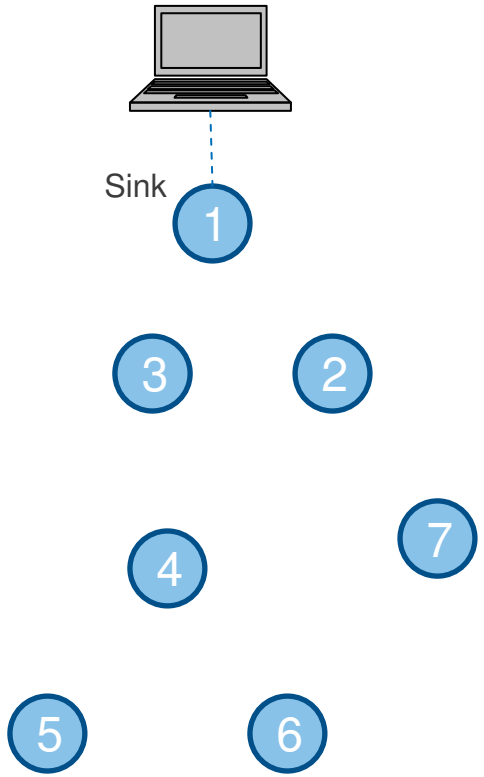
A remote reconfiguration mechanism for WiseNET wireless sensor networks

Damien Piguet, Philippe Dallemagne, Jean-Dominique Decotignie

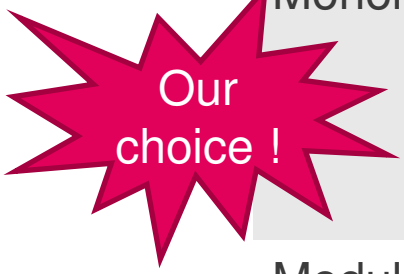
ReTrust'08, Trento, 16.10.2008

Introduction

- Because it is hard to access them physically, remote update/upgrade mechanisms are very useful for wireless sensor networks.
- Constraints: low-power, low bandwidth, scarce computing resources (CPU, memory).
- Objectives: compact update, reduced memory and CPU requirements, reliability.
- Assumptions for the first version: single sink, homogenous wireless sensor network. No mobility.



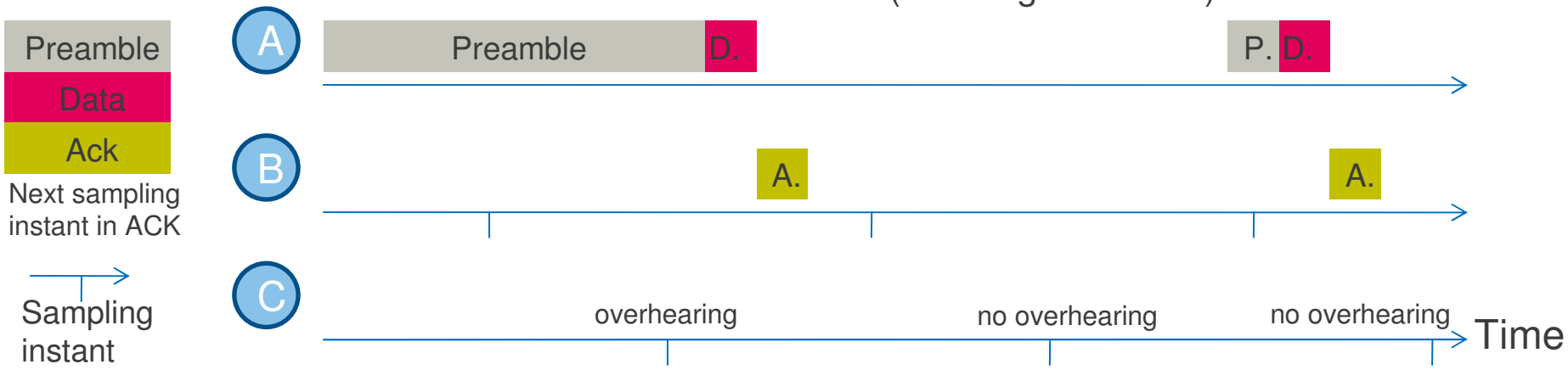
Reconfiguration schemes classification



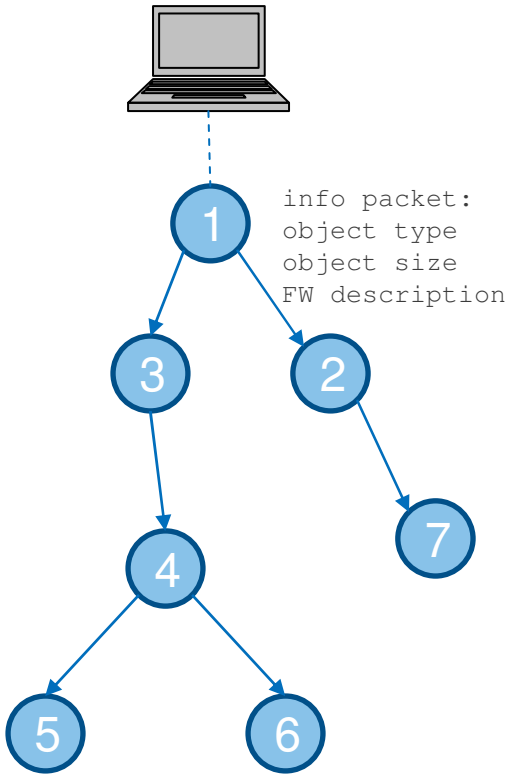
Scheme	Description	+	-
Monolithic	All parts of the program running on a node are statically linked together at compile time. The whole program is loaded as one single block. Example: **TinyOS**.	• No overhead on execution environment. • Flexibility: any part of the code (including low-level) can be modified.	• Large updates. • Node inoperative during code replacement.
Modular	Supports individual modules that can be installed/removed on top of a kernel. Example: **Contiki**, **SOS**.	• Only a module is replaced => smaller update. • Unaffected parts of the program can run.	• Small execution overhead. • The kernel of the system has to be reconfigured using a monolithic technique.
Virtual machine	An interpreter runs a script over an OS. The virtual machine interprets high-level instruction which make sense to the sensor network application. Example: **Maté**.	• Smallest upgrade size: only a script is re-installed.	• Low flexibility. • Higher execution environment overhead.

Dissemination protocol principles

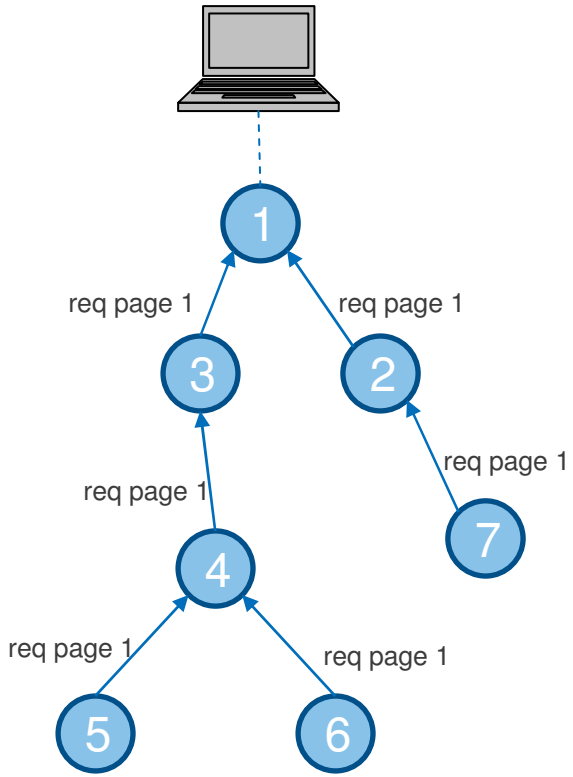
- New program is fragmented into pages of about 512 bytes.
- Nodes download pages one by one, by pulling them from a neighbor (the parent).
- After downloading a page, a node forwards it before requesting the next one to its parent.
- Unicast is used to transmit a page between two neighboring nodes.
- WiseNET unicast service is efficient because it is reliable (ACK / CRC).
- WiseMAC, the underlying MAC protocol is asynchronous and shows its best behavior in terms of low energy consumption in unicast communications (see diagram below).



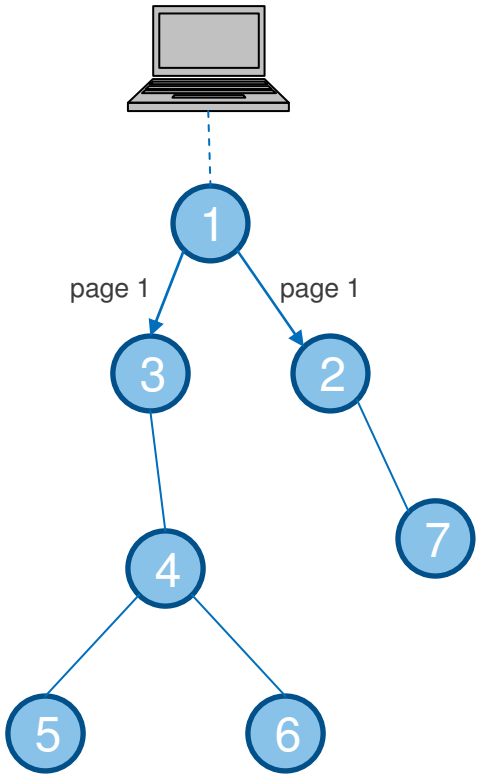
Dissemination protocol (1/4)



Sink floods information packet which reaches the whole network.

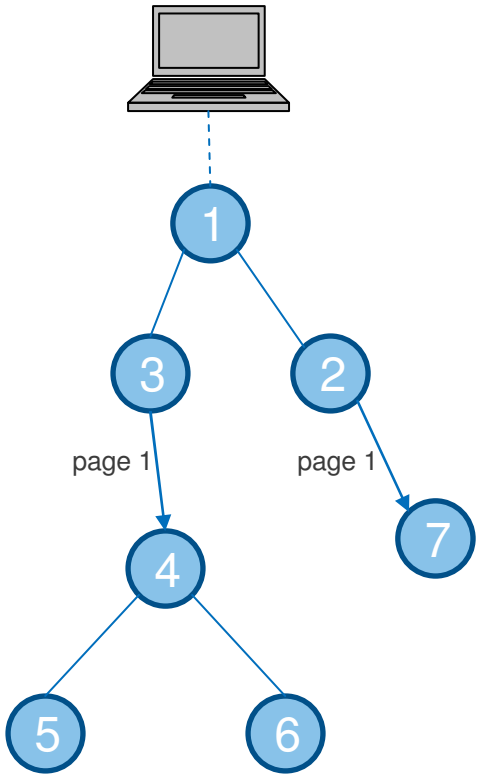


Each node requests the first page to its parent.

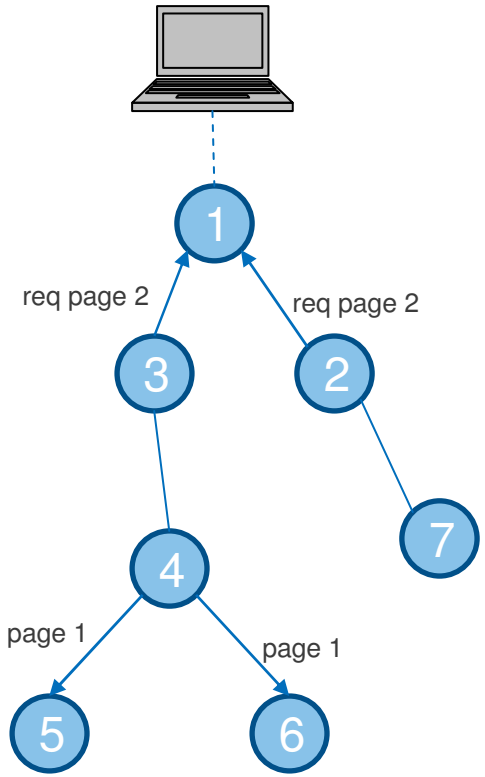


After some time, the sink decides it has two children and sends them the page.

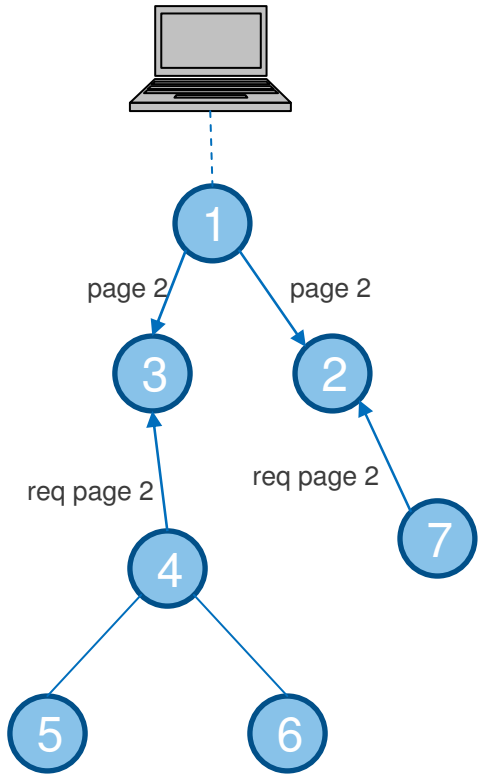
Dissemination protocol (2/4)



Nodes 2 and 3 forward the page to their children.

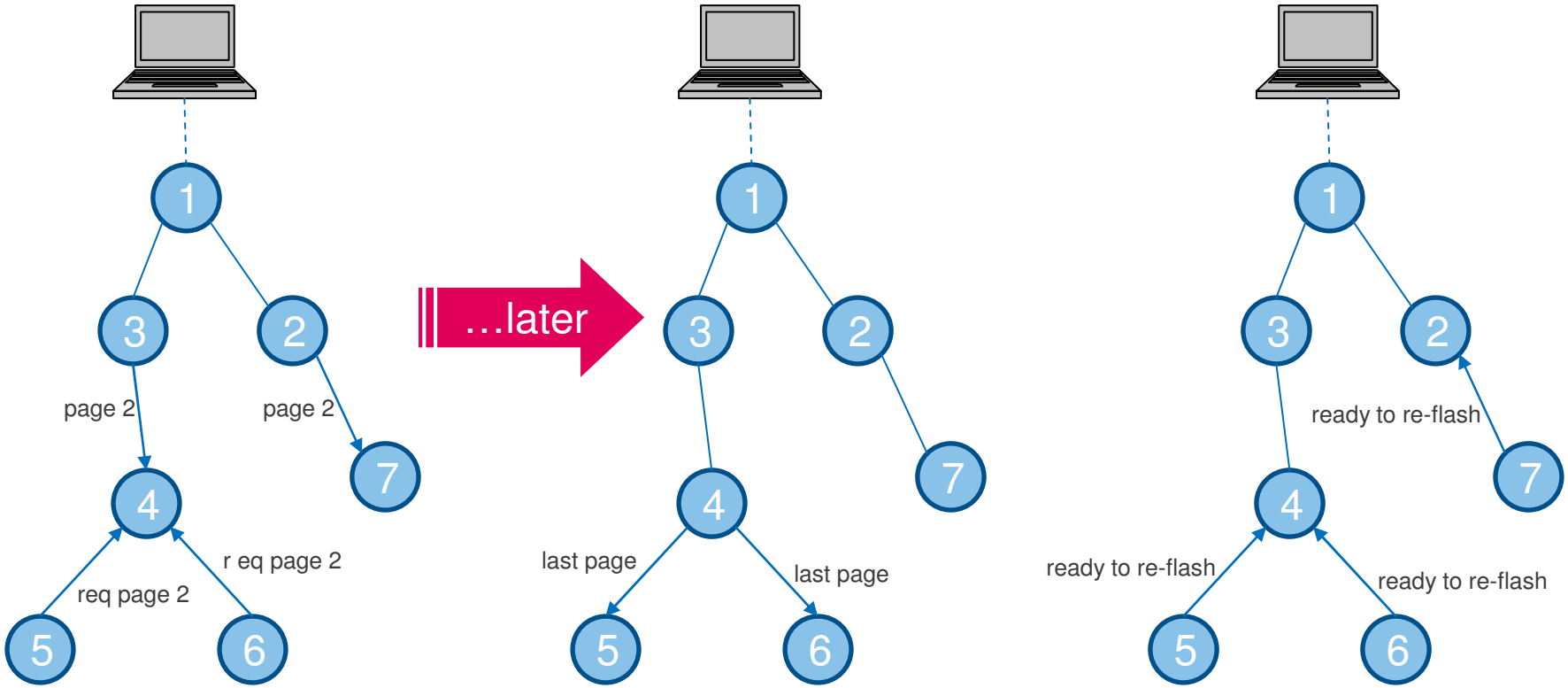


Node 4 forwards the first page. 3 and 2 request the second one.



Its job done, 4 can request page 2. 7 finds out it has no child and does the same while 2 and 3 receive the 2nd page.

Dissemination protocol (3/4)

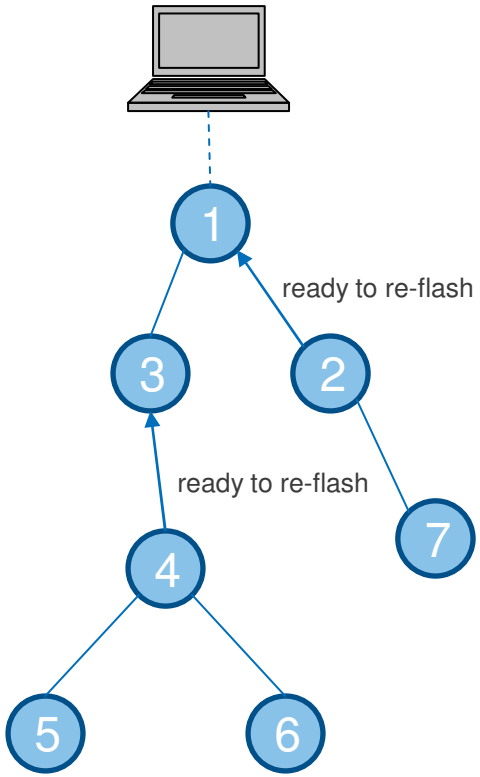


Nodes 2 and 3 forward the 2nd page to their children. 5 and 6 request it.

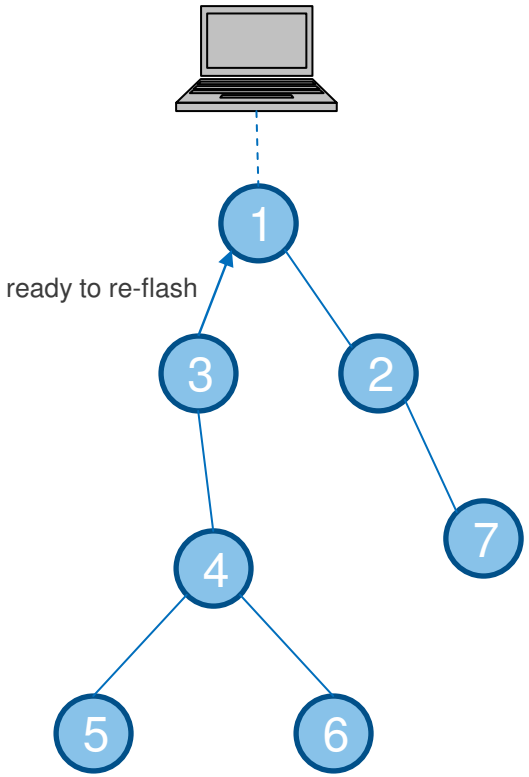
At the end, 5 and 6 get the last page.

Having no children, nodes 5, 6 and 7 are the first to signal “ready to re-flash”.

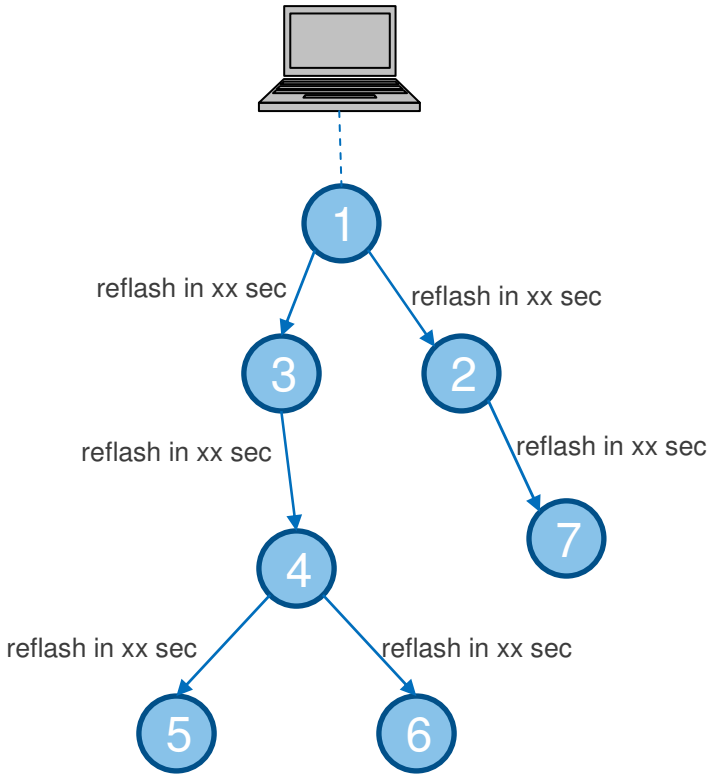
Dissemination protocol (4/4)



Nodes 2 and 4 got notified by all their children. Then they can signal “ready to re-flash” to their parent.



Same thing for node 3.

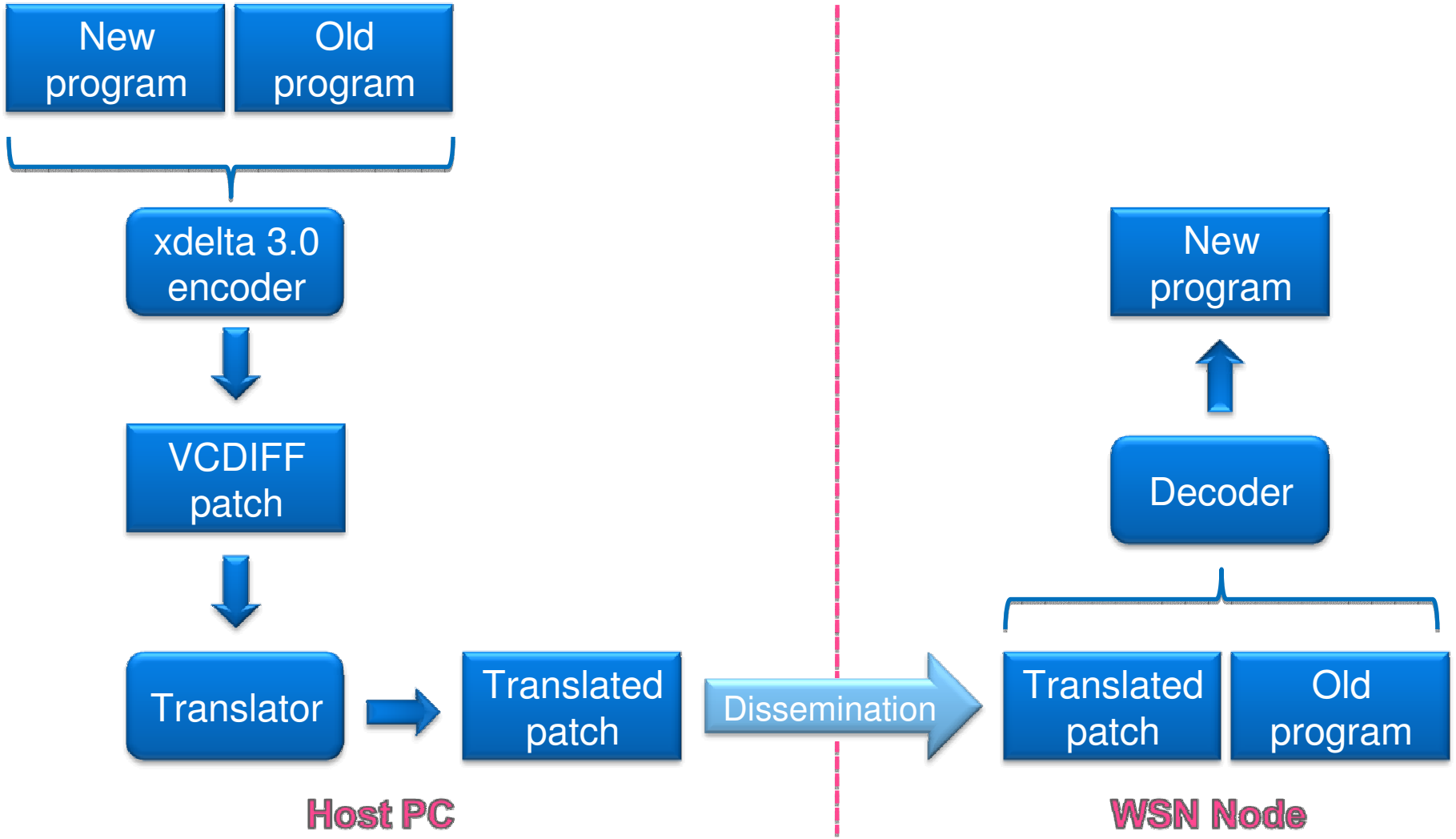


Once notified by all his children, the sink floods the re-flash command two times.

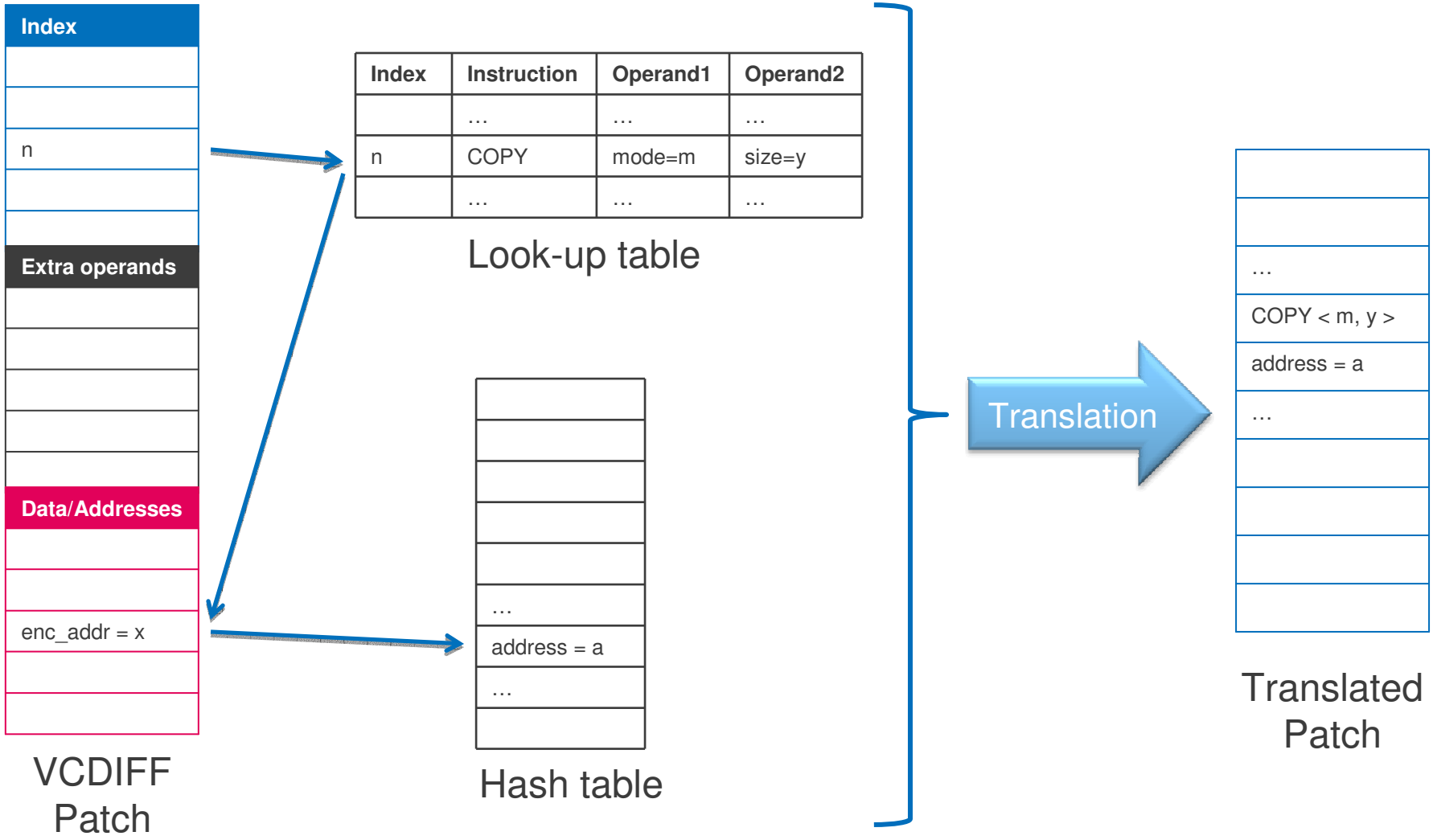
Patch technique

- **Problem:** reduce the size of the monolithic block that is disseminated so that power consumption and latency decrease accordingly.
- **Solution:** disseminate a “patch”, i.e. a binary script which encodes the difference between the current running program and the new one. A decoder on each node follows the patch instructions to reconstruct the new program from the older one.
- **Constraint:** low resource requirements (memory, CPU) on the patch decoder.
- Choice: Use VCDIFF (RFC 3284) format. Encoder: xdelta 3.0 (open source). VCDIFF is designed for supporting the back-up of large files between powerful machines over a low-bandwidth link.
- Compression rate obtained is high, but the decoder requires too many resources to be implemented “as is” on a WSN node. Therefore the VCDIFF patch is translated to our own suboptimal, easier to decode format.

Patch technique: workflow



Patch translation principle



Patch technique results

- Old program size: **42’962 bytes**.
- 3 typical program modifications:
  1. Very small (single byte).
  2. Rather small (bug fix).
  3. Large (new feature, new application).

- xdelta 3.0 is efficient when used to compress our programs.
- The cost, in terms of patch size, of the translation to a suboptimal, easily decodable format is low.

Scenario	New program size	VCDIFF size	VCDIFF compression %	WiseNET patch size	WiseNET compression %
Single byte	42’962	69	99.8	29	99.9
Bug fix	42’984	3409	92.0	3383	92.1
New application	46’670	14319	69.3	14479	69.0

Reliability

- The information packet which starts the protocol is analysed by each node that receives it. Its content is checked against a range of acceptable values. The node does not continue the download in case of failure.
- A 16-bits checksum is verified on each received page. The same method is used to check the recording to the external flash.
- If a node dies during the execution of the dissemination protocol, the other nodes continue the procedure.
- A node sends the “re-flash ready” signal to its parent only when all of its children have done so. Thus when the sink gets the signal, it means that every alive node in the network has terminated the download procedure and will accept the re-flash command. If one node fails, the whole network falls back by timeout.
- An Adler-32 checksum verification is performed on the whole new program image after reconstruction by the patch decoder. A node will send the re-flash ready signal only if the check passed.

Conclusion

- The combination of the dissemination protocol and of the patch technique form the reconfiguration system for WiseNET. In program memory, the protocol takes about 6 kB and the patch decoder about 2 kB.
- The dissemination protocol is efficient and reliable.
- The size of the disseminated program is reduced. Since the execution environment is monolithic, this is achieved thanks to a patch technique.
- The patch technique achieves compression rate of more than 90% for a small change and of almost 70% in presence of a major upgrade.
- Future work: use filters to allow heterogeneous networks. Investigate a modular solution (if it make sense for our business model). Tackle the security issue.
- The solution has been developed and financed in the context of the EU project “WASP”.

Thank you for your attention!

Color scheme

CSEM-Blue
RGB 0-112-188

Light Blue
RGB 136-193-233

Dark Grey
RGB 90-87-70

Red
RGB 225-0-90

Light Grey
RGB 196-196-186

Olive Green
RGB 193-191-0