

CSRF:

- Check if the token is present on any form it should be
 - ONLY Create, Update and Delete forms should have CSRF tokens
- Server checks if the token length is correct
- Server checks if parameter is there
- Server accepts empty parameter
- Server accepts responds without CSRF token
- Token is not session bound

JWT:

- None-signing algorithm is allowed
- Secret is leaked somewhere
- Server never checks secret
- Secret is easily guessable or brute-forceable

Open redirect bypass:

- evil.com/expected.com
- Javascript openRedirects
- Hidden link open redirects
- Using // to bypass
- https:evil.com (browser might correct this, filter might not catch it)
- ^ to bypass
- %00 to bypass (null byte)
- @ to bypass
- Parameter pollution (adding the same parameter twice)

BAC

- Test higher Priv functions should not be able to be executed by lower Priv user
 - Test ALL user levels

- Test with authorise
- JS Functions via developer console
- Copy and paste of URL

IDOR

- Test between ALL tenants (companies hosted on one server/database. Can also be divisions of companies)
 - Test with authorise
 - JS Functions via developer console
 - Copy and paste of URL

Captcha bypasses

- Try change request method
- Remove the captcha param from the request
- leave param empty
- Fill in random value

LFI

- Using // to bypass
- ^ to bypass
- \\
- %00 to bypass (null byte)
- @ to bypass
- URL encoding
- double encodings

RFI:

- Using // to bypass
- ^ to bypass
- \\
- %00 to bypass (null byte)
- @ to bypass

- URL encoding
- double encodings

SQLi:

- “ to trigger
- SQLmap

XXE:

- SVG files (images), DOCX/XLSX, SOAP, anything XML that renders
- Blind SSRF, file exfiltration, command exec

Template injections (CSTI/SST)

- \${7*7}
- If resolves, what templating engine
- Try exploit by looking at manuals
 - URL encode special chars ({}*)
 - HTML entities
 - Double encodings

XSS:

- “^> into every input field, the moment you register and start using the application
- Enter a random value into every parameter and look for reflection
- See what context reflection is in
- Craft attack vector based on context
 - JS
 - HTML
 - HTML tag attribute
 - ...

- Url encode
- HTML entities
- Capital letters
- BASE64 encode payload
- CSP might be active
 - Try bypasses
 - See what is active and where script can be gotten from
 - Encode them in base64
 - Masquerade script as data

SSRF

- SSRF against server itself
- SSRF against other servers on the network

Command injection

- Test every single parameter
- Make a list of commands + command separators for target OS

Admin panel bypass

- Try referr header
- Easy username/pass
- Directory brute forcing for unprotected pages