Model.Security needs have been identified at the project requirements phase. Each SDLC contact was considered crucial.

To construct a model of threat, DREAD and STRIDE technology were utilised. The method is utilised for the calculation of risk.

- Interactions between the fog layer and Web application server
- Web application server and database interactions
- Web application server and client browser interactions Web application server.

Development:

Some of the major findings from the Health Monitor review are as follows:

- A link string is hardcoded in the web. config file: Web.config file saves the credentials for connecting to the database. These must be encrypted to avoid leakage of passwords.
- Comments with passwords: Hard-coding credentials are a common approach for helping debugging among developers. These must be deleted before the code is used in the production environment.
- Incomplete logging: Forensic analyses in the case of a security violation or incident are permitted by logs. The logs must include extensive data and be protected against tampering, to be considered reliable. At the absolute least, logs should include the time stamp, event and user or system information that led to the occurrence.

Testing and Deployment:

To test the application in a runtime context, vulnerability scanners and interception tools are employed. These tools also offer test scenarios of the OWASP Top 10 vulnerabilities. The critical vulnerabilities discovered are as follows:

- Failure to verify input: Some input fields have not been checked such that they are subject to cross-site scripting (XSS). All user input fields are validated to correct this.

- Management of the problem: Just a few occurrences of error disclosed the stack trace of the programme. This is corrected by presenting all problems with a universal error page.

Deployment:

The remediation of the defects detected so far in all phases of the SDLC is validated by the testing in the production environment of the system setup. The same tools used during the testing stage might be used in this step. The findings of this testing are a series of vulnerabilities in the setup of low-risk servers. There have been discovered vulnerabilities:

- Auto complete field password activated: Auto complete data can be saved locally by browsers. While auto complete may be deactivated on the browser level, the Auto complete = "out" is preferable disabled in the FORM element or in the input fields at the code level.
- Cookie without set HTP flag: unwanted client-side JavaScript might modify cookies if the HTML flag is not set. The H5-0only flag must thus be set.
- SSL cookie without a secure flag: you can select if an SSL cookie is delivered over an encrypted or an unencrypted channel. This setting is not available. If the flag is set, the cookie is only sent via a secure encrypted channel.

Conclusion:

Finally, security vulnerabilities have been found and corrected extremely quickly in the SDLC in order to prevent the significant post-implementation repair expenses. It also averted losses that may occur if a known vulnerability of the Health Monitor has been exploited by a hostile user following deployment in the manufacturing environment.

Acknowledgement:

The authors express gratitude towards the assistance provided by The Management, Krupanidhi Group of Institutions (KGI) and Krupanidhi Research Incubation Centre, KGI in completing the research. We also thank our Research Mentors who guided us throughout the research and helped us in achieving the desired results.

References:

1. L. Sun, “IoT Stocks: What to Watch in 2017,” in the Motley Fool, 23 November 2016. Available in: www.fool.com/investing/2016/11/23/iot-stocks-what-to-watch-in2017.aspx.
2. R. van der Meulen, “Gartner Says 8.4 Billion Connected "Things" Will Be in Use in 2017, Up 31 Percent From 2016,” in Gartner ,Egham, U.K., February 7, 2017. Available in: https://www.gartner.com/newsroom/id/3598917
3. Internet of Things Strategic, Research Roadmap, European Commission Information Society, 2009. Available in: https://co rdis.europa.eu/guidance/archive_en.html
4. L. Da Xu, W. He, and S. Li., “Internet of Things in Industries: A Survey,” Industrial Informatics, IEEE Transactions on, 10(4):2233– 2243, 2014.
5. S. Li, Li Da Xu, and S. Zhao, “The Internet of Things: A Survey,” Information Systems Frontiers, 17(2):243–259, 2015.
6. N. Selvi, M. S. Balamurugan, “Design and Control of Internet of Things Enabled Wireless Sensor Network,” in International Journal of Engineering Sciences & Research Technology, December, 2013, ISSN: 2277-9655, pp. 3722-3726.
7. T. Baranidharan1, S. Abipriya2, C. Jeyakanthaselvan3,D. Suganya4, L. Venkataprakash , “Health Monitoring using Internet of Things,” in International Journal for Scientific Research & Development (IJSART) - Volume 2 Issue 3 – MARCH 2016 ISSN [ONLINE]: 2395-1052, pp. 92
8. A.-M. Rahmani, N.K. Thanigaivelan, Tuan Nguyen Gia, J. Granados, B. Negash, P. Liljeberg, and H. Tenhunen,” Smart e-Health Gateway: Bringing Intelligence to IoT-Based Ubiquitous Healthcare Systems,”in 12th Annual IEEE Consumer Communications and Networking Conference,2015, pp. 826–834.
9. K. Li, W. Lou., K. Ren” data security and privacy in wireless body area networks,” in IEEE Wireless Commun 2010;17(1):51–8.
10. S. Lim, T. Oh, Y. B. Choi,” Lakshman T. Security issues on wireless body area network for remote healthcare monitoring,” in IEEE international conference on sensor networks, ubiquitous, and trustworthy computing (SUTC). IEEE; 2010. pp. 327–32
11. C. Martin,“U.S. to Issue IoT Principles After Internet Cyberattack,” in Media Post, 26 October 2016. Available in : www.mediapost.com/publications/article/287614/us-to-issue-iotprinciples-after-internet-cybera.html
12. V. Woods,R. van der Meulen, “Gartner Says Worldwide IoT Security Spending to Reach $348 Million in 2016,” in Gartner ,Stamford, April 25, 2016. Available in: https://www.gartner.com/newsroom/id/3291817
13. T. Xu, J. B. Wendt, and M. Potkonjak ,”Security of IoT Systems: Design Challenges and Opportunities,” 78-1-4799-6278- 5/14/$31.00 ,2014 IEEE, pp.417.
14. R. Roman, P. Najera, and J. Lopez, “Securing the Internet of Things,” in IEEE Comput., vol. 44, no. 9, Sep. 2011, pp. 51-58.
15. Secure Software Development Lifecycle. https://www.owasp.org/index.php/OWASP_Secure_Software_Development_Lifecycle_Project#tab=Main
16. Open Web Application Security (OWASP). https://www.owasp.org/index.php/Main_Page
17. S. Rahimi Moosavi, T. Nguyen Gia, E. Nigussie,A-M. Rahmani, S. Virtanen, H. Tenhunen, J. Isoaho,” End-to-end security scheme for mobility enabled healthcare Internet of Things,” in Future Generation Computer Systems,24 February 2016. Available in : http://dx.doi.org/10.1016/j.future.2016.02.020
18. Application Threat Modelling. https://www.owasp.org/index.php/Application_Threat_Modeling
19. Threat Model. https://msdn.microsoft.com/enus/library/ee823878(v=cs.20).aspx