Der Beirag ***Sensordaten per SNMP verteilen*** ist im Tagungsband der *Chemnitzer Linux-Tage 2013*, S.77 – 84, ISBN 978-3-941003-80-4 erschienen.

Der Tagungsband kann unter

`http://chemnitzer.linux-tage.de/2013/vortraege/proceedings/`

bestellt werden.

Sensordaten per SNMP verteilen

Axel Wachtler (axel@uracoli.de) **Ralf Findeisen (rfindeis@gmail.com)**

Im Beitrag wird gezeigt, wie man bekannte Monitoring-Tools (z.B. Nagios oder Icinga) zusammen mit SNMP zur Anbindung eines drahtlosen Sensornetzwerkes verwenden kann. Dazu werden als Konzentrator ein Carambola-Board, das unter OpenWrt läuft und verschiedene 802.15.4-Funkmodule eingesetzt. Die Funkmodule kommunizieren über das einfache P2P-Protokoll aus dem *µracoli*-Projekt. Das Nagios-Monitoring-System ist auf einem Linux-Server installiert.

1 Einleitung

Mit drahtlosen Sensornetzwerken (engl. WSN: wireless sensor network), die auf Grundlage des Standards IEEE 802.15.4 [15] arbeiten, können kleine stromsparende Funkmessknoten aufgebaut werden, die sich vorteilhaft u.a. in der Gebäudeautomatisierung einsetzen lassen. Gerade auch im privaten Bereich ist es hinsichtlich von Energieeinsparung und der Überwachung des Wohnraumklimas nützlich, wenn Messwerte wie Temperatur oder Luftfeuchte sowie der Status der Fensterkontakte dezentral erfasst und zentral abgefragt werden können. Die Steuerung von Aktoren wie Heizung oder Klimaanlage ist aufgrund der bidirektionalen Kommunikation zwar möglich, bei einer selbstgebauten Aktorensteuerung sollte aber grosse Sorgfalt walten.

1.1 System-Überblick

Die Architektur des Sensor-Systems ist in Bild 1 dargestellt. Die Sensorknoten senden ihre Messwerte zu einem Gatewayknoten, der über eine serielle Schnittstelle an den Konzentrator angebunden ist. Der Konzentrator ist ein OpenWrt-Router, auf dem ein SNMP-Dämon läuft, der per WLAN oder Ethernet im Netzwerk erreichbar ist. Auf dem Monitoring-Server läuft Nagios oder ein anderes Monitoring-System, das die Daten der einzelnen Sensorknoten über SNMP-Anfragen ausliest oder Kommandos an die Sensorknoten sendet.

1.2 Anforderungen

Ein Messwert-Monitoring-System soll robust funktionieren, deshalb wurde auf langjährig erprobte Komponenten wie Net-SNMP und Nagios gesetzt. Die Verwendung von SNMP [16] macht das System generisch nutz- und erweiterbar. SNMP ist schmalbandig und verwendet das UDP-Protokoll. Einfache Implementierungen

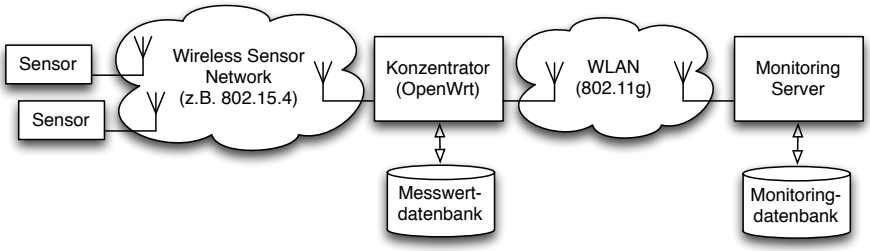

Bild 1: Architekturüberblick

von SNMP-V1 sind bereits für 8-Bit-Mikrocontroller möglich [1]. Durch die Verwendung eines OpenWrt-Routers steht allerdings mehr Rechenleistung und Speicher zur Verfügung, so dass die sichere Protokollversion SNMP-V3 verwendet werden kann.

Im Beitrag wird zu Demonstrationszwecken das sehr einfache aber kryptographisch ungesicherte *µracoli*-P2P-Protokoll verwendet. Dies stellt eine erhebliche Sicherheitsschwachstelle dar, da die ausgetauschten Nachrichten im WSN nicht authentisiert oder verschlüsselt sind. Die vorgestellte Methode kann jedoch auf jedem beliebigen WSN-Protokoll-Stack übertragen werden, da der WSN-Dämon die Daten von der seriellen Schnittstelle des WSN-Gateway-Knotens parst und weiterverarbeitet.

2 Das Sensornetzwerk und die Sensoren

Drahtlose Sensor Netzwerke verwenden je nach eingesetztem Protokollstack unterschiedliche Topologien. Im einfachsten Fall handelt es sich um eine Punkt-zu-Punkt-Verbindung oder um ein Sternnetz. In dieser Konfiguration übermitteln die Sensorknoten ihre Messwerte direkt zum Konzentrator. Beispiele für diese Netzwerktopologie sind der IEEE-802.15.4-MAC-Layer oder das *µracoli*-P2P-Protokoll.

Da die Sensorknoten vorrangig stromsparend aufgebaut sind, ist auch die abgestrahlte Sendeleistung meist nur gering. Beim Einsatz von 2.4 GHz Funktransceivern in Gebäuden ergeben sich zusätzliche Probleme durch die geringe Wellenlänge in Form von Interferenzen und Abschattungen. Zusammen mit anderen Funk-Technologien wie WLAN oder Bluetooth, die um das gleiche Frequenzband konkurrieren, ergibt sich eine eingeschränkte Funkreichweite, und der Gateway-Knoten ist unter Umständen nicht mehr direkt erreichbar. Ist das der Fall, so müssen die Informationen der Sensoren durch Router-Knoten zum Gateway-Knoten weitergeleitet werden. Beispiele für Routing-Protokolle sind ZigBee, IETF 6LowPan, Leightweight-Mesh (Atmel) oder Route Under MAC (Atmel).

Als Sensorknoten finden einige TicTacToe-Platinen aus dem Workshop vom vergangenen Jahr Verwendung [2]. Der darauf befindliche Atmega128RFA1 vereint Micro-

controller, Funkinterface und Temperatursensor in einem Bauteil. Die Firmware des Sensorknotens misst periodisch Temperatur und Spannung und sendet die Daten an den Konzentrator. Als Sendeprotokoll kommt das einfache P2P-WUART-Protokoll zum Einsatz, das in der Payload des Datenrahmens einen Klartextstring überträgt.

Am Konzentrator-Knoten ist das WSN-Gateway über eine serielle Schnittstelle angeschlossen. Auf dem Gateway läuft die *µracoli*-Wireless-UART-Firmware, die die Payload aller empfangenen Rahmen auf der seriellen Schnittstelle ausgibt. Stellt man die Ausgabe des Gateways mit einem Terminal-Programm dar, so erhält man folgende Textzeilen:

```
ADDR=0004, T=18.2, Vcc=2.93
ADDR=0001, T=-3.3, Vcc=2.99
ADDR=0003, T=26.7, Vcc=3.25
...
```

Diese einfache Methode zur Übertragung der Messwerte ist gegenüber einem angepassten Binärprotokoll vielleicht nicht besonders effizient, erlaubt aber die einfache Erweiterung um zusätzliche Messwerte, die vom Sensorknoten erfasst werden können.

Das P2P-Protokoll dient primär zu Demonstrations-, Lern- und Testzwecken. Da es keine kryptographische Sicherung der Messwerte implementiert, sollte es zur Übertragung von kritischen Informationen in einem Produktivsystem nicht verwendet werden. Die Firmware ist im Detail in [13] beschrieben.

3 Der Konzentrator

Der Konzentrator besteht aus einem OpenWrt-Router und dem Gateway-Knoten. Im Beispielprojekt wurde als Routerhardware das Carambola-Board der Firma 8devices verwendet [4]. OpenWrt [8] ist eine Linuxdistribution, die speziell für die Verwendung auf embedded Geräten angepasst ist. OpenWrt umfasst eine Vielzahl an Software-Paketen, die mit dem Paketmanager `opkg` installiert werden können.

3.1 OpenWrt-Software

Wie man die Software-Umgebung zur Erstellung eines Carambola-Images installiert und nutzt, ist in [4] beschrieben. Weitere Erfahrungen zur Inbetriebnahme des Boards sind unter http://uracoli.nongnu.org/carambola.html zu finden.

Wenn die Entwicklungsumgebung läuft, können mit dem Befehl `make menuconfig` die zusätzlich benötigten Software-Komponenten ausgewählt werden.

```
# serielle Treiber für den Gateway-Knoten
kmod-usb-acm, kmod-usb-serial, kmod-usb-serial-ftdi

# Libraries
libsqlite3, librs232, libncurses, libreadline, liblua

# Lua-Module
luaposix, luars232, luasql-sqlite3, lsqlite3

# Tools
lua, sqlite3-cli
```

Im Carambola-Repository sind nicht alle erforderlichen Komponenten enthalten. Durch das leistungsfähige Build-System ist die Erweiterung der fehlenden Komponenten jedoch einfach möglich. Um ein fehlendes Paket zu ergänzen, wird einfach das entsprechende Verzeichnis unter `packages/` anlegt und aus dem OpenWrt-Repository werden das `Makefile` sowie die lokalen Patches und Zusatzfiles kopiert. Danach erscheint automatisch das neue Paket als Eintrag im Konfigurationsmenü.

3.2 Prozessmodell

Zum Schreiben und Lesen der Sensordaten sind zwei Prozesse erforderlich. Auf der WSN-Seite werden die Daten von der seriellen Schnittstelle mit dem WSN-Dämon gelesen. Auf der Netzwerkseite läuft der SNMP-Dämon (siehe Bild 2) und liest die Daten vom WSN-Dämon. Da beide Prozesse vollständig asynchron arbeiten, muss eine Möglichkeit der kollisionsfreien Interprozesskommunikation gefunden werden. Lösungsansätze über einen, mit `flock` abgesicherten File-Zugriff oder der Einsatz von Sockets wurden aus Gründen der Robustheit, des Ressourcenbedarfs und der aufwändigeren Fehlersuche verworfen. Das Problem des kollisionsfreien Zugriffes für mehrere Prozesse wurde letztlich mit einer SQLite3-Datenbank [11] gelöst.

Der WSN-Dämon schreibt die Sensordaten mit UPDATE-Befehlen in die Datenbank. Wenn der SNMP-Dämon ein SNMP-Get-Paket empfängt, liest er die entsprechenden Datensätze mit einem SELECT-Befehl von der Datenbank und leitet sie an den SNMP-Manager weiter.

Da jeder Sensorknoten unterschiedliche physikalische Messwerte liefern kann, wird in der Datenbank je eine Tabelle pro Messwert angelegt.

```
Table tab_LOC (addr, ort)
Table tab_T   (addr, uhrzeit, wert)
Table tab_Vcc (addr, uhrzeit, wert)
...
```

Die Tabelle `tab_LOC` muss manuell gewartet werden, sie beinhaltet die Zuordnung der Adressen der Sensorknoten zu deren Installationsorten. Die Wertetabellen `tab_T` und `tab_Vcc` speichern die Adresse, den Messwert und die Uhrzeit des letzten empfangenen Datensatzes. Anhand des Zeitstempels kann so ein eventueller Ausfall des Knotens detektiert werden.

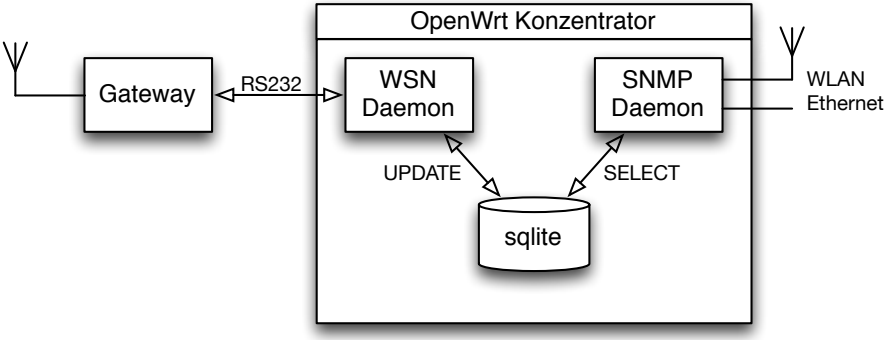

Bild 2: Datenmodell

3.3 WSN-Dämon

Der WSN-Dämon wurde als Lua-Script implementiert. Es liest kontinuierlich die Daten des Gateways von der seriellen Schnittstelle und schreibt sie in die Datenbank. Mit dem Tool `start-stop-daemon` wird das Lua-Script als Prozess auf Systemebene gestartet, angehalten und der aktuelle Status überprüft.

Der WSN-Dämon kann prinzipiell in jeder Programmiersprache geschrieben werden. Da die Speicherressourcen des Carambola-Boards jedoch begrenzt sind, fiel die Wahl auf die Script-Sprache Lua [12].

3.4 SNMP-Dämon

Der SNMP-Dämon ist ein Server-Dienst, der SNMP-Anfragen benatwortet. Die Installation erfolgt mit dem Befehl `opkg install snmpd`. Die Adressierung der Daten erfolgt über die Managment Information Base (MIB). Diese beschreibt die Struktur der Informationen und ist ein hierarchischer Baum von OIDs (Object Indentifier). Jeder OID beschreibt den Inhalt einer Variablen, die les- und/oder schreibbar ist. Für unsere Anwendung liefert die ENTITY-SENSOR-MIB nach RFC3433 [3] passenden Attribute:

```
Objekt-Name             OID
entitySensorMIB         1.3.6.1.2.1.99
entPhySensorTable       1.3.6.1.2.1.99.1.1
entPhySensorEntry       1.3.6.1.2.1.99.1.1.1
...
```

OpenWrt-Systeme werden über das Unified Configuration Interface (UCI) konfiguriert. Für den SNMP-Dämon wird die Datei `/etc/config/snmpd` verwendet. Das Starten des Prozesses erfolgt mit dem Befehl `/etc/init.d/snmpd start`. Dabei wird die eigentliche Konfigurationsdatei `snmpd.conf` erzeugt.

```
Ausschnitt /etc/config/snmpd
# Einbindung der Sensor Table
config pass
    option name MIB-SENSOR
    option prog /bin/wsn_snmpd_pass.sh
    option miboid .1.3.6.1.2.1.99
```

Im Script `/bin/wsn_snmpd_pass.sh` wird der entsprechende ENTITY-SENSOR-MIB Abschnitt implementiert. Das MIB-Attribut wird dem Script als Kommandozeilenparameter übergeben. Als Rückgabewerte liefert es die OID, den Datentyp und den eigentlichen Messwert. Nachdem der Dämon fertig konfiguriert und neu gestartet ist, können mit `snmpwalk` oder `snmpget` die entsprechenden MIB-Attriute über das Netzwerk abgefragt werden. Anzumerken sei noch, dass SNMP keinen Float-Datentyp unterstützt, d.h. man muss sich entweder mit einer Festpunktkonvertierung behelfen oder man verwendet den Datentyp String.

4 Management und Visualisierung auf dem Server

In Wikipedia sind derzeit über fünfzig Netzwerk-Monitoring-Programme aufgezählt [14]. Nagios wurde hier ausgewählt, da es eines der bekantesten Tools in diesem Bereich ist. Das sehr mächtige Programm ist vor allem unter Systemadministratoren zur Überwachung großer IT-Landschaften sehr beliebt. Auf die zu überwachenden Parameter kann mit unterschiedlichen Methoden zugegriffen werden, SNMP ist nur eine von vielen Möglichkeiten.

Nagios setzt die Installation eines Apache-Webservers voraus. Die erforderlichen Pakete werden über den Paketmanager der jeweiligen Distribution installiert. Nach der Installation sind jedoch noch weitere Schritte zur Inbetriebnahme eines funktionierenden Basis-Systems erforderlich [7].

Die Einbindung der Sensor-Messwerte erfolgt durch die Definition von Diensten. Im File `/etc/nagios3/conf.d/konzentrator_nagios2.cfg` wird ein Temperatur-Messwert wie folgt definiert:

```
define service{
    use                     generic-service
    host_name               konzentrator
    service_description     aussen_1
    check_command           check_snmp!1.3.6.1.2.1.99.1.1.1.2
}
```

Da für jeden Sensor und jeden Messwert ein Service definiert werden muss, empfiehlt es sich, dafür ein Generator-Script zu verwenden, das die entsprechenden Definitionsabschnitte anhand der SQlite-Datenbank auf dem Konzentrator-Knoten erzeugt.

Die Funktionalität von Nagios kann durch zahlreiche Plugins erweitert werden. Der zeitliche Verlauf von Messwerten kann u.a. mit dem Plugin PNP4Nagios [9] dargestellt werden.

5 Ausblick

Das in den vorangangenen Abschnitten gezeigte System ist erst ein Anfang. Die Weiterentwicklung dieser Sensorinfrastruktur hat vielfältige Optionen zur Erweiterung und Vervollkommnung.

Denkbar ist beispielsweise die Unterstützung von Mobilgeräten, die Verwendung geeigneter Interfaces zur Verwaltungssoftware oder das Schreiben eines leichtgewichtigen web-basierenden Clients, welcher dann als Applikation auf dem Mobilgerät die Messwerte darstellt und Aktionen anbieten kann. Die heute verfügbaren leistungsfähigen Bibliotheken wie D3 [5] erlauben eine ansprechende Darstellung und daher die Einbindung in publikumssichtbare Systeme, deren Darstellungsanforderungen über pure Nützlichkeit hinausgehen.

Das gezeigte Sensornetzwerk ist auch nicht die einzige Möglichkeit der Messwerterhebung und Übermittlung. Neben verwandten Systemen, wie beispielsweise autarken Sensoren von Seeedstudio, die auch die Verwendung anderer Radiointerfaces erlauben würden [10], kann man auch in Erwägung ziehen, den Nagios-Server um Skripte für die Ansteuerung eines Heizungssteuersystems, wie z.B. MAX! Cube [6] zu erweitern, um so nicht nur die Heizung anhand von Profilen und als Reaktion auf offene Fenster zu steuern, sondern auch auf Ausseneinflüsse einzugehen. Natürlich sind dabei die weiter oben erwähnten Sicherheitsfragen, speziell beim Einsatz von Aktoren zu beachten.

Insgesamt ist die Messung und Automatisierung mit Hilfe günstiger und leicht zu verwendender Sensoren und Überwachungsrechner ein Gebiet mit unserer Meinung nach grossem Potential und wir möchten die Community einladen, mit uns in der Entwicklung fortzuschreiten.

Die Webseite zum Vortrag befindet sich unter http://uracoli.nongnu.org/clt2013.

6 Literatur

[1] *Agentuino - A lightweight SNMP Agent for Arduino Platforms*. URL: http://bit.ly/Wo0QmA.

[2] A. Wachtler et. al. "Tic-Tac-Toe Reloaded". In: *Chemnitzer Linuxtage 2012*. 2012. URL: http://uracoli.nongnu.org/clt2012.

[3] A. Bierman, D. Romascanu und K.C. Norseth. *Entity Sensor Management Information Base*. RFC 3433 (Proposed Standard). Internet Engineering Task Force, 2002. URL: http://www.ietf.org/rfc/rfc3433.txt.

[4] *Carambola - a flexible 802.11n module*. URL: http://www.8devices.com/product/3/carambola.

[5] *D3.js JavaScript Bibliothek*. URL: http://d3js.org.

[6] *ELV Max! Protocol Description*. URL: http://bit.ly/W9kQsf.

[7] *Nagios: Fixing 'error: Could not stat() command file' (on Debian)*. URL: http://bit.ly/vjvkcG.

[8] *OpenWrt*. URL: http://www.openwrt.org.

[9] *PNP4Nagios*. URL: www.pnp4nagios.org.

[10] *Seeedstudio Stalker Waterproof Solar Kit*. URL: http://bit.ly/UVKHDS.

[11] *SQLite Home Page*. URL: http://www.sqlite.org.

[12] *The Programming Language Lua*. URL: http://www.lua.org.

[13] Daniel Thiele und Axel Wachtler. "Funkübertragung mit IEEE 802.15.4 Transceivern - nicht nur für Profis". In: *Embedded Projects Journal* 14 (2012), 22 ff. URL: http://bit.ly/Sf1nJk.

[14] Wikipedia. *Comparison of network monitoring systems*. 2013. URL: http://en.wikipedia.org/wiki/Comparison_of_network_monitoring_systems.

[15] Wikipedia. *IEEE 802.15.4*. 2013. URL: http://de.wikipedia.org/wiki/802.15.4.

[16] Wikipedia. *Simple Network Management Protocol*. 2013. URL: http://de.wikipedia.org/wiki/SNMP.