Privacy-Preserving Computing

Jonas Spenger (RISE, KTH, jonas.spenger@ri.se), Paris Carbone (RISE, KTH, paris.carbone@ri.se), Philipp Haller (KTH, phaller@kth.se)

This work was partially funded by the Swedish Foundation for Strategic Research (SSF grant no. BD15-0006) and by Digital Futures.

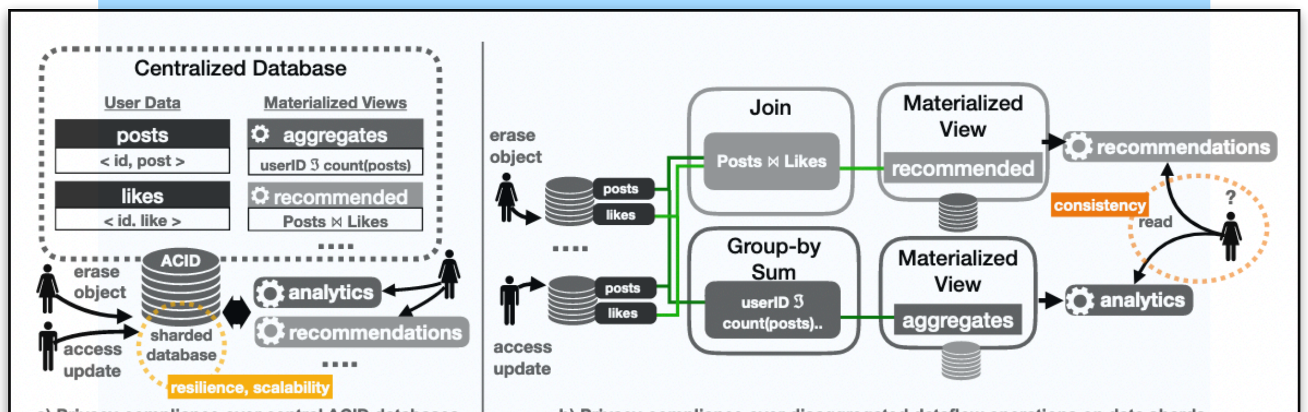
Introduction

- **Privacy-preserving/secure computation** allows multiple parties to securely share sensitive data and collaborate on this shared data, without violating the privacy of individuals.
- To support this type of computation we need a system that can compose **secure workflows** together with existing workflows.

Contributions

- We are building a system for the composition of complex (privacy-preserving) workflows
- with support for:
    - **data ownership**; **secure multi-party computation (MPC)**; and **GDPR**
    - **live serializable/consistent updates**
    - **dynamic, cyclic applications**
- that is serverless, fault-tolerant, scalable.

Use-Case 1: Privacy-Compliant Service Composition

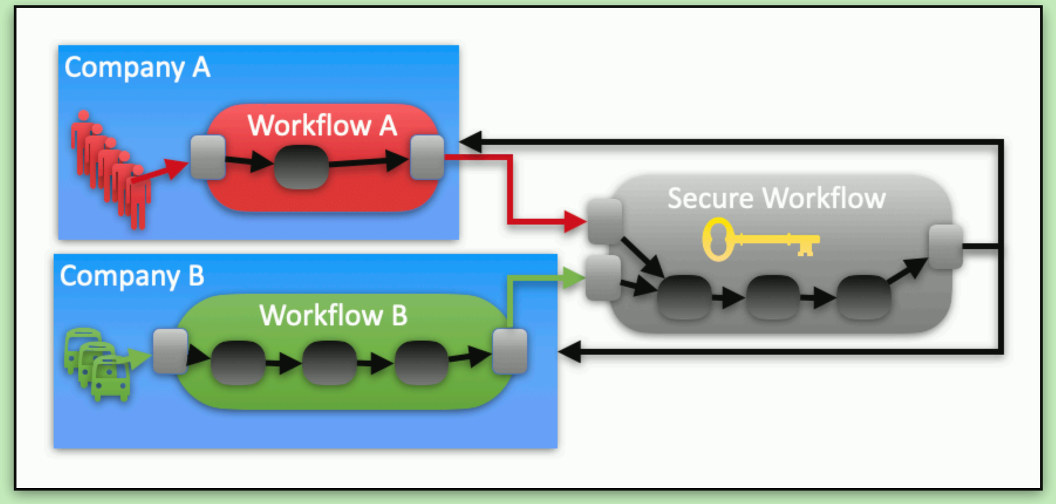


a) Privacy-compliance over central ACID databases.

b) Privacy-compliance over disaggregated dataflow operations on data shards.

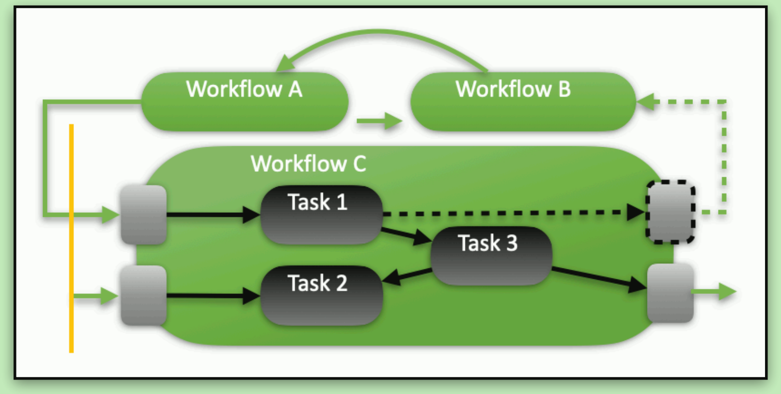
GDPR using live serializable updates
(causally-consistent operations) [Poly'21]

Use-Case 2: Pods Secure Workflows



- Secure Workflows are **collaborative computations on streams of events**
- without leaking any of the information of the ingested events to the other party.

Use-Case 3: Composition of Microservices



- Workflows ingest streams of events and execute a DAG of tasks.
- We allow cycles, and dynamic communication between workflows (workflows are actors).
- Workflows implement Live Serializable Updates (consistent updates to system execution state at runtime)

Example Code

```
val external_task = ...
val workflow = Workflows.builder().source[String]()
 .process( {(ctx, event) =>
   val future = ctx.call(external_task, event) // call external task
   val x = ctx.await(future) // await for response
   ctx.emit(x)}) // emit result
 .sink[Int]()
```

Selected Related Work

Our work is related to actor systems (Akka, Erlang, Reactors), and stateful serverless systems (Durable Functions, Flink StateFun). Our system distinguishes with support for privacy and for dynamic application patterns.

Roadmap

2022

- Release v1.0: workflows; data ownership; serializable updates

2023

- Release v2.0: integration of MPC into workflows

Conclusion

Dynamic (secure) workflows are expressive, and suitable for the composition of microservices. A system with the proposed properties would enable more collaboration, and a principled approach to privacy.

[Poly'21] Spenger, J., Carbone, P., & Haller, P. (2021). WIP: Pods: Privacy Compliant Scalable Decentralized Data Services. In Heterogeneous Data Management, Polystores, and Analytics for Healthcare (pp. 70-82). Springer, Cham.

RISE Computer Science & AI open house, May 10 2022, Stockholm, Sweden