

## Communicator. Sniffing on web applications

When we have the Internet access using free hotspots in public areas, it should realize that unencrypted traffic generated by our devices (http, FTP) may be eavesdropped and recorded, and the privacy of transmitted data may be violated. The aim of this exercise is to build a simple text messenger, the operation of which should be monitored using selected tools for eavesdropping on the network, including Wireshark ([www.wireshark.org](http://www.wireshark.org)).

Tools for eavesdropping on data transmission in the network are completely legal as long as they are used for educational or diagnostic purposes, e.g., during an audit of the operation of a network that is officially administered, but with respect for data privacy - if others use this network at the same time users e.g., company employees. In some countries, the use of the same tools becomes illegal and can be classified as a misdemeanor or even a criminal offense if they are used to violate someone else's privacy, gain unauthorized access to company data, gain access to a foreign network, etc. Interpreting what is a what is not legal depends, among others the type of network the sniffer is accessing. A different interpretation concerns open WiFi networks, and a completely different one - wired networks, to which joining may in itself in some cases be a crime, all the more so then, when the sniffer starts logging traffic on such a network.

During the implementation of this exercise, the use of eavesdropping programs is completely legal, because the communicator created as part of the exercise is only a guinea pig, carried out solely for educational purposes. Eavesdropping is used in this exercise, among others making programmers aware of the need to ensure the privacy of data transmitted over the network, e.g., by implementing methods that ensure their encryption.

**Note:** the task consists of several phases, so that the teacher can verify it - create it in the catalog intended for implementation task index.php file, which will contain links to scripts carrying out individual sub-points of the task. Give the links a title to help the presenter know the specific parts of the assignment.

### Practical implementation of the task

1. Create a simple chat application, based on your own concept or according to the example included in this manual.
2. The example below uses the mechanism for storing conversations in a text file hosted on the hosting server. Create a **conversation.txt** file in the same directory and give it write access to all users. If you are hosting on a unix / linux server - set "w" (write) permission to this file for users belonging to the same group as the owner of the file and other users.

Assigning the current time to the \$time variable

Assigning data from a HTML form to the variables \$user and \$post

If the HTML form has been sent - create a table with variables

HTML form

\$\_POST['user']

\$\_POST['post']

Display the HTML table saved in the file conversation.txt

#### index.php

```
<?php
$time = date('H:i:s', time());
$user = $_POST['user'];
$post = $_POST['post'];
if (isset($_POST['post']))
{
    $text = '<table border="1" width="90%">
    <tr><td>' . $post . '</td><td width="80%">' . $user . '</td><td width="60" bgcolor="yellow">' . $time . '</td></tr></table><br>';
    $file = fopen ("conversation.txt", "a+");
    fwrite ($file, $text);
}
?>

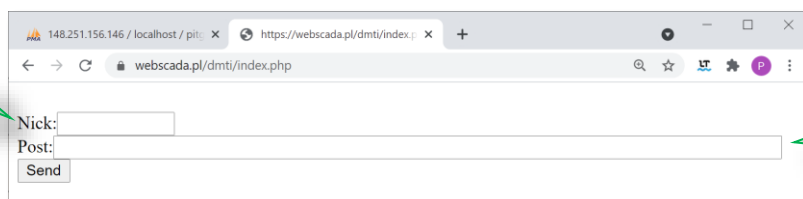
<form method="POST"><br>
    Nick:<input type="text" name="user" maxlength="10" size="10"><br>
    Post:<input type="text" name="post" maxlength="90" size="90"><br>
    <input type="submit" value="Send"/>
</form>

Posts: <br>
<? include ("conversation.txt");?> <br>
```

PHP code

HTML form

3. Test the script by entering the User Nickname and Post, then press the Send key. Note that subsequent posts are displayed lower and lower on the screen. If the content of the form is processed by the same file as the form - it is not necessary to specify the place of data transfer from this form.



HTML form

This is how your HTML form should look like before publishing the first post

This is how your HTML form should look like after placing the first post (Pit / Test text 1): The HTML form is redisplayed and there is also a table with previous posts stored in the file conversation.txt.

4. Run the webmaster tools installed in your web browser and look at the options for fixing errors in the scripts. If you do not have any extensions in your browser that allow you to get help while creating pages - you can install them quickly or, for example, use a browser that has such options built in permanently. Display your portal in it and select "View Source" by clicking on the browser screen. Review the functionality of the individual tabs, check the error messages displayed on the Console tab.
5. Your hosting server may provide WWW service with HTTP or HTTPS protocol support.
  - a. If your hosting server uses the HTTP protocol - you will be able to sniff and read texts sent from your web browser to your server.
  - b. If your hosting server uses the HTTPS protocol - it is not easy to correctly read the transmitted data between your web browser and your server, as they are encrypted.
6. Using the selected software, listen to the data sent and received by the created communicator, between your computer and hosting.
  - a. Copy the data captured from your messenger to the hosting by the eavesdropping software and put them in the report on the implementation of the task.
  - b. To compare the operation of your messenger and other applications, enter text data into the Google search engine in your Internet browser. Verify whether in this case you can easily eavesdrop on the transmitted text.
  - c. Note that the use of some Internet services and applications results in sending plain unencrypted text, as a result, it is possible to eavesdrop and save this text in various network applications, in order to use them further, e.g. to impersonate another user on the network or to gain access to private data.
7. In the next step, the above script should be divided into two parts and placed in two separate files, e.g., index.php and add.php. Extend the information in the first line of the index.php file with the "action" declaration, informing where to send the data from the form.

#### index.php

```
<form method="POST" action="add.php"><br>
  Nick:<input type="text" name="user" maxlength="10" size="10"><br>
  Post:<input type="text" name="post" maxlength="90" size="90"><br>
  <input type="submit" value="Send"/>
</form>

Posts: <br>
<?php include ("conversation.txt"); ?> <br>
```

Information on where to submit data from the HTML form: action = "add.php"

#### add.php

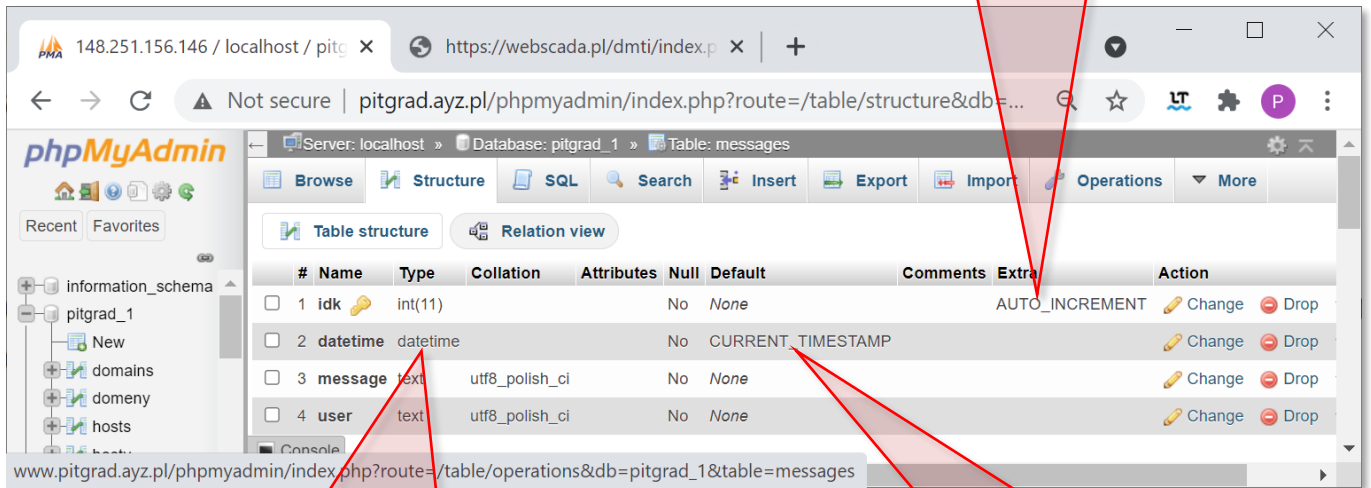
```
<?php
$time = date('H:i:s', time());
$user = $_POST['user'];
$post = $_POST['post'];
if (isset($_POST['post']))
{
    $text = '<table border="1" width="90%">
    <tr><td>'. $post.'</td><td width="80%">'. $user.'</td><td width="60" bgcolor="yellow">'. $time.'</td></tr></table><br>';
    $file = fopen ("conversation.txt", "a+");
    fwrite ($file, $text);
}
header ('Location: index.php'); // to reload index.php after executing add.php
?>
```

This code allows to go back to the index.php file, after doing add.php

8. After the original script was split, you should take into account the need to load the index.php script, each time you run the script add.php. The header () command is used to do that.
9. Using the cookie technology, modify your messenger script in such a way that the user's nickname is entered only once in a given web browser, so that you can exchange messages more comfortably, e.g. between two different browsers, i.e. focus only on typing content message, and not additionally enter the user's nickname each time. To test this functionality, use two different browsers, eg Google Chrome and Firefox.
10. Note that sending text from one browser does not automatically refresh the window of the other browser, which is also running the same application. To find out if someone has sent us new information - refresh the browser window yourself 😞.

11. Using phpMyAdmin, create a messages table in the new database on your hosting with the following format:

**messages** (idk, datetime, message, user).



12. Change the contents of your **index.php** and **add.php** files to send data from the HTML form to the MySQL database (PHASE 1):

#### index.php

```
<form method="POST" action="add.php"><br>
Nick:<input type="text" name="user" maxlength="10" size="10"><br>
Post:<input type="text" name="post" maxlength="90" size="90"><br>
<input type="submit" value="Send"/>
</form>
```

This is an HTML form that sends data to the add.php file, no data is displayed

#### add.php

```
<?php
$time = date('H:i:s', time());
$user = $_POST['user'];
$post = $_POST['post'];
if (isset($_POST['post']))
{
    $dbhost="localhost"; $dbuser="Your_dbuser"; $dbpassword="Your_dbpassword"; $dbname="Your_dbname";
    $connection = mysqli_connect($dbhost, $dbuser, $dbpassword, $dbname);
    if (!$connection)
    {
        echo " MySQL Connection error." . PHP_EOL;
        echo "Errno: " . mysqli_connect_errno() . PHP_EOL;
        echo "Error: " . mysqli_connect_error() . PHP_EOL;
        exit;
    }
    $result = mysqli_query($connection, "INSERT INTO messages (message, user) VALUES ('$post', '$user');") or die ("DB error: $dbname");
    mysqli_close($connection);
}
header ('Location: index.php');
?>
```

Put your own BD parameters here

Bug report in case of problems with BD

Inserting a new post to the database

13. Expand the index.php file to view the posts stored in the database. Do not change the add.php file (PHASE 2):

#### index.php

```
<form method="POST" action="add.php"><br>
    Nick:<input type="text" name="user" maxlength="10" size="10"><br>
    Post:<input type="text" name="post" maxlength="90" size="90"><br>
    <input type="submit" value="Send"/>
</form>

<?php
$dbhost="localhost"; $dbuser="Your_dbuser"; $dbpassword="Your_dbpassword"; $dbname="Your_dbname";
$conn = mysqli_connect($dbhost, $dbuser, $dbpassword, $dbname);
if (!$conn)
{
    echo " MySQL Connection error." . PHP_EOL;
    echo "Errno: " . mysqli_connect_errno() . PHP_EOL;
    echo "Error: " . mysqli_connect_error() . PHP_EOL;
    exit;
}
$result = mysqli_query($conn, "Select * from messages") or die ("DB error: $dbname");

print "<TABLE CELLPADDING=5 BORDER=1>";
print "<TR><TD>id</TD><TD>Date/Time</TD><TD>User</TD><TD>Message</TD></TR>\n";
while ($row = mysqli_fetch_array ($result))
{
    $id      = $row[0];
    $date    = $row[1];
    $message = $row[2];
    $user    = $row[3];
    print "<TR><TD>$id</TD><TD>$date</TD><TD>$user</TD><TD>$message</TD></TR>\n";
}
print "</TABLE>";
mysqli_close($conn);
?>
```

14. Modify the index.php file to display the posts stored in the database in descending order.

This is the SQL query shown in the previous section: **Select \* From messages**

To change the order of records selected from the database, edit this query: **Select \* from messages Order by idk Desc**

15. Expand your chat app with additional functions indicated by the teacher, e.g.

- a. Changing the format and amount of data displayed as a result of the conversation:
  - i. Display only the last 5 messages.
  - ii. Display messages of individual users in different colors, use CSS for this purpose, for formatting styles (do not use HTML 4 tags). We use the HTML 5 standard.
  - iii. Display all messages, but in an auto-scrolling window.
  - iv. Display messages from the newest to the oldest.
- b. Expand the application with options related to the portal users and their data stored in the database table, e.g.
  - i. Create a table containing three or more usernames (nickname). Provide the ability to select from a list the name of the user to whom the message goes during the conversation.
  - ii. Enable logging in to the communication portal of users defined earlier in the portal.
  - iii. Complete the registration of new users using the form and logging in to the portal.
  - iv. To increase the security of data sent in the network, register new users and log in using the methods of text modification entered into the form, including password modification.

16. Using the AJAX technology, modify the application in such a way that the browser window is updated when the next post appears in the database.

The grayed out subpoints contain content that has not been completed during the classes so far, so their implementation is optional.