

Application:

TCP Echo Server with PHP

Running a PHP script when data are received on a TCP port

The pure.box 2 is able to respond when data are received (transmission via TCP or UDP) by opening a PHP script. The received data can be read via PHP and then further processed (e.g. reply to data source, forwarding over the network or saving in the box internal database. The PHP interpreter works here only "under the hood," i.e. no classical output in the form of a web page is generated. This functionality lets you implement simple server applications, using only web technology.

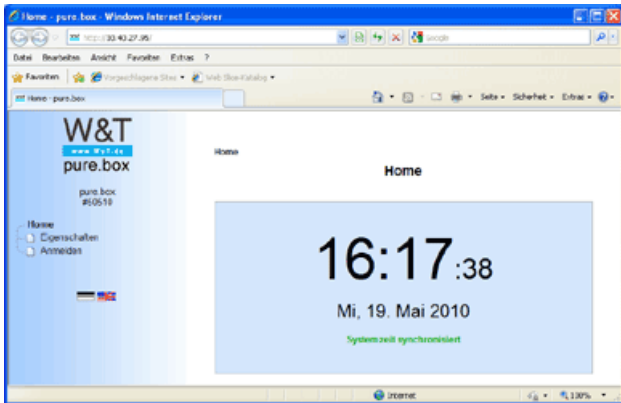


Objective

To test a network connection you want to set up an echo server on the pure.box 2 which listens to TCP port 8000. Data sent to this port will then be returned to the source unchanged.

The PHP file for implementing this example, **echo.php**, can be downloaded [here](#).

Configuration of the pure.box 2

Save the file **echo.php** using FTP or SMB sharing in the folder named **programs** on the pure.box 2. Then carry out the following configuration steps:

<p>Open the Homepage of the pure.box 2 by entering the IP address in the address line of your browser.</p>										
<p>Select Login in the menu tree and log on using your password in the opened dialog box.</p>	<p style="text-align: center;">Anmelden</p> <div style="border: 1px solid #ccc; padding: 10px; margin: 10px auto; width: 80%;"> <p>Passwort <input style="width: 80%;" type="password"/></p> <p style="text-align: center;">Senden</p> </div>									
<p>Navigate to the page Home >> Configuration >> User files. In the file overview click on the folder named programs. The file echo.php is displayed.</p>										
<p>Clicking on the icon Edit file properties (left next to echo.php) opens the Edit dialog.</p>										
<p>Check the option Event controlled. Select TCP and set Port to 8000. Save your changes by clicking on Save... you're done!</p>	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Ereignisgesteuert</td> <td style="width: 20%;"><input checked="" type="checkbox"/></td> <td style="width: 50%;">Ausführung bei Datenempfang auf angegebenem Port</td> </tr> <tr> <td>Protokoll</td> <td> <input checked="" type="radio"/> TCP <input type="radio"/> UDP </td> <td></td> </tr> <tr> <td>Port</td> <td><input style="width: 50px;" type="text" value="8000"/></td> <td>1024 - 65535</td> </tr> </table>	Ereignisgesteuert	<input checked="" type="checkbox"/>	Ausführung bei Datenempfang auf angegebenem Port	Protokoll	<input checked="" type="radio"/> TCP <input type="radio"/> UDP		Port	<input style="width: 50px;" type="text" value="8000"/>	1024 - 65535
Ereignisgesteuert	<input checked="" type="checkbox"/>	Ausführung bei Datenempfang auf angegebenem Port								
Protokoll	<input checked="" type="radio"/> TCP <input type="radio"/> UDP									
Port	<input style="width: 50px;" type="text" value="8000"/>	1024 - 65535								

Now if data are sent to TCP port 8000 on the pure.box 2, they are returned to the sender. The prerequisite for returning the data is reception of a line break (LF / Return key) as end character. In general it must be noted that running of the PHP interpreter is automatically stopped after two minutes. The programmed action should therefore be executable in less time than this.

Source text

In the first two lines the standard input for reading and the standard output for writing data is opened.

```
$in = fopen('php://stdin', 'r');  
$out = fopen('php://stdout', 'w');
```

The incoming data are read character by character in a loop and the string variable **\$input** is appended. The loop is exited when a line break (Line Feed, ASCII table: 10) is detected.

```
$input = "";  
do  
{  
    $input .= fread($in, 1);  
}  
while(substr($input, -1, 1) != chr(10));
```

After receipt, the read string is sent back out to the initiating data source through the standard output.

```
fwrite($out, $input);
```

Then both the standard input and standard output are closed.

```
fclose($in);  
fclose($out);
```

This PHP script has now been fully run and is exited. The next time data are received the script is restarted and runs again as described above.

Summary

Simple server applications can be implemented with the pure.box 2 just by knowing PHP. The server functionality is already built into the box. All that is left is to program the action you wish to occur when the data are received.



[We are available to you in person:](#)

Wiesemann & Theis GmbH
Porschestra. 12
42279 Wuppertal
Phone: +49 202/2680-110 (Mon.-Fri. 8 a.m. to 5 p.m.)
Fax: +49 202/2680-265
info@wut.de

© Wiesemann & Theis GmbH, subject to mistakes and changes: Since we can make mistakes, none of our statements should be applied without verification. Please let us know of any errors or misunderstandings you find so that we can become aware of and eliminate them.

[Data Privacy](#)