Web-Pentesting-Checklist

Pre-Engagement

Recon & analysis

- ☐ Identify web server & technologies
- ☐ Subdomains Enumeration
- ☐ Directory enumeration
- ☐ Findleaked ids, emails (pwndb)
- ☐ Identify WAF
- ☐ Crawl all the site for interesting keywords like password, token, etc
- ☐ Test for debug parameters
- ☐ Identify data entry points
- ☐ Try to locate /robots.txt /crossdomain.xml /clientaccesspolicy.xml /phpinfo.php /sitemap.xml
- ☐ Review comments on source code
- ☐ Check /.git
- ☐ Shodan
- ☐ Google dorking
- ☐ Check waybackurls (gau and waybackurls)

Network tests

- ☐ Check ICMP packets allowed
- ☐ Check DMARC policies (spoofcheck)
- ☐ Look services on other ports than 80 and 443
- ☐ Check UDP ports (udp-proto-scanner or nmap)
- ☐ Test SSL (testssl)

Preparation

- ☐ Study site structure
- ☐ Make a list with all possible test cases

User management

Registration

- ☐ Duplicate registration
- ☐ Overwrite existing user (existing user takeover)
- ☐ Username uniqueness
- ☐ Weak password policy
- ☐ Insufficient email verification process
- ☐ Weak registration implementation or allows disposable email addresses
- ☐ Fuzz after user creation to check if any folder have been overwritten or created with your profile name
- ☐ Add only spaces in password

Authentication

- ☐ Username enumeration
- ☐ Resilience to password guessing
- ☐ Account recovery function
- ☐ "Remember me" function
- ☐ Impersonation function
- ☐ Unsafe distribution of credentials
- ☐ Fail-open conditions
- ☐ Multi-stage mechanisms
- ☐ SQL Injections
- ☐ Auto-complete testing
- ☐ Lack of password confirmation on change email, password or 2FA
- ☐ Weak login function over HTTP and HTTPS if both are available
- ☐ User account lockout mechanism on brute force attack
- ☐ Check for password wordlist (cewl and burp-goldenNuggets)
- ☐ Test 0auth login functionality for Open Redirection
- ☐ Test response tampering in SAMLauthentication
- ☐ In OTP check guessable codes and race conditions
- ☐ If JWT, check common flaws
- ☐ Browser cache weakness (eg Pragma, Expires, Max-age)

Session

- ☐ Session handling
- ☐ Test tokens for meaning
- ☐ Test tokens for predictability
- ☐ Insecure transmission of tokens
- ☐ Disclosure of tokens in logs
- ☐ Mapping of tokens to sessions
- ☐ Session termination
- ☐ Session fixation
- ☐ Cross-site request forgery
- ☐ Cookie scope
- ☐ Decode Cookie (Base64, hex, URL etc.)
- ☐ Cookie expiration time
- ☐ Check HTTPOnly and Secure flags
- ☐ Use same cookie from a different effective IP address or system
- ☐ Access controls
- ☐ Effectiveness of controls using multiple accounts
- ☐ Insecure access control methods (request parameters, Referer header, etc)
- ☐ Check for concurrent login through different machine/IP
- ☐ Bypass AntiCSRFtokens

Profile/Account details

- ☐ Find parameter with user id and try to tamper in order to get the details of other users
- ☐ Create a list of features that are pertaining to a user account only and try CSRF
- ☐ Change email id and update with any existing email id. Check if its getting validated on server or not.
- ☐ Check any new email confirmation link and what if user doesn't confirm.
- ☐ File upload: Unsafe File upload, No Antivirus, No Size Limit, File extension, Filter Bypass, burp
- ☐ CSV import/export: Command Injection, XSS, macro injection
- ☐ Check profile picture URL and find email id/user info or EXIF Geolocation Data

- ☐ Imagetragick in picture profile upload
- ☐ Metadataof all downloadable files
- ☐ Account deletion option and try to reactivate with "Forgot password" feature
- ☐ Try bruteforce enumeration when change any user unique parameter.
- ☐ Check application request re-authentication for sensitive operations
- ☐ Try parameter pollution to add two values of same field

Input handling

- ☐ Fuzz all request parameters
- ☐ Identify all reflected data
- ☐ Reflected XSS
- ☐ HTTPheader injection in GET & POST (X Forwarded Host)
- ☐ Arbitrary redirection
- ☐ Stored attacks
- ☐ OS command injection
- ☐ Path traversal
- ☐ Script injection
- ☐ File inclusion
- ☐ SMTP injection
- ☐ Native software flaws (buffer overflow, integer bugs, format strings)
- ☐ SOAP injection
- ☐ LDAP injection
- ☐ XPath injection
- ☐ XXE in any request, change content-type to text/xml
- ☐ Stored XSS
- ☐ SQLinjection
- ☐ NoSQLinjection
- ☐ HTTP Request Smuggling
- ☐ Open redirect
- ☐ SSRFin previously discovered open ports
- ☐ xmlrpc.php DOS and user enumeration
- ☐ HTTP dangerous methods OPTIONS PUT DELETE

Forgot password

- ☐ Invalidate session on Logout and Password reset
- ☐ Uniqueness of forget password reset link/code
- ☐ Reset links expiration time
- ☐ Find user id or other sensitive fields in reset link and tamper them
- ☐ Request 2 reset passwords links and use the older
- ☐ Check if many requests have sequential tokens

Error handling

- ☐ Access custom pages like /whatever_fake.php (.aspx,.html,.etc)
- ☐ Add multiple parameters in GET and POST request using different values
- ☐ Add "[]", "]]", and "[[" in cookie values and parameter values to create errors
- ☐ Generate error by giving input as "/~randomthing/%s" at the end of URL
- ☐ Use Burp Intruder "Fuzzing Full" List in input to generate error codes
- ☐ Try different HTTP Verbs like PATCH, DEBUG or wrong like FAKE

Application Logic

- ☐ Identify the logic attack surface
- ☐ Test transmission of data via the client
- ☐ Test for reliance on client-side input validation
- ☐ Thick-client components (Java, ActiveX, Flash)
- ☐ Multi-stage processes for logic flaws
- ☐ Handling of incomplete input
- ☐ Trust boundaries
- ☐ Transaction logic
- ☐ Implemented CAPTCHA in email forms to avoid flooding
- ☐ Tamper product id, price, or quantity value in any action (add, modify, delete, place, pay...)
- ☐ Tamper gift or discount codes
- ☐ Reuse gift codes
- ☐ Try parameter pollution to use gift code two times in same request
- ☐ Try stored XSS in non-limited fields like address
- ☐ Check in payment form if CVV and card number is in clear text or masked

- ☐ Check if is processed by the app itself or sent to 3rd parts
- ☐ IDOR from other users details ticket/cart/shipment
- ☐ Check PRINT or PDF creation for IDOR
- ☐ Check unsubscribe button with user enumeration
- ☐ Parameter pollution on social media sharing links
- ☐ CORS (corsy)
- ☐ Change POST sensitive requests to GET

Other checks

Hosting

- ☐ Segregation in shared infrastructures
- ☐ Segregation between ASP-hosted applications
- ☐ Web server vulnerabilities
- ☐ Dangerous HTTP methods
- ☐ Proxy functionality
- ☐ Virtualhosting misconfiguration
- ☐ Check for internal numeric IP's in request
- ☐ Check for external numeric IP's and resolve it
- ☐ References to cloudassets

CAPTCHA

- ☐ Send old captcha value.
- ☐ Send old captcha value with old session ID.
- ☐ Request captcha absolute path like www.url.com/captcha/1.png
- ☐ Remove captcha with any adblocker and request again
- ☐ Bypass with OCR tool

Headers

- ☐ X-XSS-Protection
- ☐ Strict-Transport-Security
- ☐ Content-Security-Policy
- ☐ Public-Key-Pins
- ☐ X-Frame-Options
- ☐ X-Content-Type-Options
- ☐ Referer-Policy
- ☐ Cache-Control
- ☐ Expires