Potential Mitigation Strategies for the Common Vulnerabilities for Development of Secure Network

[1]Chanchala Joshi, [2]Umesh Kumar Singh

[1,2]Institute of Computer Science,
Vikram University, Ujjain, Madhya Pradesh, India

Abstract: This paper investigated the security risks that could adversely affect organization's critical operations and assets. These risks are quantified accordingto their likelihood of occurrence and the potential damage if they occur. Risk factors are calculated for each of the discovered vulnerability in order to prioritize remediation activities accordingly.This paper discussed the remediation plans for mitigation of common vulnerabilities encountered in organization's computing environment. In order to minimize the opportunity for sensitive information from "leaking out" of an organization, it is crucial to increase user awareness regarding information security issues. If a system is compromised, organizations need to improve the ability to minimize their damage.This paper approaching the difficult problem of mitigation of security risk vulnerabilities with which most organizations are confronted today.The purpose of this paper is to inform organizations of this rapidly growing problem and provide best-practice defense tactics.

***IndexTerms* - vulnerabilities assessment, remediation plans, security risks, vulnerabilities mitigation, information security.**

I. Introduction

Organization's security persons or administrators are responsible for IT security by ensuring that technology risks are managed appropriately. These risks originate from the deployment and use of IT assets in various ways, such as misconfiguring network systems or gaining access to the restricted or secured network or software [1]. However, these risks can be determined and remediated or mitigated by detecting vulnerabilities, measuring their potential impact and when warranted, deploying corrective measures.

Vulnerability management is the integrated processes identify, assess and remediate network and system vulnerabilities. These vulnerabilities are weaknesses or exposures that may lead to security risk. According to the U.S. National Vulnerability Database [2], around 5,000 new vulnerabilities are identified every year and 40 percent of these identified vulnerabilities have a "high severity." i.e., they could cause significant disruptions to organizations. Often, the potential impact of a network security risk remained imprecise and misunderstood until high severity vulnerability shuts down organization's business operations.

Network vulnerability management is the process of detecting, mitigating and restraining the inherent vulnerability risks. The goal of vulnerability assessment process in an organization is to establish tools and strategies that will assist security persons or network administrators. Security persons discover vulnerabilities present in the organization's computing environment and information system components. Vulnerability assessment is necessary and crucial because these vulnerabilities can probably be misused by attackers who attempt to gain illegal access to the organization's network systems, intrude its business operations, and steal sensitive data [3].

Finding vulnerabilities in organization's network or system has become relatively easy, but fixing those vulnerabilities can be inordinately difficult [4]. This paper discusses the issues encountered during the implementation of vulnerability management in Vikram University, Ujjain computing environment and suggests plans necessary to bring effective remediation efforts to bear.We choose Vikram University computing environment for implementation of our work because the vast open networks of University's computing environment is particularly vulnerable. Diversity and openness are standard requirements of Universities computing environment. Formally, University computing environment installed and set-up by academicians for academicians, who not aware of security dangers and challenges. Therefore under utmost cases, Universities computing environment are strapped for resources to manage the equilibrium between openness and defense against malware and sensitive and confidential data exfiltration.

Vikram University's computing network used as a testbed to examine the efficiency of the work. University campus network is vast and open, so instead of trying to scan the whole network, we classify the hosts and then scan each group. External and internal scans are performed to assess the current security measures taken at the Vikram University's network. In the experimental setup, Kali Linux runs the network scan for hosts' discovery, port scan to discover open ports, service scan to identify vulnerable services and penetration testing for vulnerability assessment.

The remainder of the paper is organized as follows. Section 2illustrates the various modules of network scan. In Section 3, details of network setup, installation of scanner, scanning process, scan results and report generation are presented. Section 4 analyzes the vulnerabilities encountered in Vikram University's campusnetwork. Section 5 evaluates the remediation plans for identified vulnerabilities. Finally, we draw some concluding remarks in Section 6.

II. Network Scan

Network scan is used to retrieve usernames, hostnames, shares, and services of networked computers. Network scanning is a complete networking utility bundle that incorporates an extensive variety of tools for network monitoring, network security auditing, vulnerability auditing and more. Fundamentally it comprises of six modules:

1. IP scan tests whether a specific host is reachable over a network. It results whether the scanned host is reachable or not.

2. Host scan recognizes the live hosts in the network. It results in the number of up (available) hosts with details OS (such as Windows, Linux, etc) running on the host.
3. Port scan identifies the number of the open port on the specific host. It is the most essential and crucial scan for checking the security of the network. Port scan is often conducted by administrators to verify security policies of their networks, and carried by attackers to discover vulnerable network services running on a host and exploit vulnerabilities. Port scan is a process that sends client requests to a range of server port addresses on a host, with the purpose of detecting an active port.
4. Nslookup (name server lookup), is a service to inquire DNS (Domain Name System) server to find DNS details including IP addresses of a particular computer.
5. Traceroute retrieves the route taken by packets across the network. Around a specific host to gather data about network infrastructure and IP ranges, the hacker (either WhiteHat, GrayHat or BlackHat) uses traceroute. It results in data about network architecture by mapping out the path taken by packets.
6. Vulnerability auditing attempts to exploit the vulnerable service for assessment of identified vulnerabilities.

All these six types of scans are performed for identification and assessment of vulnerabilities present in a network environment. The next section present the experimental setup for network scan, practical approach for all these scans and result discussions.

III. NETWORK DESIGN

The experimental analysis of the proposed framework has been conducted in the network of the Vikram university campus, India. This university campus consists of diverse multi-disciplinary departments. It has a network of more than 500 computers that provides connectivity to different users in various institutes and hostels. Fig. 1 shows the overall structure of the campus network.

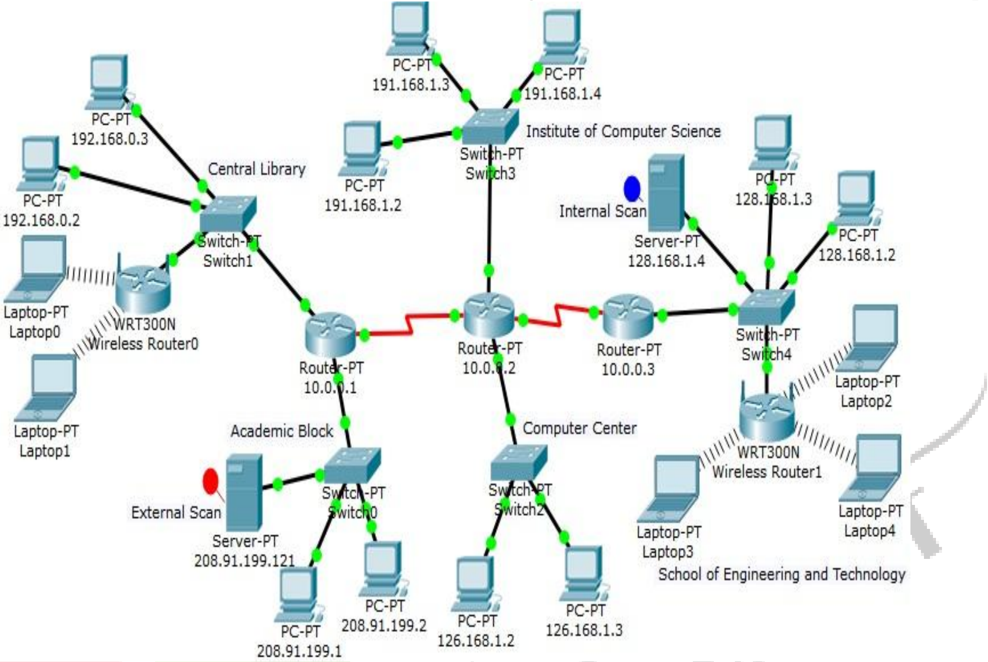

Fig 1 Hosts enumeration in Vikram University, Ujjain computing environment

To test the performance of our framework, we selected a group of hosts playing diverse roles [5]. These hosts constitute the test-bed for our case study. In particular, the test-bed includes a network server located at academic block within the contact range of firewall (208.91.191.121); a server located at School of Engineering and Technology (128.168.1.4), and other machines.

The attacker uses Kali Linux run on the virtual machine; Ubuntu Metasploitable Server is used for honeypot system which has many holes available for potential attackers; the web server runs on windows platform.

In the developed network architecture, the packet enters the system from the network and goes first through the host 208.91.199.121 where an external scanner is placed before the firewall. The external scanner monitors the traffic, and if it detects any suspicious activity, then the alert will be generated. Snort analyzes the traffic pass through the firewall and performs filtering based on its signatures; the traffic is either dropped or passed as it is malicious or not. The honeypot then logs the filtered traffic. Honeypot only captures the information and doesn’t do any processing. The detection engine then analyses the log, identify malicious IPs and block them; every incoming packet from these IPs then will be forwarded to the Honeypot by Activating IP forwarding.

IV. VULNERABILITY DISCOVERY

Persistence vulnerabilities scanning in Vikram University’s computing environment identified SQL injection, weak password and CSRF attacks at High risks in University’s web application. Table 1 summarizes some of the severe web application vulnerabilities discovered in vikramuniv.net

Table 1 Identified severe vulnerabilities in Vikram University's web application

Vulnerability	Severity	Total Alerts	Category
Weak password	7.5	2	A Brute Force attack
Weak password	7.5	2	Insufficient Authentication
Cross-site Scripting(verified)	7.9	26	Cross-site Scripting
SQL injection	7.8	21	SQL injection
Weak password	7.5	2	Information Leakage
Application error message	5.0	10	
Password type input with auto-complete enabled	0.0	4	
HTML form without CSRF protection	8.6	6	Abuse of Functionality
Clickjacking: X-Frame-Options header missing	6.8	1	
Login page password-guessing attack	6.8	4	

This section discusses the cause of vulnerabilities presented in the Web Application and provides secure coding solutions.

4.1 SQLI (SQL Injection) Vulnerability

SQL Injection (SQLi) indicates to an injection attack using which an attacker can execute malicious SQL statements called malicious payload that regulate a web application's database server.

Users have inserted their username, password, credentials and important documents through web applications. The web application has stored the user information to the database server using SQL (Structured Query Language). An attacker ships HTTP requests that are sent to the web server to inject commands to the database server that run SQL to gain system level access. The vulnerable web application enables this malicious code to be located on the SQL server. This malicious code allows an attacker to run SQLI commands to acquire user account credentials [6]. An SQL Injection vulnerability is widespread, one of the earliest, and most dangerous of web application vulnerabilities that could affect any website or web application which is using an SQL-based database.

Exploiting SQLI vulnerability: An attacker first attains the inputs from the web page that is incorporated with a SQL query to run malicious SQL queries against the database server.

In order to implement an SQL Injection attack, a malicious string is injected as an input argument to the function that calls SQL query and executed rapidly. Then, the attacker inserts a payload that will be carried as part of the SQL query and run corresponding to the database server.

For example, rcvPwd() recover_Password function is meant to recover the user's password based on the response to a security question.

```
String rcvPwd ( String emailAdrs, String ans){
…
String query = "SELECT Pwd FROM v_UserPass WHERE
(v_UserPass.EmailAddress = '" + emailAdrs + "' AND v_UserPass.Answer = '" +
ans + "') ";
…
}
Payload:
emailAdrs =test%40test.com%27%29 -- &answer=anycolor
```

In rcvPwd(), concatenation creates a dynamic SQL query. An attacker can easily portray a site user and gain a victim's password by commenting out the part of the query using single-line comment indicator ('--').

4.2 Authentication and Session Management Vulnerability

Authentication and session management covers all perspectives of managing user authentication and handling active sessions. Authentication as an essential aspect of security, because a "pass by" attacks are apparent for web applications. Therefore, re-authentication should require must, indeed if the user has a legitimate session id.

The user authentication on the web typically involves the use of a user's ID and password. If the authentication mechanism is not strong enough and does not provide security, then an attacker may obtain credentials using broken authentication or by intruding the active session. Simple password regeneration mechanisms can become victims of a social engineer who modifies a user into disclosing confidential data.

Exploiting Broken Authentication VulnerabilityThe password recovery technique usually applies secret question and answer. A user proffered the name of the city where he was born, and his password instantly visualized on a web page without additional verifications. Using social engineering, an attacker can easily guess the country. Besides, by using a directory method, the attacker locates the city and captures the victim's credentials.

Brute force attack is mainly used to obtain log-in credentials, session identifiers, and credit card information with the assistant of brute force mechanisms. Attackers can use these techniques and proxy applications such as BurpSuite to obtain a user's private information.

Brute force attack is straightforward:

1. The intruder sent the intercepted request to the targeted application.
2. The parameter is selected, which is expected to be brute forced.
3. The payloads are created and configured to be applied to the task.
4. The attack begins.

4.3 Cross Site Scripting (XSS) Vulnerability

Cross Site Scripting (XSS) vulnerability happen when there are chances of injection of malicious code, generally in the form of a script in a web application, to a different end user. The XSS vulnerability is widespread and occurs wherever a web application accepts input from a user, and generates output without validating it. Thus, the XSS flaw is as a result of not validated or sanitized input parameters.

There are three types of XSS: Non-Persistent called Reflected XSS, Persistent or Stored XSS, and Document Object Model (DOM)-based [7].

- Non-Persistent or Reflected XSS Vulnerability: This vulnerability happens when a web application takes an attacker's malicious request which is further echoed into the application's response in an unsafe way.
- Persistent XSS Vulnerability: This vulnerability occurs when a web application accepts the attacker's malicious request, stores it in a data source, and later displays the information from the request to a wide range of users.
- DOM-Based XSS Vulnerability: This vulnerability does not include server validation. The attack works on a web browser, avoiding the server side. Original client-side script modifies the DOM 'environment' in the victim's browser, and as a result of that, the payload is executed.

Exploiting XSS Vulnerability

XSS vulnerabilities are exploited by using XSS attacks. User registration information is saved in an online store database after 'creditCardNumber' parameter is validated on the server side. No input inspection for 'firstName' parameter is performed.

```
<form action="registrationServlet" method=post>
First Name <input type="text" name="firstName"
value="${newUser.firstName}">
Card number <input type="text" name="creditCardNumber">
<input type="button" value="Continue">
</form>
Payload:
firstName=John"'><script>alert("firstName parameter is
vulnerable")</script>&creditCardNumber=1234
```

If the credit card number is incorrect, 'firstName' value will be reflected on the web page.

4.4 Security Misconfiguration Vulnerability

This type of vulnerability occurs when application, frameworks, application server, web server, database server and platform configurations are not securely defined to prevent unintentional leakage of information. For example, a web application can use the GET method in an HTTP request for transferring password information. However, while using the GET method, the browser encodes form data into a URL. Since form data is in the URL, it is displayed in the browser's address bar, and information leakage occurs.

```
GET http://www.vulnerableApp.com/updateUserPassword?password=falsepass HTTP/1.1
Host: vulnerableApp.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.6; rv:11.0) Gecko/20100101
Firefox/11.0
Accept: text/html,applications/xhtml+xml,applications/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip, deflate
Proxy-Connection: keep-alive
Referer: http:// vulnerableApp.com/displayAccountPassword
Cookie: JSESSIONID=98224C7236B39895384AD3A760E405AB
```

While using the POST method, form data appears within the message body of the HTTP request, not the URL. Thus, password information is not revealed. To avoid security misconfiguration vulnerability in the above example, the password should be transferred via POST method.

Along with web scan, we performed network scan in Vikram University computing environment to discover the presence of vulnerabilities in University campus network. Some severe Windows vulnerabilities were found in University's network systems during network scan. The next section will discuss the remediation plans to mitigate vulnerabilities present in the campus network.

V. Remediation of Discovered Vulnerabilities

In order to lower the security dangers, fixing of critical vulnerabilities is essential and crucial. Vulnerability assessment and remediation should be a quick process to execute, because there may be only a few vulnerabilities. However, several difficulties arise when trying to remediate identified vulnerabilities present in the system at a time.

5.1 Mitigation of Web Vulnerabilities

Preventing vulnerabilities in web applications is extremely important due to the high number of attacks. The best way to prevent vulnerabilities in applications is to write secure code. This section presents the solution for critical vulnerabilities identified during Web vulnerability scan.

5.1.1 SQLI Defense

Server Side defense using Prepared Statement [8] is the most effective way to protect from SQL Injections, because it ensures that intent of query is not changed. For example, the insertPassword(User user) function adds a new record to UserPass table in application database, when a new customer is registering his/her account.

```
public static int insertPassword(User user) {
ConnectionPool pool = ConnectionPool.getInstance();
Connection connection = pool.getConnection();
PreparedStatement ps = null;
ResultSet rs = null;
String query ="INSERT INTO UserPass (EmailAddress, Password, Answer) VALUES (?, ?, ?)";
try {
ps = connection.prepareStatement(query);
ps.setString(1, user.getEmailAddress());
ps.setString(2, user.getPassword());
ps.setString(3, user.getAnswer());
return ps.executeUpdate();
} catch (SQLException e) {
e.printStackTrace();
return 0;
} finally {
DBUtil.closeResultSet(rs);
DBUtil.closePreparedStatement(ps);
pool.freeConnection(connection);
}
}
```

In this example, PreparedStatement object is used with parameters. Before executing the query, all special characters will be escaped. All SQL functions, those that are not intended to be exploited while stress testing [9] the application, are developed using PreparedStatements.

5.1.2 Cross-Site Scripting (XSS) Defense

For prevention code injection attacks, including SQLI and XSS, all user data should be validated. Several main rules should be followed to increase security:

- Check the data type and set length limits on any form fields on the website.
- Encode or escape the data where it is used in application to ensure that the browser treats the possibly dangerous content as text, and not as active content that could be executed.

From a security perspective, however, client-side validation is not adequate, because it does not protect the server-side code. An attacker can easily bypass the clientside using proxies.

5.1.3 Security Misconfiguration Defense

Maintaining security settings of the application, frameworks, application server, web server, database server, and the platform is a very complex problem. Web servers are frequent targets of attacks, so when trying to secure web servers, the following aspects should be taken into account [10]:

- Operating System
- Configuration
- Web content and server-side applications
- Documentation

Example:

HTTP server is subject to Slow type HTTP Attack [11].

There is several mechanisms to protect against this attack pattern [12].

The value of RequestReadTimeout directive should be set to range the time taken by the client to send the request [13].

The implementation of defense mechanisms is vital for mitigation of web vulnerabilities which significantly enhances the security of a web application. Some vulnerability can be exploited if an attacker performs several steps successively or in a specific order only.

5.2 Mitigation of Network Vulnerabilities

The practical and manageable way to mitigate the discovered vulnerability in the network or computing system is to apply patches (if any exits) that approaches the vulnerabilities. The view of an applying patch is to recognize controls and processes that will provide the organization with the proper protection against the vulnerabilities and threats identified by the vulnerability assessment program.

Network scan of Vikram University's computing environment identified security risks that could adversely affect University's critical operations and assets. These risks are quantified according to their likelihood of occurrence and the potential damage if they occur. Risk factors are calculated for each of the discovered vulnerability in order to prioritize remediation activities accordingly.

IJCRT

This next section presents the remediation plans for the issues encountered in Vikram University computing environment.

5.2.1 Remediation Plan for 128.168.1.4 (Microsoft 2012 Server)

128.168.1.4, is Microsoft Windows 2012 server located at School of Engineering and Technology in Vikram University campus network (shown in Fig 1).

During network vulnerability scan we found critical vulnerabilities listed in table 1.

Table 1 Vulnerabilities found at 128.168.1.4

SNo	CVE ID	Type	Severity Score
1	CVE-2013-3940	DoS Exec Code Overflow Mem. Corr	9.3
2	CVE-2013-3918	DoS Exec Code Overflow	9.3
3	CVE-2014-1807	Windows Shell File Association Vulnerability	7.2
4	CVE-2014-0255	Remote Denial of Service Vulnerability	8.9
5	CVE-2014-1812	Information Loss Vulnerability	8.1
6	CVE-2014-4971	Arbitrary Write Privilege Escalation Vulnerability	7.2
7	CVE-2015-6125	Windows DNS Use After Free Vulnerability	9.3

This section approaches towards mitigation of some of the discovered vulnerabilities by applying patches. Patching vulnerabilities are not like bandaging a wound or spackling a small hole. It is more like surgery. It is necessary to verify that the patch corrects the vulnerability without affecting operations of the system process and applications.

1. Remediation for CVE-2013-3940 (MS13-089)

CVE-2013-3940 allows remote attackers to execute arbitrary code or cause a denial of service (memory corruption) via a crafted image in a Windows Write (.wri) document, which is not properly handled in WordPad, aka "Graphics Device Interface Integer Overflow Vulnerability" [14].

CVSS score of CVE-2013-3940 is 9.3, other details are summarized in Table 2.

Table 2 Details of CVE-2013-3940

CVE-2013-3940	
Publish Date	2013-11-12
Access Control	Remote
Complexity	Medium (This indicates that the access conditions require some specialization).
Authentication	Not required (Authentication is not required to exploit the vulnerability).
Confidentiality Impact	Complete (all system files are revealed, means the total loss of the system)
Integrity Impact	Complete
Availability Impact	Complete

A remote code execution vulnerability exists in the way that the Windows Graphics Device Interface (GDI) processes specially crafted Windows Write files in WordPad. Impact of CVE-2013-3940 is high as the successful exploit by an attacker results in the total shutdown of the affected resource.

An attacker can take complete control of an affected system by successfully exploiting this vulnerability. Later, the attacker can install programs; view, change or delete data; or create new accounts with full user rights. Users whose accounts are configured to have fewer user rights on the system could be less impacted than users who operate with administrative user rights.

Factors of Vulnerability Mitigation

Mitigation refers to the setting up the standard or general configuration that could reduce the severity of vulnerability exploitation. For CVE-2013-3940, the following mitigating factors are considerable:

- For successful attacks using CVE-2013-3940, a user must have to open an attachment sent in an email. It cannot be exploited automatically through email.
- A website which involves a specially crafted Windows Write file can be used by an attacker to attempt to exploit vulnerability CVE-2013-3940 in a web-based attack scenario. Also, websites that allow or host user-provided content can comprise specially crafted content that could consequently exploit vulnerability CVE-2013-3940 [15]. An attacker would have no means to restrict users to view attacker-controlled content and open a specially crafted file. Alternatively, an attacker would have to persuade users to take action, typically by capturing them to click a link that takes them to the attacker's site and then turn them to open the specially crafted Write file.
- After successful exploitation of vulnerability CVE-2013-3940, an attacker can attain the most of the privileges same as the current user have. The user who operates with administrative privileges have more user rights and perform more impacted attacks than the users whose accounts are configured to have fewer user rights.

Workarounds

Workaround refers to a setting or configuration change that does not correct the underlying vulnerability but would help block known attack vectors before we apply the update [16]. The following workarounds reduce the functionality of CVE-2013-3940:

- *Disable the Word 6 converter by limiting access to mswrd8.wpc*

An administrator can implement an access control list to the concerned converter to guarantee WordPad no longer loads it. It effectively prevents exploitation of the issue.

Warning Before installing the security updates we must undo this workaround.

Run following commands using the command prompt to apply the access list. This is expected that some of these may result in an error message.

```
echo y| cacls "%ProgramFile%\Common File\MsShared\TextConv\mswrd832.cnv" /E /P/ Q everyone:N
echo y| cacls "%ProgramFile(x86)%\Common File\MSShared\TextConv\mswrd832.cnv" /E /P/Q everyone:N
echo y| cacls "%ProgramFile%\WinNT\Accessories\mswrd8.wpc" /E /P/Q everyone:N
echo y| cacls "%ProgramFile%\WinNT\Accessories\mswrd864.wpc" /E /P/Q everyone:N
echo y| cacls "%ProgramFile(x86)%\WinNT\Accessories\mswrd8.wpc" /E /P/Q /E /P/Q everyone:N
```

Impact of workaround While implementing the workaround, we will no longer be capable of converting Word 6 documents to Word 2003 or WordPad RTF documents. Microsoft Office Word will return an error message as "The file seems to be corrupted or damaged."

Undo the workaround:

```
echo y| cacls "%ProgramFile%\Common File\MS Shared\TextConv\mswrd832.cnv" /E /R/P everyone:N
echo y| cacls "% ProgramFile%\Common File\MS Shared\TextConv\mswrd832.cnv" /E /R/P everyone:N
echo y| cacls "%ProgramFile%\Win NT\Accessories\mswrd8.wpc" /E /R/P everyone:N
echo y| cacls "%ProgramFiles%\Windows NT\Accessories\mswrd864.wpc" /E /R/P everyone:N
echo y| cacls "%ProgramFiles(x86)%\Windows NT\Accessories\mswrd8.wpc" /E /R/P everyone:N
```

- *Do not open Windows Write (. wri) documents that accept from untrusted sources or that we receive surprisingly from trusted sources.*

Do not click on Windows Write (.wri) files received from untrusted sources or receive surprisingly from trusted sources. This vulnerability could be exploited when a user opens a specially crafted file.

2. Remediation for CVE-2014-1807 (MS14-027)

CVE-2014-1807 allows local users to gain privileges via a crafted application, as exploited in the wild in May 2014, aka “Windows Shell File Association Vulnerability.” CVSS score of CVE-2014-1807 is 7.2, other details are summarized in Table 3.

Table 3 Details of CVE-2014-1807

CVE-2014-1807	
Publish Date	2014-05-14
Access Control	Remote
Complexity	Low (Very little knowledge is needed to exploit the vulnerability)
Integrity Impact	Complete
Authentication	Not required
Confidentiality Impact	Complete
Availability Impact	Complete (total shutdown of the affected resource.)

3. Remediation for CVE-2015-6125 (MS15-127)

Windows Domain Name System (DNS) servers contain a remote code execution vulnerability when they fail to parse requests correctly. An attacker could run arbitrary code in the connection of the Local System Account after successful vulnerability exploitation. Windows server which is configured as DNS servers is mainly at risk from this vulnerability.

CVE-2015-6125 allows a remote attacker who is located at anywhere else, to execute arbitrary code via crafted requests, aka "Windows DNS Use After Free Vulnerability."

An attacker could create a specially crafted application to connect to a Windows DNS server and then issue malicious requests to the server. The update emphasis or demonstrates the vulnerability by modifying how Windows DNS servers parse requests.

CVSS score of CVE-2015-6125 is 9.3, other details are summarized in Table 4.

Table 4 Details of CVE-2015-6125

CVE-2015-6125	
Publish Date	2015-12-09
Access Control	Remote (The access conditions are somewhat specialized. Some preconditions must be satisfied to exploit).
Complexity	Medium
Authentication	Not required (No authentication required to exploit the vulnerability)
Confidentiality Impact	Complete (There is total information disclosure, resulting in all system files being exposed.)
Integrity Impact	Complete

Availability Impact	Complete

Patch for CVE-2015-6125 is only applicable to Windows-based servers that have the DNS server role installed. To remediate CVE-2015-6125, run the following command from command prompt to disable the background zone loading feature on the affected DNS server:

dnscmd /Config /DsMinimumBackgroundLoadThreads 0

This setting prevents incoming queries from being answered until zone loading is completed. Clients should be configured to use secondary DNS servers as a fallback in this scenario.

To re-enable background zone loading, run the following command from an elevated command prompt:

dnscmd /Config /DsMinimumBackgroundLoadThreads 1

4. Remediation for CVE-2014-4971 (MS14-062)

A vulnerability founds in the Microsoft Message Queuing (MSMQ) service. It allows an attacker to gain privileges on the targeted host or system. CVE-2014-4971 permits local users to write data to arbitrary memory locations, and subsequently, via a crafted address in an IOCTL call, it attains privileges, related to

i. The MQAC.Sys driver in MQ Access Control (MQAC) subsystem, and
ii. The BthPan.Sys driver in Bluetooth Personal Area Networking (PAN) subsystem.

CVSS score of CVE-2014-4971 is 7.2, other details summarized in Table 5.

Table 5 Details of CVE-2014-4971

CVE-2014-4971 (MS14-062)	
Publish Date	2014-07-26
Access Control	Remote
Complexity	Low (Very little knowledge or skill is required to exploit the vulnerability. Specialized access conditions or extenuating circumstances do not exist.)
Authentication	Not required (Authentication is not required to exploit the vulnerability).
Confidentiality Impact	Complete (The total information disclosure is found which subsequently results in the all system files being exposed)
Integrity Impact	Complete (It results in the total loss of system protection, that consequently compromised the entire system. i.e., MS14-062 results in a complete compromise of system integrity)
Availability Impact	Complete (The attacker can render the resource completely unavailable. There is a total shutdown of the affected resource)

Mitigating Factors

The following mitigating factors are essential to remediate CVE-2014-4971:

- Customers manually enable message Queuing component; moreover, it is not installed on any affected operating system. A user can only permit the Message Queuing component with administrative privileges, and those customers who enable it are more possibly vulnerable to this issue.
- The vulnerability could not be exploited remotely from any other machine or by anonymous users. An attacker must have valid login privileges and credentials, also should be able to log on locally from the system to exploit this vulnerability.

Workarounds

We have tested the following workarounds and to ensure that whether a workaround reduces functionality:

Disable the Message Queuing Service

We can protect the affected system from Message Queuing service by manually disabling it and protect the system from intruder's attempt to exploit this vulnerability. The following steps disable the Message Queuing service:

1. Click Start, and select Control Panel option.
2. Double-click to the Administrative Tools option.
3. Select the Services option and double-click on it.
4. Double-click onto the Message Queuing option.
5. Type list in the Startup, then click Disabled.
6. Select Stop option, and then click OK.

We can also stop and disable the MSMQ service by using the following command at the command prompt (available in Windows XP and also in the Microsoft Windows 2000 Resource Kit):

Sc stop MSMQ & sc config MSMQ start= disabled

The following primary issues and challenges were encountered while applying remediation plans for mitigation of security vulnerabilities in higher education environment [17]:

- University has too many unmanaged systems and doesn't have a universally deployed automated patching solution. Therefore, users are allowed to re-configure University's systems as they like.
- The IT department is unable to appropriately test patches to guarantee a successful deployment within the University's computing environment.
- The vulnerability management program creates a work queue that far enhances the organization's capability to address the identified vulnerability. Only the illustration of risks existence is not enough. Security persons or network

administrators of University must also be able to remediate or mitigate problem without disruptions of running operations that may worse than the originating risks.

- University either has no configuration management process, or the configuration management process is not integrated with the vulnerability assessment process.
- University has a tremendous diversity in its IT asset configuration and development and maintenance activities due to the lack of standardization or incompetent production controls.
- The organization pays a considerable amount of time performing unplanned work by maintaining IT assets (e.g., patching servers).

In an organization, the primary cause of security breaches is the unpatched software. For this reason, keeping system updated on a regular basis is critical to preventing or limiting system compromises. Mitigation of security vulnerabilities requires that the organization should standardize system configurations which significantly reduce the number of vulnerabilities. Nevertheless, validating and controlling the security and consistency of system configurations across large and open computing environments- in the data center, cloud, or hybrid infrastructures, can be a challenge on an ongoing basis. Continuous vulnerability assessment and risks monitoring ensures that critical vulnerabilities never affect or harm computing environments.

VI. Conclusion

As organizations adopt newer forms of information systems, the complexity of protecting those systems continues to increase. The lack of standards for quantifying the potential cost of computer security incidents is a significant problem for the management of information systems. Our work provides assistance for security persons to evaluate the possible damages caused by information security incidents with consideration of these factors.

We performed network scan in the real computing environment of Vikram University campus network, and results are analyzed to evaluate the security strength. A detailed account of vulnerabilities discovered in a higher educational environment such as weak password policies, remote access management and permissions to mandatory accounts is presented. The risk magnitude of identified vulnerabilities in University’s network configuration is measured which assists in design and development of remediation plans to enforce the security level of University computing environment.

The issues encountered during the implementation of vulnerability assessment process in Vikram University computing environment were investigated, and with these issues, remediation plans were discussed to mitigate vulnerabilities and bring organization at a secure level.

Thus, it can be concluded that our work will be helpful in maintaining secure networks and fewer incidents, at a reasonable cost and within limited time and resources. The increment in the management work is mostly compensated by a decrement in security problems, regarding occurrence and impact.

Overall, this paper has provided a way for approaching the severe problem of security risk vulnerability assessment with which most organizations are confronted today.

Acknowledgment

Authors are thankful to Vikram University, Ujjain for providing support and financial grant for the research work.

References

[1] C. Butler, Vulnerability Remediation, SANS Institute InfoSec Reading Room, 2014.

[2] National Vulnerability Database, Available: http://nvd.nist.gov

[3] C. Joshi, U. K. Singh, “Testbed for Mitigation of Network Vulnerabilities - Enhancing organization’s security resilience in an experimental environment”, International Journal of Computer Application (IJCA), Vol. 8, No. 2, April 2018.

[4] V. Sunkari and C. V. Guru Rao, “Preventing input type validation vulnerabilities using network based intrusion detection systems”, International Conference on Contemporary Computing and Informatics (IC3I), pp. 702 – 706, Nov. 2014.

[5] U. K. Singh, C. Joshi, “Information security risks management framework – A step towards mitigating security risks in university network”, Journal of Information Security and Applications, Elsevier, Vol. 35, Issue C, pp. 128–137, June 2017.

[6] K. K. Mookhey, Nilesh Burghate, Detection of SQL Injection and Cross-site Scripting Attacks, Symantec Connect Community, 02 November 2010.

[7] J. Weinberger, P. Saxena, D. Akhawe, M. Finifter, R. Shin, and D. Song, “A Systematic Analysis of XSS Sanitization in Web Application Frameworks”, University of California, Berkeley, 2011.

[8] Oracle Documentation. “Using Prepared Statements”, retrieved 2012 from: http://docs.oracle.com/javase/tutorial/jdbc/basics/prepared.html

[9] Yang Guang, J. J. and Jipeng, H., “System modules interaction based stress testing model”, The Second International Conference on Computer Engineering and Applications, (pp. 138-141) Bali Island, 2014.

[10] Neto, A. A., Duraes, J., Vieira, M., & Madeira, H. “Assessing and Comparing Security of Web Servers”, 14th IEEE Pacific International Symposium on Dependable Computing. IEEE Computer Society, 2008.

[11] Shekyan, S. Qualys Community. “Identifying Slow HTTP Attack Vulnerabilities on Web Applications”, 2013

[12] Shekyan, S. Qualys Community. “How to Protect Against Slow HTTP Attacks”, 2014

[13] Apache Software Foundation. “Security Tips, V 2.5”, Retrieved 2014, from: http://httpd.apache.org/docs/2.0/misc/security_tips.html

[14] Microsoft Security Bulletin MS13-089. Available: https://technet.microsoft.com/en-us/library/security/ms13-089.aspx

[15] Microsoft Security Bulletin MS14-062. Available:

https://technet.microsoft.com/library/security/MS14-062

[16] I. Muscat, “Web vulnerabilities: Identifying patterns and remedies”, Network Security, pp. 5-10, Feb 2016.

[17] U. K. Singh, C. Joshi, “Measurement of Security Dangers in University Network, International Journal of Computer Applications”, Vol. 155, No.1, pp 6-10, December 2016.