Elastic and Secure Energy Forecasting in Cloud Environments

***André Martin*[*], *Andrey Brito*[#] *and Christof Fetzer*[*]**

andre.martin@tu-dresden.de, andrey@dsc.ufcg.edu.br, christof.fetzer@tu-dresden.de

[*]SE Group - Technische Universität Dresden - Dresden, Germany

[#]LSD Lab - Universidade Federal de Campina Grande - Campina Grande, Brazil

STREAM 2016 @ March, 23rd 2016, Tyson, VA

Application Example **SmartGrid**

ACM **DEBS'14** Challenge: SmartMeter recordings

- **Query #1:** Provide ***load predication*** (two times slices ahead) based on complete set of historical collected measurements
- **Query #2:** ***Detect outliers*** based on (global) median value of a 24hrs sliding time window

Challenges when Processing of SmartMeter data

1. Data growth
   - Q1: Accumulating historic data (to improve forecasts)
   - Q2: Temporary large states due to (24hr) sliding window
   - **Solution:** Elastic stream processing & ***cloud computing***

2. Privacy concerns **cloud computing**
   - Processing of privacy sensitive data (SmartPlugs)

State of The Art Open Source Technologies
Elasticity & Privacy

State support

Feature	Seep Imperial College	STORM	samza
State support/pers	Yes	User	KV store
Exactly Once Sematic	User	Transactional proc.	Yes

Challenge #1: **Elasticity**

Scale Out (expand)	Yes	Partially (no migr)	(Yes) *
Scale In (contract)	No	No (killing proc.)	(Yes) *

*at least once

Challenge #2: **Privacy Preservation**

Channel	No	Partially (netty.io)	No
Processing	No	No	No

Our Approach to Elasticity

- Stateful stream processing using ***StreamMine3G***
  - Operator migration protocol [1] provides:
  - **Exactly once** processing semantics
  - is based on **active replication**

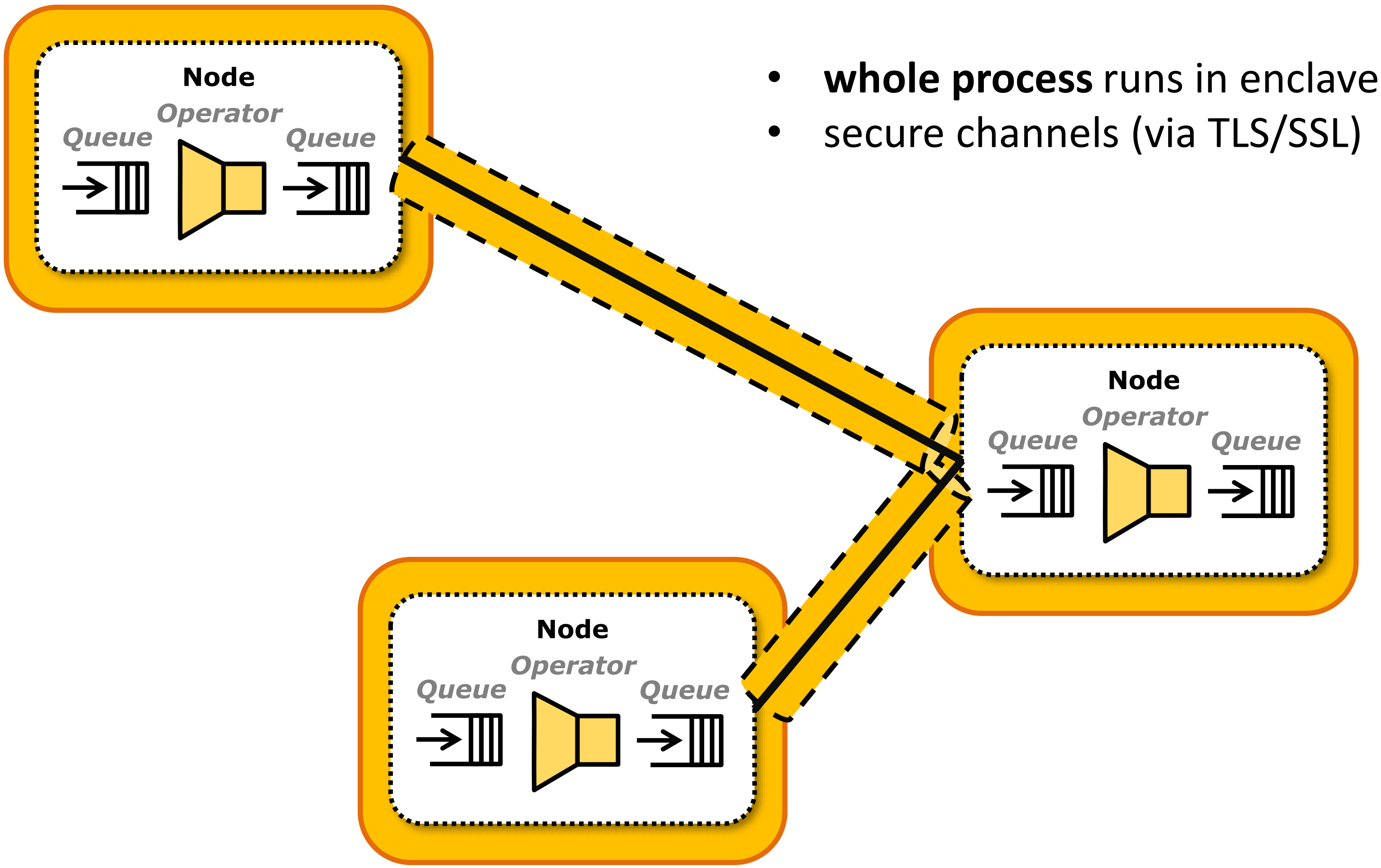
[1] **Elastic Scaling of a High-Throughput Content-Based Publish/Subscribe Engine** (Raphaël Barazzutti, Thomas Heinze, André Martin, Emanuel Onica, Pascal Felber, Christof Fetzer, Zbigniew Jerzak, Marcelo Pasin, Etienne Rivière), In ICDCS '14: 34th IEEE International Conference on Distributed Computing Systems

Our Approach to Privacy Preserving Stream Processing

Intel SGX (**S**afe **G**uard **Ex**tensions)

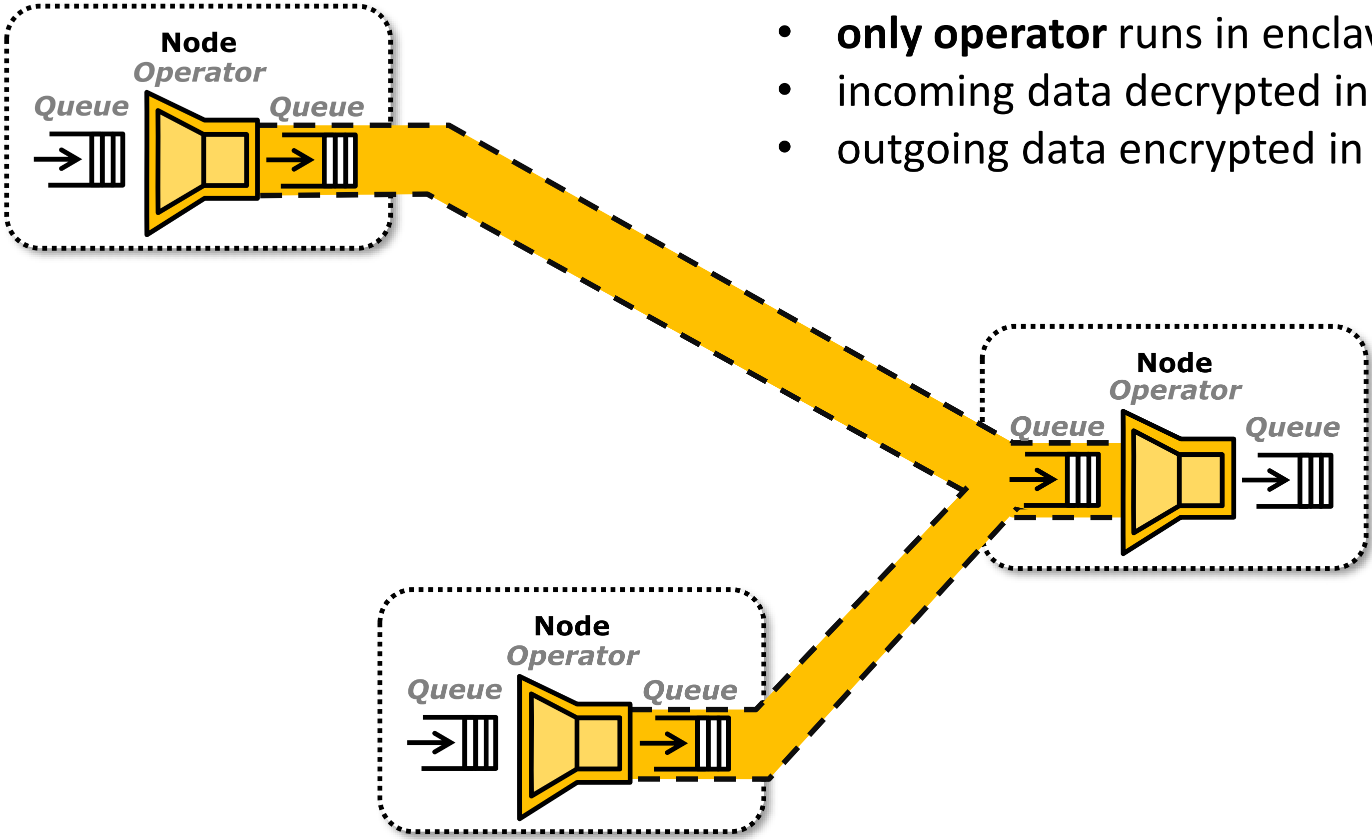
- Trusted environment (**enclave**) for arbitrary code
- Enclave memory cannot be accessed from non-enclave code
- Enclave code has access to outside code/data
- Remote attestation of enclave code
- Available in all new Skylake processors since Q4/15
- User solely need to trust Intel

Intel SGX & Stream Processing Approach #1



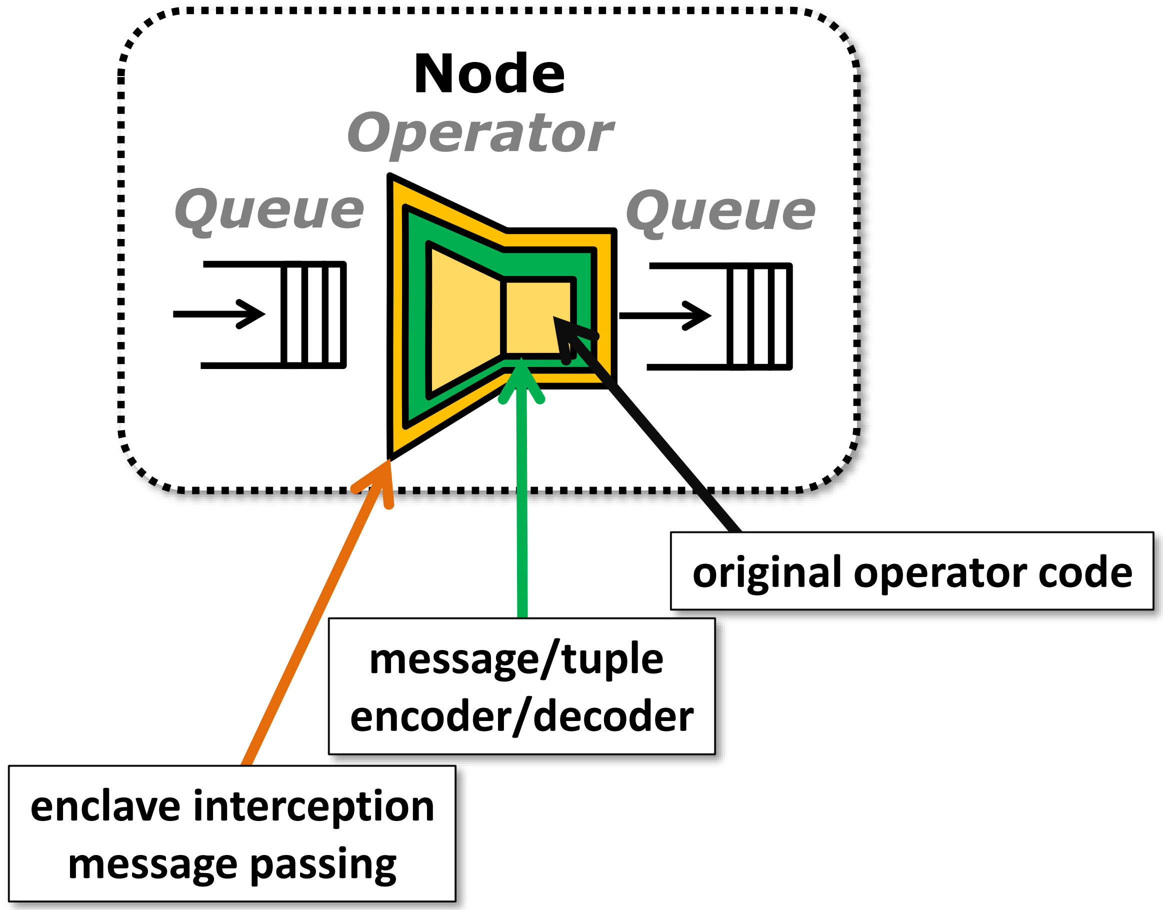
- **whole process** runs in enclave
- secure channels (via TLS/SSL)

Intel SGX & Stream Processing Approach #2



- **only operator** runs in enclave
- incoming data decrypted in op.
- outgoing data encrypted in op.

Approach #2
Transparent Wrapper



Intel SGX Research Challenges

1. Limited EPC (**E**nclave**P**age**C**ache) size (128MB) → How to deal with large operator state?
   - "Swapping": Mechanisms provided by SGX vs. state eviction & encryption strategies tailored to ESP
2. System call interface protection
   - libmusl – exchange data in a controlled manner
3. Enclave threads vs. user space threads
   - How to pass data efficiently between the two worlds?

Summary & Conclusions

1. Lack of **elasticity support** in open source technologies for highly dynamic applications
   - Explicit state support
   - Migration protocol

2. Lack of **privacy preserving stream processing**
   - Operators run in enclaves (Intel SGX)
   - Transparent/non-invasive approach
   - Promising direction – roll out of Skylake processors in Q4/15

Thank you for your attention – Q&A

andre.martin@se.inf.tu-dresden.de