MQTT in Symcon Einbinden

Stand September 2018, IPS 4.4

Diese Anleitung zeigt wie man auf einem Raspberry PI 3 das MQTT Protokoll über Mosquitto und mqttwarn in Symcon schreibend und per Modul auch Beschreibbar ist. Also in beide Richtungen Ereigniss Basierend kommuniziert so wie ich es schon seit einem halben Jahr Produktiv mit diversen Arduinos und ESP8266 (PubSub Library) einsetze.

Alle Befehle werden als root am PI ausgeführt, wenn jemand nicht weiß wie das geht sollte er es besser lassen oder sich etwas in Linux einarbeiten!

Auch müssen alle Werte in <spitzen> Klammern durch die Eigenen ersetzt werden!

Grund Installation am PI Einrichten

Zuerst muss am PI folgendes mit

```
raspi config
```

eingestellt werden:

- Expand Filesystem
- Boot Options
  - B4 (Boot to CLI)
- Internationalisation Options
  - Change Locale –> de_DE.utf8 und auch als Standard Einstellen
  - Change Timezone –> Europe / Wo auch immer ihr Wohnt
  - Change Keyboard Layout –> Other / German
- Password von user PI Ändern udn auch eines für root setzen
- in der Datei

```
nano /etc/ssh/sshd_config
```

- die Zeile

```
#PermitRootLogin without-password
```

mit dem # Vorangestellt auskommentieren damit ihr euch als root nach einem neustart des Dienstes per SSH anmelden könnt.

```
service ssh restart
```

- Einige Paket deinstallieren

```
aptitude purge wolfram-engine -y
aptitude update && aptitude upgrade -y
```

- Einige Standard Tools installieren

```
aptitude install htop iftop iotop vim -y
```

Pakete Installieren

Einen MQTT Broker z.B. das Paket mosquitto Installieren

```
aptitude install mosquitto mosquitto_pub mosquitto_sub
```

Symcon Vorbereiten

MQTT Client zum Senden in Symcon Installieren

Zum Senden von Daten per MQTT_Publish aus jedem script:

```
https://github.com/mkretzschmar/SymconMQTT
```

Mosquitto an Symcon Anbinden

Eine Kategorie anlegen

```
Datenpunkte/MQTT/devices
```

In diesem Knoten werden dann alle Geräte und Datenpunkte Automatisch erstellt / Aktualisiert, nun gehen wir aber wieder zu

```
Datenpunkte/MQTT
```

und erstellen das Script „mqtt_switch_relais“, damit sind „boolean“ Datenpunkte (einfach „True“ oder „False“ beim ersten schreiben in dem mqtt broker erstellt die Variable in Symcon und verlinkt diese mit dem Script und ist somit sofort Bedienbar.

Die Datei:

```
Datenpunkte/MQTT/mqtt_switch_relais"
```

mit folgendem Inhalt:

```
<?
    include_once(IPS_GetScript(42011 )["ScriptFile"]);

    $d = true; $msg = "";
    $value = ""; $run = true;

    if ($d){
        $msg .= "Sender: ".$_IPS['SENDER']."\n";
        $msg .= print_r($_IPS, true);
    }

    if ($_IPS['SENDER'] == "Execute"){
        //On Manual Execute check child events and set Event Trigger to "On
Change"
        foreach (IPS_GetChildrenIDs($_IPS['SELF']) as $key => $value){
            if (IPS_GetObject($value)['ObjectType'] == 4){
                IPS_SetEventTrigger($value, 1,
IPS_GetEvent($value)['TriggerVariableID']);
            }
        }
    } else {
        if (($_IPS['SENDER'] == "WebFront") or ($_IPS['SENDER'] ==
"Variable")){
            if ($_IPS['SENDER'] == "Variable"){
                //$value =
(GetValueBoolean($_IPS['VARIABLE'])?"true":"false");
                $value = ($_IPS['VALUE']=="1"?"true":"false");
            }
            if (($_IPS['SENDER'] == "WebFront")){
                $value = ($_IPS['VALUE']=="1"?"true":"false");
            }

            if (($value ==
(GetValueBoolean($_IPS['VARIABLE'])?"true":"false")) and ($_IPS['SENDER'] !=
"Variable")){
                if($d){ $msg .= "  Aktuell: >".$value."< alt:
>".(GetValueBoolean($_IPS['VARIABLE'])?"true":"false")."<\n"; }
                $run = false;
            }

            if ($d){
                $msg .= "  run: ".($run?"Ja":"Nein")."\n";
            }

            $topic =
explode(";",IPS_GetObject($_IPS['VARIABLE'])['ObjectInfo'])[1];

            if((strlen($topic) > 3) and ($run)){
                if ($d){
                    $msg .= "  Veröffentliche: ".$value."\n";
                    $msg .= "    Variable:
```

```
".IPS_GetName($_IPS['VARIABLE'])."\n";
                    $msg .= "    Topic: ".$topic."\n";
                    $msg .= "    Value: ".$value."\n";
                }
                MQTT_Publish($idMQTTBroker , $topic ,$value ,1 , true);
            }
        }
    }
   if(strlen($msg) > 0){
        IPS_LogMessage("mqtt_switch_relais", $msg);
    }

?>
```

Das Eigentliche Script welces die Daten per RPC vom mqttwarn Daemon entgegennimmt und ablegt, in dem nicht Vergessen die ID's in den ersten Zeilen durch die Eigenen ersetzen:

```
 <?
$d = true; $msg = "";
//sample value:   devices/terminal/helligkeit;319
// mqtt topic to symcon variable id mapping
$idParent = IPS_GetParent($_IPS['SELF']);
$archiveId = 32289;
$relaisScriptId = 17676; //Relais Script ID

// get value from mqtt variable
if( array_key_exists("payload", $_IPS)){
 $topic = $_IPS['topic'];
 $topics = explode("/", $topic);
 $payload = $_IPS['payload'];

 $payload = str_replace(array("\r", "\n"), '', $payload);

 if ($d){ $msg .= "topic: $topic\npayload: $payload\n"; }

 // Datentyp der payload herausfinden und ins Format für IPS Übersetzen
 $contType = 99;
 if (is_float($payload+0)){
   //Float
 $contType = 2;
 }
 if (($contType == 99) and (is_numeric($payload))){
   //Integer
 $contType = 1;
 }
 if (($contType == 99) and (is_string($payload))){
 if ((strtolower($payload) == "true") or (strtolower($payload) == "false")){
    //Boolean
  $contType = 0;
 } else {
    //String
```

```
  $contType = 3;
 }
 }
 $idf = $idParent;
 //Pfad durchlaufen und Prüfen ob die Kategorien / Variablen vorhanden sind
und bei bedarf Anlegen
 for($i = 0; $i < count($topics); $i++){
 if ($i == (count($topics) -1)){
  $type = 2;
 } else {
  $type = 0;
 }
 $idf = checkIfElementExists($idf, $type, $topics[$i], $contType);
 }

 if($d){
   $msg .= "Variablen Type. ".$contType."\n";
   $msg .= "Set Itend to: "."mqtt_".str_replace("/", "_", $topic)."\n";
 }

 IPS_SetIdent($idf, "mqtt_".str_replace("/", "_", $topic));

 //mqset($idf, $payload);
 SetValue($idf, $payload);
}
if ((strlen($msg) > 0) and ($d)){
   if($_IPS['SENDER'] == "Execute"){
      print $msg;
    }
    IPS_LogMessage("mqtt-read-run", $msg);
}

function isTrueFloat($val){
    $pattern = '/^[-+]?(((\\\\d+)\\\\.?(\\\\d+)?)|\\\\.\\\\d+)([eE]?[+-
]?\\\\d+)?$/';

    return (!is_bool($val) && (is_float($val) || preg_match($pattern,
trim($val))));
}

function mqset($id, $value){
    Global $msg, $d;
    if(GetValue($id) != $value){
        switch(IPS_GetVariable($id)["VariableType"]){
            case 0: // boolean
                $val = filter_var($value, FILTER_VALIDATE_BOOLEAN,
FILTER_NULL_ON_FAILURE);
                if($d){ $msg .= "val: ".$val."\n"; }
                SetValue($id, $val);
                break;
            case 1: // integer
```

```
                $val = intval($value);
                SetValue($id, $val);
                break;
            case 2: //  float
                $val = floatval($value);
                SetValue($id, $val);
                break;
            case 3: // string
                SetValue($id, $value);
                break;
        }
    }
}


function checkIfElementExists($varid, $type, $name, $contType ){
    Global $archiveId, $topic, $relaisScriptId;
    $exists = false;
    if (IPS_HasChildren($varid)){
        // Wenn Childs da sind alle durchlaufen
        foreach (IPS_GetChildrenIDs($varid) as $key){
            if (IPS_GetObject($key)["ObjectType"] == $type){
                // Prüfen ob schon ein Objekt mit genau dem Namen existiert
                if (IPS_GetObject($key)["ObjectName"] == $name){
                    // Dann nicht erstellen und die id merken
                    $exists = true;
                    $id = $key;
                }
            }
        }
    }
    if (!$exists){
       if($type == 2){
            //Wenn es eine Variable sein soll
           $id = IPS_CreateVariable($contType);
        } else {
            //Sonst ist es eine Kategorie
           $id = IPS_CreateCategory();
        }
        IPS_SetName($id, $name);
  IPS_SetParent($id, $varid);
  if($type == 2){ // Variable
   if ($contType == 0){ // Boolean
    if (strpos("   ".strtolower($name), "licht") > 0){
       //Wenn der Variablen Name das wort "licht" enthält das Profil Licht
zuweisen
       IPS_SetVariableCustomProfile($id, "Licht");
       IPS_SetVariableCustomAction($id, $relaisScriptId);
    } elseif(strpos("   ".strtolower($name), "motion") > 0){
       IPS_SetVariableCustomProfile($id, "Motion");
    } elseif ($name == "connected"){
```

```
       IPS_SetVariableCustomProfile($id, "Connected");
    } else {
     //Wenn es eine Boolean Variable ist das Profile ~Switch zuweisen
     IPS_SetVariableCustomProfile($id, "~Switch");
     IPS_SetVariableCustomAction($id, $relaisScriptId);
     checkCreateEvent($id, $relaisScriptId);
    }
   }
   if (strpos("   ".strtolower($name), "temp") > 0){
      IPS_SetVariableCustomProfile($id, "~Temperature");
   }
   if (strpos("   ".strtolower($name), "humidity") > 0){
      IPS_SetVariableCustomProfile($id, "~Humidity");
   }
   if (strpos("   ".strtolower($name), "rssi") > 0){
      IPS_SetVariableCustomProfile($id, "rssi");
   }
            IPS_SetInfo($id, "mqtt;$topic");
            //Archivierung aktivieren
   AC_SetLoggingStatus($archiveId, $id, true);
   IPS_ApplyChanges($archiveId);
        }
    }
    return $id;
}

function checkCreateEvent($varid , $scriptId){
    $create = true;
    if (IPS_HasChildren($varid)){
        // Wenn Childs da sind alle durchlaufen
       foreach (IPS_GetChildrenIDs($varid) as $key){
          // Wenn diese Events sind
          if (IPS_GetObject($key)["ObjectType"] == 4){
             // Prüfen ob schon ein Event mit genau dem script existiert
             if (IPS_GetEvent($key)["TriggerVariableID"] == $script){
                // Sonst erstellen
                $create = false;
             }
          }
       }
    } else {
       // Sonst erstellen
       $create = true;
    }
    if ($create){
  //Event Anlegen
  $eid = IPS_CreateEvent(0);                  //Ausgelöstes Ereignis
        IPS_SetEventTrigger($eid, 1, $varid);        //Bei
Variablenaktualisierung
        IPS_SetParent($eid, $scriptId);              //Ereignis zuordnen
        IPS_SetEventActive($eid, true);              //Ereignis aktivieren
```

```
    }
}
?>
```

MQTT Broker (Mosquitto)

```
aptitude install mosquitto mosquitto-clients python-pip git -y
```

mqttwarn (mqtt daemon der alle Ereignisse zu Symcon weiterleitet)

Installation aus der Wiki von hier Abhängige Pakete Installieren:

```
pip install paho-mqtt jinja2
```

Verzeichniss erstellen

```
cd /opt/
```

Herunterladen mit:

```
git clone https://github.com/jpmens/mqttwarn.git
cd mqttwarn
chmod +x mqttwarn.py
cp mqttwarn.ini.sample mqttwarn.ini
```

Derinhalt der mqttwarn.ini sollte so aussehen:

```
# -*- coding: utf-8 -*-
# mqttwarn example configuration file "mqttwarn.ini"

[defaults]
hostname     = 'localhost'  ; default
port         = 1883
username     = None
password     = None
clientid     = 'mqttwarn'
lwt          = 'clients/mqttwarn'
skipretained = False
cleansession = True

protocol     = 3

; logging
logformat = '%(asctime)-15s %(levelname)-5s [%(module)s] %(message)s'
logfile   = '/var/log/mqttwarn.log'

; one of: CRITICAL, DEBUG, ERROR, INFO, WARN
loglevel  = DEBUG
```

```
#loglevel  = ERROR

; path to file containing self-defined functions for formatmap and datamap
; omit the '.py' extension
functions = 'samplefuncs'

; name the service providers you will be using.
launch    = file, log, http

[config:file]
append_newline = True
targets = {
    'f01'       : ['/tmp/f.01'],
    'log-me'    : ['/tmp/log.me'],
    'mqttwarn'  : ['/tmp/mqttwarn.err'],
    }

[config:log]
targets = {
    'debug'  : [ 'debug' ],
    'info'   : [ 'info' ],
    'warn'   : [ 'warn' ],
    'crit'   : [ 'crit' ],
    'error'  : [ 'error' ]
  }

[config:http]
timeout = 60
targets = {
                #method     #URL               # query params or None
# list auth
  'get1'    : [ "get",  "http://example.org?", { 'q': '{name}', 'isod' :
'{_dtiso}', 'xx': 'yy' }, ('username', 'password') ],
  'post1'    : [ "post", "http://example.net", { 'q': '{name}', 'isod' :
'{_dtiso}', 'xx': 'yy' }, None ],
  'postIPSORIGINAL'    : [ "post", "http://<symcon ip>:3777/api/", None,
('username', 'passwd') ],
  'postIPS'    : [ "post", "http://<symcon ip>:3777/api/", None , None, True
]
  }

; special config for 'failover' events
[failover]
targets = log:error, file:mqttwarn

[devices/#]
targets = log:info, http:postIPS
template = ipsrpc.json
```

der letzte Abschnitt [devices/#] bedeutet das alle daten im MQTT Broker (mosquitto)

unter devices/* an IPS übergeben werden!

dann noch das template

```
cd /opt/mqttwarn/templates
```

in form der Datei

```
vi ipsrpc.json
```

mit folgendem Inhalt erstellen:

```
{% set data = {
                'method': "IPS_RunScriptEx",
                'params': [33107, { 'topic': topic, 'payload': payload } ],
                'jsonrpc': "2.0",
                'id': '0'
        }
%}
{{ data | jsonify }}
```

Supervisor

Damit mqttwarn als Dienst ausgeführt wird verwende ich supervisor wie hier Beschrieben.

```
aptitude install supervisor
```

in dem Ornder

```
cd /etc/supervisor/conf.d
```

die Datei mqttwarn.conf anlegen

```
vi mqttwarn.conf
```

mit folgendem Inhalt

```
[program:mqttwarn]
directory = /opt/mqttwarn
command = /opt/mqttwarn/mqttwarn.py
user = root
environment= MQTTWARNINI="/opt/mqttwarn/mqttwarn.ini"
```

Nun kann mqttwarn gestartet werden mit:

```
service supervisor start
```

und z.B. in einem zweiten Terminal Fenster die Log Datei beobachtet werden:

```
tail -f /var/log/mqttwarn.log
```

was so aussehen sollte:

```
2016-11-30 21:26:19,678 DEBUG [mqttwarn] Attempting connection to MQTT
broker localhost:1883...
2016-11-30 21:26:19,678 DEBUG [mqttwarn] Setting LWT to clients/mqttwarn...
2016-11-30 21:26:19,682 INFO  [mqttwarn] Starting 1 worker threads
2016-11-30 21:26:19,683 DEBUG [mqttwarn] Job queue has 0 items to process
2016-11-30 21:26:19,684 DEBUG [mqttwarn] Connected to MQTT broker,
subscribing to topics...
2016-11-30 21:26:19,685 DEBUG [mqttwarn] Subscribing to devices/# (qos=0)
```