A Framework for Vulnerability Detection in Software Application "H2S" and Protection Against Command Injection Flaw

Anshika pandey[1], Mr.Vishal Shrivastava[2]

[1]Student M.Tech(CS), Arya College of Engineering,Jaipur
[2]Professor, Arya College of Engineering, Jaipur
[1]*anshika1987@gmail.com*
[2]*vishal500371@yahoo.com*

Abstract- "Application Security Assessment" is a process of providing complete security to the application from various vulnerabilities. Through this paper we are trying to detect what all vulnerabilities does our application "H2S" has and try to understand how can it affect our application. Application is taken as input from the user along with application's documentation. Also, User ID and password is to be given by the customer. After having all the required documents, application is deeply studied and understood.

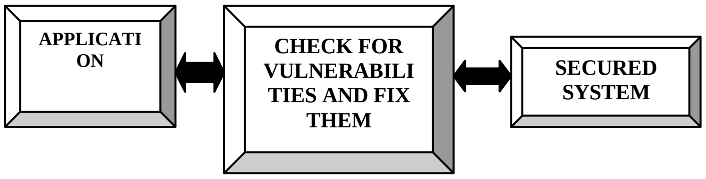
The main benefit of this application is that users can prevent their application and the essential information that an application has from getting affected by the external attackers. Firstly a threat profile is created and then vulnerabilities are checked. Various vulnerabilities checked by the project are:
INJECTION FLAW, CROSS-SITE SCRIPTING (XSS), CROSS SITE REQUEST FROGERY (CRSF), RE-DIRECTIONAL FLAW, SESSION MANAGEMENT, MALICIOUSFILE EXECUTION, INSECURE DIRECT OBJECT REFERENCE, INFORMATION LEAKAGE AND IMPROPER ERROR HANDLING
After checking for available vulnerabilities, risk is calculated using risk ranking matrix and finally provide solution so that application fully secured from external attacks.

Keywords-*H2S, Vulnerability, Assessment*

*****

(1). INTRODUCTION

More and more businesses are seeing information security as a business enabler, the enabler that can make or break a business, especially when there is heavy dependence on IT assets. A secure IT environment gives organization a competitive advantage over the competition. Threats to IT assets are ever changing. Applications are an IT component which sits on the top of IT assets. To secure and review an application it requires a great level of skills and knowledge [1]. With time there are a number of problems particularly SECURITY PROBLEMS that arises with the web applications.



A few are listed below-:

- Attacks on database and OS
- Fake emails
- Unauthorized access to the secured information
- Emails carrying malicious code
- Other hacker attacks like spoofing, DoS etc.

These security issues have come into play because the web application has some vulnerability better known as loop holes that are being exploited to breach the security wall that an application provides [2]. It is like a challenge to stop this exploitation as the exploiters keep coming up with new ways to carry out this malpractice. This dissertation takes up the challenge to counter this malpractice posing security threats to the users of the application. Some of the specific security problems (i.e. vulnerabilities) addressed by this paper are:

- Injection flaws
- Cross-site scripting (xss)
- Cross site request forgery (csrf)
- Re-directional flaw
- Session management
- Malicious file execution
- Insecure direct object reference

- Information leakage and improper error handling
- Insecure communication

(2). OVERVIEW

“H2S” is a vulnerable application build using PHP with APACHE server and SQL SERVER is used as backend tool for database storage[4].

Functionalities included in “H2S” application:

- ➢ Register new user
- ➢ Enables us to enter a new user to the application who can have access to the features available in application.
- ➢ Login/logout
- ➢ Setup/Reset the DB
- ➢ This feature enables us to reset the database back to its original state. Whenever, the check for any vulnerability is performed, there’re some changes that are made in the database [16]. So, in order to avoid the affect of such changes over the rest of application when checking it for other vulnerabilities, this feature is included[6].
- ➢ Add/view someone’s blog
- ➢ Show log
- ➢ Enables us to view all the pages that have been visited by a user in the application
- ➢ Browser info

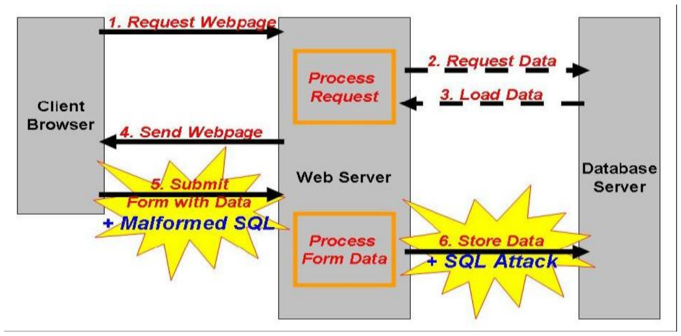
Gives us the details about the current configuration of the browser such as:

- ➢ IP address
- ➢ Hostname
- ➢ OS
- ➢ Remote client port etc
- ➢ User info Specifies all the details about the user who’s currently logged in
- ➢ DNS lookup gives the IP address of the site we put in textbox. The pre-requisite for this feature is that there has to connectivity with internet.
- ➢ Text file viewer enables user to view the content of the text file he chooses
- ➢ Source viewer enables user to view the source code of the page he wish to

(2.1) VULNERABILITIES

2.1.1 INJECTION FLAWS

Injection flaws allow attackers to relay malicious code through a web application to another system. These attacks include calls to the operating system via system calls, the use of external programs via shell commands, as well as calls to backend databases via SQL (i.e., SQL injection) Many web applications use operating system features and external programs to perform their functions[15]. Attacker can inject special (meta) characters, malicious commands, or command modifiers into the information and the web application will blindly pass these on to the external system for execution[2,8].



2.1.2 SQL INJECTION

A SQL injection attack consists of insertion or "injection" of a SQL query via the input data from the client to the application[14]. A successful SQL injection exploit can read sensitive data from the database, modify database data (Insert/Update/Delete), execute administration operations on the database (such as shutdown the DBMS), recover the content of a given file present on the DBMS file system and in some cases issue commands to the operating system. To exploit a SQL injection flaw, the attacker must find a parameter that the web application passes through to a database. By carefully embedding malicious SQL commands into the content of the parameter, the attacker can trick the web application into forwarding a malicious query to the database [3,10]. These attacks are not difficult to attempt and more tools are emerging that scan for these flaws. The consequences are particularly damaging, as an attacker can obtain, corrupt, or destroy database contents.

2.1.3 COMMAND INJECTION

The purpose of the command injection attack is to inject and execute commands specified by the attacker in the vulnerable application. In situation like this, the application, which executes unwanted system commands, is like a pseudo system shell, and the attacker may use it as any authorized system user. An OS command injection attack occurs when an attacker attempts to execute system level commands through a vulnerable application. Applications are considered vulnerable to the OS command injection attack if they utilize user input in a system level command [2].

2.1.4 CROSS-SITE SCRIPTING (XSS)

Cross-Site Scripting attacks are a type of injection problem, in which malicious scripts are injected into the otherwise benign and trusted web sites. Cross-site scripting (XSS) attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user in the output it generates without validating or encoding it[15].

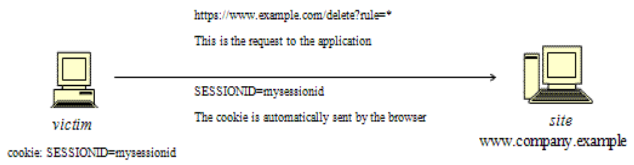
An attacker can use XSS to send a malicious script to an unsuspecting user. The end user's browser has no way to know that the script should not be trusted, and will execute the script [2,10].

2.1.5 CROSS SITE REQUEST FROGERY (CRSF)

CSRF is an attack which forces an end user to execute unwanted actions on a web application in which he/she is currently authenticated. With a little help of social engineering (like sending a link via email/chat), an attacker may force the users of a web application to execute actions of the attacker's choosing. A successful CSRF exploit can compromise end user data and operation, when it targets a normal user. If the targeted end user is the administrator account, a CSRF attack can compromise the entire web application[17].

2.1.6 RE-DIRECTIONAL FLAW

If a user gets an email from his bank stating that he has received some promotion offers so he should click on the link below to avail those offers. User ensures that the site is authentic by checking the name of his bank in the URL as he is aware of phishing attacks[2]. He finds it to be a genuine URL of the bank, so he clicks the link. On clicking the link the login page of his bank is displayed to him. He enters his username and password on the login page. He gets an error page saying "The server is unable to process your request"[3].


https://www.example.com/delete?rule=*
This is the request to the application
SESSIONID=mysessionid
The cookie is automatically sent by the browser
victim
cookie: SESSIONID=mysessionid
site
www.company.example

2.1.7 SESSION MANAGEMENT

Session Management broadly covers all controls on a user from authentication to leaving the application. HTTP is a stateless protocol, meaning that web servers respond to client requests without linking them to each other. Even simple application logic requires a user's multiple requests to be associated with each other across a "session"[5].

Most popular web application environments, such as ASP and PHP, provide developers with built-in session handling routines. Some kind of identification token will typically be issued, which will be referred to as a "Session ID" or Cookie. There are a number of ways in which a web application may interact with a user. Each is dependent upon the nature of the site, the security, and availability requirements of the application[4].

2.1.8 MALICIOUS FILE EXECUTION

Developers will often directly use or concatenate potentially hostile input with file or stream functions, or improperly trust input files. On many platforms, frameworks allow the use of external object references, such as URLs or file system references. When the data is insufficiently checked, this can lead to arbitrary remote and hostile content being included, processed or invoked by the web server[6].

This allows attackers to perform:

- Remote code execution
- Remote root kit installation and complete system compromise
- On Windows, internal system compromise may be possible through the use of PHP's SMB file wrappers

This attack is particularly prevalent on PHP, and extreme care must be taken with any stream or file function to ensure that user supplied input does not influence file names.

2.1.9 INSECURE DIRECT OBJECT REFERENCE

A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, database record, or key, as a URL or form parameter. An attacker can manipulate direct object references to access other objects without authorization, unless an access control check is in place[7].

For example, in Internet Banking applications, it is common to use the account number as the primary key. Therefore, it is tempting to use the account number directly in the web interface.

2.1.10 INFORMATION LEAKAGE AND IMPROPER ERROR HANDLING

Applications can unintentionally leak information about their configuration, internal workings, or violate privacy through a variety of application problems. Applications can also leak internal state via how long they take to process certain operations or via different responses to differing inputs, such as displaying the same error text with different error numbers. Web applications will often leak information about their internal state through detailed or debug error messages [19].

(3). FOR RUNNING APPLICATION

As we looked in the detail of all attacks vulnerability. The problem resides in our application as vulnerability assessment. We find out all the vulnerability in our application and then provide the detailed description and solution of one of the vulnerability. Here we mainly focus on Command injection flaw [2]. But before this we perform all the attacks in our application H2S to find out the vulnerability.

- We are using VMware to setup a server of window 7 operating system.
- The Ip address of server is given as-192.168.10.10
- The Ip address of client is given as-192.168.10.11
- The DNS address is given as 127.0.0.1
- IP address of www.google.com-173.194.36.52
- IP address of www.yahoo.com-206.106.90.245

3.1 Risk Rating Matrix

Level of access required	Complexity of Attack	Impact		
		High	Medium	Low
No Physical Access Required	Easy	High	High	Medium
	Requires Skills	High	Medium	Low
	Difficult	Medium	Medium	Low
Physical Access Required	Easy	Medium	Medium	Low
	Requires Skills	Medium	Low	Low
	Difficult	Low	Low	Low

(4). EXPLOTING VULNERABILITY

4.1 SQL INJECTION

There are three types of SQL comments:-

1.' or 1=1 --

2.'or1=1({

3.'or 1=1/*'

By applying simple SQL comment or 1=1--, the application shows all the account made in the application with their password and signature. If attacker wants to directly login with admin account he can use this comment on login window. After that he has full control on admin account and do whatever he wants to do in application [19].

4.2 COMMAND INJECTION

Using DIR

After applying www.google.com && DIR the application shows all the files of H2S application

Using SHUTDOWN

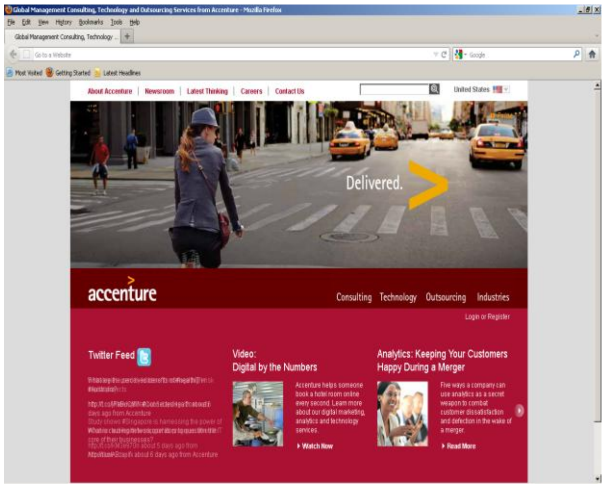
By Applying command shutdown www.google.com & shutdown –t 30 the server will shutdown in given time

4.3 CROSS SITE SCRIPTING (XSS)

By applying <script>alert("vulnerable to XSS");</script> XSS can be displayed on every blog view as alert[2].

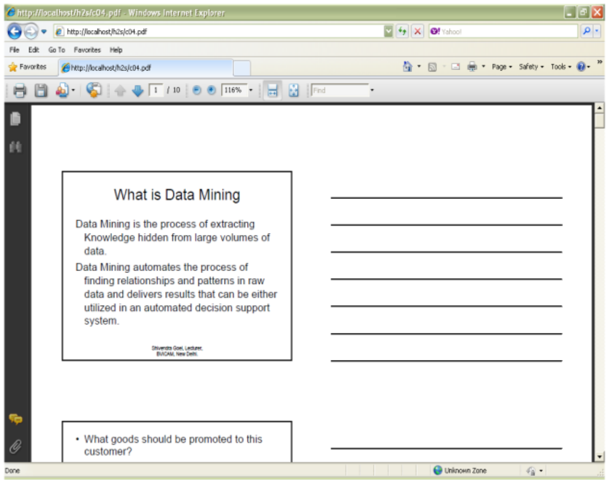
4.5 RE-DIRECTIONFLAW

By this flaw victim redirected to specific site given by attacker.


Global Management Consulting, Technology and Outsourcing Services from Accenture - Mozilla Firefox
About Accenture | Newsroom | Latest Thinking | Careers | Contact Us
United States
Delivered.
accenture
Consulting Technology Outsourcing Industries
Login or Register
Twitter Feed
Video: Digital by the Numbers
Analytics: Keeping Your Customers Happy During a Merger
Watch Now
Read More

4.6 INSECURE DIRECT OBJECT REFERENCE

By this attack attacker can view or redirect the victim to open another documents.


http://localhost/h2s/c04.pdf - Windows Internet Explorer
http://localhost/h2s/c04.pdf
File Edit Go To Favorites Help
Favorites
Page Safety Tools
1 / 10
116%
Find
Done
Unknown Zone

What is Data Mining

Data Mining is the process of extracting Knowledge hidden from large volumes of data.

Data Mining automates the process of finding relationships and patterns in raw data and delivers results that can be either utilized in an automated decision support system.

Shivendra Goel, Lecturer, BVICAM, New Delhi.

- What goods should be promoted to this customer?

(5). EVALUATION (Paper Work)

There is exponential increase in vulnerabilities found in Web Applications putting significant financial impact to the enterprise and privacy of the end users. 75% of total attacks now occur on Web applications. Systems and network administrators in last 5-10 years (end 1990s to early 00s) have achieved significant maturity on controlling OS and network level attacks. Strong OS hardening/patching procedures coupled with well managed firewalls provides sufficient surety to the business that these layers are secure and not easy to penetrate [15].

Web applications provide a logical tunnel from outside/Internet to the backend databases inside the enterprise. Web applications are complex piece of code with a mix of customized business logic, third party libraries, back-end database routines and integration to multiple other applications. Complexity increases potential points of failures [2].

The following are the activities involve in deploying the above said solution-:

- Study of application received from customer
- Create a threat profile
- Expose all possible vulnerabilities in the existing application.
- Assign risk rating to vulnerabilities
- Propose a solution for the vulnerabilities exposed.
- Implement and test the solutions proposed, independent of each other.

(6). IMPLEMENTATION

We focus on command injection flaw.

6.1 COMMAND INJECTION

Command injection is basically injection of operating system commands to be executed through a web-app. The purpose of the command injection attack is to inject and execute commands specified by the attacker in the vulnerable application. In situation like this, the application, which executes unwanted system commands, is like a pseudo system shell, and the attacker may use it as any authorized system user. However, commands are executed with the same privileges and environment as the application has. Command injection attacks are possible in most cases because of lack of correct input data validation, which can be manipulated by the attacker (forms, cookies, HTTP headers etc.)[2,3].

Command injection attacks can occur when a web application executes system commands – say – a webapp that runs nslookup queries for you. If the input that is passed to the shell command is not correctly sanitized, an attacker can *inject* extra shell commands and have your application run them under the privileges of the webapp – normally the privileges of the web-server.

Put simply, it means the attacker can execute commands on your box, leading to total system compromise. Yes, this is a very serious vulnerability.

Ok, so how does all this work?

Simple example.

```
<?php
  $host = 'google';
  if (isset( $_GET['host'] ) )
    $host = $_GET['host'];
  system("nslookup " . $host);
?>
<form method="get">
<select name="host">
    <option value="google.com">google</option>
    <option value="yahoo.com">yahoo</option>
  </select>
  <input type="submit">
</form>
```

This piece of code accepts the GET parameter "host" and runs the nslookup command on it, giving you output regarding its IP address.

The important part is to see how the $host parameter (the GET parameter) is passed directly to the system () function without any filtering or sanitization of input[19].Those of you familiar with the Unix command line will know we can "stack" commands by using a semicolon, like so…nslookup google.com;cat /etc/passwd

6.2 Command Injections types:

```
&& dir
&& wmic process list
&& wmic useraccount list
&& copy c:\WINDOWS\repair\sam && copy
c:\WINDOWS\repair\system.bak
```

6.3 SOLUTION:-

We explained above that by applying both command injection our application gives vulnerability on DNS Lookup and shutdown the server. Basically the attack name is "shell_exec".

The script is as follow

```
// Grab inputs
$targethost = $_REQUEST["target_host"];
echo "<pre>";
echo shell_exec("nslookup  " . $targethost);
echo "</pre>";
//phpinfo();
```

Here the shell_exec is the command injection attack which targeted on nslookup.

Solution:

By changing the script in application to this the output will show the ip address of google.com, yahoo.com, client.com, H2S.com.

```
// Grab inputs
$targethost = $_REQUEST["target_host"];
//echo "<pre>";
$ip = gethostbyname($targethost);
echo $ip;
```

```
//echo "</pre>";
//phpinfo();
```

6.4 AFTER APPLYING SOLUTION-
IP address of www.google.com-173.194.36.52
The IP address of www.yahoo.com-206.106.90.245
DIR command injection is not working now.
www.google.com & shutdown –t 30

(7). CONCLUSION

Any business application is vulnerable to threats. Thus, it is necessary to test the application security to make sure it is prevented from outside attacks. Due to the nature of application of being getting attacked from external sources, proposed solution was required. Threat profile is created which consists of all the threats that an application faces. System's security highly depends upon the quality of threat profile. Using this threat profile, we check the system for the available vulnerabilities and fix them, making the application more secure. We try to secure application from getting affected by the outside attackers. Not only application but database, Operating System, web browser etc are also prevented to get affected. We targeted on command injection flaw and apply the solution to remove the vulnerability.

The solution would also lead ISP's and companies to ensure that their servers are not misused and/or take steps to quickly remedy the situation when they are. The risk ranking framework enables us to determine the risk for each vulnerability. Hence, through this procedure we ensure application to be secured.

7.1 Limitations of the System

- Although this project gives an understanding of various application level vulnerabilities which are common to many applications but information security field is so vast that I can't cover all aspects of it.
- Things like creating exploits (For platforms eg. Windows, Unix etc.), Secure Network design, Vulnerability assessment of different platform and other Information security standard like would remain untouched.
- Good networking knowledge and knowledge of protocols is required.

7.2 Future Scope for Modification

As for future work, an extension of our work is possible. We would like to investigate the application for other professional services i.e. penetration testing, vulnerability assessment etc. We are confident that these extensions of the measurements cannot harm the validity of our current results, but can furthermore help us to get a more secured application Also there might be some shortcomings in our proposed solution which we would like to test and fix.

(8). ACKNOWLEDGEMENT

This work was supported by the Arya College of Engineering and Technology, Jaipur Rajasthan.

(9). REFRENCES/BIBLOGRAPHY

[1] Steven, J. "State of Application Assessment" Security & Privacy, IEEE (Volume:6 , Issue: 6) Nov.-Dec. 2008

[2] Curphey, M. ; Foundstone, Mission Viejo, CA ; Arawo, R." Web application security assessment tools" Security & Privacy, IEEE (Volume:4 , Issue: 4) July-Aug. 2006 ISSN :1540-7993

[3] Teodoro, N. ; DCTI/ADETTI, ISCTE-IUL, Lisbon, Portugal ; Serrao, C." Assessing the Portuguese Web applications security" Internet Security (WorldCIS), 2011 IEEE World Conference on 21-23 Feb. 2011 ,London

[4] Maan, F. ; Nat. Univ. of Sci. & Technol. (NUST), Islamabad, Pakistan ; Abbas, Y. ; Mazhar, N." Vulnerability assessment of AODV and SAODV routing protocols against network routing attacks and performance comparisons" Wireless Advanced (WiAd) IEEE, 2011 20-22 June 2011 London

[5] Mallah, G.A. ; Shaikh, Z.A." Vulnerability assessment through mobile agents" IEEE E-Tech 2004 31 July 2004

[6] VULNERABILITY ASSESS MENT WHITEPAPERINTRODUCTION,IMPLEMENTATION AND TECHNOLOGY DISCUSSION copyright© 2003 security METRICS

[7] Vulnerability Assessment Susan Cima July 6, 2001 Version 1.2e

[8] What is a vulnerability assessment? Prepared on July 17, 2011 by: Demyo Inc.American www.demyo.com, info@demyo.com, Miami, Florida, USA

[9] Joel Scambray, Mike Shoma "Hacking Exposed-web applications", McGraw Hill

[10] Susan Elizabeth Young, Dave Aitel "The hacker's handbook", CRC Press

[11] Mark O'Neill, Phillip Hallam-Baker "Web services security", McGraw-Hill Professional.

[12] "PLYNT"- the newsletter of PALADION

[13] Implementing a Successful Security Assessment Process, Bradley Hart, GSEC Version 1.2e, August 21, 2001
[14] The Microsoft IT group shares its experiences, White Paper, Published: January 2003
[15] V.Benjamin Livshits and Monica S. Lam"Finding Security Vulnerabilities in Java Applications, with Static Analysis" Computer Science Department Stanford University{livshits,lam}@cs.stanford.edu.
[16] Laurie Williams, Michael Gerick, and Andrew Meneely "Protection Poker: Structuring Software Security Risk "Assessment and Knowledge Transfer, North Carolina State University Department of Computer Science {lawilli3, mcgegick, impanel}@ncsu.edu
[17] Chuck Willis "WASC Web Application Security Statistics Published on November 2, 2009
[18] OWASP_Application_Security_Assessment_Standards_Project
[19] OWASP_webgoat_project