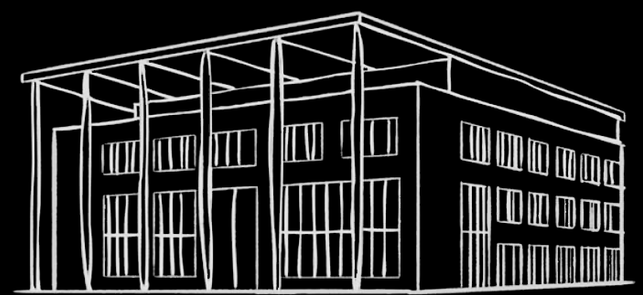
Cloud Computing

Data Science & Analytics Club
10/29/2025

NEAL MAGEE
SCHOOL OF DATA SCIENCE
UNIVERSITY OF VIRGINIA



UVA Data Science

Introduction & Concepts

Who am I?

Neal Magee

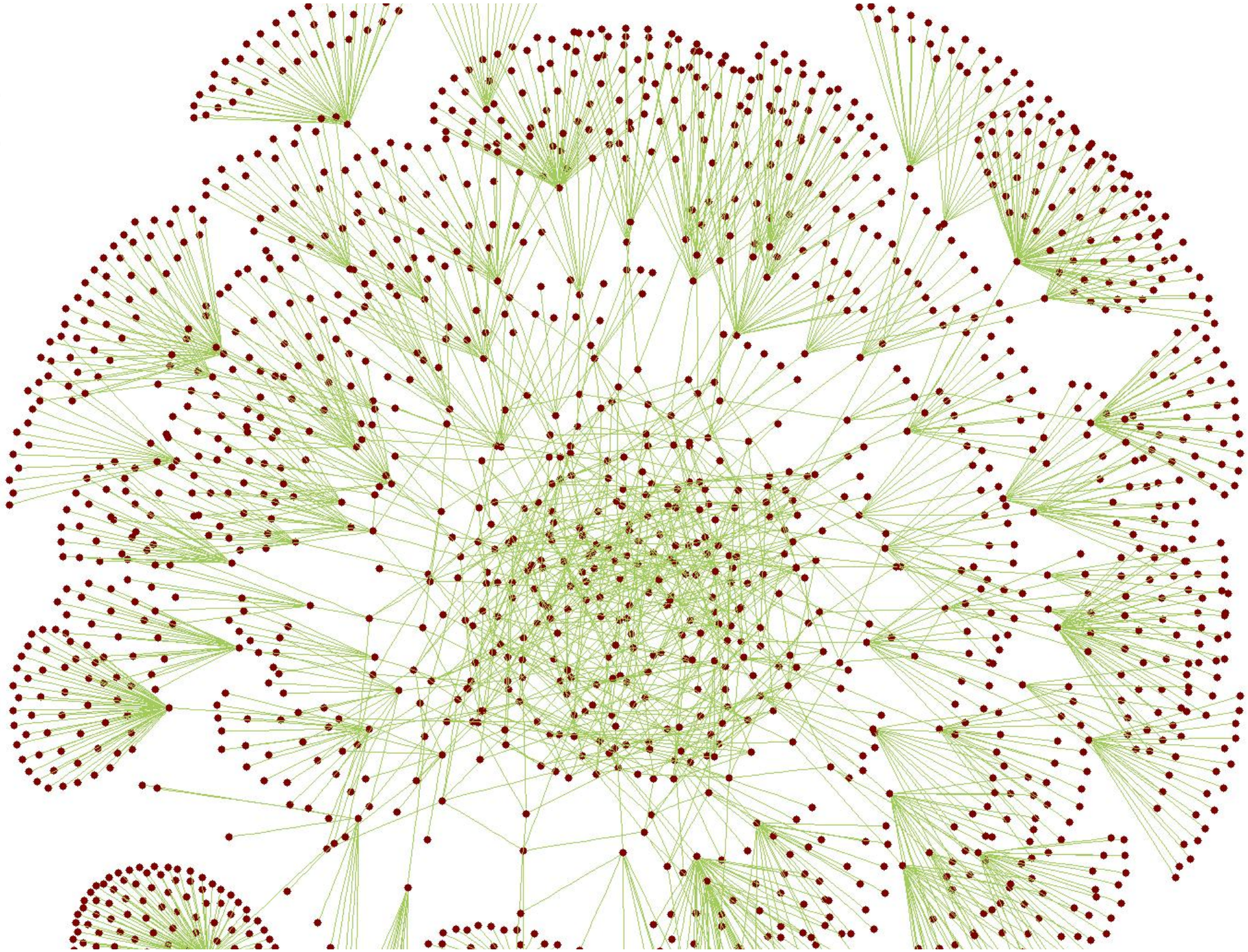
Associate Professor, UVA School of Data Science
Solution Architect, UVA Research Computing
Lead, Internet Operations 2007-2017
.NET/C# Developer 2004-2007
HTML/CGI Developer 1990s

Certifications

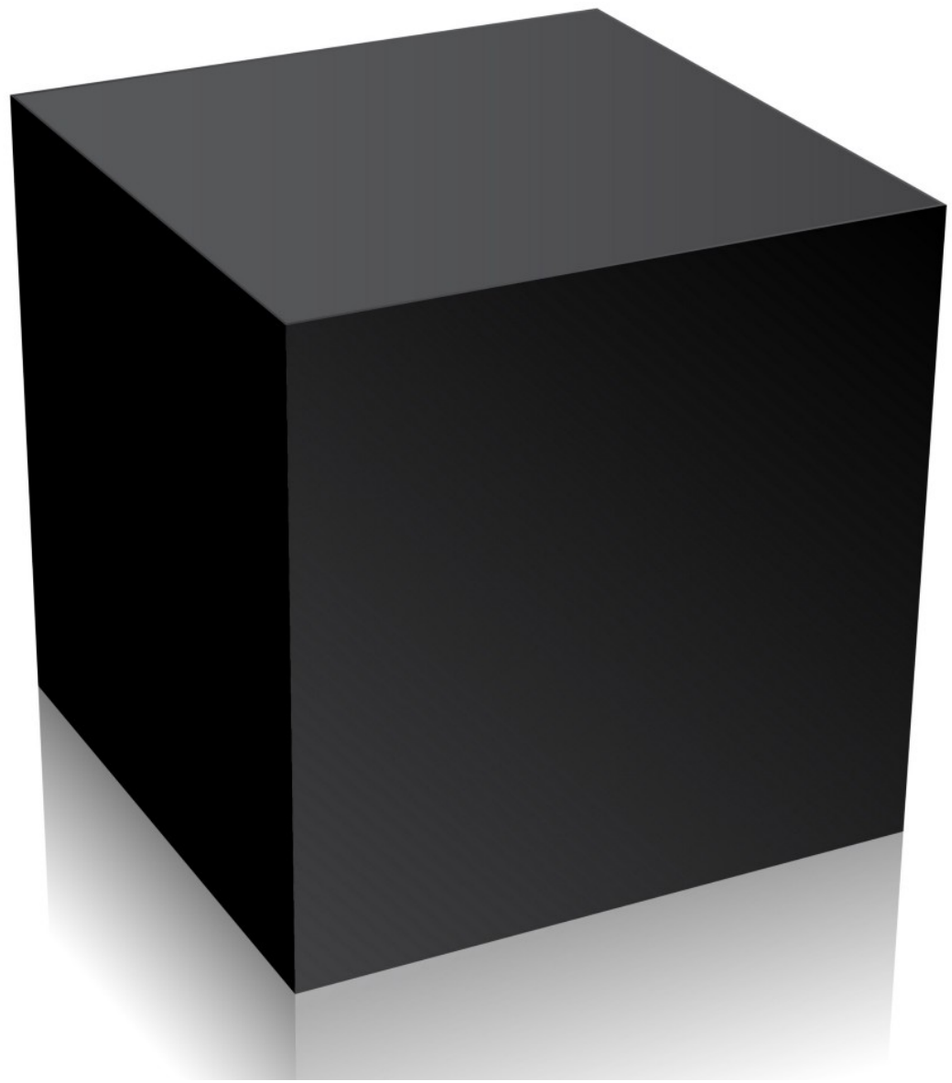
AWS Solution Architect Professional
AWS Solution Architect Associate
AWS Developer Associate

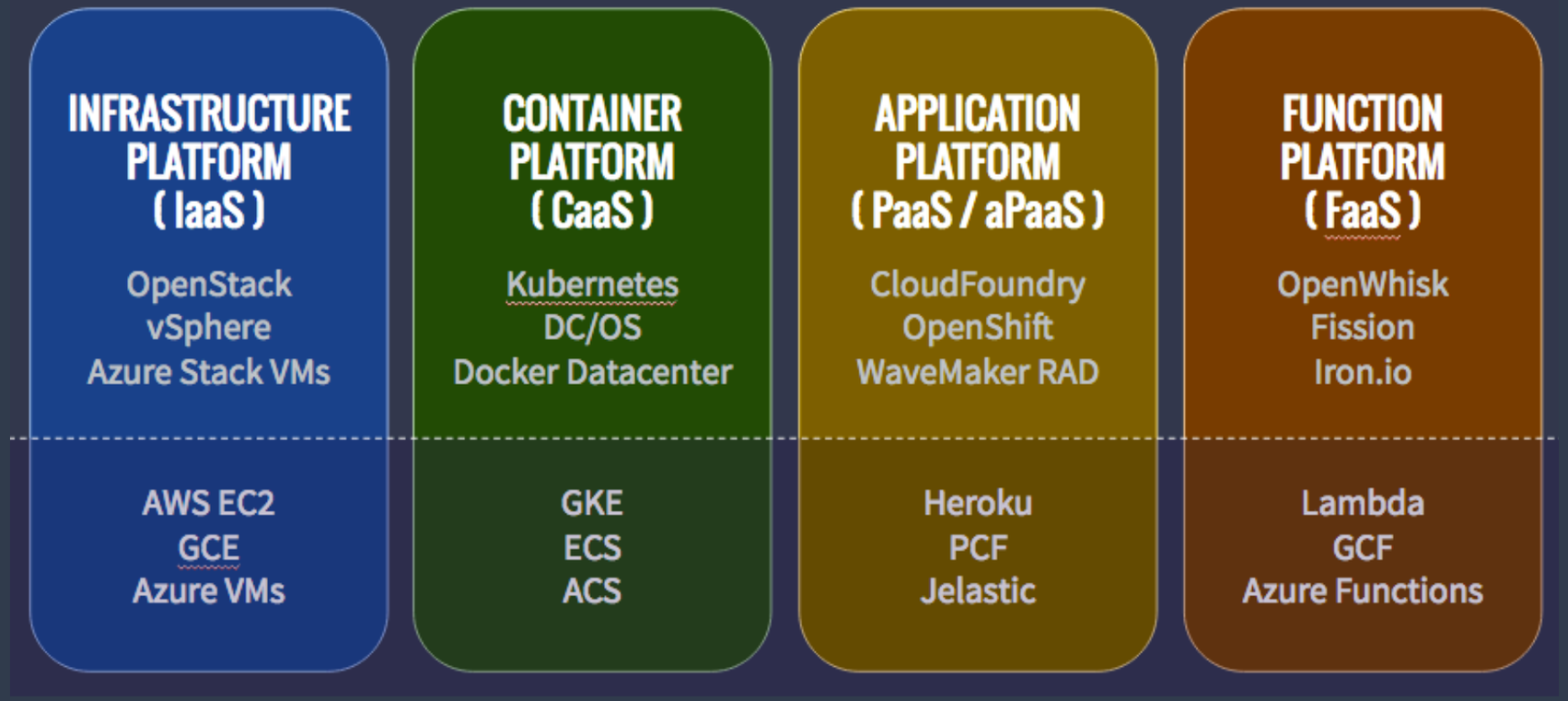
What is the cloud?

Internet



Cloud Computing

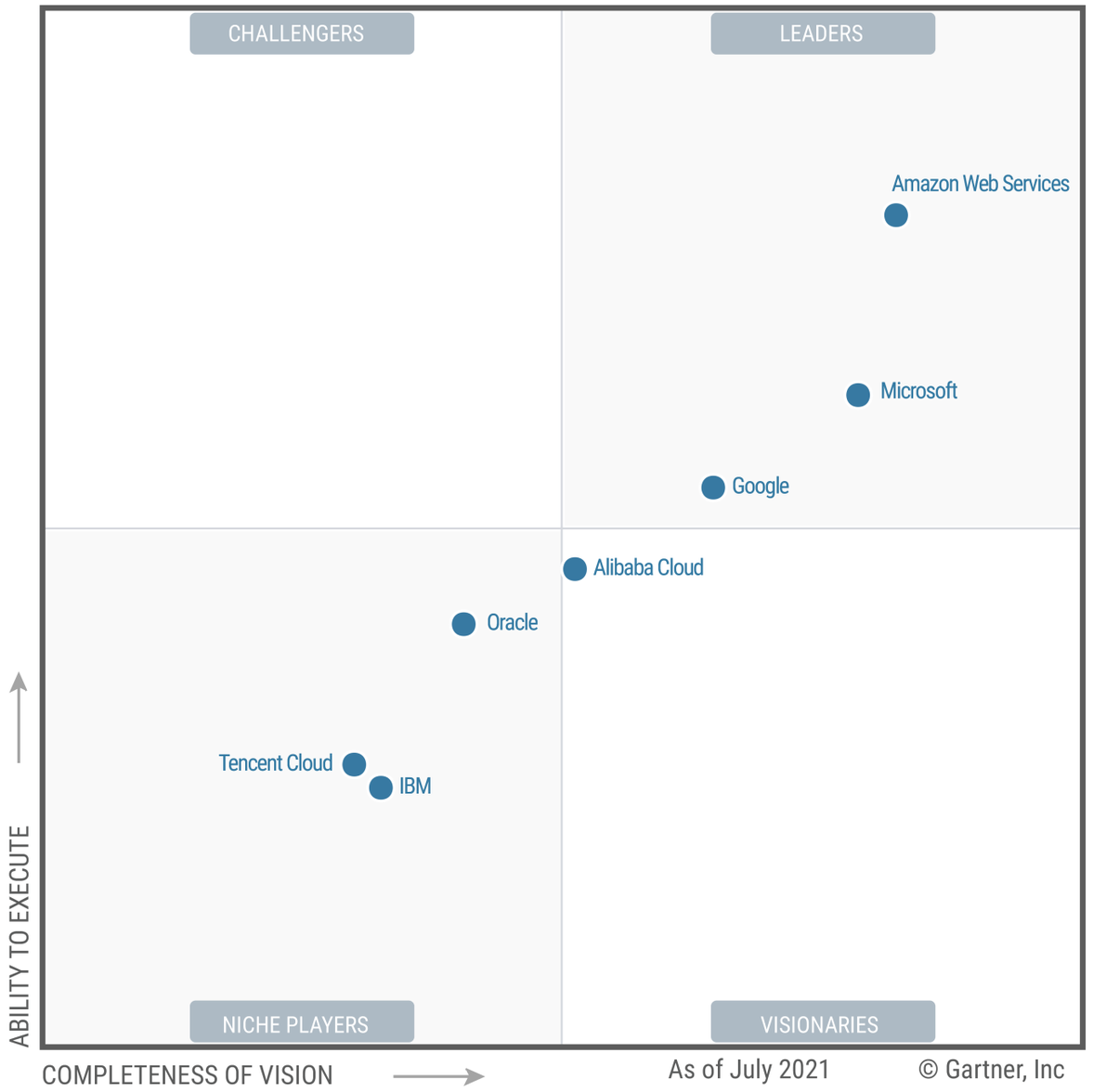



INFRASTRUCTURE PLATFORM (IaaS)
OpenStack
vSphere
Azure Stack VMs
AWS EC2
GCE
Azure VMs
CONTAINER PLATFORM (CaaS)
Kubernetes
DC/OS
Docker Datacenter
GKE
ECS
ACS
APPLICATION PLATFORM (PaaS / aPaaS)
CloudFoundry
OpenShift
WaveMaker RAD
Heroku
PCF
Jelastic
FUNCTION PLATFORM (FaaS)
OpenWhisk
Fission
Iron.io
Lambda
GCF
Azure Functions

Public Clouds

CLOUD INFRASTRUCTURE

Figure 1: Magic Quadrant for Cloud Infrastructure and Platform Services

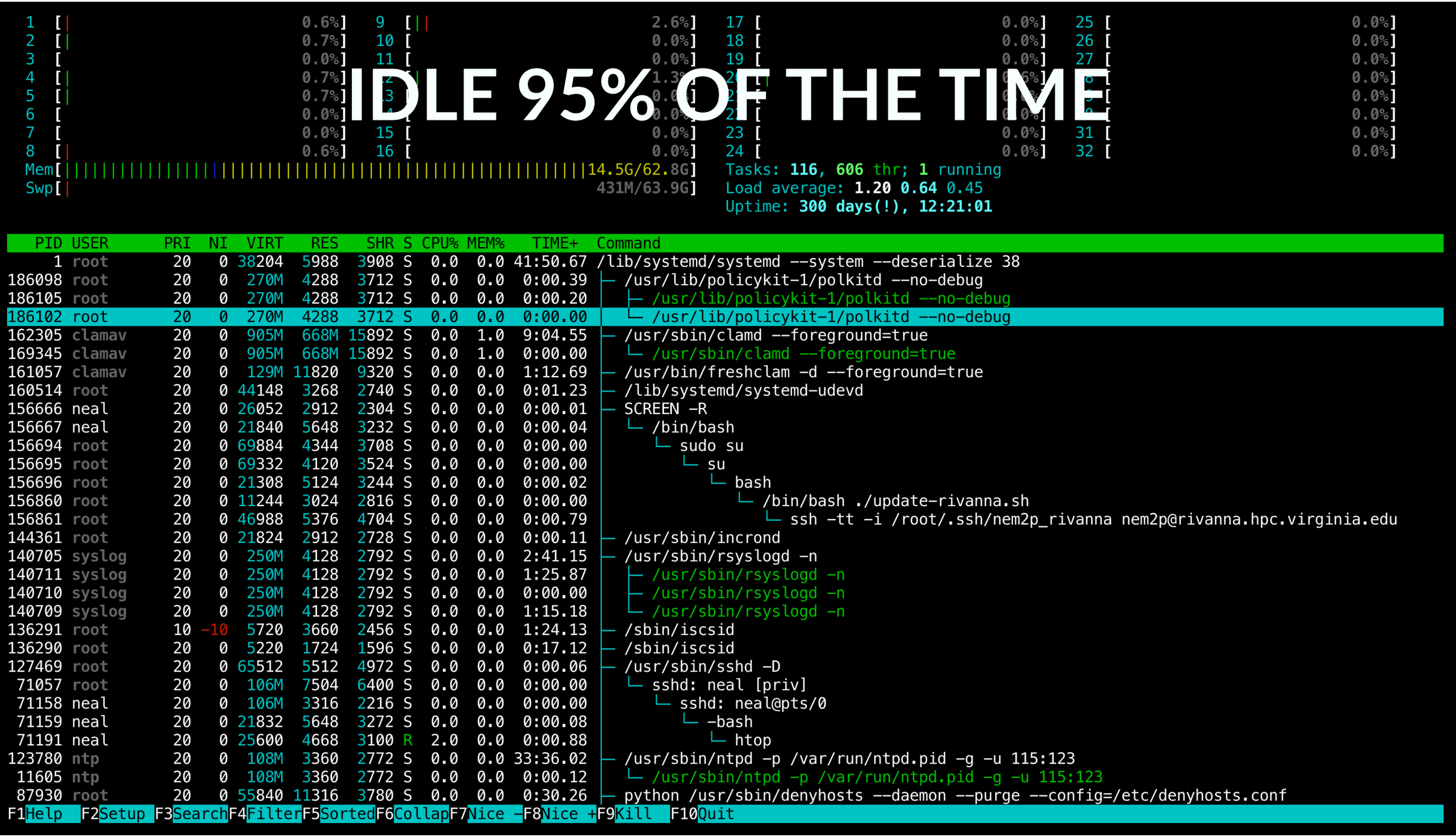


Source: Gartner (July 2021)

Why the cloud?

1. Unused capacity
2. Capital expenses
3. Scalable
4. Programmable


PARKED 95% OF THE TIME


IDLE 95% OF THE TIME
Mem[14.5G/62.8G]
Swp[431M/63.9G]
Tasks: 116, 606 thr; 1 running
Load average: 1.20 0.64 0.45
Uptime: 300 days(!), 12:21:01
PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command
F1Help F2Setup F3SearchF4FilterF5SortedF6CollapF7Nice -F8Nice +F9Kill F10Quit


UBER


airbnb


vmware®


aws

Why the cloud?

1. Unused capacity
2. Capital vs. Operational expenses
3. Scalable
4. Programmable

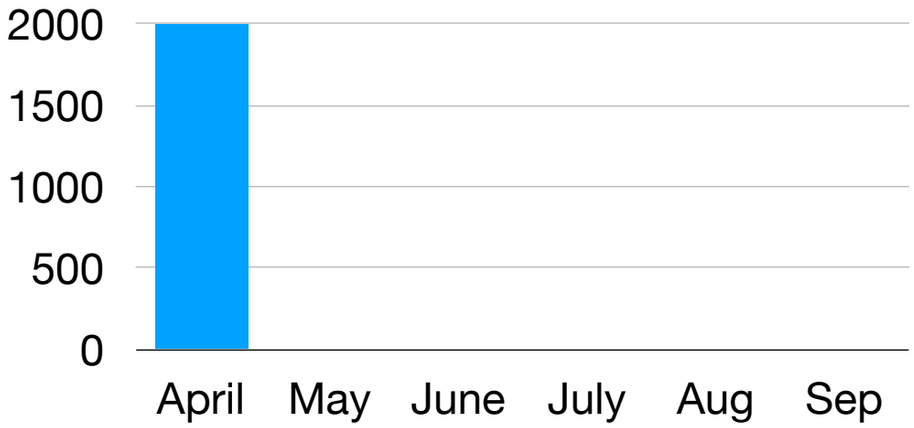

UID
DVD
HP
ProLiant
DL380 G7

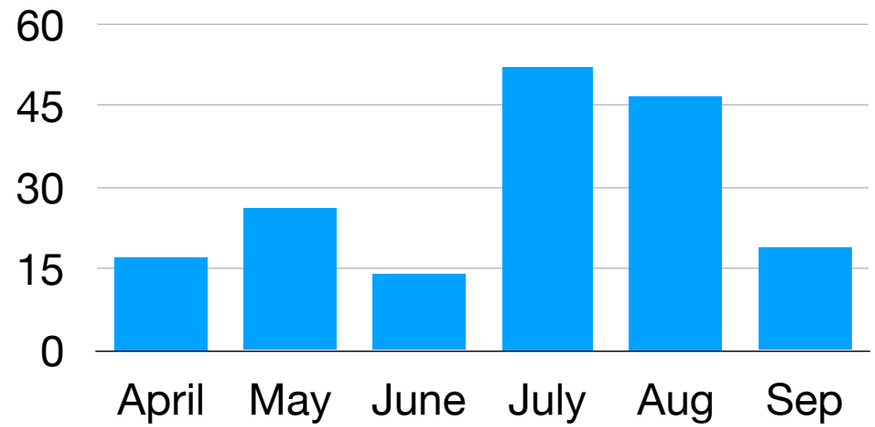
Capital Expenses

Order
Ship
Fixed CPU
Fixed RAM
Fixed Storage

Operational Expenses

Power
Connectivity
Cooling
Engineering



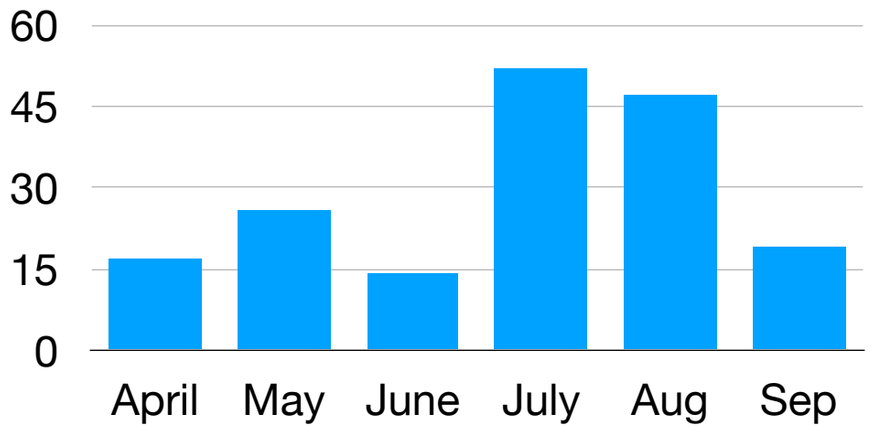


Capital Expenses

Order
Fixed CPU
Fixed RAM
Fixed Storage
Ship

Operational Expenses

Power
Connectivity
Cooling
Engineering



Cloud = Pay As You Go

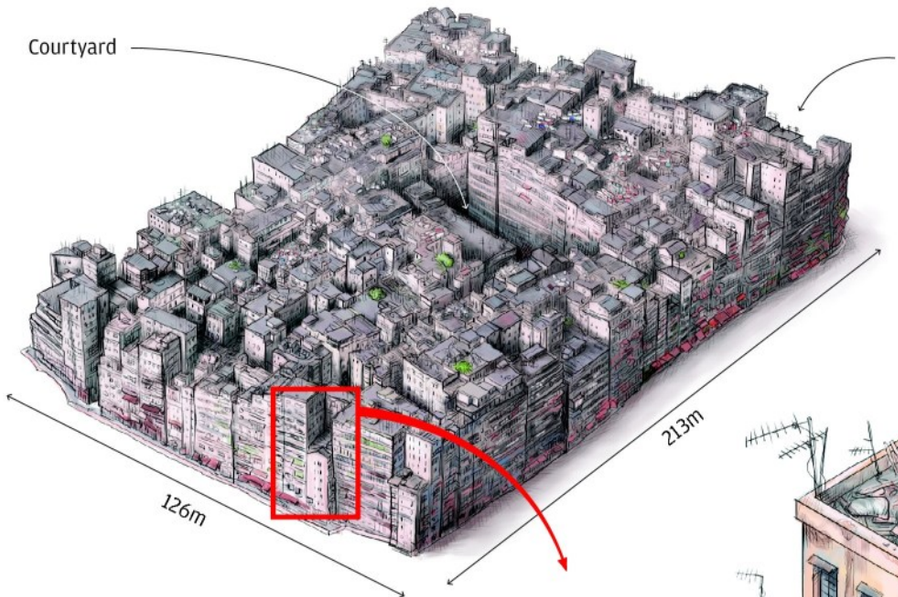
Why the cloud?

1. Unused capacity
2. Capital vs. Operational expenses
3. Scalable & Reconfigurable
4. Programmable


Kowloon Walled City
#wsjkwc
WSJ
Live

City of anarchy

Kowloon Walled City, located not far from the former Kai Tak Airport, was a remarkable high-rise squatter camp that by the 1980s had 50,000 residents. A historical accident of colonial Hong Kong, it existed in a lawless vacuum until it became an embarrassment for Britain. This month marks the 20th anniversary of its demolition.



500
Buildings built into 2.7 hecares

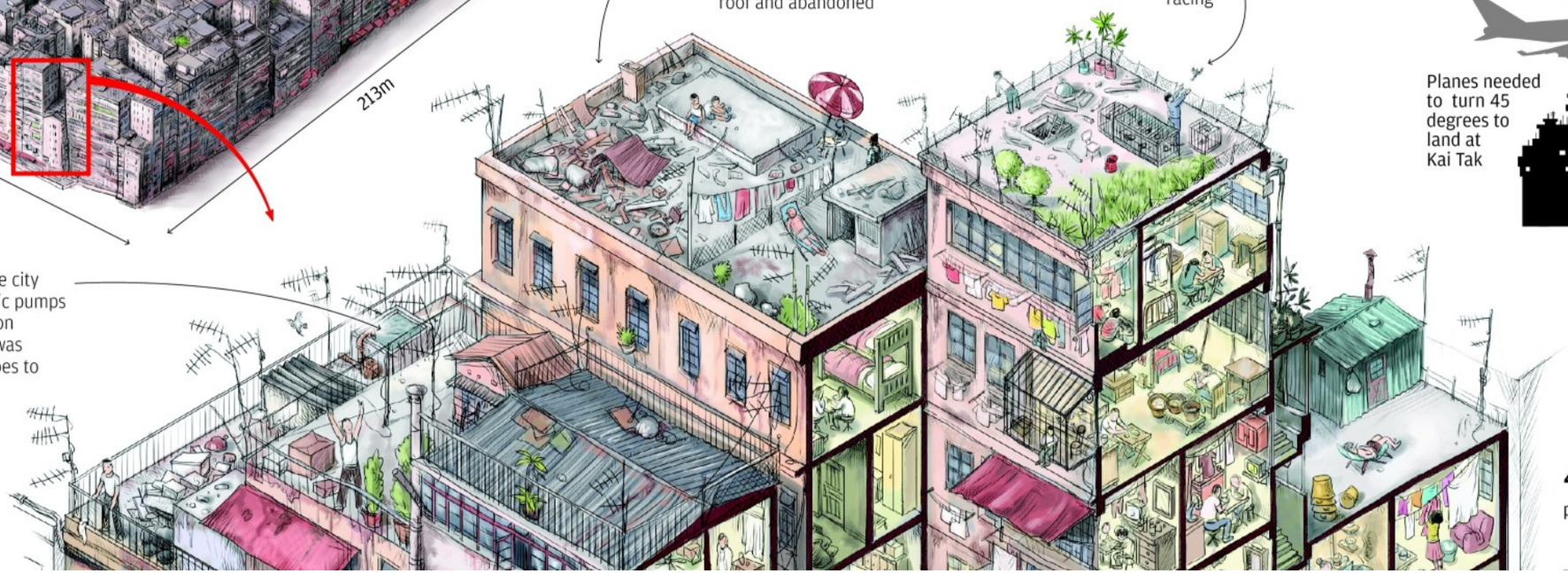
Without municipal services, there was no rubbish collection. Old television sets, broken furniture, discarded mattresses and other bulky items were hauled to the roof and abandoned

Other rooftops were used for exercise, playgrounds, relaxing and even pigeon racing

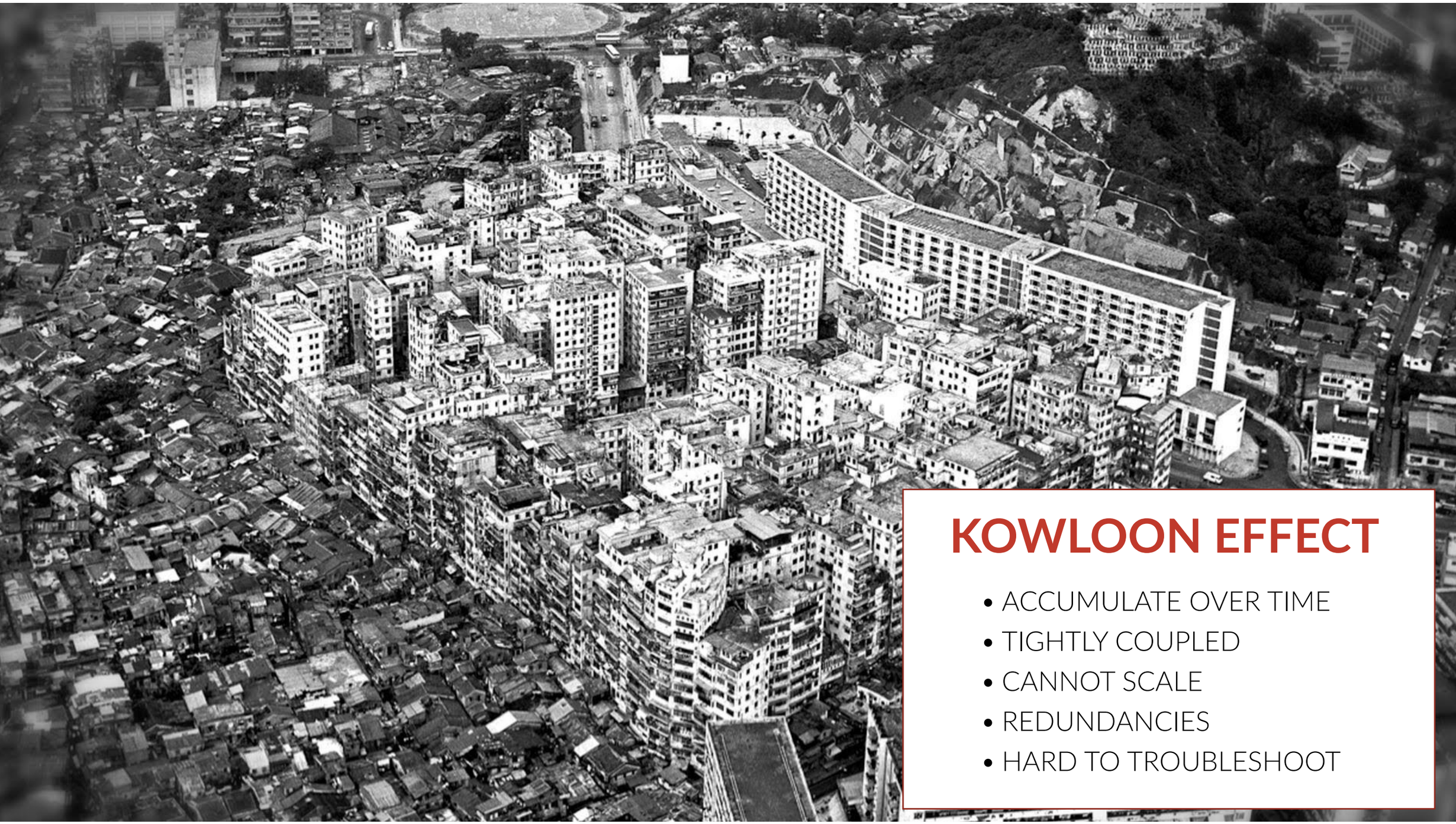
Planes needed to turn 45 degrees to land at Kai Tak

Buildings were no more than 14 floors high to avoid collisions

There were 77 wells inside the city some 90 metres deep. Electric pumps delivered water to big tanks on rooftops. From there, water was funnelled through narrow pipes to the homes

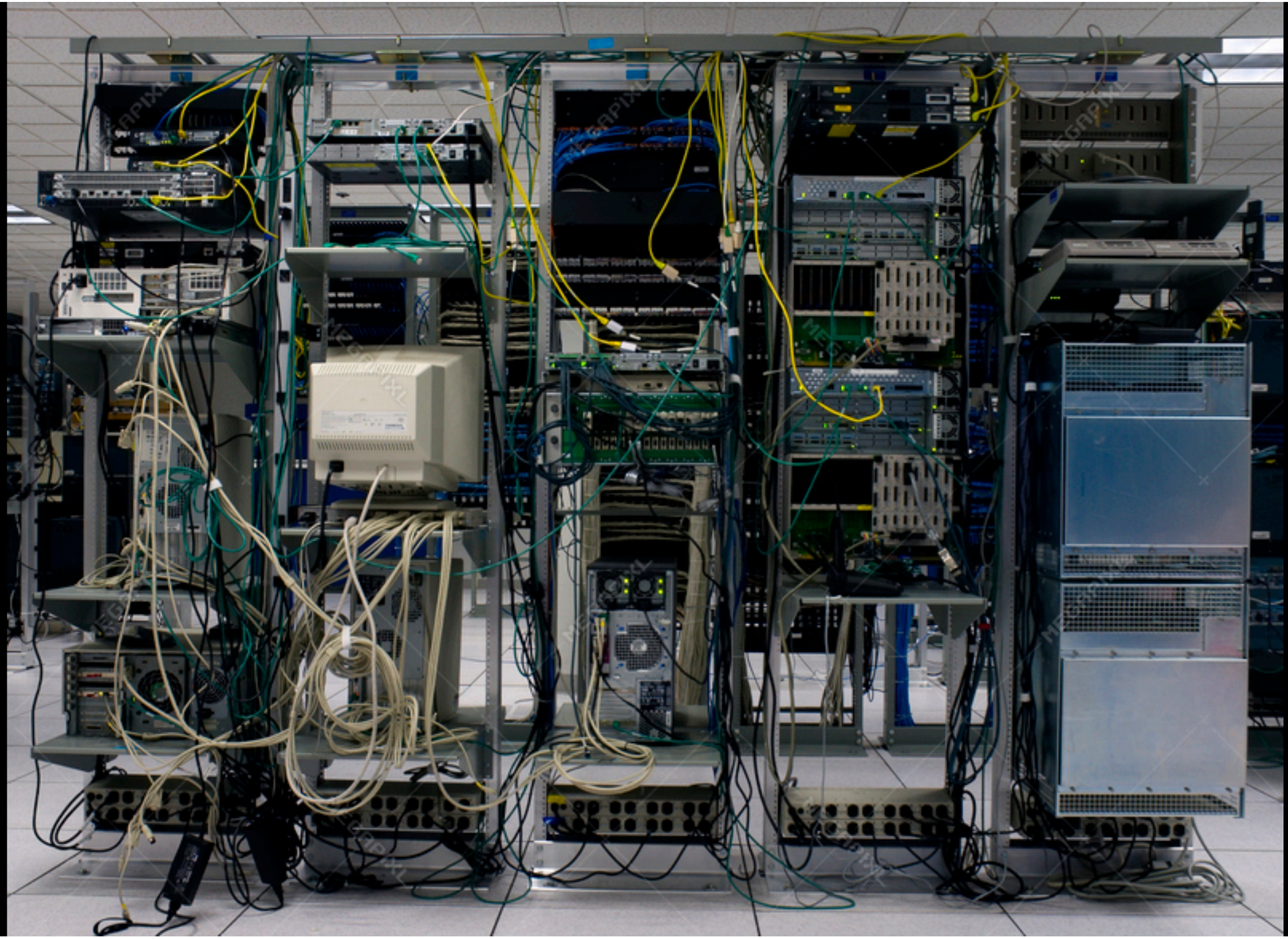


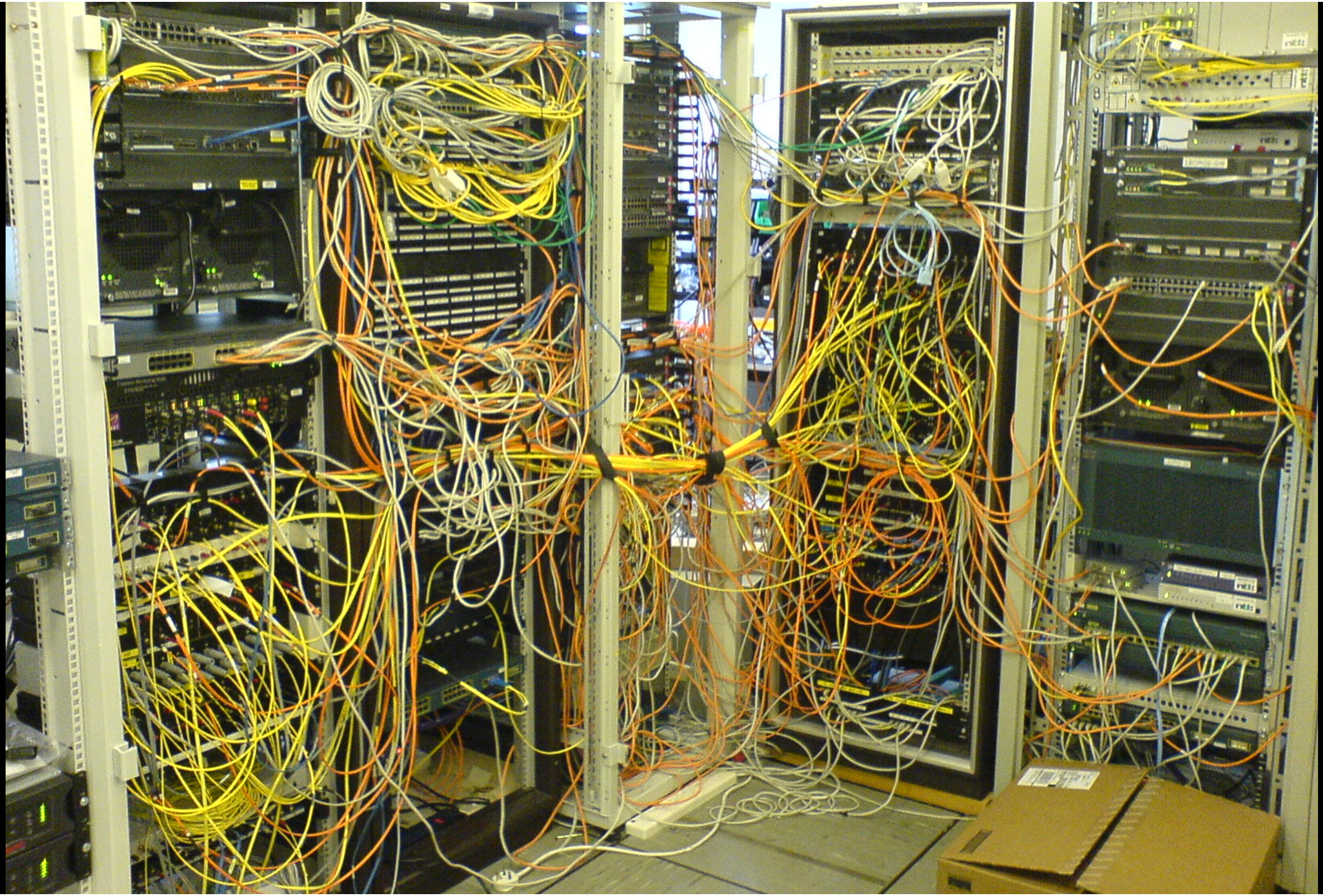
40sq ft
per person

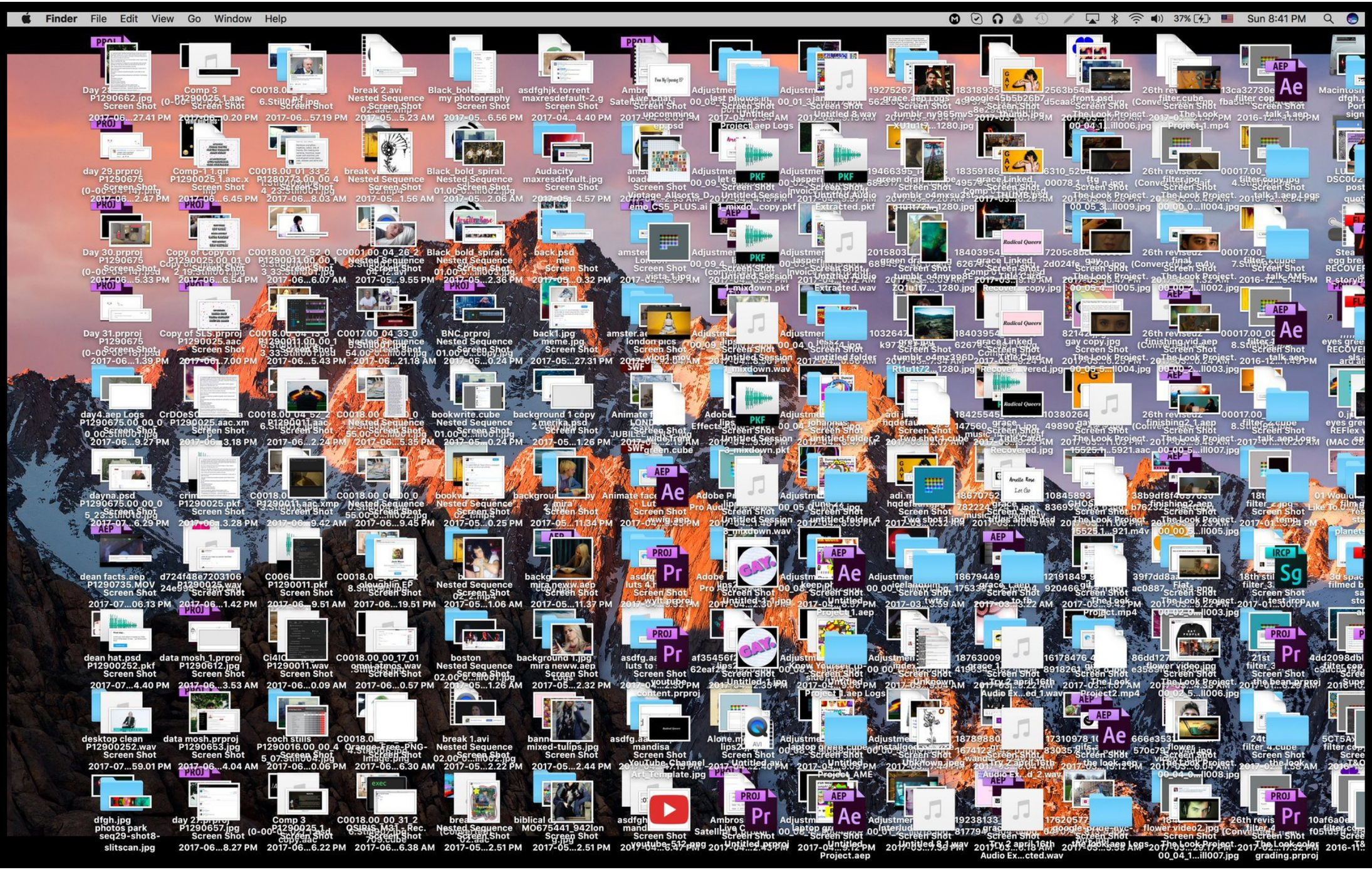


KOWLOON EFFECT

- ACCUMULATE OVER TIME
- TIGHTLY COUPLED
- CANNOT SCALE
- REDUNDANCIES
- HARD TO TROUBLESHOOT






Finder File Edit View Go Window Help
37%
Sun 8:41 PM


abort("The Rails environment
require 'spec_helper'
require 'rspec/rails'
require 'capybara/rspec'
require 'capybara/rails'
Capybara.javascript_driver
Category.delete_all;
Shoulda::Matchers.configure
config.integrate
with.test_framework
with.library :rails
end
end
Add additional requires
Requires supporting
spec/support/ and its
run as spec files by default.
in _spec.rb will both be
run twice. It is
end with _spec.rb.

Why the cloud?

1. Unused capacity
2. Capital vs. Operational expenses
3. Scalable & Reconfigurable
4. Programmable

CASE STUDY: NETFLIX

GLOBAL PRESENCE IN 190 COUNTRIES

6B HOURS OF USAGE / MONTH

6P EST - 12 PST EAK US GE

12 EVENTS / ECO

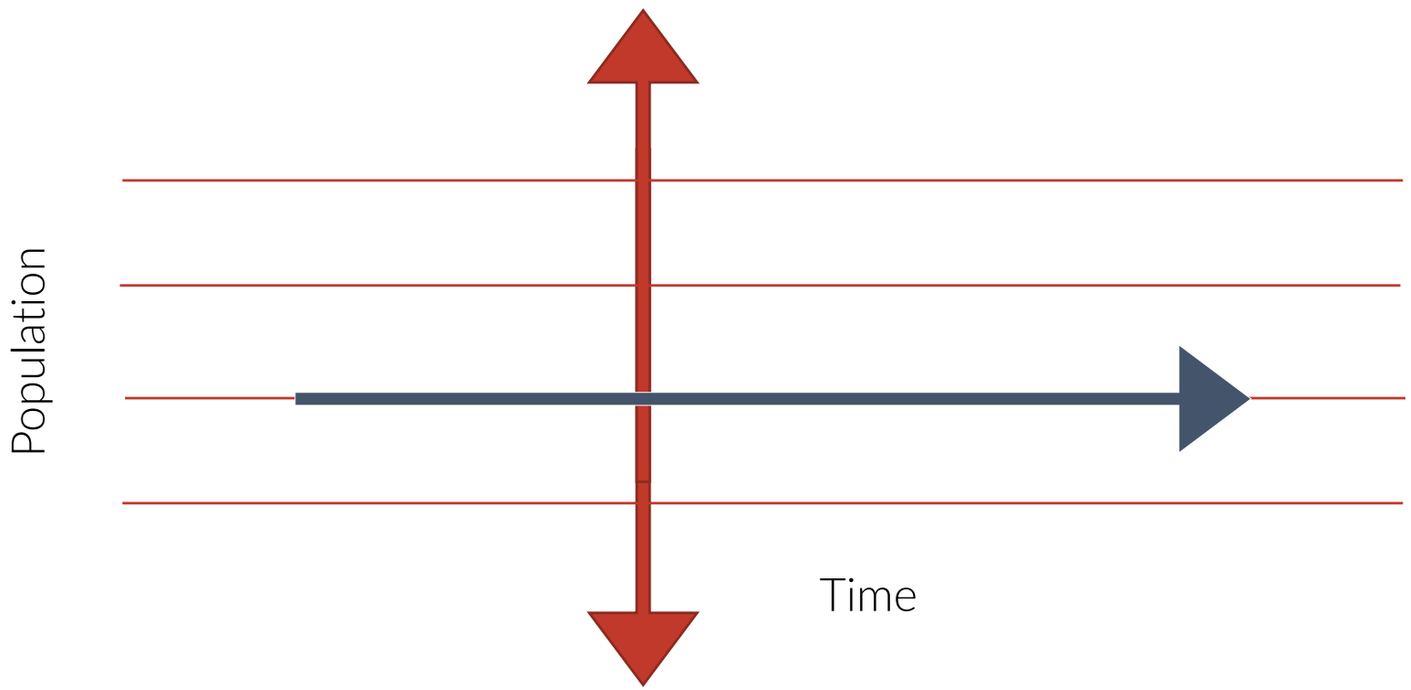
NETFLIX

2000+ A/B TESTS ANNUALLY

100,000+ INSTANCES PEAK

25-30% OF ALL INTERNET BANDWIDTH

OUTLIER DETECTION

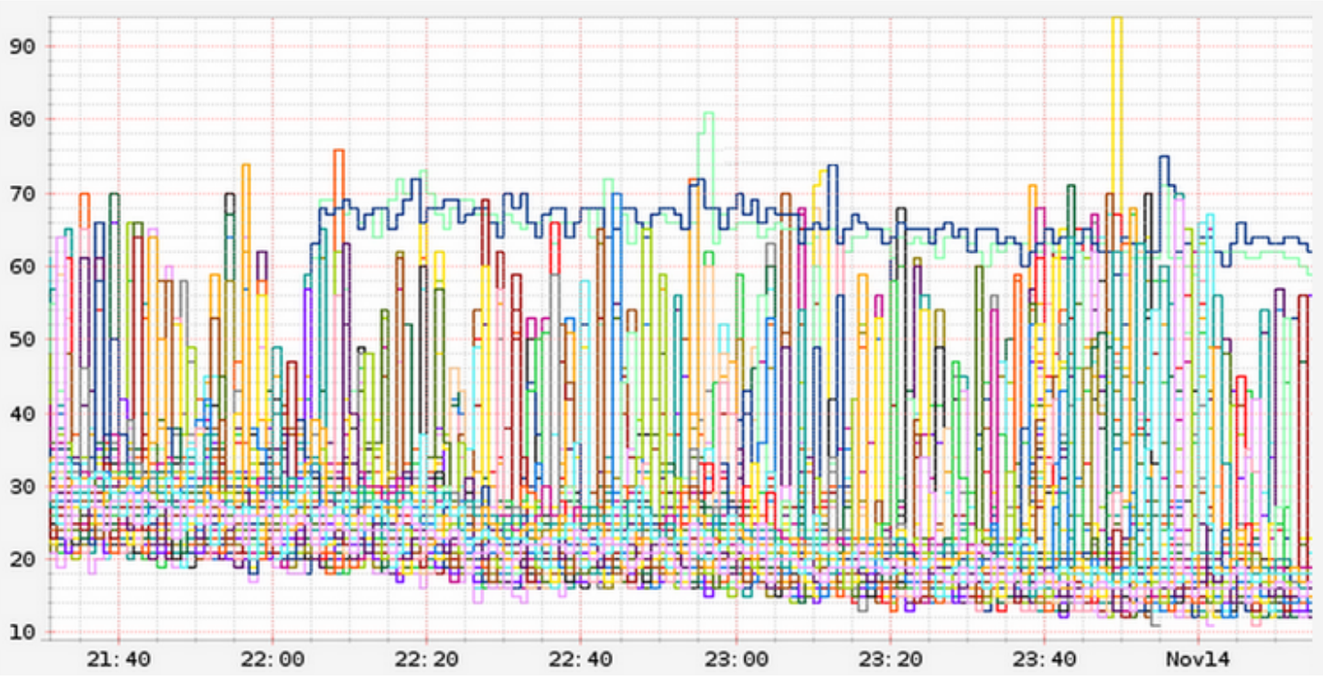


ANOMALY DETECTION

1 OUTLIERS - Stand out in a single moment

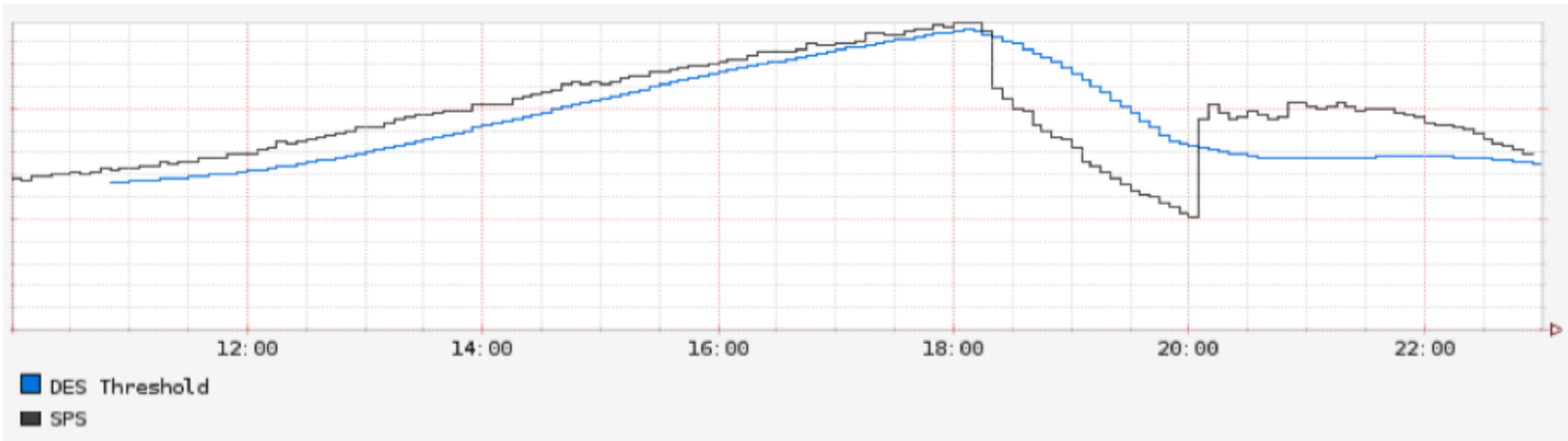
2 ANOMALIES - Stand out over time

OUTLIER DETECTION



1 OUTLIERS - Stand out in a single moment

ANOMALY DETECTION



Kalman filter smoothes time series data

Predict the future based on past history

Favor recent history

Threshold-based alerts

2 ANOMALIES - Stand out over time

CHAOS ENGINEERING TOOLS

Provoking distributed systems to understand their behavior under disruption.

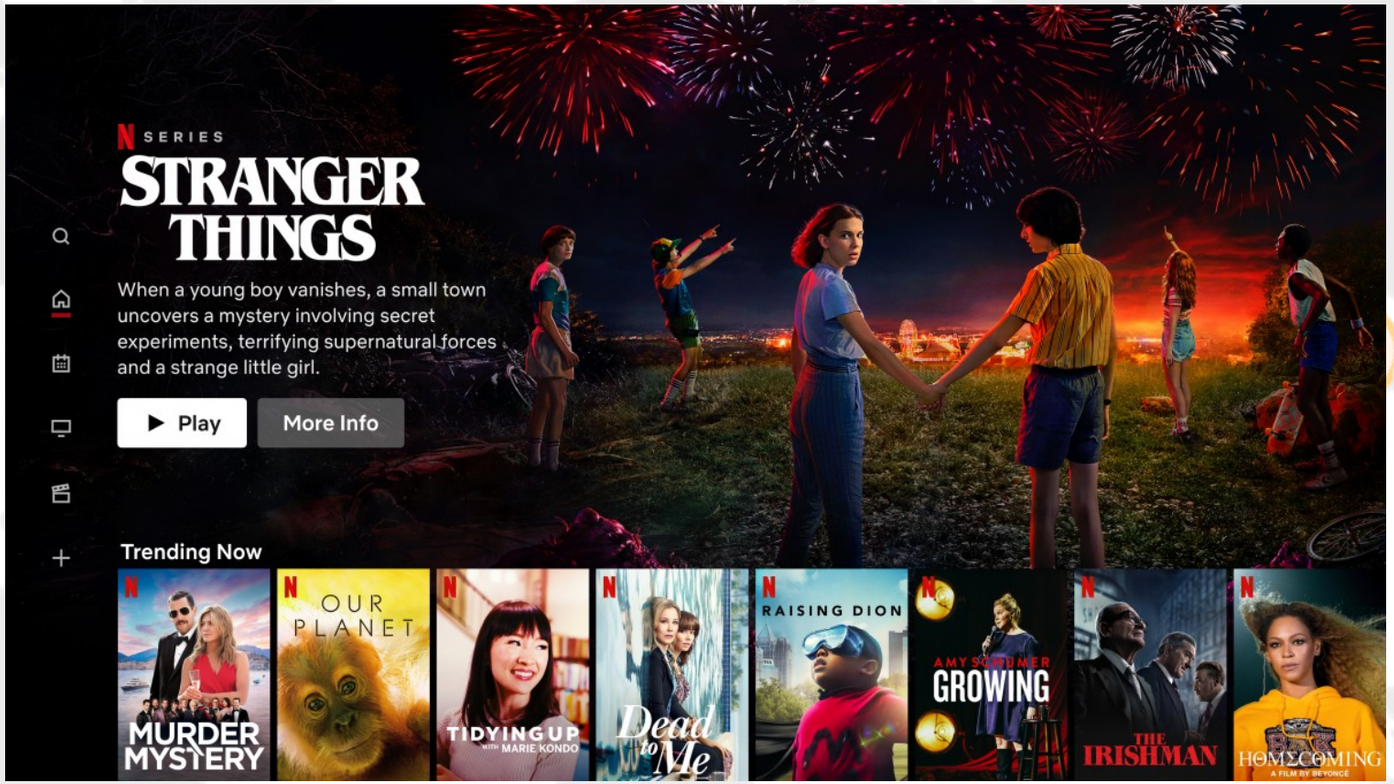
NETFLIX SIMIAN ARMY: Fault Injection

1. Chaos Monkey
2. Janitor Monkey
3. Conformity Monkey

AWS FAULT INJECTION SIMULATOR

A managed service for running fault injection experiments to improve an application's performance, observability, and resiliency.

CDN HANDOFF



TIME

NETFLIX

us-east-ATL

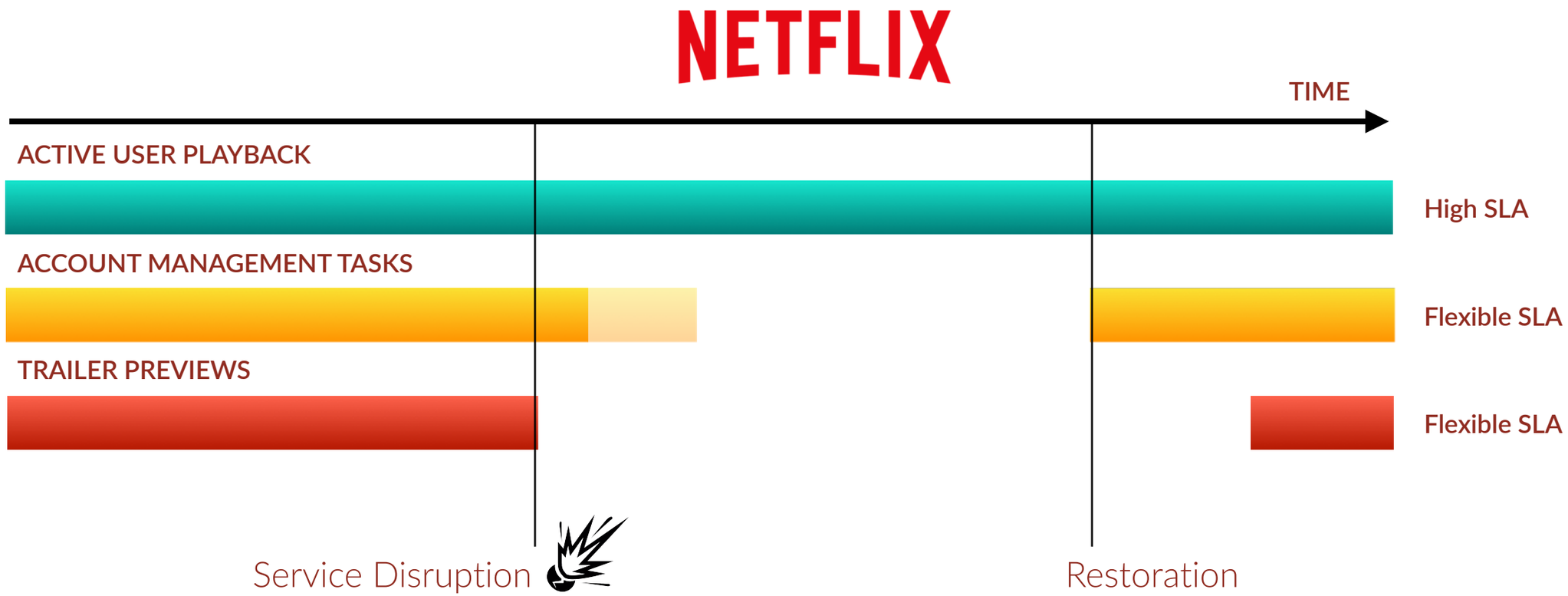
us-east-MIA

us-west-POR

us-west-LA

eu-west-DUB

LOAD SHEDDING



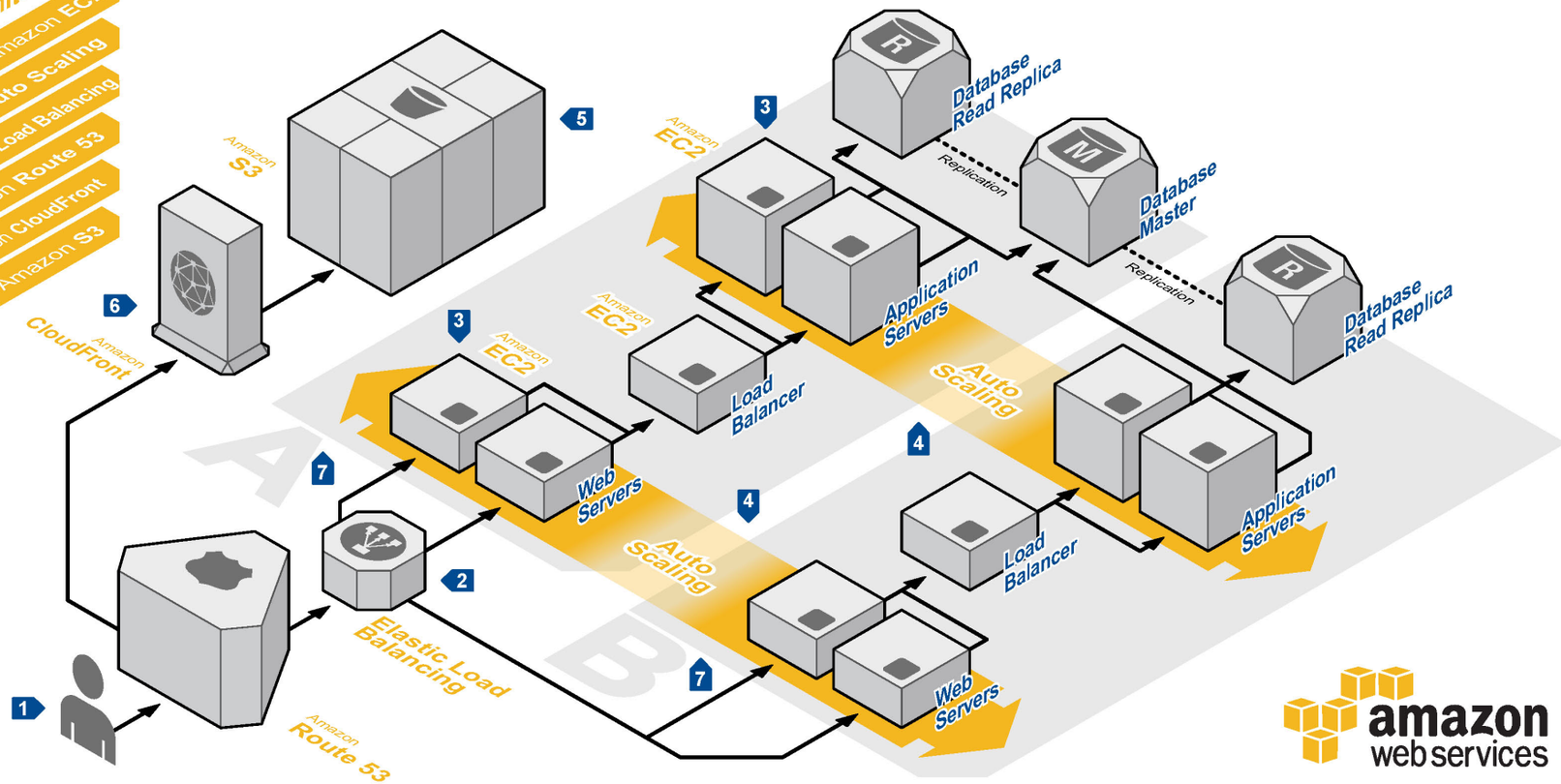
Queue/drop lower-priority tasks to sustain priority service layers

AWS Reference Architectures

- Amazon EC2
- Auto Scaling
- Elastic Load Balancing
- Amazon Route 53
- Amazon CloudFront
- Amazon S3

WEB APPLICATION HOSTING

Highly available and scalable web hosting can be complex and expensive. Dense peak periods and wild swings in traffic patterns result in low utilization rates of expensive hardware. Amazon Web Services provides the reliable, scalable, secure, and high-performance infrastructure required for web applications while enabling an elastic, scale out and scale down infrastructure to match IT costs in real time as customer traffic fluctuates.



System Overview

1 The user's DNS requests are served by **Amazon Route 53**, a highly available Domain Name System (DNS) service. Network traffic is routed to infrastructure running in Amazon Web Services.

2 HTTP requests are first handled by Elastic Load Balancing, which automatically distributes incoming application traffic across multiple **Amazon Elastic Compute Cloud (EC2)** instances across Availability Zones (AZs). It enables even greater fault tolerance in your applications, seamlessly providing the amount of load balancing capacity needed in response to incoming application traffic.

3 Web servers and application servers are deployed on **Amazon EC2** instances. Most organizations will select an **Amazon Machine Image (AMI)** and then customize it to their needs. This custom AMI will then be used as the starting point for future web development.

4 Web servers and application servers are deployed in an **Auto Scaling** group. Auto Scaling automatically adjusts your capacity up or down according to conditions you define. With Auto Scaling, you can ensure that the number of **Amazon EC2** instances you're using increases seamlessly during demand spikes to maintain performance and decreases automatically during demand lulls to minimize costs.

5 Resources and static content used by the web application are stored on **Amazon Simple Storage Service (S3)**, a highly durable storage infrastructure designed for mission-critical and primary data storage.

6 Static and streaming content is delivered by **Amazon CloudFront**, a global network of edge locations. Requests are automatically routed to the nearest edge location, so content is delivered with the best possible performance.

7 **Availability zones (AZs)** are distinct geographic locations that are engineered to insulate against failures in other AZs. Multiple AZs are combined into a region. Here, the entire web application is deployed in two different AZs for high availability.

- Well-defined
- Loosely coupled
- Scalable
- Reproducible
- Can troubleshoot easily
- Less redundancy

Why the cloud?

1. Unused capacity
2. Capital vs. Operational expenses
3. Scalable
4. Programmable

Infrastructure as Code

Infrastructure as Code

Automatable

Versionable

Repeatable

Distributable

CLOUDFORMATION


```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "AWS CloudFormation Sample Template LAMP_Multi_AZ: Create
  demonstrates using the AWS CloudFormation bootstrap scripts to install t
  creates one or more Amazon EC2 instances, an Elastic Load Balancer and a
  "Parameters": { … }, // 10 items
  "Mappings": { … }, // 3 items
  "Conditions": { … }, // 2 items
  "Resources": {
    "ElasticLoadBalancer": { … }, // 2 items
    "WebServerGroup": { … }, // 4 items
    "LaunchConfig": { … }, // 3 items
    "WebServerSecurityGroup": { … }, // 2 items
    "DBEC2SecurityGroup": { … }, // 3 items
    "DBSecurityGroup": { … }, // 3 items
    "MySQLDatabase": { … } // 2 items
  },
  "Outputs": { … } // 1 item
}
```

TERRAFORM

```
provider "aws" {
  region = "${var.aws_region}"
}

provider "archive" {}

data "archive_file" "zip" {
  type        = “zip"
  source_file = “hello_lambda.py"
  output_path = "hello_lambda.zip"
}

data "aws_iam_policy_document" "policy" {
  statement {
    sid = ""
    effect = "Allow"
    principals {
      identifiers = [“lambda.amazonaws.com"]
      type = “Service"
    }
    actions = [“sts:AssumeRole"]
  }
}
```

RESIZE AN INSTANCE

```
aws ec2 stop-instances \
  --instance-ids i-123456789012

aws ec2 modify-instance-attribute \
  --instance-id i-123456789012 \
  --instance-type p2.16xlarge

aws ec2 start-instances \
  --instance-ids i-012987654321
```

RESIZING PROGRAMMATICALLY

```
import boto3

instance = “i-123456789012”
ec2 = boto3.client(‘ec2’)

response =
client.modify_instance_attribute(
  InstanceId=instance,
  InstanceType={
    ‘Value’: ‘c4.8xlarge’,
  }
)
```

Cattle Not Pets

Pets

* Are given names
* Are unique & hand-raised
* When sick you nurse back to health

Cattle

* Are assigned IDs
* Are nearly identical
* When sick you get another

Questions?

Sign Up for AWS Academy

bit.ly/dsac-cloud

GitHub Repo

github.com/nmagee/dsac-cloud