

Data-Intensive Distributed Computing

CS 431/631 451/651 (Winter 2019)

The Final Part

April 4, 2019

Adam Roegiest

Kira Systems

These slides are available at <http://roegiest.com/bigdata-2019w/>

This work is licensed under a Creative Commons Attribution-Noncommercial-Share Alike 3.0 United States
See <http://creativecommons.org/licenses/by-nc-sa/3.0/us/> for details



The datacenter *is* the computer!

“Big ideas”

Scale “out”, not “up” *

Limits of SMP and large shared-memory machines

Assume that components will break *

Engineer software around hardware failures

Move processing to the data *

Cluster have limited bandwidth, code is a lot smaller

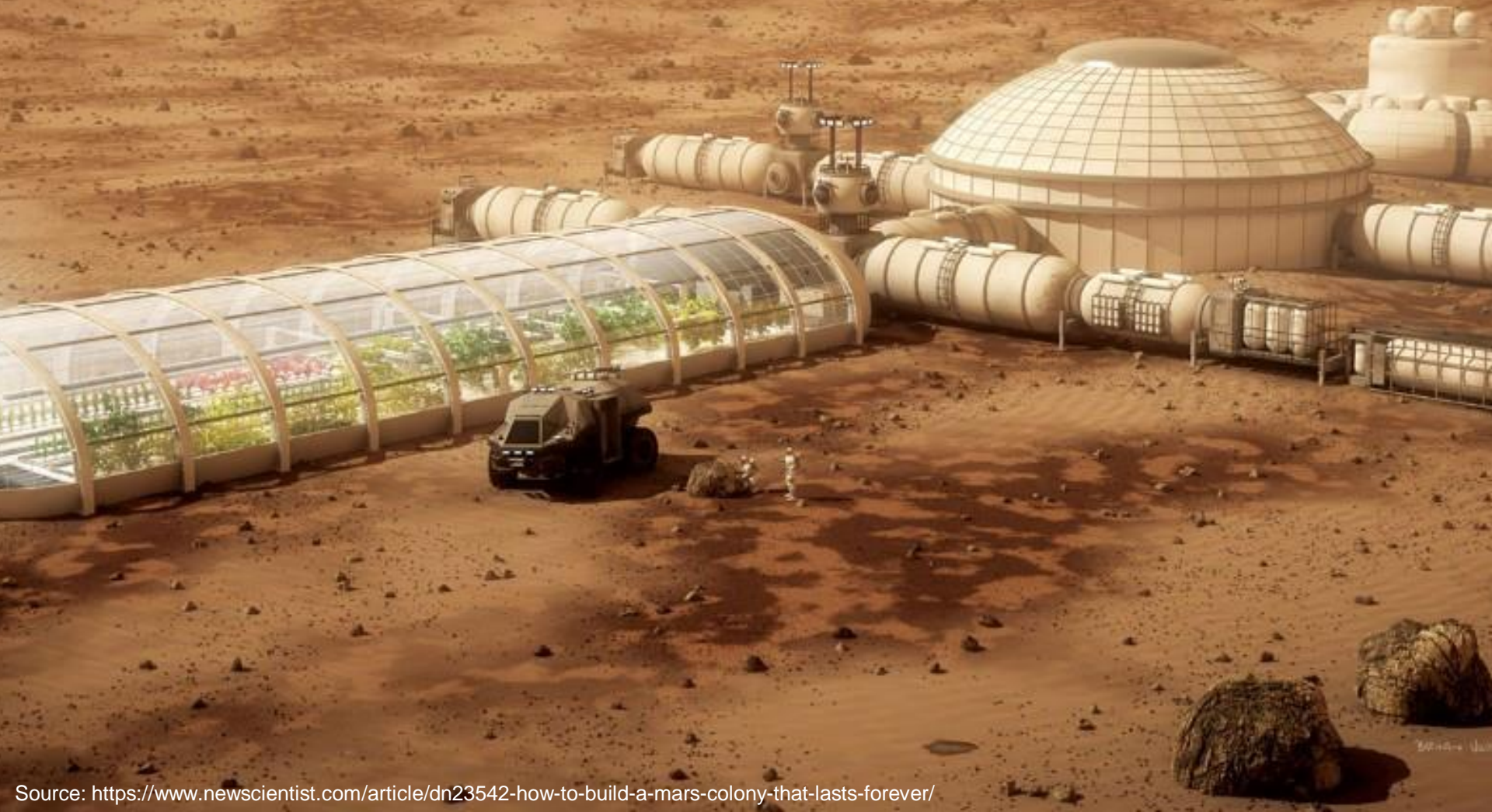
Process data sequentially, avoid random access

Seeks are expensive, disk throughput is good



Humans *will* colonize Mars

Sooner than you think



Elon Musk Charts Path to Colonizing Mars Within a Decade

By Robin Seemangal • 06/06/16 9:10am



Planning to send Dragon to Mars in 2018. Red Dragons will inform architecture, details to come



RETWEETS 15,988 LIKES 26,640

11:50 AM - 27 Apr 2016

16K 27K

Buzz Aldrin developing a 'master plan' to colonize Mars within 25 years

Aldrin and the Florida Institute of Technology are pushing for a Mars settlement by 2039, the 70th anniversary of his own Apollo 11 moon landing



Florida Tech's president, Anthony J Catanese, left, talks with Apollo 11 astronaut Buzz Aldrin as he shows him the campus on Thursday in Melbourne, Florida Photograph: Craig Rubadoux/AP

Source: <https://www.theguardian.com/science/2015/aug/27/buzz-aldrin-colonize-mars-within-25-years>

Source: <https://twitter.com/SpaceX/status/725351354537906176>

Source: <http://observer.com/2016/06/elon-musk-charts-path-to-colonizing-mars-within-a-decade/>

“Mars can’t just be a one-shot mission” – Buzz Aldrin



“The Pilgrims on the Mayflower came here to live and stay. They didn’t wait around Plymouth Rock for the return trip, and neither will people building up a population and a settlement [on Mars].”

Needs

Produce breathable air

Grow food

Build shelter

Mine fuel and materials

“Staying alive”

Conduct science

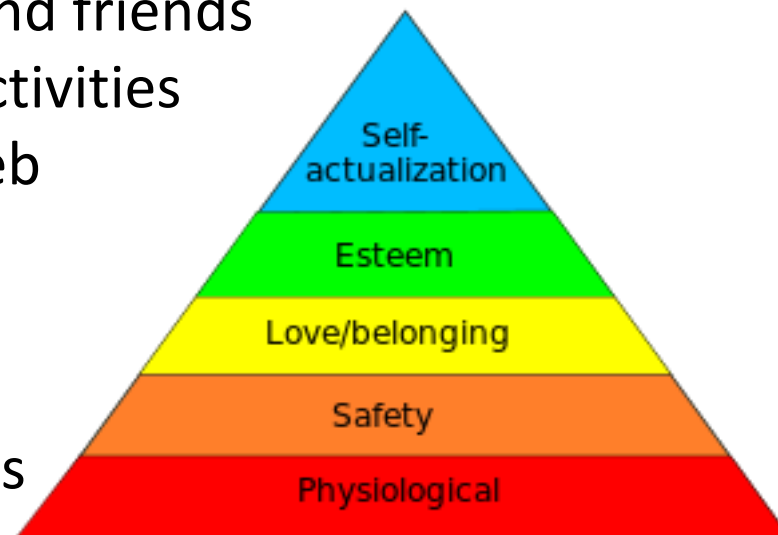
Connect with family and friends

Engage in leisure activities

Search the web

“Staying sane”

Maslow's hierarchy of needs



Searching the web should be as easy
from Mars as it is from Marseille!

The fundamental problem: Latency

speed of light: 2-24 minutes

rockets: 5-10 months

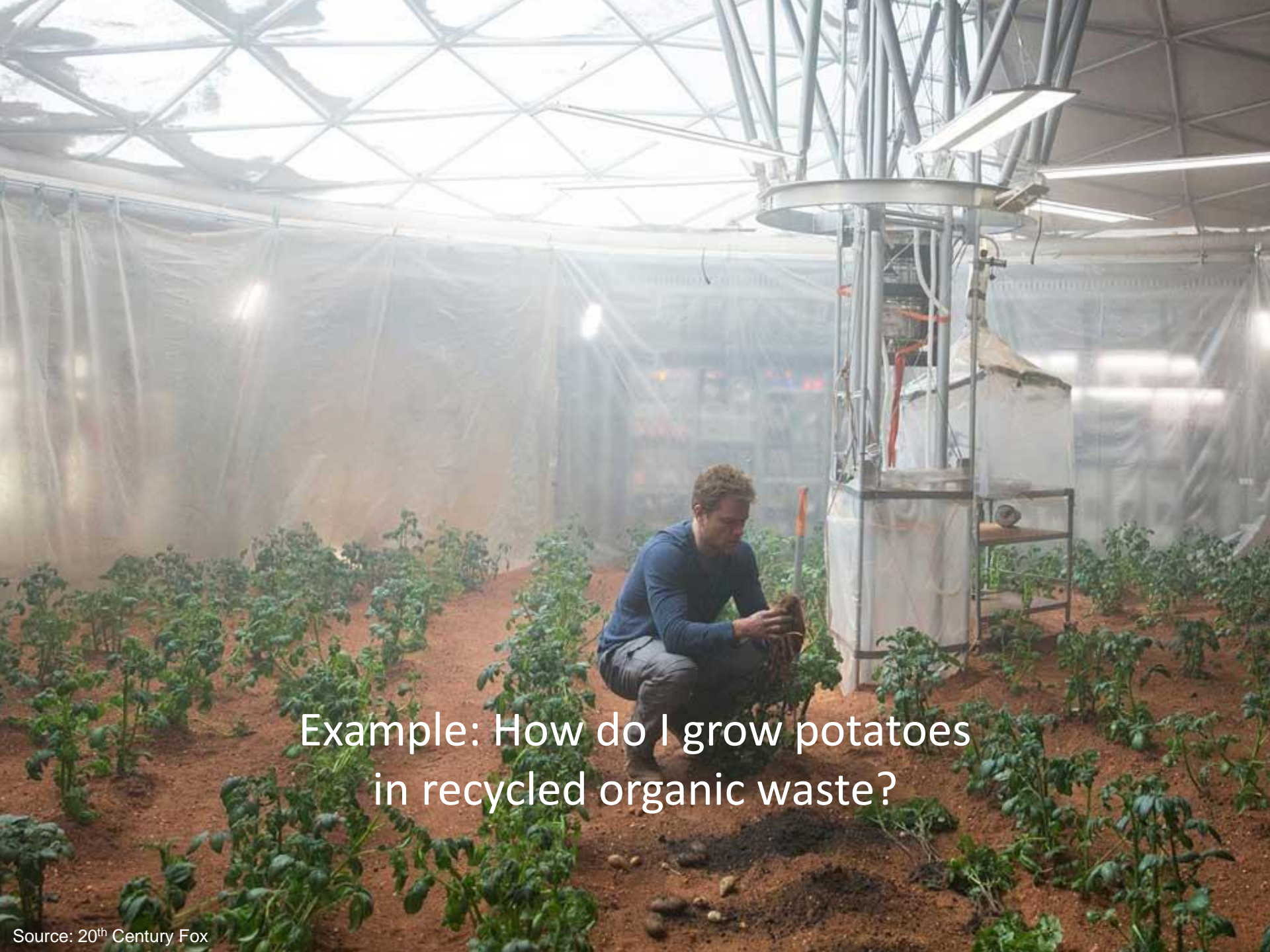
Bandwidth is “reasonable”

Lunar Laser Communications Demonstration:

622-Mbps downlink, 20-Mbps uplink

SneakerNet on rockets: Easily PBs

What's doable, what's not?



Example: How do I grow potatoes
in recycled organic waste?

Search from Mars: Implementation

Step 1. Rocket SneakerNet

Step 2. Beam the diffs

We know exactly how to do this!

Step 3. User model activate!

We have a good idea how to do this!

It's a caching problem!

We've worked out some simulations already...

C. Clarke, G. Cormack, J. Lin, and A. Roegiast. Ten Blue Links on Mars. WWW 2017.

J. Lin, C. Clarke, and G. Baruah. Searching from Mars. IEEE Internet Computing, 20(1):78-82, 2016.

For the truly skeptical...



Search from Mars ~
Search from regions on
Earth with poor connectivity

Easter Island
Canadian Arctic
Villages in rural India

More “down to Earth”
applications!

Big Data



What's growing faster?

Big Data

What do I mean here?

Moore's Law

What do I mean here?

Big Data $>$ Moore's Law

Big Data $<$ Moore's Law

Big Data \sim Moore's Law

First, a story...

What's growing faster?

Big Data

Let's restrict to
Human-generated data

Moore's Law

Bounds?

Human population
Data generation per unit time

Big Data > Moore's Law

Big Data < Moore's Law

Big Data ~ Moore's Law

Implications?

Back to my story...

X1 Instances for EC2 – Ready for Your Memory-Intensive Workloads

by Jeff Barr | on 18 MAY 2016 | in [Amazon EC2*](#) | [Permalink](#) | [Share](#)

Many AWS customers are running memory-intensive big data, caching, and analytics workloads and have been asking us for EC2 instances with ever-increasing amounts of memory.

Last fall, I first told you about our plans for the new X1 instance type. Today, we are announcing availability of this instance type with the launch of the **x1.32xlarge** instance size. This instance has the following specifications:

- Processor: 4 x Intel™ [Xeon E7 8880 v3](#) (Haswell) running at 2.3 GHz – 64 cores / 128 vCPUs.
- Memory: 1,952 GiB with Single Device Data Correction (SDDC+1).
- Instance Storage: 2 x 1,920 GB SSD.
- Network Bandwidth: 10 Gbps.
- Dedicated EBS Bandwidth: 10 Gbps (EBS Optimized by default at no additional cost).

The Xeon E7 processor supports [Turbo Boost 2.0](#) (up to 3.1 GHz), [AVX 2.0](#), [AES-NI](#), and the very interesting (to me, anyway) [TSX-NI](#) instructions. AVX 2.0 (Advanced Vector Extensions) can improve performance on HPC, database, and video processing workloads; AES-NI improves the speed of applications that make use of [AES](#) encryption. The new TSX-NI instructions support something cool called [transactional memory](#). The instructions allow highly concurrent, multithreaded applications to make very efficient use of shared memory by reducing the amount of low-level locking and unlocking that would otherwise be needed around each memory access.

If you are ready to start using the X1 instances in the US East (Oregon), US East (Virginia), US West (Oregon), US West (N. California), Europe (Ireland), Europe (Frankfurt), Europe (Paris), Asia Pacific (Singapore), or Asia Pacific (Sydney) Regions, please [request](#) these instances available in other Regions and in other sizes before t

```
top - 17:13:41 up 52 min, 3 users, load average: 10.52, 4.74, 2.70
Tasks: 943 total, 2 running, 941 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.2 us, 0.6 sy, 0.0 ni, 94.4 id, 4.8 wa, 0.0 hi, 0.0 si, 0.
KiB Mem: 20184757+total, 87502960 used, 19309728+free, 19360 buffers
KiB Swap: 0 total, 0 used, 0 free, 40866816 cached Mem

  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM    TIME+
35957 hdbadm   20   0 86.849g 0.065t 0.036t S 118.812 3.458 8:38.77
35959 hdbadm   20   0 5643852 1.491g 287768 S  1.650 0.077 0:32.41
 1120 root     20   0  12032   3792   664 S   0.990 0.000 0:02.69
37350 root     20   0  14472   2228  1008 R   0.660 0.000 0:00.20
     8 root     rt   0     0     0     0 S   0.330 0.000 0:00.53
```

What's growing faster?

Big Data

Let's restrict to
Human-generated data

Moore's Law

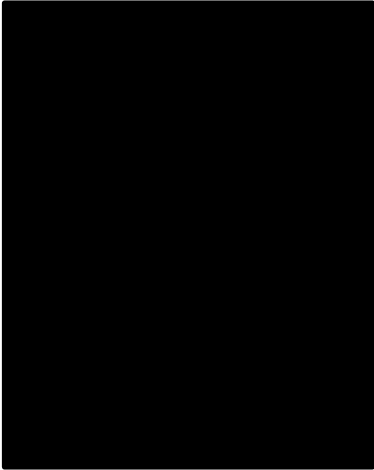
What am I forgetting?

Bounds?

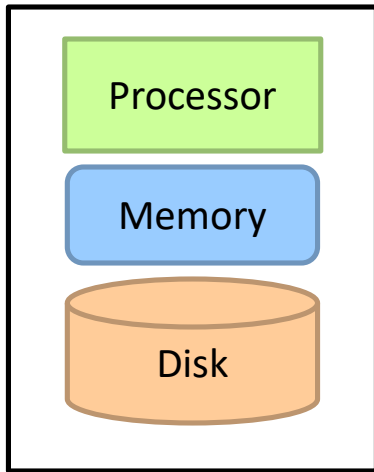
Human population
Data generation per unit time

Serverless Architectures

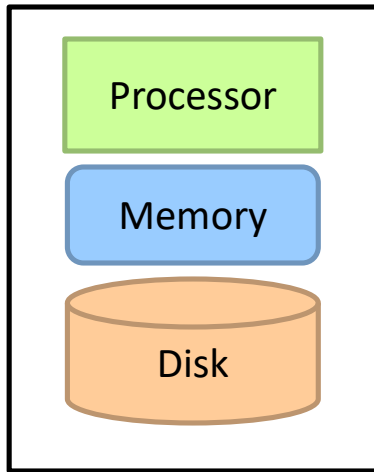
A wide-angle, high-angle photograph of a modern data center. The room is filled with rows of server racks, each illuminated with a soft blue glow. The ceiling is a complex network of dark metal beams and pipes, with several long, rectangular light fixtures hanging from it. The floor is a light-colored, polished tile. The overall atmosphere is clean, industrial, and high-tech.



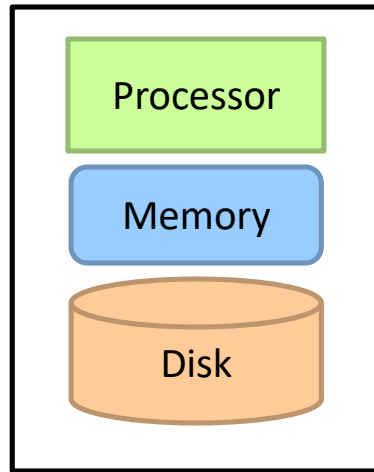
Server



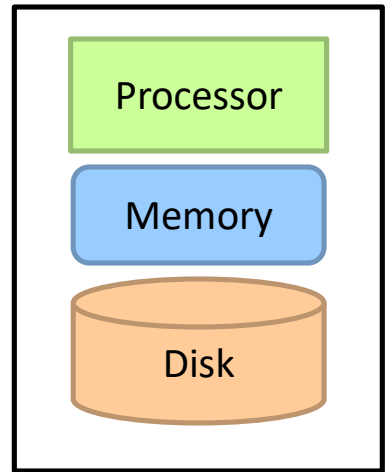
Server



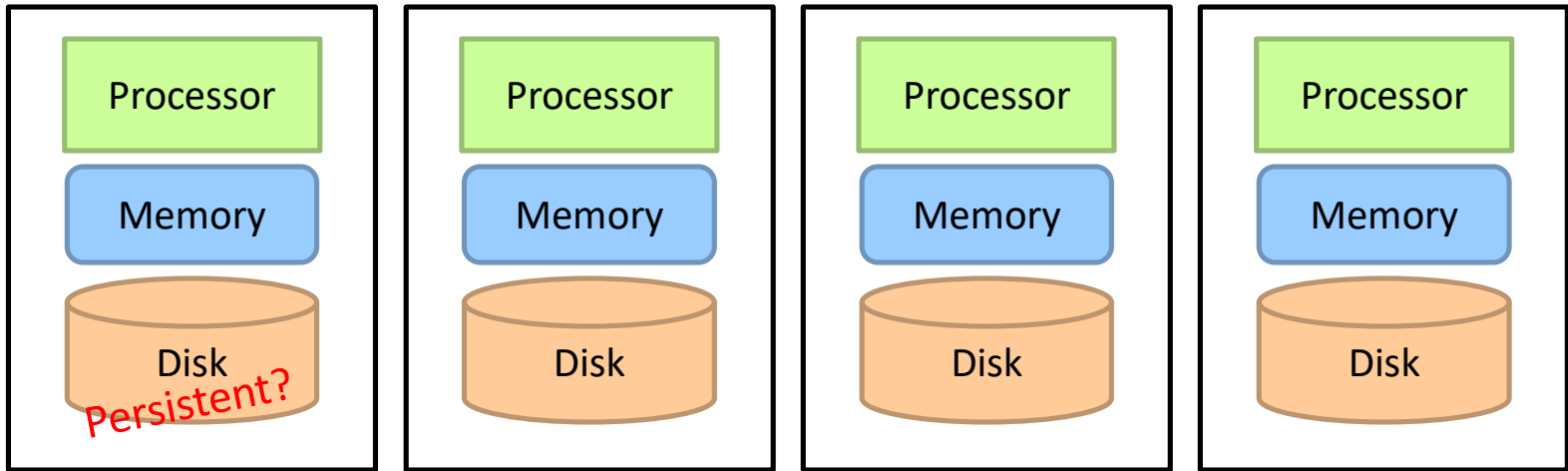
Server



Server



Server



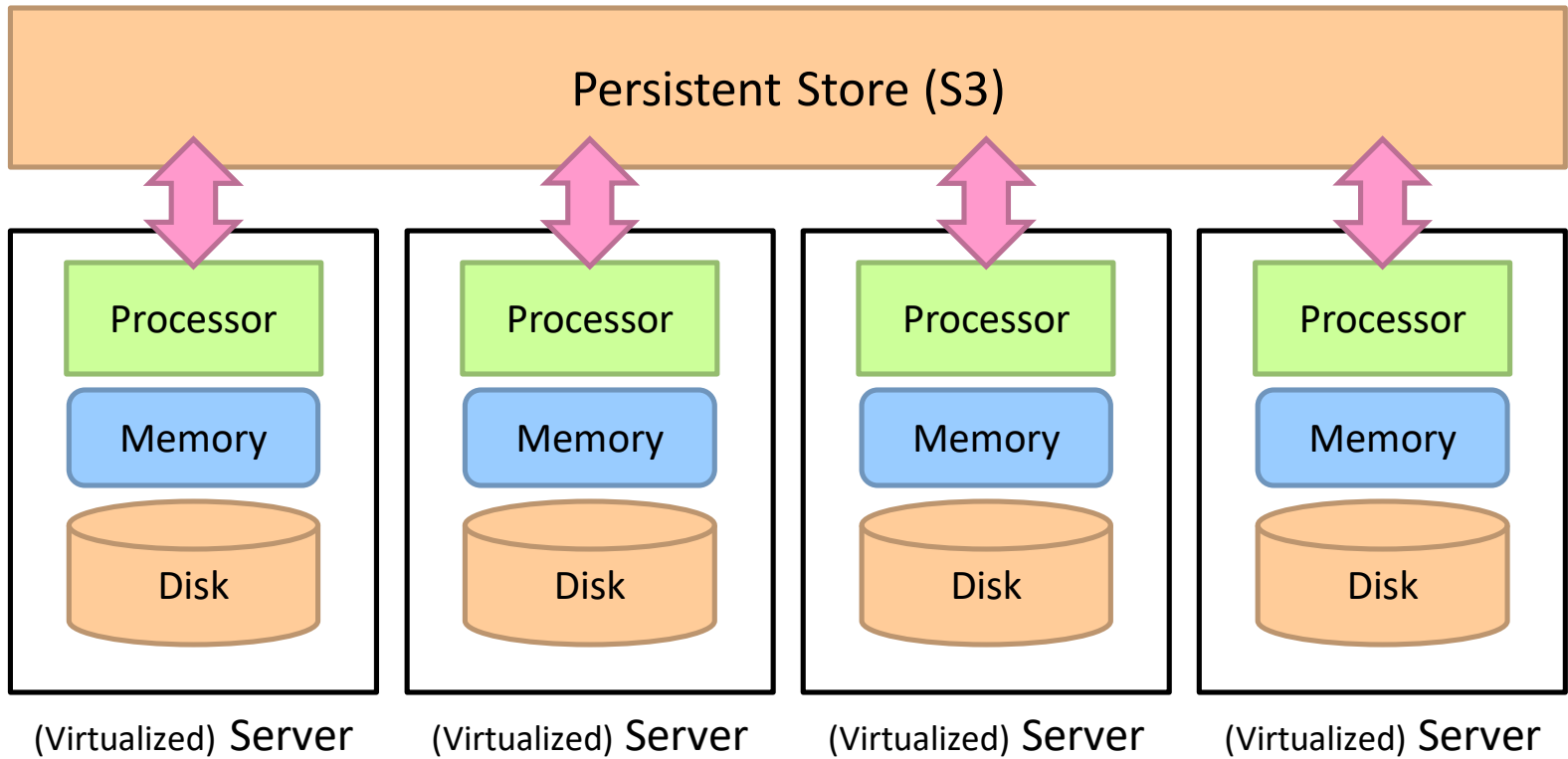
(Virtualized) Server

(Virtualized) Server

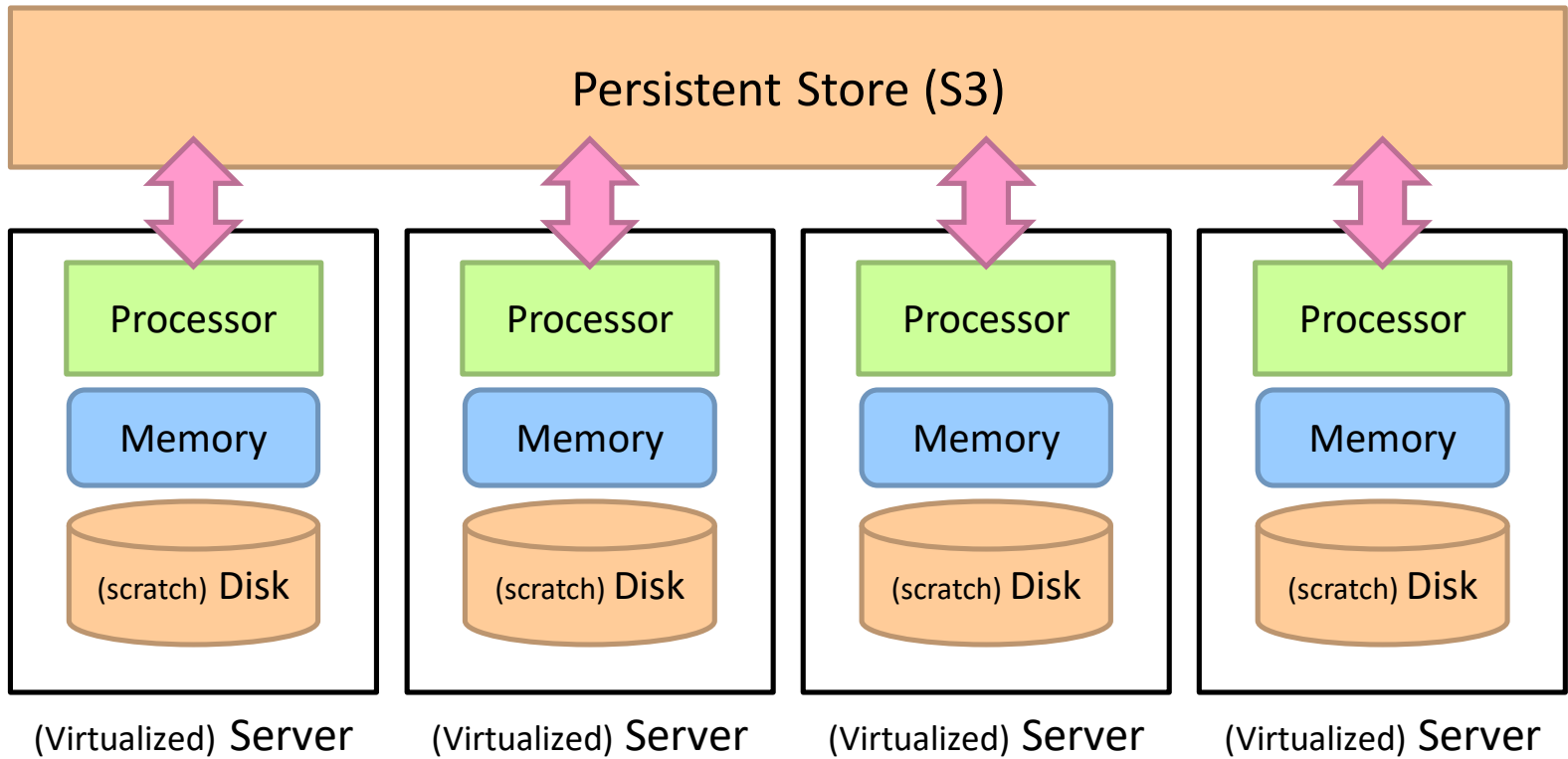
(Virtualized) Server

(Virtualized) Server

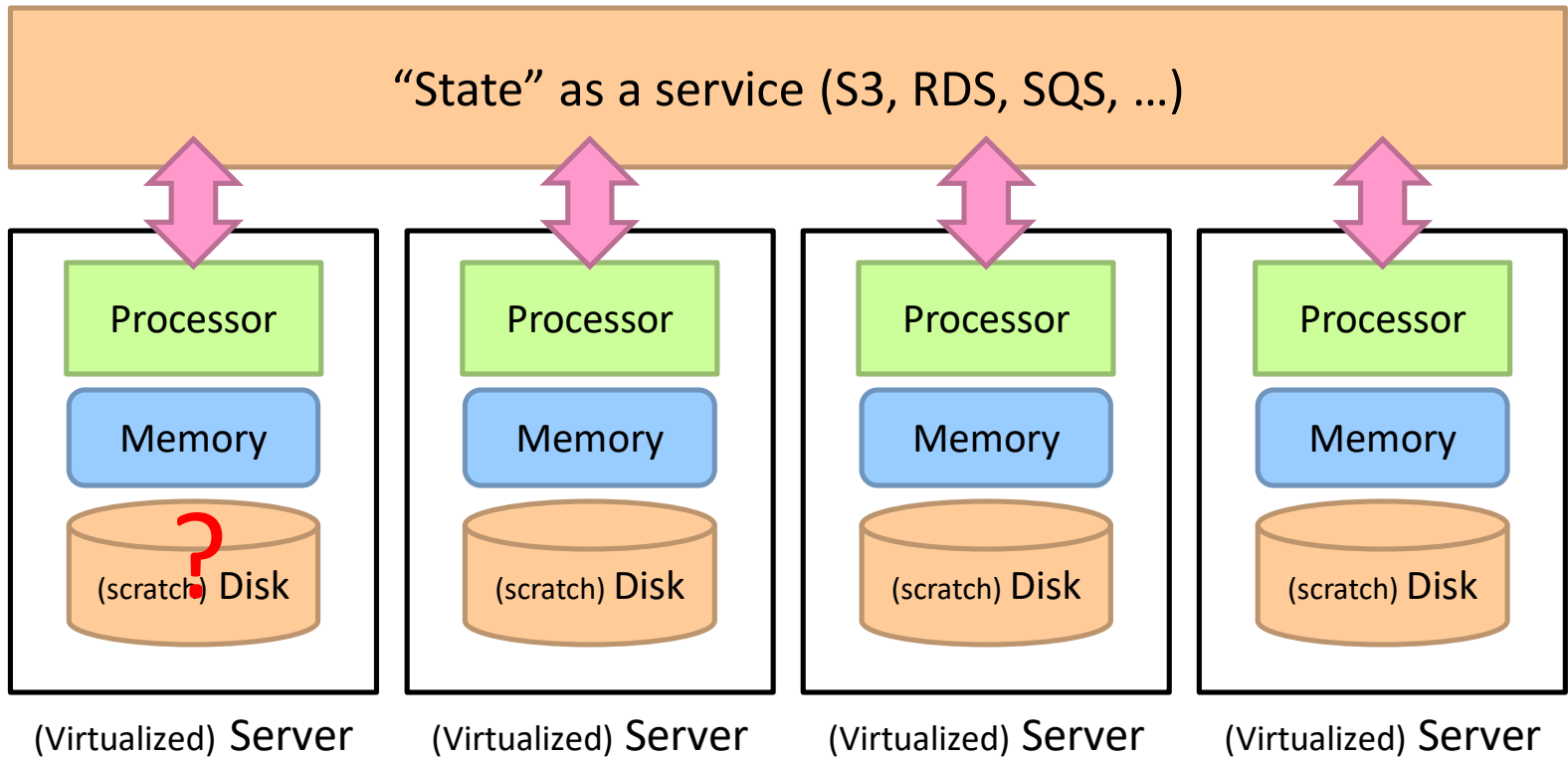
Cloud (I'm going to illustrate with AWS)



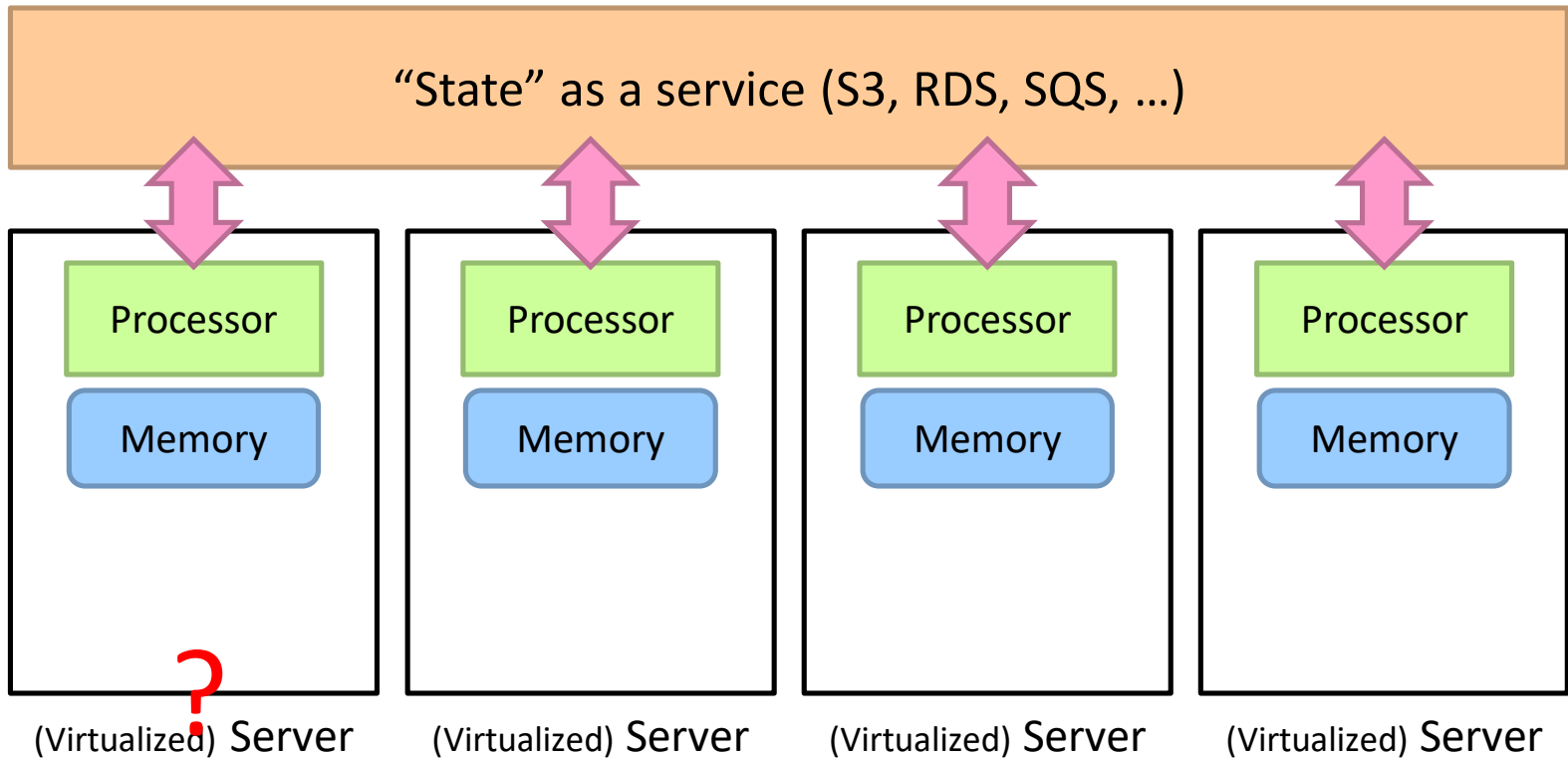
Cloud



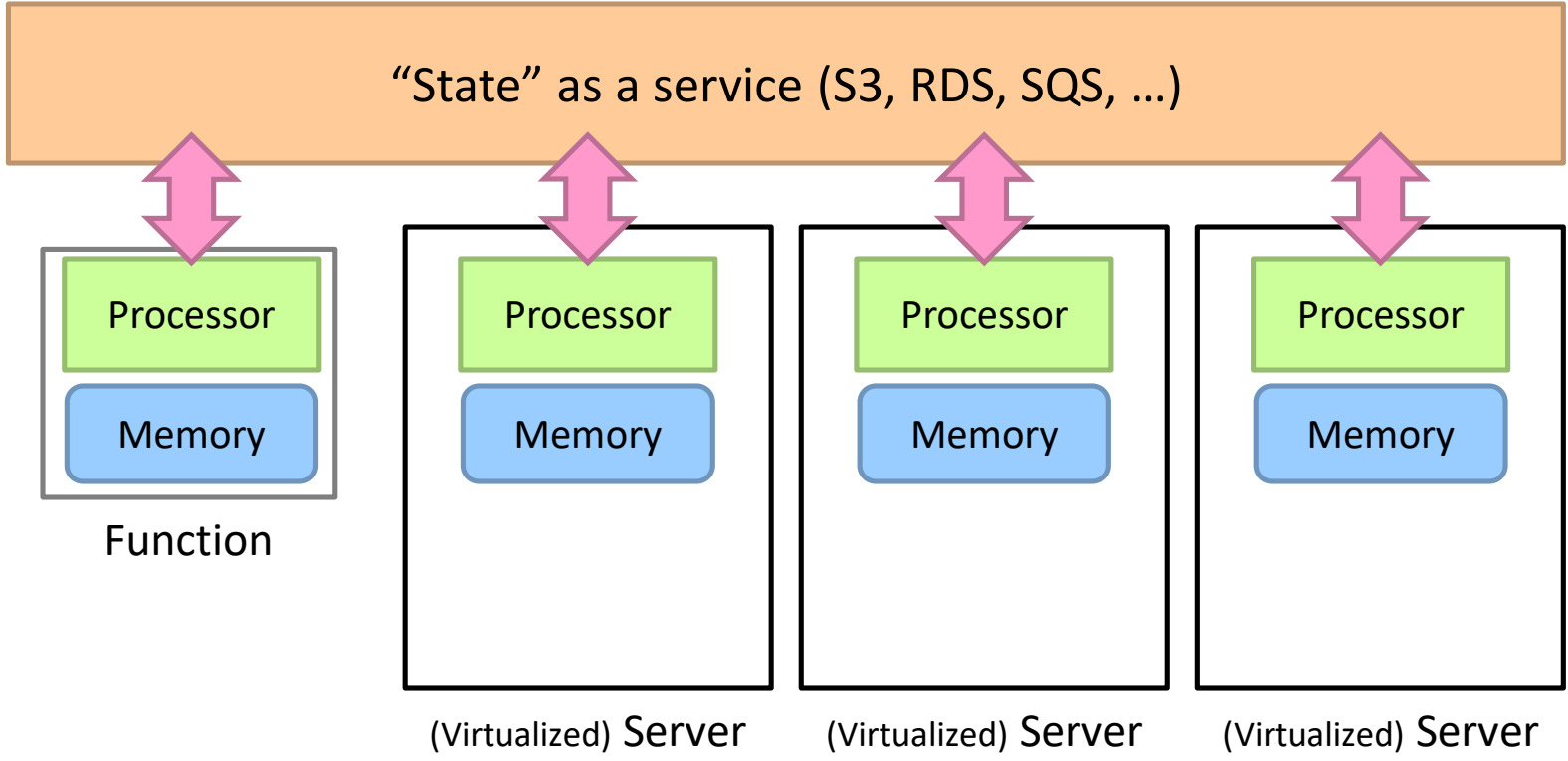
Cloud



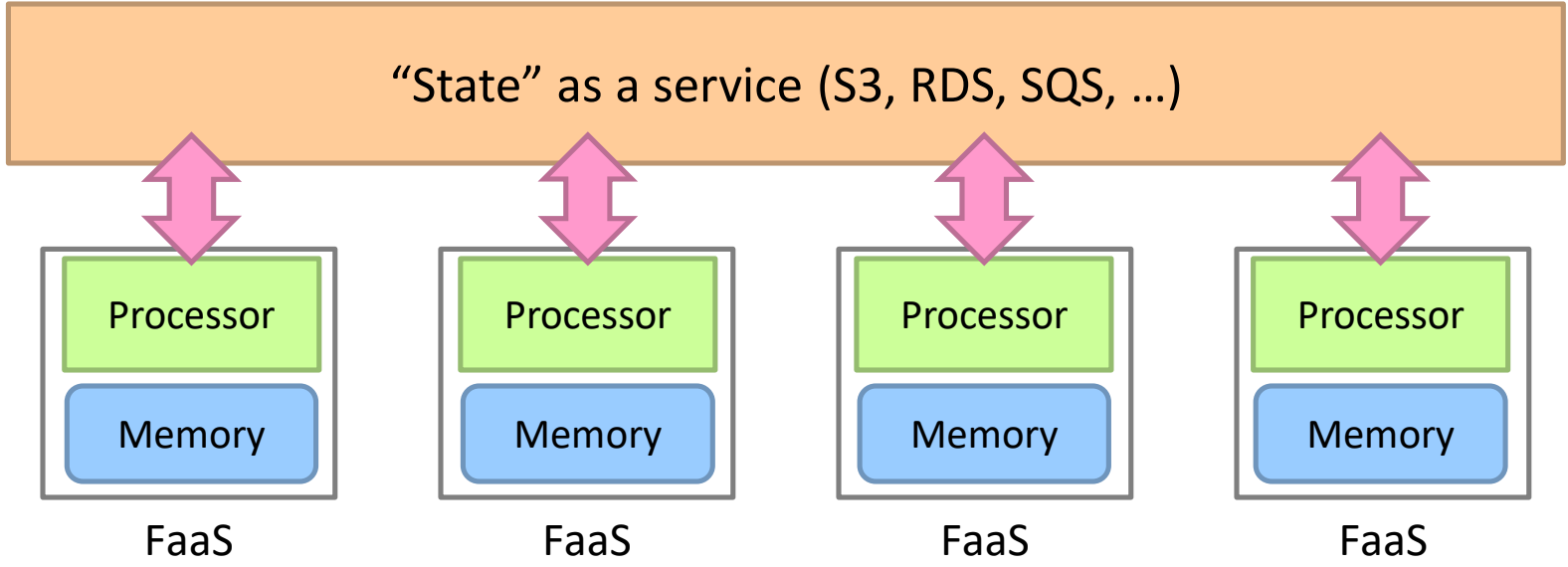
Cloud



Cloud



Cloud



Cloud

Serverless Architectures

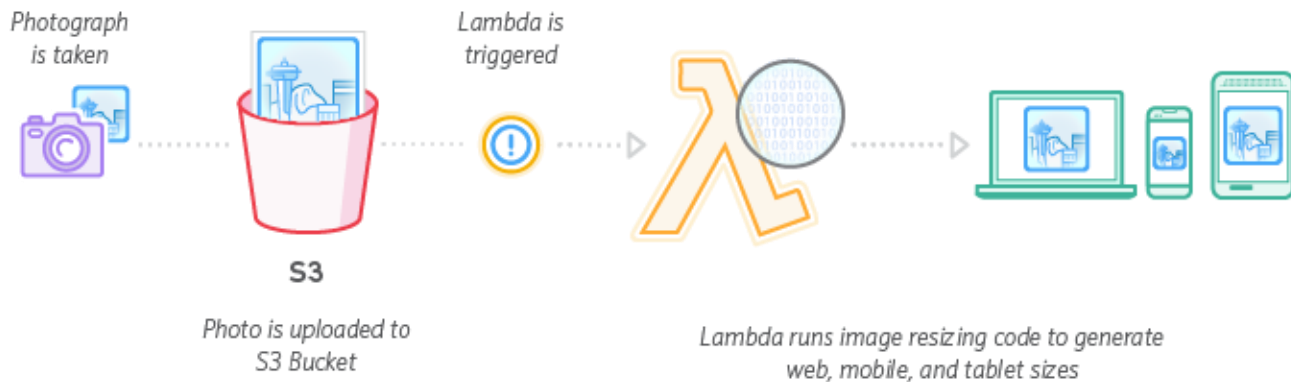
Doesn't mean you don't have servers

Just that managing them is the cloud provider's problem

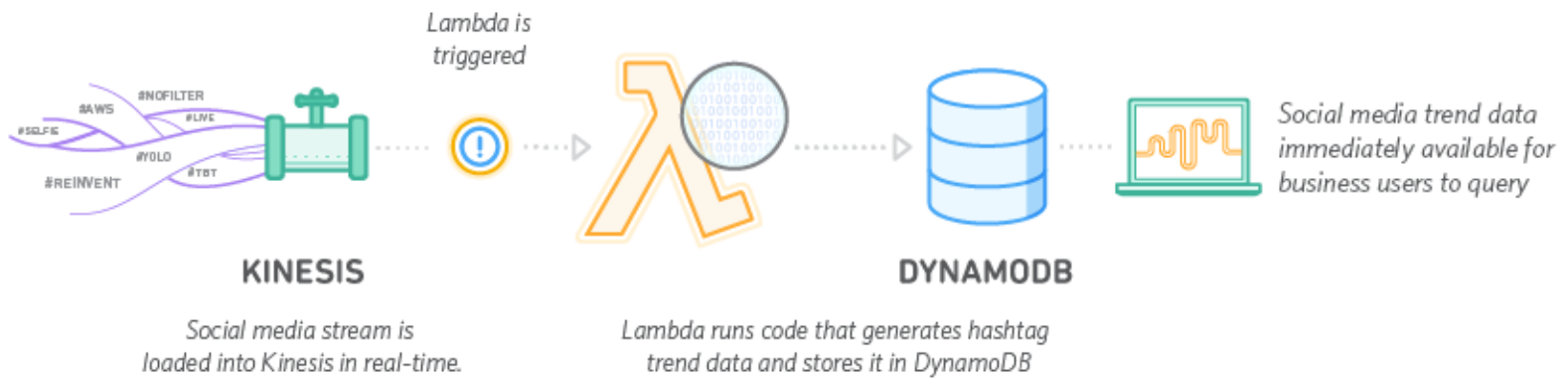
Write functions with well-defined entry and exit points

Cloud provider handles all other aspect of execution

Example: Image Thumbnail Creation



Example: Analysis of Streaming Social Media Data



(Current) Serverless Architectures

Asynchronous, loosely-coupled, event-driven

Functions touch relatively little data

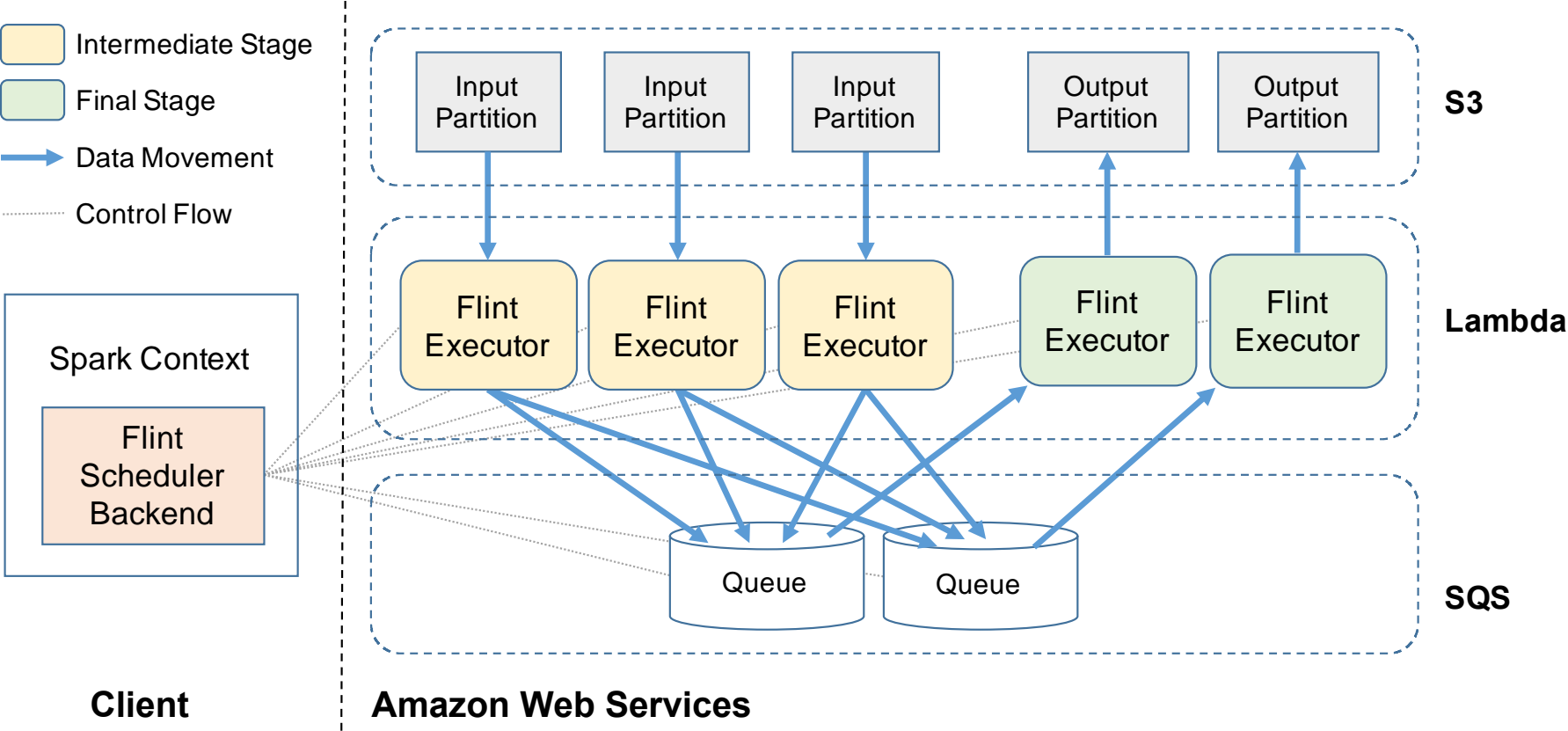
What about serverless data analytics?

Design goal: pure pay-as-you-go, zero costs for idle capacity

Compared to current options?

Flint

PySpark execution backend



The datacenter *is* the computer!

“Big ideas”

Scale “out”, not “up” *

Limits of SMP and large shared-memory machines

Assume that components will break *

Engineer software around hardware failures

Move processing to the data *

Cluster have limited bandwidth, code is a lot smaller

Process data sequentially, avoid random access

Seeks are expensive, disk throughput is good

