Plugin-Daten	
Autor	christiantf
Logo	
Status	STABLE
Version	2.1.0
Min. LB Version	1.4.3
Release	https://github.com/christianTF/LoxBerry-Plugin-MQTT-Gateway/archive/refs/tags/2.1.0.zip
Beschreibung	Installiert den MQTT-Broker Mosquitto und stellt ein MQTT-Gateway für den Miniserver bereit.
Sprachen	EN
Forum	https://www.loxforum.com/forum/projektforen/loxberry/plugins/294881-mqtt-gateway-2-x

MQTT Gateway

Version History...

Version 0.1.1

- DEVELOPER PREVIEW - Nur einsetzen mit MQTT und Linux-Kenntnissen - Fehler ausschließlich per Issue bei GitHub
- Stellt ein MQTT→Loxone Gateway bereit

Version 0.1.2

- DEVELOPER PREVIEW - Nur einsetzen mit MQTT und Linux-Kenntnissen - Fehler ausschließlich per Issue bei GitHub
- Implementiert ein UDP-Interface für Loxone→MQTT

Version 0.1.3

- DEVELOPER PREVIEW - Nur einsetzen mit MQTT und Linux-Kenntnissen - Fehler ausschließlich per Issue bei GitHub
- Eigene UI-Sektion für MQTT Broker
- Authentifizierung für MQTT Broker
- Lokaler Mosquitto abschaltbar
- Retry für den Mosquitto Service, wenn er beim Systemstart nicht automatisch gestartet wurde
- MQTT Gateway wird direkt nach der Installation gestartet
- Parameter-Update-Routine zum Hinzufügen von neuen Default-Parametern während der Installation

Version 0.1.4

- DEVELOPER PREVIEW - Nur einsetzen mit MQTT und Linux-Kenntnissen - Fehler ausschließlich per Issue bei GitHub
- UI: Expand JSON Option hinzugefügt
- Gateway: JSON wird bei HTTP expandiert (nur erster Level) und beim HTTP-Namen angehängt - EXPERIMENTELL

- Gateway: Alle übermittelten Topics werden im Speicher behalten (max. 24 Stunden)
- UI: Funktion zum Abfragen der gespeicherten Gateway-Topics (wird aber noch nicht dargestellt)

Version 0.1.5

- DEVELOPER PREVIEW - Nur einsetzen mit MQTT und Linux-Kenntnissen - Fehler ausschließlich per Issue bei GitHub
- UI: Größerer Umbau des UI (Settings, Subscriptions, Conversions, Incoming Overview, Logfiles) als Navbar
- UI: Subscriptions jetzt als eigene Seite (leichter anzupassen, man muss nicht immer auf- und abscrollen für Apply)
- UI: Conversions: Angabe von Strings, die zu Werten konvertiert werden sollen (Loxone kann mit Strings nichts anfangen)
- UI: Incoming Overview: Anzeige der an den Miniserver übermittelten Werte
- UI: Logs: Inline-Anzeige der Logfile-Übersicht
- Gateway: Benutzerdefinierte Conversion (eingehende Nachrichten können in Werte konvertiert werden)
- Gateway: Logfile: Parsing der Conversion wird geloggt, inklusive "komischer" Werte und Duplikate
- Gateway: Logfile wird beim Beenden des Gateways sauber abgeschlossen

Version 0.2.0 PRE-RELEASE

- Erster Pre-Release
- Allgemein: Neues Logo (MQTT-Logo statt Mosquitto-Logo)
- Gateway: Besseres Fehlerhandling beim Restart des Gateways
- UI: Filter bei den übermittelten Nachrichten (erleichtert die Suche)
- Donation: Erster Versuch, per Spende einen eigenen Test-Miniserver zusammen zu bekommen

Version 0.3 PRE-RELEASE

- Lokale Mosquitto-Authentifizierung als Standard (Benutzer: loxberry, Passwort: wird generiert)
- Broker-Anmeldedaten werden nur dann geladen, wenn der SecurePIN angegeben wird
- Ermöglicht direkt, die lokalen Mosquitto-Anmeldedaten zu ändern oder zu deaktivieren
- Änderung der Anmeldedaten eines nicht lokalen Brokers ändert nur die Anmeldung des Gateways
- Änderung der Konfiguration übernimmt diese sofort für das MQTT Gateway und/oder Mosquitto

Version 0.3.1 PRE-RELEASE

- Fix: Bei anonymer Verwendung von Mosquitto startet der Mosquitto-Dienst nach Reboot nicht mehr
- Aktivierung von Plugin AutoUpdate (ab dieser Version können Versions-Notifications oder automatisches Update aktiviert werden)

Version 0.4.1 PRE-RELEASE

- Fix: Nach Konfigurationsänderungen und "Apply" wurden diese nicht übernommen
- Neu: Per UDP eingehende Nachrichten werden vollständig weitergegeben (so können beispielsweise auch json Nachrichten gesendet werden)
- Neu: Das Gateway published seinen Connection-Status selbst unter dem Topic <hostname>/mqttgateway/status
- Neu: Das Gateway published im Minutentakt den aktuellen Epoch-Timestamp.

- Neu: Das Gateway subscribed sich selbst automatisch - so kann der Timestamp als Prüfung verwendet werden, ob das Gateway noch funktioniert
- Neu: Unterstützt Last Will and Testament (d.h. Broker setzt den Status auf Disconnected, wenn das Gateway stirbt)
- Neu: UDP-Nachrichten können mit publish oder retain gesendet werden
- Neu: UDP-Nachricht "reconnect" forciert das Neu-Verbinden und Übertragen an den Miniserver
- Neu: Default-UDP-In-Port am Gateway: 11884 (bisher 11883)

Version 0.5.0 PRE-RELEASE

- Fix: Evt. falsche Auswertung der Checkboxen im UI (#5)
- Erweitert: JSON Expansion: Vollständig generische Expandierung ohne Einschränkung auf die Hierarchie
- Erweitert: JSON Expansion für HTTP *und* UDP

Version 0.5.1 PRE-RELEASE

- Fix: JSON expansion not working

Version 0.6.0 PRE-RELEASE

- FIX: Anonymous connection was still possible with password auth (after installation, please save once to fix)
- Enhanced: Incoming Overview: Now is a live visualisation of the topic messages (updated once a second, without browser refresh)
- Enhanced: Incoming Overview: Enable advanced infos to show the original topics, and to delete messages on the broker
- Info for delete event: The data in the overview may not disappear, as the last transmission with "last sent to Miniserver" is still valid

Version 0.7.1 RELEASE

- FIX: Json expansion for UDP only worked when HTTP was enabled
- Added: Overview now shows ready-to-use command recognition for UDP inputs
- Added: Subscription topics are now validated on-the-fly (errors are displayed on the left)

Version 0.8.0 PRE-RELEASE

- FIX: Password was re-created on every update
- New: mosquitto.log on the Logfiles tab
- New: TLS-PSK for encrypted connections, listening on port 8883
- New: In the Incoming Overview, with Advanced Table Info enabled, you can **disable caching for selected data,** and **enable Reset-After-Send**
    - **Disable Cache:** Every incoming value from MQTT will be sent again, also if it has not changed.
    - **Reset-After-Send:** This is especially useful to generate impulses/triggers, if your device e.g. on a keypress does not send "pressed" and "released" events, but only one "pressed" impulse. Reset-After-Send will send a 0 after each incoming event.
    - UI: Enabling and disabling the checkbox responses with some seconds delay - the checkbox toggles back and after some seconds, shows the currently set value.
- Requires LoxBerry V1.4.0+

Version 0.8.1 RELEASE

- Incoming values are trimmed (removed leading and terminating blanks) to work with the Text-To-Value Conversion.

Version 0.8.2 PRE-RELEASE

- Updated Perl Net::MQTT::Simple lib to 1.23
- Corrects a connection issue of the lib to Mosquitto with version higher V1.5.8

Version 0.8.3 RELEASE

- Performance improvements for the Incoming Overview
    - Close HTTP collapsible if UDP is opened, and vice-versa
    - Only render opened collapsible
    - Optimized updating HTML table
    - Update interval of the Overview is calculated dependent to the number of elements
    - "Advanced Table View" checkboxes now are viewed in classic design instead of jQuery Mobile style

Version 0.9.0.1 PRE-RELEASE

- Incoming Overview: Button to purge Mosquitto's retain database (local Mosquitto installation only)
- Incoming Overview: Button to retransmit all data to Miniserver without cache (for testing of Miniserver configuration)
- Incoming Overview: Incoming data are shown also if no Miniserver is configured
- Restart button also restarts Mosquitto (local Mosquitto installation only). With external broker, still only the Gateway is restarted
- Mosquitto:
    - MQTT Websocket protocol. Websocket is available on port 9001. Use the same credentials. Websocket TLS currently not supported. (after update, Mosquitto may need to be restarted)
    - Changed Mosquitto database auto-save interval to 1 day (before: 30 min) to reduce sd writes. Shutdown/Restart still triggers saving of the db on sd card.
- New installations (not update) changed default configuration:
    - HTTP Transmission : enabled
    - UDP Transmission: disabled
    - Expand JSON data: enabled
- Gateway: Fixed wrong encoding in topic names (e.g. with umlauts)
- Plugin developers:
    - Plugin's UDP interface now also supports json as data format (compared to the simple udp interface, this also supports blanks in topic names) → MQTT Gateway - HTTP- und UDP-Interface
    - Your own plugin can inject subscriptions, conversions to the MQTT Gateway plugin during installation and runtime → MQTT Gateway for plugin developers. Your injections are applied from the Gateway on-the-fly without restart.

Version 0.9.1 PRE-RELEASE

- Generic POST-GET-JSON Receiver - Generic Callback URL for non-MQTT devices: MQTT-Gateway - Generischer POST / GET / JSON Empfänger
- Fixed: JavaScript error

Version 0.9.2 PRE-RELEASE

- Fixed an error with Mosquitto during update of the plugin. Mosquitto did not update properly during the installation process.

Version 1.0 RELEASE

- Uninstall will uninstall Mosquitto

Version 1.0.1 RELEASE

- Enhanced: Implemented a delay of 10ms on "reset-after-send" topics between sending the value and sending the 0 (Miniserver network processing was too slow in some situations). The delay is adjustable in the mqtt.json config file (Main.resetaftersendms)
- Enhanced: Boolean Conversion does not translate an empty string to 0 anymore. Empty now stays empty.
- Enhanced: Deleting a value in the webif (X button) now also deletes the line in the webif.
- Enhanced: Removed logfile entries about Incoming Overview state requests for more clearness in the log
- Fixed: Corrected encoding (utf8) inconsistencies via the different interfaces (mqtt, udp, http, webif), e.g. with German umlauts.

Version 1.1 PRE-RELEASE

- New: Publish test messages from the UI: Open the right-side help flyout and open "Quick Publisher"
- New: Send different subscription data to different/multiple Miniservers
- Enhanced: Daemon creates a logfile on startup
- Fixed: Overview: Delete button was hidden in some resolutions

Version 1.1.1 RELEASE

- Updated Net::MQTT::Simple library to 1.24, with LoxBerry/Mosquitto rapid publich patch (see https://github.com/Juerd/Net-MQTT-Simple/issues/11)

Version 1.1.2 PRE-RELEASE

- Allows to skip forwarding of selected data to the Miniserver from UI (to save Miniserver resources)
- Fixes topic matching for redirection of data to other Miniservers (pipe in the subscriptions)

Version 1.1.3 PRE-RELEASE

- Introducing healthcheck for LoxBerry Healthcheck (LB2.2+)
- Fixed another issue with topic matching for redirection of data to other Miniservers

Version 1.1.4 RELEASE

- Update for Net::MQTT::Simple library (adjusted send delay for Raspberry Pi 4 to 17ms)

Version 2.0.0 PRE-RELEASE

- See the What's New document: MQTT Gateway 2.0 What's New
- Known Issues: UTF-8 encoding issues may occour with receiver and transformer functionality.

Will be fixed until the final release.

Version 2.0.1 PRE-RELEASE

- Fixed utf-8 issue in Generic POST/GET receiver
- Fixed utf-8 issues in UDP Transformers

Version 2.0.2 PRE-RELEASE

- Fixed an issue of added blank at the end of a incoming udp message, with publish/retain

Version 2.0.3 PRE-RELEASE

- Fixes: Topics older than 24 hours in UDP Incoming Overview leave "undefined" entries in list
- Fixes: Mqttgateway stops after some seconds on LoxBerry 1.x (LB2.0 feature used in Transformers throws exception on LB1.x)
- UDP View: Added filter buttons for filtered and non-filtered entries

Version 2.0.4 RELEASE

- Copy button to copy VI names / command recognitions to clipboard
- UDP port is hidden if UDP transfer is disabled
- New UDP Transformer: sort (https://www.loxwiki.eu/x/kYRWBQ)
- Enhanced Shelly UDP Transformer: tunable white (https://www.loxwiki.eu/x/_QBABQ)
- Transformer calls are logged
- % character is now replaced by _ (% is not allowed in Loxone)
- Project MQTT to LoxBerry-Core: MQTT connection settings are now stored in LoxBerry's general.json

Version 2.1.0 RELEASE

- Fixes issue "ERROR: Error on JSON expansion: Inappropriate ioctl for device" with newer json library
- receive.php: Also use json from post variables

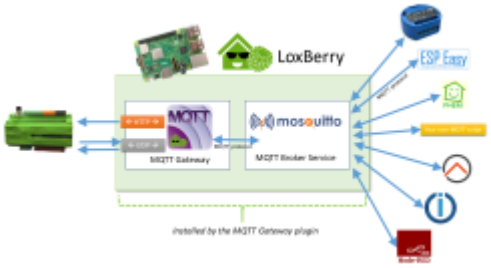
Einführung / Video Tutorial

Download

Der direkte Download des Releases befindet sich in der Tabelle oben.

Repository: https://github.com/christianTF/LoxBerry-Plugin-MQTT-Gateway

Kommunikationsdiagramm



(Anklicken zum Vergrößern)

Sieh dir links im Navigationsbaum weiterführende Artikel zu MQTT an, um die Funktionsweise besser zu verstehen.

Begriffserklärung

Der **"MQTT Broker"** ist ein Dienst, der als Vermittler von Nachrichten dient. Das MQTT Gateway installiert automatisch den Broker **Mosquitto**. Der Broker speichert den aktuellen Zustand aller angeschlossenen Geräte. Über die Konfiguration kann auch ein Broker auf einem anderen Server angegeben werden. Das Plugin unterstützt Authentifizierung, jedoch keine Verschlüsselung zum Broker.

Ein **"Topic"** im MQTT-Jargon ist eine eindeutige Adresse, mit der ein Gerät oder eine Schaltaktion erreicht wird. Ein Topic sieht ein bisschen aus wie eine Pfadangabe, z.B. `shellies/shelly-5345672/relay/1` (ohne / am Beginn) und adressiert eindeutig das Gerät, das auf dieses Topic hört.

Mit einer **"Subscription"** wird ein Topic (oder ein Topic-Baum) abonniert, d.h. mit Subscriptions definierst du, welche Daten du empfangen möchtest. Eine Subscription shellies/# (# steht für einen Joker) abonniert alle Daten, die unterhalb von shellies/ eingehen. Diese Daten werden vom Plugin an den Miniserver weitergeleitet. Mit den Subscriptions kannst du gezielt steuern, welche Daten du tatsächlich zum Miniserver weiterleiten möchtest.

Funktion des Plugins

Das Plugin installiert zuerst automatisch den Mosquitto MQTT Broker (Hostname "localhost"). Erfahrene Benutzer können auch eine Verbindung zu einem anderen Broker einrichten. Das Plugin unterstützt kein TLS zum MQTT-Broker.

Das Plugin fungiert verbindet mit diesem MQTT-Broker, und leitet MQTT-Nachrichten an den Miniserver weiter. Außerdem kannst du vom Miniserver aus Nachrichten an das MQTT Gateway Plugin senden, die ins MQTT-Netzwerk, und somit an deine Geräte, weitergeleitet werden. Somit kannst du mit dem Plugin sowohl Stati deiner Geräte empfangen, als auch Schaltbefehle an diese senden.

Über das Webinterface in den Subscriptions können Topics abonniert werden. Die abonnierten Topics werden wahlweise per UDP, oder per HTTP Webservice an den eingestellten Miniserver übermittelt.

Das Senden von Nachrichten vom Miniserver an MQTT geht am besten per UDP.

Schnell-Konfiguration am Beispiel eines Shelly 2

Settings - MQTT an Loxone Miniserver

Einstellungen

Einstellung	Standard	Beschreibung
Per HTTP Webservice übermitteln ("Set virtual inputs via HTTP webservice")	Aktiv	**Ressourcenschonender als UDP!** Die Nachricht wird direkt per HTTP-REST Webservice an den Miniserver übertragen. Dafür müssen selbst entsprechende Virtuelle Eingänge mit exakt dem hier dokumentierten Namen verwendet werden. Das Plugin verwendet die Nachricht als Bezeichnung für den Eingang, wobei / in _ konvertiert werden: shellies/shelly-12345/relay/1 wird zu **shellies_shelly-12345_relay_1** Schau ins Logfile, dort werden die Bezeichnungen angezeigt. Beachte die Berechtigungen in Loxone! Der Loxberry-Benutzer muss Rechte auf die VIs haben!
Per UDP übermitteln ("Send data via UDP")	Inaktiv	Nachrichtenformat (für die Befehlserkennung): MQTT: shellies/shelly-12345/relay/1=off (oder =0) Befehlserkennung: **MQTT: shellies/shelly-12345/relay/1=\v**
Miniserver	Erster Miniserver	Die abonnierten Daten werden an diesen Miniserver gesendet.
Miniserver-UDP-Port	11883	Ist UDP aktiviert, werden Nachrichten an diesen UDP-Port des Miniservers übermittelt. (wird nur angezeigt, wenn UDP-Übertragung aktiv ist)
Booleans konvertieren	Ein	In MQTT werden häufig Strings bei der Nachricht verwendet, beispielsweise "off" oder "on". Das Plugin konvertiert als Booleans identifizierte Strings in die Werte 0 oder 1. Strings, die erkannt werden, siehe LoxBerry::System::is_enabled bzw. LoxBerry::System::is_disabled

JSON expandieren	Ein	Werden JSON-Daten übertragen, wird die JSON-Struktur expandiert und an den http-Namen angehängt. Topic/abcde={"action":"on","linkquality":39} wird zu Topic_abcde_action = on (oder 1) Topic_abcde_linkquality = 39 Auch Objekte und Arrays in beliebiger Hierarchie werden aufgelöst. Am besten in der Incoming Overview anschauen, wie die Daten nach der Expandierung aussehen.
TAB Subscriptions Abonnierte Topics		Die Angabe erfolgt mit einer Zeile pro Topic. Die Topics müssen vom MQTT-Device dokumentiert sein. Eine # definiert dabei einen "Joker" (wie man sonst den * verwendet), z.B. shellies/# abonniert alle Meldungen innerhalb des Topics shellies/. Siehe unten die Kurzübersicht bekannter MQTT-Geräte und Topics.

TAB Conversions
 	
Eigene Wert-Zu-Text Konvertierung
		Da der Miniserver keine Texte weiterverarbeiten kann, können eingehende Nachrichten von einem String zu einem Wert konvertiert werden.
		
		Beispiel:
		
		`button-up=1`
		`button-down=2`
		
		Sendet ein Device `Topic_abcde_status=button-down`, wird dies als `Topic_abcde_status=2` an den Miniserver übertragen.
		
		Beispielsweise in einem Status-Baustein kann der Wert weiter verarbeitet werden.

TAB Incoming Overview Anzeige der übermittelten Daten		Hier werden die per HTTP und UDP übermittelten Daten angezeigt. Die Aktualisierung erfolgt automatisch im Sekundentakt, ohne dass ein Browser-Refresh notwendig ist. In diesem Fenster können mit Copy&Paste virtuelle Eingänge/Texteingänge, oder für UDP die entsprechenden Befehlserkennungen kopiert werden.

Den Button zum Speichern ganz unten nicht vergessen. Die neue Konfiguration wird beim Speichern automatisch übernommen.

Siehe auch: MQTT - Schritt für Schritt: MQTT -> Loxone

Settings - Miniserver an MQTT

Einstellung	Standard	Beschreibung

Gateway UDP IN-Port	11884	Eingangsport am LoxBerry für UDP-Nachrichten, die an MQTT-Geräte weitergeleitet werden. Folgende Befehle werden unterstützt: * `<topic> <value>` → löst eine Publish-Nachricht aus * `publish <topic> <value>` → löst eine Publish-Nachricht aus * `retain <topic> <value>` → löst eine Retain-Nachricht aus (wird am Broker gespeichert) * `publish <transformer> <topic> <value>` → sendet eine Publish-Nachricht an einen UDP Transformer * `retain <transformer> <topic> <value>` → sendet eine Retain-Nachricht an einen UDP-Transformer * `reconnect` → Löst eine Neuverbindung und ein forciertes Übermitteln der Daten an den Miniserver aus Beispiel 1: `shellies/shellyswitch-32BA7F/relay/1/command on` Beispiel 2: `retain wohnzimmer/temperature 12.5` Beispiel 3: `reconnect` Beispiel 4: `publish http2mqtt web/request https://1.2.3.4/`

Wenn der Miniserver neu startet, empfiehlt es sich, mittels der Loxone-Zeit "Startimpuls" ein `reconnect` per UDP-Nachricht an das Gateway zu übermitteln. Beispiel im "Siehe auch"-Link.

Siehe auch: MQTT - Schritt für Schritt: Loxone -> MQTT

Settings - MQTT Broker-Einstellungen

Einstellung	Standard	Beschreibung
Lokalen Mosquitto-Broker verwenden ("Use the local Mosquitto MQTT broker")	Ja	Ist dies aktiviert, prüft das Plugin beim Reboot, ob Mosquitto läuft, und startet diesen gegebenenfalls. Aktiv: Lokale Mosquitto-Inszanz wird vom Plugin verwaltet (inkl. Änderung von Benutzername und Passwort) Inaktiv: Du kümmserst dich selbst um den Broker
MQTT Broker Adresse	localhost	Hier kann der Hostname angegeben werden, mit dem das Gateway verbinden soll. Wird beim Hostnamen kein Port angegeben, wird der Standardport 1883 verwendet. Eine Portangabe erfolgt per hostname:port

MQTT Broker Benutzername MQTT Broker Passwort	loxberry <generiert>	Der automatisch installierte Mosquitto-Broker wird mit dem Benutzer loxberry und einem generierten Passwort installiert. Diese Informationen kannst du bei der Einrichtung von Devices von hier kopieren. **Wenn du Benutzername und Kennwort nachträglich änderst, musst du das auch bei allen bereits konfigurierten Geräten machen!** Leerer Benutzer und leeres Kennwort deaktivieren die Authentifizierung am lokalen Broker. Bei externen Brokern wird hier deren Benutzername und Kennwort eingegeben.
Mosquitto WebSocket port	9001	MQTT Clients mit Websocket-Support können mit diesem Port verbinden. WebSocket-Clients können im Webbrowser laufen und erhalten die Daten gepushed, statt regelmäßig abfragen zu müssen.
Broker Pre-Shared Key (TLS-PSK)	<generiert>	Das TLS-PSK (Pre-Shared Key) Verfahren erlaubt die verschlüsselte Verbindung von Clients mit zuvor ausgetauschtem Schlüssel. Der Client muss TLS-PSK unterstützen. Zertifikatsbasierte Verschlüsselung ist derzeit nicht implementiert.

Settings - Weitere Einstellungen ("Still more settings")

image2021-2-16_8-32-28.png?version=1&modificationDate=1613460748000&api=v2

Performance-Einstellung des Gateways: "Data Transfer Performance"

Die Einstellung regelt, wie viel CPU-Zeit das Gateway zur Abarbeitung der Nachrichten (in beide Richtungen) beanspruchen darf.

Mittels eines integrierten Reglers wird auf die gewünschte *CPU-Belastung* ausgeregelt. Diese Logik passt sich automatisch an die Systemgeschwindigkeit, und an die Anzahl und den Durchsatz der zu übertragenden Daten an. Die Regelung läuft kontinuierlich, etwaige Lastspitzen werden ausgeglichen. **Eine Änderung der Performance-Einstellung dauert einige Minuten, bis sie spürbar wird.**

Standardwert ist "**Fast (Default)**", was im Vergleich zu den Version 1.x in etwa einer *zehnfachen* Beschleunigung entspricht (auf einem Raspberry Pi 1 / Zero ca. dreifach).
Die Einstellung "**Moderate**" ist etwa dreimal so schnell wie früher, "**Energy Saver**" wird (je nach Hardware) vermutlich langsamer sein als früher, und ist für Taster-Übertragungen eher nicht geeignet.
Mit der Einstellung "**Very fast**" ist praktisch keine Verzögerung mehr spürbar, wohingegen mit "**NO LIMITS!**" je nach Hardware der LoxBerry fast voll ausgelastet ist und alles andere träge wird.

Wenn du willst, kannst du die Wirkung der Regelung beobachten, indem du an der Shell mit `top` die CPU-Auslastung des ''%%mqttgateway.pl%%'' Prozesses verfolgst. Die Regelanpassung passiert im Minutentakt.

MQTT Subscriptions

Hier definierst du, welche Daten du abonnieren möchtest. Jede Subscription steht in einer Zeile. Unter welchem Topic die Daten gesendet werden, bestimmt das Absender-Gerät (bzw. das, was du dort einstellst). Siehe dazu den ausführlichen Artikel, wie Subscriptions und Topics funktionieren, unter MQTT Gateway - Subscriptions und Topics.

Beispiel:

```
shellies/#
nuki/#
```

Verteilung der Daten an verschiedene Miniserver

Wenn du mehrere Miniserver hast (z.B. Gateway-Konzentrator-Betrieb, oder mehrere Wohneinheiten mit unabhängigen Miniservern), möchtest du vielleicht unterschiedliche Daten an unterschiedliche Miniserver senden.

Wenn du eine Subscription ganz normal eingibst (z.B. `shellies/#`) , werden deren Daten immer an den Standard-Miniserver gesendet, den du auf der ersten Seite angeben hast.

Mit einer Pipe hinter der Subscription und der oder den Miniserver-Nummern (aus dem Miniserver-Widget) kannst du das Datenziel ändern:

```
shellies/#|2
shellies/shellyem3-DC4F227649B8/#|3
nuki/#|1,2,3
fhem/#
```

- Zeile 1: Alle `shellies/`-Daten werden an den Miniserver 2 gesendet (außer `shellies/shellyem3-DC4F227649B8/`)
- Zeile 2: `shellies/shellyem3-DC4F227649B8/`-Daten gehen an Miniserver 3
- Zeile 3: Daten von `nuki/` werden sowohl an MS1, MS2 als auch MS3 gesendet.
- Zeile 4: Keine Pipe, daher gehen die Daten von `fhem/#` an den Standard-Miniserver

Dazu noch folgende Hinweise:

- Die Priorisierung der Reihung erfolgt entsprechend der Selektivität des Topics, d.h. `shellies/shellyem3-DC4F227649B8/#` ist spezifischer als `shellies/#`, deswegen "gewinnt" in unserem Beispiel bei Daten von `shellyem3-DC4F227649B8` der selektivere Eintrag (Daten gehen an MS3, die restlichen shellies/# Daten an MS2). Die Reihenfolge im Textfeld hat keinen Einfluss darauf!
- Die Topic-Joker **+** und **#** werden vollständig unterstützt.
- Du kannst eine Plugin-Subscription auch an einen anderen Miniserver senden lassen, indem du die gleiche Subscription oben im Textfeld mit einer Pipe definierst. Hat das Plugin beispielsweise `nuki/#` registriert, kannst du mit einer Zeile `nuki/#|2` die Daten an einen anderen Miniserver senden lassen.
- Alle Einstellungen auf der *Settings*-Seite und in der Incoming Overview (z.B. HTTP oder UDP, Boolean Conversion, Reset-After-Send,...) treffen für alle Miniserver zu. Das Datencaching von

LoxBerry wird pro Miniserver angewandt.
- Fehler bei der Übertragung zu einem Miniserver führen in der Regel zu einer Warnung oder Fehlermeldung im Log, nicht aber zum Abbruch/Absturz des Programms.

Subscription Filter Expressions (RegEx)

Eine Subscription kann eine Menge Daten übertragen, die du möglicherweise gar nicht auf dem Miniserver brauchst. Anhand von Filtern kannst du bestimmte Daten wegfiltern. Diese Filter-Möglichkeit funktioniert auch innerhalb extrahierter JSON-Daten.

Jede Zeile enthält eine Regular Expression - wenn das RegEx bei einem eingehenden Datensatz zutrifft, wird dieser Datensatz gefiltert und nicht an den Miniserver weitergeleitet.

In der Incoming Overview werden gefilterte Datensätze durchgestrichen angezeigt - so kannst du prüfen, ob deine RegEx-Zeilen funktionieren.

Während der Eingabe ins Feld wird geprüft, ob deine Regular Expression auch wirklich valide ist.

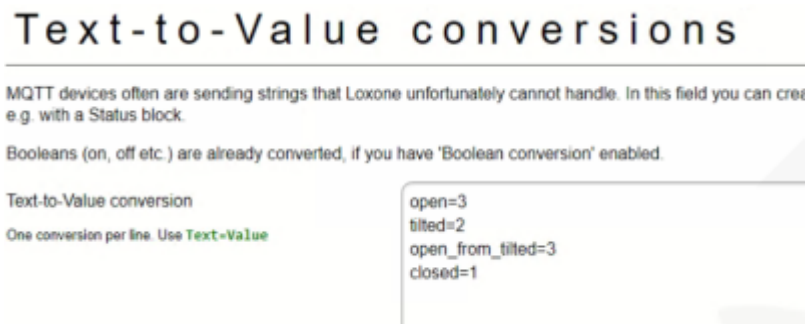
Subscriptions durch Plugins

Um dir die Einrichtung zu erleichtern, können andere Plugins direkt im MQTT Gateway Subscriptions definieren. Wenn Plugin-Subscriptions vorhanden sind, findest du diese auf der *Subscriptions*-Seite ganz unten. Du brauchst dann diese Subscriptions nicht nochmals oben einzugeben.

Text-to-Value Conversions

Viele Geräte senden Text für ihren Status, z.B. "up" und "down", oder "open" oder "closed". Loxone ist nicht sehr geschickt, was den Umgang und Weiterverarbeitung von Texten angeht. Um diese Texte in Zahlen umzuwandeln (mit denen der Miniserver mehr anfangen kann), kannst du hier Text-to-Value Conversions, also Konvertierungen durchführen.

In jede Zeile eine Konvertierung.



Dieses Beispiel zeigt den Status von Enocean Fenstergriffen. Sendet der Fenstergriff "open", wird 3 an den Miniserver weitergesendet, ist gekippt ("tilted"), wird 2 weitergeleitet. Mit den Zahlen kann man Logiken abbilden, oder diese mit einem Status-Baustein wieder als Text darstellen lassen.

Diese Konvertierungen gelten für sämtliche Daten, die eingehen (unabhängig vom Topic). Es kann also nur eine Konvertierung pro Text geben. Duplikate werden ausgefiltert und im Log als Warnung angezeigt.

Conversions von anderen Plugins

Auch andere Plugins können automatisch Conversions anlegen. Die findest du am Ende der Seite, wenn welche vorhanden sind.

Incoming Overview

In der "Incoming Overview" werden alle von MQTT eingehenden Daten der letzten 24 Stunden angezeigt.

Je nach in Settings gewählten Übertragungsmethode zum Miniserver - HTTP oder UDP - werden zwei unterschiedliche Listen dargestellt. Die Anzeige der Daten aktualisiert sich automatisch. Wenn du Text markierst, wird die Aktualisierung pausiert.

HTTP Virtual Inputs (Übertragung per HTTP)

Oben kannst du Einträge suchen und nach dem Status filtern.

Die Symbole links zeigen, ob

- die letzte Übertragung erfolgreich war, oder
- der VI am Miniserver nicht existiert,
- der Datensatz seit dem letzten Start noch nicht übertragen wurde (z.B. durch den Cache),
- der Datensatz gefiltert wurde (Subscription Filter oder "Do-Not-Forward" Flag) , oder ob vielleicht
- das Gateway keine Berechtigungen auf den VI am Miniserver hat.

Ein Symbol bedeutet, dass dieser Virtuelle Eingang bei der letzten Übertragung am Miniserver nicht existiert hat (noch nicht angelegt, oder vertippt?)

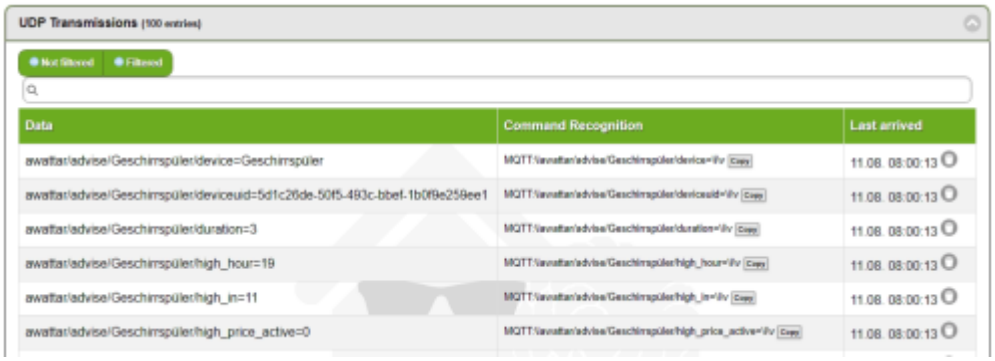
Das Symbol bedeutet, dass dieser Wert bisher nicht neu an den Miniserver übertragen wurde. Das liegt in der Regel am Cache, dass gleiche Werte nicht mehrfach übertragen werden.

Wenn du bei Einträgen ein Symbol bekommst, prüfe in der Loxone Config, ob der LoxBerry-Benutzer an deinem Miniserver die nötigen Berechtigungen für diesen VI oder insgesamt an deinem Miniserver hat.

Der VI hat einen Copy-Button. Damit kannst du zügig neue Virtuelle Eingänge am Miniserver einrichten.

UDP Transmissions (Übertragung per UDP)

Wenn du UDP als Übertragungsart gewählt hast, werden dir die Daten, und die möglichen Befehlserkennungen angezeigt.



Du kannst mit dem Copy-Button direkt die Befehlserkennung kopieren, um sie am Miniserver in neue Virtuelle UDP-Eingangs-Befehle (Befehlserkennung) zu kopieren.

Show details and advanced settings

Je nach Übertragungsart - HTTP oder UDP - werden dir zusätzliche Informationen und Optionen angezeigt.

- **Topic:** Zeigt die originale Topic-Bezeichnung an, auch bei JSON-extrahierten Daten.
- **Disable Cache:** Dieser Wert übergeht das Caching von LoxBerry, und sendet den Wert immer, auch wenn er gleich bleibt.
- **Reset after send:** Nachdem der Wert gesendet wurde, wird automatisch eine 0 nachgesendet. Das eignet sich für Geräte (z.B. Taster), die lediglich einen Wert (Impuls) senden und dann auf diesem Wert "stehen bleiben". Ein weiterer Impuls würde verursachen, dass wieder der gleiche Wert gesendet würde, und der Miniserver erkennt keine Änderung.
- **Do not forward:** Mit aktivierter Checkbox wird dieser Datensatz nicht an den Miniserver weitergeleitet. Das schont Ressourcen am Miniserver. Die Daten von diesen Topics werden in der Incoming Overview durchgestrichen dargestellt.
- **Lösch-Symbol:** Damit wird der Wert am Broker und in der Incoming Overview gelöscht.

Bei HTTP-Übertragung wird dir rechts angezeigt, wann der Datensatz zuletzt zu welchem Miniserver übertragen wurde. Wenn du in den Subscriptions die Übertragung an mehrere Miniserver nutzt, wird das hier für jeden Miniserver angezeigt. Das ist insbesondere interessant in Bezug auf das Caching des Plugins: Bei "Last Arrived" wird angezeigt, wann der Datensatz per MQTT eingegangen ist. In der Details-Ansicht wird angezeigt, wann der letzte Übertragungsversuch an den Miniserver war. Liegen die Zeiten auseinander, hat der Cache die Übertragung eines gleichen Wertes "eingespart".

Transformers & Logs

Neu im MQTT Gateway 2.x sind sogenannte Transformers. Damit ist es möglich, vom Miniserver

eingehende Daten vor der Übertragung an MQTT zu modizieren.

Die detaillierte Anleitung zu den Transformers findest du hier: MQTT Gateway - UDP Transformers

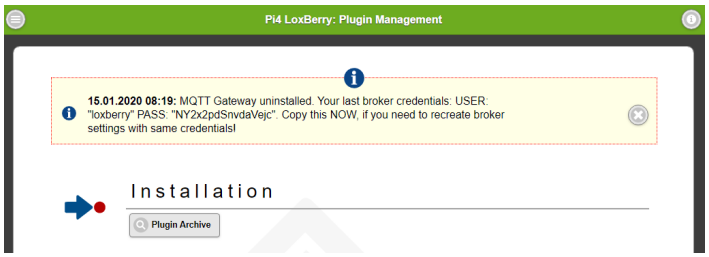
Bekannte MQTT-Geräte und Subscription-Topics

Liebe Plugin-Benutzer! Bitte pflegt diese Tabelle mit euren eingesetzten Geräten, MQTT-Infos und Links!

Die Liste ist jetzt in diesem Unter-Artikel: MQTT-Gateway - Bekannte MQTT-Geräte und Subscription-Topics

Weitere Hinweise

- Die Deinstallation des MQTT-Gateway Plugins **deinstalliert** den Mosquitto-Broker. Bei der Deinstallation werden die zuletzt verwendeten Mosquitto-Zugangsdaten als Benachrichtigung in die Plugin-Verwaltung gesendet.
- Das Plugin integriert sich in den LoxBerry Selbsttest / LoxBerry Healthcheck. Geprüft wird, ob das MQTT Gateway Service läuft, ob es Meldungen bei der Öffnung der Netzverbindungen gab, und ob die Broker-Verbindung korrekt funktioniert.



- Neue MQTT-Geräte ausprobieren, etwa weil Befehle weniger gut dokumentiert sind oder man die idealen Werte erst durch ausprobieren finden muss, ist über die Loxone Config schwierig (auf Miniserver laden, Neustart, Warten...) Da bietet es sich an, die Steuerung via MQTT erst einmal mit einer separaten Client Software auszuprobieren, bevor dann die fertigen Befehle in die Loxone Config übernommen werden. Einen Überblick über passende Software bzw. Apps dafür bietet https://www.hivemq.com/blog/seven-best-mqtt-client-tools/
- Ebenfalls zum Testen nutzen kannst du den Quick Publisher des MQTT Gateways. Wenn du im Plugin ganz rechts oben das "i" klickst, oder auf die "Transformers & Logs" Seite gehst, findest du dort den "Quick Publisher". Damit kannst du rasch Werte an MQTT-Topics senden zum Ausprobieren.

Roadmap

- Fehlerkorrekturen
- Gateway: Eigenes Topic zur Steuerung des MS
- Integration in LoxBerry-Core

Fragen stellen und Fehler melden

MQTT Gateway - Troubleshooting Guide

GitHub: https://github.com/christianTF/LoxBerry-Plugin-MQTT-Gateway/issues

Forum: https://www.loxforum.com/forum/projektforen/loxberry/plugins/176025-loxberry-mqtt

Fehlerbeschreibungen

Für jeglichen Kontakt bzw. Probleme **immer** mitliefern:

- das Logfile
- die MQTT-Spezifikation des angebundenen Devices (z.B. direkter Link zur MQTT-Dokumentation des Geräts)
- Screenshots bzw. Beschreibung der durchgeführten Einstellungen usw.

Siehe auch

- MQTT-Gateway - Bekannte MQTT-Geräte und Subscription-Topics
- MQTT - Schritt für Schritt: MQTT -> Loxone
- MQTT - Schritt für Schritt: Loxone -> MQTT
- MQTT Gateway - HTTP- und UDP-Interface
- MQTT Gateway - UDP Transformers
- MQTT Gateway - FAQ
- MQTT Gateway for plugin developers

- Informationen und Best Practices zu Topics und Subscrictions: https://www.hivemq.com/blog/mqtt-essentials-part-5-mqtt-topics-best-practices/
- Miele Home MQTT Gateway: https://www.loxforum.com/forum/projektforen/loxberry/plugins/176967-miele-mqtt-loxberry-mqtt
- Verschiedene MQTT-Interfaces von Devices und Diensten: https://github.com/hobbyquaker/awesome-mqtt#interfaces
- Fronius Hybrid und *go-eCharger* Bridge: https://github.com/akleber/mqtt-connectors