Emily Ng's Semester Project: Optimizing Caching Workflows with Kubernetes and Bloom Filters

CS5340: Software Maintenance
Spring 2025
Permission is granted for publication on the UCCS website.
Professor: Dr. Armin Moin
TA: Ariful Rabbani

Project Idea and Topic

- **Implement phases of a full-scale URL-shortening system (YOURLS) using Docker networks and Kubernetes clusters**
  - Add scale of redis caching and bloom filtering as well
- **Evaluate the latency, CPU usage, throughput, and response time of a system at variable levels of scaling.**
  - Also evaluate the key operations of the application “shorten” and “redirect

Vocabulary

- **Kubernetes:** an open-source system that automatically manages, scales, and deploys containerized applications, kind of like a smart operating system for clusters of servers.
- **Container:** a lightweight, portable package that bundles an application's code with everything it needs to run like libraries, dependencies, and settings so it works consistently across different environments.
- **Cache:** a cache is a temporary storage area that holds frequently accessed data so it can be retrieved faster instead of recalculating or fetching it from a slower source like a database or disk.
- **Bloom Filter:** a space-efficient, probabilistic data structure used to quickly test whether an element is possibly in a set or definitely not in it, with a small chance of false positives but no false negatives.

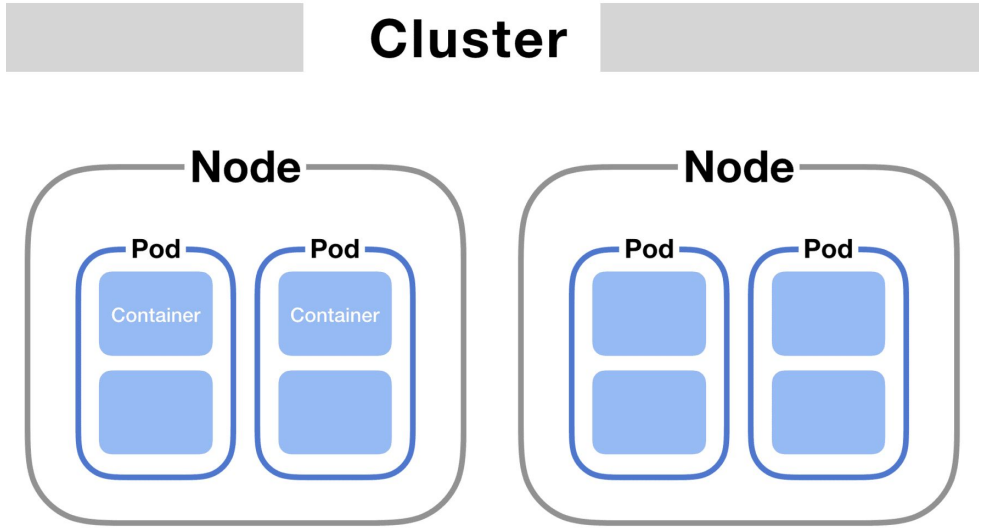
Kubernetes Keys

Cluster: A coordinated group of nodes that work together to efficiently manage and run containerized applications at scale.

Node: An individual “worker” machine (VM) like a warehouse that provides the compute resources needed to run apps.

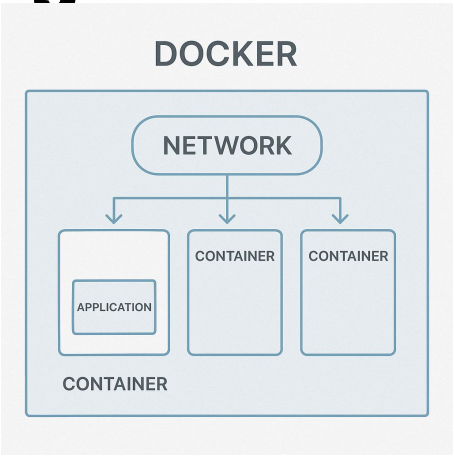
Pod: A lightweight, efficient unit inside the warehouse that wraps around one or more containers. It optimizes resource sharing like storage, networking, and lifecycle making the containers inside it run more smoothly and cohesively.

Container: The actual product inside the shipment (pod), holding the app and all its dependencies. Containers are portable and fast to start, enabling efficient and consistent app deployment.

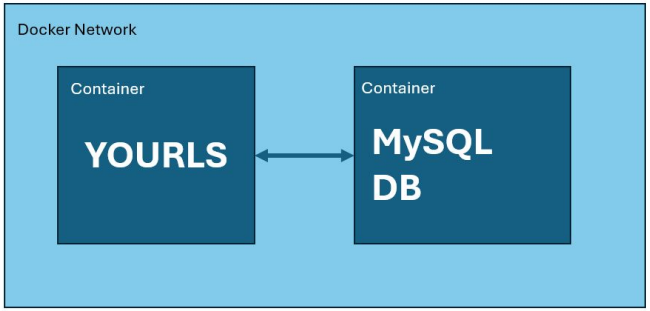


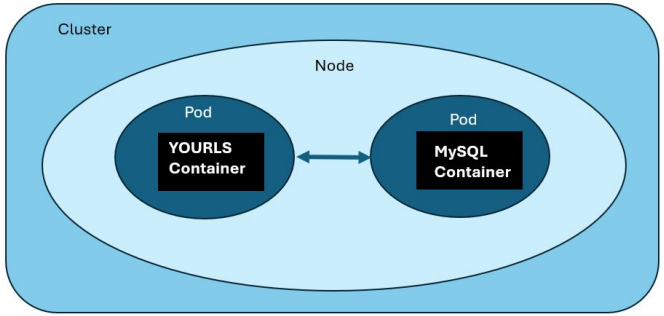
Docker Network

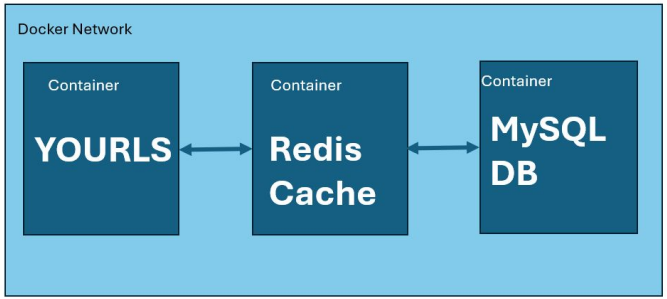
A virtual “bridge” that lets containers securely communicate with each other and the outside world simulating a mini network environment for your applications.

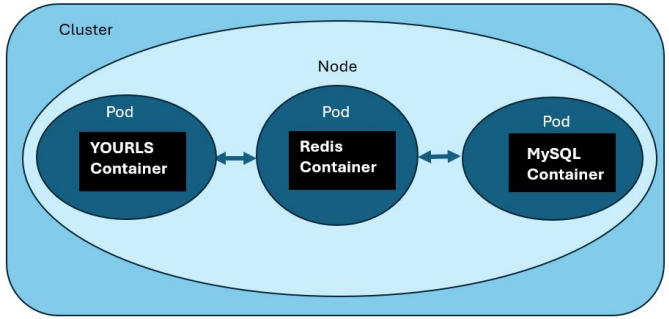


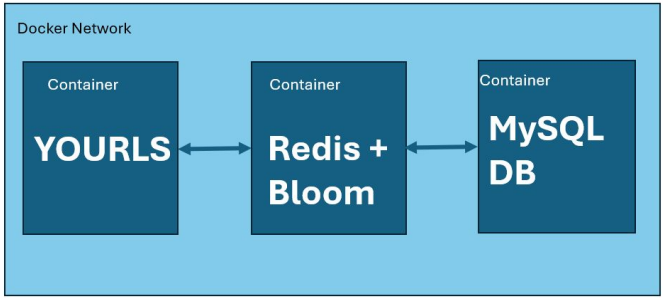
Kubernetes	Docker Network
● Self-healing and Resilient by design ● Declarative configuration through yaml files (it'll figure out what to do if you provide some specifications) ● More allocatable resources balanced easily by a scheduler than local instances ● Highly Cloud Portable and flexible ● Heavy learning curve ● Best for larger-scale system and more users	● Manual healing and re-upstart ● Imperative Configurations (you must be exact in telling it what to do) ● More stringent resource allocations that must be manually set as not based in a "cloud" ● Fragile and needs strategic re-architecting for cloud environments ● Simple to use ● Easy for small-scale, low-throughput requirements

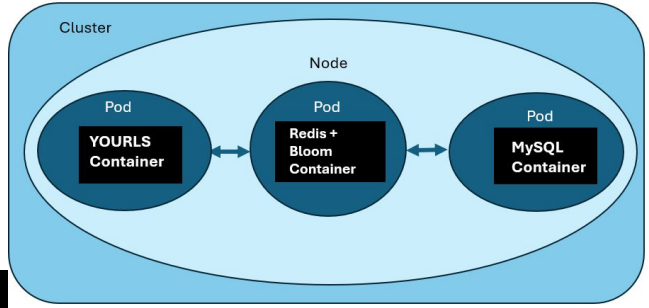

Docker Network
Container
YOURLS
Container
MySQL
DB


Cluster
Node
Pod
YOURLS
Container
Pod
MySQL
Container


Docker Network
Container
YOURLS
Container
Redis
Cache
Container
MySQL
DB


Cluster
Node
Pod
YOURLS
Container
Pod
Redis
Container
Pod
MySQL
Container


Docker Network
Container
YOURLS
Container
Redis +
Bloom
Container
MySQL
DB

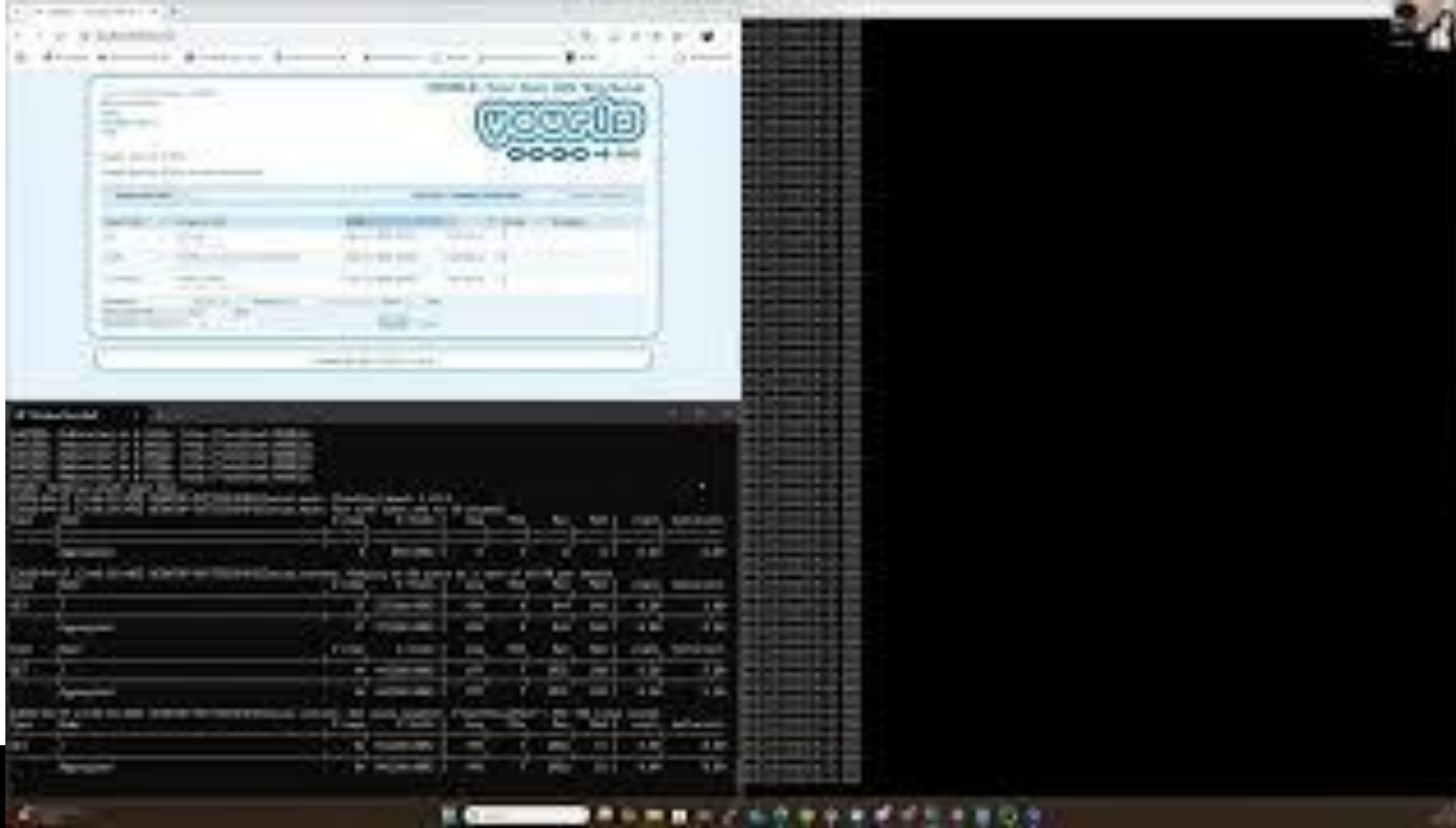

Cluster
Node
Pod
YOURLS
Container
Pod
Redis +
Bloom
Container
Pod
MySQL
Container

Related Work and Unique Selling Point

Prior Research:	USP:
• Kubernetes on Efficiency/ Performance Evaluation (scheduling, automation vs defaults, etc) • Bloom Filter efficiency • Redis Cache: What it is and how it improves usage • Mobile Cloud Computing using Redis caches + Kubernetes, no bloom filter for latency reduction	● Evaluate the performance of the default schedulings and configurations of Kubernetes versus a simple network on an application of small scale. ● Better understand the empirical benefits of Kubernetes rather than the theoretical yields.

Results

Metric	YOURLS Net	Redis Net	Bloom Net	YOURLS k3d	Redis k3d	Bloom k3d
Average Shorten Time (s)	0.3388	0.3704	0.348	0.3469	0.3116	0.3302
Average Redirect Time (s)	0.0158	0.0166	0.0166	0.0319	0.0305	0.0295
Average CPU Usage (%)	12.08	10.82	13.64	18.96	18	18.08
Average Memory Usage (%)	83.5	84.7	88.7	83.4	84.5	91.62
Throughput (requests/sec)	1.47	1.38	1.41	1.42	1.45	1.43



Tie in to the Class

- **Software Evolution to address scalability needs**
- **Refactoring “old” code implementations**
- **Leverages microservices to enhance the performance of the system over time**
- **E-Type Program**
- **AWS Lab 2**

Additional Mentions

- **The project is not open-source, but the GitHub Repo is public here**
- **YOURLS software**
- **Redis software**

Sources

Carrión, C. (2022). Kubernetes scheduling: Taxonomy, ongoing issues and challenges. *ACM Computing Surveys*, *55*(7), 1-37.

Eddelbuettel, D. (2022). A brief introduction to redis. *arXiv preprint arXiv:2203.06559*.

Malhotra, R., Bansal, A., & Kessentini, M. (2024). Deployment and performance monitoring of docker based federated learning framework for software defect prediction. *Cluster Computing*, *27*(5), 6039-6057.

Patel, J., & Halabi, T. (2021). *Optimizing the performance of web applications in mobile cloud computing*. In *2021 IEEE 6th International Conference on Smart Cloud (SmartCloud)* (pp. 1-6). IEEE. https://doi.org/10.1109/SmartCloud52277.2021.00013

Poniszewska-Marańda, A., & Czechowska, E. (2021). Kubernetes cluster for automating software production environment. *Sensors*, *21*(5), 1910.

S. Tarkoma, C. E. Rothenberg and E. Lagerspetz, "Theory and Practice of Bloom Filters for Distributed Systems," in *IEEE Communications Surveys & Tutorials*, vol. 14, no. 1, pp. 131-155, First Quarter 2012, doi: 10.1109/SURV.2011.031611.00024. keywords: {Probabilistic logic;Arrays;Fingerprint recognition;Filtering theory;Peer to peer computing;Bismuth;Bloom filters;probabilistic structures;distributed systems},

Vayghan, L. A., Saied, M. A., Toeroe, M., & Khendek, F. (2018, July). Deploying microservice based applications with kubernetes: Experiments and lessons learned. In *2018 IEEE 11th international conference on cloud computing (CLOUD)* (pp. 970-973). IEEE.

Xu, A. (2020). *System design interview: An insider's guide (Vol. 1)*. Independently published.

chatGPT

University of Colorado
Colorado Springs

University of Colorado

Boulder | Colorado Springs | Denver | Anschutz Medical Campus