REPORT ON VULNWEB

This report shows the vulnerabilities founded in http://testasp.vulnweb.com/. The following vulnerabilities are some critical & important vulnerabilities founded in the website http://testasp.vulnweb.com/.

Domain : vulnweb

Sub-Domain : testasp.vulnweb

1.Cross-site Scripting:

Attack Pattern: "><scRipt>alert(1)</scRipt>

Vulnerability Details:

The cross-site scripting, which allows an attacker to execute a dynamic script (JavaScript, VBScript) in the context of the application.

This allows several different attack opportunities, mostly hijacking the current session of the user or changing the look of the page by changing the HTML on the fly to steal the user's credentials.

This happens because the input entered by a user has been interpreted as HTML/JavaScript/VBScript by the browser. Cross-site scripting targets the users of the application instead of the server.

Although this is a limitation, since it allows attackers to hijack other users' sessions, an attacker might attack an administrator to gain full control over the application.

Impact:

There are many different attacks that can be leveraged through the use of cross-site scripting, including:

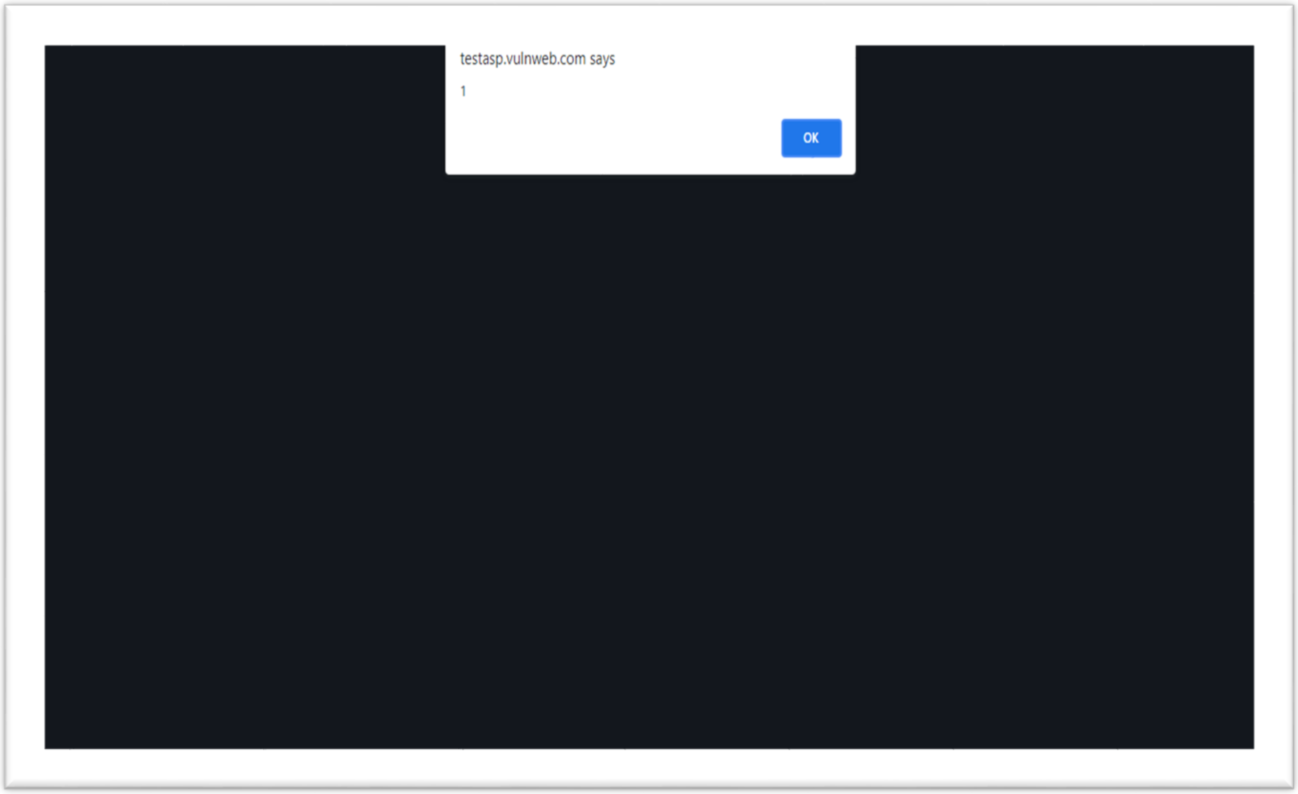
- Hijacking user's active session.
- Mounting phishing attacks.
- Intercepting data and performing man-in-the-middle attacks.

Rectify:

The issue occurs because the browser interprets the input as active HTML, JavaScript or VBScript.

To avoid this, output should be encoded according to the output location and context. For example, if the output goes in to a JavaScript block within the HTML document, then output needs to be encoded accordingly.

Encoding can get very complex; therefore, it's strongly recommended to use an encoding library such as OWASP ESAPI and Microsoft Anti-cross-site scripting.


testasp.vulnweb.com says

1

2. Password Transmitted over HTTP:

Vulnerability Details:

The password data is being transmitted over HTTP.

Impact:

If an attacker can intercept network traffic, he/she can steal users' credent.

Move all of your critical forms and pages to HTTPS and do not serve them over HTTP.

Rectify:

All sensitive data should be transferred over HTTPS rather than HTTP. Forms should be served over HTTPS.

All aspects of the application that accept user input, starting from the login process, should only be served over HTTPS.

3.Blind SQL Injection:

Attack Pattern: %27+WAITFOR+DELAY+%270%3a0%3a25%27--

Vulnerability Details:

The blind SQL injection, which occurs when data input by a user is interpreted as an SQL command rather than as normal data by the backend database.

This is an extremely common vulnerability and its successful exploitation can have critical implications.

The vulnerability is confirmed by executing a test SQL query on the backend database. In these tests, SQL injection was not obvious, but the different responses from the page based on the injection test allowed us to identify and confirm the SQL injection.

Impact:

Depending on the backend database, the database connection settings, and the operating system, an attacker can mount one or more of the following attacks successfully:

- Reading, updating and deleting arbitrary data or tables from the database
- Executing commands on the underlying operating system

Actions to Take:

If you are not using a database access layer (DAL), consider using one. This will help you centralize the issue.

You can also use ORM (object relational mapping). Most of the ORM systems use only parameterized queries and this can solve the whole SQL injection problem.

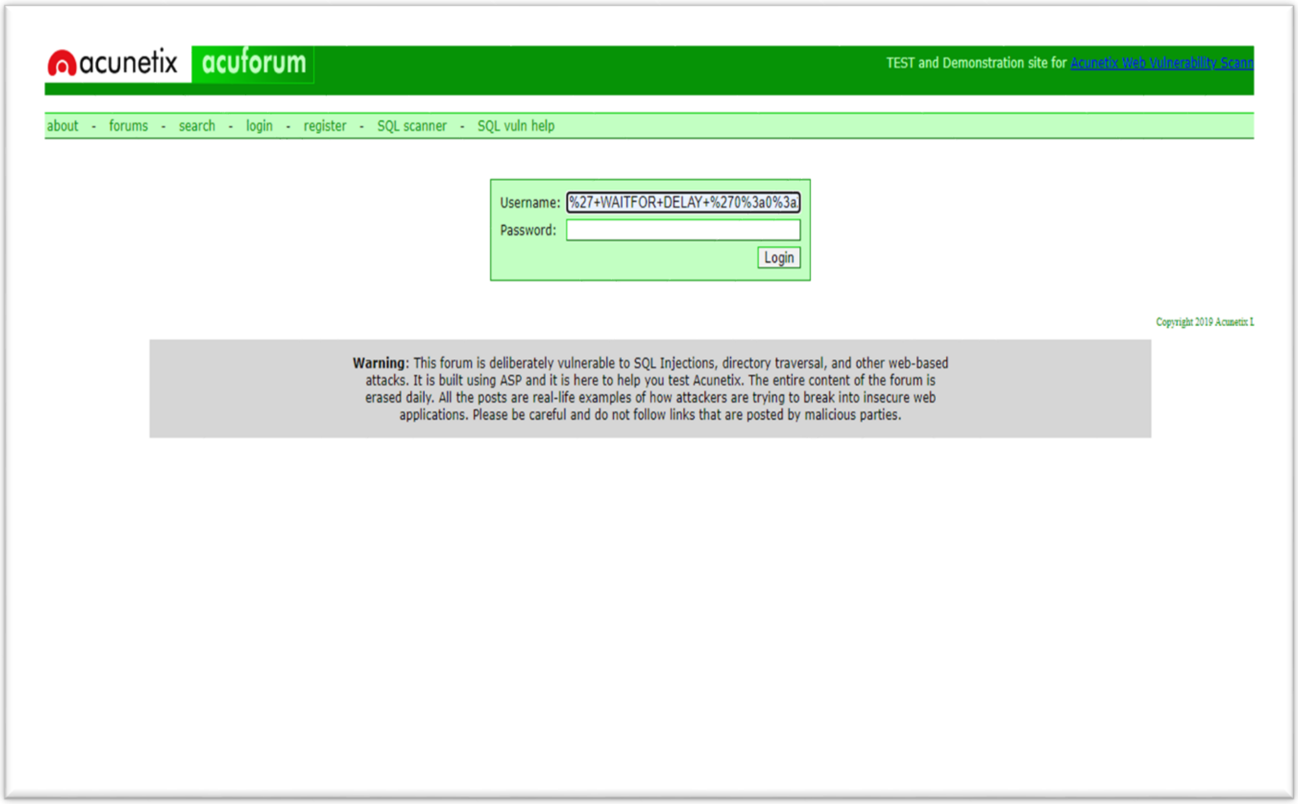
Locate the all dynamically generated SQL queries and convert them to parameterized queries. (If you decide to use a DAL/ORM, change all legacy code to use these new libraries.)

Use your weblogs and application logs to see if there were any previous but undetected attacks to this resource.

Rectify:

A robust method for mitigating the threat of SQL injection-based vulnerabilities is to use parameterized queries (prepared statements).

Almost all modern languages provide built-in libraries for this. Wherever possible, do not create dynamic SQL queries or SQL queries with string concatenation.


acunetix acuforum
TEST and Demonstration site for Acunetix Web Vulnerability Scann
about - forums - search - login - register - SQL scanner - SQL vuln help
Login
Copyright 2019 Acunetix L

Username: %27+WAITFOR+DELAY+%270%3a0%3a

Password:

Warning: This forum is deliberately vulnerable to SQL Injections, directory traversal, and other web-based attacks. It is built using ASP and it is here to help you test Acunetix. The entire content of the forum is erased daily. All the posts are real-life examples of how attackers are trying to break into insecure web applications. Please be careful and do not follow links that are posted by malicious parties.

4. Source Code -Disclosure:

Attack Pattern: Templatize.asp

Vulnerability Details:

The possible source code disclosure (Generic).

An attacker can obtain server-side source code of the web application, which can contain sensitive data - such as database connection strings, usernames and passwords - along with the technical and business logic of the application.

Impact:

Depending on the source code, database connection strings, username and passwords, the internal workings and business logic of the application might be revealed. With such information, an attacker can mount the following types of attacks:

Access the database or other data resources. Depending on the privileges of the account obtained from the source code, it may be possible to read, update or delete arbitrary data from the database.

Gain access to password protected administrative mechanisms such as dashboards, management consoles and admin panels, hence gaining full control of the application.

Develop further attacks by investigating the source code for input validation errors and logic vulnerabilities.

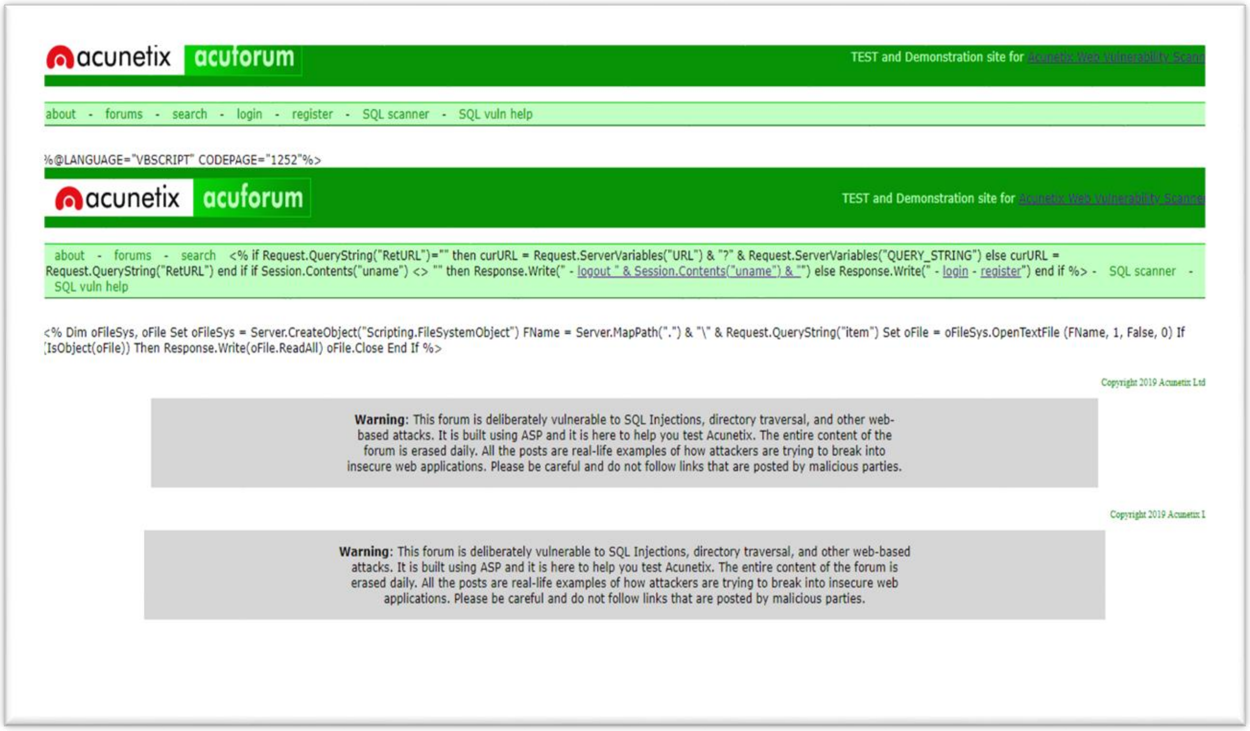
Actions to Take:

Confirm exactly what aspects of the source code are actually disclosed; due to the limitations of these types of vulnerability, it might not be possible to confirm this in all instances. Confirm this is not an intended functionality.

If it is a file required by the application, change its permissions to prevent public users from accessing it. If it is not, then remove it from the web server.

Ensure that the server has all the current security patches applied.

Remove all temporary and backup files from the web server.


acunetix acuforum
TEST and Demonstration site for Acunetix Web Vulnerability Scann
about - forums - search - login - register - SQL scanner - SQL vuln help

```
%@LANGUAGE="VBSCRIPT" CODEPAGE="1252"%>
```

acunetix acuforum
TEST and Demonstration site for Acunetix Web Vulnerability Scanner
about - forums - search

```
<% if Request.QueryString("RetURL")="" then curURL = Request.ServerVariables("URL") & "?" & Request.ServerVariables("QUERY_STRING") else curURL =
Request.QueryString("RetURL") end if if Session.Contents("uname") <> "" then Response.Write(" - logout " & Session.Contents("uname") & "") else Response.Write(" - login - register") end if %>
```

- SQL scanner - SQL vuln help

```
<% Dim oFileSys, oFile Set oFileSys = Server.CreateObject("Scripting.FileSystemObject") FName = Server.MapPath(".") & "\" & Request.QueryString("item") Set oFile = oFileSys.OpenTextFile (FName, 1, False, 0) If
(IsObject(oFile)) Then Response.Write(oFile.ReadAll) oFile.Close End If %>
```

Copyright 2019 Acunetix Ltd

Warning: This forum is deliberately vulnerable to SQL Injections, directory traversal, and other web-based attacks. It is built using ASP and it is here to help you test Acunetix. The entire content of the forum is erased daily. All the posts are real-life examples of how attackers are trying to break into insecure web applications. Please be careful and do not follow links that are posted by malicious parties.

Copyright 2019 Acunetix L

Warning: This forum is deliberately vulnerable to SQL Injections, directory traversal, and other web-based attacks. It is built using ASP and it is here to help you test Acunetix. The entire content of the forum is erased daily. All the posts are real-life examples of how attackers are trying to break into insecure web applications. Please be careful and do not follow links that are posted by malicious parties.