# WEB APPLICATION PENTESTING CHECKLIST



5. Review The Web Contents
☐ Inspect the page source for sensitive info
☐ Try to find Sensitive Javascript codes
☐ Try to find any keys
☐ Make sure the autocomplete is disabled
6. Identifying Application's Entry Points
☐ Identify what the methods used are?
☐ Identify where the methods used are?
☐ Identify the Injection point
7. Mapping Execution Paths
☐ Use Burp Suite
☐ Use Dirsearch
☐ Use Gobuster
8. Fingerprint Web Application Framework
☐ Use the Wappalyzer browser extension
☐ Use Whatweb
☐ View URL extensions
☐ View HTML source code
☐ View the cookie parameter
☐ View the HTTP headers
9. Map Application Architecture
☐ Map the overall site structure

#### **CONFIGURATION & DEPLOYMENT MANAGEMENT TESTING**

1. Test Network Configuration	
☐ Check the network configuration	
☐ Check for default settings	
☐ Check for default credentials	
2. Test Application Configuration	
☐ Ensure only required modules are used	
☐ Ensure unwanted modules are disabled	
☐ Ensure the server can handle DOS	
☐ Check how the application is handling 4xx & 5xx errors	
☐ Check for the privilege required to run	
☐ Check logs for sensitive info	
3. Test File Extension Handling	
☐ Ensure the server won't return sensitive extensions	
☐ Ensure the server won't accept malicious extensions	
☐ Test for file upload vulnerabilities	
4. Davieus Baeless O Hausferen and Eilea	
4. Review Backup & Unreferenced Files	
☐ Ensure unreferenced files don't contain any sensitive info	
☐ Ensure the namings of old and new backup files	
☐ Check the functionality of unreferenced pages	
5. Enumerate Infrastructure & Admin Interfaces	
☐ Try to find the Infrastructure Interface	
☐ Try to find the Admin Interface	
☐ Identify the hidden admin functionalities	

6. Testing HTTP Methods
☐ Discover the supported methods
☐ Ensure the PUT method is disabled
☐ Ensure the OPTIONS method is disabled
☐ Test access control bypass
☐ Test for XST attacks
☐ Test for HTTP method overriding
7. Test HSTS
☐ Ensure HSTS is enabled
8. Test RIA Cross Domain Policy
☐ Check for Adobe's Cross Domain Policy
☐ Ensure it has the least privilege
9. Test File Permission
☐ Ensure the permissions for sensitive files
☐ Test for directory enumeration
10. Test For Subdomain Takeover
☐ Test DNS, A, and CNAME records for subdomain takeover
☐ Test NS records for subdomain takeover
☐ Test 404 response for subdomain takeover
11. Test Cloud Storage
☐ Check the sensitive paths of AWS
☐ Check the sensitive paths of Google Cloud
Check the sensitive paths of Azure

### **IDENTITY MANAGEMENT TESTING**

1. T	est Role Definitions
	Test for forced browsing
	Test for IDOR (Insecure Direct Object Reference)
	Test for parameter tampering
	Ensure low privilege users can't able to access high privilege resources
2. 1	est User Registration Process
	Ensure the same user or identity can't register again and again
	Ensure the registrations are verified
	Ensure disposable email addresses are rejected
	Check what proof is required for successful registration
3. 1	est Account Provisioning Process
	Check the verification for the provisioning process
	Check the verification for the de-provisioning process
	Check the provisioning rights for an admin user to other users
	Check whether a user is able to de-provision themself or not?
	Check for the resources of a de-provisioned user
4. T	esting For Account Enumeration
	Check the response when a valid username and password entered
	Check the response when a valid username and an invalid password entered
	Check the response when an invalid username and password entered
	Ensure the rate-limiting functionality is enabled in username and password fields

5. Te	est For Weak Username Policy
	Check the response for both valid and invalid usernames
	Check for username enumeration
AUT	HENTICATION TESTING
1. Te	est For Un-Encrypted Channel
	Check for the HTTP login page
	Check for the HTTP register or sign-in page
	Check for HTTP forgot password page
	Check for HTTP change password
	Check for resources on HTTP after logout
	Test for forced browsing to HTTP pages
2. Te	est For Default Credentials
	Test with default credentials
	Test organization name as credentials
	Test for response manipulation
	Test for the default username and a blank password
☐ F	Review the page source for credentials
3. Te	est For Weak Lockout Mechanism
	Ensure the account has been locked after 3-5 incorrect attempts
	Ensure the system accepts only the valid CAPTCHA
	Ensure the system rejects the invalid CAPTCHA
	Ensure CAPTCHA code regenerated after reloaded
	Ensure CAPTCHA reloads after entering the wrong code
$\square$ F	Ensure the user has a recovery option for a lockout account

4. Test For Bypassing Authentication Schema
☐ Test forced browsing directly to the internal dashboard without login
☐ Test for session ID prediction
☐ Test for authentication parameter tampering
☐ Test for SQL injection on the login page
☐ Test to gain access with the help of session ID
☐ Test multiple logins allowed or not?
5. Test For Vulnerable Remember Password
☐ Ensure that the stored password is encrypted
☐ Ensure that the stored password is on the server-side
6. Test For Browser Cache Weakness
☐ Ensure proper cache-control is set on sensitive pages
☐ Ensure no sensitive data is stored in the browser cache storage
7. Test For Weak Password Policy
☐ Ensure the password policy is set to strong
☐ Check for password reusability
☐ Check the user is prevented to use his username as a password
☐ Check for the usage of common weak passwords
☐ Check the minimum password length to be set
☐ Check the maximum password length to be set
8. Testing For Weak Security Questions
☐ Check for the complexity of the questions
☐ Check for brute-forcing

9. T	est For Weak Password Reset Function
	Check what information is required to reset the password
	Check for password reset function with HTTP
	Test the randomness of the password reset tokens
	Test the uniqueness of the password reset tokens
	Test for rate limiting on password reset tokens
	Ensure the token must expire after being used
	Ensure the token must expire after not being used for a long time
10.	Test For Weak Password Change Function
	Check if the old password asked to make a change
	Check for the uniqueness of the forgotten password
	Check for blank password change
	Check for password change function with HTTP
	Ensure the old password is not displayed after changed
	Ensure the other sessions got destroyed after the password change
11.	Test For Weak Authentication In Alternative Channel
	Test authentication on the desktop browsers
	Test authentication on the mobile browsers
	Test authentication in a different country
	Test authentication in a different language
	Test authentication on desktop applications
	Test authentication on mobile applications

#### **AUTHORIZATION TESTING**

1. I	esting Directory Traversal File Include
	Identify the injection point on the URL
	Test for Local File Inclusion
	Test for Remote File Inclusion
	Test Traversal on the URL parameter
	Test Traversal on the cookie parameter
2. T	esting Traversal With Encoding
	Test Traversal with Base64 encoding
	Test Traversal with URL encoding
	Test Traversal with ASCII encoding
	Test Traversal with HTML encoding
	Test Traversal with Hex encoding
	Test Traversal with Binary encoding
	Test Traversal with Octal encoding
	Test Traversal with Gzip encoding
3. T	esting Travesal With Different OS Schemes
Ш	Test Traversal with Unix schemes
	Test Traversal with Windows schemes
	Test Traversal with Mac schemes
4. T	est Other Encoding Techniques
	Test Traversal with Double encoding
	Test Traversal with all characters encode
	Test Traversal with only special characters encode

<b>5</b> . T	Test Authorization Schema Bypass
	Test for Horizontal authorization schema bypass
	Test for Vertical authorization schema bypass
	Test override the target with custom headers
6. T	Test For Privilege Escalation
	Identify the injection point
	Test for bypassing the security measures
	Test for forced browsing
	Test for IDOR
	Test for parameter tampering to high privileged user
7. 1	Test For Insecure Direct Object Reference
	Test to change the ID parameter
	Test to add parameters at the endpoints
	Test for HTTP parameter pollution
	Test by adding an extension at the end
	Test with outdated API versions
	Test by wrapping the ID with an array
	Test by wrapping the ID with a JSON object
	Test for JSON parameter pollution
	Test by changing the case
	Test for path traversal
	Test by changing words
	Test by changing methods

### **SESSION MANAGEMENT TESTING**

1. Test For Session Management Schema
☐ Ensure all Set-Cookie directives are secure
☐ Ensure no cookie operation takes place over an unencrypted channel
☐ Ensure the cookie can't be forced over an unencrypted channel
☐ Ensure the HTTPOnly flag is enabled
☐ Check if any cookies are persistent
☐ Check for session cookies and cookie expiration date/time
☐ Check for session fixation
☐ Check for concurrent login
☐ Check for session after logout
☐ Check for session after closing the browser
☐ Try decoding cookies (Base64, Hex, URL, etc)
2. Test For Cookie Attributes
☐ Ensure the cookie must be set with the secure attribute
☐ Ensure the cookie must be set with the path attribute
☐ Ensure the cookie must have the HTTPOnly flag
3. Test For Session Fixation
☐ Ensure new cookies have been issued upon a successful authentication
☐ Test manipulating the cookies
4. Test For Exposed Session Variables
☐ Test for encryption
☐ Test for GET and POST vulnerabilities
☐ Test if GET request incorporating the session ID used
☐ Test by interchanging POST with GET method

5. Test For Back Refresh Attack
☐ Test after password change
☐ Test after logout
6. Test For Cross Site Request Forgery
☐ Check if the token is validated on the server-side or not
☐ Check if the token is validated for full or partial length
☐ Check by comparing the CSRF tokens for multiple dummy accounts
☐ Check CSRF by interchanging POST with GET method
☐ Check CSRF by removing the CSRF token parameter
☐ Check CSRF by removing the CSRF token and using a blank parameter
☐ Check CSRF by using unused tokens
☐ Check CSRF by replacing the CSRF token with its own values
☐ Check CSRF by changing the content type to form-multipart
☐ Check CSRF by changing or deleting some characters of the CSRF token
☐ Check CSRF by changing the referrer to Referrer
☐ Check CSRF by changing the host values
☐ Check CSRF alongside clickjacking
7. Test For Logout Functionality
☐ Check the logout function on different pages
☐ Check for the visibility of the logout button
☐ Ensure after logout the session was ended
☐ Ensure after logout we can't able to access the dashboard by pressing the back button
☐ Ensure proper session timeout has been set

8. Test For Session Timeout
☐ Ensure there is a session timeout exists
☐ Ensure after the timeout, all of the tokens are destroyed
9. Test For Session Puzzling
☐ Identify all the session variables
☐ Try to break the logical flow of the session generation
10. Test For Session Hijacking
☐ Test session hijacking on target that doesn't has HSTS enabled
☐ Test by login with the help of captured cookies
INPUT VALIDATION TESTING
1. Test For Reflected Cross Site Scripting
☐ Ensure these characters are filtered <>"&""
Test with a character escape sequence
☐ Test by replacing < and > with HTML entities &It and >
☐ Test payload with both lower and upper case
☐ Test to break firewall regex by new line /r/n
☐ Test with double encoding
☐ Test with recursive filters
☐ Test injecting anchor tags without whitespace
☐ Test by replacing whitespace with bullets
☐ Test by changing HTTP methods

2. I	est For Stored Cross Site Scripting
	Identify stored input parameters that will reflect on the client side
	Look for input parameters on the profile page
	Look for input parameters on the shopping cart page
	Look for input parameters on the file upload page
	Look for input parameters on the settings page
	Look for input parameters on the forum, comment page
	Test uploading a file with XSS payload as its file name
	Test with HTML tags
3. T	est For HTTP Parameter Pollution
	Identify the backend server and parsing method used
	Try to access the injection point
	Try to bypass the input filters using HTTP Parameter Pollution
<b>⊿</b> Т	est For SQL Injection
	Test SQL Injection on authentication forms
	Test SQL Injection on the search bar
	Test SQL Injection on editable characteristics
	Try to find SQL keywords or entry point detections
	Try to inject SQL queries
	Use tools like SQLmap or Hackbar
	Use Google dorks to find the SQL keywords
	Try GET based SQL Injection
	Try POST based SQL Injection
	Try COOKIE based SQL Injection
	Try HEADER based SQL Injection

☐ Try SQL Injection with null bytes before the SQL query
☐ Try SQL Injection with URL encoding
☐ Try SQL Injection with both lower and upper cases
☐ Try SQL Injection with SQL Tamper scripts
☐ Try SQL Injection with SQL Time delay payloads
☐ Try SQL Injection with SQL Conditional delays
☐ Try SQL Injection with Boolean based SQL
☐ Try SQL Injection with Time based SQL
5. Test For LDAP Injection
☐ Use LDAP search filters
☐ Try LDAP Injection for access control bypass
6. Testing For XML Injection
☐ Check if the application is using XML for processing
☐ Identify the XML Injection point by XML metacharacter
☐ Construct XSS payload on top of XML
7. Test For Server Side Includes
☐ Use Google dorks to find the SSI
☐ Construct RCE on top of SSI
☐ Construct other injections on top of SSI
$\hfill \square$ Test Injecting SSI on login pages, header fields, referrer, etc
8. Test For XPATH Injection
☐ Identify XPATH Injection point
☐ Test for XPATH Injection

9. T	est For IMAP SMTP Injection
	Identify IMAP SMTP Injection point
	Understand the data flow
	Understand the deployment structure of the system
	Assess the injection impact
10.	Test For Local File Inclusion
	Look for LFI keywords
	Try to change the local path
	Use LFI payload list
	Test LFI by adding a null byte at the end
11.	Test For Remote File Inclusion
	Look for RFI keywords
	Try to change the remote path
	Use RFI payload list
12.	Test For Command Injection
	Identify the Injection points
	Look for Command Injection keywords
	Test Command Injection using different delimiters
	Test Command Injection with payload list
	Test Command Injection with different OS commands
13.	Test For Format String Injection
	Identify the Injection points
	Use different format parameters as payloads
	Assess the injection impact

14. Test For Host Header Injection
☐ Test for HHI by changing the real Host parameter
☐ Test for HHI by adding X-Forwarded Host parameter
$\hfill\square$ Test for HHI by swapping the real Host and X-Forwarded Host parameter
☐ Test for HHI by adding two Host parameters
$\hfill\Box$ Test for HHI by adding the target values in front of the original values
$\hfill\Box$ Test for HHI by adding the target with a slash after the original values
☐ Test for HHI with other injections on the Host parameter
☐ Test for HHI by password reset poisoning
15 Tost For Sanyar Sida Dagast Forgany
15. Test For Server Side Reqest Forgery
☐ Look for SSRF keywords
☐ Search for SSRF keywords only under the request header and body
☐ Identify the Injection points
☐ Test if the Injection points are exploitable
☐ Assess the injection impact
16. Test For Server Side Template Injection
☐ Identify the Template injection vulnerability points
☐ Identify the Templating engine
☐ Use the tplmap to exploit
ERROR HANDLING TESTING
1. Test For Improper Error Handling
☐ Identify the error output
☐ Analyze the different outputs returned

☐ Look for common error handling flaws
☐ Test error handling by modifying the URL parameter
☐ Test error handling by uploading unrecognized file formats
☐ Test error handling by entering unrecognized inputs
☐ Test error handling by making all possible errors
WEAK CRYPTOGRAPHY TESTING
1. Test For Weak Transport Layer Security
☐ Test for DROWN weakness on SSLv2 protocol
☐ Test for POODLE weakness on SSLv3 protocol
☐ Test for BEAST weakness on TLSv1.0 protocol
☐ Test for FREAK weakness on export cipher suites
☐ Test for Null ciphers
☐ Test for NOMORE weakness on RC4
☐ Test for LUCKY 13 weakness on CBC mode ciphers
☐ Test for CRIME weakness on TLS compression
☐ Test for LOGJAM on DHE keys
☐ Ensure the digital certificates should have at least 2048 bits of key length
☐ Ensure the digital certificates should have at least SHA - 256 signature
algorithm
☐ Ensure the digital certificates should not use MDF and SHA - 1
☐ Ensure the validity of the digital certificate
☐ Ensure the minimum key length requirements
☐ Look for weak cipher suites

## **BUSINESS LOGIC TESTING**

1. T	est For Business Logic
	Identify the logic of how the application works
	Identify the functionality of all the buttons
	Test by changing the numerical values into high or negative values
	Test by changing the quantity
	Test by modifying the payments
	Test for parameter tampering
2. T	est For Malicious File Upload
	Test malicious file upload by uploading malicious files
	Test malicious file upload by putting your IP address on the file name
	Test malicious file upload by right to left override
	Test malicious file upload by encoded file name
	Test malicious file upload by XSS payload on the file name
	Test malicious file upload by RCE payload on the file name
	Test malicious file upload by LFI payload on the file name
	Test malicious file upload by RFI payload on the file name
	Test malicious file upload by SQL payload on the file name
	Test malicious file upload by other injections on the file name
	Test malicious file upload by Inserting the payload inside of an image by the bmp.pl tool
	Test malicious file upload by uploading large files (leads to DOS)

### **CLIENT SIDE TESTING**

1. Test For DOM Based Cross Site Scripting
☐ Try to identify DOM sinks
☐ Build payloads to that DOM sink type
2. Test For URL Redirect
☐ Look for URL redirect parameters
☐ Test for URL redirection on domain parameters
☐ Test for URL redirection by using a payload list
☐ Test for URL redirection by using a whitelisted word at the end
☐ Test for URL redirection by creating a new subdomain with the same as the target
☐ Test for URL redirection by XSS
☐ Test for URL redirection by profile URL flaw
3. Test For Cross Origin Resource Sharing
☐ Look for "Access-Control-Allow-Origin" on the response
☐ Use the CORS HTML exploit code for further exploitation
4. Test For Clickjacking
☐ Ensure "X-Frame-Options" headers are enabled
Exploit with iframe HTML code for POC

## **OTHER COMMON ISSUES**

1. Test For No-Rate Limiting
☐ Ensure rate limiting is enabled
$\ \square$ Try to bypass rate limiting by changing the case of the endpoints
$\hfill\Box$ Try to bypass rate limiting by adding / at the end of the URL
☐ Try to bypass rate limiting by adding HTTP headers
☐ Try to bypass rate limiting by adding HTTP headers twice
☐ Try to bypass rate limiting by adding Origin headers
☐ Try to bypass rate limiting by IP rotation
☐ Try to bypass rate limiting by using null bytes at the end
☐ Try to bypass rate limiting by using race conditions
2. Test For EXIF Geodata
☐ Ensure the website is striping the geodata
☐ Test with EXIF checker
3. Test For Broken Link Hijack
☐ Ensure there is no broken links are there
☐ Test broken links by using the blc tool
4. Test For SPF
☐ Ensure the website is having SPF record
Test SPF by nslookup command

	est For Weak 2FA
	Γry to bypass 2FA by using poor session management
	Γry to bypass 2FA via the OAuth mechanism
	Γry to bypass 2FA via brute-forcing
	Γry to bypass 2FA via response manipulation
	Γry to bypass 2FA by using activation links to login
	Γry to bypass 2FA by using status code manipulation
	Γry to bypass 2FA by changing the email or password
	Γry to bypass 2FA by using a null or empty entry
	Γry to bypass 2FA by changing the boolean into false
	Γry to bypass 2FA by removing the 2FA parameter on the request
6 T	
<b>0</b> . I	est For Weak OTP Implementation
	ry to bypass OTP by entering the old OTP
	Try to bypass OTP by entering the old OTP
	Try to bypass OTP by entering the old OTP  Try to bypass OTP by brute-forcing
	Try to bypass OTP by entering the old OTP  Try to bypass OTP by brute-forcing  Try to bypass OTP by using a null or empty entry  Try to bypass OTP by response manipulation  Try to bypass OTP by status code manipulation
	Try to bypass OTP by entering the old OTP  Try to bypass OTP by brute-forcing  Try to bypass OTP by using a null or empty entry  Try to bypass OTP by response manipulation  Try to bypass OTP by status code manipulation
	Try to bypass OTP by entering the old OTP  Try to bypass OTP by brute-forcing  Try to bypass OTP by using a null or empty entry  Try to bypass OTP by response manipulation  Try to bypass OTP by status code manipulation
	Try to bypass OTP by entering the old OTP  Try to bypass OTP by brute-forcing  Try to bypass OTP by using a null or empty entry  Try to bypass OTP by response manipulation
	Try to bypass OTP by entering the old OTP  Try to bypass OTP by brute-forcing  Try to bypass OTP by using a null or empty entry  Try to bypass OTP by response manipulation  Try to bypass OTP by status code manipulation