Advisor: Kunwadee Sripanidkulchai

Natawat Kwanpum
5931017021

Final Presentation

Container Live Migration Using PageServer

Across cloud providers

Agenda

Introduction

Background

System Design

Evaluation

Implications

Conclusions

Cloud computing?

Cloud computing is the on-demand availability of computer system resources, especially data storage and computing power, without direct active management by the user.

https://en.wikipedia.org/wiki/Cloud_computing

Container?

“They are a type of virtualization technology, with their own CPU, memory and resources like a virtual machine. The difference, though, is that containers share the kernel (the brain) of the host operating system and don’t need a guest operating system.”

https://www.otava.com/blog/containers-and-docker-in-a-nutshell/

Why do we use the container?

- Isolation an application and its dependencies
- More efficient use of system resources (OS overhead)
- Portability
- Better application development

Deployment Model

IaaS

Virtual Machine

+

Container runtime

docker

PaaS

Amazon ECS

Cloud Run

Azure Container Instances

docker

Microsoft

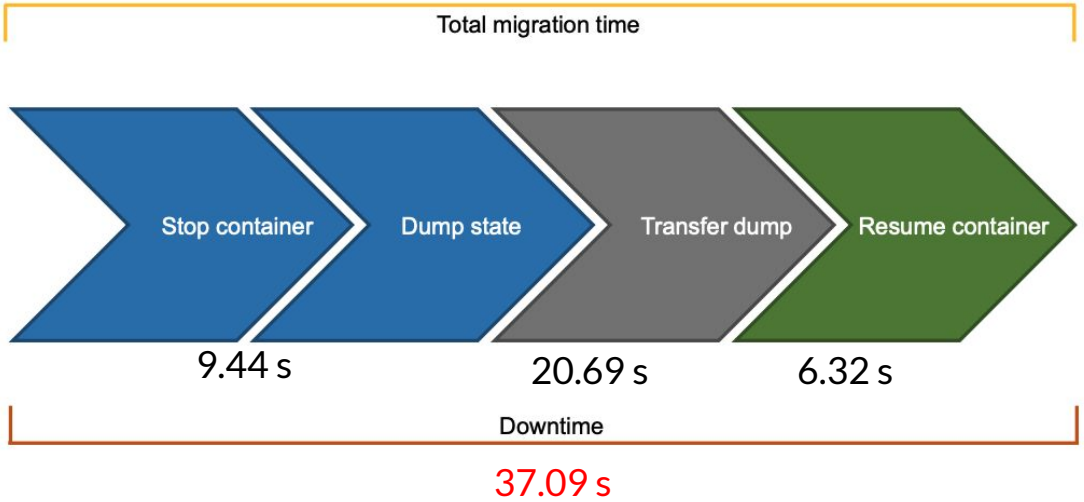
Why do we need container live migration?

Background

Live Migration?

Live migration refers to the process of moving a running application between different physical machines without disconnecting the client. Memory, storage, and network connectivity of the virtual machine are transferred from the original guest machine to the destination

Basic migration



1. **Checkpoint** : **Stop** application and **dump memory states** to files
2. **Transfer**: Copy checkpoint files to destination machine
3. **Restore** from checkpoint files and start application again

Downtime = service unavailable

More memory = More Downtime

“Let’s decrease downtime!”

Project Objectives

01 | To develop process of migration the containers across the cloud providers with low downtime

02 | To develop tools to facilitate migration process

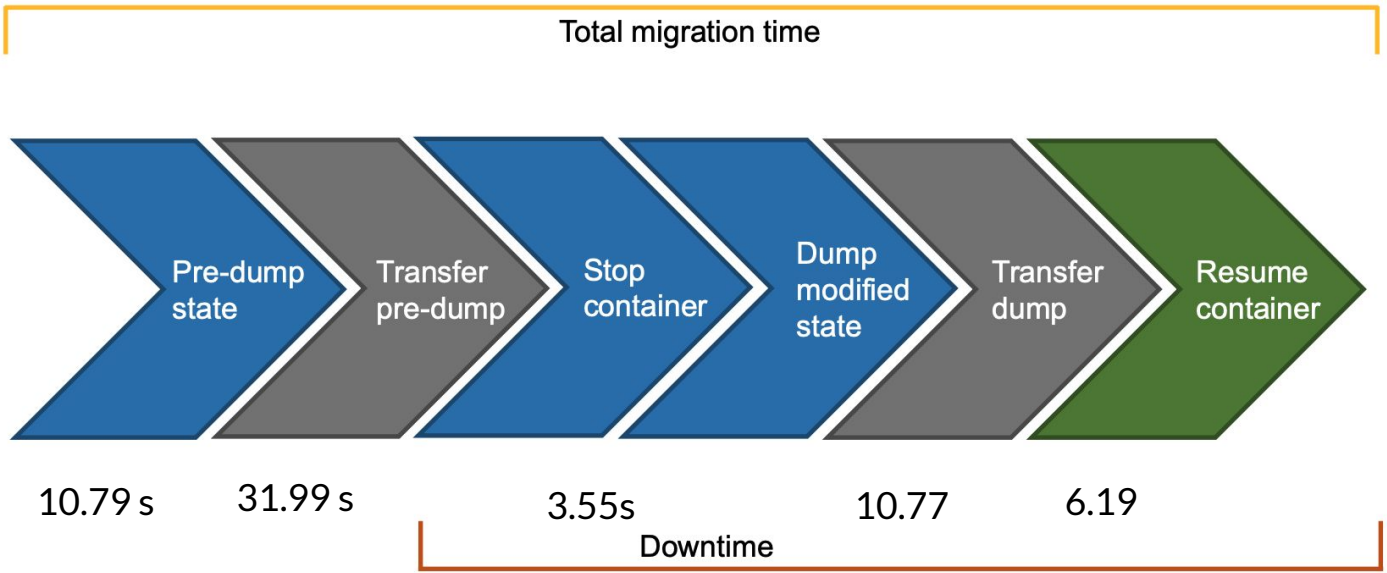
How to decrease downtime?

1. **Pre-copy migration**
   Predump -> Precopy -> Checkpoint -> Transfer -> Restore

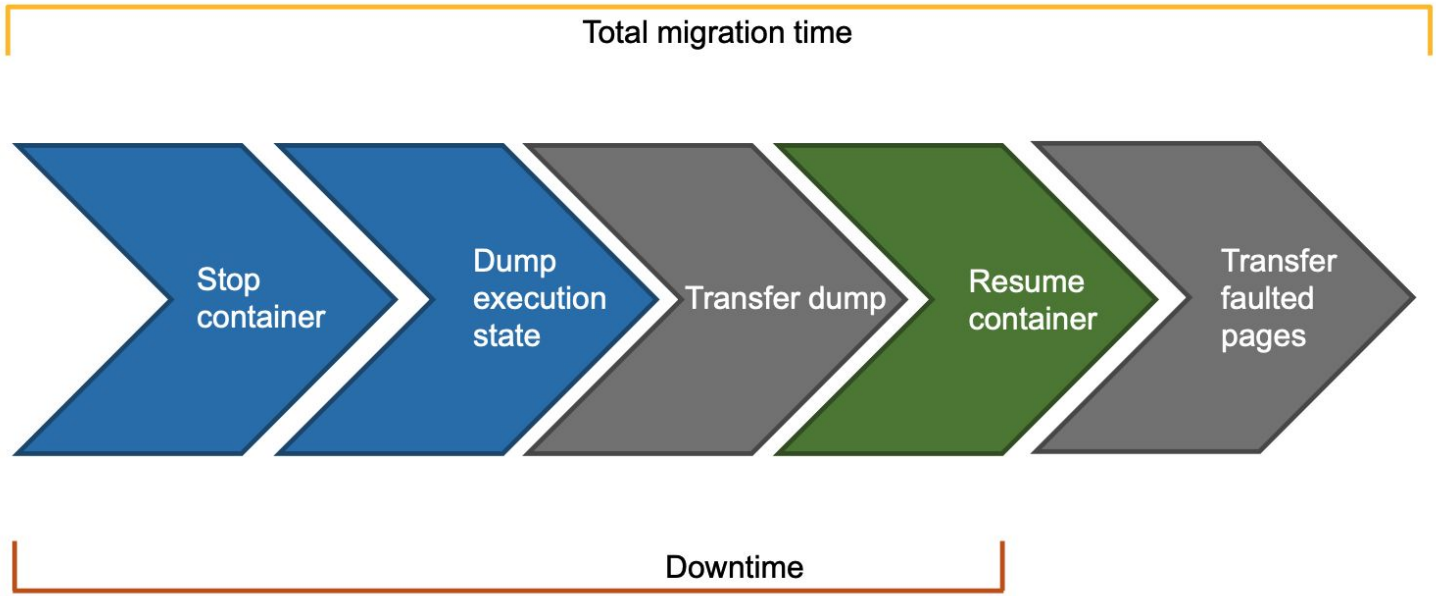
2. **Post-copy Migration (Lazy Migration)**
   Checkpoint (the minimal task state) -> Transfer -> Restore -> transfer faulted page

Pre-copy Migration

Downtime: 20.98



Post-copy Migration



Pageserver

“The component that allows to copy (rather than dump) user memory to a destination system during the course of live migration"

Normal Migration

BEFORE

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt labore dolore magna aliqua. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Client Implications:
Consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Lorem ipsum dolor sit amet, consectetur adipiscing elit tempor incididunt ut labore et dolore magna aliqua.

Diskless Migration (using Pageserver)

AFTER

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt labore dolore magna aliqua. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Client Implications:
Consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Lorem ipsum dolor sit amet, consectetur adipiscing elit tempor incididunt ut labore et dolore magna aliqua.

Proof of concept

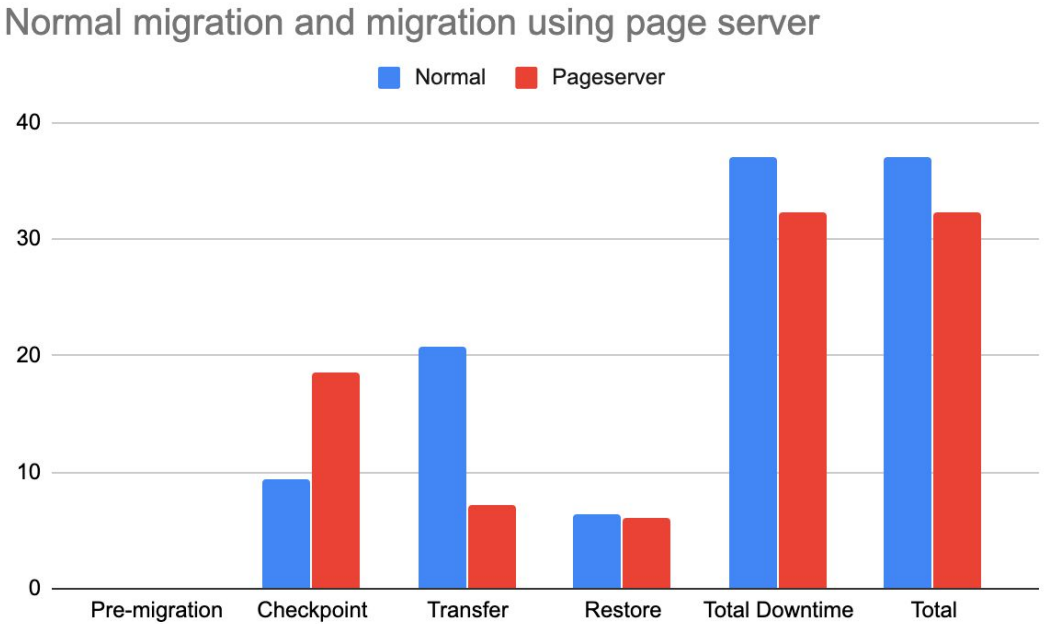
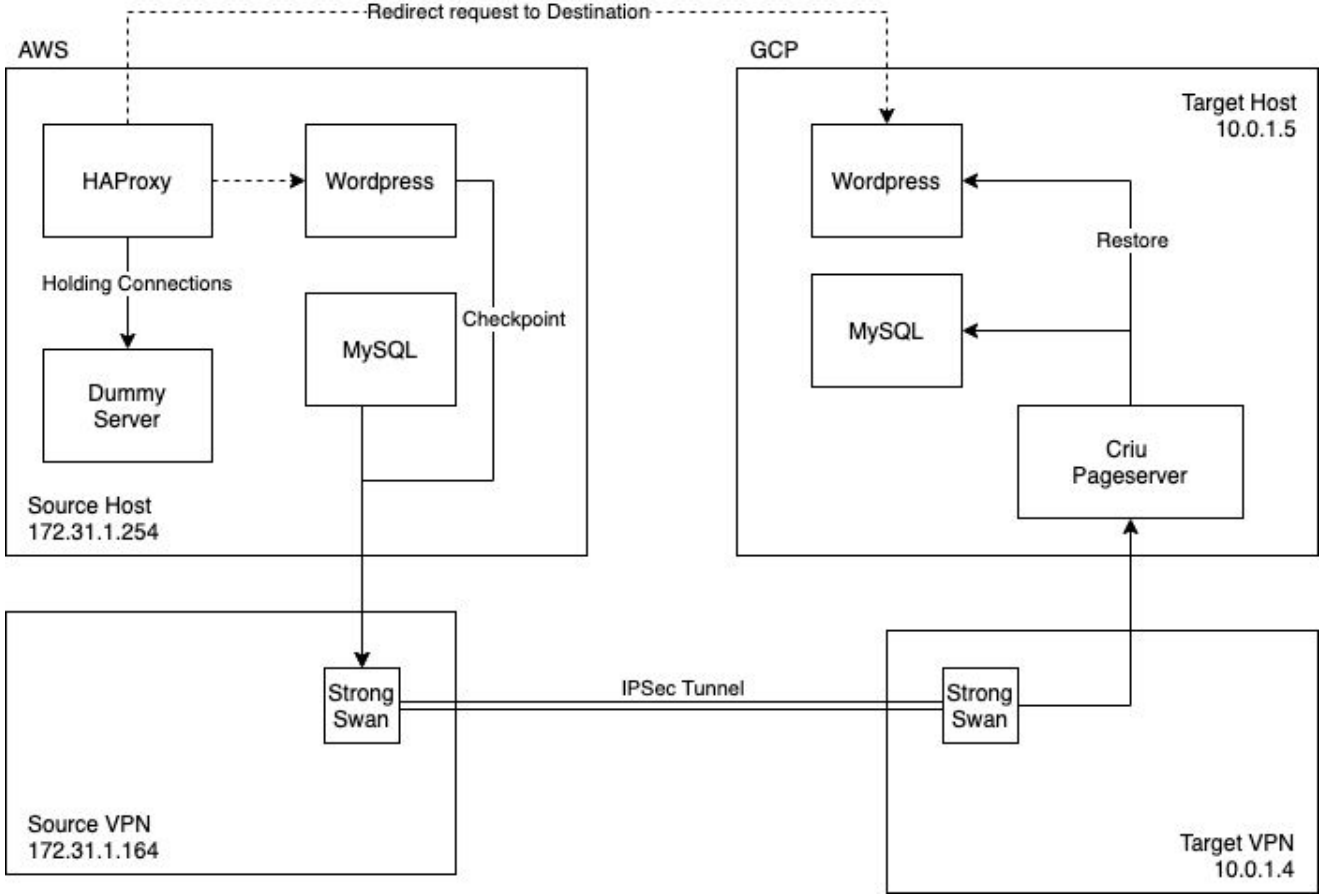


TABLE II
EVALUATION RESULT

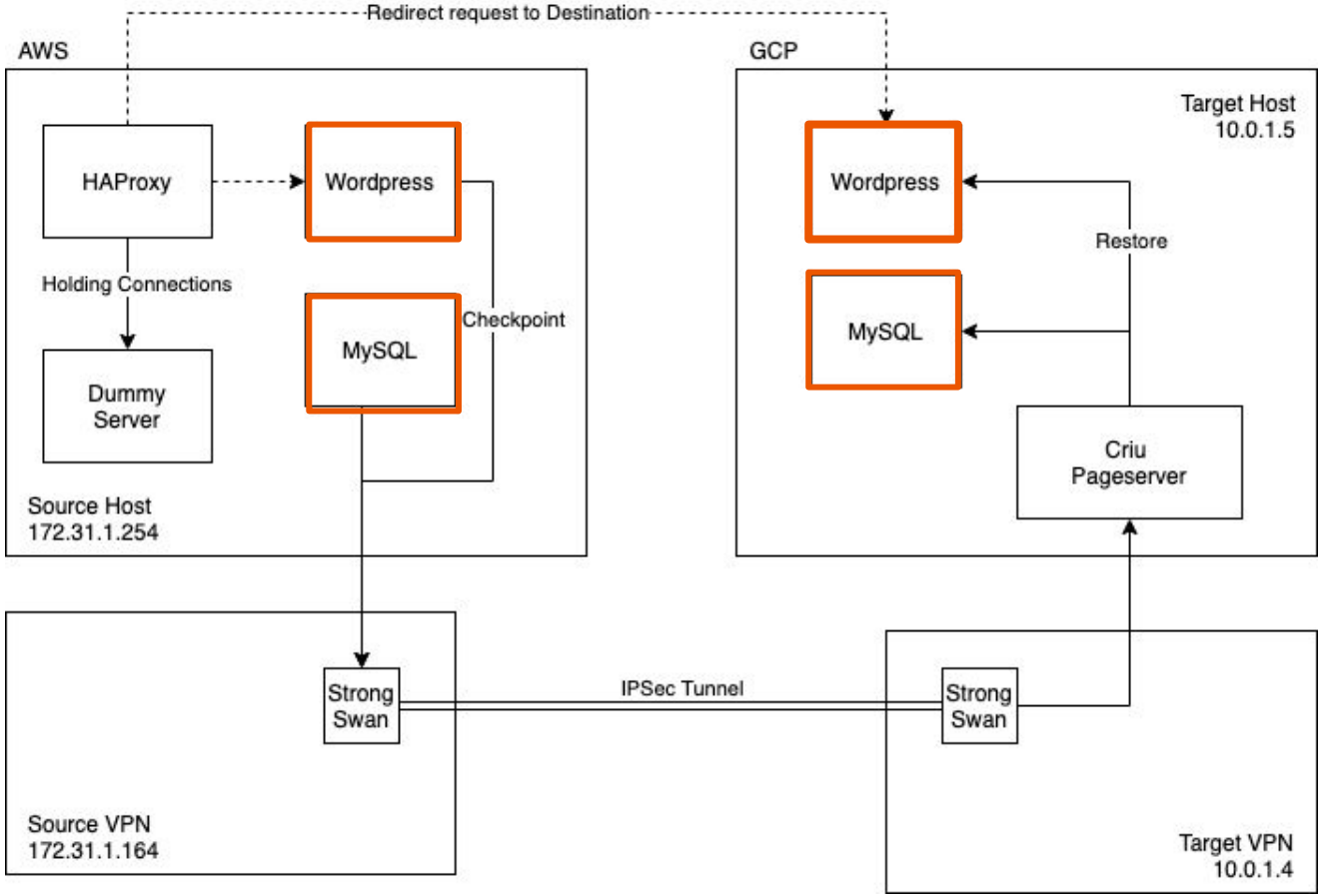
Method	Scenario	Pre-dump	Pre-copy	Pre-migration	Checkpoint	Transfer	Restore	Downtime	Total Time
Normal	400c	-	-	-	9.44	20.69	6.32	37.09	37.09
Page Server	400c	-	-	-	18.62	7.16	6.01	32.23	32.23


System Design

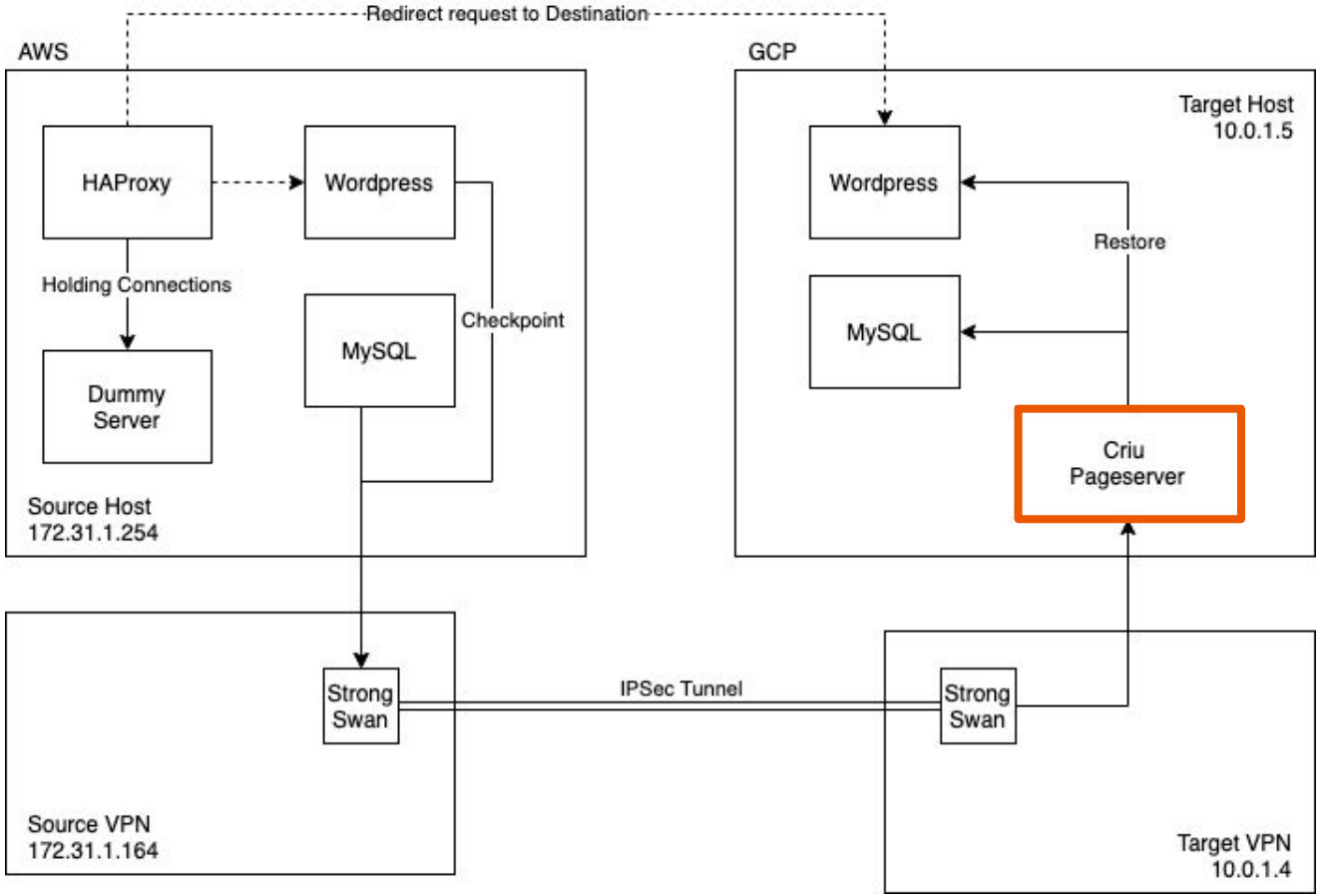
System overview



Containers

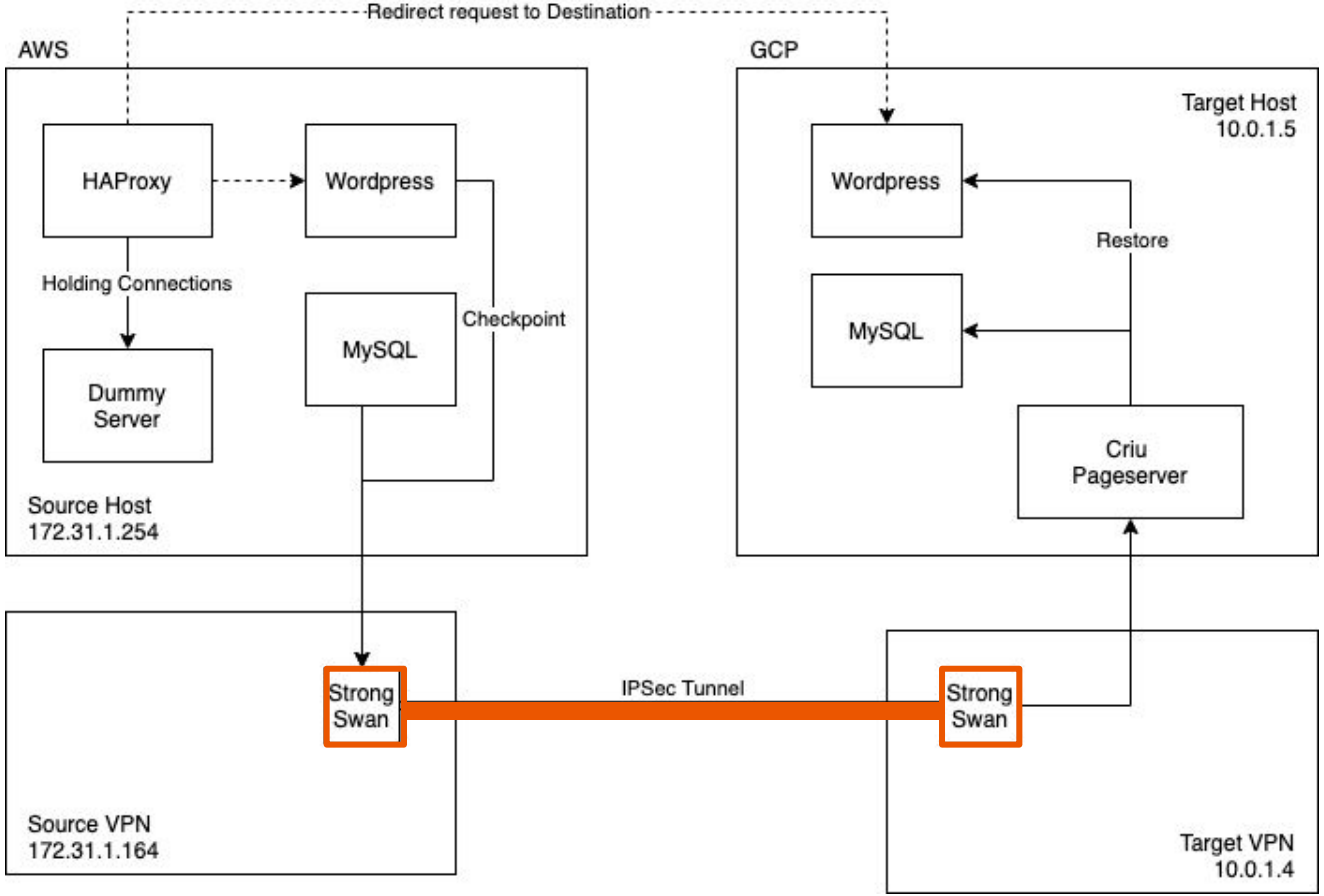

Redirect request to Destination
AWS
GCP
HAProxy
Wordpress
Holding Connections
MySQL
Checkpoint
Dummy Server
Source Host
172.31.1.254
Target Host
10.0.1.5
Wordpress
Restore
MySQL
Criu Pageserver
Strong Swan
IPSec Tunnel
Strong Swan
Source VPN
172.31.1.164
Target VPN
10.0.1.4

Page server


Redirect request to Destination
AWS
GCP
HAProxy
Wordpress
Holding Connections
Dummy Server
MySQL
Checkpoint
Source Host
172.31.1.254
Target Host
10.0.1.5
Wordpress
Restore
MySQL
Criu Pageserver
Strong Swan
IPSec Tunnel
Strong Swan
Source VPN
172.31.1.164
Target VPN
10.0.1.4

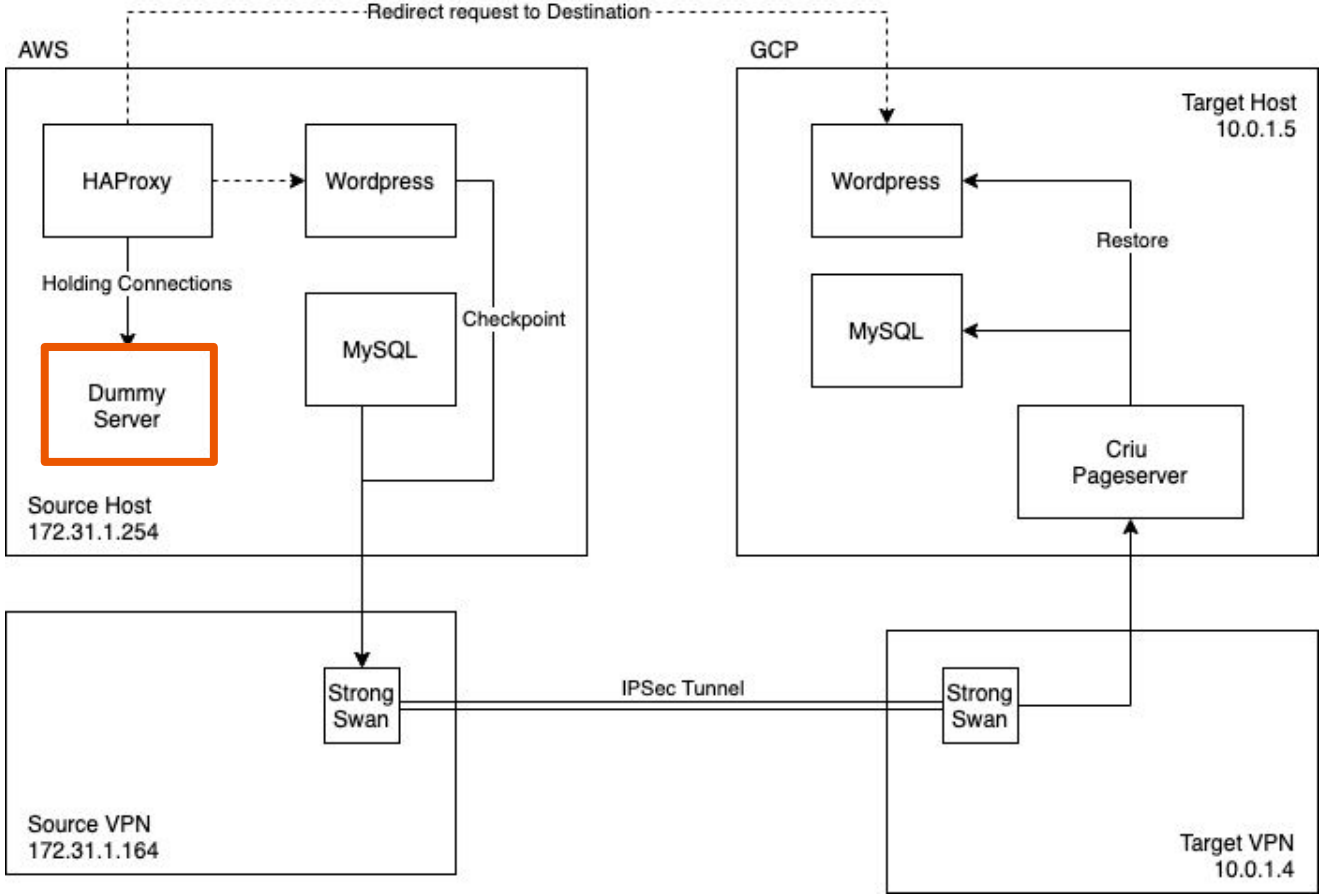
VPN

"Secure
Data Transfer"



Dummy Server

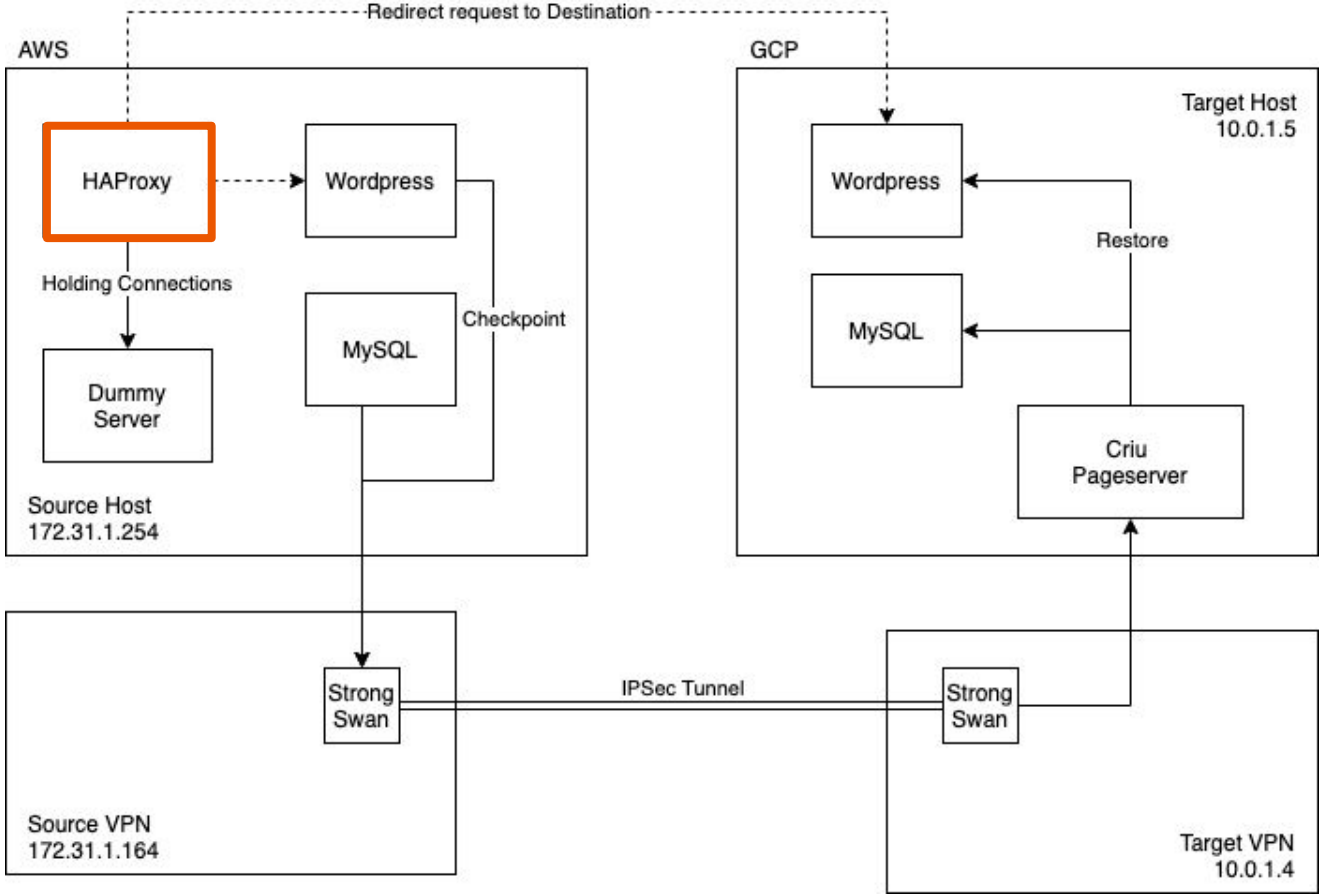
"Conceal
Service unavailable"



HAProxy

“Load balancer”

“Forward requests”



Automated Script

Evaluation

Evaluation Goal

“Ensure that pageserver can help to minimize downtime”

Application & instance size

Application
- Wordpress as a web server
- MySQL as a database

Source Instances
- AWS EC2
- Size t3.medium (2 vCPUs, 4 GB RAM)
- Region: ap-northeast-1 (Tokyo)

Destination Instances
- GCP Compute engine
- Size: n1-standard-1 (1 vCPU, 3.75 GB RAM)
- Region: asia-northeast-a (Tokyo)

Workload Simulator

We use **Siege** to generate request to web server (wordpress)

- Run each evaluation 10 times and use mean as a result
- Add workload in different scenario in term of concurrent connections
  - 0, 1, 5, 10, 50, 100, 200, 400

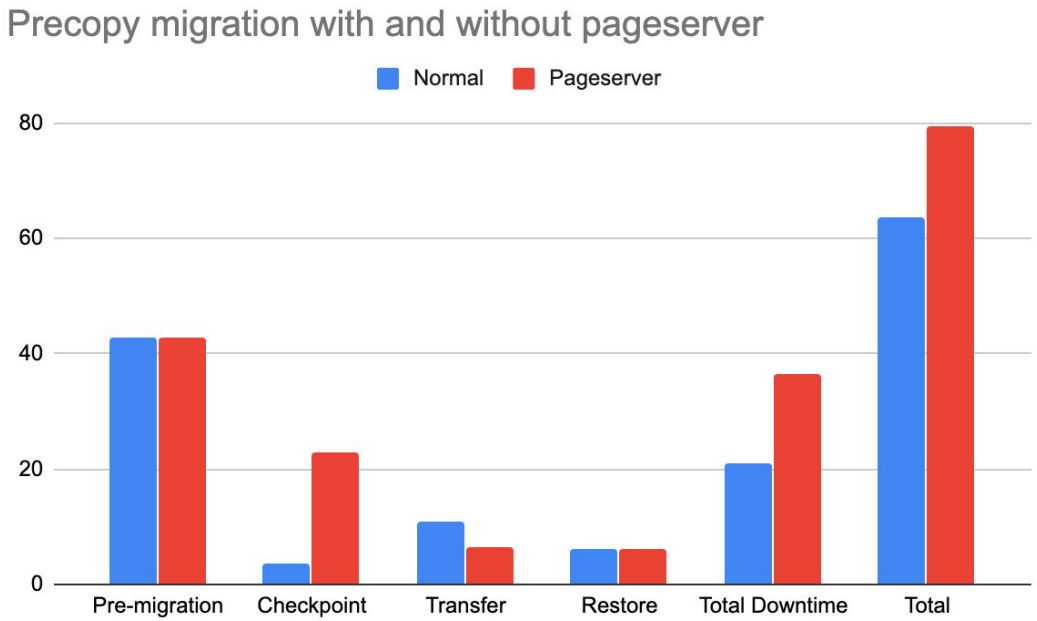
https://www.joedog.org/siege-home/

Result

Pre-copy = 20.98 s
Pageserver = 21.15 s

Rsync optimization = 18.27 s
Dummy server = 12.29 s

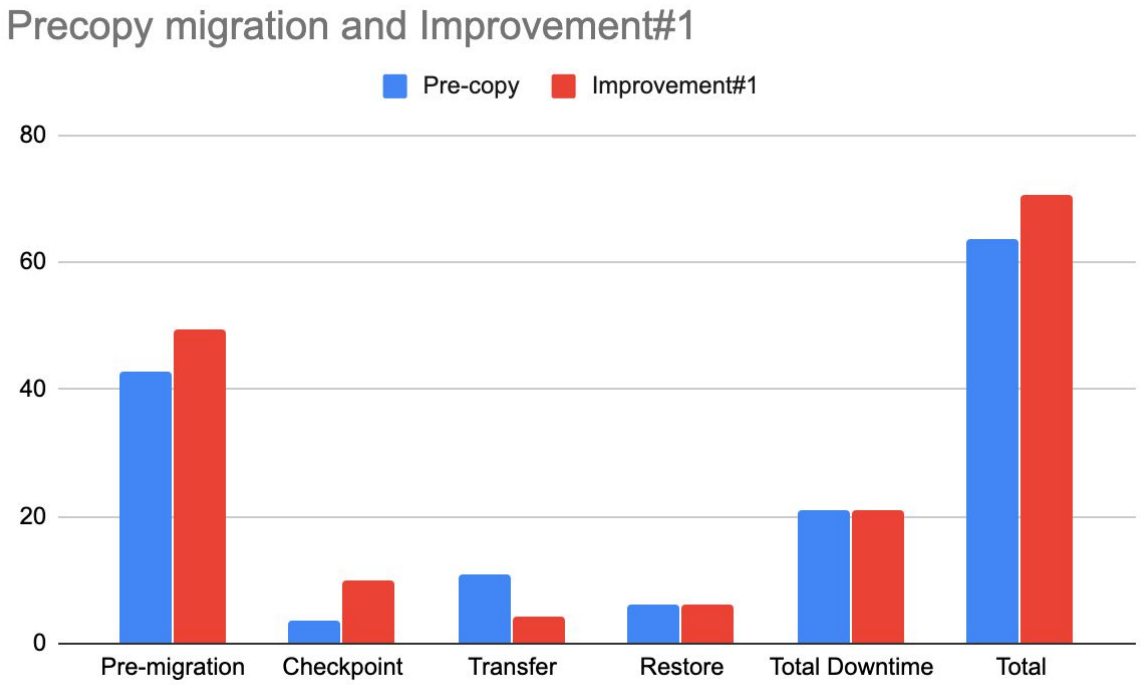
Pre-copy and Page server



experiment	scenario	predump	precopy	premigration	checkpoint	transfer	restore	downtime	totaltime
precopy	400	10.794	31.994	42.788	3.547	10.774	6.194	20.980	63.768
precopy-pageserver	400	10.943	31.966	42.909	22.935	6.406	6.141	36.549	79.458

Pre-copy Pageserver Improve#1

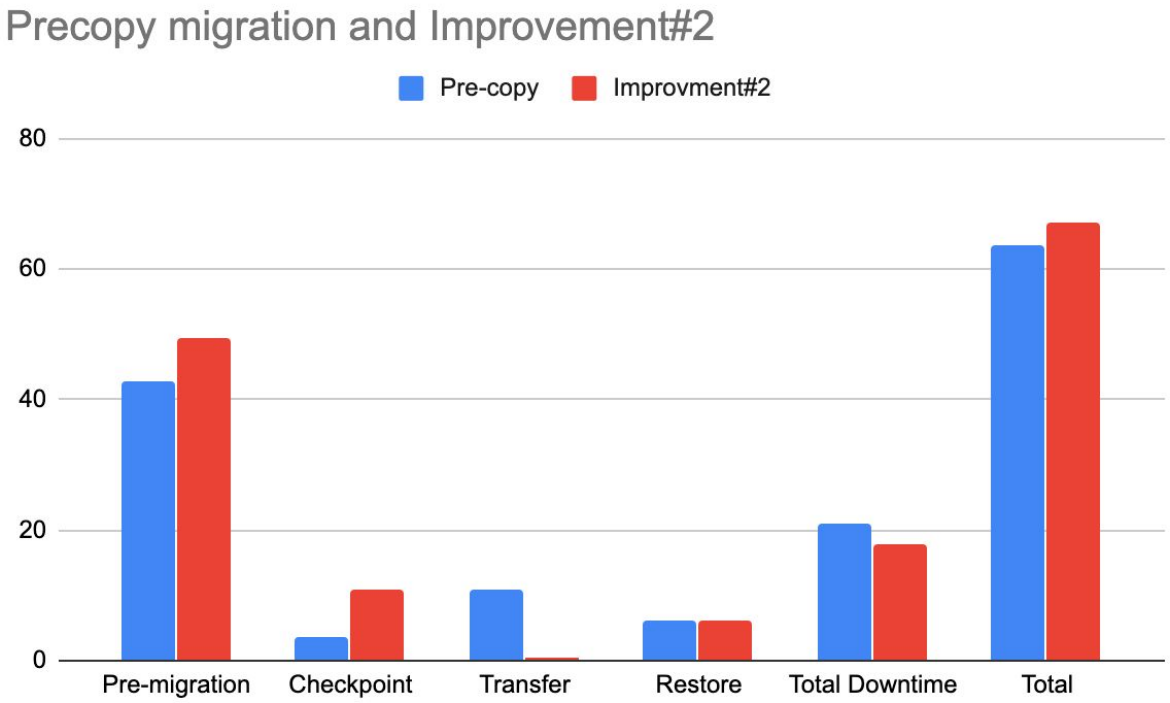
- Use previous images



experiment	scenario	predump	precopy	premigration	checkpoint	transfer	restore	downtime	totaltime
precopy	400	10.794	31.994	42.788	3.547	10.774	6.194	20.980	63.768
test-pageserver-prev	400	21.009	28.384	49.393	10.090	4.351	6.129	21.152	70.545

Pre-copy Pageserver Improve#2

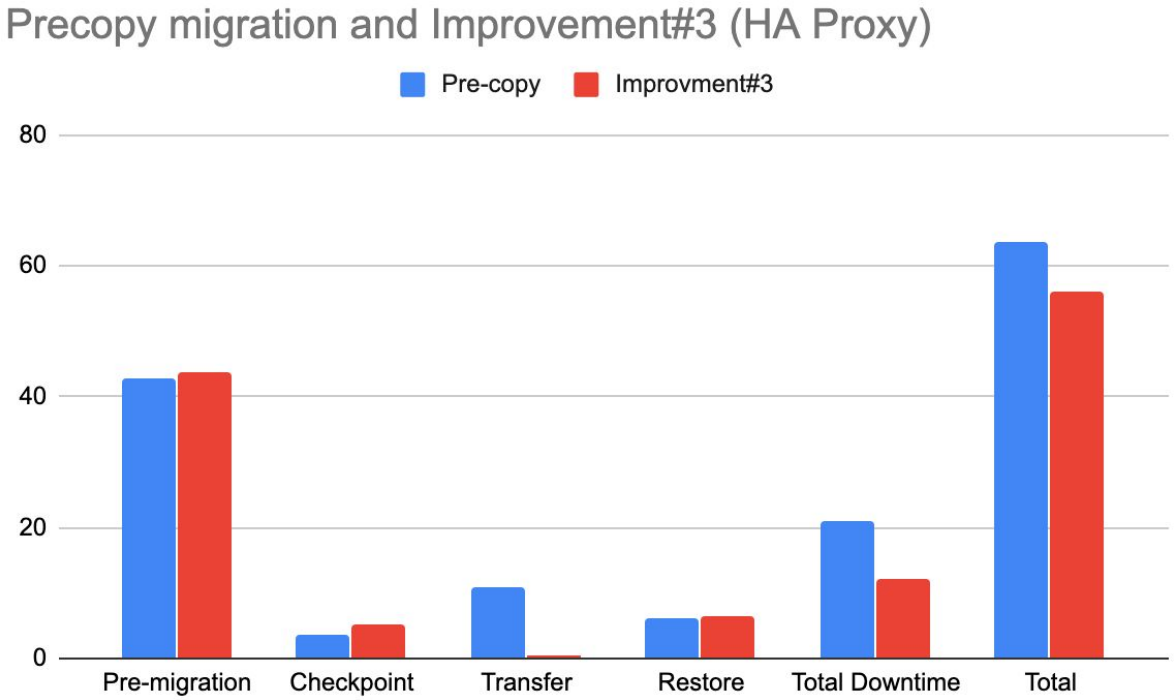
- Rsync optimization



experiment	scenario	predump	precopy	premigration	checkpoint	transfer	restore	downtime	totaltime
precopy	400	10.794	31.994	42.788	3.547	10.774	6.194	20.980	63.768
test-pageserver-prev-mysql	400	20.504	28.839	49.343	10.872	0.386	6.070	17.905	67.248

Pre-copy Pageserver Improve#3

- Hold traffic with HAproxy



experiment	scenario	predump	precopy	premigration	checkpoint	transfer	restore	downtime	totaltime
precopy	400	10.794	31.994	42.788	3.547	10.774	6.194	20.980	63.768
precopy-pageserver-haproxy	400	18.220	25.607	43.827	5.102	0.328	6.451	12.294	56.121

Final Results

TABLE II
EVALUATION RESULT

Method	Scenario	Pre-dump	Pre-copy	Pre-migration	Checkpoint	Transfer	Restore	Downtime	Total Time
Normal	400c	-	-	-	9.44	20.69	6.32	37.09	37.09
Page Server	400c	-	-	-	18.62	7.16	6.01	32.23	32.23
Pre-copy	400c	10.79	31.99	42.79	3.55	10.77	6.19	20.98	63.77
PC&PS	400c	21.01	28.38	49.39	10.09	4.35	6.13	21.15	70.54
PC & PS & Rsync	400c	20.63	28.96	49.59	11.24	0.39	6.05	18.27	67.87
PC & PS & HAProxy	400c	18.22	25.61	43.83	5.10	0.33	6.45	12.29	56.12

[a]PC is refer to pre-copy and PS is refer to page server.

Conclusions

“Page server can help to minimize downtime
by reducing writing and reading files from disk which are
overhead of migration”

Demo time!!

Thank you
Q&A