Comprehensive Report

Acunetix Threat Level 3

One or more high-severity type vulnerabilities have been discovered by the scanner. A malicious user can exploit these vulnerabilities and compromise the backend database and/or deface your website.

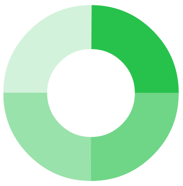
Scan Detail

Target	https://www.anorc.org
Scan Type	Full Scan
Start Time	Sep 4, 2022, 8:31:02 AM GMT+4
Scan Duration	40 minutes
Requests	98704
Average Response Time	2ms
Maximum Response Time	9025ms

4	6	4	4
High	Medium	Low	Informational

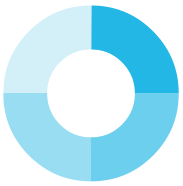
Severity	Vulnerabilities	Instances
● High	4	4
● Medium	6	6
● Low	4	4
● Informational	4	4
Total	18	18

Informational



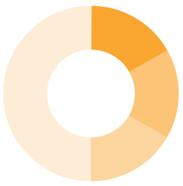
		Instances
🟩	Microsoft IIS version disclosure	1
🟩	Subresource Integrity (SRI) not implemented	1
🟩	TLS/SSL (EC)DHE Key Reuse	1
🟩	Others	1

Low Severity



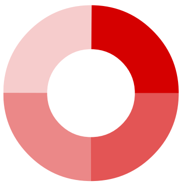
		Instances
🟦	Clickjacking: CSP frame-ancestors missing	1
🟦	Cookies with missing, inconsistent or contrad…	1
🟦	Cookies without HttpOnly flag set	1
🟦	Others	1

Medium Severity



		Instances
🟧	Application error messages	1
🟧	TLS 1.1 enabled	1
🟧	TLS/SSL LOGJAM attack	1
🟧	Others	3

High Severity



		Instances
🟥	Cross site scripting	1
🟥	Microsoft IIS tilde directory enumeration	1
🟥	SQL injection	1
🟥	Others	1

Impacts

SEVERITY	IMPACT	
High	1	Cross site scripting
High	1	Microsoft IIS tilde directory enumeration
High	1	SQL injection
High	1	TLS 1.0 enabled
Medium	1	Application error messages
Medium	1	TLS 1.1 enabled
Medium	1	TLS/SSL LOGJAM attack
Medium	1	TLS/SSL Sweet32 attack
Medium	1	TLS/SSL Weak Cipher Suites
Medium	1	URL redirection
Low	1	Clickjacking: CSP frame-ancestors missing
Low	1	Cookies with missing, inconsistent or contradictory properties
Low	1	Cookies without HttpOnly flag set
Low	1	Cookies without Secure flag set
Informational	1	Microsoft IIS version disclosure
Informational	1	Subresource Integrity (SRI) not implemented
Informational	1	TLS/SSL (EC)DHE Key Reuse
Informational	1	Web Application Firewall detected

Cross site scripting

Cross-site Scripting (XSS) refers to client-side code injection attack wherein an attacker can execute malicious scripts into a legitimate website or web application. XSS occurs when a web application makes use of unvalidated or unencoded user input within the output it generates.

Impact

Malicious JavaScript has access to all the same objects as the rest of the web page, including access to cookies and local storage, which are often used to store session tokens. If an attacker can obtain a user's session cookie, they can then impersonate that user.

Furthermore, JavaScript can read and make arbitrary modifications to the contents of a page being displayed to a user. Therefore, XSS in conjunction with some clever social engineering opens up a lot of possibilities for an attacker.

---

https://www.anorc.org/fa/

URI was set to **"onmouseover='qTsi(91432)'bad="**

The input is reflected inside a tag parameter between double quotes.

Request

```
GET /fa/news2/4?"onmouseover='qTsi(91432)'bad=" HTTP/1.1
Referer: https://www.anorc.org/
Cookie: ASPSESSIONIDQEAQCTQR=KBNPCCPBPNGDAEEEIJFJEDLP; theme=1; Lang=fa
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4512.0 Safari/537.36
Host: www.anorc.org
Connection: Keep-alive
```

Recommendation

Apply context-dependent encoding and/or validation to user input rendered on a page

Description

In order for a Cross-site scripting (XSS) attack to take place, an attacker does not directly target a victim. Instead, an attacker exploits a vulnerability in a web application visited by a victim, where the web application is used to deliver the malicious JavaScript. The victim's browser is not able to distinguish between malicious and legitimate JavaScript, and therefore, executes the attacker's malicious payload.

Since cross-site scripting (XSS) is user input which is interpreted as code. In order to prevent XSS, secure input handling is necessary. The two fundamental methods of handling untrusted user input are **encoding** and **validation**.

Encoding - Escapes user input so that browsers interpret it as **data**, not as code
Validation - Filters user input so that browsers interpret it as code without malicious commands

Encoding and validation are two different techniques to preventing cross-site scripting (XSS). Deciding which should be used highly depends on the **context** within which the untrusted user input is being inserted.

The following are two examples of the most common cross-site scripting (XSS) contexts.

```
<!-- HTML element -->
<div>userInput</div>

<!-- HTML attribute -->
<input value="userInput">
```

The method for preventing cross-site (XSS) scripting in the two examples above is different. In the first example, where user input is inserted in an HTML element, HTML encoding is the correct way to prevent XSS. However, in the second example, where user input is inserted in an HTML attribute, validation (in this case, filtering out `'` and `"`)is the appropriate prevention method.

```
<!-- Application code -->
<input value="userInput">

<!-- Malicious string -->
"><script>...</script><input value="

<!-- Resulting code -->
<input value=""><script>...</script><input value="">
```

In **most** of the time, encoding should be performed whenever user input is included in a page, however, as with the above example, in some cases, encoding has to be replaced by or complemented with validation.

It's important to remember that secure input handling has to take into account which context of a page the user input is inserted into.

References

Cross-site Scripting (XSS) Attack - Acunetix
https://www.acunetix.com/websitesecurity/cross-site-scripting/

Types of XSS - Acunetix
https://www.acunetix.com/websitesecurity/xss/

XSS Filter Evasion Cheat Sheet
https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet

Excess XSS, a comprehensive tutorial on cross-site scripting
https://excess-xss.com/

Cross site scripting

https://en.wikipedia.org/wiki/Cross-site_scripting

Microsoft IIS tilde directory enumeration

It is possible to detect short names of files and directories which have an 8.3 file naming scheme equivalent in Windows by using some vectors in several versions of Microsoft IIS. For instance, it is possible to detect all short-names of ".aspx" files as they have 4 letters in their extensions. This can be a major issue especially for the .Net websites which are vulnerable to direct URL access as an attacker can find important files and folders that they are not normally visible.

Impact

Possible sensitive information disclosure.

---

https://www.anorc.org/

Request

```
GET /userfiles/images/menue//*~1*/a.aspx?aspxerrorpath=/ HTTP/1.1
Cookie: ASPSESSIONIDQEAQCTQR=KBNPCCPBPNGDAEEEIJFJEDLP; theme=1; Lang=fa
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4512.0 Safari/537.36
Host: www.anorc.org
Connection: Keep-alive
```

Recommendation

Consult the "Prevention Technique(s)" section from Soroush Dalili's paper on this subject. A link to this paper is listed in the Web references section below.

References

Windows Short (8.3) Filenames - A Security Nightmare?
https://www.acunetix.com/blog/web-security-zone/windows-short-8-3-filenames-web-security-problem/

Detectify KB: Microsoft IIS Tilde Vulnerability
https://web.archive.org/web/20150921104258/https://support.detectify.com/customer/portal/articles/1711520-microsoft-iis-tilde-vulnerability

Microsoft IIS Shortname Scanner PoC
https://github.com/irsdl/iis-shortname-scanner/

Microsoft IIS tilde character “~” Vulnerability/Feature – Short File/Folder Name Disclosure https://soroush.secproject.com/blog/2012/06/microsoft-iis-tilde-character-vulnerabilityfeature-short-filefolder-name-disclosure/

IIS Short File Name Disclosure is back! Is your server vulnerable?
https://soroush.secproject.com/blog/2014/08/iis-short-file-name-disclosure-is-back-is-your-server-vulnerable/

SQL injection

SQL injection (SQLi) refers to an injection attack wherein an attacker can execute malicious SQL statements that control a web application's database server.

Impact

An attacker can use SQL injection to bypass a web application's authentication and authorization mechanisms and retrieve the contents of an entire database. SQLi can also be used to add, modify and delete records in a database, affecting data integrity. Under the right circumstances, SQLi can also be used by an attacker to execute OS commands, which may then be used to escalate an attack even further.

---

https://www.anorc.org/fa/pages/28

HTTP Header input **Referer** was set to **@@fPh1T**

Error message found:

```
Microsoft OLE DB Provider for ODBC Drivers
```

Request

```
GET /fa/pages/28 HTTP/1.1
Referer: @@fPh1T
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4512.0 Safari/537.36
Cookie: theme=1; Lang=fa
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
Host: www.anorc.org
Connection: Keep-alive
```

Recommendation

Use parameterized queries when dealing with SQL queries that contain user input. Parameterized queries allow the database to understand which parts of the SQL query should be considered as user input, therefore solving SQL injection.

Description

In order for an SQL injection attack to take place, the vulnerable website needs to directly include user input within an SQL statement. An attacker can then insert a payload that will be included as part of the SQL query and run against the database server.

The following server-side **pseudo-code** is used to authenticate users to the web application.

```
# Define POST variables
uname = request.POST['username']
passwd = request.POST['password']

# SQL query vulnerable to SQLi
sql = "SELECT id FROM users WHERE username='" + uname + "' AND password='" + passwd +
"'"

# Execute the SQL statement
database.execute(sql)
```

The above script is a simple example of authenticating a user with a username and a password against a database with a table named users, and a username and password column.

The above script is vulnerable to SQL injection because an attacker could submit malicious input in such a way that would alter the SQL statement being executed by the database server.

A simple example of an SQL injection payload could be something as simple as setting the password field to `password' OR 1=1`.

This would result in the following SQL query being run against the database server.

```
SELECT id FROM users WHERE username='username' AND password='password' OR 1=1'
```

An attacker can also comment out the rest of the SQL statement to control the execution of the SQL query further.

```
-- MySQL, MSSQL, Oracle, PostgreSQL, SQLite
' OR '1'='1' --
' OR '1'='1' /*
-- MySQL
' OR '1'='1' #
-- Access (using null characters)
' OR '1'='1' %00
' OR '1'='1' %16
```

Once the query executes, the result is returned to the application to be processed, resulting in an authentication bypass. In the event of authentication bypass being possible, the application will most likely log the attacker in with the first account from the query result — the first account in a database is usually of an administrative user.

What's the worst an attacker can do with SQL?

SQL is a programming language designed for managing data stored in an RDBMS, therefore SQL can be used to access, modify and delete data. Furthermore, in specific cases, an RDBMS could also run commands on the operating system from an SQL statement.

Keeping the above in mind, when considering the following, it's easier to understand how lucrative a successful SQL injection attack can be for an attacker.

An attacker can use SQL injection to bypass authentication or even impersonate specific users.
One of SQL's primary functions is to select data based on a query and output the result of that query. An SQL injection vulnerability could allow the complete disclosure of data residing on a database server.
Since web applications use SQL to alter data within a database, an attacker could use SQL injection to alter data stored in a database. Altering data affects data integrity and could cause repudiation issues, for instance, issues such as voiding transactions, altering balances and other records.
SQL is used to delete records from a database. An attacker could use an SQL injection vulnerability to delete data from a database. Even if an appropriate backup strategy is employed, deletion of data could affect an application's availability until the database is restored.
Some database servers are configured (intentional or otherwise) to allow arbitrary execution of operating system commands on the database server. Given the right conditions, an attacker could use SQL injection as the initial vector in an attack of an internal network that sits behind a firewall.

Preventing SQL injection using parameterized queries

SQL injection is one of the most widely spread and most damaging web application vulnerabilities. Fortunately, both the programming languages, as well as the RDBMSs themselves have evolved to provide web application developers with a way to safely query the database — parameterized SQL queries.

Parameterized queries are simple to write and understand while forcing a developer to define the entire SQL statement before hand, using placeholders for the actual variables within that statement. A developer would then pass in each parameter to the query after the SQL statement is defined, allowing the database to be able to distinguish between the SQL command and data inputted by a user. If SQL commands are inputted by an attacker, the parameterized query would treat the input as a string as opposed to an SQL command.

Application developers should avoid sanitizing their input by means of escaping or removing special characters (several encoding tricks an attacker could leverage to bypass such protections) and stick to using parameterized queries in order to avoid SQL injection vulnerabilities.

References

SQL Injection (SQLi) - Acunetix
https://www.acunetix.com/websitesecurity/sql-injection/

Types of SQL Injection (SQLi) - Acunetix
https://www.acunetix.com/websitesecurity/sql-injection2/

Prevent SQL injection vulnerabilities in PHP applications and fix them - Acunetix
https://www.acunetix.com/blog/articles/prevent-sql-injection-vulnerabilities-in-php-applications/

SQL Injection - OWASP

https://www.owasp.org/index.php/SQL_Injection

Bobby Tables: A guide to preventing SQL injection
https://bobby-tables.com/

SQL Injection Cheet Sheets - Pentestmonkey
http://pentestmonkey.net/category/cheat-sheet/sql-injection

TLS 1.0 enabled

The web server supports encryption through TLS 1.0, which was formally deprecated in March 2021 as a result of inherent security issues. In addition, TLS 1.0 is not considered to be "strong cryptography" as defined and required by the PCI Data Security Standard 3.2(.1) when used to protect sensitive information transferred to or from web sites. According to PCI, "30 June 2018 is the deadline for disabling SSL/early TLS and implementing a more secure encryption protocol – TLS 1.1 or higher (TLS v1.2 is strongly encouraged) in order to meet the PCI Data Security Standard (PCI DSS) for safeguarding payment data.

Impact

An attacker may be able to exploit this problem to conduct man-in-the-middle attacks and decrypt communications between the affected service and clients.

---

https://www.anorc.org/ Confidence: 100%

The SSL server (port: 443) encrypts traffic using TLSv1.0.

Recommendation

It is recommended to disable TLS 1.0 and replace it with TLS 1.2 or higher.

References

RFC 8996: Deprecating TLS 1.0 and TLS 1.1
https://tools.ietf.org/html/rfc8996

Are You Ready for 30 June 2018? Saying Goodbye to SSL/early TLS
https://blog.pcisecuritystandards.org/are-you-ready-for-30-june-2018-sayin-goodbye-to-ssl-early-tls

PCI 3.1 and TLS 1.2 (Cloudflare Support)
https://support.cloudflare.com/hc/en-us/articles/205043158-PCI-3-1-and-TLS-1-2

Application error messages

This alert requires manual confirmation

Acunetix found one or more error/warning messages. Application error or warning messages may expose sensitive information about an application's internal workings to an attacker.
These messages may also contain the location of the file that produced an unhandled exception.
Consult the 'Attack details' section for more information about the affected page(s).

Impact

Error messages may disclose sensitive information which can be used to escalate attacks.

---

https://www.anorc.org/

Application error messages:

- https://www.anorc.org/Shop/Ajax_functionsShop.asp
  Microsoft OLE DB Provider for ODBC Drivers

- https://www.anorc.org/Shop/Ajax_functionsShop.asp
  ODBC SQL Server Driver

- https://www.anorc.org/Shop/Ajax_functionsShop.asp
  ODBC Driver

- https://www.anorc.org/Shop/Ajax_functionsShop.asp
  ODBC SQL

- https://www.anorc.org/Shop/Ajax_functionsShop.asp
  ODBC SQL Server

- https://www.anorc.org/Shop/Ajax_functionsShop.asp
  Invalid column name

- https://www.anorc.org/inc/Ajax_functions.asp
  Microsoft VBScript runtime </font> <font face="Arial" size=2>error '800a000d'</font>

- https://www.anorc.org/inc/submit.asp
  Microsoft VBScript runtime </font> <font face="Arial" size=2>error '800a000d'</font>

Request

```
POST /Shop/Ajax_functionsShop.asp?p=LikePro HTTP/1.1
Host: www.anorc.org
```

```
Content-Length: 36
accept: */*
accept-language: en-US
content-type: application/x-www-form-urlencoded; charset=UTF-8
origin: https://www.anorc.org
cookie: ASPSESSIONIDQEAQCTQR=LBNPCCPBGBGJJCDIOMFLCGFJ; theme=1; Lang=fa
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Accept-Encoding: gzip,deflate,br
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4512.0 Safari/537.36

ProID=productIDVal&UserID=userIDVal&
```

Recommendation

Verify that these page(s) are disclosing error or warning messages and properly configure the application to log errors to a file instead of displaying the error to the user.

Description

While information disclosure vulnerabilities are not directly exploitable by an attacker, they may help an attacker to learn about system specific information. The following is a list of **some** of the information an attacker may be able to obtain from application error disclosure.

Internal IP addresses
Secrets (passwords, keys, tokens...)
Operating system distributions
Software version numbers
Missing security patches
Application stack traces
SQL statements
Location of sensitive files (backups, temporary files...)
Location of sensitive resources (databases, caches, code repositories...)

References

PHP Runtime Configuration
https://www.php.net/manual/en/errorfunc.configuration.php#ini.display-errors

Improper Error Handling
https://www.owasp.org/index.php/Improper_Error_Handling

TLS 1.1 enabled

The web server supports encryption through TLS 1.1, which was formally deprecated in March 2021 as a result of inherent security issues. When aiming for Payment Card Industry (PCI) Data Security Standard (DSS) compliance, it is recommended to use TLS 1.2 or higher instead. According to PCI, "30 June 2018 is the deadline for disabling SSL/early TLS and implementing a more secure encryption protocol – TLS 1.1 or higher (TLS v1.2 is strongly encouraged) in order to meet the PCI Data Security Standard (PCI DSS) for safeguarding payment data.

Impact

An attacker may be able to exploit this problem to conduct man-in-the-middle attacks and decrypt communications between the affected service and clients.

---

https://www.anorc.org/ Confidence: 100%

The SSL server (port: 443) encrypts traffic using TLSv1.1.

Recommendation

It is recommended to disable TLS 1.1 and replace it with TLS 1.2 or higher.

References

RFC 8996: Deprecating TLS 1.0 and TLS 1.1
https://tools.ietf.org/html/rfc8996

Are You Ready for 30 June 2018? Saying Goodbye to SSL/early TLS
https://blog.pcisecuritystandards.org/are-you-ready-for-30-june-2018-sayin-goodbye-to-ssl-early-tls

PCI 3.1 and TLS 1.2 (Cloudflare Support)
https://support.cloudflare.com/hc/en-us/articles/205043158-PCI-3-1-and-TLS-1-2

TLS/SSL LOGJAM attack

The LOGJAM attack is a SSL/TLS vulnerability that allows attackers to intercept HTTPS connections between vulnerable clients and servers and force them to use 'export-grade' cryptography, which can then be decrypted or altered. This vulnerability alert is issued when a web site is found to support DH(E) export cipher suites, or non-export DHE cipher suites using either DH primes smaller than 1024 bits, or commonly used DH standard primes up to 1024 bits.

Impact

An attacker may intercept HTTPS connections between vulnerable clients and servers.

---

https://www.anorc.org/

Weak DH Key Parameters (p < 1024 bits, or <= 1024 bits for common primes):

- TLS1.0, TLS_DHE_RSA_WITH_AES_256_CBC_SHA: 1024 bits (common prime)
- TLS1.0, TLS_DHE_RSA_WITH_AES_128_CBC_SHA: 1024 bits (common prime)
- TLS1.1, TLS_DHE_RSA_WITH_AES_256_CBC_SHA: 1024 bits (common prime)
- TLS1.1, TLS_DHE_RSA_WITH_AES_128_CBC_SHA: 1024 bits (common prime)
- TLS1.2, TLS_DHE_RSA_WITH_AES_256_GCM_SHA384: 1024 bits (common prime)
- TLS1.2, TLS_DHE_RSA_WITH_AES_128_GCM_SHA256: 1024 bits (common prime)
- TLS1.2, TLS_DHE_RSA_WITH_AES_256_CBC_SHA: 1024 bits (common prime)
- TLS1.2, TLS_DHE_RSA_WITH_AES_128_CBC_SHA: 1024 bits (common prime)

Recommendation

Reconfigure the affected SSL/TLS server to disable support for any DHE_EXPORT suites, for DH primes smaller than 1024 bits, and for DH standard primes up to 1024 bits. Refer to the "Guide to Deploying Diffie-Hellman for TLS" for further guidance on how to configure affected systems accordingly.

References

Weak Diffie-Hellman and the Logjam Attack
https://weakdh.org/

Guide to Deploying Diffie-Hellman for TLS
https://weakdh.org/sysadmin.html

TLS/SSL Sweet32 attack

The Sweet32 attack is a SSL/TLS vulnerability that allows attackers to compromise HTTPS connections using 64-bit block ciphers.

Impact

An attacker may intercept HTTPS connections between vulnerable clients and servers.

https://www.anorc.org/

Cipher suites susceptible to Sweet32 attack (TLS1.0 on port 443):

- TLS_RSA_WITH_3DES_EDE_CBC_SHA

Cipher suites susceptible to Sweet32 attack (TLS1.1 on port 443):

- TLS_RSA_WITH_3DES_EDE_CBC_SHA

Cipher suites susceptible to Sweet32 attack (TLS1.2 on port 443):

- TLS_RSA_WITH_3DES_EDE_CBC_SHA

Recommendation

Reconfigure the affected SSL/TLS server to disable support for obsolete 64-bit block ciphers.

References

Sweet32: Birthday attacks on 64-bit block ciphers in TLS and OpenVPN
https://sweet32.info/

TLS/SSL Weak Cipher Suites

The remote host supports TLS/SSL cipher suites with weak or insecure properties.

Impact

---

https://www.anorc.org/

Weak TLS/SSL Cipher Suites: (offered via TLS1.0 on port 443):

- TLS_RSA_WITH_3DES_EDE_CBC_SHA (Medium strength encryption algorithm (3DES).)
- TLS_RSA_WITH_RC4_128_SHA (Weak encryption algorithm (RC4).)
- TLS_RSA_WITH_RC4_128_MD5 (Weak encryption algorithm (RC4). MD5-HMAC.)

Weak TLS/SSL Cipher Suites: (offered via TLS1.1 on port 443):

- TLS_RSA_WITH_3DES_EDE_CBC_SHA (Medium strength encryption algorithm (3DES).)
- TLS_RSA_WITH_RC4_128_SHA (Weak encryption algorithm (RC4).)
- TLS_RSA_WITH_RC4_128_MD5 (Weak encryption algorithm (RC4). MD5-HMAC.)

Weak TLS/SSL Cipher Suites: (offered via TLS1.2 on port 443):

- TLS_RSA_WITH_3DES_EDE_CBC_SHA (Medium strength encryption algorithm (3DES).)
- TLS_RSA_WITH_RC4_128_SHA (Weak encryption algorithm (RC4).)
- TLS_RSA_WITH_RC4_128_MD5 (Weak encryption algorithm (RC4). MD5-HMAC.)

Recommendation

Reconfigure the affected application to avoid use of weak cipher suites.

References

OWASP: TLS Cipher String Cheat Sheet
https://cheatsheetseries.owasp.org/cheatsheets/TLS_Cipher_String_Cheat_Sheet.html

OWASP: Transport Layer Protection Cheat Sheet
https://cheatsheetseries.owasp.org/cheatsheets/Transport_Layer_Protection_Cheat_Sheet.html

Mozilla: TLS Cipher Suite Recommendations
https://wiki.mozilla.org/Security/Server_Side_TLS

SSLlabs: SSL and TLS Deployment Best Practices
https://github.com/ssllabs/research/wiki/SSL-and-TLS-Deployment-Best-Practices

URL redirection

This script is possibly vulnerable to URL redirection attacks.

URL redirection is sometimes used as a part of phishing attacks that confuse visitors about which web site they are visiting.

Impact

A remote attacker can redirect users from your website to a specified URL. This problem may assist an attacker to conduct phishing attacks, trojan distribution, spammers.

https://www.anorc.org/inc/submit.asp

URL encoded GET input **l** was set to **/xfs.bxss.me**

Request

```
POST /inc/submit.asp?id=86&l=/xfs.bxss.me&module=2&p=comment HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Cookie: ASPSESSIONIDQEAQCTQR=KBNPCCPBPNGDAEEEIJFJEDLP; theme=1; Lang=fa
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
Content-Length: 314
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4512.0 Safari/537.36
Host: www.anorc.org
Connection: Keep-alive

CAPTCHA_Postback=true&email=%D8%A7%DB%8C%D9%85%DB%8C%D9%84%20%D8%B4%D9%85%D8%A7%20(%D9%86%D9%85%D8%A7%
DB%8C%D8%B4%20%D8%AF%D8%A7%D8%AF%D9%87%20%D9%86%D9%85%DB%8C%D8%B4%D9%88%D8%AF)&fname=%D9%86%D8%A7%D9%8
5%20%D8%B4%D9%85%D8%A7&securityCode=94102&txtComment=555&website=%D8%B3%D8%A7%DB%8C%D8%AA%20%D8%B4%D9%
85%D8%A7
```

Recommendation

Your script should properly sanitize user input.

References

Unvalidated Redirects and Forwards Cheat Sheet
https://cheatsheetseries.owasp.org/cheatsheets/Unvalidated_Redirects_and_Forwards_Cheat_Sheet.html

HTTP Response Splitting, Web Cache Poisoning Attacks, and Related Topics
https://packetstormsecurity.com/papers/general/whitepaper_httpresponse.pdf

Clickjacking: CSP frame-ancestors missing

Clickjacking (User Interface redress attack, UI redress attack, UI redressing) is a malicious technique of tricking a Web user into clicking on something different from what the user perceives they are clicking on, thus potentially revealing confidential information or taking control of their computer while clicking on seemingly innocuous web pages.

The server didn't return a **frame-ancestors** directive in the Content-Security-Policy header which means that this website could be at risk of a clickjacking attack. The frame-ancestors directives can be used to indicate whether or not a browser should be allowed to render a page inside a frame. Sites can use this to avoid clickjacking attacks, by ensuring that their content is not embedded into other sites.

Impact

The impact depends on the affected web application.

https://www.anorc.org/

Paths without CSP frame-ancestors:

- https://www.anorc.org/fa/index.asp
- https://www.anorc.org/fa/news2/6
- https://www.anorc.org/fa/newsview/106
- https://www.anorc.org/fa/pages/105
- https://www.anorc.org/fa/newsview/107
- https://www.anorc.org/fa/newsview/108
- https://www.anorc.org/fa/pages/28
- https://www.anorc.org/fa/newsview/109
- https://www.anorc.org/fa/pages/29
- https://www.anorc.org/fa/newsview/113
- https://www.anorc.org/fa/pages/30
- https://www.anorc.org/fa/newsview/114
- https://www.anorc.org/fa/pages/36
- https://www.anorc.org/fa/newsview/115
- https://www.anorc.org/fa/pages/40
- https://www.anorc.org/fa/newsview/119
- https://www.anorc.org/fa/pages/41
- https://www.anorc.org/fa/newsview/121
- https://www.anorc.org/fa/pages/42
- https://www.anorc.org/fa/newsview/122
- https://www.anorc.org/fa/pages/43

Request

```
GET /fa/index.asp?p=search&search=the HTTP/1.1
Referer: https://www.anorc.org/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4512.0 Safari/537.36
Host: www.anorc.org
Connection: Keep-alive
```

Recommendation

Configure your web server to include a CSP header with frame-ancestors directive and an X-Frame-Options header. Consult Web references for more information about the possible values for this header.

References

OWASP Clickjacking
https://cheatsheetseries.owasp.org/cheatsheets/Clickjacking_Defense_Cheat_Sheet.html

CSP: frame-ancestors
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Security-Policy/frame-ancestors

The X-Frame-Options response header
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options

Cookies with missing, inconsistent or contradictory properties

At least one of the following cookies properties causes the cookie to be invalid or incompatible with either a different property of the same cookie, of with the environment the cookie is being used in. Although this is not a vulnerability in itself, it will likely lead to unexpected behavior by the application, which in turn may cause secondary security issues.

Impact

Cookies will not be stored, or submitted, by web browsers.

---

https://www.anorc.org/ Verified

List of cookies with missing, inconsistent or contradictory properties:

- https://www.anorc.org/fa/index.asp

  Cookie was set with:

```
Set-Cookie: theme=1; path=/
```

This cookie has the following issues:

```
- Cookie without SameSite attribute.
When cookies lack the SameSite attribute, Web browsers may apply different and
sometimes unexpected defaults. It is therefore recommended to add a SameSite
attribute with an appropriate value of either "Strict", "Lax", or "None".
```

- https://www.anorc.org/fa/index.asp

  Cookie was set with:

  ```
  Set-Cookie: Lang=fa; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/fa/index.asp

  Cookie was set with:

  ```
  Set-Cookie: ASPSESSIONIDQEAQCTQR=KBNPCCPBPNGDAEEEIJFJEDLP; secure; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/inc/submit.asp

  Cookie was set with:

  ```
  Set-Cookie: ASPSESSIONIDQEAQCTQR=OBNPCCPBFMOIDBOEMNPDNPDE; secure; path=/
  ```

  This cookie has the following issues:

```
- Cookie without SameSite attribute.
When cookies lack the SameSite attribute, Web browsers may apply different and
sometimes unexpected defaults. It is therefore recommended to add a SameSite
attribute with an appropriate value of either "Strict", "Lax", or "None".
```

- https://www.anorc.org/fa/news2/6

  Cookie was set with:

  ```
  Set-Cookie: Lang=fa; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/fa/news2/6

  Cookie was set with:

  ```
  Set-Cookie: theme=1; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/fa/newsview/106

  Cookie was set with:

  ```
  Set-Cookie: Lang=fa; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/fa/newsview/106

  Cookie was set with:

  ```
  Set-Cookie: theme=1; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/fa/pages/105

  Cookie was set with:

  ```
  Set-Cookie: Lang=fa; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/fa/pages/105

  Cookie was set with:

  ```
  Set-Cookie: theme=1; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/fa/newsview/107

  Cookie was set with:

```
Set-Cookie: Lang=fa; path=/
```

This cookie has the following issues:

```
- Cookie without SameSite attribute.
When cookies lack the SameSite attribute, Web browsers may apply different and
sometimes unexpected defaults. It is therefore recommended to add a SameSite
attribute with an appropriate value of either "Strict", "Lax", or "None".
```

- https://www.anorc.org/fa/newsview/107

  Cookie was set with:

  ```
  Set-Cookie: theme=1; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/fa/newsview/108

  Cookie was set with:

  ```
  Set-Cookie: Lang=fa; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/fa/newsview/108

  Cookie was set with:

  ```
  Set-Cookie: theme=1; path=/
  ```

  This cookie has the following issues:

```
- Cookie without SameSite attribute.
When cookies lack the SameSite attribute, Web browsers may apply different and
sometimes unexpected defaults. It is therefore recommended to add a SameSite
attribute with an appropriate value of either "Strict", "Lax", or "None".
```

- https://www.anorc.org/fa/pages/28

  Cookie was set with:

  ```
  Set-Cookie: Lang=fa; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/fa/pages/28

  Cookie was set with:

  ```
  Set-Cookie: theme=1; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/fa/newsview/109

  Cookie was set with:

  ```
  Set-Cookie: Lang=fa; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/fa/newsview/109

  Cookie was set with:

  ```
  Set-Cookie: theme=1; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/fa/pages/29

  Cookie was set with:

  ```
  Set-Cookie: Lang=fa; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/fa/pages/29

  Cookie was set with:

  ```
  Set-Cookie: theme=1; path=/
  ```

  This cookie has the following issues:

  ```
  - Cookie without SameSite attribute.
  When cookies lack the SameSite attribute, Web browsers may apply different and
  sometimes unexpected defaults. It is therefore recommended to add a SameSite
  attribute with an appropriate value of either "Strict", "Lax", or "None".
  ```

- https://www.anorc.org/fa/newsview/113

  Cookie was set with:

```
Set-Cookie: Lang=fa; path=/
```

This cookie has the following issues:

```
- Cookie without SameSite attribute.
When cookies lack the SameSite attribute, Web browsers may apply different and
sometimes unexpected defaults. It is therefore recommended to add a SameSite
attribute with an appropriate value of either "Strict", "Lax", or "None".
```

Request

```
GET /fa/index.asp?p=search&search=the HTTP/1.1
Referer: https://www.anorc.org/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4512.0 Safari/537.36
Host: www.anorc.org
Connection: Keep-alive
```

Recommendation

Ensure that the cookies configuration complies with the applicable standards.

References

MDN | Set-Cookie
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie

Securing cookies with cookie prefixes
https://www.sjoerdlangkemper.nl/2017/02/09/cookie-prefixes/

Cookies: HTTP State Management Mechanism
https://tools.ietf.org/html/draft-ietf-httpbis-rfc6265bis-05

SameSite Updates - The Chromium Projects
https://www.chromium.org/updates/same-site

draft-west-first-party-cookies-07: Same-site Cookies
https://tools.ietf.org/html/draft-west-first-party-cookies-07

Cookies without HttpOnly flag set

One or more cookies don't have the HttpOnly flag set. When a cookie is set with the HttpOnly flag, it instructs the browser that the cookie can only be accessed by the server and not by client-side scripts. This is an important security protection for session cookies.

Impact

Cookies can be accessed by client-side scripts.

---

https://www.anorc.org/ Verified

Cookies without HttpOnly flag set:

- https://www.anorc.org/fa/index.asp

```
Set-Cookie: theme=1; path=/
```

- https://www.anorc.org/fa/index.asp

```
Set-Cookie: Lang=fa; path=/
```

- https://www.anorc.org/fa/index.asp

```
Set-Cookie: ASPSESSIONIDQEAQCTQR=KBNPCCPBPNGDAEEEIJFJEDLP; secure; path=/
```

- https://www.anorc.org/inc/submit.asp

```
Set-Cookie: ASPSESSIONIDQEAQCTQR=OBNPCCPBFMOIDBOEMNPDNPDE; secure; path=/
```

- https://www.anorc.org/fa/news2/6

```
Set-Cookie: Lang=fa; path=/
```

- https://www.anorc.org/fa/news2/6

```
Set-Cookie: theme=1; path=/
```

- https://www.anorc.org/fa/newsview/106

```
Set-Cookie: Lang=fa; path=/
```

- https://www.anorc.org/fa/newsview/106

```
Set-Cookie: theme=1; path=/
```

- https://www.anorc.org/fa/pages/105

```
Set-Cookie: Lang=fa; path=/
```

- https://www.anorc.org/fa/pages/105

```
Set-Cookie: theme=1; path=/
```

- https://www.anorc.org/fa/newsview/107

```
Set-Cookie: Lang=fa; path=/
```

- https://www.anorc.org/fa/newsview/107

```
Set-Cookie: theme=1; path=/
```

- https://www.anorc.org/fa/newsview/108

```
Set-Cookie: Lang=fa; path=/
```

- https://www.anorc.org/fa/newsview/108

```
Set-Cookie: theme=1; path=/
```

- https://www.anorc.org/fa/pages/28

```
Set-Cookie: Lang=fa; path=/
```

- https://www.anorc.org/fa/pages/28

  ```
  Set-Cookie: theme=1; path=/
  ```

- https://www.anorc.org/fa/newsview/109

  ```
  Set-Cookie: Lang=fa; path=/
  ```

- https://www.anorc.org/fa/newsview/109

  ```
  Set-Cookie: theme=1; path=/
  ```

- https://www.anorc.org/fa/pages/29

  ```
  Set-Cookie: Lang=fa; path=/
  ```

- https://www.anorc.org/fa/pages/29

  ```
  Set-Cookie: theme=1; path=/
  ```

- https://www.anorc.org/fa/newsview/113

  ```
  Set-Cookie: Lang=fa; path=/
  ```

Request

```
GET /fa/index.asp?p=search&search=the HTTP/1.1
Referer: https://www.anorc.org/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4512.0 Safari/537.36
Host: www.anorc.org
Connection: Keep-alive
```

Recommendation

If possible, you should set the HttpOnly flag for these cookies.

Cookies without Secure flag set

One or more cookies does not have the Secure flag set. When a cookie is set with the Secure flag, it instructs the browser that the cookie can only be accessed over secure SSL/TLS channels. This is an important security protection for session cookies.

Impact

Cookies could be sent over unencrypted channels.

---

https://www.anorc.org/ Verified

Cookies without Secure flag set:

- https://www.anorc.org/fa/index.asp

  ```
  Set-Cookie: theme=1; path=/
  ```

- https://www.anorc.org/fa/index.asp

  ```
  Set-Cookie: Lang=fa; path=/
  ```

- https://www.anorc.org/fa/news2/6

  ```
  Set-Cookie: Lang=fa; path=/
  ```

- https://www.anorc.org/fa/news2/6

  ```
  Set-Cookie: theme=1; path=/
  ```

- https://www.anorc.org/fa/newsview/106

  ```
  Set-Cookie: Lang=fa; path=/
  ```

- https://www.anorc.org/fa/newsview/106

```
Set-Cookie: theme=1; path=/
```

- https://www.anorc.org/fa/pages/105

```
Set-Cookie: Lang=fa; path=/
```

- https://www.anorc.org/fa/pages/105

```
Set-Cookie: theme=1; path=/
```

- https://www.anorc.org/fa/newsview/107

```
Set-Cookie: Lang=fa; path=/
```

- https://www.anorc.org/fa/newsview/107

```
Set-Cookie: theme=1; path=/
```

- https://www.anorc.org/fa/newsview/108

```
Set-Cookie: Lang=fa; path=/
```

- https://www.anorc.org/fa/newsview/108

```
Set-Cookie: theme=1; path=/
```

- https://www.anorc.org/fa/pages/28

```
Set-Cookie: Lang=fa; path=/
```

- https://www.anorc.org/fa/pages/28

```
Set-Cookie: theme=1; path=/
```

- https://www.anorc.org/fa/newsview/109

```
Set-Cookie: Lang=fa; path=/
```

- https://www.anorc.org/fa/newsview/109

```
Set-Cookie: theme=1; path=/
```

- https://www.anorc.org/fa/pages/29

```
Set-Cookie: Lang=fa; path=/
```

- https://www.anorc.org/fa/pages/29

```
Set-Cookie: theme=1; path=/
```

- https://www.anorc.org/fa/newsview/113

```
Set-Cookie: Lang=fa; path=/
```

- https://www.anorc.org/fa/newsview/113

```
Set-Cookie: theme=1; path=/
```

- https://www.anorc.org/fa/pages/30

```
Set-Cookie: Lang=fa; path=/
```

Request

```
GET /fa/index.asp?p=search&search=the HTTP/1.1
Referer: https://www.anorc.org/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
```

```
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4512.0 Safari/537.36
Host: www.anorc.org
Connection: Keep-alive
```

Recommendation

If possible, you should set the Secure flag for these cookies.

Microsoft IIS version disclosure

The HTTP responses returned by this web application include a header named **Server**. The value of this header includes the version of Microsoft IIS server.

Impact

The HTTP header may disclose sensitive information. This information can be used to launch further attacks.

---

https://www.anorc.org/

Version information found:

```
Microsoft-IIS/8.5
```

Request

```
GET /|~.aspx HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4512.0 Safari/537.36
Host: www.anorc.org
Connection: Keep-alive
```

Recommendation

Microsoft IIS should be configured to remove unwanted HTTP response headers from the response. Consult web references for more information.

References

Remove Unwanted HTTP Response Headers

https://blogs.msdn.microsoft.com/varunm/2013/04/23/remove-unwanted-http-response-headers/

Subresource Integrity (SRI) not implemented

Subresource Integrity (SRI) is a security feature that enables browsers to verify that third-party resources they fetch (for example, from a CDN) are delivered without unexpected manipulation. It works by allowing developers to provide a cryptographic hash that a fetched file must match.

Third-party resources (such as scripts and stylesheets) can be manipulated. An attacker that has access or has hacked the hosting CDN can manipulate or replace the files. SRI allows developers to specify a base64-encoded cryptographic hash of the resource to be loaded. The integrity attribute containing the hash is then added to the <script> HTML element tag. The integrity string consists of a base64-encoded hash, followed by a prefix that depends on the hash algorithm. This prefix can either be sha256, sha384 or sha512.

The script loaded from the external URL specified in the Details section doesn't implement Subresource Integrity (SRI). It's recommended to implement Subresource Integrity (SRI) for all the scripts loaded from external hosts.

Impact

An attacker that has access or has hacked the hosting CDN can manipulate or replace the files.

https://www.anorc.org/fa/index.asp

Pages where SRI is not implemented:

- https://www.anorc.org/fa/index.asp
  Script SRC: **https://www.oil-price.net/TINY_CHART/gen.php?lang=en&center**

Request

```
GET /fa/index.asp?p=search&search=the HTTP/1.1
Referer: https://www.anorc.org/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4512.0 Safari/537.36
Host: www.anorc.org
Connection: Keep-alive
```

Recommendation

Use the SRI Hash Generator link (from the References section) to generate a <script> element that implements Subresource Integrity (SRI).

For example, you can use the following <script> element to tell a browser that before executing the https://example.com/example-framework.js script, the browser must first compare the script to the expected hash, and verify that there's a match.

```
<script src="https://example.com/example-framework.js"
integrity="sha384-oqVuAfXRKap7fdgcCY5uykM6+R9GqQ8K/uxy9rx7HNQlGYl1kPzQho1wx4JwY8wC"
crossorigin="anonymous"></script>
```

References

Subresource Integrity
https://developer.mozilla.org/en-US/docs/Web/Security/Subresource_Integrity

SRI Hash Generator
https://www.srihash.org/

TLS/SSL (EC)DHE Key Reuse

The remote host reuses Diffie-Hellman Ephemeral public server keys with (EC)DHE cipher suites.

Impact

https://www.anorc.org/

Diffie-Hellman Public Key Reuse:

- DHE public server key reuse: 97 d2 cd a1 3b 7a 1a 74 c8 92 bd 28 25 47 42 5b 90 80 e0 4a bf 07 5c fd 0b 45 31 56 51 f7 7a 3f 9d c0 ee 86 2f a6 a8 fe 6f 1a d8 56 f9 a2 8b 68 5a f7 4c e6 07 92 31 39 3b 64 9a 36 4f a7 28 ff b6 5b 1a f1 7c 0a 78 46 15 f9 56 dc b6 16 69 0a f6 11 16 9b 6f 52 bf 80 74 02 25 df d8 7b 6c 26 c0 cb 26 30 ca b5 22 8e 3a ec f1 29 8f 0d 71 ef a9 59 b6 71 95 98 40 d4 7a 8e 8a 4a 4d 04 93 25 (with TLS_DHE_RSA_WITH_AES_256_CBC_SHA)
- ECDHE public server key reuse: 04 7a f0 46 7d 0e 32 7e f1 93 2c 3c 01 45 8d 8f 3c 84 40 3c 0b 66 3d 5e 0d 21 34 1b 41 37 46 6a a9 bf 27 47 1a 1e 8d 7b 2a 4c 0c 09 de 49 c1 63 2a 34 6b 59 60 ec eb 11 90 b1 37 36 90 5b 4e 2d 7b (with TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA)

Recommendation

Reconfigure the affected application to always generate new keys when using tmp_dh/tmp_ecdh parameters.

References

Raccoon Attack
https://raccoon-attack.com/

Raccoon Attack (Technical Paper, PDF)
https://raccoon-attack.com/RacoonAttack.pdf

Logjam Attack
https://weakdh.org/

Logjam Attack (Technical Paper, PDF)
https://weakdh.org/imperfect-forward-secrecy-ccs15.pdf

List of SSL OP Flags (see: SSL_OP_SINGLE_DH_USE, SSL_OP_SINGLE_ECDH_USE)
https://wiki.openssl.org/index.php/List_of_SSL_OP_Flags

Web Application Firewall detected

This server is protected by an IPS (Intrusion Prevention System), IDS (Intrusion Detection System) or an WAF (Web Application Firewall). Acunetix detected this by sending various malicious payloads and detecting changes in the response code, headers and body.

Impact

You may receive incorrect/incomplete results when scanning a server protected by an IPS/IDS/WAF. Also, if the WAF detects a number of attacks coming from the scanner, the IP address can be blocked after a few attempts.

https://www.anorc.org/

Detected ASP.NET URLScan from the response body.

Request

```
GET /9664933 HTTP/1.1
Cookie: ASPSESSIONIDQEAQCTQR=KBNPCCPBPNGDAEEEIJFJEDLP; theme=1; Lang=fa
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4512.0 Safari/537.36
Host: www.anorc.org
Connection: Keep-alive
```

Recommendation

If possible, it's recommended to scan an internal (development) version of the web application where the WAF is not active.

Coverage

- https://www.anorc.org
  - assets
    - css
      - bootstrap-5
        - bootstrap.min.css
      - animate.min.css
      - box.css
      - datepicker.css
      - fa-window.css
      - Flickity.css
      - glyphicon.css
      - Hover.css
      - menu-top.css
      - normalize.css
      - responsive.css
      - site-all.css
        - #fragments
          - threshold
    - font
      - fontawesome-5.14.0
        - webfonts
    - fonts
      - glyphicons
    - js
      - bootstrap.min.js
      - core.js
      - jquery.magnific-popup.min.js
      - popper.min.js
      - vanilla-tilt.min.js
      - wow.js
    - slider
      - amazingslider

    - images
- vendor
  - css
    - samples.css
  - icon
  - img
  - js
    - sf.js
  - lightbox
    - #fragments
      - advanced
      - help-content
    - css
      - fontello.css
    - font
    - js
      - abstracttoolbarmodifier.js
      - fulltoolbareditor.js
      - toolbarmodifier.js
      - toolbartextmodifier.js
    - lib
      - codemirror
        - codemirror.css
        - codemirror.js
        - javascript.js
        - neo.css
        - show-hint.css
        - show-hint.js
    - v7.5
      - froogaloop2.min.js
    - index.html
      - #fragments
        - advanced
        - help-content

```
        index.html
    ckeditor.js
    config.js
CAPTCHA
    cap.js
    CAPTCHA_image.asp
core
    slider
        js
            amazingslider
                amazingslider.min.js
                Effect4.js
    social
        social-sticky.asp
en
    pages
        55
fa
    FormView
        4
    news2
        16
        17
        4
        6
    newsview
        101
        103
        104
        105
        106
        107
        108
        109
```

110

113

114

115

116

118

119

121

122

124

126

127

129

130

131

132

134

137

139

144

146

147

149

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

170

172

174

176

177

180

182

185

187

188

190

192

194

196

200

201

202

204

205

206

207

209

211

214

216

217

220

223

229

230

231

234

237

238

239

243

244

245

246

247

248

251

255

256

257

261

262

263

264

265

266

267

269

270

274

275

276

277

36

37

4

43

44

45

46

47

48

52

54

55

56

58

59

60

63

64

65

66

68

69

70

71

74

75

76

77

78

79

80

81

85

86

88

89

90

91

92

93

- 94
- 95
- 96
- 97
- 98
- 99

pages
- 1
- 100
- 105
- 2
- 28
- 29
- 30
- 36
- 40
- 41
- 42
- 43
- 48
- 88
- 89
- 90
- 94
- 96
- 97
- 99

Contact

GalleryAlbum

index.asp
- Inputs
  - GET p, search

font
- IranSans

- v5
  - style.css
- inc
  - Ajax_functions.asp
    - Inputs
      - POST p
      - POST , CommentID, ModuleID
  - submit.asp
    - Inputs
      - POST id, l, module, p
      - POST CAPTCHA_Postback, email, fname, securityCode, txtComment, website
- Shop
  - Ajax_functionsShop.asp
    - Inputs
      - POST p
      - POST , ProID, UserID
- thumbs
  - lg
  - md
- userfiles
  - files
  - images
    - menue
    - monasebat
    - site
- 9664933
- robots.txt
- sanadata
  - Inputs
    - GET Login
- script.asp
- service-worker.js
- sitemap.xml
- style.css