

A decorative graphic on the left side of the slide consisting of two overlapping parallelograms. The front one is blue and the back one is a light green color. They are positioned diagonally, with the blue one in front of the green one.

Intro Web App Security

Attacks and Defenses



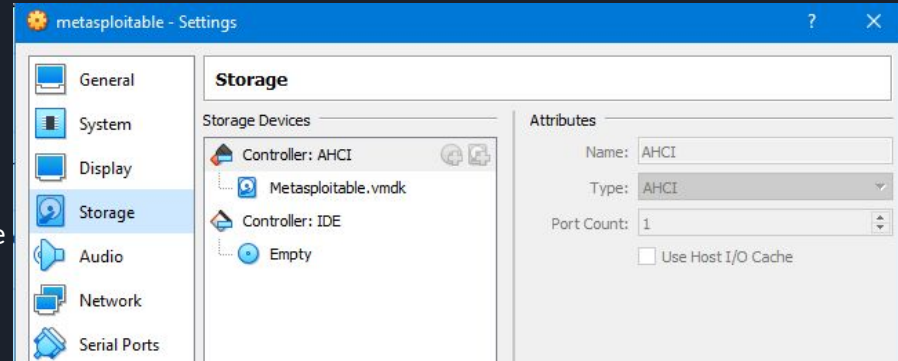
Damn Vulnerable Web App

<https://dvwa.co.uk/>

“Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goals are to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and aid teachers/students to teach/learn web application security in a classroom environment.”

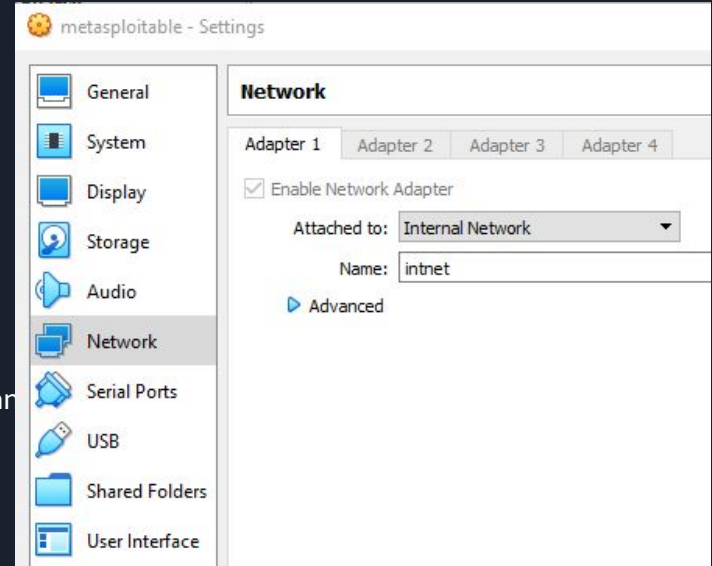
Setup: Get Metasploitable 2 - VM Setup

- Download from:
<https://information.rapid7.com/metasploitable-download.html>
- Import into VirtualBox
 - Unzip file
 - Create new virtual machine with type Linux-Ubuntu(64bit)
 - 1024MB RAM
 - Do not create a hard disk in the wizard
 - Open newly create VM settings
 - Open the "Storage" tab
 - Under "Controller:IDE" option, click on the "Empty" disk placeholder and add the "metasploitable.vmdk" disk there



Setup: Get Metasploitable 2 - VM Network

- Download from:
<https://information.rapid7.com/metasploitable-download.html>
- Import into VirtualBox
 - Unzip file
 - Create new virtual machine with type Linux-Ubuntu(64bit)
 - 1024MB RAM
 - Do not create a hard disk in the wizard
 - Open newly create VM settings
 - Open the "Storage" tab
 - Under "Controller:IDE" option, click on the "Empty" disk placeholder and add the "metasploitable.vmdk" disk there
 - Under the "Network" settings of the VM, change the network to "Internal Network"





Setup: Get Metasploitable 2 - Set Static IP

- Run metasploitable VM
- Login with “msfadmin:msfadmin”
- Run these commands to set static IP
 - `sudo ifconfig eth0 10.0.0.3 netmask 255.255.255.0 up`
 - `sudo route add default gw 10.0.0.1`

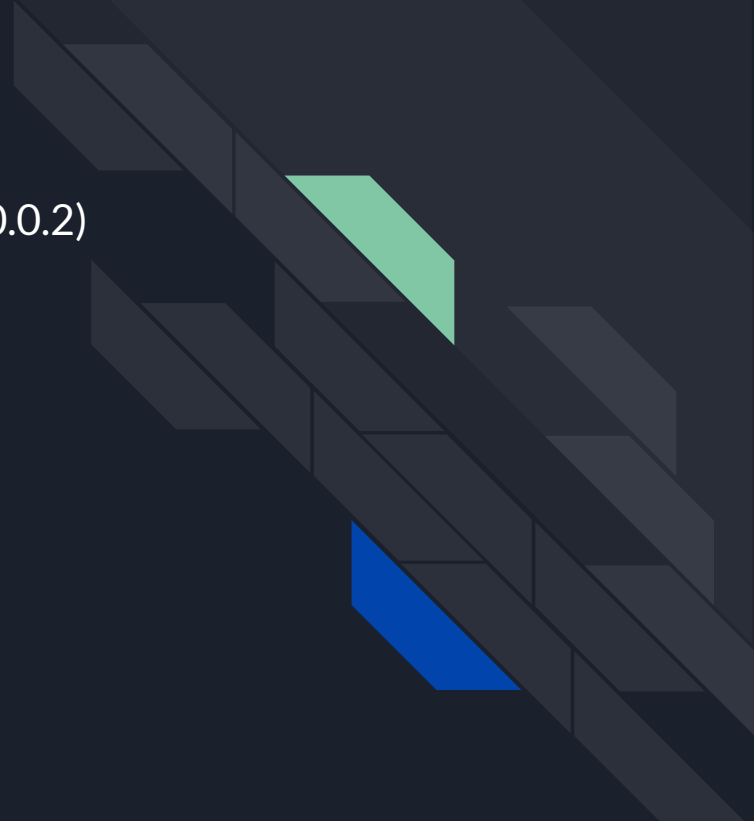
Download and Setup Kali



<https://github.com/DATDA/main/wiki/Getting-Started#linux>

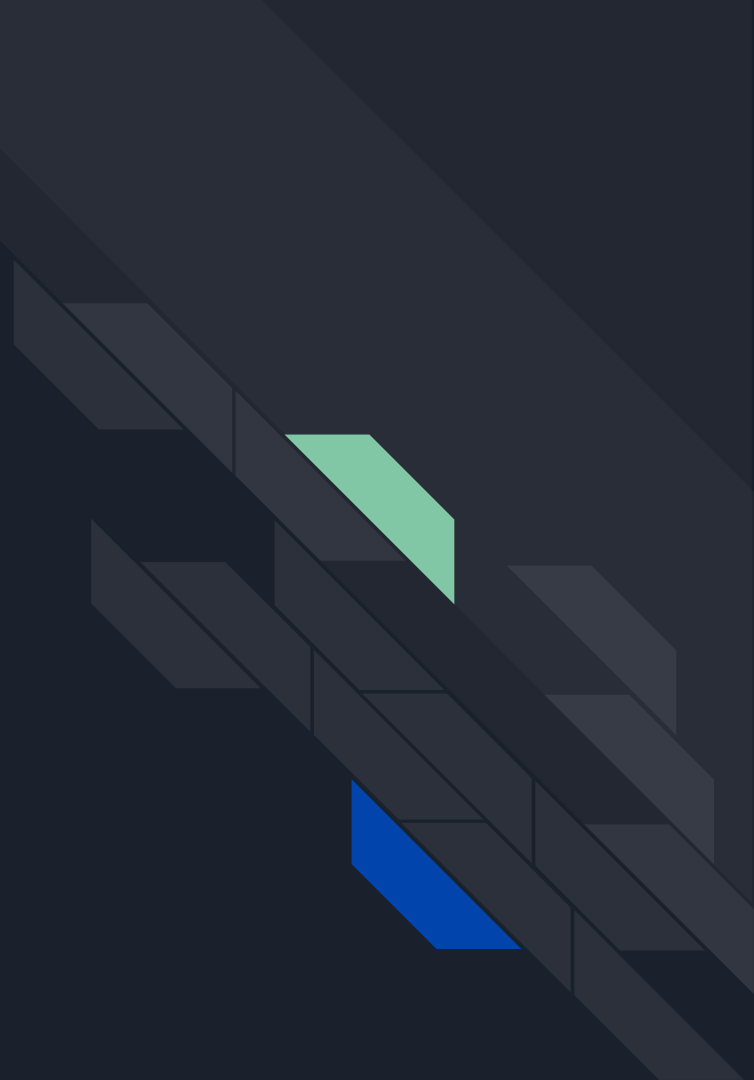
Download and Setup Kali

- Set VM Network adapter to “Internal Network”
- Launch Kali VM
- Set static IP through settings GUI (we'll use 10.0.0.2)



Now with both
VMs running
lets get
started

cracks knuckles



Lets figure out what we're dealing with

NMAP Scan from Kali

- `nmap -sT -A -PO 10.0.0.3`

```
stupid@hush:~$ nmap -sT -A -PO 10.0.0.3
Starting Nmap 7.80 ( https://nmap.org ) at 2021-02-11 16:41 MST
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 10.0.0.3
Host is up (0.0020s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE        VERSION
21/tcp    open  ftp            vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_ftp-syst:
|   STAT:
|   FTP server status:
|     Connected to 10.0.0.2
|     Logged in as ftp
|     TYPE: ASCII
|     No session bandwidth limit
|     Session timeout in seconds is 300
|     Control connection is plain text
|     Data connections will be plain text
|     vsFTPd 2.3.4 - secure, fast, stable
|_End of status
22/tcp    open  ssh            OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
|_ssh-hostkey:
|   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|   2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
|_
23/tcp    open  telnet?
25/tcp    open  smtp?
|_smtp-commands: Couldn't establish connection on port 25
53/tcp    open  domain         ISC BIND 9.4.2
|_dns-nsid:
|   bind.version: 9.4.2
80/tcp    open  http            Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
|_http-title: Metasploitable2 - Linux
111/tcp   open  rpcbind         2 (RPC #100000)
139/tcp   open  netbios-ssn     Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn     Samba smbd 3.0.20-Debian (workgroup: WORKGROUP)
512/tcp   open  exec?
513/tcp   open  login?
514/tcp   open  shell?
1099/tcp  open  java-rmi        GNU Classpath grmiregistry
1524/tcp  open  bindshell       Metasploitable root shell
2049/tcp  open  nfs             2-4 (RPC #100003)
2121/tcp  open  ccproxy-ftp?
3306/tcp  open  mysql?
|_mysql-info: ERROR: Script execution failed (use -d to debug)
5432/tcp  open  postgresql      PostgreSQL DB 8.3.0 - 8.3.7
|_ssl-date: 2021-02-11T23:46:07+00:00; 0s from scanner time.
5900/tcp  open  vnc             VNC (protocol 3.3)
|_vnc-info:
|   Protocol version: 3.3
|   Security types:
|     VNC Authentication (2)
|_
6000/tcp  open  X11             (access denied)
6667/tcp  open  irc             UnrealIRCd
8009/tcp  open  ajp13           Apache Jserv (Protocol v1.3)
|_ajp-methods: Failed to get a valid response for the OPTION request
8180/tcp  open  unknown
```

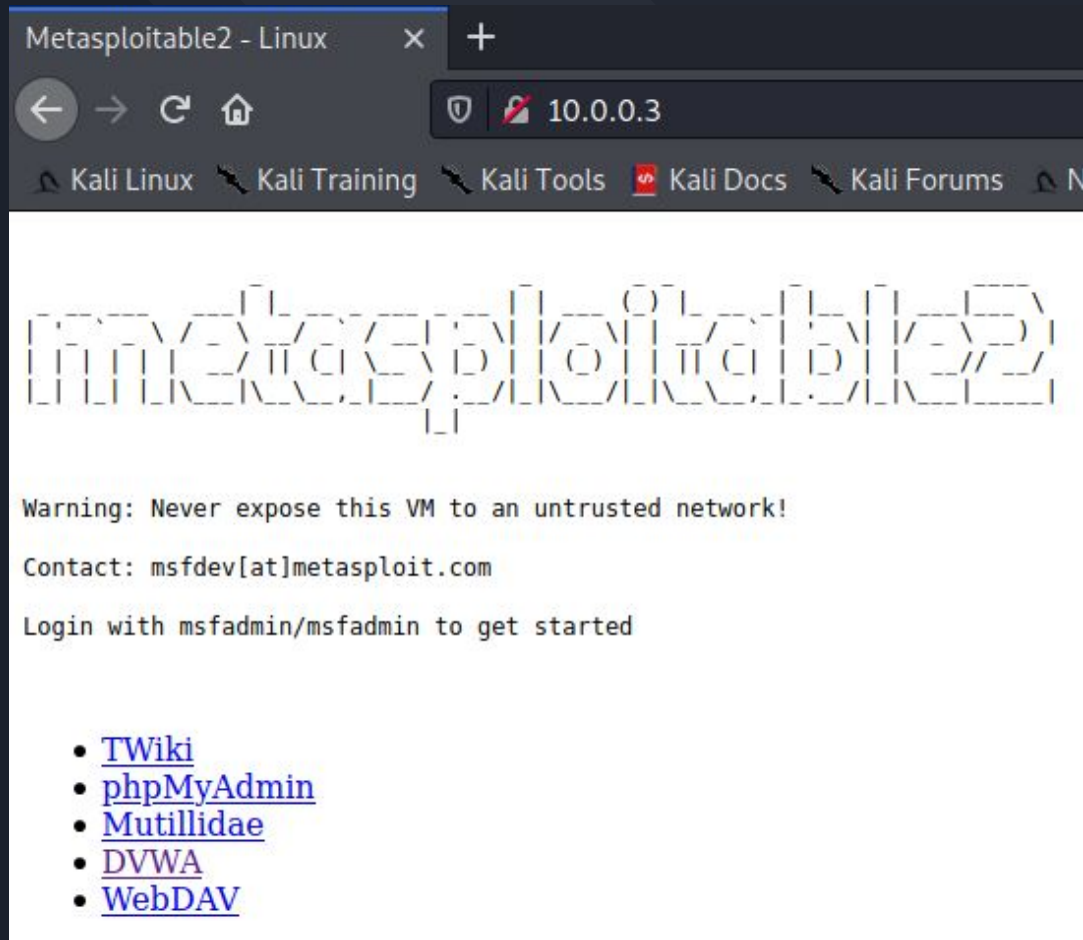
Lets figure out what we're dealing with

NMAP Scan from Kali


- `nmap -sT -A -P0 10.0.0.3`

```
stupid@hush:~$ nmap -sT -A -P0 10.0.0.3
Starting Nmap 7.80 ( https://nmap.org ) at 2021-02-11 16:41 MST
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 10.0.0.3
Host is up (0.0020s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE        VERSION
21/tcp    open  ftp            vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_ftp-syst:
|   STAT:
|   FTP server status:
|     Connected to 10.0.0.2
|     Logged in as ftp
|     TYPE: ASCII
|     No session bandwidth limit
|     Session timeout in seconds is 300
|     Control connection is plain text
|     Data connections will be plain text
|     vsFTPd 2.3.4 - secure, fast, stable
|_End of status
22/tcp    open  ssh            OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
ssh-hostkey:
|_ 1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_ 2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
23/tcp    open  telnet?
25/tcp    open  smtp?
|_smtp-command: Couldn't establish connection on port 25
53/tcp    open  domain         ISC BIND 9.4.2
|_dns-nsid:
|_  bind.version: 9.4.2
80/tcp    open  http            Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
|_http-title: Metasploitable2 - Linux
111/tcp   open  rpcbind         2 (RPC #100000)
139/tcp   open  netbios-ssn     Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn     Samba smbd 3.0.20-Debian (workgroup: WORKGROUP)
512/tcp   open  exec?
513/tcp   open  login?
514/tcp   open  shell?
1099/tcp  open  java-rmi        GNU Classpath grmiregistry
1524/tcp  open  bindshell       Metasploitable root shell
2049/tcp  open  nfs             2-4 (RPC #100003)
2121/tcp  open  ccproxy-ftp?
3306/tcp  open  mysql?
|_mysql-info: ERROR: Script execution failed (use -d to debug)
5432/tcp  open  postgresql      PostgreSQL DB 8.3.0 - 8.3.7
|_ssl-date: 2021-02-11T23:46:07+00:00; 0s from scanner time.
5900/tcp  open  vnc             VNC (protocol 3.3)
|_vnc-info:
|   Protocol version: 3.3
|   Security types:
|     VNC Authentication (2)
6000/tcp  open  X11             (access denied)
6667/tcp  open  irc             UnrealIRCd
8009/tcp  open  ajp13           Apache Jserv (Protocol v1.3)
|_ajp-methods: Failed to get a valid response for the OPTION request
8180/tcp  open  unknown
```

Makes sense, so lets
visit that website



Login to DVWA with
“admin:password”



The DVWA logo features the text "DVWA" in a bold, dark grey sans-serif font. To the right of the text is a stylized circular emblem composed of two curved, overlapping lines, one in a light green color and the other in a dark grey color, creating a sense of motion or a shield.

Username

Password

Login

These are what we will be attacking, today just the SQL Injection section



Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

DVWA Security

PHP Info

About

Logout

Username: admin
Security Level: low
PHPIDS: disabled

Welcome to Damn Vulnerable Web App!

Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goals are to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and aid teachers/students to teach/learn web application security in a class room environment.

WARNING!

Damn Vulnerable Web App is damn vulnerable! Do not upload it to your hosting provider's public html folder or any internet facing web server as it will be compromised. We recommend downloading and installing [XAMPP](#) onto a local machine inside your LAN which is used solely for testing.

Disclaimer

We do not take responsibility for the way in which any one uses this application. We have made the purposes of the application clear and it should not be used maliciously. We have given warnings and taken measures to prevent users from installing DVWA on to live web servers. If your web server is compromised via an installation of DVWA it is not our responsibility it is the responsibility of the person/s who uploaded and installed it.

General Instructions

The help button allows you to view hits/tips for each vulnerability and for each security level on their respective page.

You have logged in as 'admin'

We'll start out easy, set the script security to low

Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

DVWA Security

PHP Info

About

DVWA Security

Script Security

Security Level is currently **low**.

You can set the security level to low, medium or high.

The security level changes the vulnerability level of DVWA.

low ▼

Submit

PHPIDS

PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

You can enable PHPIDS across this site for the duration of your session.

PHPIDS is currently **disabled**. [\[enable PHPIDS\]](#)

[\[Simulate attack\]](#) - [\[View IDS log\]](#)

Just a simple web form
to return user
information

Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

Vulnerability: SQL Injection

User ID:

Submit

ID: 1
First name: admin
Surname: admin

More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>

http://en.wikipedia.org/wiki/SQL_injection

<http://www.unixwiz.net/techtips/sql-injection.html>

When we submit this query, we get a little bit too much information

[Home](#)[Instructions](#)[Setup](#)[Brute Force](#)[Command Execution](#)[CSRF](#)[File Inclusion](#)[SQL Injection](#)[SQL Injection \(Blind\)](#)[Upload](#)[XSS reflected](#)[XSS stored](#)[DVWA Security](#)[PHP Info](#)[About](#)

Vulnerability: SQL Injection

User ID:

ID: %' or '0'='0
First name: admin
Surname: admin

ID: %' or '0'='0
First name: Gordon
Surname: Brown

ID: %' or '0'='0
First name: Hack
Surname: Me

ID: %' or '0'='0
First name: Pablo
Surname: Picasso

ID: %' or '0'='0
First name: Bob
Surname: Smith

Lets look at the source code to see why this happens

No input sanitization

Directly executes what the user inputs

```
<?php
if(isset($_GET['Submit'])) {
    // Retrieve data

    $id = $_GET['id'];

    $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";
    $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre> ');

    $num = mysql_numrows($result);

    $i = 0;

    while ($i < $num) {

        $first = mysql_result($result,$i,"first_name");
        $last = mysql_result($result,$i,"last_name");

        echo '<pre>';
        echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
        echo '</pre>';

        $i++;
    }
}
?>
```



Now change the script security level to see if that same attack will work

Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

DVWA Security

PHP Info

DVWA Security

Script Security

Security Level is currently **high**.

You can set the security level to low, medium or high.

The security level changes the vulnerability level of DVWA.

high ▼

Submit

PHPIDS

PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security la

You can enable PHPIDS across this site for the duration of your

PHPIDS is currently **disabled**. [[enable PHPIDS](#)]

[[Simulate attack](#)] - [[View IDS log](#)]

Nothing happens even though we can see our query is being sent from the info in the URL

10.0.0.3/dvwa/vulnerabilities/sqli/?id=%25'+or+'0'%3D'0&Submit=Submit

Kali Tools Kali Docs Kali Forums NetHunter Offensive Security

DVWA

Vulnerability: SQL Injection

User ID:

More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

Here is the source code that runs when the script security level is set to high

Notice how they are getting the ID parameter

High SQL Injection Source

```
<?php

if (isset($_GET['Submit'])) {

    // Retrieve data

    $id = $_GET['id'];
    $id = stripslashes($id);
    $id = mysql_real_escape_string($id);

    if (is_numeric($id)){

        $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";
        $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre> ');

        $num = mysql_numrows($result);

        $i=0;

        while ($i < $num) {

            $first = mysql_result($result,$i,"first_name");
            $last = mysql_result($result,$i,"last_name");

            echo '<pre>';
            echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
            echo '</pre>';

            $i++;

        }

    }

}

?>
```

Low SQL Injection Source

```
<?php

if(isset($_GET['Submit'])) {

    // Retrieve data

    $id = $_GET['id'];

    $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";
    $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre> ');

    $num = mysql_numrows($result);

    $i = 0;

    while ($i < $num) {

        $first = mysql_result($result,$i,"first_name");
        $last = mysql_result($result,$i,"last_name");

        echo '<pre>';
        echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
        echo '</pre>';

        $i++;
    }
}
?>
```

High SQL Injection Source

```
<?php

if (isset($_GET['Submit'])) {

    // Retrieve data

    $id = $_GET['id'];
    $id = stripslashes($id);
    $id = mysql_real_escape_string($id);

    if (is_numeric($id)){

        $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";
        $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre> ');

        $num = mysql_numrows($result);

        $i=0;

        while ($i < $num) {

            $first = mysql_result($result,$i,"first_name");
            $last = mysql_result($result,$i,"last_name");

            echo '<pre>';
            echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
            echo '</pre>';

            $i++;
        }
    }
}
?>
```



Code Walkthrough

`$_GET['id']` -- When a GET request is performed, retrieve the 'id' parameter that is sent in the URL

`stripslashes($id);` -- Strip any backslashes from the \$id string variable ("Who\'s on first" ⇒ "Who's on first")

`mysql_real_escape_string($id);` -- Escapes any special characters in the \$id string variable