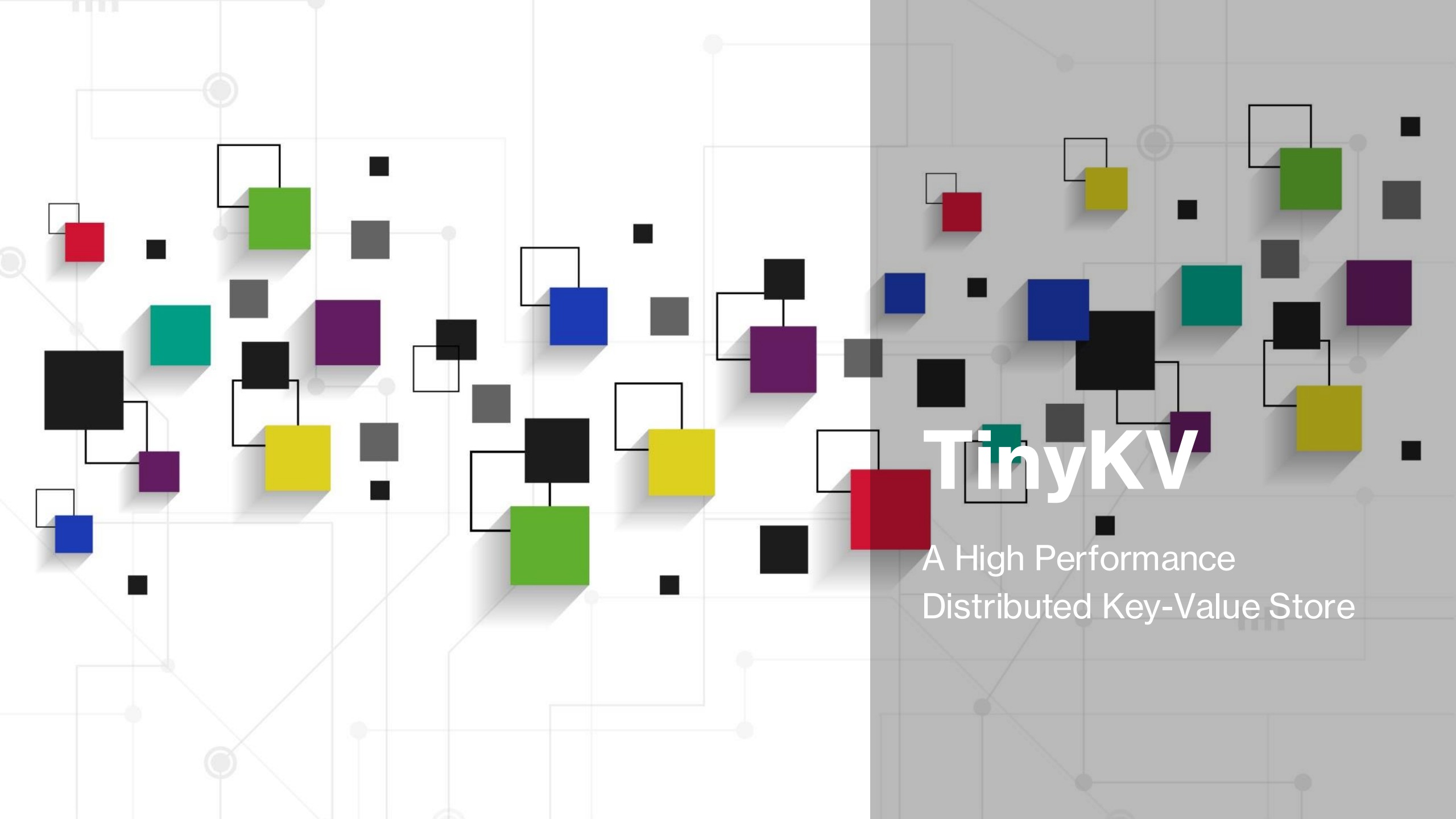


The background features a light gray grid with thin white lines and small circles, resembling a circuit board. Scattered across this grid are numerous squares of various colors (red, green, blue, yellow, purple, black, and gray) and sizes. Some squares are solid, while others are outlined. The right half of the image is shaded in a darker gray, providing a backdrop for the title and subtitle.

TinyKV

A High Performance
Distributed Key-Value Store

Concept

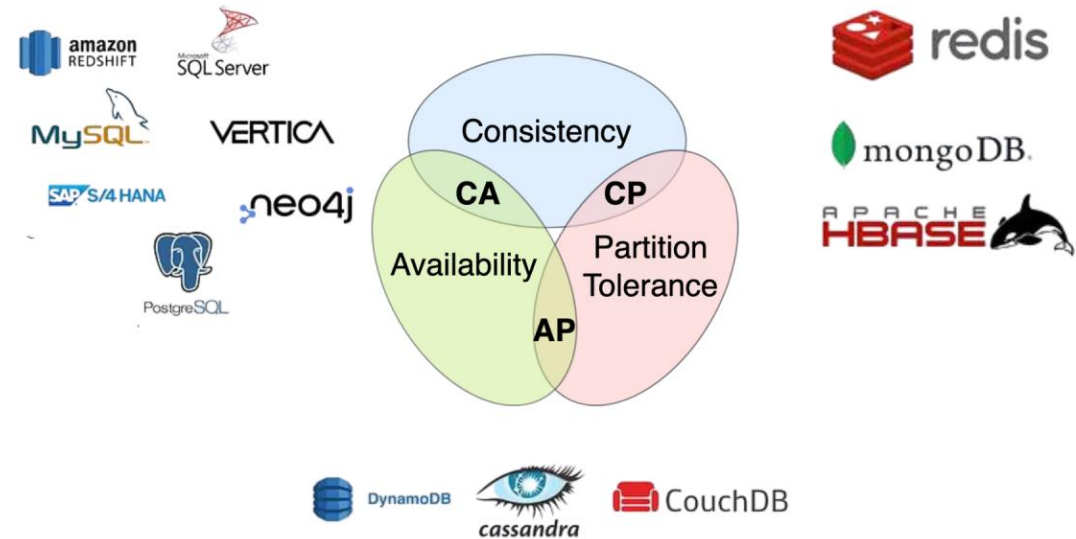


Amazon
DynamoDB

- Decentralized peer-to-peer architecture
- Availability and network partitioning
- Performance
- Tunable consistency

Motivation and Learning goals

- CAP theorem in practice
- Performance and latency in distributed systems
- Practical experience with replication, partitioning, quorums and failure detection

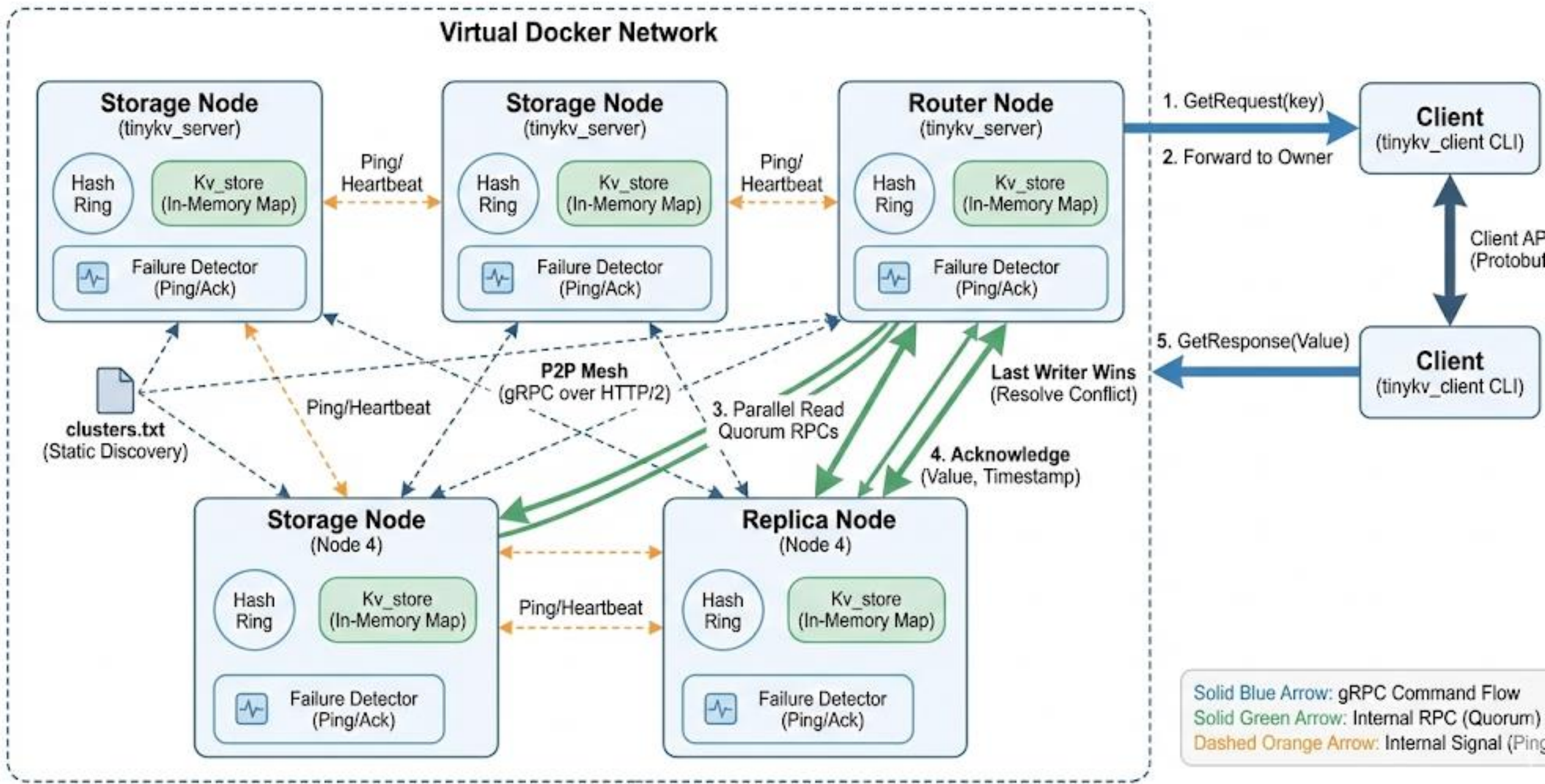




Tools and technologies

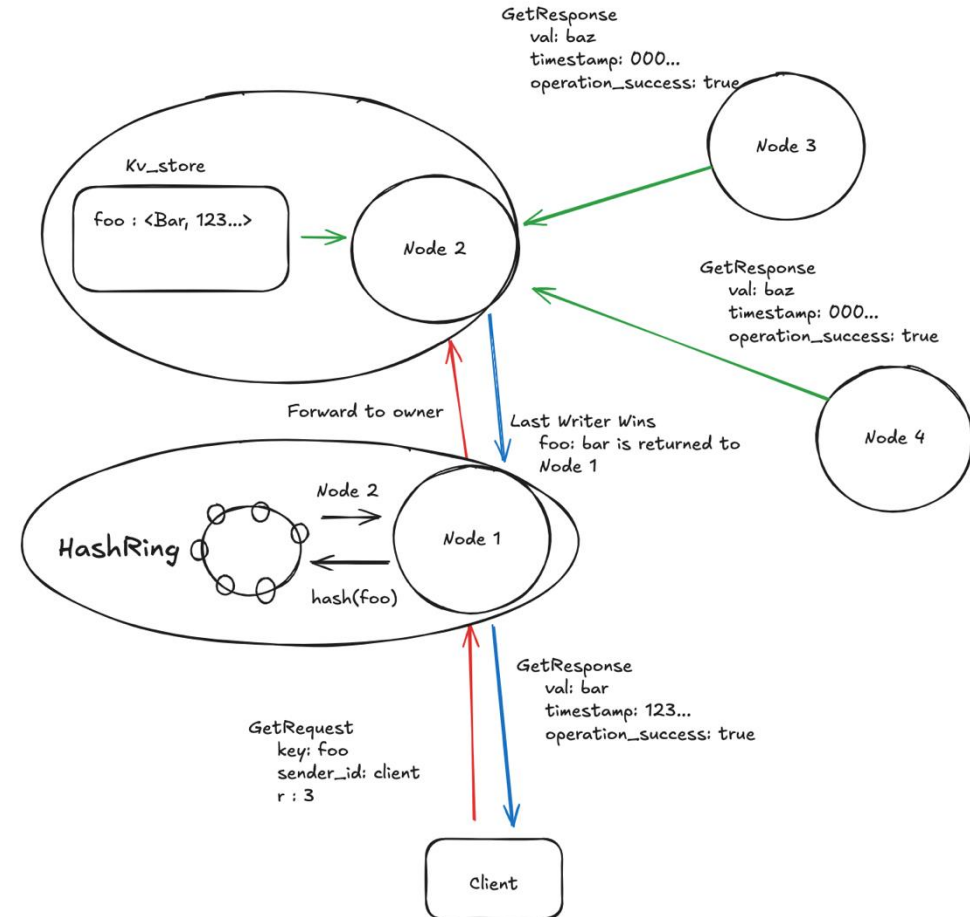
- C++20
 - High performance and fine grained control
- gRPC
 - Efficient communication and memory compact payloads
- Docker
 - Portability and containerization

TinyKV System Architecture Diagram

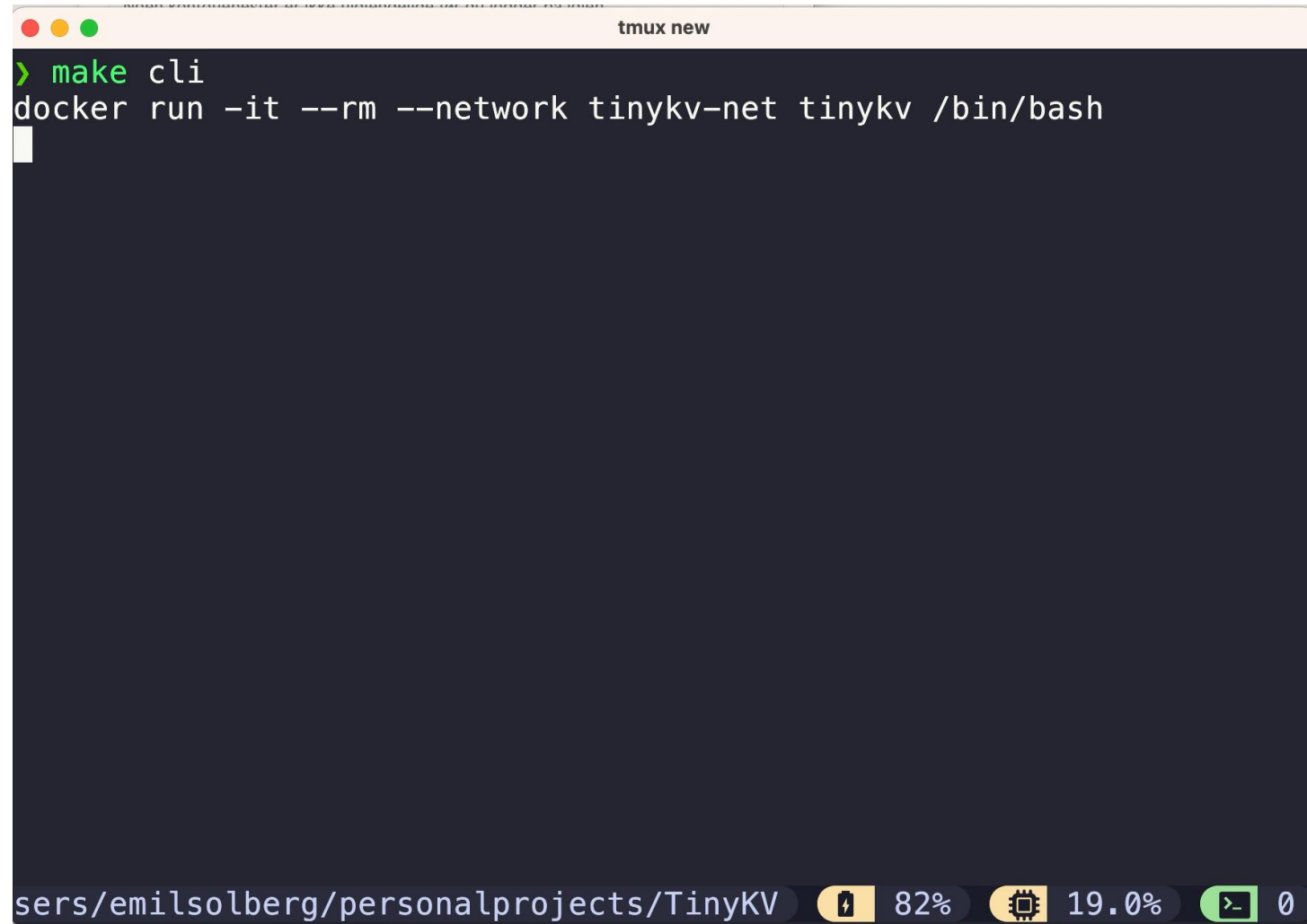


How it works

- Hash ring
- In memory hashmap with timestamped values
- Read and write quorums



Demonstration

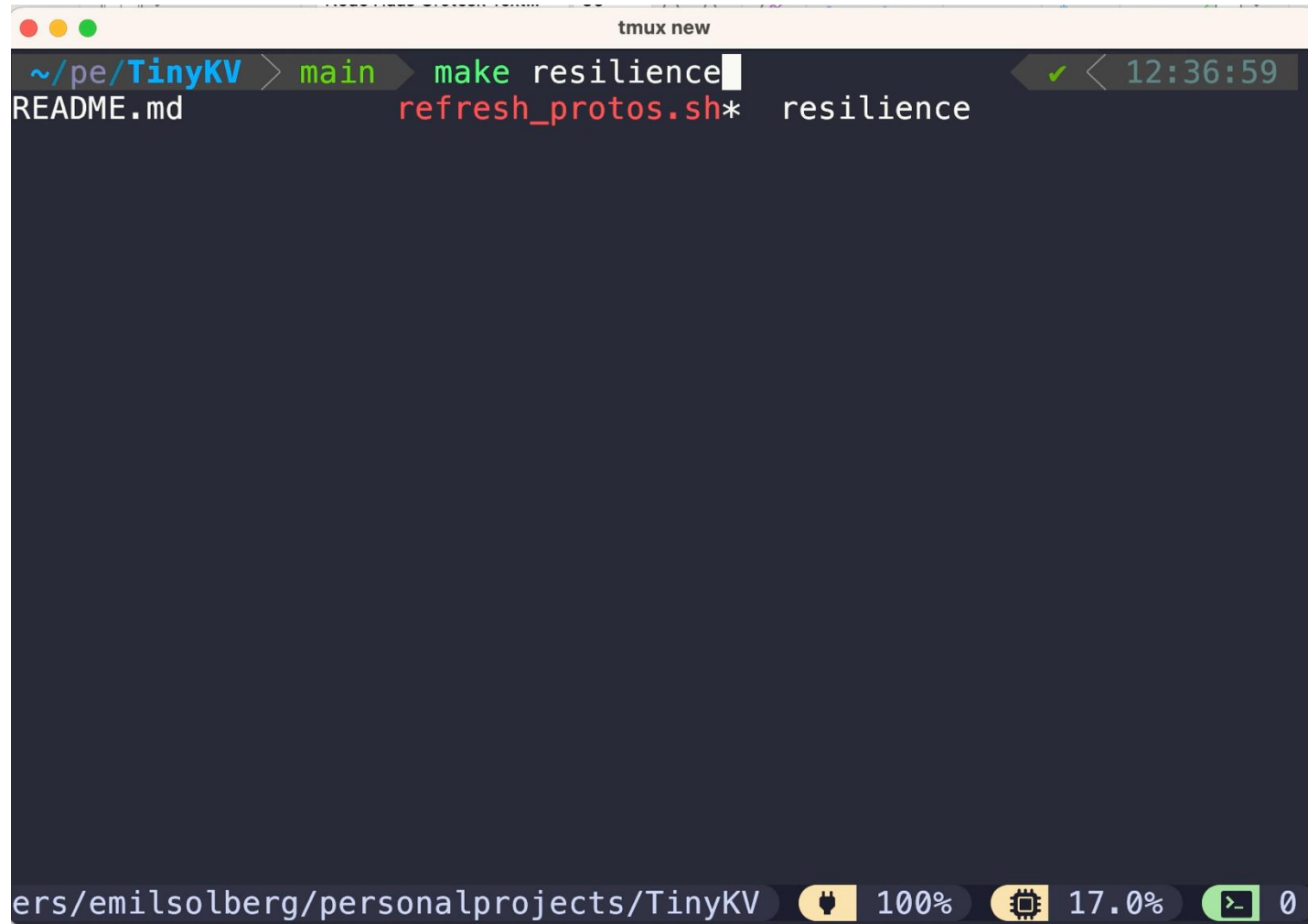


A terminal window titled "tmux new" with a dark background. The window shows two lines of text: a green prompt character followed by "make cli" and a white prompt character followed by "docker run -it --rm --network tinykv-net tinykv /bin/bash". The window has standard macOS window controls (red, yellow, green buttons) in the top-left corner. At the bottom of the screen, there is a status bar showing the path "sers/emilsolberg/personalprojects/TinyKV", a battery icon at 82%, a CPU icon at 19.0%, and a terminal icon with a green cursor and the number 0.

```
> make cli  
docker run -it --rm --network tinykv-net tinykv /bin/bash
```

sers/emilsolberg/personalprojects/TinyKV 82% 19.0% 0

Tunable consistency

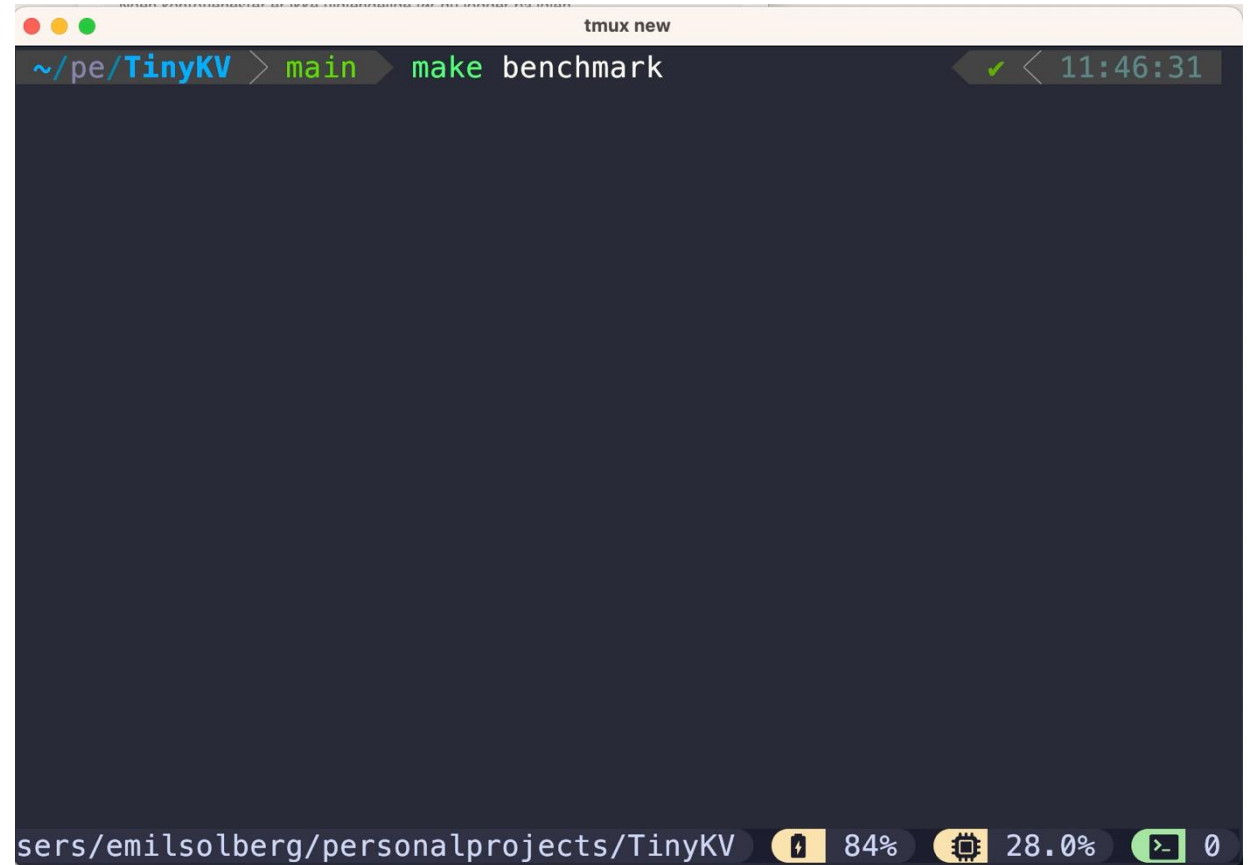


```
tmux new
~/pe/TinyKV > main make resilience
README.md refresh_protos.sh* resilience
```

ers/emilsolberg/personalprojects/TinyKV 100% 17.0% 0

Performance

- 10'000 requests, 50 clients
- Up to 5000 OPS
- Achieved through parallelism
- Limited by terminal output, docker overhead and router node bottleneck



Improvements and future works



Gossip protocol



Asynchronous network requests



Comprehensive test suite



DELETE command



Security and authorization



Dynamic membership



Load balancing