Management of 6LoWPAN Networks

Jürgen Schönwälder, Siarhei Kuryla

JACOBS UNIVERSITY

IETF 78, Maastricht, 2010-07-26

6LoWPAN = IPv6 over IEEE 802.15.4

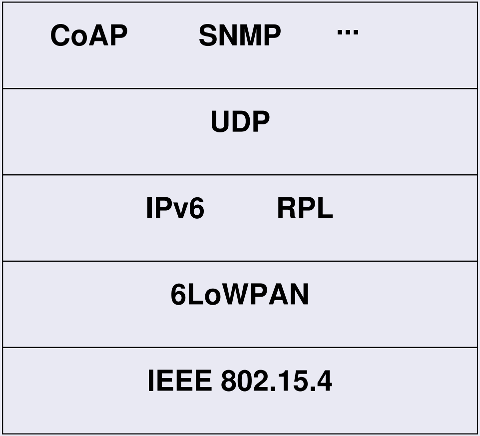
IEEE 802.15.4 Characteristics

- 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

6LoWPAN

- IPv6 over Low-Power Wireless Personal Area Networks
- RFCs 4919, 4944, several Internet-Drafts

Protocol Stack — The Big Picture



Why 6LoWPAN Management?

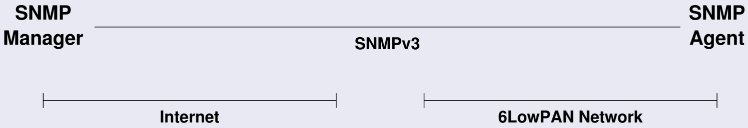
- "Autonomic devices won't need management — so don't waste your time on the wrong problem..."
- Well, no, for the foreseeable future, you will end up managing the autonomic system (one more control loop)
- Key management is for example a largely unsolved problem (but specialized keying protocols might help)

Example Management Questions:

- How much energy is left in my nodes/network?
- How many nodes disappeared during the last night/day?
- What is the temperature, pressure, (add your favorite sensor here) distribution within the network?
- What is wrong with my 6LoWPAN network?

SNMP and 6LoWPAN

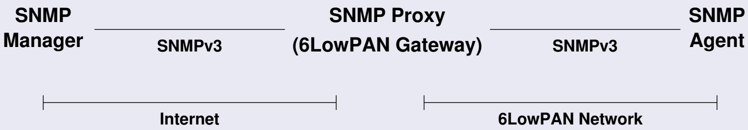
Using SNMPv3 End-to-End



- \+ Straight forward access to individual 6LoWPAN nodes
- \+ Reuse of existing deployed SNMP-based tools
- o 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

SNMP and 6LoWPAN

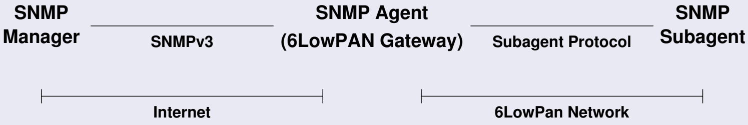
Using SNMPv3 Proxies



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

SNMP and 6LoWPAN

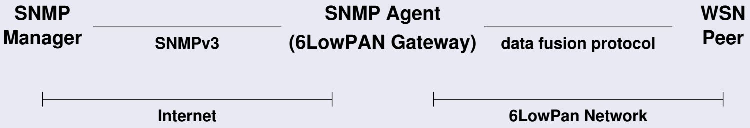
Using SNMPv3 Subagents



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

SNMP and 6LoWPAN

Using SNMPv3 with Data Fusion Protocols



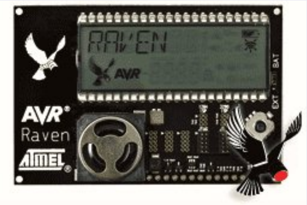
+ Indirect access to individual 6LoWPAN nodes
+ Leveraging data fusion protocols
+ SNMP agent acting as a cache, no expensive polling
o Reuse of existing SNMP-based tools supporting contexts
o Two security domains, different key management schemes
? No real advantage of 6LoWPAN technology — oops

SNMP over 6LoWPAN Prototype

Targeted Hardware

ATmega1284PV microcontroller:

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



Contiki-SNMP

- implemented in C programming language
- build on top of Contiki uIPv6 stack

Contiki-SNMP Overview

General Features

- SNMP messages up to 484-byte length
- `Get`, `GetNext` and `Set` operations
- SNMPv1 and SNMPv3 message processing models
- USM security model
- MIB API to define and configure accessible managed objects

USM Security

- HMAC-MD5-96 authentication protocol
- CFB128-AES-128 symmetric encryption protocol

Implemented MIB Modules and Static Memory Usage

MIB Modules

- `SNMPv2-MIB` – SNMP entity information
- `IF-MIB` – network interface information
- `ENTITY-SENSOR-MIB` – temperature sensor readings

SNMPv1 and SNMPv3 enabled

- 31220 bytes of ROM (around 24% of the available ROM)
- 235 bytes of statically allocated RAM

SNMPv1 enabled

- 8860 bytes of ROM (around 7% of the available ROM)
- 43 bytes of statically allocated RAM

Stack and Heap Usage

Maximum Observed Stack Usage

Version	Security mode	Max. stack size
SNMPv1	–	688 bytes
SNMPv3	noAuthNoPriv	708 bytes
SNMPv3	authNoPriv	1140 bytes
SNMPv3	authPriv	1144 bytes

Heap Usage

- not more than 910 bytes for storing an SNMPv1 message
- 16 bytes for a managed object in the MIB
- if a managed object is of a string-based type heap is used to store its value

Response Latency

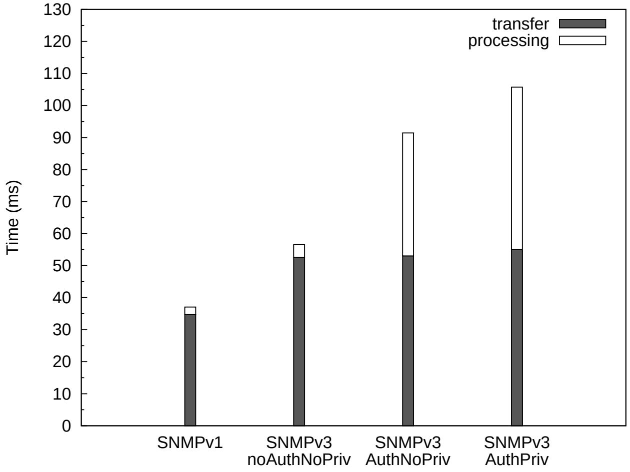


Figure: Time taken for transferring and processing an SNMP request.

Response Latency

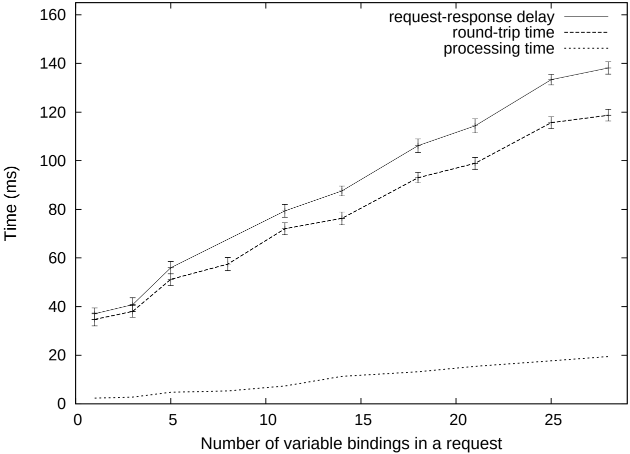


Figure: Time spent in transferring and processing SNMPv1 requests and responses as a function of the number of variable bindings in a request.

Protocol Stack — The MIB Picture

CoAP	**SNMP** **MIB**	**...**
	UDP **MIB**	
IPv6 **MIB**		**RPL** **MIB**
	6LoWPAN **???**	
	IEEE 802.15.4 **MIB / ???**	

SNMP Applicability to 6LoWPAN Networks

draft-hamid-6lowpan-snmp-optimizations-02.txt

- Implementing SNMP in resource constrained networks
  - How to implement minimal VACM and USM?
  - Impact of different security mechanisms
  - ...
- Deploying SNMP in very resource constrained networks
  - Naming and addressing issues
  - Timeouts and retransmissions
  - Polling intervalls and caching issues
  - Usage of the protocol operations
  - ...
- Applicable MIB objects/modules
  - Which MIB objects are essential to support?
  - Which MIB objects are recommended to support?
  - Which MIB objects are optional?
  - ...