Web Application Security Report

Sites: https://testphp.vulnweb.com https://data-edge.smartscreen.microsoft.com https://telem-edge.smartscreen.microsoft.com https://nav-edge.smartscreen.microsoft.com http://testphp.vulnweb.com

Generated on Wed, 24 Dec 2025 14:08:11

ZAP Version: 2.17.0

ZAP by Checkmarx

Summary of Alerts

Risk Level	Number of Alerts
High	4
Medium	7
Low	5
Informational	6

Insights

Level	Reason	Site	Description	Statistic
Low	Warning		ZAP errors logged - see the zap.log file for details	58
Low	Warning		ZAP warnings logged - see the zap.log file for details	48
Low	Exceeded High	http://testphp.vulnweb.com	Percentage of slow responses	100 %
Info	Informational	http://testphp.vulnweb.com	Percentage of responses with status code 2xx	90 %
Info	Informational	http://testphp.vulnweb.com	Percentage of responses with status code 3xx	3 %
Info	Exceeded Low	http://testphp.vulnweb.com	Percentage of responses with status code 4xx	6 %
Info	Informational	http://testphp.vulnweb.com	Percentage of endpoints with content type application/octet-stream	1 %
Info	Informational	http://testphp.vulnweb.com	Percentage of endpoints with content type application/x-shockwave-flash	1 %
Info	Informational	http://testphp.vulnweb.com	Percentage of endpoints with content type image/gif	2 %

Info	Informational	http://testphp.vulnweb.com	Percentage of endpoints with content type image/jpeg	6 %
Info	Informational	http://testphp.vulnweb.com	Percentage of endpoints with content type image/x-icon	1 %
Info	Informational	http://testphp.vulnweb.com	Percentage of endpoints with content type text/css	3 %
Info	Informational	http://testphp.vulnweb.com	Percentage of endpoints with content type text/html	83 %
Info	Informational	http://testphp.vulnweb.com	Percentage of endpoints with method GET	93 %
Info	Informational	http://testphp.vulnweb.com	Percentage of endpoints with method POST	6 %
Info	Informational	http://testphp.vulnweb.com	Count of total endpoints	78
Info	Informational	https://data-edge.smartscreen.microsoft.com	Percentage of responses with status code 2xx	100 %
Info	Informational	https://data-edge.smartscreen.microsoft.com	Percentage of endpoints with content type application/octet-stream	100 %
Info	Informational	https://data-edge.smartscreen.microsoft.com	Percentage of endpoints with method POST	100 %
Info	Informational	https://data-edge.smartscreen.microsoft.com	Count of total endpoints	2
Info	Informational	https://data-edge.smartscreen.microsoft.com	Percentage of slow responses	100 %
Info	Informational	https://nav-edge.smartscreen.microsoft.com	Percentage of responses with status code 2xx	100 %
Info	Informational	https://nav-edge.smartscreen.microsoft.com	Percentage of endpoints with content type application/json	100 %
Info	Informational	https://nav-edge.smartscreen.microsoft.com	Percentage of endpoints with method POST	100 %
Info	Informational	https://nav-edge.smartscreen.microsoft.com	Count of total endpoints	1
Info	Informational	https://nav-edge.smartscreen.microsoft.com	Percentage of slow responses	100 %
Info	Informational	https://telem-edge.smartscreen.microsoft.com	Percentage of responses with status code 2xx	100 %
Info	Informational	https://telem-edge.smartscreen.microsoft.com	Percentage of endpoints with method POST	100 %
Info	Informational	https://telem-edge.smartscreen.microsoft.com	Count of total endpoints	1
Info	Informational	https://telem-edge.smartscreen.microsoft.com	Percentage of slow responses	100 %
Info	Informational	https://testphp.vulnweb.com	Percentage of endpoints with method GET	100 %
Info	Informational	https://testphp.vulnweb.com	Count of total endpoints	1

Summary of Sequences

For each step: result (Pass/Fail) - risk (of highest alert(s) for the step, if any).

Alerts

Name	Risk Level	Number of Instances
Cross Site Scripting (Reflected)	High	20
SQL Injection - MySQL	High	10
SQL Injection - MySQL (Time Based)	High	6
SQL Injection - SQLite (Time Based)	High	2
Absence of Anti-CSRF Tokens	Medium	Systemic
Content Security Policy (CSP) Header Not Set	Medium	Systemic
Directory Browsing	Medium	3
HTTP Only Site	Medium	1
Hidden File Found	Medium	1
Missing Anti-clickjacking Header	Medium	Systemic
XSLT Injection	Medium	2
In Page Banner Information Leak	Low	Systemic
Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)	Low	Systemic
Server Leaks Version Information via "Server" HTTP Response Header Field	Low	Systemic
Strict-Transport-Security Header Not Set	Low	4
X-Content-Type-Options Header Missing	Low	Systemic
Authentication Request Identified	Informational	1
Charset Mismatch (Header Versus Meta Content-Type Charset)	Informational	Systemic
GET for POST	Informational	3
Modern Web Application	Informational	Systemic
User Agent Fuzzer	Informational	Systemic
User Controllable HTML Element Attribute (Potential XSS)	Informational	2

Alert Detail

High	Cross Site Scripting (Reflected)
Description	Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied code into a user's browser instance. A browser instance can be a standard web browser client, or a browser object embedded in a software product such as the browser within WinAmp, an RSS reader, or an email client. The code itself is usually written in HTML/JavaScript, but may also extend to VBScript, ActiveX, Java, Flash, or any other browser-supported technology. When an attacker gets a user's browser to execute his/her code, the code will run within the security context (or zone) of the hosting web site. With this level of privilege, the code has the ability to read, modify and transmit any sensitive data accessible by the browser. A Cross-site Scripted user could have his/her account hijacked (cookie theft), their browser redirected to another location, or possibly shown fraudulent content delivered by the web site they are visiting. Cross-site Scripting attacks essentially compromise the trust relationship between a user and the web site. Applications utilizing browser object instances which load content from the file system may execute code under the local machine zone allowing for system compromise.

There are three types of Cross-site Scripting attacks: non-persistent, persistent and DOM-based.

Non-persistent attacks and DOM-based attacks require a user to either visit a specially crafted link laced with malicious code, or visit a malicious web page containing a web form, which when posted to the vulnerable site, will mount the attack. Using a malicious form will oftentimes take place when the vulnerable resource only accepts HTTP POST requests. In such a case, the form can be submitted automatically, without the victim's knowledge (e.g. by using JavaScript). Upon clicking on the malicious link or submitting the malicious form, the XSS payload will get echoed back and will get interpreted by the user's browser and execute. Another technique to send almost arbitrary requests (GET and POST) is by using an embedded client, such as Adobe Flash.

Persistent attacks occur when the malicious code is submitted to a web site where it's stored for a period of time. Examples of an attacker's favorite targets often include message board posts, web mail messages, and web chat software. The unsuspecting user is not required to interact with any additional site/link (e.g. an attacker site or a malicious link sent via email), just simply view the web page containing the code.

URL	http://testphp.vulnweb.com/artists.php?artist=%3CscrIpt%3Ealert%281%29%3B%3C%2FscRipt%3E
Node Name	http://testphp.vulnweb.com/artists.php (artist)
Method	GET
Parameter	artist
Attack	<scrIpt>alert(1);</scRipt>
Evidence	<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/hpp/?pp=%22+onMouseOver%3D%22alert%281%29%3B
Node Name	http://testphp.vulnweb.com/hpp/ (pp)
Method	GET
Parameter	pp
Attack	" onMouseOver="alert(1);
Evidence	" onMouseOver="alert(1);
Other Info	
URL	http://testphp.vulnweb.com/hpp/params.php?p=%3CscrIpt%3Ealert%281%29%3B%3C%2FscRipt%3E&pp=12
Node Name	http://testphp.vulnweb.com/hpp/params.php (p,pp)
Method	GET
Parameter	p
Attack	<scrIpt>alert(1);</scRipt>
Evidence	<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/hpp/params.php?p=valid&pp=%3CscrIpt%3Ealert%281%29%3B%3C%2FscRipt%3E
Node Name	http://testphp.vulnweb.com/hpp/params.php (p,pp)
Method	GET
Parameter	pp

Attack	<scrIpt>alert(1);</scRipt>
Evidence	<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/listproducts.php?artist=%3CscrIpt%3Ealert%281%29%3B%3C%2FscRipt%3E
Node Name	http://testphp.vulnweb.com/listproducts.php (artist)
Method	GET
Parameter	artist
Attack	<scrIpt>alert(1);</scRipt>
Evidence	<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/listproducts.php?cat=%3CscrIpt%3Ealert%281%29%3B%3C%2FscRipt%3E
Node Name	http://testphp.vulnweb.com/listproducts.php (cat)
Method	GET
Parameter	cat
Attack	<scrIpt>alert(1);</scRipt>
Evidence	<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/product.php?pic=%3CscrIpt%3Ealert%281%29%3B%3C%2FscRipt%3E
Node Name	http://testphp.vulnweb.com/product.php (pic)
Method	GET
Parameter	pic
Attack	<scrIpt>alert(1);</scRipt>
Evidence	<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/search.php?test=%27%22%3CscrIpt%3Ealert%281%29%3B%3C%2FscRipt%3E
Node Name	http://testphp.vulnweb.com/search.php (test)
Method	GET
Parameter	test
Attack	'"<scrIpt>alert(1);</scRipt>
Evidence	'"<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/guestbook.php
Node Name	http://testphp.vulnweb.com/guestbook.php ()(name,submit,text)
Method	POST

Parameter	name
Attack	<scrIpt>alert(1);</scRipt>
Evidence	<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/guestbook.php
Node Name	http://testphp.vulnweb.com/guestbook.php ()(name,submit,text)
Method	POST
Parameter	text
Attack	</td><scrIpt>alert(1);</scRipt><td>
Evidence	</td><scrIpt>alert(1);</scRipt><td>
Other Info	
URL	http://testphp.vulnweb.com/search.php?test=query
Node Name	http://testphp.vulnweb.com/search.php (test)(goButton,searchFor)
Method	POST
Parameter	searchFor
Attack	<scrIpt>alert(1);</scRipt>
Evidence	<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/search.php?test=%27%22%3CscrIpt%3Ealert%281%29%3B%3C%2FscRipt%3E
Node Name	http://testphp.vulnweb.com/search.php (test)(goButton,searchFor)
Method	POST
Parameter	test
Attack	'"<scrIpt>alert(1);</scRipt>
Evidence	'"<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/secured/newuser.php
Node Name	http://testphp.vulnweb.com/secured/newuser.php () (signup,uaddress,ucc,uemail,upass,upass2,uphone,urname,uuname)
Method	POST
Parameter	uaddress
Attack	<scrIpt>alert(1);</scRipt>
Evidence	<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/secured/newuser.php
Node Name	http://testphp.vulnweb.com/secured/newuser.php () (signup,uaddress,ucc,uemail,upass,upass2,uphone,urname,uuname)
Method	POST

Parameter	ucc
Attack	<scrIpt>alert(1);</scRipt>
Evidence	<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/secured/newuser.php
Node Name	http://testphp.vulnweb.com/secured/newuser.php () (signup,uaddress,ucc,uemail,upass,upass2,uphone,urname,uuname)
Method	POST
Parameter	uemail
Attack	<scrIpt>alert(1);</scRipt>
Evidence	<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/secured/newuser.php
Node Name	http://testphp.vulnweb.com/secured/newuser.php () (signup,uaddress,ucc,uemail,upass,upass2,uphone,urname,uuname)
Method	POST
Parameter	uphone
Attack	<scrIpt>alert(1);</scRipt>
Evidence	<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/secured/newuser.php
Node Name	http://testphp.vulnweb.com/secured/newuser.php () (signup,uaddress,ucc,uemail,upass,upass2,uphone,urname,uuname)
Method	POST
Parameter	urname
Attack	<scrIpt>alert(1);</scRipt>
Evidence	<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/secured/newuser.php
Node Name	http://testphp.vulnweb.com/secured/newuser.php () (signup,uaddress,ucc,uemail,upass,upass2,uphone,urname,uuname)
Method	POST
Parameter	uuname
Attack	<scrIpt>alert(1);</scRipt>
Evidence	<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/userinfo.php
Node Name	http://testphp.vulnweb.com/userinfo.php ()(pass,uname)
Method	POST
Parameter	pass

Attack	'"<scrIpt>alert(1);</scRipt>
Evidence	'"<scrIpt>alert(1);</scRipt>
Other Info	
URL	http://testphp.vulnweb.com/userinfo.php
Node Name	http://testphp.vulnweb.com/userinfo.php ()(pass,uname)
Method	POST
Parameter	uname
Attack	'"<scrIpt>alert(1);</scRipt>
Evidence	'"<scrIpt>alert(1);</scRipt>
Other Info	
Instances	20
Solution	Phase: Architecture and Design

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.

Examples of libraries and frameworks that make it easier to generate properly encoded output include Microsoft's Anti-XSS library, the OWASP ESAPI Encoding module, and Apache Wicket.

Phases: Implementation; Architecture and Design

Understand the context in which your data will be used and the encoding that will be expected. This is especially important when transmitting data between different components, or when generating outputs that can contain multiple encodings at the same time, such as web pages or multi-part mail messages. Study all expected communication protocols and data representations to determine the required encoding strategies.

For any data that will be output to another web page, especially any data that was received from external inputs, use the appropriate encoding on all non-alphanumeric characters.

Consult the XSS Prevention Cheat Sheet for more details on the types of encoding and escaping that are needed.

Phase: Architecture and Design

For any security checks that are performed on the client side, ensure that these checks are duplicated on the server side, in order to avoid CWE-602. Attackers can bypass the client-side checks by modifying values after the checks have been performed, or by changing the client to remove the client-side checks entirely. Then, these modified values would be submitted to the server.

If available, use structured mechanisms that automatically enforce the separation between data and code. These mechanisms may be able to provide the relevant quoting, encoding, and validation automatically, instead of relying on the developer to provide this capability at every point where output is generated.

Phase: Implementation

For every web page that is generated, use and specify a character encoding such as ISO-8859-1 or UTF-8. When an encoding is not specified, the web browser may choose a different encoding by guessing which encoding is actually being used by the web page. This can cause the web browser to treat certain sequences as special, opening up the client to subtle XSS attacks. See CWE-116 for more mitigations related to encoding/escaping.

To help mitigate XSS attacks against the user's session cookie, set the session cookie to be HttpOnly. In browsers that support the HttpOnly feature (such as more recent versions of Internet Explorer and Firefox), this attribute can prevent the user's session cookie from being accessible to

malicious client-side scripts that use document.cookie. This is not a complete solution, since HttpOnly is not supported by all browsers. More importantly, XMLHTTPRequest and other powerful browser technologies provide read access to HTTP headers, including the Set-Cookie header in which the HttpOnly flag is set.

Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use an allow list of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. Do not rely exclusively on looking for malicious or malformed inputs (i.e., do not rely on a deny list). However, deny lists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright.

When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid because it only contains alphanumeric characters, but it is not valid if you are expecting colors such as "red" or "blue."

Ensure that you perform input validation at well-defined interfaces within the application. This will help protect the application even if a component is reused or moved elsewhere.

Reference	https://owasp.org/www-community/attacks/xss/ https://cwe.mitre.org/data/definitions/79.html
CWE Id	79
WASC Id	8
Plugin Id	40012

High **SQL Injection - MySQL**

Description	SQL injection may be possible.
URL	http://testphp.vulnweb.com/artists.php?artist=%27
Node Name	http://testphp.vulnweb.com/artists.php (artist)
Method	GET
Parameter	artist
Attack	'
Evidence	You have an error in your SQL syntax
Other Info	RDBMS [MySQL] likely, given error message regular expression [\QYou have an error in your SQL syntax\E] matched by the HTML results. The vulnerability was detected by manipulating the parameter to cause a database error message to be returned and recognised.
URL	http://testphp.vulnweb.com/listproducts.php?artist=%27
Node Name	http://testphp.vulnweb.com/listproducts.php (artist)
Method	GET
Parameter	artist
Attack	'
Evidence	You have an error in your SQL syntax
Other Info	RDBMS [MySQL] likely, given error message regular expression [\QYou have an error in your SQL syntax\E] matched by the HTML results. The vulnerability was detected by manipulating the parameter to cause a database error message to be returned and recognised.
URL	http://testphp.vulnweb.com/listproducts.php?cat=%27

Node Name	http://testphp.vulnweb.com/listproducts.php (cat)
Method	GET
Parameter	cat
Attack	'
Evidence	You have an error in your SQL syntax
Other Info	RDBMS [MySQL] likely, given error message regular expression [\QYou have an error in your SQL syntax\E] matched by the HTML results. The vulnerability was detected by manipulating the parameter to cause a database error message to be returned and recognised.
URL	http://testphp.vulnweb.com/product.php?pic=%27
Node Name	http://testphp.vulnweb.com/product.php (pic)
Method	GET
Parameter	pic
Attack	'
Evidence	You have an error in your SQL syntax
Other Info	RDBMS [MySQL] likely, given error message regular expression [\QYou have an error in your SQL syntax\E] matched by the HTML results. The vulnerability was detected by manipulating the parameter to cause a database error message to be returned and recognised.
URL	http://testphp.vulnweb.com/search.php?test=%27
Node Name	http://testphp.vulnweb.com/search.php (test)
Method	GET
Parameter	test
Attack	'
Evidence	You have an error in your SQL syntax
Other Info	RDBMS [MySQL] likely, given error message regular expression [\QYou have an error in your SQL syntax\E] matched by the HTML results. The vulnerability was detected by manipulating the parameter to cause a database error message to be returned and recognised.
URL	http://testphp.vulnweb.com/search.php?test=query
Node Name	http://testphp.vulnweb.com/search.php (test)(goButton,searchFor)
Method	POST
Parameter	searchFor
Attack	ZAP'
Evidence	You have an error in your SQL syntax
Other Info	RDBMS [MySQL] likely, given error message regular expression [\QYou have an error in your SQL syntax\E] matched by the HTML results. The vulnerability was detected by manipulating the parameter to cause a database error message to be returned and recognised.
URL	http://testphp.vulnweb.com/search.php?test=%27
Node Name	http://testphp.vulnweb.com/search.php (test)(goButton,searchFor)
Method	POST
Parameter	test

Attack	'
Evidence	You have an error in your SQL syntax
Other Info	RDBMS [MySQL] likely, given error message regular expression [\QYou have an error in your SQL syntax\E] matched by the HTML results. The vulnerability was detected by manipulating the parameter to cause a database error message to be returned and recognised.
URL	http://testphp.vulnweb.com/secured/newuser.php
Node Name	http://testphp.vulnweb.com/secured/newuser.php () (signup,uaddress,ucc,uemail,upass,upass2,uphone,urname,uuname)
Method	POST
Parameter	uuname
Attack	'
Evidence	You have an error in your SQL syntax
Other Info	RDBMS [MySQL] likely, given error message regular expression [\QYou have an error in your SQL syntax\E] matched by the HTML results. The vulnerability was detected by manipulating the parameter to cause a database error message to be returned and recognised.
URL	http://testphp.vulnweb.com/userinfo.php
Node Name	http://testphp.vulnweb.com/userinfo.php ()(pass,uname)
Method	POST
Parameter	pass
Attack	'
Evidence	You have an error in your SQL syntax
Other Info	RDBMS [MySQL] likely, given error message regular expression [\QYou have an error in your SQL syntax\E] matched by the HTML results. The vulnerability was detected by manipulating the parameter to cause a database error message to be returned and recognised.
URL	http://testphp.vulnweb.com/userinfo.php
Node Name	http://testphp.vulnweb.com/userinfo.php ()(pass,uname)
Method	POST
Parameter	uname
Attack	'
Evidence	You have an error in your SQL syntax
Other Info	RDBMS [MySQL] likely, given error message regular expression [\QYou have an error in your SQL syntax\E] matched by the HTML results. The vulnerability was detected by manipulating the parameter to cause a database error message to be returned and recognised.
Instances	10
Solution	Do not trust client side input, even if there is client side validation in place. In general, type check all data on the server side. If the application uses JDBC, use PreparedStatement or CallableStatement, with parameters passed by '?' If the application uses ASP, use ADO Command Objects with strong type checking and parameterized queries. If database Stored Procedures can be used, use them.

	Do *not* concatenate strings into queries in the stored procedure, or use 'exec', 'exec immediate', or equivalent functionality! Do not create dynamic SQL queries using simple string concatenation. Escape all data received from the client. Apply an 'allow list' of allowed characters, or a 'deny list' of disallowed characters in user input. Apply the principle of least privilege by using the least privileged database user possible. In particular, avoid using the 'sa' or 'db-owner' database users. This does not eliminate SQL injection, but minimizes its impact. Grant the minimum database access that is necessary for the application.
Reference	https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
CWE Id	89
WASC Id	19
Plugin Id	40018

High **SQL Injection - MySQL (Time Based)**

Description	SQL injection may be possible.
URL	http://testphp.vulnweb.com/artists.php?artist=2
Node Name	http://testphp.vulnweb.com/artists.php (artist)
Method	GET
Parameter	artist
Attack	2 and 0 in (select sleep(15)) --
Evidence	
Other Info	The query time is controllable using parameter value [2 and 0 in (select sleep(15)) --], which caused the request to take [15,690] milliseconds, when the original unmodified query with value [2] took [0] milliseconds.
URL	http://testphp.vulnweb.com/listproducts.php?artist=2
Node Name	http://testphp.vulnweb.com/listproducts.php (artist)
Method	GET
Parameter	artist
Attack	2 and 0 in (select sleep(15)) --
Evidence	
Other Info	The query time is controllable using parameter value [2 and 0 in (select sleep(15)) --], which caused the request to take [15,883] milliseconds, when the original unmodified query with value [2] took [0] milliseconds.
URL	http://testphp.vulnweb.com/listproducts.php?cat=2
Node Name	http://testphp.vulnweb.com/listproducts.php (cat)
Method	GET
Parameter	cat
Attack	2 and 0 in (select sleep(15)) --

Evidence	
Other Info	The query time is controllable using parameter value [2 and 0 in (select sleep(15)) --], which caused the request to take [15,871] milliseconds, when the original unmodified query with value [2] took [0] milliseconds.
URL	http://testphp.vulnweb.com/product.php?pic=7
Node Name	http://testphp.vulnweb.com/product.php (pic)
Method	GET
Parameter	pic
Attack	7 and 0 in (select sleep(15)) --
Evidence	
Other Info	The query time is controllable using parameter value [7 and 0 in (select sleep(15)) --], which caused the request to take [15,582] milliseconds, when the original unmodified query with value [7] took [0] milliseconds.
URL	http://testphp.vulnweb.com/secured/newuser.php
Node Name	http://testphp.vulnweb.com/secured/newuser.php () (signup,uaddress,ucc,uemail,upass,upass2,uphone,urname,uuname)
Method	POST
Parameter	uuname
Attack	ZAP' / sleep(15) / '
Evidence	
Other Info	The query time is controllable using parameter value [ZAP' / sleep(15) / '], which caused the request to take [15,695] milliseconds, when the original unmodified query with value [ZAP] took [0] milliseconds.
URL	http://testphp.vulnweb.com/userinfo.php
Node Name	http://testphp.vulnweb.com/userinfo.php ()(pass,uname)
Method	POST
Parameter	uname
Attack	ZAP' / sleep(15) / '
Evidence	
Other Info	The query time is controllable using parameter value [ZAP' / sleep(15) / '], which caused the request to take [15,693] milliseconds, when the original unmodified query with value [ZAP] took [0] milliseconds.
Instances	6
Solution	Do not trust client side input, even if there is client side validation in place. In general, type check all data on the server side. If the application uses JDBC, use PreparedStatement or CallableStatement, with parameters passed by '?' If the application uses ASP, use ADO Command Objects with strong type checking and parameterized queries. If database Stored Procedures can be used, use them. Do *not* concatenate strings into queries in the stored procedure, or use 'exec', 'exec immediate', or equivalent functionality!

	Do not create dynamic SQL queries using simple string concatenation. Escape all data received from the client. Apply an 'allow list' of allowed characters, or a 'deny list' of disallowed characters in user input. Apply the principle of least privilege by using the least privileged database user possible. In particular, avoid using the 'sa' or 'db-owner' database users. This does not eliminate SQL injection, but minimizes its impact. Grant the minimum database access that is necessary for the application.
Reference	https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
CWE Id	89
WASC Id	19
Plugin Id	40019

High **SQL Injection - SQLite (Time Based)**

Description	SQL injection may be possible.
URL	http://testphp.vulnweb.com/listproducts.php?artist=2
Node Name	http://testphp.vulnweb.com/listproducts.php (artist)
Method	GET
Parameter	artist
Attack	case randomblob(10000000) when not null then 1 else 1 end
Evidence	
Other Info	The query time is controllable using parameter value [case randomblob(10000000) when not null then 1 else 1 end], which caused the request to take [769] milliseconds, parameter value [case randomblob(100000000) when not null then 1 else 1 end], which caused the request to take [1,124] milliseconds, when the original unmodified query with value [2] took [630] milliseconds.
URL	http://testphp.vulnweb.com/secured/newuser.php
Node Name	http://testphp.vulnweb.com/secured/newuser.php () (signup,uaddress,ucc,uemail,upass,upass2,uphone,urname,uuname)
Method	POST
Parameter	uemail
Attack	case randomblob(100000) when not null then 1 else 1 end
Evidence	
Other Info	The query time is controllable using parameter value [case randomblob(100000) when not null then 1 else 1 end], which caused the request to take [1,070] milliseconds, parameter value [case randomblob(1000000) when not null then 1 else 1 end], which caused the request to take [1,841] milliseconds, when the original unmodified query with value [ZAP] took [664] milliseconds.
Instances	2
Solution	Do not trust client side input, even if there is client side validation in place. In general, type check all data on the server side. If the application uses JDBC, use PreparedStatement or CallableStatement, with parameters passed by '?'

If the application uses ASP, use ADO Command Objects with strong type checking and parameterized queries.

If database Stored Procedures can be used, use them.

Do *not* concatenate strings into queries in the stored procedure, or use 'exec', 'exec immediate', or equivalent functionality!

Do not create dynamic SQL queries using simple string concatenation.

Escape all data received from the client.

Apply an 'allow list' of allowed characters, or a 'deny list' of disallowed characters in user input.

Apply the principle of least privilege by using the least privileged database user possible.

In particular, avoid using the 'sa' or 'db-owner' database users. This does not eliminate SQL injection, but minimizes its impact.

Grant the minimum database access that is necessary for the application.

Reference	https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
CWE Id	89
WASC Id	19
Plugin Id	40024

Medium **Absence of Anti-CSRF Tokens**

Description	No Anti-CSRF tokens were found in a HTML submission form. A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL/form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) exploits the trust that a user has for a web site. Like XSS, CSRF attacks are not necessarily cross-site, but they can be. Cross-site request forgery is also known as CSRF, XSRF, one-click attack, session riding, confused deputy, and sea surf. CSRF attacks are effective in a number of situations, including: * The victim has an active session on the target site. * The victim is authenticated via HTTP auth on the target site. * The victim is on the same local network as the target site. CSRF has primarily been used to perform an action against a target site using the victim's privileges, but recent techniques have been discovered to disclose information by gaining access to the response. The risk of information disclosure is dramatically increased when the target site is vulnerable to XSS, because XSS can be used as a platform for CSRF, allowing the attack to operate within the bounds of the same-origin policy.
URL	http://testphp.vulnweb.com
Node Name	http://testphp.vulnweb.com
Method	GET
Parameter	
Attack	
Evidence	<form action="search.php?test=query" method="post">

Other Info	No known Anti-CSRF token [anticsrf, CSRFToken, __RequestVerificationToken, csrfmiddlewaretoken, authenticity_token, OWASP_CSRFTOKEN, anoncsrf, csrf_token, _csrf, _csrfSecret, __csrf_magic, CSRF, _token, _csrf_token, _csrfToken] was found in the following HTML form: [Form 1: "goButton" "searchFor"].
URL	http://testphp.vulnweb.com/cart.php
Node Name	http://testphp.vulnweb.com/cart.php
Method	GET
Parameter	
Attack	
Evidence	<form action="search.php?test=query" method="post">
Other Info	No known Anti-CSRF token [anticsrf, CSRFToken, __RequestVerificationToken, csrfmiddlewaretoken, authenticity_token, OWASP_CSRFTOKEN, anoncsrf, csrf_token, _csrf, _csrfSecret, __csrf_magic, CSRF, _token, _csrf_token, _csrfToken] was found in the following HTML form: [Form 1: "goButton" "searchFor"].
URL	http://testphp.vulnweb.com/categories.php
Node Name	http://testphp.vulnweb.com/categories.php
Method	GET
Parameter	
Attack	
Evidence	<form action="search.php?test=query" method="post">
Other Info	No known Anti-CSRF token [anticsrf, CSRFToken, __RequestVerificationToken, csrfmiddlewaretoken, authenticity_token, OWASP_CSRFTOKEN, anoncsrf, csrf_token, _csrf, _csrfSecret, __csrf_magic, CSRF, _token, _csrf_token, _csrfToken] was found in the following HTML form: [Form 1: "goButton" "searchFor"].
URL	http://testphp.vulnweb.com/guestbook.php
Node Name	http://testphp.vulnweb.com/guestbook.php
Method	GET
Parameter	
Attack	
Evidence	<form action="" method="post" name="faddentry">
Other Info	No known Anti-CSRF token [anticsrf, CSRFToken, __RequestVerificationToken, csrfmiddlewaretoken, authenticity_token, OWASP_CSRFTOKEN, anoncsrf, csrf_token, _csrf, _csrfSecret, __csrf_magic, CSRF, _token, _csrf_token, _csrfToken] was found in the following HTML form: [Form 1: "name" "submit"].
URL	http://testphp.vulnweb.com/guestbook.php
Node Name	http://testphp.vulnweb.com/guestbook.php
Method	GET
Parameter	
Attack	
Evidence	<form action="search.php?test=query" method="post">
Other Info	No known Anti-CSRF token [anticsrf, CSRFToken, __RequestVerificationToken, csrfmiddlewaretoken, authenticity_token, OWASP_CSRFTOKEN, anoncsrf, csrf_token, _csrf,

	_csrfSecret, __csrf_magic, CSRF, _token, _csrf_token, _csrfToken] was found in the following HTML form: [Form 2: "goButton" "searchFor"].
Instances	Systemic
Solution	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid. For example, use anti-CSRF packages such as the OWASP CSRFGuard. Phase: Implementation Ensure that your application is free of cross-site scripting issues, because most CSRF defenses can be bypassed using attacker-controlled script. Phase: Architecture and Design Generate a unique nonce for each form, place the nonce into the form, and verify the nonce upon receipt of the form. Be sure that the nonce is not predictable (CWE-330). Note that this can be bypassed using XSS. Identify especially dangerous operations. When the user performs a dangerous operation, send a separate confirmation request to ensure that the user intended to perform that operation. Note that this can be bypassed using XSS. Use the ESAPI Session Management control. This control includes a component for CSRF. Do not use the GET method for any request that triggers a state change. Phase: Implementation Check the HTTP Referer header to see if the request originated from an expected page. This could break legitimate functionality, because users or proxies may have disabled sending the Referer for privacy reasons.
Reference	https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html https://cwe.mitre.org/data/definitions/352.html
CWE Id	352
WASC Id	9
Plugin Id	10202

Medium **Content Security Policy (CSP) Header Not Set**

Description	Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.
URL	http://testphp.vulnweb.com
Node Name	http://testphp.vulnweb.com
Method	GET

Parameter	
Attack	
Evidence	
Other Info	
URL	http://testphp.vulnweb.com/AJAX/index.php
Node Name	http://testphp.vulnweb.com/AJAX/index.php
Method	GET
Parameter	
Attack	
Evidence	
Other Info	
URL	http://testphp.vulnweb.com/guestbook.php
Node Name	http://testphp.vulnweb.com/guestbook.php
Method	GET
Parameter	
Attack	
Evidence	
Other Info	
URL	http://testphp.vulnweb.com/robots.txt
Node Name	http://testphp.vulnweb.com/robots.txt
Method	GET
Parameter	
Attack	
Evidence	
Other Info	
URL	http://testphp.vulnweb.com/sitemap.xml
Node Name	http://testphp.vulnweb.com/sitemap.xml
Method	GET
Parameter	
Attack	
Evidence	
Other Info	
Instances	Systemic
Solution	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.
Reference	https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/CSP https://cheatsheetseries.owasp.org/cheatsheets/Content_Security_Policy_Cheat_Sheet.html https://www.w3.org/TR/CSP/

	https://w3c.github.io/webappsec-csp/ https://web.dev/articles/csp https://caniuse.com/#feat=contentsecuritypolicy https://content-security-policy.com/
CWE Id	693
WASC Id	15
Plugin Id	10038

Medium **Directory Browsing**

Description	It is possible to view a listing of the directory contents. Directory listings may reveal hidden scripts, include files, backup source files, etc., which can be accessed to reveal sensitive information.
URL	http://testphp.vulnweb.com/Flash/
Node Name	http://testphp.vulnweb.com/Flash/
Method	GET
Parameter	
Attack	
Evidence	<title>Index of /Flash/</title>
Other Info	Web server identified: Apache 2
URL	http://testphp.vulnweb.com/Mod_Rewrite_Shop/images/
Node Name	http://testphp.vulnweb.com/Mod_Rewrite_Shop/images/
Method	GET
Parameter	
Attack	
Evidence	<title>Index of /Mod_Rewrite_Shop/images/</title>
Other Info	Web server identified: Apache 2
URL	http://testphp.vulnweb.com/images/
Node Name	http://testphp.vulnweb.com/images/
Method	GET
Parameter	
Attack	
Evidence	<title>Index of /images/</title>
Other Info	Web server identified: Apache 2
Instances	3
Solution	Configure the web server to disable directory browsing.
Reference	https://cwe.mitre.org/data/definitions/548.html
CWE Id	548
WASC Id	16
Plugin Id	10033

Medium **HTTP Only Site**

Description	The site is only served under HTTP and not HTTPS.
URL	http://testphp.vulnweb.com/secured/
Node Name	https://testphp.vulnweb.com/secured/
Method	GET
Parameter	
Attack	
Evidence	
Other Info	Failed to connect. ZAP attempted to connect via: https://testphp.vulnweb.com/secured/
Instances	1
Solution	Configure your web or application server to use SSL (https).
Reference	https://cheatsheetseries.owasp.org/cheatsheets/Transport_Layer_Protection_Cheat_Sheet.html https://letsencrypt.org/
CWE Id	311
WASC Id	4
Plugin Id	10106

Medium **Hidden File Found**

Description	A sensitive file was identified as accessible or available. This may leak administrative, configuration, or credential information which can be leveraged by a malicious individual to further attack the system or conduct social engineering efforts.
URL	http://testphp.vulnweb.com/secured/phpinfo.php
Node Name	http://testphp.vulnweb.com/secured/phpinfo.php
Method	GET
Parameter	
Attack	
Evidence	HTTP/1.1 200 OK
Other Info	phpinfo
Instances	1
Solution	Consider whether or not the component is actually required in production, if it isn't then disable it. If it is then ensure access to it requires appropriate authentication and authorization, or limit exposure to internal systems or specific source IPs, etc.
Reference	https://blog.hboeck.de/archives/892-Introducing-Snallygaster-a-Tool-to-Scan-for-Secrets-on-Web-Servers.html https://www.php.net/manual/en/function.phpinfo.php
CWE Id	538
WASC Id	13
Plugin Id	40035

Medium **Missing Anti-clickjacking Header**

Description	The response does not protect against 'ClickJacking' attacks. It should include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame-Options.
URL	http://testphp.vulnweb.com

Node Name	http://testphp.vulnweb.com
Method	GET
Parameter	x-frame-options
Attack	
Evidence	
Other Info	
URL	http://testphp.vulnweb.com/AJAX/index.php
Node Name	http://testphp.vulnweb.com/AJAX/index.php
Method	GET
Parameter	x-frame-options
Attack	
Evidence	
Other Info	
URL	http://testphp.vulnweb.com/cart.php
Node Name	http://testphp.vulnweb.com/cart.php
Method	GET
Parameter	x-frame-options
Attack	
Evidence	
Other Info	
URL	http://testphp.vulnweb.com/categories.php
Node Name	http://testphp.vulnweb.com/categories.php
Method	GET
Parameter	x-frame-options
Attack	
Evidence	
Other Info	
URL	http://testphp.vulnweb.com/guestbook.php
Node Name	http://testphp.vulnweb.com/guestbook.php
Method	GET
Parameter	x-frame-options
Attack	
Evidence	
Other Info	
Instances	Systemic
Solution	Modern Web browsers support the Content-Security-Policy and X-Frame-Options HTTP headers. Ensure one of them is set on all web pages returned by your site/app.

	If you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. Alternatively consider implementing Content Security Policy's "frame-ancestors" directive.
Reference	https://developer.mozilla.org/en-US/docs/Web/HTTP/Reference/Headers/X-Frame-Options
CWE Id	1021
WASC Id	15
Plugin Id	10020

Medium	**XSLT Injection**
Description	Injection using XSL transformations may be possible, and may allow an attacker to read system information, read and write files, or execute arbitrary code.
URL	http://testphp.vulnweb.com/showimage.php?file=%3Cxsl%3Avalue-of+select%3D%22document%28%27http%3A%2F%2Ftestphp.vulnweb.com%3A22%27%29%22%2F%3E
Node Name	http://testphp.vulnweb.com/showimage.php (file)
Method	GET
Parameter	file
Attack	<xsl:value-of select="document('http://testphp.vulnweb.com:22')"/>
Evidence	failed to open stream
Other Info	Port scanning may be possible.
URL	http://testphp.vulnweb.com/showimage.php?file=%3Cxsl%3Avalue-of+select%3D%22document%28%27http%3A%2F%2Ftestphp.vulnweb.com%3A22%27%29%22%2F%3E&size=160
Node Name	http://testphp.vulnweb.com/showimage.php (file,size)
Method	GET
Parameter	file
Attack	<xsl:value-of select="document('http://testphp.vulnweb.com:22')"/>
Evidence	failed to open stream
Other Info	Port scanning may be possible.
Instances	2
Solution	Sanitize and analyze every user input coming from any client-side.
Reference	https://book.hacktricks.wiki/en/pentesting-web/xslt-server-side-injection-extensible-stylesheet-language-transformations.html
CWE Id	91
WASC Id	23
Plugin Id	90017

Low	**In Page Banner Information Leak**
Description	The server returned a version banner string in the response content. Such information leaks may allow attackers to further target specific issues impacting the product and version in use.
URL	http://testphp.vulnweb.com/Mod_Rewrite_Shop/BuyProduct-1

Node Name	http://testphp.vulnweb.com/Mod_Rewrite_Shop/BuyProduct-1
Method	GET
Parameter	
Attack	
Evidence	nginx/1.19.0
Other Info	There is a chance that the highlight in the finding is on a value in the headers, versus the actual matched string in the response body.
URL	http://testphp.vulnweb.com/Mod_Rewrite_Shop/BuyProduct-2
Node Name	http://testphp.vulnweb.com/Mod_Rewrite_Shop/BuyProduct-2
Method	GET
Parameter	
Attack	
Evidence	nginx/1.19.0
Other Info	There is a chance that the highlight in the finding is on a value in the headers, versus the actual matched string in the response body.
URL	http://testphp.vulnweb.com/high
Node Name	http://testphp.vulnweb.com/high
Method	GET
Parameter	
Attack	
Evidence	nginx/1.19.0
Other Info	There is a chance that the highlight in the finding is on a value in the headers, versus the actual matched string in the response body.
URL	http://testphp.vulnweb.com/robots.txt
Node Name	http://testphp.vulnweb.com/robots.txt
Method	GET
Parameter	
Attack	
Evidence	nginx/1.19.0
Other Info	There is a chance that the highlight in the finding is on a value in the headers, versus the actual matched string in the response body.
URL	http://testphp.vulnweb.com/sitemap.xml
Node Name	http://testphp.vulnweb.com/sitemap.xml
Method	GET
Parameter	
Attack	
Evidence	nginx/1.19.0

Other Info	There is a chance that the highlight in the finding is on a value in the headers, versus the actual matched string in the response body.
Instances	Systemic
Solution	Configure the server to prevent such information leaks. For example: Under Tomcat this is done via the "server" directive and implementation of custom error pages. Under Apache this is done via the "ServerSignature" and "ServerTokens" directives.
Reference	https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/08-Testing_for_Error_Handling/
CWE Id	497
WASC Id	13
Plugin Id	10009

Low **Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)**

Description	The web/application server is leaking information via one or more "X-Powered-By" HTTP response headers. Access to such information may facilitate attackers identifying other frameworks/components your web application is reliant upon and the vulnerabilities such components may be subject to.
URL	http://testphp.vulnweb.com
Node Name	http://testphp.vulnweb.com
Method	GET
Parameter	
Attack	
Evidence	X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
Other Info	
URL	http://testphp.vulnweb.com/AJAX/index.php
Node Name	http://testphp.vulnweb.com/AJAX/index.php
Method	GET
Parameter	
Attack	
Evidence	X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
Other Info	
URL	http://testphp.vulnweb.com/cart.php
Node Name	http://testphp.vulnweb.com/cart.php
Method	GET
Parameter	
Attack	
Evidence	X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
Other Info	
URL	http://testphp.vulnweb.com/categories.php

Node Name	http://testphp.vulnweb.com/categories.php
Method	GET
Parameter	
Attack	
Evidence	X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
Other Info	
URL	http://testphp.vulnweb.com/guestbook.php
Node Name	http://testphp.vulnweb.com/guestbook.php
Method	GET
Parameter	
Attack	
Evidence	X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
Other Info	
Instances	Systemic
Solution	Ensure that your web server, application server, load balancer, etc. is configured to suppress "X-Powered-By" headers.
Reference	https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application_Security_Testing/01-Information_Gathering/08-Fingerprint_Web_Application_Framework https://www.troyhunt.com/shhh-dont-let-your-response-headers/
CWE Id	497
WASC Id	13
Plugin Id	10037

Low	**Server Leaks Version Information via "Server" HTTP Response Header Field**
Description	The web/application server is leaking version information via the "Server" HTTP response header. Access to such information may facilitate attackers identifying other vulnerabilities your web/application server is subject to.
URL	http://testphp.vulnweb.com
Node Name	http://testphp.vulnweb.com
Method	GET
Parameter	
Attack	
Evidence	nginx/1.19.0
Other Info	
URL	http://testphp.vulnweb.com/AJAX/index.php
Node Name	http://testphp.vulnweb.com/AJAX/index.php
Method	GET
Parameter	
Attack	

Evidence	nginx/1.19.0
Other Info	
URL	http://testphp.vulnweb.com/guestbook.php
Node Name	http://testphp.vulnweb.com/guestbook.php
Method	GET
Parameter	
Attack	
Evidence	nginx/1.19.0
Other Info	
URL	http://testphp.vulnweb.com/robots.txt
Node Name	http://testphp.vulnweb.com/robots.txt
Method	GET
Parameter	
Attack	
Evidence	nginx/1.19.0
Other Info	
URL	http://testphp.vulnweb.com/sitemap.xml
Node Name	http://testphp.vulnweb.com/sitemap.xml
Method	GET
Parameter	
Attack	
Evidence	nginx/1.19.0
Other Info	
Instances	Systemic
Solution	Ensure that your web server, application server, load balancer, etc. is configured to suppress the "Server" header or provide generic details.
Reference	https://httpd.apache.org/docs/current/mod/core.html#servertokens https://learn.microsoft.com/en-us/previous-versions/msp-n-p/ff648552(v=pandp.10) https://www.troyhunt.com/shhh-dont-let-your-response-headers/
CWE Id	497
WASC Id	13
Plugin Id	10036

Low	**Strict-Transport-Security Header Not Set**
Description	HTTP Strict Transport Security (HSTS) is a web security policy mechanism whereby a web server declares that complying user agents (such as a web browser) are to interact with it using only secure HTTPS connections (i.e. HTTP layered over TLS/SSL). HSTS is an IETF standards track protocol and is specified in RFC 6797.
URL	https://data-edge.smartscreen.microsoft.com/api/browser/edge/data/settings/3
Node Name	https://data-edge.smartscreen.microsoft.com/api/browser/edge/data/settings/3 ()({identity:{user:{locale},device:{id,customId,onlineIdTicket,family,locale,osVersion,browser:

	{internetExplorer},netJoinStatus,enterprise:{},cloudSku,architecture},caller:{locale,name,version},client:{version,data:{topTraffic,customSynchronousLookupUris,edgeSettings,customSettings}}},correlationId,debugInfo:{clientId}})
Method	POST
Parameter	
Attack	
Evidence	
Other Info	
URL	https://data-edge.smartscreen.microsoft.com/api/browser/edge/data/toptraffic/3
Node Name	https://data-edge.smartscreen.microsoft.com/api/browser/edge/data/toptraffic/3 ()({identity:{user:{locale},device:{id,customId,onlineIdTicket,family,locale,osVersion,browser:{internetExplorer},netJoinStatus,enterprise:{},cloudSku,architecture},caller:{locale,name,version},client:{version,data:{topTraffic,customSynchronousLookupUris,edgeSettings,customSettings}}},correlationId,debugInfo:{clientId}})
Method	POST
Parameter	
Attack	
Evidence	
Other Info	
URL	https://nav-edge.smartscreen.microsoft.com/api/browser/edge/navigate/3
Node Name	https://nav-edge.smartscreen.microsoft.com/api/browser/edge/navigate/3 ()({userAgent,redirectChain:{source,chain:[]},identity:{user:{locale},device:{id,customId,onlineIdTicket,family,locale,osVersion,browser:{internetExplorer},netJoinStatus,enterprise:{},cloudSku,architecture},caller:{locale,name,version},client:{version,data:{topTraffic,customSynchronousLookupUris,edgeSettings,customSettings}}},config:{user:{uriReputation:{enforcedByPolicy,level}},device:{appControl:{level},appReputation:{enforcedByPolicy,level}}},destination:{uri,ip},type,forceServiceDetermination,correlatio...)
Method	POST
Parameter	
Attack	
Evidence	
Other Info	
URL	https://telem-edge.smartscreen.microsoft.com/api/browser/edge/telemetry/3
Node Name	https://telem-edge.smartscreen.microsoft.com/api/browser/edge/telemetry/3 ()({executionTime,random,samplingRates:{evaluateModel,serverCall},config:{user:{uriReputation:{enforcedByPolicy,level}},device:{appControl:{level},appReputation:{enforcedByPolicy,level}}},correlationId,identity:{user:{locale},device:{id,customId,onlineIdTicket,family,locale,osVersion,browser:{internetExplorer},netJoinStatus,enterprise:{},cloudSku,architecture},caller:{locale,name,version},client:{version,data:{topTraffic,customSynchronousLookupUris,edgeSettings,customSettings}}},userAgent,events:[{$type,nam...)
Method	POST
Parameter	
Attack	

Evidence	
Other Info	
Instances	4
Solution	Ensure that your web server, application server, load balancer, etc. is configured to enforce Strict-Transport-Security.
Reference	https://cheatsheetseries.owasp.org/cheatsheets/HTTP_Strict_Transport_Security_Cheat_Sheet.html https://owasp.org/www-community/Security_Headers https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security https://caniuse.com/stricttransportsecurity https://datatracker.ietf.org/doc/html/rfc6797
CWE Id	319
WASC Id	15
Plugin Id	10035

Low **X-Content-Type-Options Header Missing**

Description	The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.
URL	http://testphp.vulnweb.com
Node Name	http://testphp.vulnweb.com
Method	GET
Parameter	x-content-type-options
Attack	
Evidence	
Other Info	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.
URL	http://testphp.vulnweb.com/AJAX/index.php
Node Name	http://testphp.vulnweb.com/AJAX/index.php
Method	GET
Parameter	x-content-type-options
Attack	
Evidence	
Other Info	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.
URL	http://testphp.vulnweb.com/cart.php
Node Name	http://testphp.vulnweb.com/cart.php
Method	GET

Parameter	x-content-type-options
Attack	
Evidence	
Other Info	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.
URL	http://testphp.vulnweb.com/categories.php
Node Name	http://testphp.vulnweb.com/categories.php
Method	GET
Parameter	x-content-type-options
Attack	
Evidence	
Other Info	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.
URL	http://testphp.vulnweb.com/guestbook.php
Node Name	http://testphp.vulnweb.com/guestbook.php
Method	GET
Parameter	x-content-type-options
Attack	
Evidence	
Other Info	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.
URL	https://data-edge.smartscreen.microsoft.com/api/browser/edge/data/settings/3
Node Name	https://data-edge.smartscreen.microsoft.com/api/browser/edge/data/settings/3 ()({identity:{user:{locale},device:{id,customId,onlineIdTicket,family,locale,osVersion,browser:{internetExplorer},netJoinStatus,enterprise:{},cloudSku,architecture},caller:{locale,name,version},client:{version,data:{topTraffic,customSynchronousLookupUris,edgeSettings,customSettings}}},correlationId,debugInfo:{clientId}})
Method	POST
Parameter	x-content-type-options
Attack	
Evidence	
Other Info	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.
URL	https://data-edge.smartscreen.microsoft.com/api/browser/edge/data/toptraffic/3
Node Name	https://data-edge.smartscreen.microsoft.com/api/browser/edge/data/toptraffic/3 ()({identity:{user:{locale},device:{id,customId,onlineIdTicket,family,locale,osVersion,browser:{internetExplorer},netJoinStatus,enterprise:{},cloudSku,architecture},caller:

	{locale,name,version},client:{version,data:{topTraffic,customSynchronousLookupUris,edgeSettings,customSettings}}},correlationId,debugInfo:{clientId}})
Method	POST
Parameter	x-content-type-options
Attack	
Evidence	
Other Info	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.
URL	https://nav-edge.smartscreen.microsoft.com/api/browser/edge/navigate/3
Node Name	https://nav-edge.smartscreen.microsoft.com/api/browser/edge/navigate/3 ()({userAgent,redirectChain:{source,chain:[]},identity:{user:{locale},device:{id,customId,onlineIdTicket,family,locale,osVersion,browser:{internetExplorer},netJoinStatus,enterprise:{},cloudSku,architecture},caller:{locale,name,version},client:{version,data:{topTraffic,customSynchronousLookupUris,edgeSettings,customSettings}}},config:{user:{uriReputation:{enforcedByPolicy,level}},device:{appControl:{level},appReputation:{enforcedByPolicy,level}}},destination:{uri,ip},type,forceServiceDetermination,correlatio...)
Method	POST
Parameter	x-content-type-options
Attack	
Evidence	
Other Info	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.
Instances	Systemic
Solution	Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages. If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.
Reference	https://learn.microsoft.com/en-us/previous-versions/windows/internet-explorer/ie-developer/compatibility/gg622941(v=vs.85) https://owasp.org/www-community/Security_Headers
CWE Id	693
WASC Id	15
Plugin Id	10021

Informational **Authentication Request Identified**

Description	The given request has been identified as an authentication request. The 'Other Info' field contains a set of key=value lines which identify any relevant fields. If the request is in a context which has an Authentication Method set to "Auto-Detect" then this rule will change the authentication to match the request identified.
URL	http://testphp.vulnweb.com/secured/newuser.php
Node Name	http://testphp.vulnweb.com/secured/newuser.php ()(signup,uaddress,ucc,uemail,upass,upass2,uphone,urname,uuname)

Method	POST
Parameter	uemail
Attack	
Evidence	upass
Other Info	userParam=uemail userValue=ZAP passwordParam=upass referer=http://testphp.vulnweb.com/signup.php
Instances	1
Solution	This is an informational alert rather than a vulnerability and so there is nothing to fix.
Reference	https://www.zaproxy.org/docs/desktop/addons/authentication-helper/auth-req-id/
CWE Id	
WASC Id	
Plugin Id	10111

Informational **Charset Mismatch (Header Versus Meta Content-Type Charset)**

Description	This check identifies responses where the HTTP Content-Type header declares a charset different from the charset defined by the body of the HTML or XML. When there's a charset mismatch between the HTTP header and content body Web browsers can be forced into an undesirable content-sniffing mode to determine the content's correct character set. An attacker could manipulate content on the page to be interpreted in an encoding of their choice. For example, if an attacker can control content at the beginning of the page, they could inject script using UTF-7 encoded text and manipulate some browsers into interpreting that text.
URL	http://testphp.vulnweb.com
Node Name	http://testphp.vulnweb.com
Method	GET
Parameter	
Attack	
Evidence	
Other Info	There was a charset mismatch between the HTTP Header and the META content-type encoding declarations: [UTF-8] and [iso-8859-2] do not match.
URL	http://testphp.vulnweb.com/AJAX/index.php
Node Name	http://testphp.vulnweb.com/AJAX/index.php
Method	GET
Parameter	
Attack	
Evidence	
Other Info	There was a charset mismatch between the HTTP Header and the META content-type encoding declarations: [UTF-8] and [iso-8859-1] do not match.
URL	http://testphp.vulnweb.com/cart.php
Node Name	http://testphp.vulnweb.com/cart.php
Method	GET
Parameter	

Attack	
Evidence	
Other Info	There was a charset mismatch between the HTTP Header and the META content-type encoding declarations: [UTF-8] and [iso-8859-2] do not match.
URL	http://testphp.vulnweb.com/categories.php
Node Name	http://testphp.vulnweb.com/categories.php
Method	GET
Parameter	
Attack	
Evidence	
Other Info	There was a charset mismatch between the HTTP Header and the META content-type encoding declarations: [UTF-8] and [iso-8859-2] do not match.
URL	http://testphp.vulnweb.com/guestbook.php
Node Name	http://testphp.vulnweb.com/guestbook.php
Method	GET
Parameter	
Attack	
Evidence	
Other Info	There was a charset mismatch between the HTTP Header and the META content-type encoding declarations: [UTF-8] and [iso-8859-2] do not match.
Instances	Systemic
Solution	Force UTF-8 for all text content in both the HTTP header and meta tags in HTML or encoding declarations in XML.
Reference	https://code.google.com/archive/p/browsersec/wikis/Part2.wiki#Character_set_handling_and_detection
CWE Id	436
WASC Id	15
Plugin Id	90011

Informational **GET for POST**

Description	A request that was originally observed as a POST was also accepted as a GET. This issue does not represent a security weakness unto itself, however, it may facilitate simplification of other attacks. For example if the original POST is subject to Cross-Site Scripting (XSS), then this finding may indicate that a simplified (GET based) XSS may also be possible.
URL	http://testphp.vulnweb.com/cart.php
Node Name	http://testphp.vulnweb.com/cart.php (addcart,price)
Method	GET
Parameter	
Attack	
Evidence	GET http://testphp.vulnweb.com/cart.php?addcart=7&price=15000 HTTP/1.1
Other Info	

URL	http://testphp.vulnweb.com/guestbook.php
Node Name	http://testphp.vulnweb.com/guestbook.php (name,submit,text)
Method	GET
Parameter	
Attack	
Evidence	GET http://testphp.vulnweb.com/guestbook.php?name=ZAP&submit=add%20message&text= HTTP/1.1
Other Info	
URL	http://testphp.vulnweb.com/search.php?test=query
Node Name	http://testphp.vulnweb.com/search.php (goButton,searchFor)
Method	GET
Parameter	
Attack	
Evidence	GET http://testphp.vulnweb.com/search.php?goButton=go&searchFor=ZAP HTTP/1.1
Other Info	
Instances	3
Solution	Ensure that only POST is accepted where POST is expected.
Reference	
CWE Id	16
WASC Id	20
Plugin Id	10058

Informational	**Modern Web Application**
Description	The application appears to be a modern web application. If you need to explore it automatically then the Ajax Spider may well be more effective than the standard one.
URL	http://testphp.vulnweb.com/AJAX/index.php
Node Name	http://testphp.vulnweb.com/AJAX/index.php
Method	GET
Parameter	
Attack	
Evidence	titles
Other Info	Links have been found that do not have traditional href attributes, which is an indication that this is a modern web application.
URL	http://testphp.vulnweb.com/artists.php
Node Name	http://testphp.vulnweb.com/artists.php
Method	GET
Parameter	
Attack	

Evidence	comment on this artist
Other Info	Links have been found that do not have traditional href attributes, which is an indication that this is a modern web application.
URL	http://testphp.vulnweb.com/artists.php?artist=1
Node Name	http://testphp.vulnweb.com/artists.php (artist)
Method	GET
Parameter	
Attack	
Evidence	comment on this artist
Other Info	Links have been found that do not have traditional href attributes, which is an indication that this is a modern web application.
URL	http://testphp.vulnweb.com/listproducts.php?cat=1
Node Name	http://testphp.vulnweb.com/listproducts.php (cat)
Method	GET
Parameter	
Attack	
Evidence	comment on this picture
Other Info	Links have been found that do not have traditional href attributes, which is an indication that this is a modern web application.
URL	http://testphp.vulnweb.com/listproducts.php?cat=2
Node Name	http://testphp.vulnweb.com/listproducts.php (cat)
Method	GET
Parameter	
Attack	
Evidence	comment on this picture
Other Info	Links have been found that do not have traditional href attributes, which is an indication that this is a modern web application.
Instances	Systemic
Solution	This is an informational alert and so no changes are required.
Reference	
CWE Id	
WASC Id	
Plugin Id	10109

Informational	**User Agent Fuzzer**
Description	Check for differences in response based on fuzzed User Agent (eg. mobile sites, access as a Search Engine Crawler). Compares the response statuscode and the hashcode of the response body with the original response.

URL	http://testphp.vulnweb.com/
Node Name	http://testphp.vulnweb.com/
Method	GET
Parameter	Header User-Agent
Attack	Mozilla/5.0 (iPhone; CPU iPhone OS 8_0_2 like Mac OS X) AppleWebKit/600.1.4 (KHTML, like Gecko) Version/8.0 Mobile/12A366 Safari/600.1.4
Evidence	
Other Info	
URL	http://testphp.vulnweb.com/Mod_Rewrite_Shop/BuyProduct-1/
Node Name	http://testphp.vulnweb.com/Mod_Rewrite_Shop/BuyProduct-1/
Method	GET
Parameter	Header User-Agent
Attack	Mozilla/5.0 (iPhone; CPU iPhone OS 8_0_2 like Mac OS X) AppleWebKit/600.1.4 (KHTML, like Gecko) Version/8.0 Mobile/12A366 Safari/600.1.4
Evidence	
Other Info	
URL	http://testphp.vulnweb.com/Mod_Rewrite_Shop/BuyProduct-2/
Node Name	http://testphp.vulnweb.com/Mod_Rewrite_Shop/BuyProduct-2/
Method	GET
Parameter	Header User-Agent
Attack	Mozilla/5.0 (iPhone; CPU iPhone OS 8_0_2 like Mac OS X) AppleWebKit/600.1.4 (KHTML, like Gecko) Version/8.0 Mobile/12A366 Safari/600.1.4
Evidence	
Other Info	
URL	http://testphp.vulnweb.com/Mod_Rewrite_Shop/BuyProduct-3/
Node Name	http://testphp.vulnweb.com/Mod_Rewrite_Shop/BuyProduct-3/
Method	GET
Parameter	Header User-Agent
Attack	Mozilla/5.0 (iPhone; CPU iPhone OS 8_0_2 like Mac OS X) AppleWebKit/600.1.4 (KHTML, like Gecko) Version/8.0 Mobile/12A366 Safari/600.1.4
Evidence	
Other Info	
URL	http://testphp.vulnweb.com/cart.php
Node Name	http://testphp.vulnweb.com/cart.php ()(addcart,price)
Method	POST
Parameter	Header User-Agent
Attack	Mozilla/5.0 (iPhone; CPU iPhone OS 8_0_2 like Mac OS X) AppleWebKit/600.1.4 (KHTML, like Gecko) Version/8.0 Mobile/12A366 Safari/600.1.4

Evidence	
Other Info	
Instances	Systemic
Solution	
Reference	https://owasp.org/wstg
CWE Id	
WASC Id	
Plugin Id	10104

Informational	**User Controllable HTML Element Attribute (Potential XSS)**
Description	This check looks at user-supplied input in query string parameters and POST data to identify where certain HTML attribute values might be controlled. This provides hot-spot detection for XSS (cross-site scripting) that will require further review by a security analyst to determine exploitability.
URL	http://testphp.vulnweb.com/guestbook.php
Node Name	http://testphp.vulnweb.com/guestbook.php ()(name,submit,text)
Method	POST
Parameter	submit
Attack	
Evidence	
Other Info	User-controlled HTML attribute values were found. Try injecting special characters to see if XSS might be possible. The page at the following URL: http://testphp.vulnweb.com/guestbook.php appears to include user input in: a(n) [input] tag [value] attribute The user input found was: submit=add message The user-controlled value was: add message
URL	http://testphp.vulnweb.com/search.php?test=query
Node Name	http://testphp.vulnweb.com/search.php (test)(goButton,searchFor)
Method	POST
Parameter	goButton
Attack	
Evidence	
Other Info	User-controlled HTML attribute values were found. Try injecting special characters to see if XSS might be possible. The page at the following URL: http://testphp.vulnweb.com/search.php?test=query appears to include user input in: a(n) [input] tag [name] attribute The user input found was: goButton=go The user-controlled value was: gobutton
Instances	2
Solution	Validate all input and sanitize output it before writing to any HTML attributes.
Reference	https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html
CWE Id	20
WASC Id	20
Plugin Id	10031

Sequence Details

With the associated active scan results.