Container-based Video Streaming Service

International Symposium on Computational Intelligence and Informatics
22.11.2022, Budapest, Hungary

Martin Bobák
Institute of Informatics
Slovak Academy of Sciences

Introduction

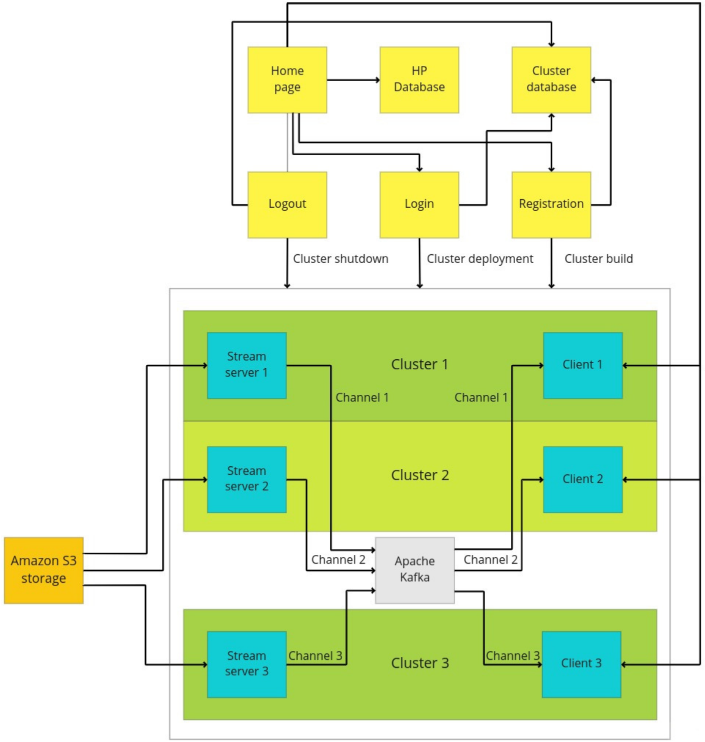
- effective processes utilization
- virtualization as seamless access to hardware resources
- containerization, microservice architecture
- microservice scalability via containerization
- video distribution
- effective streaming

Motivation

- cross-platform interoperability
- difficult network management
- security and isolation
- delayed and lost packets

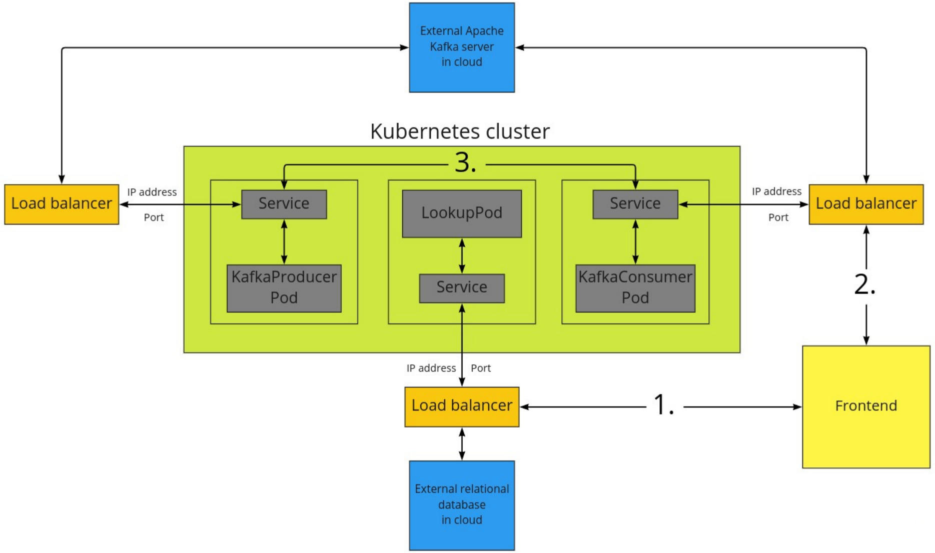
Architecture - overview

- service-oriented architecture
- microservice-driven approach
- each user has an isolated server cluster



Architecture - Communication

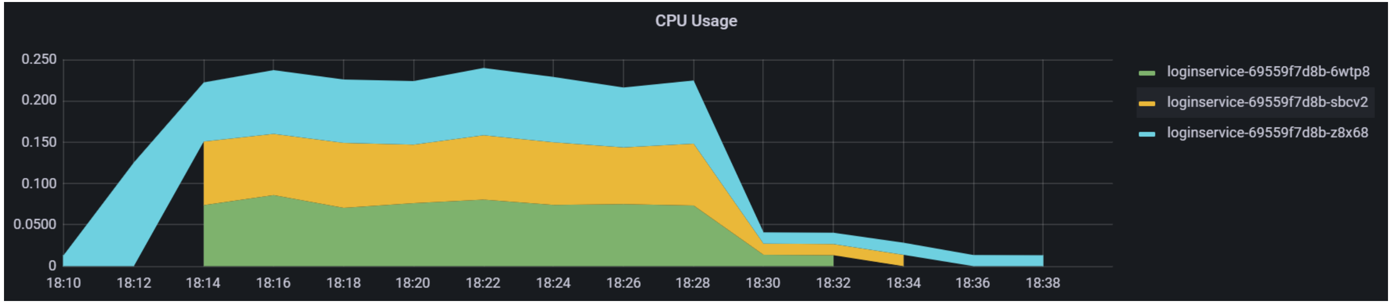
- a user plays a video from the frontend
- searching the IP address and port of the pod in the database
- the consumer requests the Kafka producer, which sends the required parts of the video



Experiments

- containerization focussed
- CPU utilization
- RAM utilization
- Network communication

CPU utilization



Containerization + horizontal scaling

	CPU units		
Pods	Before	During application load	After
First pod	0.01	0.075	0.01
Second pod	-	0.075	-
Third pod	-	0.075	-

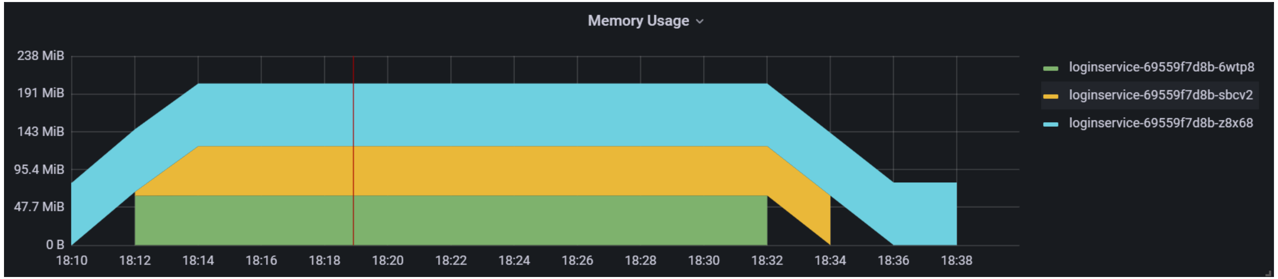
Containerization

	CPU units		
Pods	Before	During application load	After
First pod	0.01	0.245	0.01

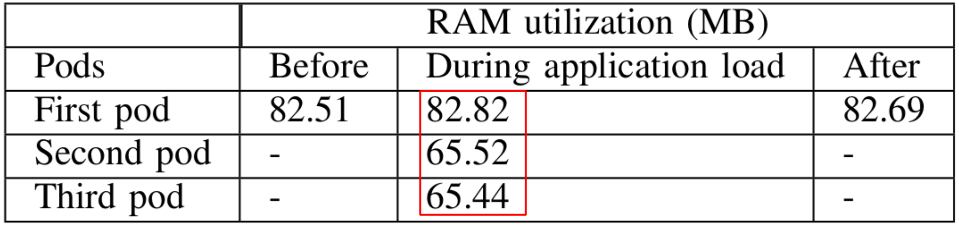
Without containerization

	CPU units		
Processes	Before	During application load	After
First process	0.24	3.1	0.22

RAM utilization

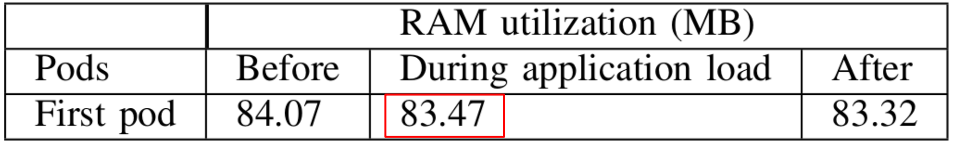


Containerization + horizontal scaling



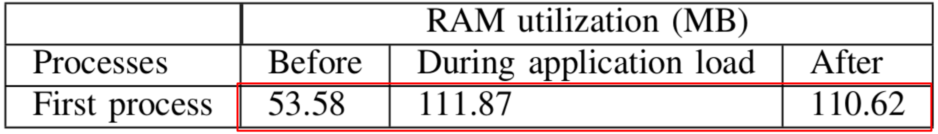
	RAM utilization (MB)		
Pods	Before	During application load	After
First pod	82.51	82.82	82.69
Second pod	-	65.52	-
Third pod	-	65.44	-

Containerization



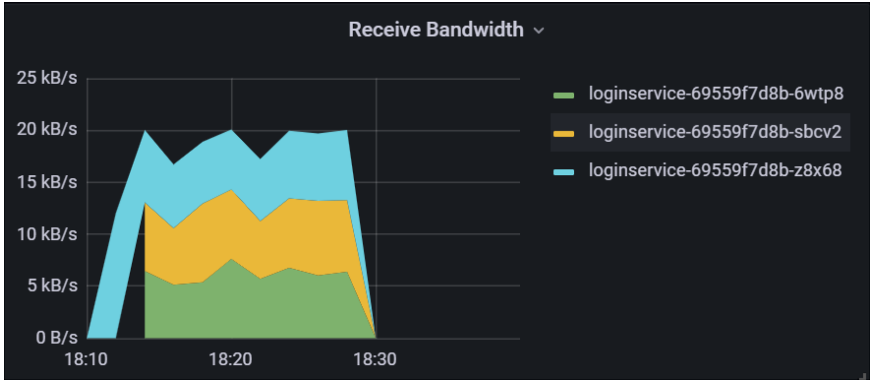
	RAM utilization (MB)		
Pods	Before	During application load	After
First pod	84.07	83.47	83.32

Without containerization

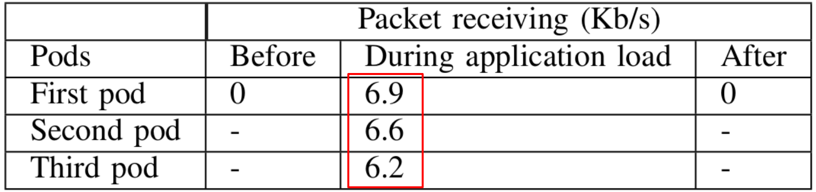


	RAM utilization (MB)		
Processes	Before	During application load	After
First process	53.58	111.87	110.62

Network utilization

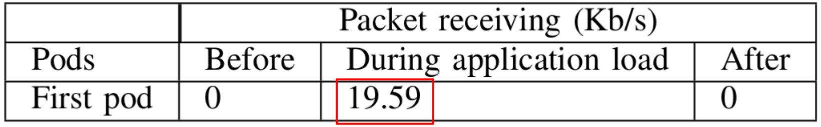


Containerization + horizontal scaling



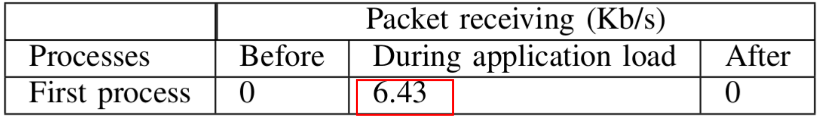
	Packet receiving (Kb/s)		
Pods	Before	During application load	After
First pod	0	6.9	0
Second pod	-	6.6	-
Third pod	-	6.2	-

Containerization



	Packet receiving (Kb/s)		
Pods	Before	During application load	After
First pod	0	19.59	0

Without containerization



	Packet receiving (Kb/s)		
Processes	Before	During application load	After
First process	0	6.43	0

Conclusion

- containerization improves the utilization of computing resources and the modularity of the whole application
- containerized services can receive and send a much higher amount of data
- modularity enhances the scaling properties of the microservice application, which increases its availability in higher load
- horizontal scaling strongly depends on the type of service (database, etc.)

Thank you for your attention!

martin.bobak@savba.sk

INSTITUTE OF INFORMATICS
SLOVAK ACADEMY OF SCIENCES