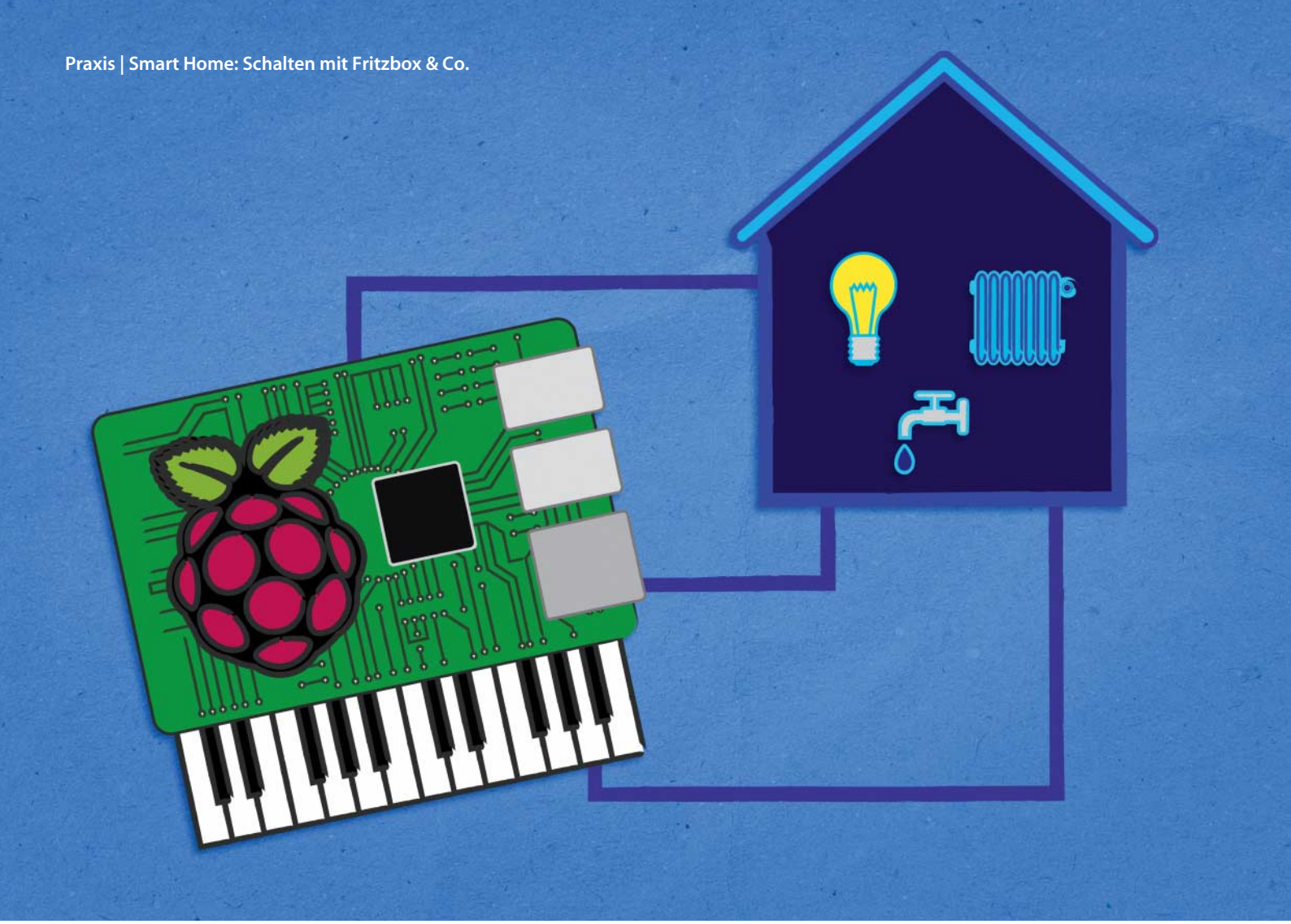

Thomas Dreßler

Code-Schalter

Schalten mit Fritzbox & Co.

Viele Fritzbox-Benutzer wissen gar nicht, dass ihr Router auch messen und schalten kann. Mit ein paar Handgriffen lassen sich AVMs Smart-Home-Adapter sogar unabhängig von der Fritzbox-Oberfläche in eigene Programme einbinden.

Für seine DECT-fähigen Fritzboxen bietet die Berliner Router-Schmiede AVM seit 2013 passende Smart-Home-Komponenten an. Sie kommunizieren über den zuvor nur für die Telefonie genutzten DECT-Standard mit der Box, erfassen Verbrauchs- oder Temperaturwerte und ermöglichen Fernbedienungsoptionen per App. Das ist für den Anfang schon nicht schlecht, aber komplexere Wenn-dann-Schaltungen sind mit Fritzbox-Bordmitteln nicht umzusetzen.

Stand März 2016 sind vier zur Fritzbox passende Smart-Home-Komponenten verfügbar. Da sind zum einen die Zwischenstecker DECT!200 und Fritz!Powerline 546e. Beide lassen sich schalten und messen auch noch ganz nebenbei den Stromverbrauch der angeschlossenen Geräte. Der Fritz!DECT Repeater 100 lässt sich zwar nicht schalten, erlaubt – wie der DECT!200 – aber zumindest das Auslesen seines integrierten Temperatursensors. Als jüngstes Mitglied der Familie kam vor Kurzem noch der Heizkörperregler Comet DECT der Firma Eurotronic hinzu.

Löblicherweise hat AVM alle für eine externe Ansteuerung notwendigen Grundlagen seiner AVM-Home-Automation-Schnittstelle (AHA-API) offen gelegt (siehe c't-Link). Alle Zugriffe erfolgen über den Webservice des Fritz-OS (homeautoswitch.lua). Über einen HTTP-GET-Request wird eine Anfrage gestellt, das Fritz-OS antwortet in der Regel mit der Ausgabe von XML- oder JSON-Code (siehe Kasten auf Seite 115).

Dieser Beitrag beschreibt, wie man AVMs Smart-Home-Komponenten auch außerhalb der Bedienoberfläche der Fritzbox unter Anwendung des AHA-API nutzen kann – entweder um Daten auszulesen oder um die Schaltzustände der Aktoren zu verändern. Das funktioniert entweder durch direkte Eingabe in der Kommandozeile oder über einen automatisierten Aufruf durch zusätzliche Smart-Home-Software.

Als Programmierplattform nutzen wir einen Raspberry Pi mit einem Standard-Image (Raspbian Jessie). Der Raspi bringt alle notwendigen Voraussetzungen mit, um die

Komponenten auszulesen und zu steuern. Anschließend lässt er sich sogar im Dauereinsatz als Server nutzen, um größere Smart-Home-Projekte zu realisieren. Wir nutzen für unser Projekt die Skriptsprache PHP. Grundsätzlich kann man die hier beschriebenen Schritte auch mit einem entsprechend präparierten Windows-PC oder Mac durchführen.

Unter dem c't-Link am Ende des Artikels finden Sie die in diesem Artikel erwähnten Beispiel-Skripte, die Sie nach Ihren Bedürfnissen anpassen können.

Ärmel hoch

Zum Experimentieren benötigen Sie eine DECT-fähige Fritzbox mit aktuellem Fritz-OS nebst AHA-API – die recht verbreiteten Modelle 7390 und 7490 funktionieren in jedem Fall. Handelt es sich um ein OEM-Modell eines bestimmten Providers, kommt es auf einen Versuch an. Eventuell wurde dessen angepasste Firmware um das AHA-API erleichtert.

Der Zugriff auf die Aktoren erfolgt meist indirekt: Der DECT!200 zum Beispiel sendet seine Daten via DECT an die Fritzbox – von dort sollen sie vom Raspi ausgelesen werden. Andersherum übermitteln Sie Schaltbefehle an die Fritzbox, die diese wiederum per DECT an den Aktor sendet. Für beide Operationen ist zunächst eine gültige Anmeldung am Fritz-OS notwendig.

Auf dem Raspi benötigen Sie neben einem halbwegs aktuellen PHP ab Version 5.3 die Erweiterungen „Multibyte String" (mbstring) sowie „SimpleXML". All dies holen Sie sich mit der Eingabe eines Einzeilers im Terminalfenster des Raspi aus dem Netz:

```
sudo apt-get install -y php5
```

Ein Wort zur Sicherheit: Für den Zugriff auf die Fritzbox müssen Sie deren Zugangsdaten in den Skripten hinterlegen. Dadurch übernehmen Sie die Verantwortung für die Sicherheit des Routers und der angeschlossenen Geräte. Als Minimalvorkehrung sollte man zumindest das Standard-Passwort des Raspi durch die Eingabe des passwd-Befehls in der Kommandozeile von „raspberry" in ein individuelles Passwort ändern.

Für die Bearbeitung der Skripte genügen Raspi-Bordmittel. Als Programmierwerkzeuge dienen der vorinstallierte Text-Editor, File-Manager und Browser – nun kann es losgehen.

Zugriffskontrolle

Um mit der Fritzbox zu kommunizieren, benötigt man eine gültige Session-ID. Sie wird im sogenannten Challenge-Response-Verfahren mit der Fritzbox ausgehandelt (siehe Kasten). Wird eine aktive ID für zehn Minuten nicht für Anfragen genutzt, verfällt sie automatisch.

Das unter dem c't-Link als Download verfügbare PHP-Skript SIDauslesen.php nimmt Ihnen das Erstellen einer gültigen Session-ID ab, indem es die im Kasten beschriebenen Schritte ausführt. Legen Sie über File-Explorer oder Kommandozeile im pi-Verzeichnis einen Arbeitsordner AVM an, in den Sie das Skript anschließend kopieren. Danach müssen Sie den Code an Ihre Umgebung anpassen: Als Hostname geben Sie im Kopfbereich `fritz.box` oder die IP-Adresse Ihrer Fritzbox ein, unter Password das von Ihnen zur Anmeldung auf der Fritz-Oberfläche genutzte Passwort. Der Username wird nur benötigt, wenn Sie ihn auch für die Anmeldung auf der Weboberfläche der Fritzbox eingerichtet haben – ansonsten bleibt das Feld leer.

Die fertige Funktion lässt sich nun über den `include`-Befehl in beliebige PHP-Skripte einbetten und ausführen. Über den Editor erstellen Sie nun ein kleines PHP-Skript, um die Funktion zu testen:

```
<?php
  //Funktionstest
  include_once('SIDauslesen.php')
  echo "Session-ID: ".
    get_sid($loginurl,$username,$password);
?>
```

Im Editor speichern Sie das Skript unter dem Namen funktion.php ebenfalls im AVM-Ordner ab. Über das Terminalfenster lässt sich die Funktion nun testen. Wechseln Sie mit dem Befehl „cd AVM" zunächst in das richtige Verzeichnis und starten Ihr Testprogramm:

```
php funktion.php
```

Das Programm liefert Ihnen auf der Kommandozeile nun die von Ihrer Fritzbox ausgegebene, 16-stellige Session-ID. Mit der frisch generierten ID kann man nun an die Fritzbox herantreten. Ein über den Browser des Raspi abgesetzter http-Request zeigt alle verbundenen Smart-Home-Komponenten. Er nutzt die AHA-Funktion `getdevicelistinfos`:

```
http://fritz.box/webservices/homeautoswitch.lua?↵
      switchcmd=getdevicelistinfos&sid=<16-stellige SID>
```

Die Fritzbox antwortet im Browser nun mit einer XML-Response, die Informationen über die angeschlossenen Aktoren enthält. Bei einer Funksteckdose kann dies beispielsweise so aussehen:

```
<devicelist version="1">
  <device identifier="08761 0123456" id="16"
    functionbitmask="896" fwversion="03.37"
    manufacturer="AVM"
    productname="FRITZ!DECT 200">
    <present>1</present>
    <name>Mein DECT 200</name>
    <switch>
      <state>1</state>
      <mode>auto</mode>
      <lock>0</lock>
    </switch>
```

Neben Smart-Home-Zwischensteckern von AVM wie dem Fritz DECT!200 (Mitte) gibt es inzwischen auch einen passenden Thermostat von Eurotronic (rechts).

```
    <temperature>
      <celsius>241</celsius>
      <offset>0</offset>
    </temperature>
  </device>
</devicelist>
```

Jede Komponente verfügt über eine eindeutige Actor Identification Number (AIN) – die in der XML-Ausgabe im Attribut identifier des device-Tags zu finden ist. Hat man den zu schaltenden Aktor identifiziert, lässt sich über die Funktion setswitchtoggle der Schaltzustand des Aktors ebenfalls per HTTP-Request verändern:

```
http://fritz.box/webservices/homeautoswitch.lua?ain=<12-stellige AIN>&switchcmd=setswitchtoggle&sid=<16-stellige SID>
```

Klick

Zum Auslesen des Zwischensteckers per PHP dient das Skript list-devices.php aus dem Download-Paket (c't-Link). Es gehört ebenfalls in das AVM-Verzeichnis auf dem Raspi. Der Start des Skriptes mit der Zeile php list-devices.php im Terminalfenster sollte nun den aktuellen Status aller Aktoren liefern:

```
AIN 08761 0123456 (Mein DECT 200)
Temperatur:(Temp:24.1C, Offset:0.0) ;
Steckdose(Status:On);
```

Das Skript zeigt das Zusammenspiel von Anmeldung und Auswertung. Für jede Teilfunktion einer Komponente liest es die Antworten individuell aus und interpretiert sie. Nimmt man statt des DECT!200 beispielsweise den Heizkörperregler Comet zur Hand bedeutet dies, dass die gelieferten Werte zwischen 16 und 52 einer Temperatur zwischen 8 und 26 °C in 0,5-Grad-Schritten entsprechen – ähnliche Umrechnungen sind auch bei anderen Aktoren und Werten nötig.

Die Zeilen 26 ff. des „list-devices"-Skriptes verdienen besondere Beachtung:

```
//Check Funktionsbitmask
$mask=(integer)$attributes['functionbitmask'];
$has_temperatur=($mask & (1<<8))>0;
$has_switch=($mask & (1<<9))>0;
```

Hier wertet das kleine Programm die sogenannte Funktionsbitmaske aus, anhand derer die Smart-Home-Komponenten ihre jeweiligen Fähigkeiten bekannt geben. Das Attribut functionbitmask wird als Integer-Wert ausgegeben, dessen zwölf Bits anhand des AHA-Apis zu entschlüsseln sind. Für den DECT!200 liefert die Abfrage mit getdeviceinfos den Wert 896, binär also 0011 1000 0000. Bit 7 steht für die Eigenschaft „Energiemessgerät, Bit 8 für einen Temperatursensor und Bit 9 signalisiert die Schaltbarkeit des Aktors (siehe c't-Link).

Für die Interpretation der Bitmaske nutzt das „list-devices"-Skript logische Bitoperationen. Der erhaltene Wert für die Funktionsbitmaske wird nacheinander mit einem durch eine Bit-Schiebeoperation gewonnenen Vergleichswert UND-verknüpft. Ist in der Maske das entsprechende Bit gesetzt, ist der resultierende Wert größer Null und der Aktor unterstützt die jeweilige Funktion.

Schaltautomat

Wie man die Schaltsteckdose per Skript ein- und ausschaltet, demonstriert die Datei schalter.php. Auch diese macht von unserem SessionID-Skript SIDauslesen.php Gebrauch. Im Texteditor müssen Sie noch die AIN des zu schaltenden Aktors eintragen. Durch Verändern des Parameters in Zeile 7 $action=false in $action=true lassen sich durch einfaches Speichern zwei PHP-Skripte (on.php/off.php) zum An- und Ausschalten des Aktors generieren.

Unter Linux kann man solche Skripte zum Beispiel in einem Cronjob nutzen, der nachts einen Warmwasserboiler zwecks Energieersparnis abschaltet. Dazu trägt man die AIN des betroffenen Zwischensteckers in das off-Skript ein und legt es als „warmwasser_aus.php" im AVM-Ordner ab. Danach legt man als root (sudo --i) einfach eine passende Definition an. Über den Texteditor erstellen Sie die Datei wasser im Verzeichnis /etc/cron.d/, deren Inhalt aus folgender Zeile besteht:

```
30 22 * * * pi cd /home/pi(AVM && /usr/bin/php warmwasser_aus.php >>cron.log 2>&1
```

Nun wird jeden Tag um 22:30 Uhr der Warmwasserboiler ausgeschaltet. Zugegeben – diesen Job hätte man auch über die Web-Oberfläche der Fritzbox erledigen können. Dennoch verdeutlicht das Beispiel, wie man über die Kommandozeile automatische Vorgänge auf den Aktoren auslösen kann.

Abseits des AHA-API bietet Fritz-OS viele weitere Möglichkeiten, um Daten abzufragen. Das Skript zum Auslesen der Session-ID (SIDauslesen.php) leistet auch hierbei gute Dienste. Im Netz finden sich zahlreiche Programmierbeispiele, mit denen man mehr aus seiner Fritzbox herausholen kann. Eine gute Anlaufstelle für Interessierte ist das IP-Phone Forum (siehe c't-Link).

Mit den beschriebenen PHP-Skripten können Sie Ihre AVM-Smart-Home-Geräte nun relativ einfach per Kommandozeile oder in Shell/Batch-Skripten ansteuern. Richtig Spaß macht es, wenn man die Geräte in eine herstellerübergreifende Smart-Home-Software einbindet. Der Autor verwendet für seine Heimautomatisierung zu diesem Zweck IP-Symcon. Weitere Tipps zur Einbindung der AVM-Skripte finden Sie auf der Projektseite.

(Thomas Dreßler/sha@ct.de)

c't Skripte und Dokumentation: ct.de/yd1b

Frage-Antwort-Spiel

Beim Challenge-Response-Verfahren sendet der Server auf eine initiale Session-Anfrage einen Code (Challenge), den der Client nach einem beidseitig bekannten Verfahren zusammen mit seiner eigentlichen Antwort verarbeitet. In einem zweiten Request legt der Client das Ergebnis als Response vor.

Der Server vollzieht die gleichen Verarbeitungsschritte mit der übermittelten Challenge und dem bei ihm hinterlegten Passwort nach. Am Ende muss er für eine gültige Anmeldung auf den gleichen Wert wie die Response vom Client kommen.

Bei AVM wird zur Abfrage der Challenge die Login-Applikation (login_sid.lua) zunächst mit einem GET-Request ohne Parameter aufgerufen:

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

Als Antwort liefert sie einen XML-String mit der vorgeschlagenen Challenge und dem Tag „SID" gefüllt mit Nullen:

```
<?xml version="1.0" encoding="utf-8"?>
<SessionInfo>
  <SID>0000000000000000</SID>
  <Challenge>8a1532c1</Challenge>
  <BlockTime>0</BlockTime>
  <Rights></Rights>
</SessionInfo>
```

Mit Hilfe der erhaltenen Challenge kann man nun die Antwort konstruieren. An die Challenge wird durch Minus-Zeichen getrennt das Passwort der Fritzbox angehängt, der ganze String dann mit Hilfe einer Multibyte-Funktion in die erforderliche UTF16-LE-Kodierung konvertiert und letztlich daraus ein MD5-Hash gebildet. Der Hash verbunden mit der Challenge ist das kodierte Password, das übergeben wird:

```
http://fritz.box/login_sid.lua?username=&response=8a1532c1-e416f62700997cab86e49c6c080dbe27
```

Das eigentliche Password wird nie offen übers Netz übertragen, aber das Fritz-OS auf der Box kann durch Nachvollziehen der Schritte mit dem hinterlegten Password prüfen, ob es auf den gleichen Hash kommt. Als Antwort erhält man im XML-Tag „SID" die benötigte SID:

```
<?xml version="1.0" encoding="utf-8"?>
<SessionInfo>
  <SID>bcaa5cea0ae71889</SID>
  <Challenge>abd36dc9</Challenge>
  <BlockTime>0</BlockTime>
  <Rights>,,,,,</Rights>
</SessionInfo>
```

Besteht die SID nur aus Nullen wie beim Request, war die Anmeldung fehlerhaft und man kann den Versuch mit der neu übertragenen Challenge wiederholen.