Contiki-SNMP
—
Does SNMP fit on constrained devices?

Jürgen Schönwälder, Siarhei Kuryla

AIMS 2011, Nancy, 2011-06-16
AVR Raven Hardware

ATmega1284PV microcontroller:
- runs at 20 MHz
- 16K of RAM
- 128K of ROM (Flash)

Contiki-SNMP

- Contiki is an operating system for embedded devices
- SNMP engine (written in C) for constrained devices
- built on top of the Contiki uIPv6 stack (6LoWPAN)
IPv6 over Low-Power Wireless Personal Area Networks

IEEE 802.15.4

- Small frame size (max frame size = 127 bytes)
- Low power devices (some battery operated)
- Limited memory and processing power
- Low bandwidth (max data rate = 250 kbps)
- Large scale and dense deployments
- Devices and channels tend to be unreliable
- Devices may use sleep schedules to conserve energy

IETF 6LoWPAN

- IPv6 over IEEE 802.15.4 (see RFC 4944)
- General motivation and overview (see RFC 4919)
- RPL (routing), COAP (web transfer protocol), ...
Why 6LoWPAN management?

A: “Autonomic devices won’t need management — so don’t waste your time on the wrong problem…”

B: Well, no, for the foreseeable future, you will end up managing the autonomic system (it’s just one more control loop)

A: But even then, SNMP is clearly the wrong choice since we now have RESTful protocols with modern data encoding

B: So let’s do a comparison / competition…

Typical management questions

- How many nodes disappeared during the last night/day?
- How many nodes joined during the last week?
- What is the temperature, pressure, energy usage (add your favorite sensor here) distribution within the network?
- What is wrong with my home automation network?
SNMPv3 end-to-end

- Straightforward direct access to individual 6LoWPAN nodes
- Reuse of existing deployed SNMP-based tools
  - End-to-end security, end-to-end key management
  - Message size and potential fragmentation issues
  - 6LoWPAN nodes must run an SNMP engine
  - Trap-directed polling nature of SNMP has high (energy) costs
SNMPv3 proxies

- Indirect access to individual 6LoWPAN nodes
- Alternate transport encoding can reduce message sizes
- Reuse of existing SNMP-based tools supporting proxies
- Two security domains, different key management schemes
  - 6LoWPAN nodes must run an SNMP engine
  - Trap-directed polling nature of SNMP has high (energy) costs
### SNMPv3 subagents

<table>
<thead>
<tr>
<th>SNMP Manager</th>
<th>SNMPv3</th>
<th>SNMP Agent (6LoWPAN Gateway)</th>
<th>Subagent Protocol</th>
<th>SNMP Subagent</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Internet</td>
<td>6LoWPan Network</td>
<td></td>
</tr>
</tbody>
</table>

- Indirect access to individual 6LoWPAN nodes
- Alternate transport encoding can reduce message sizes
  - Reuse of existing SNMP-based tools supporting contexts
  - Two security domains, different key management schemes
  - 6LoWPAN nodes must run an SNMP subagent
    - Trap-directed polling nature of SNMP has high (energy) costs
SNMPv3 interfacing to data fusion protocols

+ Indirect access to individual 6LoWPAN nodes
+ Leveraging data fusion protocols (in-network aggregation)
+ SNMP agent acting as a cache, no expensive polling
  - Reuse of existing SNMP-based tools supporting contexts
  - Two security domains, different key management schemes
? No direct advantage of 6LoWPAN technology — oops
Contiki-SNMP Overview

General features / limitations

- SNMP messages up to 484-byte length
- Get, GetNext and Set operations
- SNMPv1 and SNMPv3 message processing models
- USM security model, no VACM access control model
- API to define and implement managed objects

USM security algorithms

- HMAC-MD5-96 authentication protocol (RFC 3414)
- CFB128-AES-128 symmetric encryption protocol (RFC 3826)
### Implemented MIB Modules and Static Memory Usage

<table>
<thead>
<tr>
<th>MIB modules</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SNMPv2-MIB</strong> – SNMP entity information</td>
</tr>
<tr>
<td><strong>IF-MIB</strong> – network interface information (no 802.14.5 ifType)</td>
</tr>
<tr>
<td><strong>ENTITY-SENSOR-MIB</strong> – temperature sensor readings</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SNMPv1 and SNMPv3 enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>31220 bytes of ROM (around 24% of the available ROM)</td>
</tr>
<tr>
<td>235 bytes of statically allocated RAM</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SNMPv1 enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>8860 bytes of ROM (around 7% of the available ROM)</td>
</tr>
<tr>
<td>43 bytes of statically allocated RAM</td>
</tr>
</tbody>
</table>
## Flash ROM and Static Memory Usage

### Memory usage by software module (bytes)

<table>
<thead>
<tr>
<th>Module</th>
<th>Flash ROM</th>
<th>RAM (static)</th>
</tr>
</thead>
<tbody>
<tr>
<td>snmpd.c</td>
<td>172</td>
<td>2</td>
</tr>
<tr>
<td>dispatch.c</td>
<td>1076</td>
<td>26</td>
</tr>
<tr>
<td>msg-proc-v1.c</td>
<td>634</td>
<td>6</td>
</tr>
<tr>
<td>msg-proc-v3.c</td>
<td>1184</td>
<td>30</td>
</tr>
<tr>
<td>cmd-responder.c</td>
<td>302</td>
<td>0</td>
</tr>
<tr>
<td>mib.c</td>
<td>1996</td>
<td>6</td>
</tr>
<tr>
<td>ber.c</td>
<td>4264</td>
<td>3</td>
</tr>
<tr>
<td>usm.c</td>
<td>1160</td>
<td>122</td>
</tr>
<tr>
<td>aes_cfb.c</td>
<td>9752</td>
<td>40</td>
</tr>
<tr>
<td>md5.c</td>
<td>10264</td>
<td>0</td>
</tr>
<tr>
<td>utils.c</td>
<td>416</td>
<td>0</td>
</tr>
</tbody>
</table>
### Stack and Heap Usage

#### Maximum observed stack usage

<table>
<thead>
<tr>
<th>Version</th>
<th>Security mode</th>
<th>Max. stack size</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNMPv1</td>
<td>–</td>
<td>688 bytes</td>
</tr>
<tr>
<td>SNMPv3</td>
<td>noAuthNoPriv</td>
<td>708 bytes</td>
</tr>
<tr>
<td>SNMPv3</td>
<td>authNoPriv</td>
<td>1140 bytes</td>
</tr>
<tr>
<td>SNMPv3</td>
<td>authPriv</td>
<td>1144 bytes</td>
</tr>
</tbody>
</table>

#### Heap usage

- not more than 910 bytes for storing an SNMPv1 message
- approximately 16 bytes for every managed object in the MIB
- if a managed object is of a string-based type, then additional heap memory is used to store its value
SNMP Request/Response Latency (varying security)

The graph shows the time (ms) for different versions of SNMP and security levels:

- **SNMPv1**
- **SNMPv3**
  - noAuthNoPriv
  - AuthNoPriv
  - AuthPriv

The bars represent the time taken for transfer and processing, with the total time indicated above each bar.

- **SNMPv1** shows a shorter total time compared to the other versions.
- **SNMPv3 noAuthNoPriv** has a shorter transfer time but a longer processing time compared to the other versions.
- **SNMPv3 AuthNoPriv** and **SNMPv3 AuthPriv** show longer times, with AuthPriv being the longest.
SNMPv1 Request/Response Latency (varying # varbinds)
Bigger Picture (resource requirements of various protocols)

- **mDNS**: 1.0 kB ROM / 0.5 kB RAM
- **SNMP / Netconf**: 8.7 kB ROM / 0.1 kB RAM
- **HTTP / CoAP**: 4.0 kB ROM / 0.2 kB RAM
- **...**

**Security (DTLS, TLS, etc.)**
- 3 kB ROM / 1.2 kB RAM

- **UDP**: 1.3 kB ROM / 0.2 kB RAM
- **TCP**: 4 kB ROM / 0.2 kB RAM
- **IPv6**: 11.5 kB ROM / 1.8 kB RAM
- **RPL**: 7.5 kB ROM / 0.01 kB RAM
## Directly Related Work at Jacobs University

### SNMP applicability to constrained devices
- Guidelines how to fit SNMP into constrained devices
- Tricks like making VACM a simple read-only/read-write switch
- [draft-hamid-6lowpan-snmp-optimizations-02.txt](draft-hamid-6lowpan-snmp-optimizations-02.txt)

### RPL MIB module specification and implementation
- Definition of a MIB module for the RPL routing protocol
- Implementation and evaluation on Econotags
- [draft-sehgal-roll-rpl-mib-01.txt](draft-sehgal-roll-rpl-mib-01.txt)

### DTLS for constrained devices
- Contiki-SNMP over DTLS (RFC 5590, RFC 5591, RFC 5953)
Other Related Work at Jacobs University

**NETCONF Lite implementation and specification**

- Profile (subset) of NETCONF 1.1 (RFC 6241)
  - Single session, hence trivial locking
  - No `<edit-config>`, no `<get>` / `<get-config>` filtering
  - No optional capabilities
  - No security (yet) . . .
- First prototype shown at the Prague IETF (on AVR Ravens)
- `<draft-schoenw-netconf-light-00.txt>`

**Multicast DNS for network management service discovery**

- Managers use mDNS to discover manageable devices
- Devices discover management services via mDNS
- Contiki-mDNS implementation already running
- `<draft-schoenw-opsawg-nm-srv-02.txt>`
Connecting Low-Power and Lossy Networks to the Internet.  

S. Kuryla and J. Schönwälder.  

SNMP Optimizations for Constrained Devices.  
Internet Draft <draft-hamid-6lowpan-snmp-optimizations-03.txt>, ETRI, Jacobs University, Ajou University, October 2010.

Internet-Draft (work in progress) <draft-schoenw-opsawg-nm-srv-01>, Jacobs University, November 2010.

V. Perelman, J. Schönwälder, and M. Ersue.  
Network Configuration Protocol for Constrained Devices (NETCONF Light).  
Internet-Draft (work in progress) <draft-schoenw-netconf-light-00>, Jacobs University, Nokia Siemens Networks, June 2011.