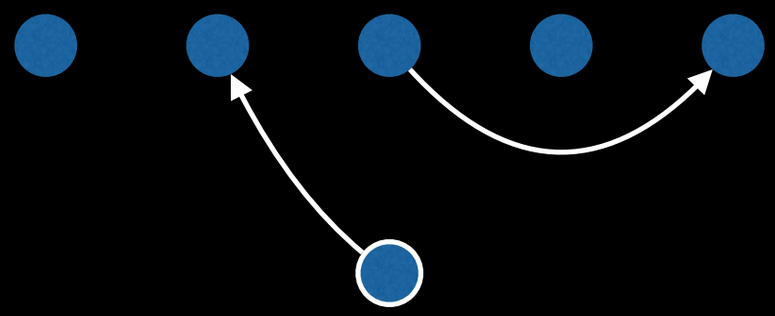
Bachelor PO - RIOT im Internet of Things

Sebastian Meiling
iNET RG, HAW Hamburg
sebastian.meiling@haw-hamburg.de

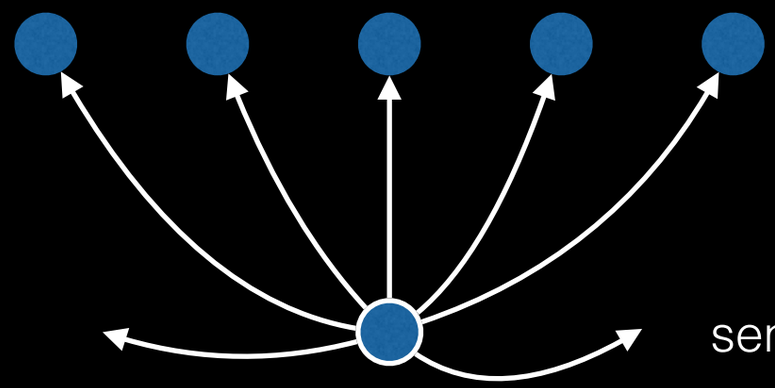
Patterns

- Unicast 1:1



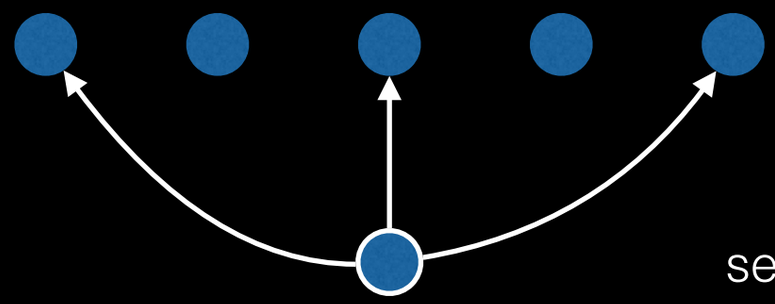
client-server like

- Broadcast 1:*



send to broadcast address

- Multicast 1:n



send to multicast address

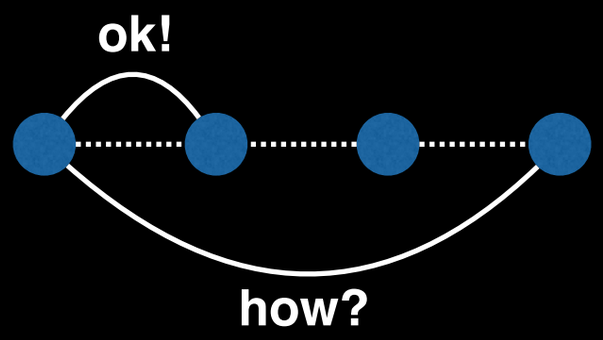
RIOT

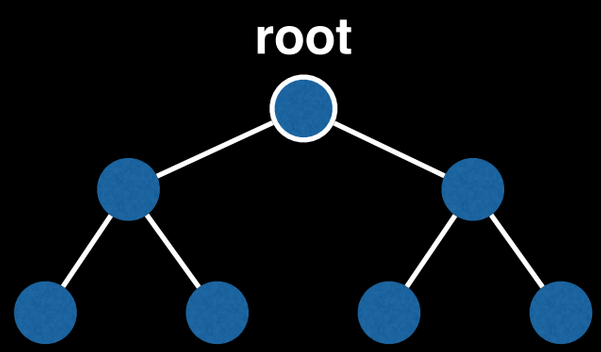
Signalling

- Polling
  - (periodic) request data from sensor node
  - 1 request [+ 1 ACK] + 1 response/data [+ 1 ACK]
- Timer
  - periodically send sensor data to server/gateway
  - 1 data message [+ 1 ACK]
- Event
  - send sensor data triggered by event, e.g., threshold
  - 1 data message [+ 1 ACK]

IoT Networking

- typical characteristics:
  - wireless communication using low power, lossy radios
  - nodes may sleep, thus don't (want to) receive at all times
  - (likely) multiple hops between sender and receiver
- multi hop communication requires routing protocol





- RIOT uses RPL
  - designed for 1:n and m:1 communication,
  - 1:1 unicast also possible, but less efficient

RIOT

RESTful API

- uses standardised HTTP methods:
  - GET retrieve data item, 1 GET-Request + 1 Response [+ 2 ACKs]
  - PUT update data item, 1 PUT-Message + 1 Response [+ 2 ACKs]
  - POST create data item, 1 POST+ 1 Response (new ID) [+ 2 ACKs]
- resources are encoded and accessed via URLs:

```
https://en.wikipedia.org/wiki/Wireless_sensor_network
schema  <- host = IP ->  <----        PATH        ---->
send [GET /wiki/Wireless_sensor_network] to en.wikipedia.org
```

- example usages:
  - GET /temperature or GET /temperature/node01/
  - PUT /temperatures/node01/2015-10-16_08-55-10
  - POST /temperatures/node01/

RIOT

CoAP

- Constrained Application Protocol, RFC 7252
- lightweight HTTP equivalent for the IoT
- wide variety of payload types (like MIME)
- uses UDP transport, unlike HTTP+TCP
- optional ACK-like mechanism and retries
- libraries for C/C++, Java, Python, etc...
- see: http://coap.technology

RIOT

CoAP in RIOT

- native CoAP support by *gcoap*
- or third party libraries:
  - *libcoap*: (nearly) feature complete, standard conform
  - *microcoap*: small, simple, but server side only
- we recommend *gcoap*:
  - lightweight and simple, based on nanocoap
  - supports client and server side
  - caveat: under development, API changes possible
- See example: https://github.com/RIOT-OS/RIOT/tree/master/examples/gcoap

CoAP Demo

- RIOT as CoAP client PUT sensor data
- Raspberry Pi as protocol gateway
  - L1+L2: IEEE-802.15.4 to Ethernet
  - L3: 6LowPAN to IPv6
  - L4: UDP to TCP
  - L5: CoAP to HTTP

www.riot-os.org