BS1000 messenger to web server

Introduction

The BS1000 LAN base station for the Arexx Multilogger system has built-in messenger functionality. With the messenger it is possible source http requests to external web servers by the base station. The http requests are used to transport measurement data to a database via a script based web service like MySQL/Apache or Microsoft's SQLServer/asp.net.
The messenger applies given rules for each incoming measurement. A rule is an action that is executed as soon as its accompanying condition is met.
Rules are composed by the Rule Editor tool, and the resulting rule file is uploaded to the BS1000 were it becomes active immediately after upload. The rule action can be an email message, a HTTP request or turning on the built-in buzzer. Here we focus on the HTTP requests.

HTTP request

The HTTP request contains the following data:

- Request type: POST or GET
- URL: the URL of the web service, a port number can be added to this URL, separated by a colon.
- Request data: a user defined string that contains the actual data. This string is base64 encoded.

When the HTTP request type is POST, the request data string is added to the http request, when the request type is GET, the request data string is appended to the URL separated by the '?' character. On the server side the chosen request method defines how the data is extracted.

The request data string is composed by the BS1000 to contain actual measurement data by the given request data string. Data tags (starting with the '$' character) are replaced by the actual data, like measurement value, sensor id etc. The resulting string is base64 encoded thereafter, and sent to the web server as a HTTP request. The following data tags can be used:

Variable	Description
$v	Measured value
$q	Sensor type 1 = Temperature (°C), 3 = RH% (%), 5=CO2 (ppm)
$i	Identification number of the sensor
$r	rssi-value (signal level value in dBm)
$h	Indication of the hours in the time indication of the measurement
$m	Indication of the minutes in the time indication of the measurement
$s	Indication of the seconds in the time indication of the measurement
$Y	Indication of the year in the time indication of the measurement
$M	Indication of the month in the time indication of the measurement
$D	Indication of the day in the time indication of the measurement
$S	Measurement time in seconds since 1-1-2000 UTC
$w	Missing; Time when the latest measured value has not been transmitted to the http server. Is required for the update of the temp-logger.
$t	*time string*; Time of measurement in the format: hh:mm:ss
$d	*datum string*; Date of the measurement in the short date format

Except for the $w and $S tags, all time indications are expressed in UTC under consideration of the time zone offset indication in the configuration page screen Time server.
The time indications $w and $S are expressed in UTC.

The HTTP request message is base64 encoded. This means that non-alphanumerical characters are converted into "%hh"-strings where „hh“ represents a hexadecimal figure. The lines '&&' and '==' are an exception: these are converted into '&', and '=' respectively. The message for the HTTP request is transmitted via the request header POST, or else added to the URL of the GET request. In this case, the separating sign '?' is added between the URL and the message.

Example of a message:

id==$i&&value==$v

In this example, a web server is programmed to decode the indicated string in two parameters 'id' and 'value'. This method allows to supply up-to-date data from the BS1000 to a web page without a running PC.

Server side

Usually the HTTP request would point to a dedicated web server page with scripting capabilities. For example we assume a page called www.server.com/multilogger.php. On the server side this page would contain some scripting that decodes the data, checks its contents and store the data into a data storage. Other webpages can be used to report measurements from this storage. This document is not intended to be a scripting manual; we refer to the many help available elsewhere. Please look at the PHP help topic 'variables from outside PHP' for example. In order to provide a quick start we show how variables can be evaluated on a PHP page:

```
<?php

// multilogger.php
// needs 6 arguments, separated by '&':
// The message would be: abcdef&&$d&&$t&&$i&&$v

// argument 0 = 'password' (abcdef)
// argument 1 = $d date
// argument 2 = $t  time
// argument 3 = $i  sensor id
// argument 4 = $v  sensor value

$args = explode ("&", $QUERY_STRING  );
$nargs = count($args);

if ($nargs != 5)
{
        die();
}

if ($args[0] != "abcdef")
{
        die();
}

$date     = urldecode($args[1])  ;
$time     = urldecode($args[2])  ;
$device     = urldecode($args[3]);
$temperature = urldecode($args[4]);

$date = str_replace("'", " ", $date);
$time = str_replace("'", " ", $time);
$device  = str_replace("'", " ", $device);
$temperature = str_replace("'", " ", $temperature);
// log it
        $db = mysql_connect('server', 'user', 'password');
        $result = mysql_select_db('database_name', $db);

        $result = mysql_query("delete from temperature where (device ='$device')");
        $result = mysql_query("INSERT INTO temperature (logdate, logtime, device, temperature)
VALUES ('$date', '$time', '$device', '$temperature'  )", $db);

        $result = mysql_close($db);
?>
```

PHP example page

In this case arguments are provided without argument names. This means the BS1000 message should match the expected arguments exactly. The message should be formed as follows:

abcdef&&$d&&$t&&$i&&$v

The argument list is decomposed into an array or strings ($args). The number of arguments should be equal to 5 in this case, and the first argument serves as a password. The arguments are base64 decoded, and a simple character replacement is done to prevent sql injection. This is shown here to remind you precausions should be taken to prevent misuse of the database. Also, but not shown here, some argument checking should be done like checking date and time. Since the BS1000 will only send in actual data, measurements with time stamps that deviate from the actual time can be rejected. The last step is where data is stored into the database by the sql insert statement.