Comprehensive Web Application Bug Bounty Checklist

by beta0x01 @ X | @ GitHub

1. Reconnaissance

- [ ] **Identify Web Server, Technologies, and Database**
- [ ] **Subsidiary and Acquisition Enumeration**
- [ ] **Reverse Lookup**
- [ ] **ASN & IP Space Enumeration and Service Enumeration**
- [ ] **Google Dorking**
- [ ] **GitHub Recon**
- [ ] **Directory Enumeration**
- [ ] **IP Range Enumeration**
- [ ] **JS Files Analysis**
- [ ] **Subdomain Enumeration and Bruteforcing**
- [ ] **Subdomain Takeover**
- [ ] **Parameter Fuzzing**
- [ ] **Port Scanning**
- [ ] **Template-Based Scanning (Nuclei)**
- [ ] **Wayback Machine History**
- [ ] **Broken Link Hijacking**
- [ ] **Internet Search Engine Discovery**
- [ ] **Misconfigured Cloud Storage**
- [ ] **Identify underlying web client and server technology**
- [ ] **Uncover HTTP/HTTPS services running on ports other than 80 and 443**
- [ ] **Find leaked email IDs, passwords using ‘We Leak Info’ and ‘Hunter.io’**
- [ ] **Identify firewall**
- [ ] **Find sensitive information through keywords after crawling entire site** (Keywords: admin, password, todo, http)

2. Network Testing

- [ ] **Test for Ping (ICMP Packets Allowed or Filtered)**
- [ ] **DNS Testing for Zone Transfer, Missing DNSSEC Policies**
- [ ] **Missing DMARC Policies**
- [ ] **Perform Nessus Scan**
- [ ] **Banner Disclosure for Open Ports and Network Services**
- [ ] **Find All Web and Network Services Other Than Port 80 and 443**
- [ ] **Perform UDP Scan Using UDP Proto Scanner**

3. Application Features Mapping

- [ ] **Generate Site Structure in Any Mindmap Tool**
- [ ] **List All Dynamic Features**
- [ ] **Add All Possible Theoretical Test Cases Within Your Mind Map for Testing Security of Those Features**

4. Application Component Audit

- [ ] **Test SSL/TLS Weaknesses Using Qualys SSL Scanner**
- [ ] **Identify Known Vulnerabilities in Running Web and Network Components Using Known CVE, Searchsploits, Metasploit Auxiliaries, and Exploits**

5. Session Management Testing

- [ ] **Identify Actual Session Cookie Out of Bulk Cookies in the Application**
- [ ] **Decode Cookies Using Some Standard Decoding Algorithms Such as Base64, Hex, URL, etc.**
- [ ] **Modify Cookie Session Token Value by 1 Bit/Byte. Then Resubmit and Do the Same for All Tokens. Reduce the Amount of Work You Need to Perform in Order to Identify Which Part of Token Is Actually Being Used and Which Is Not**
- [ ] **If Self-Registration Is Available and You Can Choose Your Username, Log In with a Series of Similar Usernames Containing Small Variations Between Them, Such as A, AA, AAA, AAAA, AAAB, AAAC, AABA, and So On. If Other User-Specific Data Is Submitted at Login or Stored in User Profiles (Such as an Email Address)**
- [ ] **Token Leakage via Referer Header - Untrusted 3rd Party**
- [ ] **Check for Session Cookies and Cookie Expiration Date/Time**
- [ ] **Identify Cookie Domain Scope**
- [ ] **Check for HttpOnly Flag in Cookie**
- [ ] **Check for Secure Flag in Cookie If the Application Is Over SSL**
- [ ] **Check for Session Fixation - Value of Session Cookie Before and After Authentication**
- [ ] **Replay the Session Cookie from a Different Effective IP Address or System to Check Whether the Server Maintains the State of the Machine or Not**
- [ ] **Check for Concurrent Login Through Different Machine/IP**
- [ ] **Check If Any User Pertaining Information Is Stored in Cookie Value or Not. If Yes, Tamper It with Other User’s Data**

6. Registration Feature Testing

- [ ] **Check for Duplicate Registration/Overwrite Existing User**
- [ ] **Check for Weak Password Policy**
- [ ] **Check for Reuse Existing Usernames**
- [ ] **Check for Insufficient Email Verification Process**
- [ ] **Weak Registration Implementation - Allows Disposable Email Addresses**
- [ ] **Weak Registration Implementation - Over HTTP**
- [ ] **Overwrite Default Web Application Pages by Specially Crafted Username Registrations**
    - [ ] **After Registration, Does Your Profile Link Appear Something as www.chintan.com/chintan?**

        - [ ] **If So, Enumerate Default Folders of Web Application Such as /images, /contact, /portfolio**
        - [ ] **Do a Registration Using the Username Such as Images, Contact, Portfolio**
        - [ ] **Check If Those Default Folders Have Been Overwritten by Your Profile Link or Not**

7. Authentication Testing

- [ ] **Username Enumeration**
- [ ] **Bypass Authentication Using Various SQL Injections on Username and Password Fields**
    - [ ] **Use Combinations of Below Injections:**
        - [ ] `chintan' --`
        - [ ] `chintan' #`
        - [ ] `chintan'/*`
        - [ ] `' or 1=1 --`
        - [ ] `' or 1=1 #`
        - [ ] `' or 1=1/*`
        - [ ] `') or '1'='1 --`
        - [ ] `') or ('1'='1 --`
- [ ] **Auto-Complete Testing**
- [ ] **Lack of Password Confirmation on**
    - [ ] **Change Email Address**
    - [ ] **Change Password**
    - [ ] **Manage 2FA**
- [ ] **Is It Possible to Use Resources Without Authentication? Access Violation**
- [ ] **Check If User Credentials Are Transmitted Over SSL or Not**
- [ ] **Weak Login Function - HTTP and HTTPS Both Are Available**
- [ ] **Test User Account Lockout Mechanism on Brute Force Attack**
    - [ ] **Variation: If Server Blocks Instant User Requests, Then Try with Time Throttle Option from Intruder and Repeat the Process Again**
    - [ ] **Bypass Rate Limiting by Tampering User Agent to Mobile User Agent**
    - [ ] **Bypass Rate Limiting by Tampering User Agent to Anonymous User Agent**
    - [ ] **Bypass Rate Limiting by Using Null Byte**
- [ ] **Create a Password Wordlist Using Cewl Command**
- [ ] **Test OAuth Login Functionality**
    - [ ] **OAuth Roles**
        - [ ] **Resource Owner → User**
        - [ ] **Resource Server → Twitter**
        - [ ] **Client Application → Twitterdeck.com**
        - [ ] **Authorization Server → Twitter**
        - [ ] **client_id → Twitterdeck ID (This is a public, non-secret unique identifier)**
        - [ ] **client_secret → Secret Token Known to the Twitter and Twitterdeck to Generate Access_Tokens**
        - [ ] **response_type → Defines the Token Type (e.g., code, token, etc.)**
        - [ ] **scope → The Requested Level of Access Twitterdeck Wants**
        - [ ] **redirect_uri → The URL User Is Redirected to After the Authorization Is Complete**

        - [ ] state → Main CSRF Protection in OAuth Can Persist Data Between the User Being Directed to the Authorization Server and Back Again
        - [ ] grant_type → Defines the Grant_Type and the Returned Token Type
        - [ ] code → The Authorization Code Twitter Generated, Will Be Like ?code=, the Code Is Used with Client_ID and Client_Secret to Fetch an Access_Token
        - [ ] access_token → The Token Twitterdeck Uses to Make API Requests on Behalf of the User
        - [ ] refresh_token → Allows an Application to Obtain a New Access_Token Without Prompting the User
    - [ ] Code Flaws
        - [ ] Re-Using the Code
        - [ ] Code Predict/Bruteforce and Rate-Limit
        - [ ] Is the Code for Application X Valid for Application Y?
    - [ ] Redirect_URI Flaws
        - [ ] URL Isn’t Validated at All: ?redirect_uri=https://attacker.com
        - [ ] Subdomains Allowed (Subdomain Takeover or Open Redirect on Those Subdomains): ?redirect_uri=https://sub.twitterdeck.com
        - [ ] Host Is Validated, Path Isn’t (Chain Open Redirect): ?redirect_uri=https://twitterdeck.com/callback?redirectUrl=https://evil.com
        - [ ] Host Is Validated, Path Isn’t (Referer Leakages): Include External Content on HTML Page and Leak Code via Referer
        - [ ] Weak Regexes
            - [ ] Bruteforcing the URL Encoded Chars After Host: redirect_uri=https://twitterdeck.com§FUZZ§
            - [ ] Bruteforcing the Keywords Whitelist After Host (or on Any Whitelist Open Redirect Filter): ?redirect_uri=https://§FUZZ§.com
            - [ ] URI Validation in Place: Use Typical Open Redirect Payloads
    - [ ] State Flaws
        - [ ] Missing State Parameter? (CSRF)
        - [ ] Predictable State Parameter?
        - [ ] Is State Parameter Being Verified?
    - [ ] Misc
        - [ ] Is Client_Secret Validated?
        - [ ] Pre ATO Using Facebook Phone-Number Signup
        - [ ] No Email Validation Pre ATO
    - [ ] Test 2FA Misconfiguration
        - [ ] Response Manipulation
        - [ ] Status Code Manipulation
        - [ ] 2FA Code Leakage in Response
        - [ ] 2FA Code Reusability
        - [ ] Lack of Brute-Force Protection
        - [ ] Missing 2FA Code Integrity Validation
            - [ ] With Null or 000000

8. Error Codes Testing

- [ ] Generate Custom Pages Such as /chintan.php, chintan.aspx and Identify Error Page

- [ ] **Add Multiple Parameters in Same Post Get Request Using Different Values and Generate Error**
- [ ] **Add [],]] and [[in Cookie Values and Parameter Values to Create Errors**
- [ ] **Try to Generate Unusual Error Code by Giving Input as /~chintan/%s at the End of Website URL**
- [ ] **Fuzz Using the Burp Intruder with Malicious Input and Try to Generate Error Codes**

9. My Account (Post Login) Testing

- [ ] **Find Parameter Which Uses Active Account User ID. Try to Tamper It in Order to Change the Details of Other Account**
- [ ] **Create a List of Features That Are Pertaining to a User Account Only**
  - [ ] **Change Email**
  - [ ] **Change Password**
  - [ ] **Change Account Details (Name, Number, Address, etc.)**
  - [ ] **Try CSRF**
- [ ] **Post Login Change Email ID and Update with Any Existing Email ID. Check If Its Getting Validated on Server Side or Not. Does the Application Send Any New Email Confirmation Link to a New User or Not? What If a User Does Not Confirm the Link in Some Time Frame?**
- [ ] **Perform All File Upload Tests Using Extension Tampering and File Content Modifying**
  - [ ] **Unsafe File Upload - No Antivirus - No Size Limit - File Extension Filter Bypass**
- [ ] **Open Profile Picture in New Tab and Check the URL. Find Email ID/User ID Info. EXIF Geolocation Data Not Stripped From Uploaded Images**
- [ ] **Check Account Deletion Option If Application Provides It and Confirm That Via Forgot Password Feature**
- [ ] **Change Email ID, Account ID, User ID Parameter and Try to Brute Force Other User’s Password**
- [ ] **Check Whether Application Re-Authenticates for Performing Sensitive Operation for Post Authentication Features**

10. Forgot Password Testing

- [ ] **Failure to Invalidate Session on Logout and Password Reset**
- [ ] **Check If Forget Password Reset Link/Code Uniqueness**
- [ ] **Check If Reset Link Does Get Expire or Not If Its Not Used by the User for Certain Amount of Time**
- [ ] **Find User Account Identification Parameter and Tamper Id or Parameter Value to Change Other User’s Password**
- [ ] **Check for Weak Password Policy**
- [ ] **Weak Password Reset Implementation - Token Is Not Invalidated After Use**
- [ ] **If Reset Link Has Another Params Such as Date and Time, Then. Change Date and Time Value in Order to Make Active & Valid Reset Link**
- [ ] **Check If Security Questions Are Asked? How Many Guesses Allowed? -> Lockout Policy Maintained or Not?**
- [ ] **Add Only Spaces in New Password and Confirmed Password. Then Hit Enter and See the Result**

- [ ] **Does It Display Old Password on the Same Page After Completion of Forget Password Formality?**
- [ ] **Ask for Two Password Reset Links and Use the Older One from User's Email**
- [ ] **Check If Active Session Gets Destroyed Upon Changing the Password or Not?**
- [ ] **Weak Password Reset Implementation - Password Reset Token Sent Over HTTP**
- [ ] **Send Continuous Forget Password Requests So That It May Send Sequential Tokens**

11. Contact Us Form Testing

- [ ] **Is CAPTCHA Implemented on Contact Us Form in Order to Restrict Email Flooding Attacks?**
- [ ] **Does It Allow to Upload File on the Server?**

12. Product Purchase Testing

- [ ] **Buy Now**
  - [ ] **Tamper Product ID to Purchase Other High-Valued Product with Low Prize**
  - [ ] **Tamper Product Data in Order to Increase the Number of Product with the Same Prize**
- [ ] **Gift/Voucher**
  - [ ] **Tamper Gift/Voucher Count in the Request (If Any) to Increase/Decrease the Number of Vouchers/Gifts to Be Used**
  - [ ] **Tamper Gift/Voucher Value to Increase/Decrease the Value of Voucher in Terms of Money. (E.g., $100 Is Given as a Voucher, Tamper Value to Increase, Decrease Money)**
  - [ ] **Reuse Gift/Voucher by Using Old Gift Values in Parameter Tampering**
  - [ ] **Check the Uniqueness of Gift/Voucher Parameter and Try Guessing Other Gift/Voucher Code**
  - [ ] **Use Parameter Pollution Technique to Add Same Voucher Twice by Adding Same Parameter Name and Value Again with & in the BurpSuite Request**
- [ ] **Add/Delete Product from Cart**
  - [ ] **Tamper User ID to Delete Products from Other User's Cart**
  - [ ] **Tamper Cart ID to Add/Delete Products from Other User's Cart**
  - [ ] **Identify Cart ID/User ID for Cart Feature to View the Added Items from Other User's Account**
- [ ] **Address**
  - [ ] **Tamper BurpSuite Request to Change Other User's Shipping Address to Yours**
  - [ ] **Try Stored-XSS by Adding XSS Vector on Shipping Address**
  - [ ] **Use Parameter Pollution Technique to Add Two Shipping Addresses Instead of One Trying to Manipulate Application to Send Same Item on Two Shipping Addresses**
- [ ] **Place Order**
  - [ ] **Tamper Payment Options Parameter to Change the Payment Method. E.g., Consider Some Items Cannot Be Ordered for Cash on Delivery But Tampering Request Parameters from Debit/Credit/PayPal/Net Banking Option to Cash on Delivery May Allow You to Place Order for That Particular Item**
  - [ ] **Tamper the Amount Value for Payment Manipulation in Each Main and Sub Requests and Responses**
  - [ ] **Check If CVV Is Going in Cleartext or Not**
  - [ ] **Check If Credit/Debit Card Details Are Masked or Not**

    - [ ] **Check If Application Itself Processes Your Card Details and Then Performs Transaction or It Calls Any Third-Party Payment Processing Company to Perform Transaction**
- [ ] **Track Order**
    - [ ] **Track Other User’s Order by Guessing Order Tracking Number**
    - [ ] **Brute Force Tracking Number Prefix or Suffix to Track Mass Orders for Other Users**

13. Wish List Page Testing

- [ ] **Check If User A Can Add/Remove Products in Wishlist of Other User B’s Account**
- [ ] **Check If User A Can Add Products into User B’s Cart from His/Her (User A’s) Wishlist Section**

14. Post Product Purchase Testing

- [ ] **Check If User A Can Cancel Orders for User B’s Purchase**
- [ ] **Check If User A Can View/Check Orders Already Placed by User B**
- [ ] **Check If User A Can Modify the Shipping Address of Placed Order by User B**

15. Flight/Railway/Hotel Booking Testing

- [ ] **Booking Details**
    - [ ] **View/Manage Other User’s Booking Details**
    - [ ] **Check Reservation Status for Other Users/Behalf of Other Users**
- [ ] **Ticket/Voucher**
    - [ ] **View Other Users Vouchers/E-Tickets from PRINT Option**
    - [ ] **Check If Sensitive Data Is Passed in GET Request**
    - [ ] **If E-Ticket/Voucher Is Sent on Email Then Check for the Email Flooding Attack**
- [ ] **Refund**
    - [ ] **View Other User’s Refund Status**
    - [ ] **Refund More Money Than the Intended One by Parameter Manipulation**
    - [ ] **If Refund Tracking Is Allowed Then Gain Other User’s Refund Tracking Status**
- [ ] **Cancellation**
    - [ ] **Gain Higher Cancellation Amount with Parameter Modifying for Amount Value**
- [ ] **Booking**
    - [ ] **Do 1st Person Booking and Add 3 Other Persons in Same Prize**
    - [ ] **Hotel - Book Normal Room - Select Deluxe Room in the Same Prize**

16. Cross-Site Scripting (XSS) Testing

- [ ] **Locator: ‘’ ;!— “<chintan>=&{()}**
- [ ] **Try XSS Using XSSStrike Tool by Somdev Sangwan**
- [ ] **Upload File Using ‘ “><img src=x onerror=alert(document.domain)>.txt**
- [ ] **Standard Payload for URI and All Inputs:**
    - [ ] `"><img src=x onerror=prompt(document.cookie);><!--`
    - [ ] `"><img src=x onerror=confirm(document.cookie);><!--`
    - [ ] `"><img src=x onerror=alert(document.cookie);><!--`
- [ ] **If Script Tags Are Banned, Use <h1> and Other HTML Tags**

- [ ] **If Output Is Reflected Back Inside the JavaScript as a Value of Any Variable Just Use Alert(1)**
- [ ] **If " Are Filtered Then Use This Payload /><img src=d onerror=confirm(/chintan/);>`**
- [ ] **Upload a JavaScript Using Image File**
- [ ] **Unusual Way to Execute Your JS Payload Is to Change Method from POST to GET. It Bypasses Filters Sometimes**
- [ ] **Tag Attribute Value**
  - [ ] **Input Landed - [ ] <input type=" text" name=" state" value=" INPUT_FROM_USER" >**
  - [ ] **Payload to Be Inserted - [ ] " onfocus=" alert(document.cookie)"**
- [ ] **Syntax Encoding Payload "%3cscript%3ealert(document.cookie)%3c/script%3e"**
- [ ] **ASP.NET IE9 chintan Filter Evasion for HTML Entities**
  - [ ] `<%tag style="chintan:expression(alert('chintan'))">`
  - [ ] `<%tag style=" chintan:expression(alert(123))``
  - [ ] `<%tag style="chintan:expression(alert(123))"`
- [ ] **Try Base64 Payload**
- [ ] **If the Logout Button Just Performs the Redirection Then Use Old Classic XSS Payload**
- [ ] **Polyglot Payload**
- [ ] **Use Pure JS Payload That Worked for Many Popular Websites If Your Input Is Reflected Back in the JavaScript**
- [ ] **Blind XSS**

17. SQL Injection Testing

- [ ] **Locator (Error Based)**
- [ ] **Test' " " 123' "" Þ}j%Ùÿ' " " ' "" " " ';' ' "" ();=,%+-// —«****
- [ ] **If Parameter=Static_Integer_Value Then Follow Below Method. If Id=4, Then Try Id=3+1 or Id=6-2 (If Page Loads in Same Way, It Is Vulnerable)**
- [ ] **Use SQLmap to Identify Vulnerable Parameters**
  - [ ] **Fill Form in Browser GUI Submit It Normally**
  - [ ] **Go to History Tab in BurpSuite and Find the Relevant Request**
  - [ ] **Right Click and Select the Option "Copy to File"**
  - [ ] **Save File as Anyname.txt**
  - [ ] **SQLmap Command to Run**
  - [ ] **Python sqlmap.py -r ~/Desktop/textsqli.txt —proxy=http://127.0.0.1:8080**
  - [ ] **Run SQL Injection Scanner on All Requests**
- [ ] **Bypassing WAF**
  - [ ] **Using Null Byte Before SQL Query**
  - [ ] **Using SQL Inline Comment Sequence**
  - [ ] **URL Encoding**
  - [ ] **Changing Cases (Uppercase/Lowercase)**
  - [ ] **Use SQLMAP Tamper Scripts**
  - [ ] **Time Delays**
    - [ ] **Oracle dbms_pipe.receive_message(('a'),10)**
    - [ ] **Microsoft WAITFOR DELAY '0:0:10'**
    - [ ] **PostgreSQL SELECT pg_sleep(10)**
    - [ ] **MySQL SELECT sleep(10)**

- [ ] Conditional Delays
  - [ ] Oracle SELECT CASE WHEN (YOUR-CONDITION-HERE) THEN 'a'||dbms_pipe.receive_message(('a'),10) ELSE NULL END FROM dual
  - [ ] Microsoft IF (YOUR-CONDITION-HERE) WAITFOR DELAY '0:0:10'
  - [ ] PostgreSQL SELECT CASE WHEN (YOUR-CONDITION-HERE) THEN pg_sleep(10) ELSE pg_sleep(0) END
  - [ ] MySQL SELECT IF(YOUR-CONDITION-HERE,sleep(10),'a')

18. Open Redirection Testing

- [ ] Use Burp 'Find' Option in Order to Find Parameters Such as URL, Red, Redirect, Redir, Origin, Redirect_Uri, Target, etc.
- [ ] Check the Value of These Parameters Which May Contain a URL
- [ ] Change the URL Value to www.chintan.com and Check If Gets Redirected or Not
- [ ] Give Below URL in Web Browser and Check If Application Redirects to the www.chintan.com Website or Not
  - [ ] https://www.target.com/ÿ/www.twitter.com/
  - [ ] https://www.target.com//www.twitter.com/
  - [ ] https://www.target.com/Ã¿/www.twitter.com/
  - [ ] https://www.target.com//www.twitter.com/
- [ ] Bypass Filter Using ReturnTo=///chintan.com/
- [ ] Bypass Filter Using ReturnTo=http:///chintan.com/

19. Host Header Injection

- [ ] Insert New Header in the GET/POST Request as Follows:
  - [ ] X-Forwarded-Host: www.chintan.com
- [ ] If It Gets Redirected from the Target Application Then Its Vulnerable
- [ ] Capture Any Request
- [ ] Change the Host to Google.com and See If Its Getting Redirected or Not

20. ASP.NET Application Testing

- [ ] Check If ASP.net ViewState Parameter Is Encrypted or Not
- [ ] Check If Any ASP Configuration Is Disclosed Publicly or Not
- [ ] Check If Error Codes Reveal the Version of ASP.NET Used in the Application

21. Cross-Site Request Forgery (CSRF) Testing

- [ ] Re-Use Anti-CSRF Token for CSRF Attack
- [ ] Check If Token Is Validated on Server Side or Not
- [ ] Check If Token Validation for Full Length or Partial Length
- [ ] Create Few Dummy Accounts and Compare the CSRF Token for All Those Accounts
- [ ] Bypass CSRF Token Using 2 Input Type Fields in For Updating User's Information in the Same HTML File
- [ ] Convert POST Request to GET and Remove _csrf (Anti-CSRF Token) to Bypass the CSRF Protection

- [ ] **Check If the Value You Are Trying to Change Is Passed in Multiple Parameters Such as Cookie, HTTP Headers Along With GET and POST Request**

22. XML Injection Testing

- [ ] **Change the Content Type to Text/XML Then Insert Below Code. Check Via Repeater**
  - [ ] `<?xml version="1.0" encoding="ISO-8859-1"?><!DOCTYPE chintan [<!ELEMENT chintan ANY ><!ENTITY xxe SYSTEM "file:///etc/passwd" >]><chintan>&xxe;</foo>`
  - [ ] `<?xml version="1.0" encoding="ISO-8859-1"?><!DOCTYPE tushar [<!ELEMENT tushar ANY ><!ENTITY xxe SYSTEM "file:///etc/hosts" >]><tushar>&xxe;</foo>`
  - [ ] `<?xml version="1.0" encoding="ISO-8859-1"?><!DOCTYPE tushar [<!ELEMENT tushar ANY ><!ENTITY xxe SYSTEM "file:///proc/self/cmdline" >]><tushar>&xxe;</foo>`
  - [ ] `<?xml version="1.0" encoding="ISO-8859-1"?><!DOCTYPE tushar [<!ELEMENT tushar ANY ><!ENTITY xxe SYSTEM "file:///proc/version" >]><tushar>&xxe;</foo>`
- [ ] **Blind XXE with Out-of-Band Interaction**

23. Cross-Origin Resource Sharing (CORS)

- [ ] **Errors Parsing Origin Headers**
- [ ] **Whitelisted Null Origin Value**

24. Server-Side Request Forgery (SSRF)

- [ ] **Try Basic Localhost Payloads**
- [ ] **Bypassing Filters**
  - [ ] **Bypass Using HTTPS**
  - [ ] **Bypass with [::]**
  - [ ] **Bypass with a Domain Redirection**
  - [ ] **Bypass Using a Decimal IP Location**
  - [ ] **Bypass Using IPv6/IPv4 Address Embedding**
  - [ ] **Bypass Using Malformed URLs**
  - [ ] **Bypass Using Rare Address (Short-Hand IP Addresses by Dropping the Zeros)**
  - [ ] **Bypass Using Enclosed Alphanumerics**
- [ ] **Cloud Instances**
  - [ ] **AWS**
    - [ ] **http://instance-data**
    - [ ] **http://169.254.169.254**
    - [ ] **http://169.254.169.254/latest/user-data**
    - [ ] **http://169.254.169.254/latest/user-data/iam/security-credentials/[ROLE NAME]**
    - [ ] **http://169.254.169.254/latest/meta-data/**
    - [ ] **http://169.254.169.254/latest/meta-data/iam/security-credentials/[ROLE NAME]**
    - [ ] **http://169.254.169.254/latest/meta-data/iam/security-credentials/PhotonInstance**
    - [ ] **http://169.254.169.254/latest/meta-data/ami-id**
    - [ ] **http://169.254.169.254/latest/meta-data/reservation-id**
    - [ ] **http://169.254.169.254/latest/meta-data/hostname**
    - [ ] **http://169.254.169.254/latest/meta-data/public-keys/**

        - [ ] http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key
        - [ ] http://169.254.169.254/latest/meta-data/public-keys/[ID]/openssh-key
        - [ ] http://169.254.169.254/latest/meta-data/iam/security-credentials/dummy
        - [ ] http://169.254.169.254/latest/meta-data/iam/security-credentials/s3access
        - [ ] http://169.254.169.254/latest/dynamic/instance-identity/document
    - [ ] Google Cloud
        - [ ] http://169.254.169.254/computeMetadata/v1/
        - [ ] http://metadata.google.internal/computeMetadata/v1/
        - [ ] http://metadata/computeMetadata/v1/
        - [ ] http://metadata.google.internal/computeMetadata/v1/instance/hostname
        - [ ] http://metadata.google.internal/computeMetadata/v1/instance/id
        - [ ] http://metadata.google.internal/computeMetadata/v1/project/project-id
    - [ ] DigitalOcean
        - [ ] curl http://169.254.169.254/metadata/v1/id
        - [ ] http://169.254.169.254/metadata/v1.json
        - [ ] http://169.254.169.254/metadata/v1/
        - [ ] http://169.254.169.254/metadata/v1/id
        - [ ] http://169.254.169.254/metadata/v1/user-data
        - [ ] http://169.254.169.254/metadata/v1/hostname
        - [ ] http://169.254.169.254/metadata/v1/region
        - [ ] http://169.254.169.254/metadata/v1/interfaces/public/0/ipv6/address
    - [ ] Azure
        - [ ] http://169.254.169.254/metadata/v1/maintenance
        - [ ] http://169.254.169.254/metadata/instance?api-version=2017-04-02
        - [ ] http://169.254.169.254/metadata/instance/network/interface/0/ipv4/ipAddress/0/publicIpAddress?api-version=2017-04-02&format=text

25. Bypassing via Open Redirection

- [ ] File Upload Testing
    - [ ] Upload the Malicious File to the Archive Upload Functionality and Observe How the Application Responds
    - [ ] Upload a File and Change Its Path to Overwrite an Existing System File
    - [ ] Large File Denial of Service
- [ ] Metadata Leakage
- [ ] ImageMagick Library Attacks
- [ ] Pixel Flood Attack
- [ ] Bypasses
    - [ ] Null Byte (%00) Bypass
    - [ ] Content-Type Bypass
    - [ ] Magic Byte Bypass
    - [ ] Client-Side Validation Bypass
    - [ ] Blacklisted Extension Bypass
    - [ ] Homographic Character Bypass

26. CAPTCHA Testing

- [ ] **Missing Captcha Field Integrity Checks**
- [ ] **HTTP Verb Manipulation**
- [ ] **Content Type Conversion**
- [ ] **Reusable Captcha**
- [ ] **Check If Captcha Is Retrievable with the Absolute Path Such as www.tushar.com/internal/captcha/images/24.png**
- [ ] **Check for the Server-Side Validation for CAPTCHA. Remove Captcha Block from GUI Using Firebug Addon and Submit Request to the Server**
- [ ] **Check If Image Recognition Can Be Done with OCR Tool?**

27. JWT Token Testing

- [ ] **Brute-Forcing Secret Keys**
- [ ] **Signing a New Token with the "none" Algorithm**
- [ ] **Changing the Signing Algorithm of the Token (for Fuzzing Purposes)**
- [ ] **Signing the Asymmetrically-Signed Token to Its Symmetric Algorithm Match (When You Have the Original Public Key)**

28. WebSockets Testing

- [ ] **Intercepting and Modifying WebSocket Messages**
- [ ] **WebSockets MITM Attempts**
- [ ] **Testing Secret Header WebSocket**
- [ ] **Content Stealing in WebSockets**
- [ ] **Token Authentication Testing in WebSockets**

29. GraphQL Vulnerabilities Testing

- [ ] **Inconsistent Authorization Checks**
- [ ] **Missing Validation of Custom Scalars**
- [ ] **Failure to Appropriately Rate-Limit**
- [ ] **Introspection Query Enabled/Disabled**

30. WordPress Common Vulnerabilities

- [ ] **XSPA in WordPress**
- [ ] **Bruteforce in wp-login.php**
- [ ] **Information Disclosure WordPress Username**
- [ ] **Backup File wp-config Exposed**
- [ ] **Log Files Exposed**
- [ ] **Denial of Service via load-styles.php**
- [ ] **Denial of Service via load-scripts.php**
- [ ] **DDoS Using xmlrpc.php**
- [ ] **Denial of Service**
  - [ ] **Cookie Bomb**
  - [ ] **Pixel Flood, Using Image with a Huge Pixels**
  - [ ] **Frame Flood, Using GIF with a Huge Frame**
  - [ ] **ReDoS (Regex DoS)**

- [ ] **CPDoS (Cache Poisoned Denial of Service)**

31. Other Test Cases (All Categories)

- [ ] **Check for Security Headers and At Least:**
  - [ ] **X-Frame-Options**
  - [ ] **X-XSS Header**
  - [ ] **HSTS Header**
  - [ ] **CSP Header**
  - [ ] **Referrer-Policy**
  - [ ] **Cache Control**
  - [ ] **Public Key Pins**
- [ ] **Testing for Role Authorization**
  - [ ] **Check If Normal User Can Access the Resources of High Privileged Users?**
  - [ ] **Forced Browsing**
  - [ ] **Insecure Direct Object Reference**
  - [ ] **Parameter Tampering to Switch User Account to High Privileged User**
- [ ] **Blind OS Command Injection**
  - [ ] **Using Time Delays**
  - [ ] **By Redirecting Output**
  - [ ] **With Out-of-Band Interaction**
  - [ ] **With Out-of-Band Data Exfiltration**
- [ ] **Command Injection on CSV Export (Upload/Download)**
- [ ] **CSV Excel Macro Injection**
- [ ] **If You Find phpinfo.php File, Check for the Configuration Leakage and Try to Exploit Any Network Vulnerability**
- [ ] **Parameter Pollution - Social Media Sharing Buttons**
- [ ] **Broken Cryptography**
  - [ ] **Cryptography Implementation Flaw**
  - [ ] **Encrypted Information Compromised**
  - [ ] **Weak Ciphers Used for Encryption**
- [ ] **Web Services Testing**
  - [ ] **Test for Directory Traversal**
  - [ ] **Web Services Documentation Disclosure - Enumeration of Services, Data Types, Input Types Boundaries and Limits**

32. Automated Scanner

- [ ] **Run Automated Scanner at Least**
  - [ ] **Netsparker**
  - [ ] **BurpSuite Scanner**
  - [ ] **For WordPress – Pecan; For Joomla – Groomsman**
  - [ ] **Nessus for Network Services Scan**
  - [ ] **Nexpose for Network Services Scan**

33. Banking Application Testing

- [ ] **Billing Activity**

- [ ] Check If User ‘A’ Can View the Account Statement for User ‘B’
- [ ] Check If User ‘A’ Can View the Transaction Report for User ‘B’
- [ ] Check If User ‘A’ Can View the Summary Report for User ‘B’
- [ ] Check If User ‘A’ Can Register for Monthly/Weekly Account Statement via Email Behalf of User ‘B’
- [ ] Check If User ‘A’ Can Update the Existing Email ID of User ‘B’ in Order to Retrieve Monthly/Weekly Account Summary

- [ ] Deposit/Loan/Linked/External Account Checking
  - [ ] Check If User ‘A’ Can View the Deposit Account Summary of User ‘B’
  - [ ] Check for Account Balance Tampering for Deposit Accounts
- [ ] Tax Deduction Inquiry Testing
  - [ ] Check If User ‘A’ with It’s Customer ID ‘a’ Can See the Tax Deduction Details of User ‘B’ by Tampering His/Her Customer ID ‘b’
  - [ ] Check Parameter Tampering for Increasing and Decreasing Interest Rate, Interest Amount, and Tax Refund
  - [ ] Check If User ‘A’ Can Download the TDS Details of User ‘B’
- [ ] Check If User ‘A’ Can Request for the Cheque Book Behalf of User ‘B’
- [ ] Fixed Deposit Account Testing
  - [ ] Check If Is It Possible for User ‘A’ to Open FD Account Behalf of User ‘B’
  - [ ] Check If Can User Open FD Account with the More Amount Than the Current Account Balance
- [ ] Stopping Payment on Basis of Cheque/Date Range
  - [ ] Can User ‘A’ Stop the Payment of User ‘B’ via Cheque Number
  - [ ] Can User ‘A’ Stop the Payment on Basis of Date Range for User ‘B’
- [ ] Status Enquiry Testing
  - [ ] Can User ‘A’ View the Status Enquiry of User ‘B’
  - [ ] Can User ‘A’ Modify the Status Enquiry of User ‘B’
  - [ ] Can User ‘A’ Post and Enquiry Behalf of User ‘B’ from His Own Account
- [ ] Fund Transfer Testing
  - [ ] Is It Possible to Transfer Funds to User ‘C’ Instead of User ‘B’ from the User ‘A’ Which Was Intended to Transfer from User ‘A’ to User ‘B’
  - [ ] Can Fund Transfer Amount Be Manipulated?
  - [ ] Can User ‘A’ Modify the Payee List of User ‘B’ by Parameter Manipulation Using His/Her Own Account
  - [ ] Is It Possible to Add Payee Without Any Proper Validation in User ‘A’ ‘s Own Account or to User ‘B’ ‘s Account
- [ ] Schedule Transfer Testing
  - [ ] Can User ‘A’ View the Schedule Transfer of User ‘B’
  - [ ] Can User ‘A’ Change the Details of Schedule Transfer for User ‘B’
- [ ] Testing of Fund Transfer via NEFT
  - [ ] Amount Manipulation via NEFT Transfer
  - [ ] Check If User ‘A’ Can View the NEFT Transfer Details of User ‘B’
- [ ] Testing for Bill Payment
  - [ ] Check If User Can Register Payee Without Any Checker Approval
  - [ ] Check If User ‘A’ Can View the Pending Payments of User ‘B’
  - [ ] Check If User ‘A’ Can View the Payment Made Details of User ‘B’