Automatic Management of Cloud Resources for Distributed Systems Experiments

Chunrui Huang[1 2], David Chu[3], Joseph M. Hellerstein[3]
[1]Department of Computer Science, City College of San Francisco
[2]Transfer-to-Excellence Research Experience for Undergraduates, University of California, Berkeley
[3]Department of Electrical Engineering and Computer Sciences, University of California, Berkeley

2022 Transfer-to-Excellence Research Experiences for Undergraduates Program (TTE REU Program)

Abstract

Existing distributed protocols cannot effectively make use of additional machines and are hard to redesign due to hidden complexities [1]. As a part of the optimizer experiments for distributed system, we create a pipeline of Python scripts to automatically interact with Amazon Web Services, allowing for simple changes in deployment parameters, automating common tasks such as data retrieval for analysis and log generation for debugging. It will simplify the process of testing and allow for rapid prototyping.

Background

- Distributed systems (cloud computing) is important

  Throughput | Latency | Fault tolerance | Autoscaling

- Common problems in distributed systems

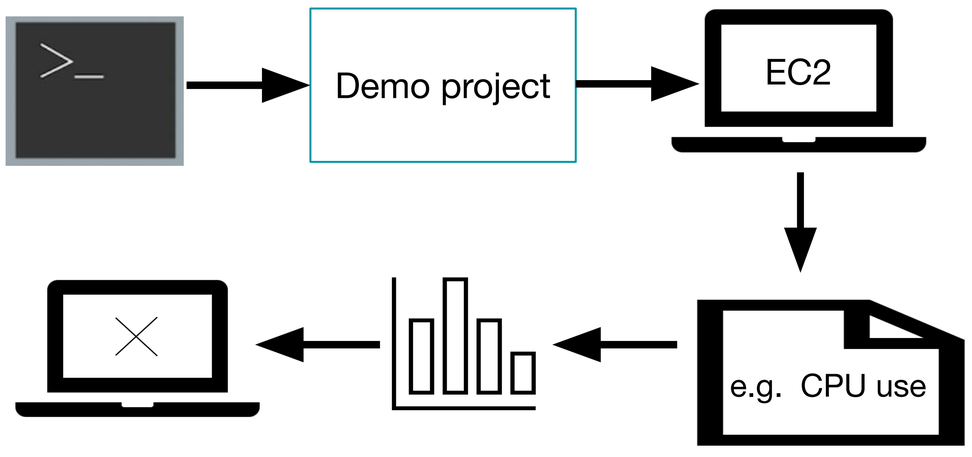
  Failures | Scaling → e.g. more/fewer machines?

- Therefore, we do research to make it better!
- Why deploying these research projects is hard:
  1. Bugs are hard to replicate.
  2. Hard to debug for remote computer
  3. Profiling can change behavior.
- **Project goal: Automate deployment and profiling of distributed system experiments**

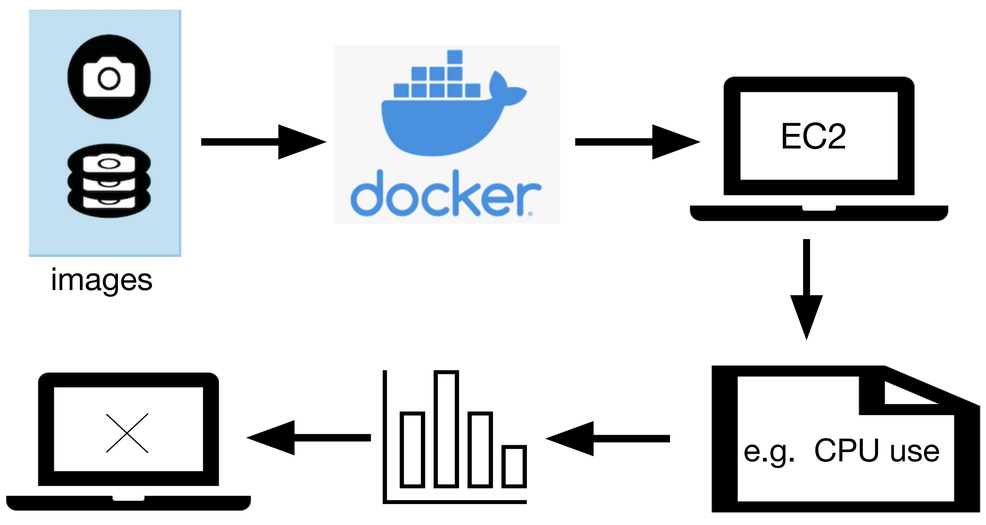
Methods

We interact with Amazon Elastic Compute Cloud (EC2), a cloud platform, in two ways.

- 1



- 2



Results

A pipeline automatically completes the following tasks:

1. Launch EC2 instance
2. Connect to EC2 instance
3. Install related applications/packages
4. Run the demo program
5. Collect metrics
6. Generate graphs
7. Shut down instance

Demo Project: Countdown Timer

```
Enter the time in seconds: 20
00:19
```

Fig. 1 Screen shot of the demo project

Collect Metric Statistics


```
{
    "Datapoints": [
        {
            "Timestamp": "2022-08-04T23:46:00Z",
            "Average": 0.1005649717514164,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2022-08-04T23:56:00Z",
            "Average": 0.1333518569973064,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2022-08-04T23:51:00Z",
            "Average": 0.16559229415578378,
            "Unit": "Percent"
        }
    ],
    "Label": "CPUUtilization"
}
```


Fig. 2 Example of metric statistics

Generate Graphs

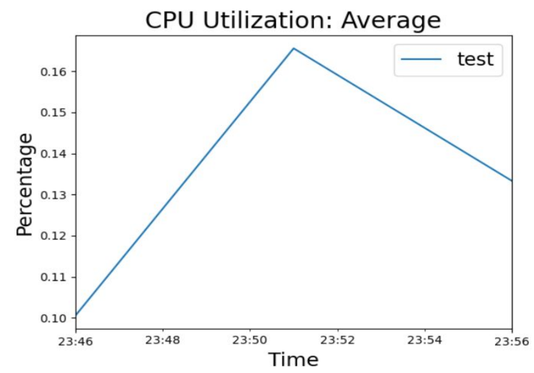


Fig. 3 Graph of CPU utilization

The percentage shows the proportion of the compute units that are currently in use on the instance.

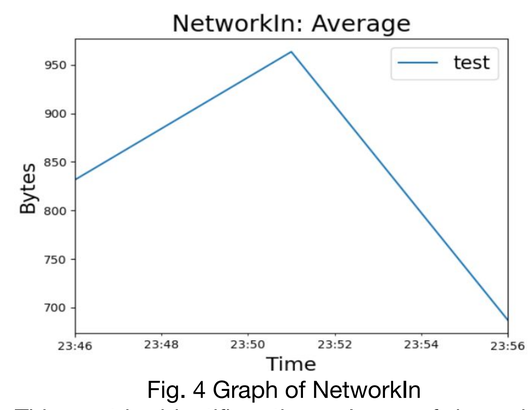


Fig. 4 Graph of NetworkIn

This metric identifies the volume of incoming network traffic to a single instance.

Discussion

The pipeline works for small projects that don't rely on multiple libraries. For complicated projects, additional libraries and dependencies need to be added manually to EC2 accordingly.

The pipeline can generate graphs of selected metric statistics. Future study will expand the function of the pipeline to collect all metric statistics.

References

[1] M. Whittaker et al., "Scaling replicated state machines with compartmentalization," Proc. VLDB Endow., vol. 14, no. 11, pp. 2203–2215, Jul. 2021, doi: 10.14778/3476249.3476273.

Acknowledgements

Thank you to my mentor David Chu for his patient guidance and technical support for this project. Thank you, Nicole McIntyre, Tony Vo Hoang, and Naz Khairallah for your exceptional administrative support and advice throughout the Transfer-to-Excellence Summer Research Program.

Contact Information

Chunrui (Corinne) Huang

Email: chuan215@mail.ccsf.edu

Support Information

This work was funded by Transfer-to-Excellence Summer Research Program at the University of California, Berkeley and the Hopper Dean Foundation.