CS312 Solutions #5

March 11, 2015

Solutions

1. Define both horizontal and vertical scaling. Explain the differences between them. Give one example of each that is not in the slides.

   A **Horizontal scaling is adding more nodes to a system, where both nodes and system are loosely defined. An example is adding another backend node to a MapReduce cluster.**

   B **Vertical scaling is adding more resources to a node. an example is adding a faster network card.**

   The differences are that vertical scaling is easier and simpler, but cannot scale nearly as far as horizontal scaling can. Horizontal scaling also introduces some abstract complexity to a system that can make it harder to understand, but easier to reason about.

2. Explain the CoreOS update process. What is it modeled after? Do you think this is a good or bad update model? Explain.

   The CoreOS update process is a dual-partition scheme based on Omaha, the update engine for Chrome and ChromeOS. Essentially, there are two partitions, A and B. The machine runs on partition A until an update comes along, and it updates

on partition B, and reboots to use partition B. This provides many advantages, the most important of which is atomic (and therefore safe) rollbacks.

3. Which consensus algorithm does CoreOS use for `etcd`? Give a very brief summary of how the algorithm works.

   CoreOS uses Raft. Raft is designed to be equivalent to Paxos, but subdivided into smaller independent problems. It divides into three primary problems: leader election; log replication; and safety. The paper in which Raft was originally published, *In search of an Understandable Consensus Algorithm* can be found at `http://ramcloud.stanford.edu/raft.pdf`

4. Name two features of the Linux Kernel that makes container technology possible for Linux.

   A **cgroups**

   B **namespace isolation**

5. Describe the primary differences between virtual machines and containers. Why would you use one over the other?

   Virtual Machines require a hypervisor and sometimes hardware support, while containers only require a Linux Kernel with cgroups and namespace support. Virtual Machines typically emulate the whole hardware stack while containers shares the host's hardware and kernel directly (basically, a fancy chroot) thus has a smaller resource requirement. If you need to emulate a full operating system and require the features, then a virtual machine is a good choice. If you're just trying to deploy a small application that needs to be isolated, then a container is a good choice.

6. Name three scenarios that virtual ips are good for increasing redundancy.

Virtual IPs can provide redundancy in routers (and other networking gear), databases that have some form of replication (for providing a single ip for everyone to talk to), and redundant VPNs.

7. Name three ways that horizontal scaling can add complexity. Briefly explain why the complexity is necessary to scale horizontally.

   Horizontal scaling can add complexity in the following ways:

   A **Load Balancing**

   B **Latency**

   C **Consensus**

   Horizontal Scaling almost always requires some increased form of complexity because it often introduces new problems into the domain, such as keeping multiple nodes in the same state.

8. Use a Dockerfile to build the cs312 repo and output it over nginx.

```
FROM nginx
MAINTAINER jme@prismaticgreen.com

RUN apt-get update
RUN apt-get install -y build-essential python-dev python-
    ↪ virtualenv git

RUN git clone https://github.com/osuosl/cs312
WORKDIR /cs312
RUN bash scripts/build.sh
RUN cp -r build/html/* /usr/share/nginx/html
```

9. Create a systemd unit file for the docker image in the previous question.

```
[Unit]
Description=cs312 service
BindsTo=cs312.service
```

```
[Service]
ExecStartPre=-/usr/bin/docker kill cs312
ExecStartPre=-/usr/bin/docker rm cs312
ExecStart=/usr/bin/docker run -p 80:80 --name cs312 cs312/
    ↪ website
ExecStop=/usr/bin/docker stop cs312
```

10. Suppose you have a system running with a single load balancer, three web nodes, and two database nodes. What is the single point of failure? How could you get rid of this single point of failure? After fixing it, what other points of failure might be singular?

    The load balancer is the single point of failure. You could fix it by introducing a second load balancer backed by a virtual ip for failover. After fixing the load balancer, other single points of failure could include things like: power; networking; cooling.