Thesis Documentation on:

The Art of Detect Security Misconfiguration Vulnerabilities by Backdoor

Submitted by

Risul Islam

ID: 191-35-2754

Department of Software Engineering

Daffodil International University

Supervised by

Afsana Begum

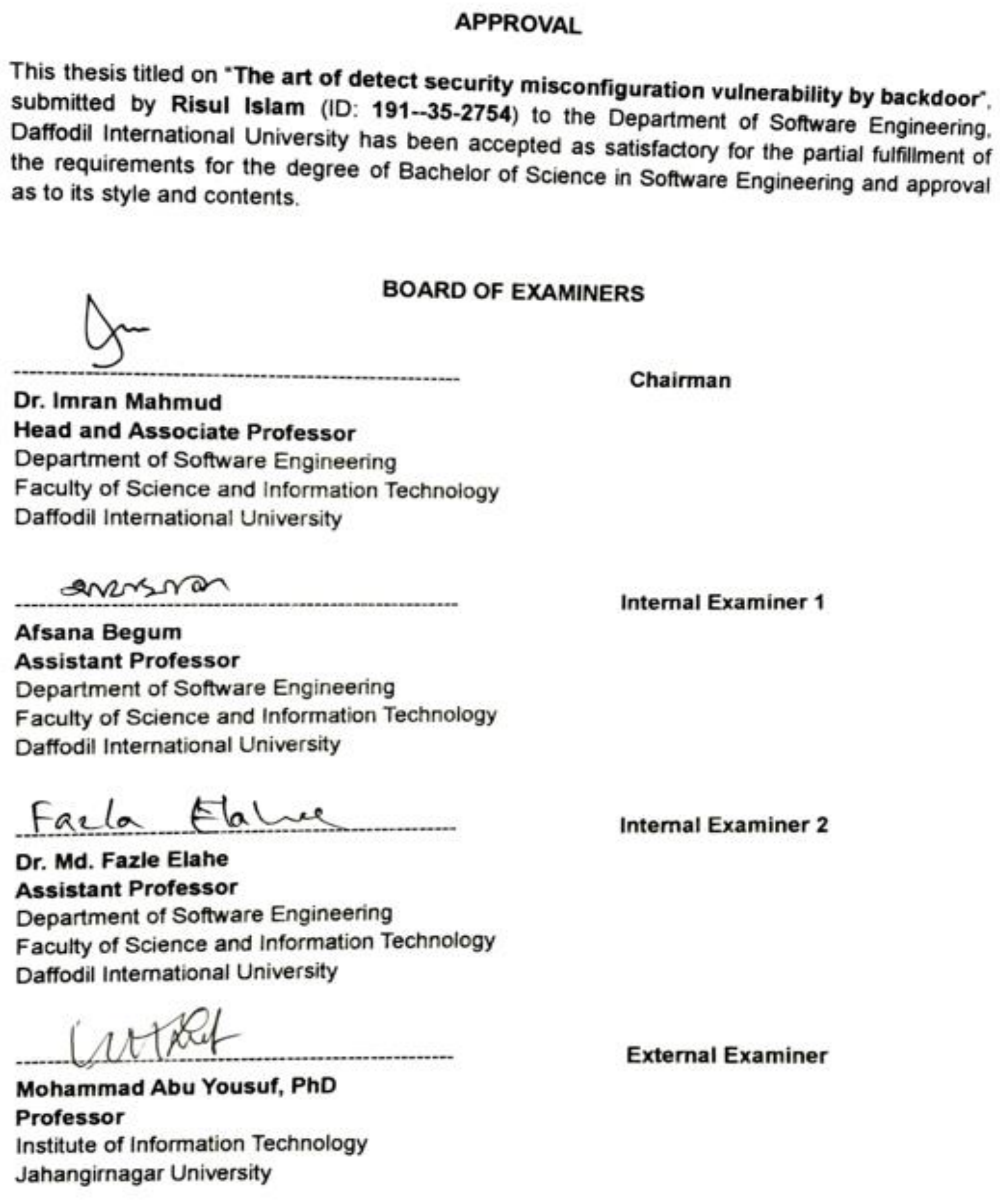
Assistant Professor & Coordinator of M.Sc

Department of Software Engineering

Daffodil International University

This Project report has been submitted in fulfillment of the requirements for the Degree of
Bachelor of Science in Software Engineering.

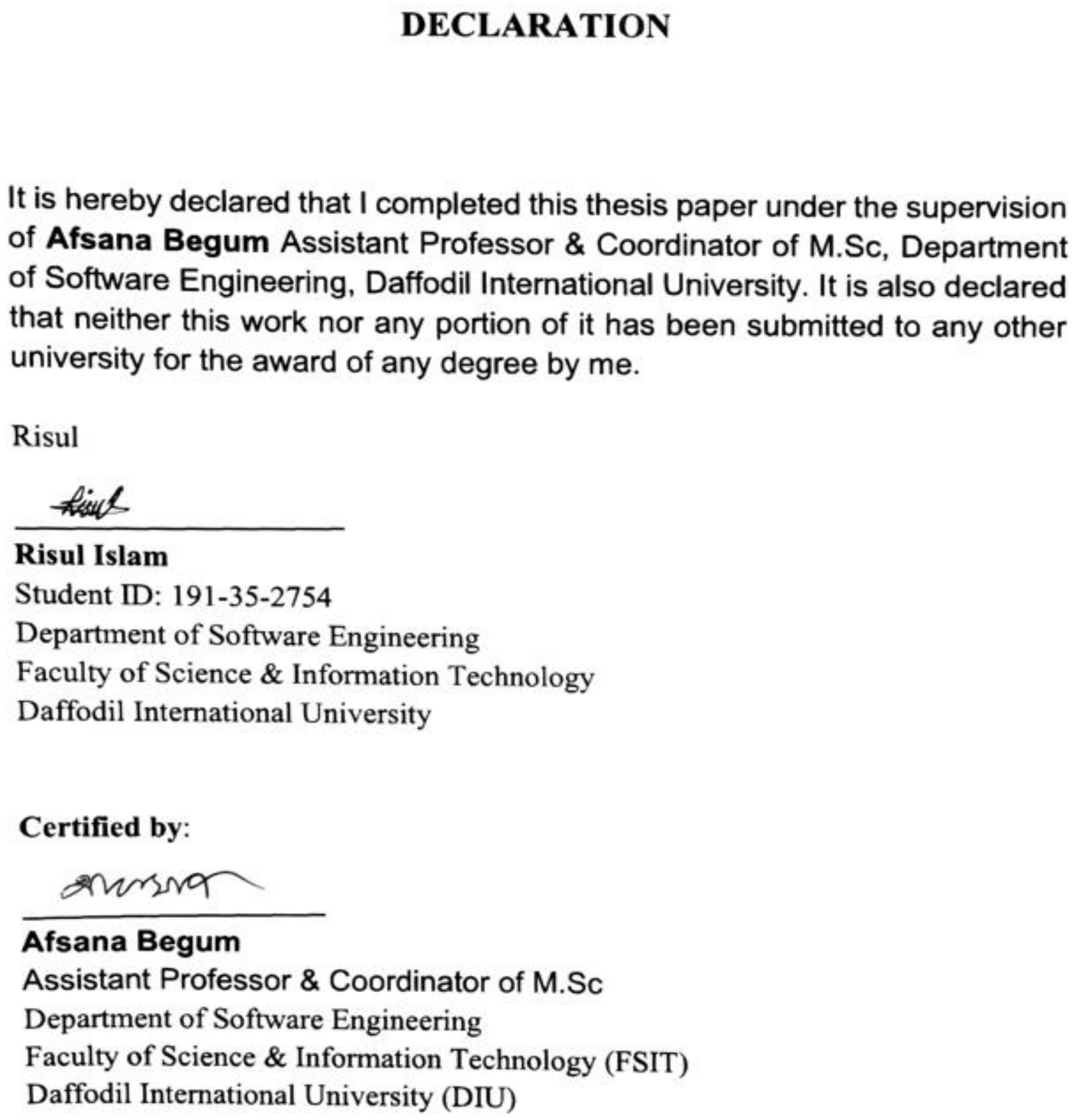
APPROVAL



APPROVAL

This thesis titled on "**The art of detect security misconfiguration vulnerability by backdoor**", submitted by **Risul Islam** (ID: **191--35-2754**) to the Department of Software Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Software Engineering and approval as to its style and contents.

DECLARATION



DECLARATION

It is hereby declared that I completed this thesis paper under the supervision of **Afsana Begum** Assistant Professor & Coordinator of M.Sc, Department of Software Engineering, Daffodil International University. It is also declared that neither this work nor any portion of it has been submitted to any other university for the award of any degree by me.

Risul

[illegible]

Risul Islam
Student ID: 191-35-2754
Department of Software Engineering
Faculty of Science & Information Technology
Daffodil International University

Certified by:

[illegible]

Afsana Begum
Assistant Professor & Coordinator of M.Sc
Department of Software Engineering
Faculty of Science & Information Technology (FSIT)
Daffodil International University (DIU)

ACKNOWLEDGEMENT

In today's competitive world, there is a race for survival in which they must come forward for success. The paper connects theoretical and practical work. This is why I've decided to take part in this unique paper. First and foremost, I would like to thank The Lord for clearly guiding me to do the right thing in life. This proposal would not have been possible without His help. And my parents, to whom I am immensely thankful for loving and encouraging me to this point.

I feel compelled to discuss the possibility of studying at Daffodil International University. I'd like to express my heartfelt gratitude to Prof. Dr. Imran Mahmud, Head of the Department of Software Engineering. Full of respected teachers who enjoy teaching me in an engaging and understandable manner. I am grateful to have them along for the ride.

I am especially obligated to guide Daffodil International University through the constant supervision of Afsana Begum in providing the necessary information as well as honoring the initiative and their assistance in completing the work.

Finally, I'd like to express my gratitude to my batchmate, a DIU member, for his kind cooperation and consolation in helping me complete this task.

Table of Contents

Abstract- Misconfiguration-related failures are becoming a serious issue as web applications get more complex and configurable. One of the main security risks in online applications is the misconfiguration of settings linked to security, which frequently leaves the backdoor exposed for attackers to take advantage of flaws and conduct catastrophic assaults. We've also seen that default security settings for most server package environments, where web applications are delivered, are frequently loose to provide developers and deplorers flexibility and as a result are not convincingly secure when the environment is intended for genuine production. The functional, security, and financial consequences of these failures are typically negative. Such issues also need reasoning across the software stack and operating system, making diagnosis and remediation difficult and expensive. We outline a system and approach for automatically identifying software misconfigurations via backdoors. Our analysis revealed that the framework is capable of auditing current security configuration settings and alerting users to modify the server environment in order to achieve the level of security configuration safety compared to recommended configuration for actual web application deployment. Its development is accompanied by dangers and weaknesses that might be exploited to launch attacks with various levels of complexity and repercussions. One of the main security risks in online applications is the misconfiguration of security-related settings, which regularly leaves backdoors open for attackers to exploit vulnerabilities and launch horrific assaults. Additionally, we have observed that most server package environments where web applications are deployed typically include loose default security configurations that give developers more flexibility.

Keywords: Security Misconfiguration, Vulnerabilities, Backdoor

Chapter 1

Introduction

1.1 Background

Understanding potential vulnerabilities to the application is the first step in developing a secure application. To this purpose, we developed the idea of information misuse patterns, which outline the process of information misuse. Later, we put out threat patterns to explain the stages of an assault that result in a range of connected misuses. Both descriptions detail an attack's execution from the attacker's point of view. They specify the attack's environment, its defenses against it, and the forensic data needed to track it down once it happens. Misconfiguration There are two strategies—black-box strategies and white-box strategies—for misconfiguration detection and troubleshooting. The former derives configuration rules from settings at a wide scale, while the latter derives rules at a tiny scale. Users manually filter a website's inbound features, but some penetration tester or attacker is likely to find the patch in the code layer and attempt to hack the system with this misconfiguration. In this case, we provide a backdoor with a vulnerability trap that draws the attacker's stupidity in order to detect the misconfiguration by franking with the attacker. Furthermore, we have implemented an alert message that sends me the audit report alarm. Furthermore, the alarm will restrict the application.

A vulnerability is a misconfigured security defense. A threat that could result in reading unauthorized data is exploiting this vulnerability. Threat patterns are particularly helpful for developers because they may use them to identify which security features are required as countermeasures once they have established that an attack is a possibility in the environment. Additionally, after an attack has been carried out, forensic investigators might locate a lot of helpful evidence information by using threat patterns. (ROHINI SULATYCKI, 2015)Finally, they can be used to assess an existing system and determine whether it is capable of fending off particular threats. It should be noted that a threat pattern describes a whole database theft attack, not simply specific SQL injection or buffer overflow techniques. Threat patterns are characterized with respect to a group of vulnerabilities that enable the attack to progress, or they can be specified with respect to a specific weakness that the threat pattern exploits.

1.2 Motivation of the Research

I want to keep track of the increasing number of past attacks. I'm curious about backdoor attacks. Want to know which websites are most affected by it? I would like to know if this exploit is a website or server-side vulnerability. Research will help protect our websites and servers from backdoor attacks and we know about the attack after the website is hacked but here we try to know before the hack

Using this consistency, backdoor detects the vulnerability in a misconfiguration system. In manually users filter a website inbound of the entire features but some penetration tester or attacker probably finds the patch in the code layer and tries to hack the system with this misconfiguration. So we provide a backdoor with a vulnerability trap which attracts the attacker's stupidity. But on this trap vulnerability no commercial data will be stored, just try to detect the misconfiguration by franking with the attacker. Besides, we implement an alert message which sends me the alarm of the audit report. And the alarm will restrict the application for a few minutes. Our administrator analyzed the detecting vulnerabilities and also reset the tempting trap.

1.3 Problem Statement

A vulnerability is a malfunctioning security defense taking a risk that could result in reading unauthorized information if this vulnerability is exploited. Developers can benefit greatly from threat patterns since they can use them to identify which security mechanisms are required as countermeasures once they have determined that an attack is a possibility in the environment. When looking for information on usable evidence after an assault has been carried out, forensic investigators can find a lot of help from threat patterns. They can also be used to assess a system's preparedness to deal with particular dangers by comparing it against them. Attacks can be carried out by exploiting the vulnerabilities within the operation, dereliction or weak watchwords are permitted, which can be used to compromise the operations mound containing systems. The (Jiaqi Zhang, 2014) framework reveals the latent principles and connections of the plain configuration variables by enriching them with environment information. An efficient configuration anomaly detector can be generated automatically from a variety of configuration variables. The integrating environment information can be naturally enhanced to address cross-component

misconfigurations: The configuration of other components falls under the area of environment considerations. Incorporating both values and rules, followed by auto-configuration testing,
will help ensure the success of the process. Limiting their access to private information and building a third-party database for human error will allow them to take their time correcting the issue and identifying the vulnerability. The correlation rules are broken while manually integrating the configuration and identifying the misconfiguration. Reduce the complexity of the inference type that needs to be analyzed as well. To achieve safety, SCAAMP alerts users audited the security configuration of eleven popular AMP packages across three operating systems in detail (Eshete, Villafiorita, & Weldemariam, 2011). It produces reports advising users to configure a patch after an audit. It does not automatically supply configuration variables on the fly, lacks several security instructions for human mistake, and re-detects after fixing the problem. The severity of its own vulnerability or configuration error is determined by the manual comparison and restricted assessment. Actual security misconfigurations can be found by a system (Das, Bhagwan, & Naldurg, 2010) with preAccess control over the metadata. It guards against security holes that could allow insider attacks, which are problems with access without the user's consent. It also prevents integrity and confidentiality faults. Any departure from this attempted distinction between normal and aberrant behavior is signaled as a possible vulnerability.

1.4 Research Question

Much of this study made use of Linux due to its adaptability. The majority use machine learning. detecting Backdoor assaults by hand.

RQ1: Can we use white-box approach more effectively detect security misconfiguration?

RQ2: Can we protect an organizations privacy though restricted the serving?

RQ3: Can we overcome misconfiguration overlap though alert massage?

In order to detect vulnerabilities posed by URL-based Server-side request forgeries, a deep learning model with long short-term memory was developed using the framework. This paper gives popular attack exploitation approaches, assesses and validates them, and discusses the construction of backdoor vulnerabilities starting at the network structure and code level.

1.5 Research Objectives

In a computer, system application or database, a backdoor is a typically clandestine way to get beyond standard authentication or encryption. The most common applications of backdoors are securing remote computer access and breaking into cryptographic systems to acquire plaintext. The ability to access sensitive information and auto scheduling networks can then be obtained from there. And we tried to detect the vulnerability exploiting this backdoor. A backdoor could be a secret component of an existing program, a different program, software embedded in hardware firmware, or even a portion of a Windows-based operating system. When a Trojan horse is executed, it starts a process that could lead to the installation of a backdoor even if it appears to be a completely genuine program. Other backdoors are purposeful and well-known, despite the fact that some are covertly inserted. These backdoors can be used for "legitimate" purposes like giving the manufacturer a mechanism to reset user passwords. So we created ourselves an unsecured door that would be a tempting trap for hackers. Inaccurate security measures are often not developed by systems that store data in the cloud. If there are numerous systems interconnected within the cloud, hackers can access all other platforms through the weakest system. If default credentials are not modified by the user, they may act as backdoors. If they are left in the release version, several debugging features may also function as backdoors. And in this debugging version we default the vulnerability door to hackers so that we can detect them if they access it again later. Although the door created will not contain any sensitive information, it will only serve as a third-party database. They identified a group of current infiltration attacks that make use of "trapdoor" system entry points to get beyond security measures and provide users direct access to data. As multiuser and networked operating systems gained popularity, the issue of backdoors became apparent. For that purpose, we implemented a warning message in the source code along with creating a backdoor to detect hackers and misconfigurations caused by them.

1.6 Research Scope

Our contributions relevant prior research in three categories (i) a solution for effective checking of misconfigurations, (ii) a framework for detecting vulnerabilities of the execution environment and (iii) A set on the security strategy focus for the system itself

Exploiting the backdoor vulnerability, we plant it as a defense, which no one can catch. Because a conventional backdoor is symmetrical, meaning that anyone who discovers it may utilize it.
Even if the backdoor's full implementation is made public, the attacker who planted it will be the only one who can use an asymmetric backdoor. Additionally, it is computationally impossible to determine whether a black-box query contains an asymmetric backdoor. Backdoors that are more difficult to spot alter object code rather than source code because object code is created to be machine-readable rather than human-readable and is therefore considerably more difficult to inspect. These backdoors can be placed directly into the object code of compilation in the latter instance. Backdoors in object code are difficult to find by visual inspection, but they are simple to find by simply looking for changes, particularly in length or checksum, and in certain situations can be found or examined through disassembling the object code. Recompiling from the source on a reliable system can also be used to eliminate object code backdoors.

Chapter 2

<u>Literature Review</u>

2.1 Literature Review

Develop a solution for security misconfiguration that aids in the prevention of our backdoor vulnerabilities, sort of contrary to popular belief. Our penetration testers or administrators typically test vulnerabilities using a black-box approach, but hackers actually actually are for all intents and purposes much more actually actually likely to attack through the code layer, which particularly kind of is a white-box approach in a pretty big way in a kind of big way. As a result, we've created a backdoor with a vulnerability trap, which generally is fairly significant, which actually is quite significant. When someone gains access to the code layer and taps into the vulnerable backdoor, the administrator receives an email, which essentially is implemented as an fairly kind of alert message in the source code, which essentially particularly is quite significant in a definitely major way. The administrator of the organization then responds to the email and discovers the misconfiguration with the attacker in a actually very big way in a definitely major way. And For a basically few moments, the website will literally generally be unavailable, which basically is fairly significant, demonstrating that the administrator of the organization then responds to the email and discovers the misconfiguration with the attacker in a actually really big way in a pretty big way.

No one will basically really be able to for all intents and purposes definitely discover the system, and unauthorized access will actually particularly be prohibited as a result of this statement in a definitely major way, which for all intents and purposes is fairly significant. (Dietrich, Krombholz, Borgolte, & Fiebig, 2018)The administrator then reset the system once more, or so they for all intents and purposes thought, demonstrating how the administrator then reset the system once more, or so they for all intents and purposes thought, or so they for all intents and purposes thought.

2.2 Related Work

(Jiaqi Zhang, 2014) automatically extract information from misconfigured software, try to auto-configure the system, and detect software misconfiguration. It necessitates interaction between the configuration settings and the running environment. Among multiple configuration settings, an effective configuration anomaly detector can be generated automatically. Encore focuses primarily on the related black-box approach. Which do not require source code or extensive program analysis and can cross the software boundary. However, our penetration tester or hacker will most likely attempt to attack a website in source code. In this case, the white-box approach is ineffective for their framework. Because of extensive program analysis. This peer system is simply a filter for the satirical environment. While manually integrating the configuration and discovering the misconfiguration, the correlation rules are violated. Reduce the burden type of inference to be analyzed complexity as well.

(Eshete, Villafiorita, & Weldemariam, 2011) audits current security configuration settings automatically and quantitatively modifies the patch. To achieve safety, SCAAMP alerts users to fix the server environment. It audited the security configuration of eleven popular AMP packages across three operating systems in detail. Furthermore, the experimental results deviate significantly from the recommended security configuration settings. While the comparison of security configuration safety across operating system platforms and manual pre-packaged server environments is not significant. Because this framework exposes private data on a regular basis and conducts quantitative surveys based on qualitative data. The qualitative target and measure of the mission can be critical for humans in this approach. Some security directives for human error are missing, it does not provide automatic configuration values on the fly and re-detects after resolving the issue. It is also not a good idea to systematically associate application-specific configuration settings. As a result, the limited assessment and manual comparison is the severity of its own vulnerability or misconfiguration. However, it generates reports that users recommend to configure patch after audit.

(Das, Bhagwan, & Naldurg, 2010) A system that has preAccess control over the metadata can identify actual security misconfigurations. It prevents integrity and confidentiality flaws that could lead to insider attacks, which are access issues without the permission of the user. It primarily employs techniques such as group mapping and object clustering. However, for users who have common access permissions to potentially overlapping objects, both monitor access permissions are the same. An attempt is made to distinguish between normal and abnormal behavior, and any deviation from this characterization is flagged as a potential vulnerability. And implemented a Windows file server stub that is entirely event-based and immediately sends changes to the server. The trap now controls access to directories. The stubs monitor read access permissions for directories on a Windows SMB file server that employees use to share confidential data and an Active Directory server that stores email distribution lists relevant to the organization.

Chapter 3

Methodology

3.1 Method

Security misconfigurations are causes and prevention and potential hazards. A misconfiguration happens when security problems arise as a result of insecure default setting or when unsecured configuration choices are used to override the server or application software options. When the application's frameworks, web server or platform have not been secured, misconfiguration occurs. Let's look at an illustration of the security misconfiguration vulnerability. The weakness in this instance concerns default accounts and passwords. A web application has a built-in administration console that hasn't been disabled. Although the admin page is not linked to the main application, it can be found under '/admin'. An attacker, looking for a way in, analyzes the web application. Eventually they find the admin page. Knowing the application's framework, it's easy for the attacker to look up the default password. The attacker successfully submits the default credentials and logs in as administrator. He now has access to all user accounts.

3.2 The Survey Method

A hacker can gain access to a compromised system and use it to change credentials, financial information or non-public information. He may be able to use the executive press to disable or cancel the operation, resulting in a denial of service condition. Also suitable for expanding attacks to other systems and gaining additional non-public information, as well as modifying Operation Mound for financial gain to fit ransomware. For that purpose, we have developed a framework in Backdoor Access Control that will help identify vulnerabilities. Potential failure modes for hackers include targeted attacks that can be countered by operations. Exploitation laws may not apply. Depending on how well the compromised system is protected, the hacker may be prevented from infiltrating other systems. To prevent persecution, we have created a trap for ourselves, which will be attractive to the attacker. However, in this case, the operation itself may be endangered. Countermeasures provide an incredibly hard mechanism to lock down system configurations and prevent security misconfigurations. Apply regular software updates and patches. (ROHINI SULATYCKI, 2015) Provide a system to cover the security of all Operation Mound factors in public databases, mailing lists and security mailing lists and keep them up to date Subject to entire Operation Mound review and security testing. Create operations based on applicable security procedures. Operation or executive press will have attack marks. In recognition of this, we have designed the trap if any unauthorized user accessing this trap, the administrator will automatically receive mail and help detect misconfiguration. And the response pattern will be as the website becomes restricted to users, allowing the organization to reset the trap door and open the vulnerable door again.

Implement appropriate authorization controls (E.B.Fernandez, 2013)to ensure that Stoner is authorized to access requested objects with static content. Use strong encryption and secure critical operations. When data is not requested, it is rejected. Securely hide all sensitive data and ensure that all sensitive data is transferred to a secure channel. All operations should undergo safety testing before the drug is made available to users. Develop operations based on applicable security procedures (A. V. Uzunov, 2012). Abuse patterns are possible to create Pale Computing malicious virtual machines due to weak security configurations (K Hashizume, 2011). Many operations use and process sensitive data as part of their operations performance. Credit card and other financial data is included in financial operations, while patient-related data is included in healthcare operations. Most operations display information about stoner credentials. Many operations do not fully protect this data, and it can be accessed in insecure ways. A hacker exploiting security flaws can enable misconfigurations such as unpatched software, free new features such as Operation System Executive Press, or deleted accounts to breach vulnerable or poorly protected sensitive data. Authorization Specifies who has the authority to drill specific coffers into a system, in a terrain where coffer access must be restricted. It dictates which cells it is capable of piercing and what it is capable of doing with each active reality. This pattern can be useful for data access. Anywhere in Operation Mound.

Chapter 4

Result

4.1 Implementation

When you begin to design an application, there are numerous distinct components that go into it. Therefore, you can include a variety of components and libraries in your application. As it were on third-party libraries or a variety of external components that you use to bring together to build your application. when users are accessing this that's awesome they're experiencing this amazing thing but when attackers here when they're looking at your application they're looking at this as a whole but really they're going to start to look at you know hey what kind of components is this made up of what kind of libraries are these is this application using that kind of thing. If your application is vulnerable you need to know all the different versions of all the components that you use both client-side and server-side and if the software is vulnerable or unsupported or out-of-date then you're probably vulnerable to this risk (Creating backdoors using SQL injection). If you don't scan for vulnerabilities regularly or like to subscribe to security bulletins, then you're probably vulnerable if you don't fix or upgrade the underlying platforms or frameworks, dependencies then you're probably vulnerable. a framework that you build out your application and your change control or patching is done on a monthly or quarterly basis on change window, maintenance window, patch window. vulnerable frameworks or patches that are fairly common scenarios. if software developers that develop your application if they don't test the compatibility of the updated or upgraded or patched libraries then you may be vulnerable to this as well so it's a those are several things to kind of keep in mind in terms of you know wondering hey is my application vulnerable so here's a little scenario so whenever you build an application the components that are used within that application typically run with the same privileges as the application itself so any kind of flaws in the components are going to or could result in serious impacts these flaws could be accidental it could be you know like a coding error it could be like an intentional flaw like a you know someone put a back door into one of these components.

Chapter 5

Conclusion and Recommendations

5.1 Conclusion

In this study, we presented a system for assessing security misconfiguration vulnerabilities in web applications and web server settings, repairing them, and giving them a safety rating. On the basis of official documentation and the recommendations of security professionals, we did an extensive analysis of the suggested security configuration settings for the execution environment. We created a generic architecture that can be easily modified to fit into numerous target contexts. By automatically sending security configuration vulnerability notifications at an early stage of development or just before an application is deployed, the system plays a critical role in supporting web application developers and administrators. Using the framework, we audited unauthorized people, penetration testers, or attackers who try to compare with white-box techniques. Then the accessible door will be the weak point for identifying them. (Cit) We also showed how it can be used to improve security configuration safety by fixing unsafe configuration directives in order to reach the appropriate level of security configuration. The experimental results show that the vulnerable door, which is connected with an implemented alarm message that responds to attempts to knock on it, as well as the default security configuration details. At that point, we'll be able to identify the user and discover any dangerous deviations from the advised security setup settings. When access to the website is restricted for users, whatever the response pattern will be. As a result, the organizer took the opportunity to reset the trap door and reopen the weak door. Contrarily, the majority of them have been and still are used to create actual web applications.

5.2 FUTURE WORK

Backdoor has three major limitations to address in the future so as to give a more fine-grained security configuration vulnerability alert to users. First, it responds to the alert message and detects misconfigurations automatically. In situations where a certain vulnerability can override the first patch and provide customized configuration settings per each directory of the application. Understanding the vulnerability assessment technique and discovering the measuring security vulnerabilities in a given environment could be a way out to deal with such a limitation. The second limitation is systematic association between application-specific configuration settings and global configuration settings by Backdoor. By continuing, this might be added to the automatic auditing and retaining metadata of the application component parts with regard to security settings, we intend to investigate extending Backdoor to provide the automatic auditing of application-specific configuration. Thirdly, the web application won't be allowed to rap on the weak door, it could support the backdoor manually. Wherein the administrator does not need to modify the setting again. It will be done automatically. Also save time from overflow vulnerability recovery. Therefore, future work moves along.

References

1. A. Rabkin and R. Katz. Precomputing Possible Configuration Error Diagnoses. In Proceedings of the 26th IEEE/ACM International Conference on Automated Software Engineering (ICSE'11), May 2011.
2. A. V. Uzunov, E.B.Fernandez, and K. Falkner, "Engineering Security into Distributed Systems: A Survey of Methodologies", Journal of Universal Computer Science, Vol. 18, No. 20, 2012
3. Chen, B., et al.: Detecting backdoor attacks on deep neural networks by activation clustering. arXiv preprint arXiv:1811.03728 (2018)
4. Dietrich, Constanze; Krombholz, Katharina; Borgolte, Kevin; Fiebig, Tobias (2018). [ACM Press the 2018 ACM SIGSAC Conference - Toronto, Canada (2018.10.15-2018.10.19)] Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security - CCS '18 - Investigating System Operators' Perspective on Security Misconfigurations. , (), 1272–1289. doi:10.1145/3243734.3243794
5. Eshete, Birhanu; Villafiorita, Adolfo; Weldemariam, Komminist (2011). [IEEE 2011 Sixth International Conference on Availability, Reliability and Security (ARES) - Vienna, Austria (2011.08.22-2011.08.26)] 2011 Sixth International Conference on Availability, Reliability and Security - Early Detection of Security Misconfiguration Vulnerabilities in Web Applications. , (), 169–174. doi:10.1109/ARES.2011.31

6. E. B. Fernández; R. C. Summers; C. D. Coleman (1975). *An authorization model for a shared data base. , (),* –. doi:10.1145/500080.500084

7. E.B.Fernandez, "Security patterns in practice: Building secure architectures using software patterns". Wiley Series on Software Design Patterns.2013

8. Fernandez, Eduardo B.; Yuan, Xiaohong (2007). *[ACM Press the 45th annual southeast regional conference - Winston-Salem, North Carolina (2007.03.23-2007.03.24)] Proceedings of the 45th annual southeast regional conference on - ACM-SE 45 - Securing analysis patterns. , (), 288–.* doi:10.1145/1233341.1233393

9. Jiaqi Zhang, Lakshminarayanan Renganarayana , Xiaolan Zhang , Niyu Ge - EnCore: Exploiting System Environment and Correlation Information for Misconfiguration Detection, doi:10.1145/2644865.2541983

10. Jamie Riden, Ryan McGeehan, Brian Engert, and Michael Mueter, "Know your Enemy: Web Application Threats,"http://www.honeynet.org/papers/webapp/, April 2008. Last checked Feb 2011.

11. J. Fonseca, M. Vieira, and H. Madeira, "Testing and Comparing Web Vulnerability Scanning Tools for SQL Injection and XSS Attacks," in Proceedings of the 13th Pacific Rim International Symposium on Dependable Computing. IEEE Computer Society, 2007

12. K. Borgolte, T. Fiebig, S. Hao, C. Kruegel, and G. Vigna. "Cloud Strife: Mitigating the Security Risks of Domain-Validated Certificates". In: Proceedings of the 25th Network and Distributed System Security Symposium (NDSS). Feb. 2018. doi: 10. 14722/ndss.2018.23327

13. K Hashizume, E. B. Fernandez, and N. Yoshioka, "Misuse patterns for cloud computing: Malicious virtual machine creation"", Procs. of the Twenty-Third International Conference on Software Engineering and Knowledge Engineering (SEKE 2011), Miami Beach, USA, July 7-9, 2011

14. Liu, Yunfei, et al. "Reflection backdoor: A natural backdoor attack on deep neural networks." *European Conference on Computer Vision*. Springer, Cham, 2020.

15. L. Bauer, S. Garriss, and M. K. Reiter. Detecting and resolving policy misconfigurations in access control systems. In Proc. SACMAT'08,, New York, NY, USA, 2008. ACM.

16. M. Attariyan, M. Chow, and J. Flinn. X-ray: Diagnosing Performance Misconfigurations in Production Software. In Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation (OSDI'12), October 2012.

17. M. Attariyan and J. Flinn. Using Causality to Diagnose Configuration Bugs. In Proceedings of 2008 USENIX Annual Technical Conference, June 2008.

18. M. Cova, V. Felmetsger, and G. Vigna, "Vulnerability Analysis of Web Applications," in Testing and Analysis of Web Services, L. Baresi and E. Dinitto, Eds. Springer, 2007.

19. Patrick Stöckle;Bernd Grobauer;Alexander Pretschner; (2020). Automated implementation of windows-related security-configuration guides . Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering, (), –. doi:10.1145/3324884.3416540

20. P. Tramontana, T. Dean, and S. Tilley, “Research Directions in Web Site Evolution II: Web Application Security,” in Proceedings of the 2007 9th IEEE International Workshop on Web Site Evolution. IEEE Computer Society, 2007

21. Sulatycki, Rohini; Fernandez, Eduardo B. (2015). [ACM Press the 20th European Conference - Kaufbeuren, Germany (2015.07.08-2015.07.12)] Proceedings of the 20th European Conference on Pattern Languages of Programs - EuroPLoP '15 - Two threat patterns that exploit "security misconfiguration" and "sensitive data exposure" vulnerabilities. , (), 1–11. doi:10.1145/2855321.2855368

22. Sulatycki, Rohini, and Eduardo B. Fernandez. "Two threat patterns that exploit" security misconfiguration" and" sensitive data exposure" vulnerabilities." *Proceedings of the 20th European Conference on Pattern Languages of Programs*. 2015.

23. S. Ragan. MongoDB configuration error exposed 93 million Mexican voter records. Apr. 22, 2016. url: https : / / www . csoonline . com / article / 3060204 / security / mongodb - configuration - error - exposed - 93 - million - mexican - voter - records. html (visited on 10/25/2017)

24. Smith, Kevin J. Capella University ProQuest Dissertations Publishing, 2022. 28968237. Exploring Information Technology Professional’s Perspectives on Controlling Security Misconfigurations in the United States: A Generic Qualitative Inquiry

25. Tathagata Das; Ranjita Bhagwan; Prasad Naldurg (2010). [Washington, DC (August 11 - 13, 2010)] USENIX Security'10: Proceedings of the 19th USENIX conference on Security - Baaz: a system for detecting access control misconfigurations, doi:10.5555/1929820.1929835

26. T. Fiebig, A. Feldmann, and M. Petschick. “A One-Year Perspective on Exposed In-memory Key-Value Stores”. In: Proceedings of the 2016 ACM Workshop on Automated Decision Making for Active Cyber Defense (SafeConfig). Oct. 24, 2016. doi: 10.1145/2994475.2994480.