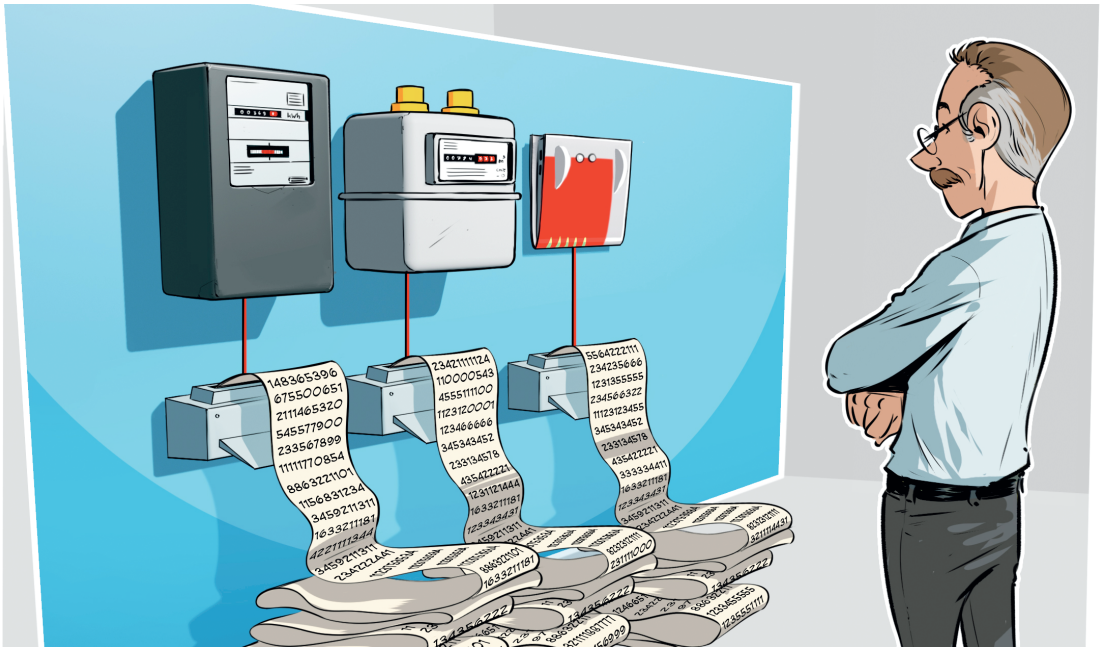


Daten in InfluxDB speichern

In Smart Homes fallen oft Unmengen von Daten an. Viel davon ist unnütz, aber einige Informationen können Gold wert sein. Mit ihnen kann man Wartungsintervalle antizipieren oder auch schauen, wann und wo der Haushalt den meisten Strom verbraucht. Wer Node-Red einsetzt, kann die anfallenden Daten im Nu in der Zeitreihendatenbank InfluxDB speichern und später mit Werkzeugen wie Grafana visualisieren.

Von Merlin Schumacher

Wird der Staubsaugerroboter wirklich mit der Zeit langsamer oder ist das nur Einbildung? Braucht die Waschmaschine tatsächlich länger für Kochwäsche als für Buntwäsche? Lohnt es sich die Heizung abzuschalten, wenn man nur eine halbe Stunde weg ist, oder läuft der Kessel danach erst mal eine Stunde auf Volllast? All diese Fragen sind schwer zu beantworten, falls man sich nicht mit einer Stoppuhr und einem Block neben die Haushaltsgeräte setzen möchte. Wohl dem, der seinen Lebensraum versmartet hat und die zur Auswertung nötigen Daten frei Haus bekommt. Die smarte Schaltsteckdose misst den Energiebedarf der Waschmaschine, ver-

sendet ihn per MQTT und liefert so ein Indiz, wann diese mit ihrer Arbeit fertig ist. Die intelligente Heizungssteuerung kommuniziert mit dem Heizkessel und man sieht per Cloud-API, wie lange es dauert, die Wohnung um 1 °C zu erwärmen. Der Staubsaugerroboter zählt die Quadratmeter, wenn er arbeitet und sein lokales API liefert die Daten auf Anfrage.

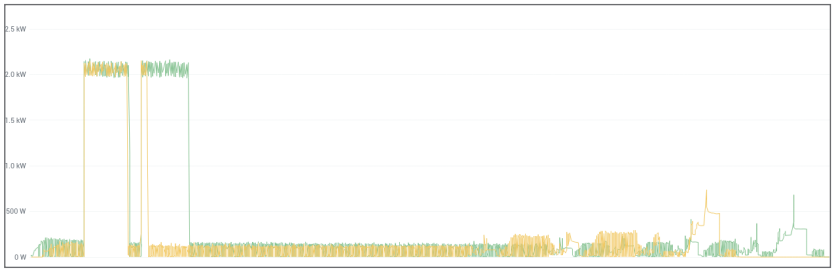
Einrichtung

All diese Daten landen dann in einer Smart-Home-Steuerung wie Node-Red. Sie kann die Werte direkt in eine InfluxDB-Datenbank überführen. Die Auswertung und Visualisierung kann dann ein Werkzeug wie Grafana übernehmen. Die Grundlagen zu InfluxDB und Grafana haben wir auf Seite 128 [1]. Die Lektüre des Grundlagenartikels ist empfehlenswert, wenn Sie noch keine Erfahrungen mit InfluxDB haben. Für Node-Red ist lediglich die Installation eines Node-Moduls über die Palette notwendig. Klicken Sie dazu auf das Hamburger-Menü rechts oben und anschließend auf „Palette verwalten“. Wechseln Sie auf den Reiter „installieren“. Tippen Sie in das Suchfeld „node-red-contrib-influxdb“ ein und klicken Sie auf „installieren“. Anschließend finden Sie in der Liste der Nodes im Abschnitt „Speicher“ drei neue Nodes: „influxdb in“, „influxdb out“ und „influx batch“. Die Dokumentation des Moduls finden Sie über ct.de/wvw4.

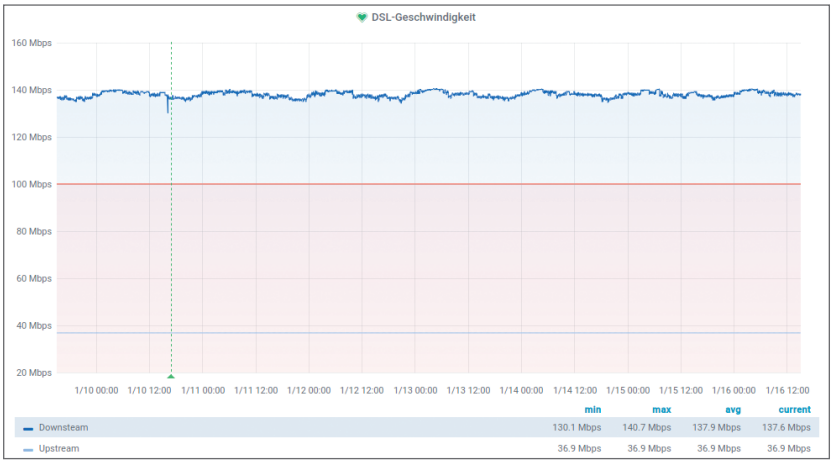
Ziehen Sie nun einen Node vom Typ „influxdb out“ in den Flow und doppelklicken Sie darauf. Im dadurch geöffneten Einstellungsreiter klicken Sie auf den kleinen Stift in der Zeile „Server“. Tragen Sie dort den Hostnamen des InfluxDB-Servers ein. Den Port brauchen Sie im Normalfall nicht zu ändern. Das Feld „Database“ nimmt den Namen einer bestehenden Datenbank entgegen. Haben Sie noch keine erzeugt, können Sie das mit dem zu InfluxDB gehörenden Kommandozeilenwerkzeug `influx` erledigen. Geben Sie dort `CREATE DATABASE smarthome` ein, um eine Datenbank mit dem Namen `smarthome` anzulegen. Bei Benutzername und Passwort tragen Sie die entsprechenden Daten ein, sofern Sie einen Benutzer für InfluxDB festgelegt haben. Versenden Sie die Daten über das Internet über eine mit TLS verschlüsselte Verbindung, legen Sie noch eine „TLS Configuration“ fest. Liefert der Server ein gültiges Zertifikat (etwa von Let’s Encrypt) aus, brauchen Sie nichts zu tun, außer auf Hinzufügen zu klicken. Haben Sie ein selbstsigniertes Zertifikat hinterlegt, müssen Sie die entsprechenden Zertifikate und Schlüssel hochladen. Hört der Server auf mehrere Hostnamen, müssen Sie den gewünschten in das Feld Servername eintragen, damit der TLS-Terminator weiß, an wen die Anfrage gehen soll. Um die Konfiguration des „influxdb in“-Node abzuschließen, legen Sie mit dem Feld Measurement noch den Titel der Messreihe fest. Hier bietet sich der Name des Geräts, Raumes oder Sensors an, dessen Daten Sie speichern. Mit Grafana können Sie Daten aus beliebigen Messreihen miteinander kombinieren, daher ist es nicht nötig, die Daten eines Sensors auseinanderzudröseln, weil man dessen Messwert und Batteriestatus zu unterschiedlichen Zwecken erfassen möchte. Dazu später mehr.

Ob die Verbindung zur InfluxDB klappt, können Sie testen, indem Sie ein Inject-Node mit dem Eingang des „influxdb in“-Node verbinden. Ändern Sie dessen Nutzdatentyp auf Number und den Wert auf eine beliebige Zahl (zum Beispiel 4). Klicken Sie auf „deploy“ und anschließend auf den Knopf am Inject-Node. Durch einen Blick in den Debug-Reiter rechts sehen Sie, ob das Speichern geklappt hat. Lief alles durch, sollte der Reiter leer sein. Bei Problemen sehen Sie die Fehlermeldung von InfluxDB.

Nun wurde der Messwert 4 in der Datenbank gespeichert und gleich mit einem Zeitpunkt verknüpft. Sie brauchen nicht dafür zu sorgen, dass InfluxDB mitgeteilt bekommt, wann gemessen wurde. Wer komplexere Daten speichern will, kann diese im JSON-Format übergeben. Letztendlich senden die Nodes nur JSON-Daten an die Endpunkte von InfluxDB. Komplexere Datenstrukturen senden Sie entweder mit zusätzlichen Tochterelementen von `msg.payload` wie etwa `msg.payload.energy` oder mit einem eigenen Objekt als Nutzdaten. Letzteres ist sinnvoll, wenn man Mess- und Metadaten übertragen will.



Das Trio aus Node-Red, InfluxDB und Grafana verarbeitet, speichert und visualisiert Messwerte und Daten aus dem Smart Home, zum Beispiel den Energieverbrauch zweier Waschgänge.



Mit Node-Red lässt sich auch die aktuelle DSL-Geschwindigkeit aus der Fritzbox in eine InfluxDB überführen.

Quellenarbeit

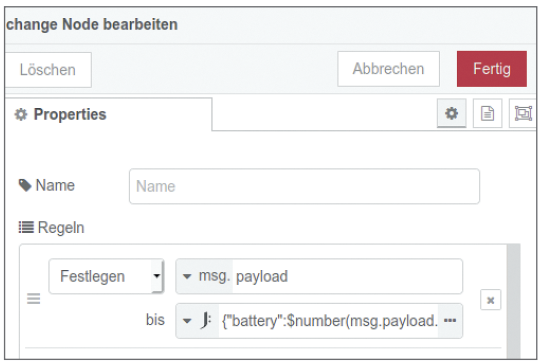
Auch wenn die meisten Messwerte in der InfluxDB Zahlen sein werden, kann sie auch Strings oder binäre Werte speichern. Der Typ des gespeicherten Werts hat Einfluss auf die möglichen Darstellungsformen in Grafana. Aus einer Folge von Strings lassen sich nun mal keine schönen Diagramme bauen, dafür aber Statusanzeigen, Listen und Heatmaps. Man muss daher aufpassen, welche Daten man an die Datenbank schickt, denn schnell hat man übersehen, dass die Zahlenwerte im Nachrichtenobjekt gar keine sind, sondern Strings.

Wer Daten im JSON-Format von einem MQTT-Broker bezieht, muss sie vorher konvertieren. Das übernimmt der MQTT-Input-Node. Dazu müssen Sie in dessen Einstellungen unter Output die Option „Parsed JSON object“ wählen. Bei anderen Quellen, die falsche Datentypen liefern, kann man sich mit JSONata helfen. JSONata ist eine Umwandlungssprache für JSON-Daten. Sie ist sehr mächtig und beherrscht allerhand Operationen. Dieser Artikel kann nur einen kleinen Einblick in JSONata vermitteln. Die vollständige Dokumentation finden Sie über ct.de/wvw4.

Mit dessen Funktion `$number()` können Sie Strings in Zahlen umwandeln. Dazu fügen Sie ein Change-Node in einen der Flows ein und definieren „Festlegen“ als Operation für eine oder mehrere der Regeln. Als Typ für den Nachrichteninhalt wählen Sie die Option „Ausdruck“. Tragen Sie als Wert `$number(msg.payload.energy)` ein, dann wird der Wert von `msg.payload.energy` durch JSONata verarbeitet und in eine Zahl umgewandelt – je nach Bedarf als Ganzzahl oder Gleitkommawert. Wollen Sie mehrere Werte verändern, können Sie diese direkt als Objekt oder Array ausgeben. Für ein Objekt mit zwei Werten kommt der folgende Code zum Einsatz:

```
{
  "power": $number(
  msg.payload.battery),
  "link": $number(
  msg.payload.linkquality)
}
```

Das Ergebnis ist ein Objekt, das die Elemente `power` und `link` mit entsprechenden Zahlenwerten, die vorher als Strings in `battery` und `linkquality` enthalten waren. Auch wenn Sie nur eine Auswahl der gelieferten Daten in die InfluxDB leiten wollen, bietet sich dieser Weg an. Er ist oft übersichtlicher als der über eine Function-Node oder sich mit zig Change-Node-Regeln zu behelfen.



Mit JSONata kann man Werte, die in Node-Red ankommen, flexibel umwandeln.

Nicht jeder Messwert ist taufrisch. Geräte, die auf Batterien laufen, messen möglicherweise eine Zeit und schicken die Messwerte dann gebündelt ab. Um diese Werte in die InfluxDB zu senden, muss man beim Speichern nur den Zeitpunkt der Messung mitschicken. Dafür reicht es, den Schlüssel `time` mit einer JavaScript-Date-Instanz des Messzeitpunkts zu übergeben:

```
{
  power: 10,
  link: 90,
  time: new Date(
  "2020-01-02T19:41:14Z")
},
```

Das Beispiel sorgt dafür, dass zwei Messwerte mit einem festgelegten Zeitpunkt in der InfluxDB landen. Um für mehr Ordnung in den gesammelten Daten zu sorgen, bieten sich Tags an. Diese lassen sich ebenfalls mittels JSON-Daten festlegen.

```
[{
  power: 10,
  link: 90,
  time: new Date(
  "2020-01-02T19:41:14Z")
},
{
  root: "kitchen",
  type: "plug"
}]
```

Die Tags landen als Objekt gemeinsam mit den Messdaten in einem Array. Hier werden mit `room` und `type` Attribute festgelegt, die später in Grafana ausgewählt werden können, um alle Messdaten eines Raumes oder eines Gerätetyps aus verschiedenen Messreihen zu sammeln.

Mess en masse

Aus einigen Quellen kommt nicht nur ein Wert nach dem anderen, sondern gleich ein ganzer Batzen verschiedenen Ursprungs, die in unterschiedlichen Messreihen landen sollen. Diese kann man mit dem „influx batch"-Node speichern. Dafür müssen sie jedoch im richtigen Format vorliegen. Im Kasten oben sehen Sie ein Beispiel.

Darin stehen Messwerte von zwei Sensoren. Neu sind `measurement` als Bezeichnung der Messreihe, `fields` ist ein Objekt mit den Messwerten und deren Namen, einem Objekt mit den gewünschten Tags und `timestamp` mit einem Zeitpunkt der Messung. InfluxDB legt die gelieferten Werte dann am festgelegten Zeitpunkt in der Datenbank ab.

```
01 [{
02     Measurement: "environment",
03     fields: {
04       temp: 5.5,
05       light: 678,
06       humidity: 51
07 },
08     tags: {
09       room: "bathroom"
10 },
11     timestamp: new Date("2020-01-
                            02T19:41:14Z")
12 },
13 {
14     measurement: "motion",
15     fields: {
16       presence: true
17 },
18     tags: {
19       room:"hallway"
20 },
21     timestamp: new Date("2017-12-
                            28T19:41:14Z")
22 }]
```

Mit einem „influx batch"-Node lassen sich Massen von Werten in einem Rutsch in der InfluxDB speichern.

Schön machen

Das Einsammeln der Daten ist nur der kleinste Teil der Arbeit. Davor sollte man überlegen, welche davon wirklich nötig sind und welche weg können. Auch deren passende Visualisierung ist eine Kunst für sich. Es ist ratsam, entspannt ein paar Tage zu tüfteln. Wenn man wichtige Daten doch nicht mitgenommen hat, merkt man das eher, als wenn man alle mitnimmt und hinterher den Wald vor lauter Bäumen nicht mehr sieht. Um Probleme zu erkennen, kann man dann gleich Grafana einspannen, das Alarme auf unterschiedlichsten Kanälen (etwa E-Mail oder Messenger) ausgeben kann. (mls) ct

Literatur

[1] Jan Mahn, Geschichtsschreiber, InfluxDB: Spezialisierte Datenbank für Messwerte und Logging, c't 5/2019, S. 154

Weiterführende Informationen:

www.ct.de/wvw4