Login to the FRITZ!Box Web Interface

Login Procedure and Session IDs in the FRITZ!Box Web Interface

Login to a FRITZ!Box can take place in three basic ways:

- With user name and password
- With just a passport (factory settings)
- Without a password (not recommended)

Starting with FRITZ!OS 5.50 a session ID is also required in all three cases.

The use of session IDs offers effective protection from what are known as "cross-site request forgeries", in which an attacker sends unauthorized commands to a web application. The following document discusses the use of session IDs and is addressed to developers who program tools for the FRITZ!Box.

Protection from misuse

Besides the aspect of requiring authorization to web sites, using session IDs also offers active protection against possible attacks. If an application without a valid session ID (or with no session ID at all) attempts to access the FRITZ!Box, all active sessions are cleared for security reasons. This means no access to the FRITZ!Box is possible until the user logs in again.

If a program running in the background, like a call monitor, for instance, constantly accesses the FRITZ!Box without a valid session ID, such access will end an active session. In practice, this is apparent in the fact the FRITZ!Box requires you to log back in at regular intervals.

All programs that access the FRITZ!Box web interface must therefore support session IDs, because otherwise they will not only be denied access, but also, as described above, can interfere with normal access to the FRITZ!Box interface via the browser.

The start page

Since FRITZ!OS 5.50, information on the login method is presented and the session ID assigned via the start page

```
http://fritz.box/login_sid.lua
```

Please note: If the page cannot be opened, the firmware in use is too old to support session IDs.

Depending on the parameters transferred, the following actions can be performed:

1) Check whether login is required, and if not, receive a valid session ID immediately Parameter: none

2) Perform login.

Parameters: "username" (name of user or empty),

"response" (a response generated from the password and challenge)

3) Check whether a certain session ID is valid.
Parameters: "sid" (the session ID to be checked)

4) Logout, i.e. invalidate a session ID. Parameters: "logout", "sid" (a valid session ID)

The XML response file always has the same structure:

```
<SessionInfo>
<SID>8b4994376ab804ca</SID>
<Challenge>a0fa42bb</Challenge>
<BlockTime>0</BlockTime>
<Rights>
<Name>NAS</Name>
<Access>2</Access>
<Name>App</Name>
<Access>2</Access>
<Name>HomeAuto</Name>
<Access>2</Access>
<Name>BoxAdmin</Name>
<Access>2</Access>
<Name>Phone</Name>
<Access>2</Access>
</Rights>
</SessionInfo>
```

The number and sequence of values in this area may vary. Only those rights are supplied which are actually assigned to the current session ID.

Meaning of the values:

- <SID>: If this value consists of only zeroes, no rights are granted. A login is required. Otherwise he receives a valid session ID for further access to the FRITZ!Box. The current access rights can be viewed in the <Rights> area.
- <Challenge>: Contains a challenge with can be used to perform a login via the challenge-response method.
- <BlockTime>: The time in seconds during which no further login attempt is allowed.
- <Rights>: The individual rights granted to the current session ID. Possible values are 1 (read only) and 2 (read and write access).

Registration

If "Login without password" is configured in the FRITZ!Box, a valid session ID will be transferred when the start page is opened. In this case you can skip directly to the "Using the Session ID" section.

If login is required, a valid session ID must be generated first as part of the login process. For security reasons the login is performed not with the plain text password, but via a challenge-response procedure.

Determining the Response Value

The response value is determined from the plain text password and a challenge:

`<response> = <challenge>-<md5>`

`<challenge>` is read from the start page login_sid.lua.

<md5> is the MD5 (<challenge>-<plain text password>) in 32 hexadecimals with lower-case letters).

The MD5 hash is generated from the byte sequence of the UTF-16LE coding of this string (without BOM and without closing 0 bytes).

For reasons of compatibility, the coding of the "." character (0x2e 0x00 in UTF-16LE) must be used for every character that has a unicode codepoint > 255. So this affects all characters that cannot be rendered in ISO-8859-1, e.g. the euro symbol.

Example with a German umlaut:

The challenge

```
<challenge> = "1234567z"
```

combined with the password

```
<plain text password> = "äbc"
```

yields the value

```
<response> = "1234567z-9e224a41eeefa284df7bb0f26c2913e2"
```

In this case the login looks like the following:

```
http://fritz.box/login_sid.lua? response=1234567z-
9e224a41eeefa284df7bb0f26c2913e2
```

The response XML will now contain the valid SID.

Using the Session ID

The session ID is a 64-bit number rendered by 16 hexadecimals. It is assigned at login and must be carried along for the duration of the session. A program should use only one session ID for each FRITZ!Box, since only a restricted number of sessions to each FRITZ!Box is permitted.

Once it has been assigned, a session ID is valid for 20 minutes. The validity is extended automatically whenever access to the FRITZ!Box is active.

The session ID 0 (0000000000000000) is always invalid.

The session ID is passed in the parameter "sid".

Access without a Session ID

In principle, all dynamically generated pages can be opened only with a valid session ID. Even reading or writing web variables requires a session ID.

The following content can be opened without a valid session ID:

- homepages (e.g., a login page)
- static contents (e.g. graphics)

Ending a Session

A session can be ended at any time by deleting the session ID, even before the 20-minute timeout kicks in.

A session is ended by opening the login page with the "logout" parameter:

Example Code for Receiving a Session ID with FRITZ!OS 5.50 or Higher

Beispiel-Code .NET

```
using System;
using System.Text;
using System.Net;
using System.Xml.Linq;
using System.Security.Cryptography;
using System.IO;

namespace FritzLogin
{
  class LoginData
  {
    public LoginData(string scheme, string host, String userName, string
password, string path)
    {
      Scheme = scheme;
      Host = host;
      UserName = userName;
      Password = password;
      Path = path;
      Url = string.Format("{0}://{1}/{2}", Scheme, Host, Path);
    }

    public string Scheme { get; set; }
    public string Host { get; set; }
    public string UserName { get; set; }
    public string Password { get; set; }
    public string Path { get; set; }
    public string Url { get; private set; }
  }

  class Program
  {
    static void Main(string[] args)
    {
      var loginData = new LoginData("http", "fritz.box", "username",
"password", "login_sid.lua");
      Test(loginData); Console.ReadKey();
    }

    static public void Test(LoginData loginData)
    {
      Console.WriteLine("Url: " + loginData.Url);
      // determine sessionID
      string sid = GetSessionId(loginData);
      Console.WriteLine("SID: " + sid);
      string page = ReadPage(loginData.Url, sid);
      Console.WriteLine("Page content:" + System.Environment.NewLine +
page.ToString());
    }
```

```
    static public string ReadPage(string url, string sid)
    {
      url = String.Format("{0}?sid={1}", url, sid);
      HttpWebRequest request = WebRequest.Create(url) as HttpWebRequest;
      HttpWebResponse response = request.GetResponse() as HttpWebResponse;
      StreamReader reader = new StreamReader(response.GetResponseStream());
      return reader.ReadToEnd();
    }

    static public string GetSessionId(LoginData loginData)
    {
      XDocument doc = XDocument.Load(loginData.Url);
      // Brute-Force-Protection.
      string szBlocktime = GetValue(doc, "BlockTime");
      int blockTime = Int32.Parse(szBlocktime);
      if (blockTime > 0)
      {
        Console.WriteLine("waiting {0}sec. ...", blockTime);
        System.Threading.Thread.Sleep(blockTime * 1000);
      }
      string sid = GetValue(doc, "SID");
      if (sid == "0000000000000000")
      {
        string challenge = GetValue(doc, "Challenge");
        string url = String.Format("{0}?username={1}&response={2}",
loginData.Url, loginData.UserName, GetResponse(challenge,
loginData.Password));
        doc = XDocument.Load(url);
        sid = GetValue(doc, "SID");
      }
      return sid;
    }

    static public string GetResponse(string challenge, string password)
    {
      return String.Format("{0}-{1}", challenge, GetMD5Hash(challenge + "-"
+ password));
    }

    static public string GetMD5Hash(string input)
    {
      MD5 md5Hasher = MD5.Create();
      // UTF-8 > UTF-16LE
      byte[] data =
md5Hasher.ComputeHash(Encoding.Unicode.GetBytes(input));
      StringBuilder sb = new StringBuilder();
      for (int i = 0; i < data.Length; i++) {
        sb.Append(data[i].ToString("x2"));
      }
      return sb.ToString();
    }

    static public string GetValue(XDocument doc, string name)
    {
      XElement info = doc.FirstNode as XElement;
      return info.Element(name).Value;
    }
  }
}
```

Beispiel-Code PHP

```
<?php

$fritz_host = 'fritz.box';
$fritz_url = 'http://' . $fritz_host . '/login_sid.lua';
$fritz_user = 'username';     //optional
$fritz_pwd = 'password';
$with_debug_output = false;


// Get Challenge-String
$tmp = simplexml_load_string(file_get_contents($fritz_url));
if ($tmp->BlockTime > 0) {
      sleep($tmp->BlockTime);
      $tmp = simplexml_load_string(file_get_contents($fritz_url));
}
if ($with_debug_output) {
      print_r('Got challenge: ' . $tmp->asXML() . PHP_EOL);
}
$challenge = $tmp->Challenge;

// Get SID
$challenge_str = $challenge . '-' . $fritz_pwd;
$md_str = md5(iconv("UTF-8", "UTF-16LE",  $challenge_str));
$response = $challenge . '-' . $md_str;
$tmp = simplexml_load_string(file_get_contents($fritz_url . '?user=' .
$fritz_user . '&response=' . $response));
if ($with_debug_output) {
      print_r(' Got session ID: ' . $tmp->asXML() . PHP_EOL);
}
$sid = $tmp->SID;

// Logout
$tmp = simplexml_load_string(file_get_contents($fritz_url .
'?logout=1&sid=' . $sid));
if ($with_debug_output) {
      print_r(' Logout: ' . $tmp->asXML() . PHP_EOL);
}
?>
```