Analysis and Enforcement of GDPR Rules on Key-Value Stores

Ertugrul Aypek
Advisor: Dimitrios Stavrakakis, M.Eng.
Chair of Distributed Systems and Operating Systems
https://dse.in.tum.de/

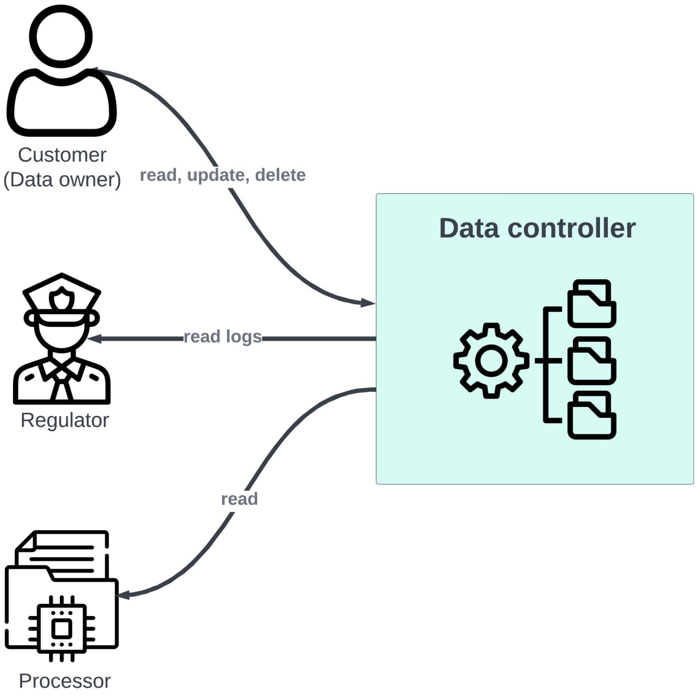
15.02.2023 – 15.08.2023

Background: GDPR definition

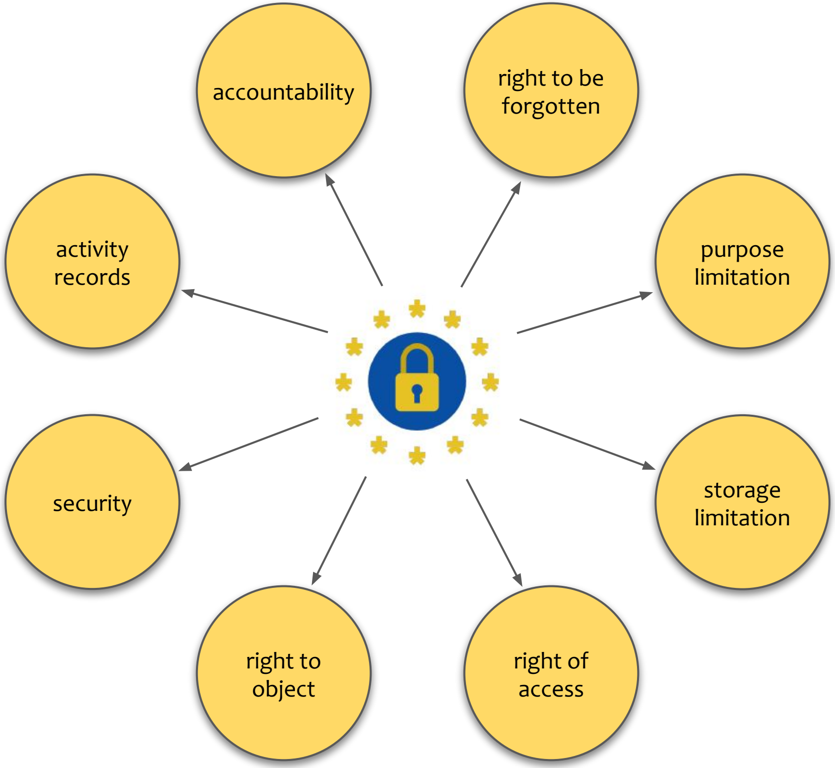
General Data Protection Regulation (GDPR)

- Introduced in 2018 by EU
- Resulted in more than €2.5B in fines
- Blocks businesses in EU market
- Requires system redesign

GDPR Actors



Background: GDPR requirements

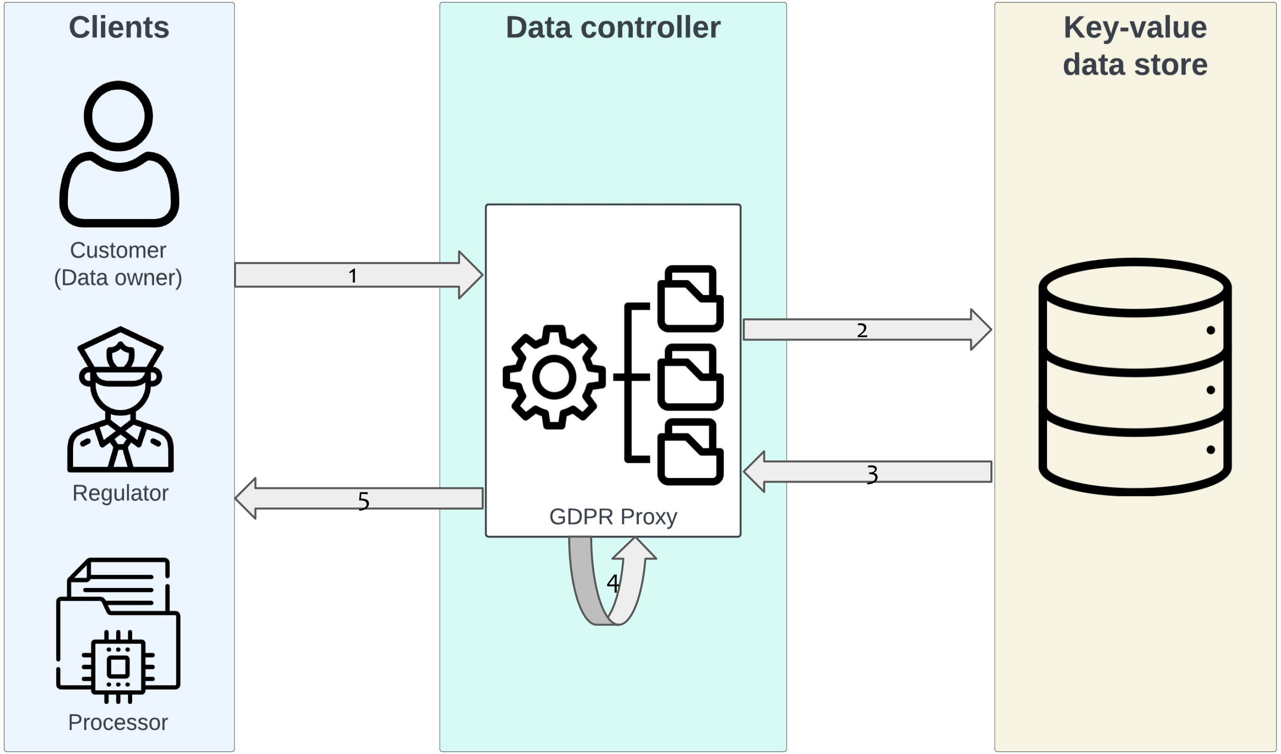


GDPR icon from: https://www.flaticon.com/free-icon/gdpr_1355238

Motivation

- ➢ For GDPR compliance, each system require internal redesign
- ➢ Hard to comply with all the rules and verify compliance
- ➢ Performance overhead implications
- ➢ Little to no focus on key-value store GDPR compliance in literature

System Design



Design: GDPR Metadata

Values are enriched with metadata to be able to comply with GDPR rules:

- user key
- origin
- purpose
- objection
- share
- expiration

Design: Policy Language

- The unified way to interact with the GDPR Proxy
- Default policy as JSON and query overrides
- Policy Compiler to parse queries and enforce query language syntax
- Query Rewriter to merge default policy and query predicates

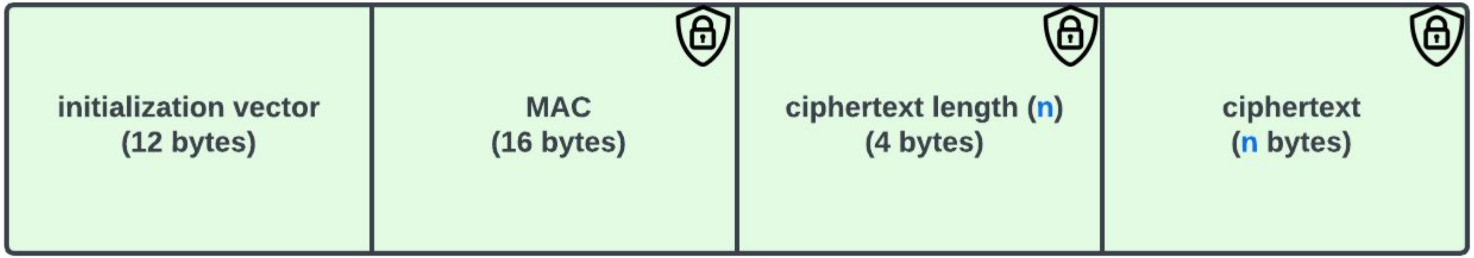
```
query(PUT("gdpr1","VAL"))&userKey("user1")&origin("src1")

&objection("purpose3")&purpose("purpose0,purpose1,purpose2")

&share("user0")&expiration("0")
```

Design: Cipher Engine

- Enables encryption/decryption of metadata enriched values and logs
- Implemented using AES-GCM 128 bit algorithm



initialization vector (12 bytes)

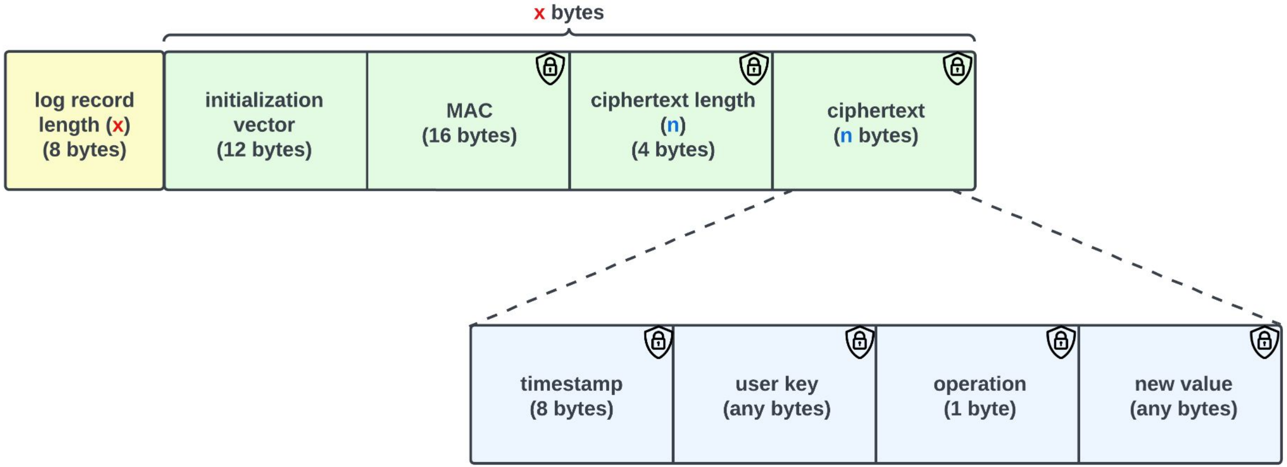
MAC (16 bytes)

ciphertext length (n) (4 bytes)

ciphertext (n bytes)

Design: Logging Engine

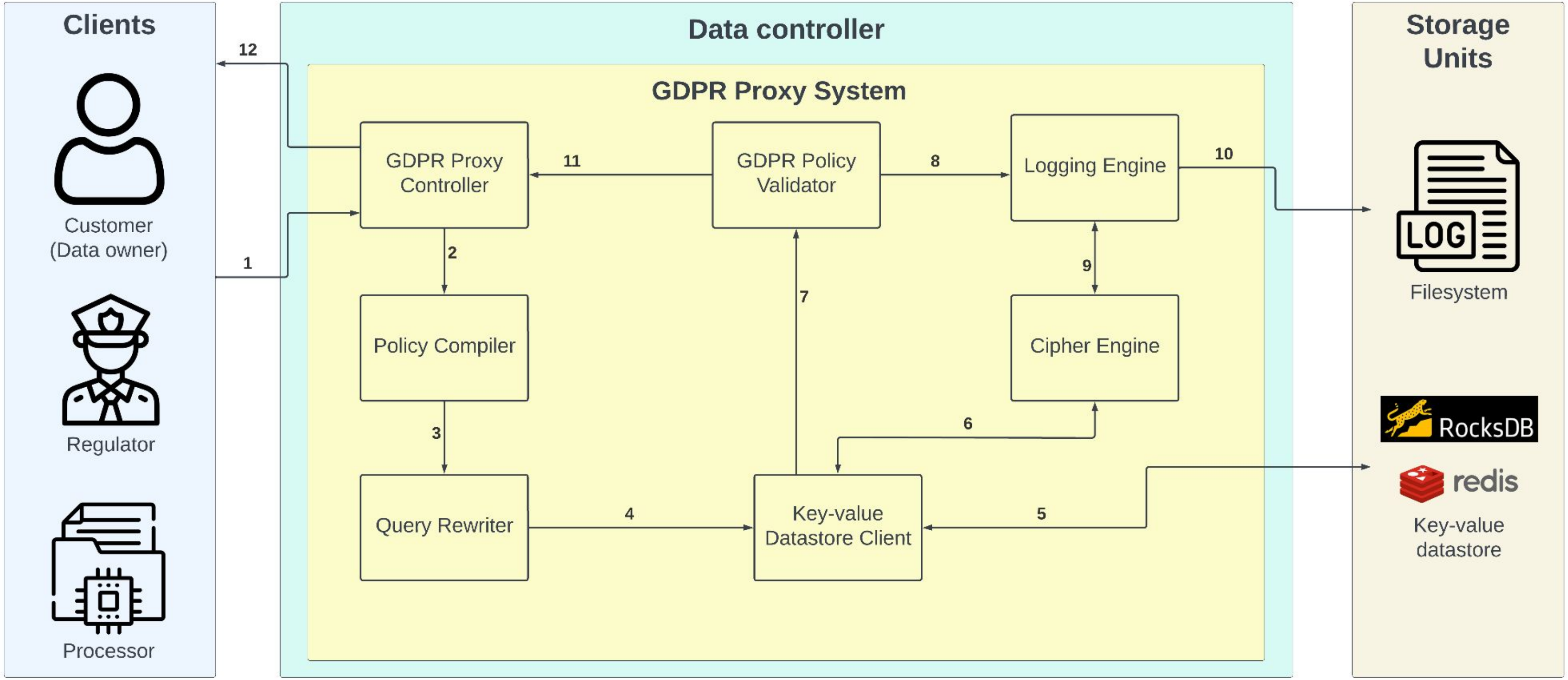
- Used to prove the GDPR compliant processing activities
- Can be kept encrypted in file system



Design: Proxy Controller

- Entrypoint to the system
- Handles user sessions over secure TCP to execute queries
- Parses and stores default policies for each session
- Orchestrates connection to different datastore backends

Design: Design Revisited



Evaluation

Evaluation aspects:

- Correctness (via GDPRBench workloads)
- Speed
- Space

Evaluation: Workloads

Workloads A-F in GDPRBench with

- 1 million records (via put queries)
- 1 million operations (put/get/delete queries)

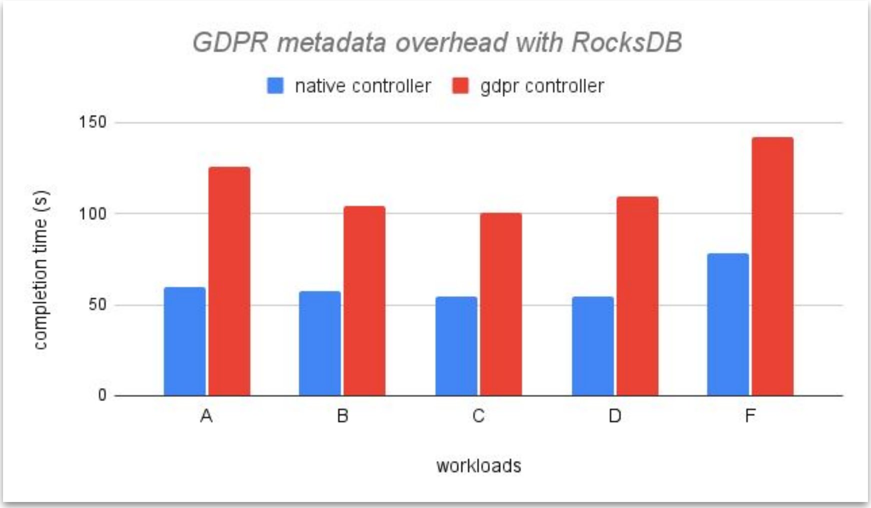
Workload	Operation	Application
A	Read/Update (50/50%)	Session store
B	Read/Update (95/5%)	Photo tagging
C	Read (100%)	User profile cache
D	Read/Insert (95/5%)	User status update
F	Read-Modify-Write (100%)	User activity record

Evaluation: Controllers

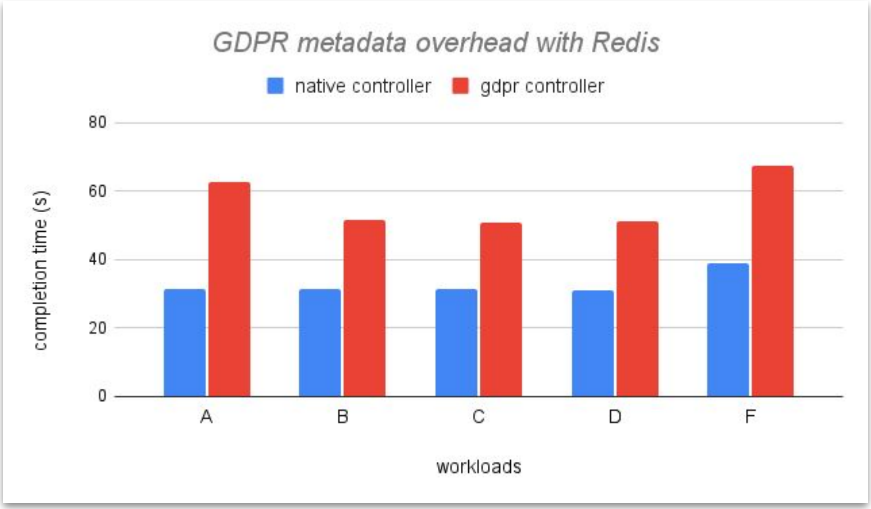
Different types of controllers to measure system performance

Functionality	**Native Controller**	**GDPR Controller**
GDPR Metadata	no	yes
Policy Language	no	yes
Policy Compiler	no	yes
Query Rewriter	no	yes
Cipher Engine	no	optional
Key-value Client	yes	yes
Logging Engine	no	optional
Policy Validator	no	yes

Evaluation: GDPR Metadata

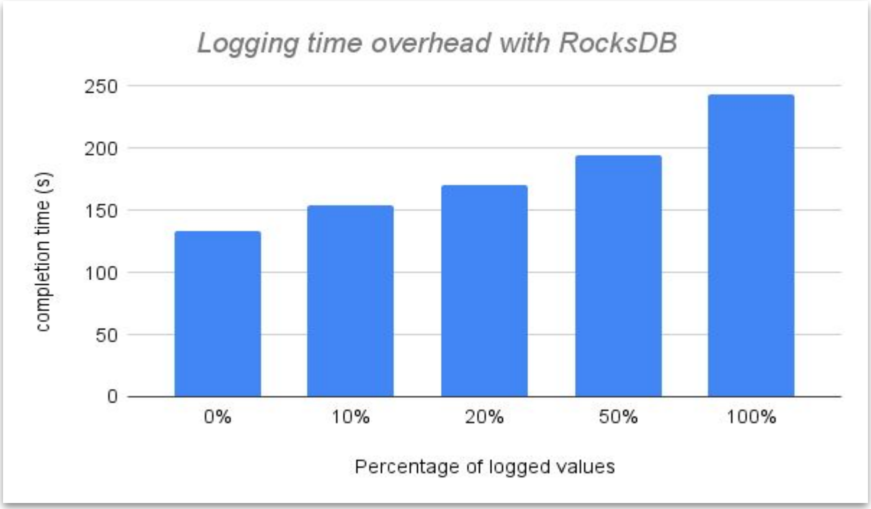


Lower is better

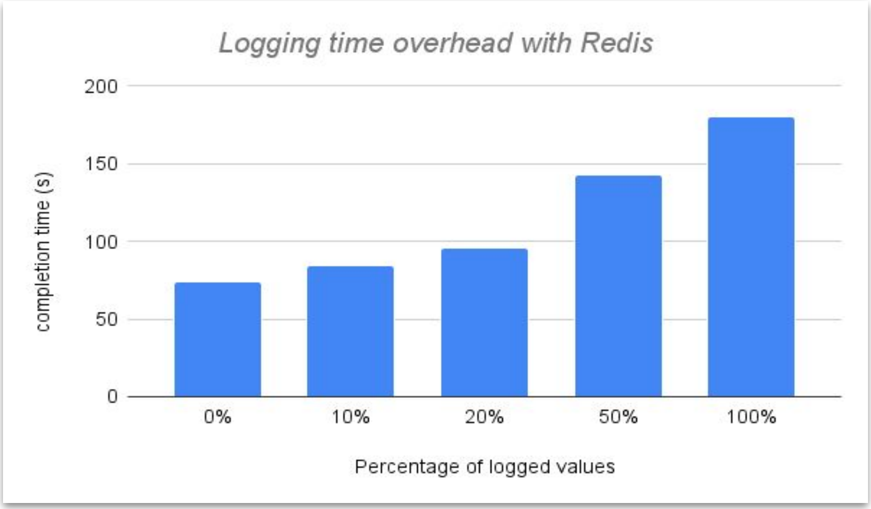


GDPR metadata processing of all workloads is 90% for RocksDB and 73% for Redis

Evaluation: Logging

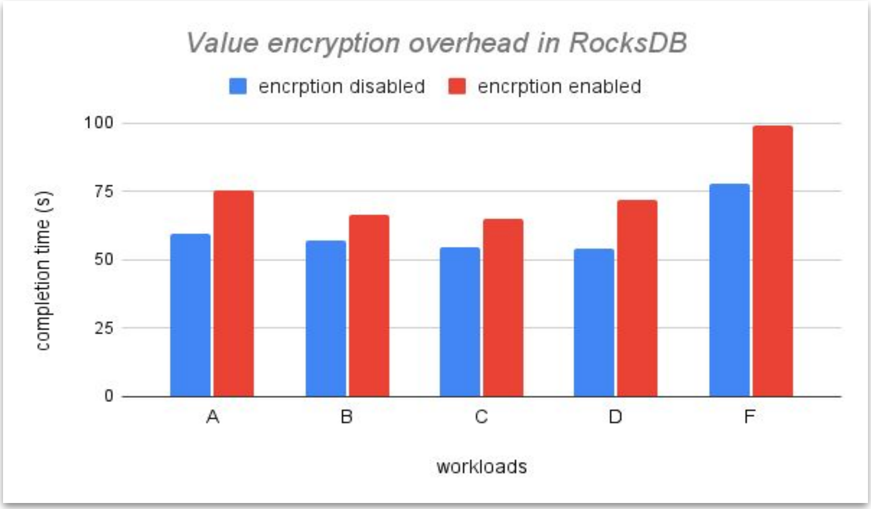


Lower is better

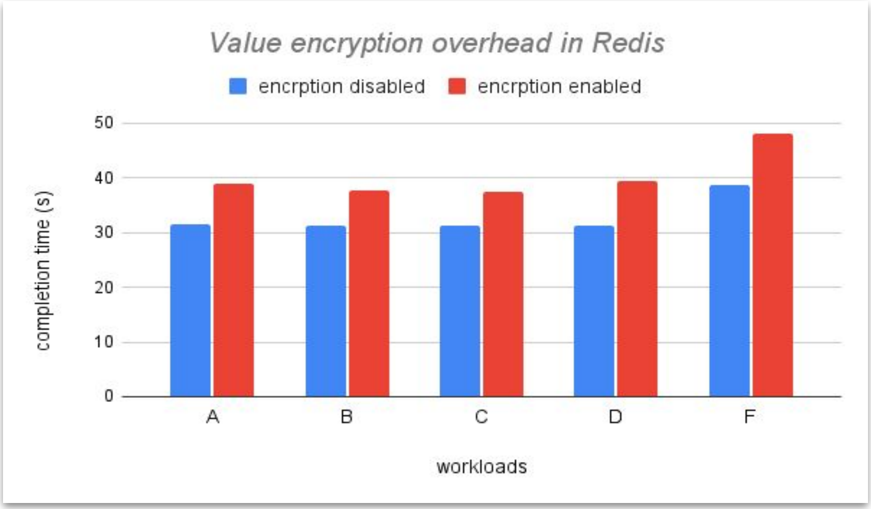


Logging 0% vs 100% of queries adds around 100 seconds overhead

Evaluation: Value Encryption

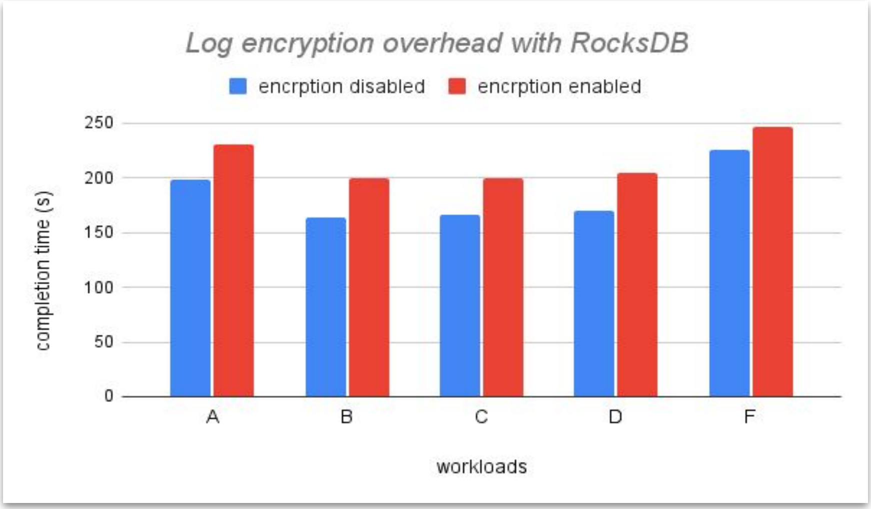


Lower is better

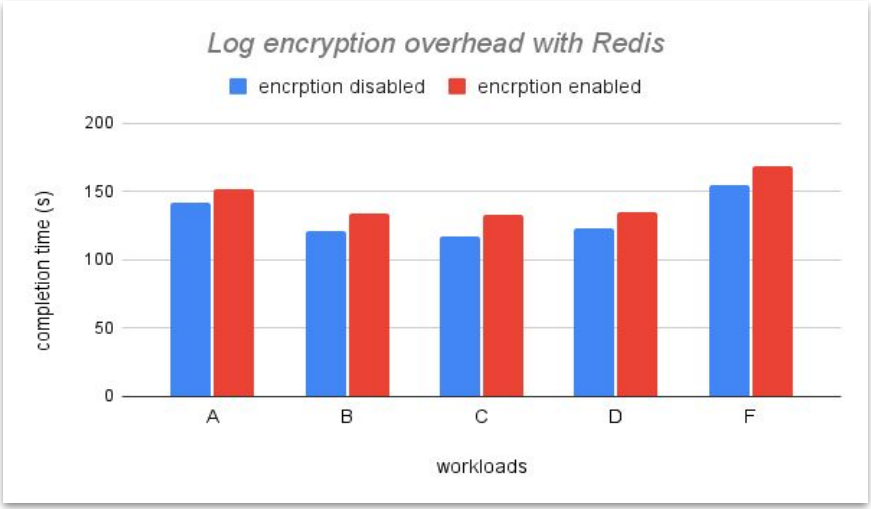


Value encryption overhead is 24% for RocksDB and 23% for Redis.

Evaluation: Log Encryption

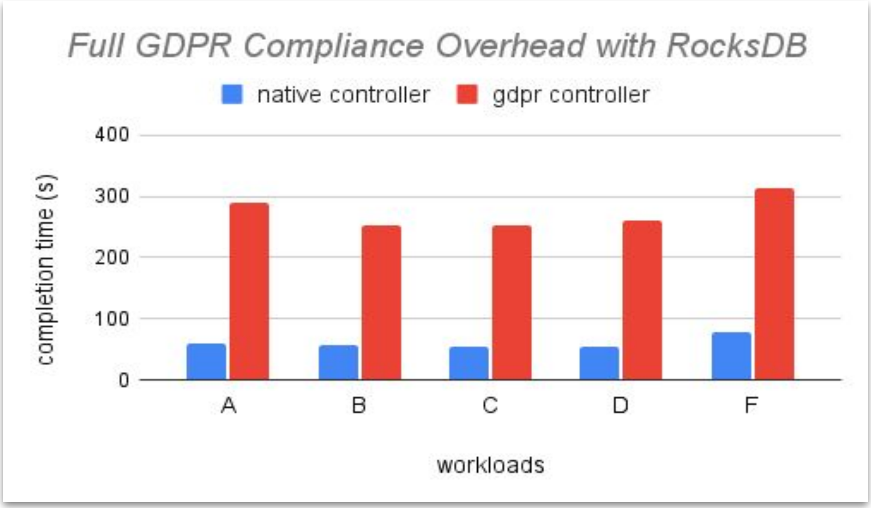


Lower is better

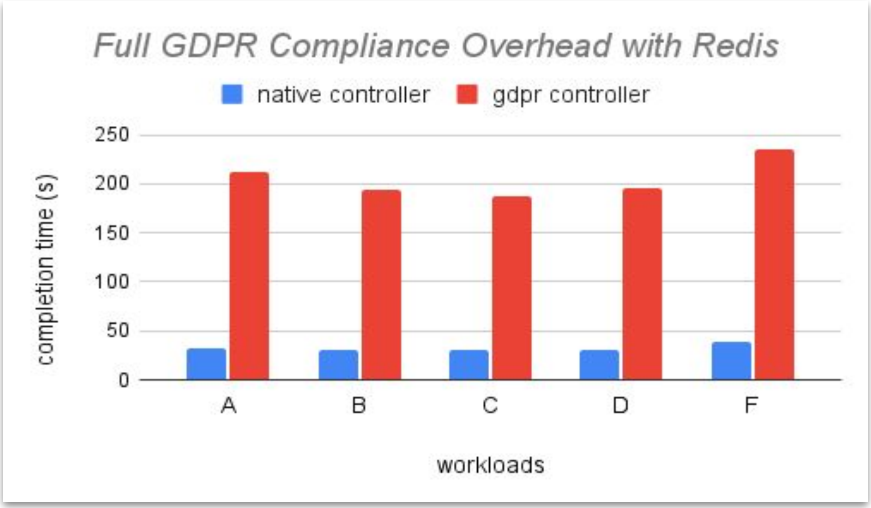


Log encryption of all workloads is 18% for RocksDB and 10% for Redis

Evaluation: Full GDPR Compliance Overhead



Lower is better



Full GDPR compliance overhead is 3.5x for RocksDB and 5x for Redis

Ways to reduce it: faster encryption algorithms and asynchronous logging

Summary

Current solutions to GDPR are not feasible

- Changes in application business logic
- Changes in database internals
- Varying performance overheads

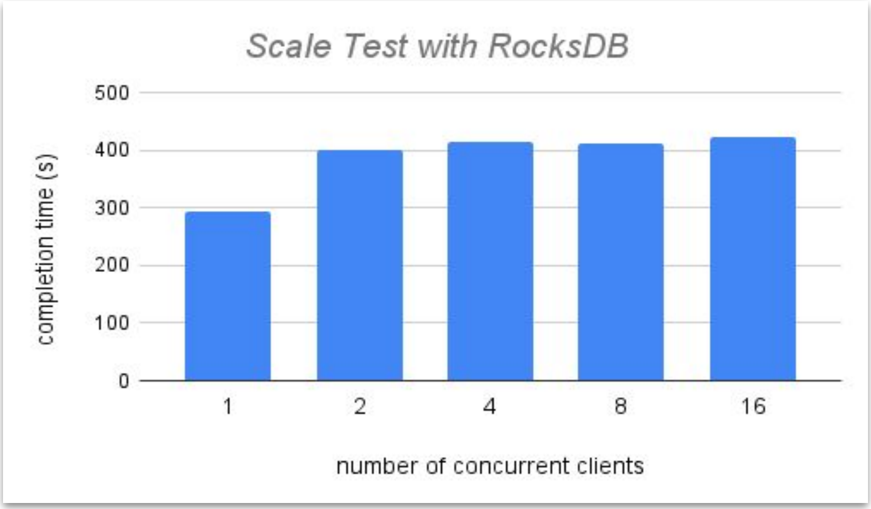
GDPR Proxy:

- Generic GDPR compliant proxy for key-value stores
- No change in database internals
- Easily adaptable and extensible

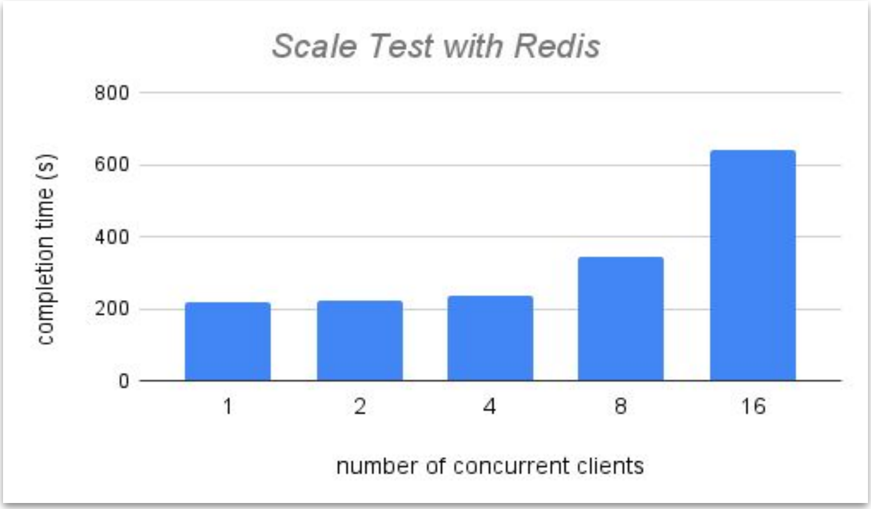
https://github.com/ertugrulaypek/GDPRoxy

Backup

Evaluation: Scaling

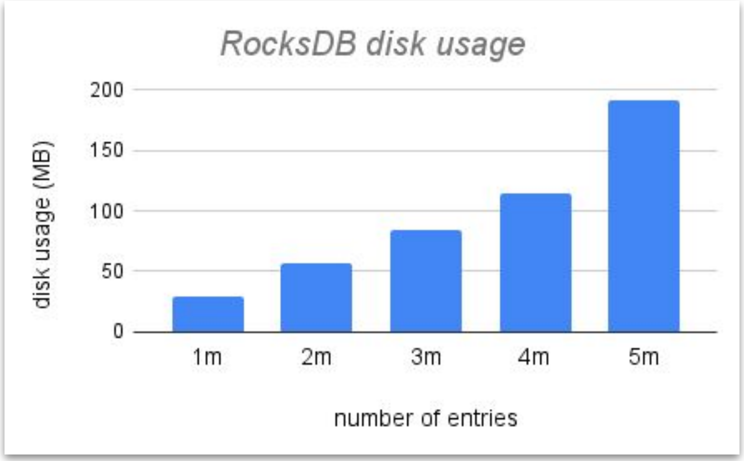


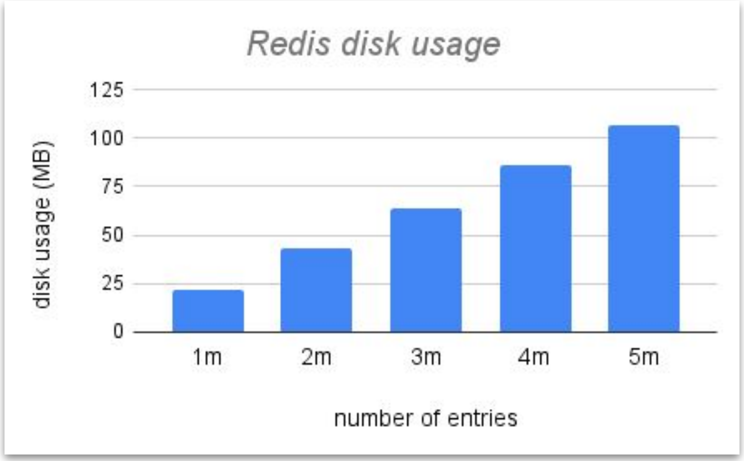
Lower is better

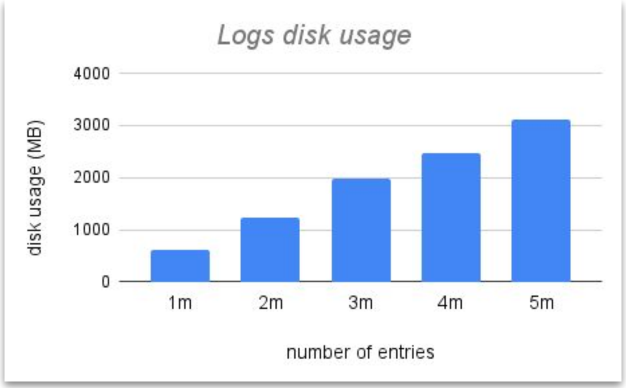


The completion time is not linearly increasing with concurrent clients

Evaluation: Disk usage


RocksDB disk usage
disk usage (MB)
1m
2m
3m
4m
5m
number of entries


Redis disk usage
disk usage (MB)
1m
2m
3m
4m
5m
number of entries


Logs disk usage
disk usage (MB)
1m
2m
3m
4m
5m
number of entries