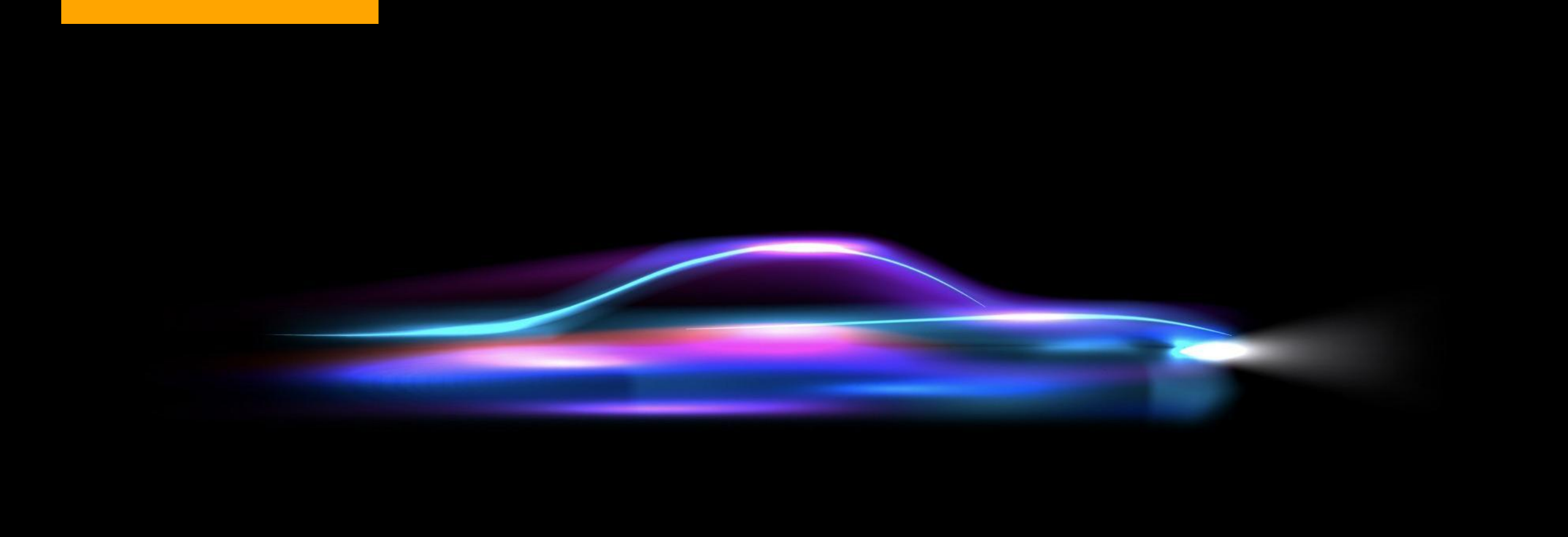


eCAL Registration / Monitoring Layer

A deep dive

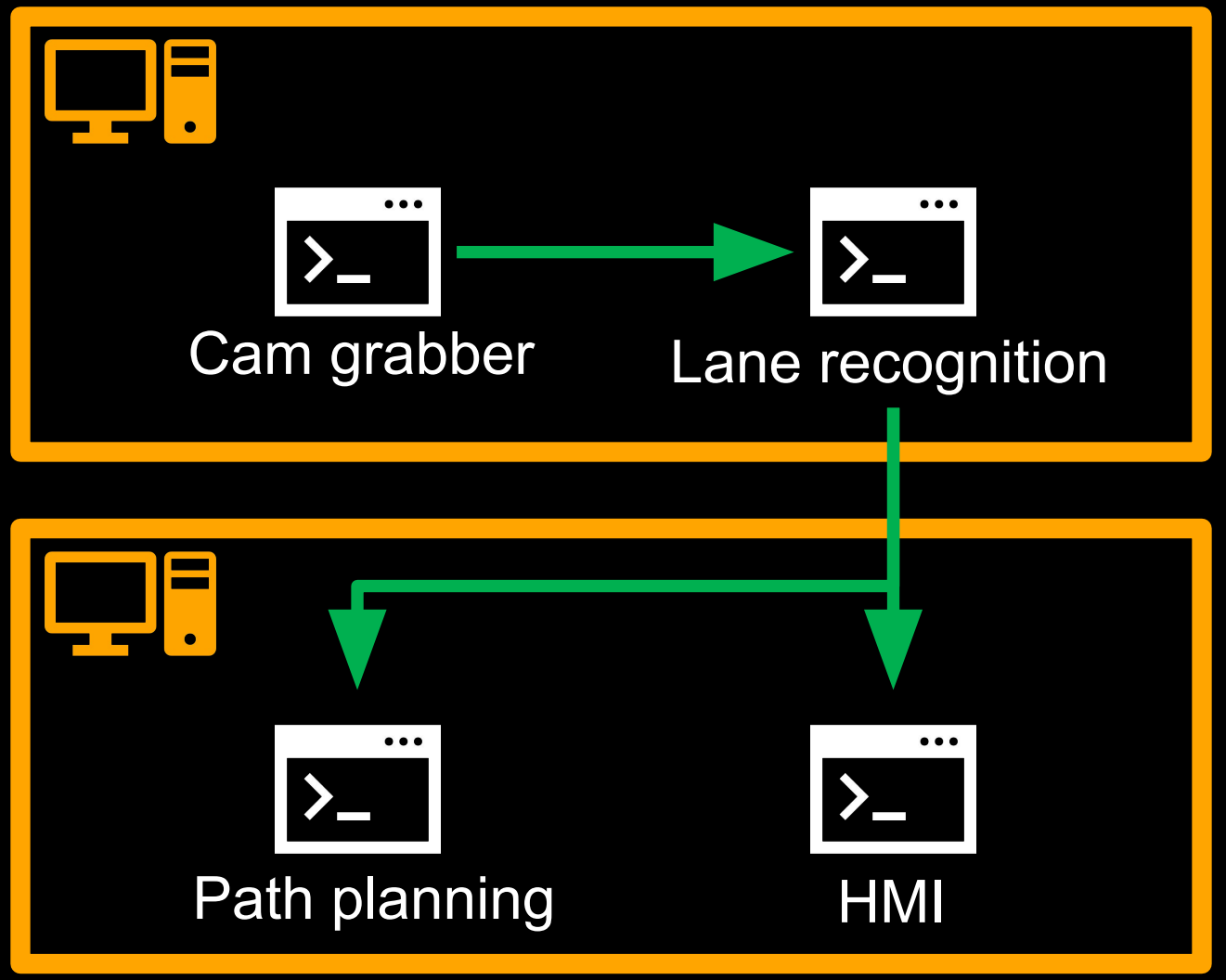
http://ecal.io

About us


Kristof Hannemann
Florian Reimold
TGX 18.560
Automated Driving
Continental

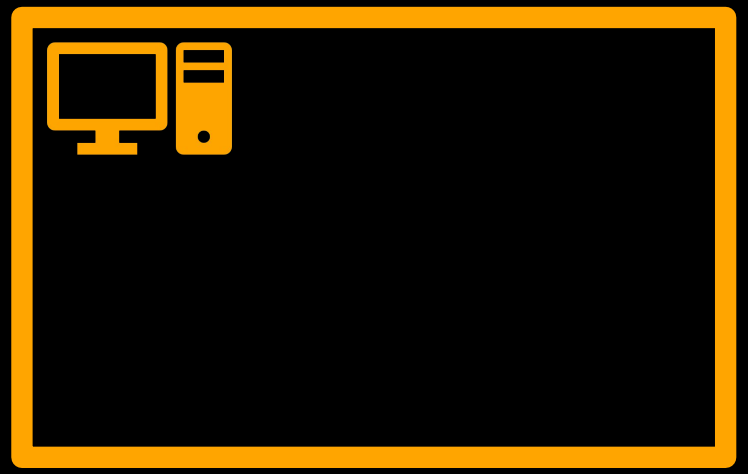
Publish / Subscribe

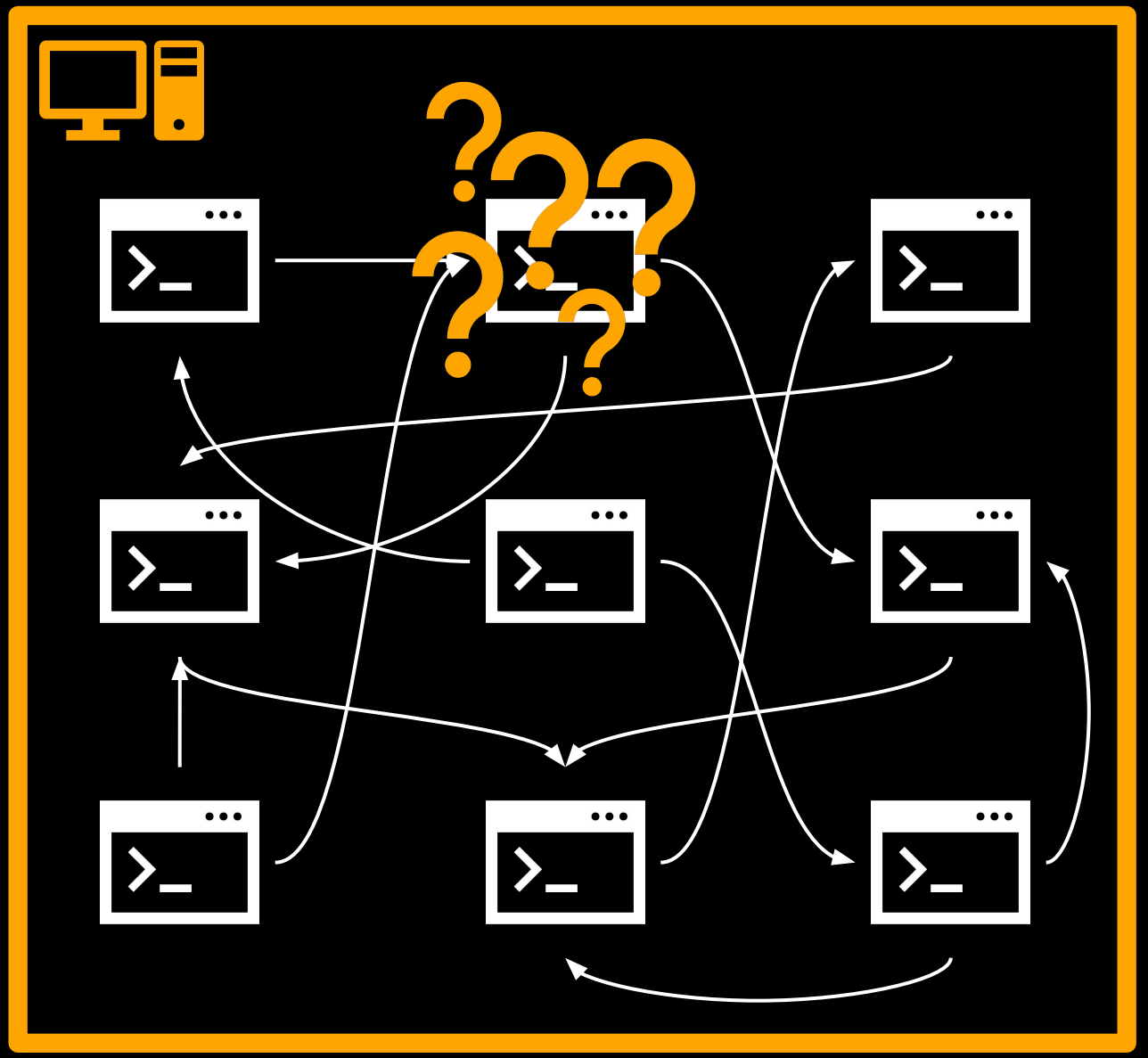
- Data is transferred through the network
- How do nodes know where to send the data to?
- New technique in eCAL 5.11



Idea 1: Manually

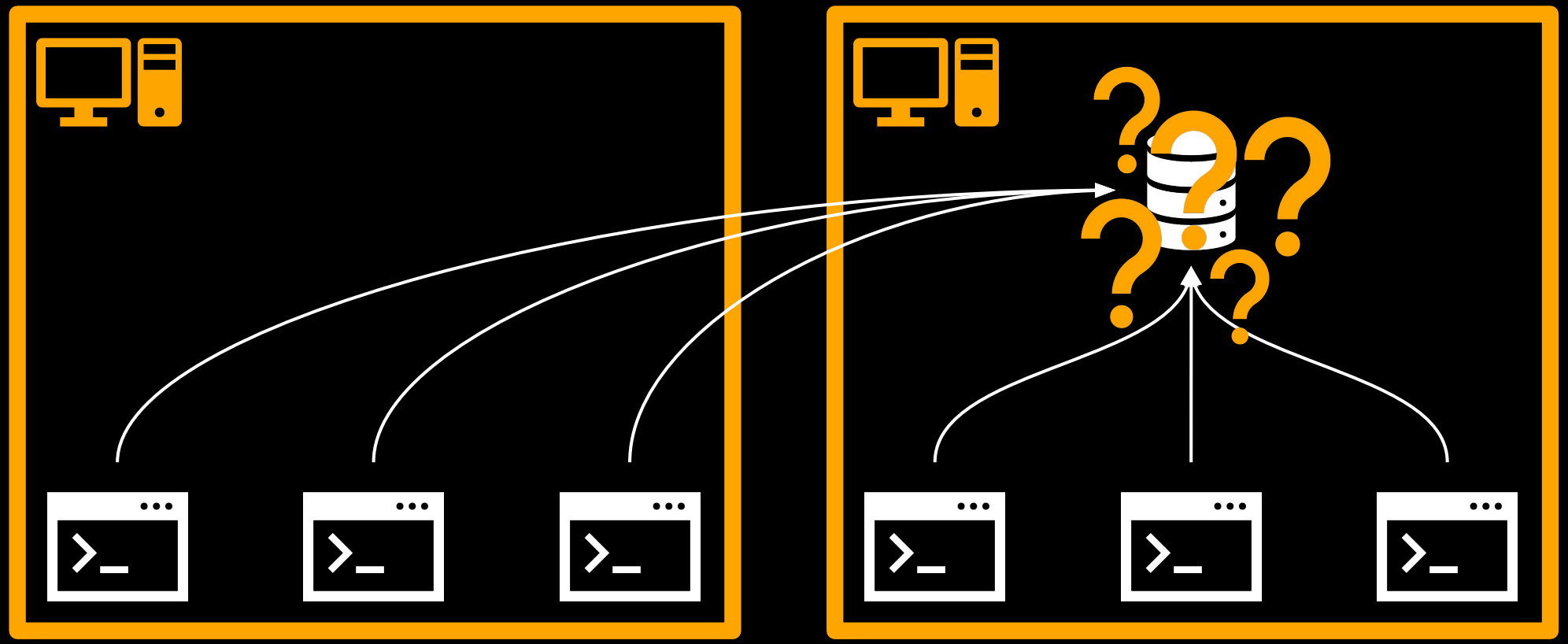
- Manual direct connections, e.g.:
  - IP + Port
  - SHM Filename
- Configuration Nightmare





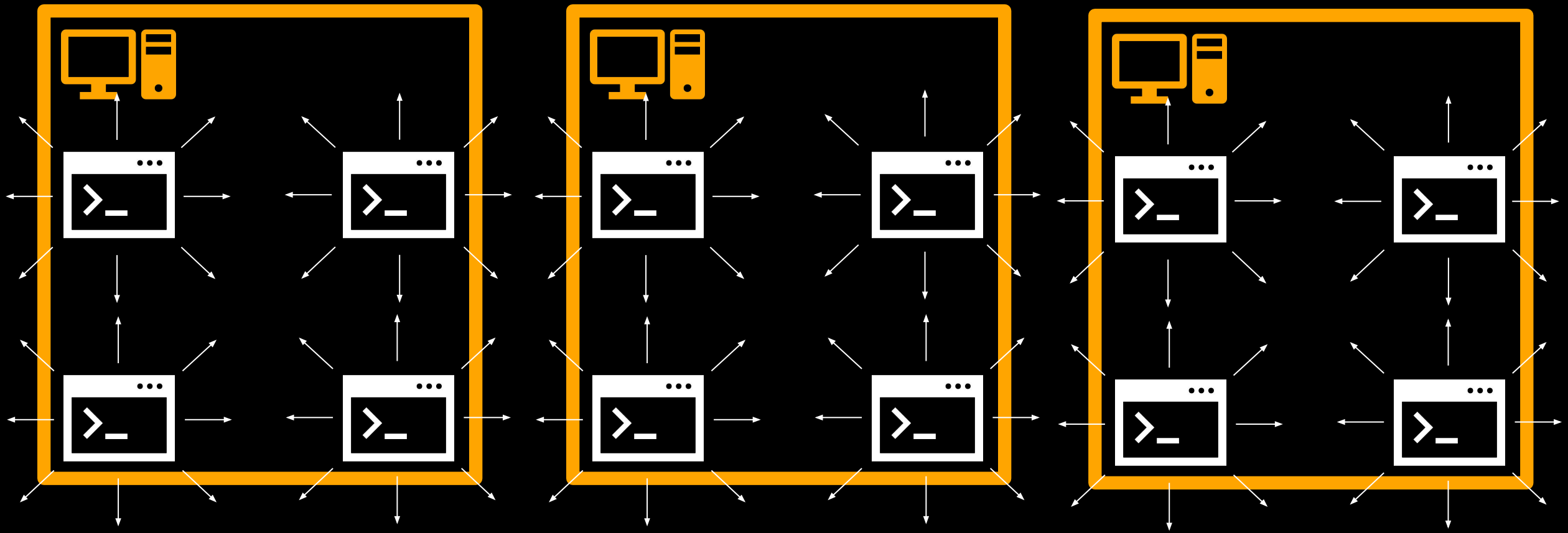
Idea 2: (Central) Broker

› Broker can manage connections



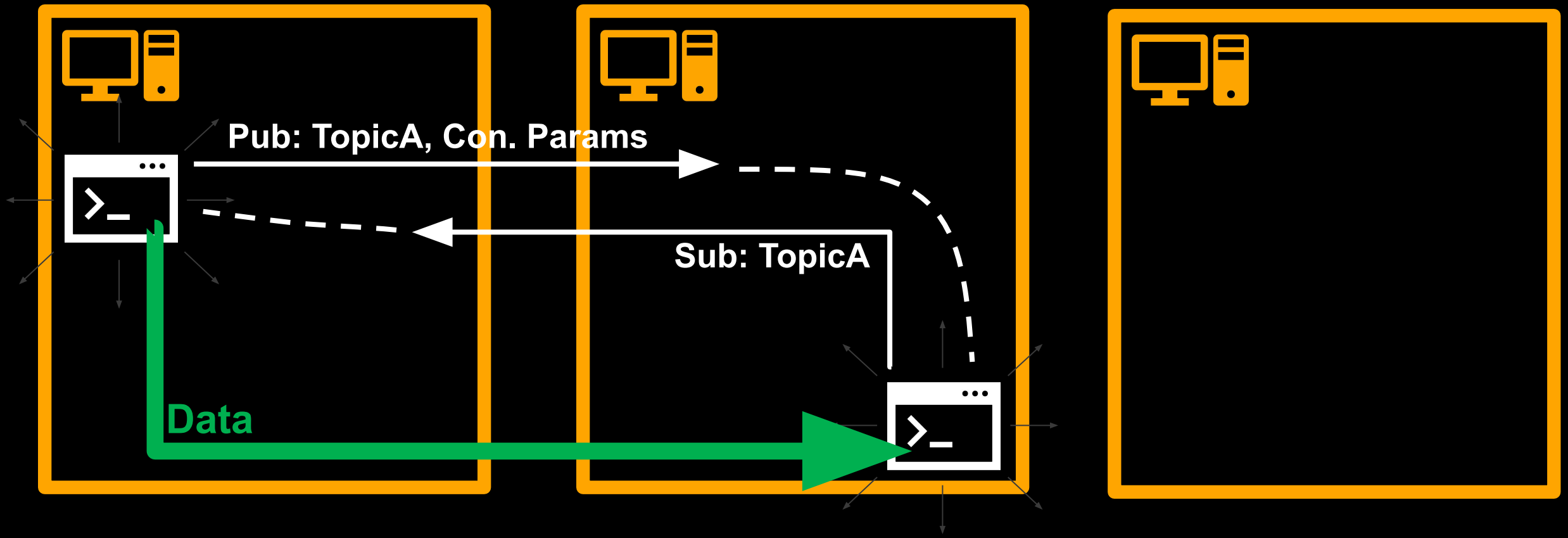
Our Solution: Brokerless Zeroconf via UDP

› Use UDP Multicast to connect Nodes automatically to each other



Our Solution: Brokerless Zeroconf via UDP

› Data layer is initialized based on registration layer



Consolidation back to 1 PC

- Decreasing number of PCs, Increasing number of Nodes
- => UDP becomes a bottleneck

Introducing shared memory-based registration in eCAL

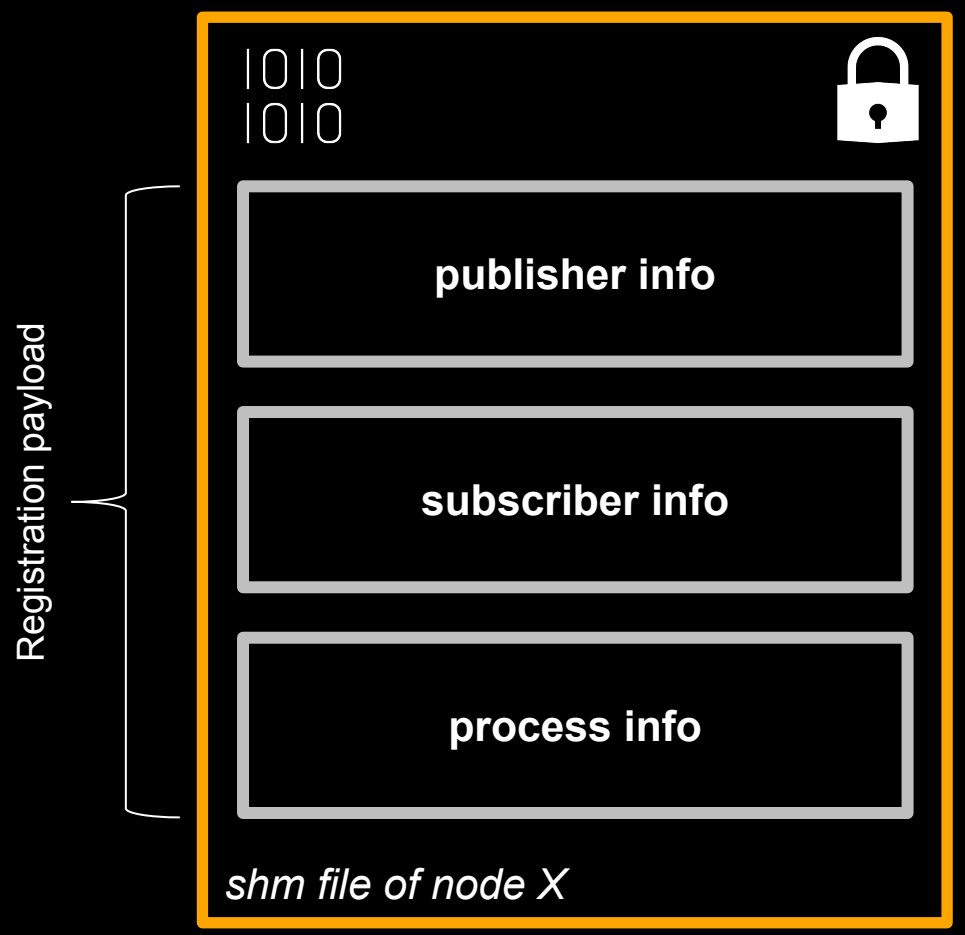
Requirements of local broadcast concept

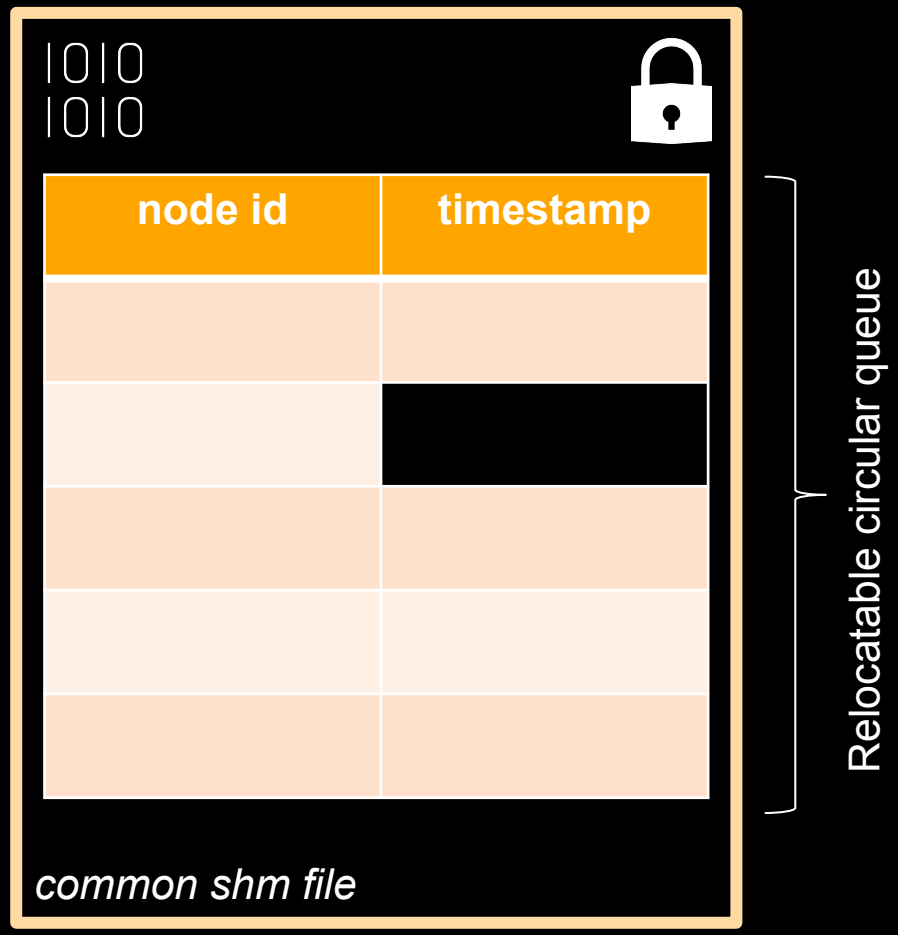
Simple and flexible design of a local brokerless broadcast concept

Designed for high-data bandwidths on various operating systems

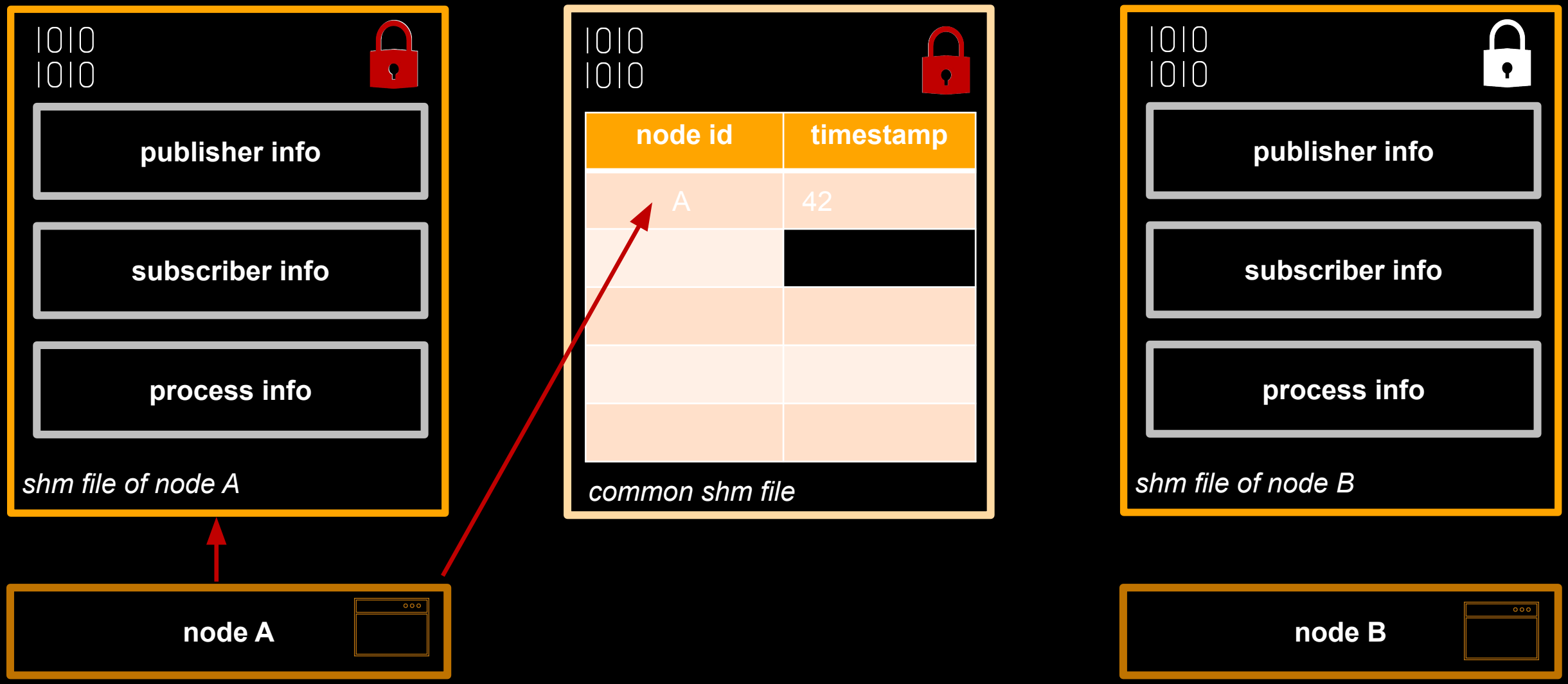
Automatic recovery in case of an error (no SPOF)

Idea: Broadcasting based on shared memory

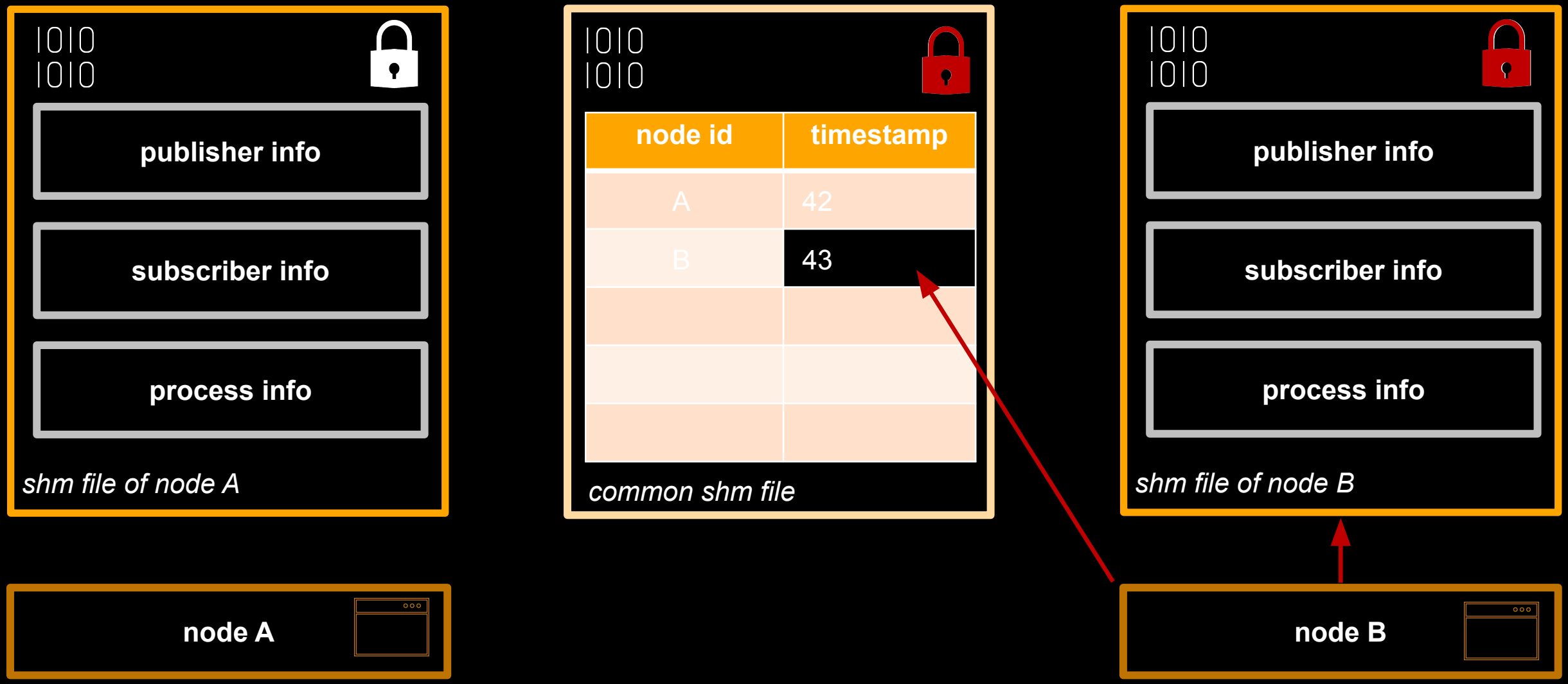




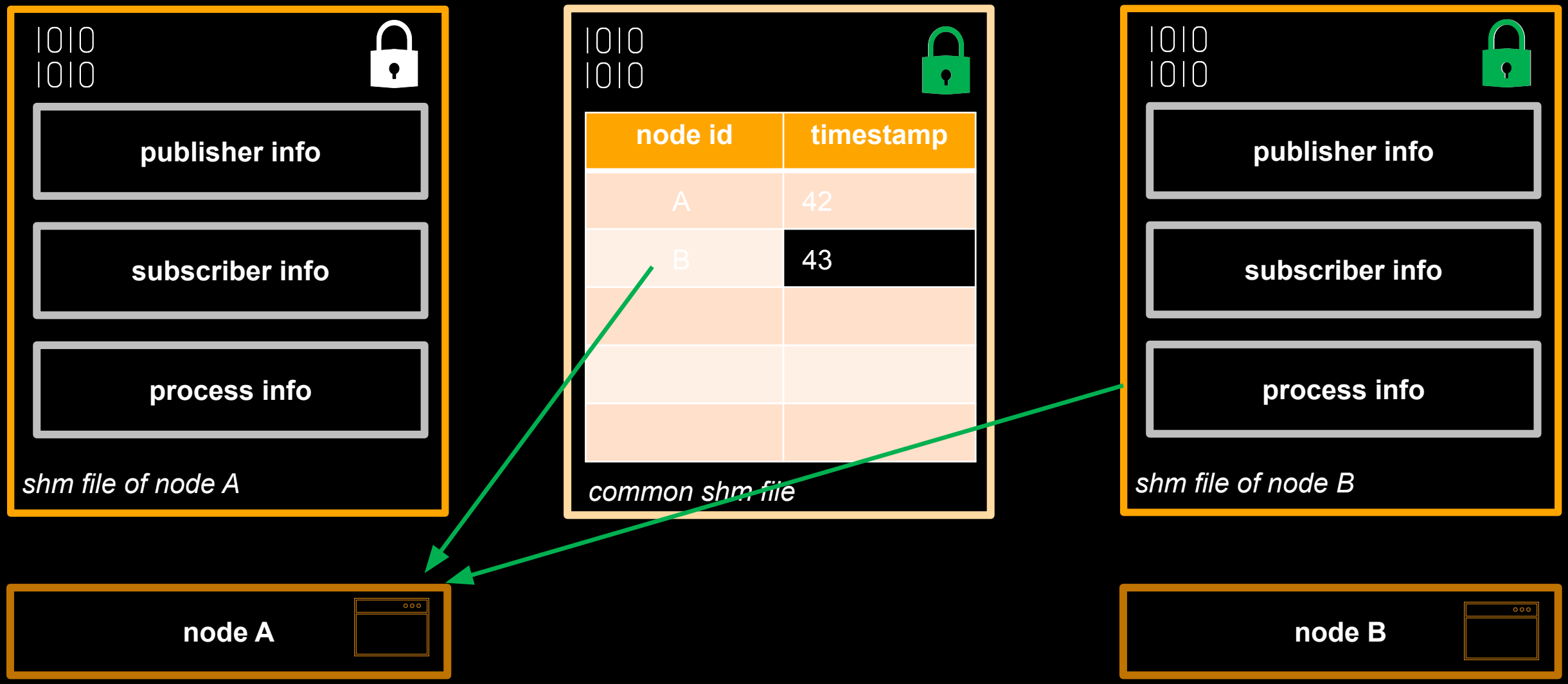
Workflow: Sending broadcasts to shared memory



Workflow: Sending broadcasts to shared memory



Workflow: Receiving broadcasts from shared memory



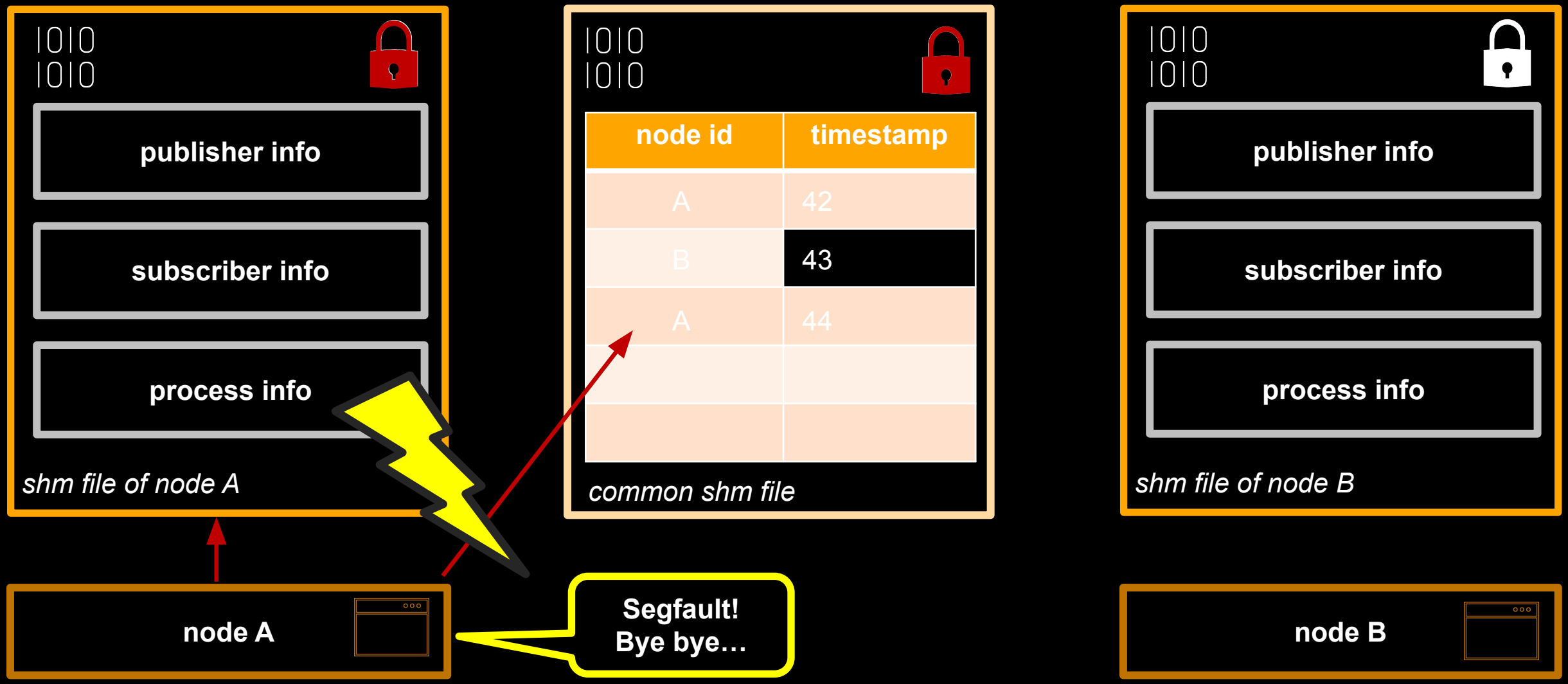
Requirements of local broadcast concept

Simple and flexible design of a local brokerless broadcast concept ✓

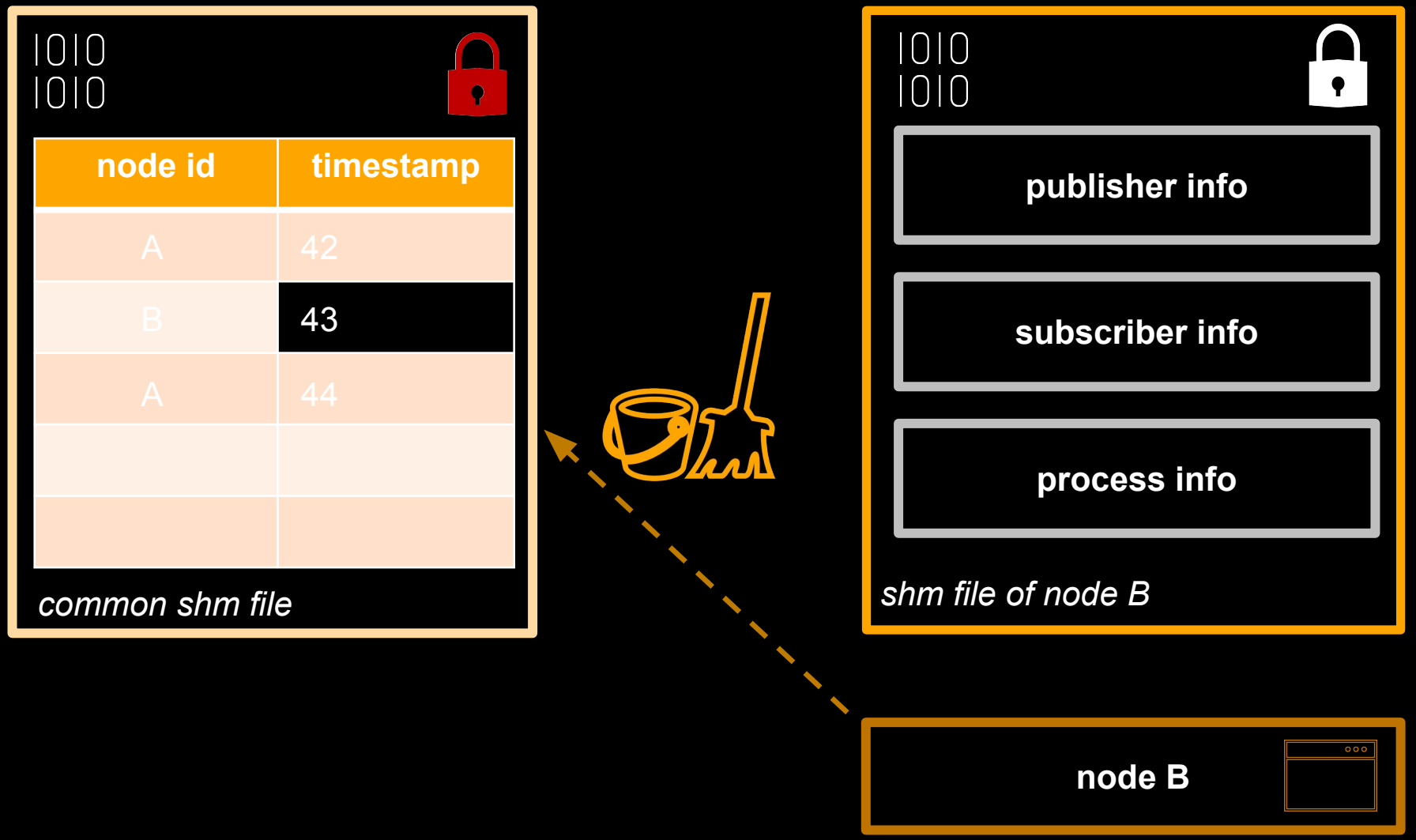
Designed for high-data bandwidths on various operating systems ✓

Automatic recovery in case of an error (no SPOF) ?

Potential SPOF of local broadcast concept



Automatic failure recovery



Requirements of local broadcast concept

Simple and flexible design of a local brokerless broadcast concept ✓

Designed for high-data bandwidths on various operating systems ✓

Automatic recovery in case of an error (no SPOF) ✓

State-of-the-art middleware

“eCAL is first pub-sub middleware that can run brokerless, zeroconf and without network stack”

How to enable the shared memory-based registration layer?

Enabling shared memory-based registration

→ Fetch 5.11.x release from https://github.com/eclipse-ecal/ecal or PPA

→ Navigate to the [experimental] section of ecal.ini config file

→ Set the option shm_monitoring_enabled to True

Thank you
for your attention!

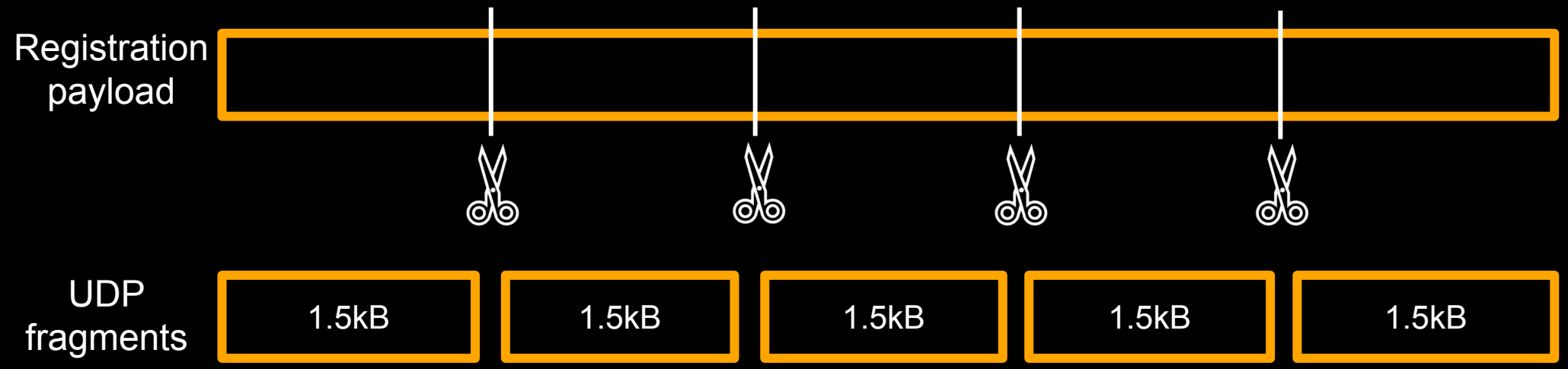
Root cause analysis: UDP bottleneck

Use-case:

Single host setups with large registration payloads

Weak spot:

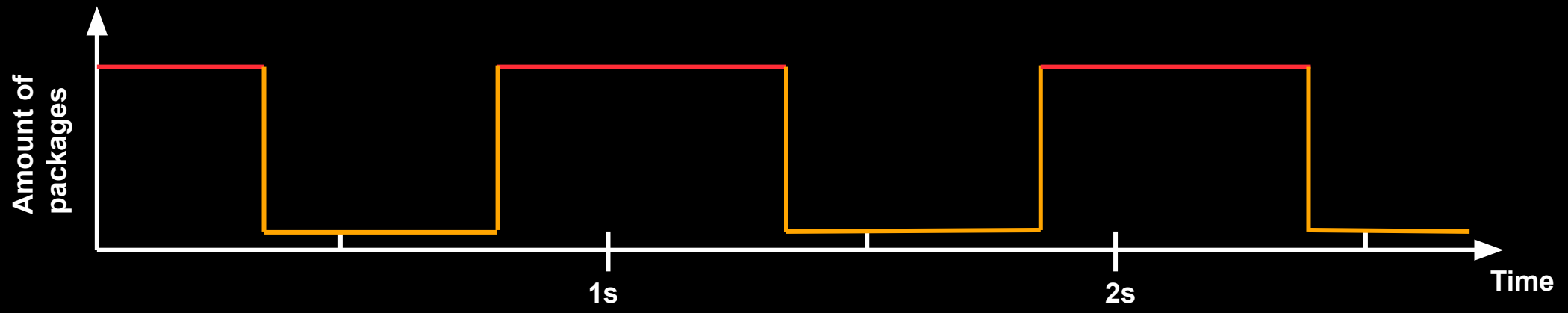
UDP package fragmentation due to network MTU



Root cause analysis: UDP bottleneck

Issue:

Cyclic registration update leads to recurrent UDP package bursts



→ Package bursts generates excessive CPU utilization

→ CPU peaks can cause package drops

→ **Missing UDP fragments prevent reassembling of registration payload**