Notes: Basic Cloud Native Blueprint

What is cloud native

- There used to be tons of data centers (takes up power, space, and money)
- Cloud: consolidates all computing needs
- Cloud native - adopts new way of thinking
- In-house: host your own data centers
- CNCF = Cloud Native Computing Foundation (maintain cloud native landscape)

Landscape

Containers - Way to isolate process so it does not mess with other things on the box

- Efficient storage
- Portable
- Store once and then can run many at a time
- Repeatable
- Quick to start-up

1. <u>Container Runtimes</u>
    - Speed, security, weight
    - Type that you use controls how many containers you can fit on your machine
    - Provides security
        - Provides isolations (different levels and different ways)

2. <u>Container Registries</u>
    - Storage and revival
    - Where you store the container image which you can then spin up
    - Where you find the images

3. <u>Container Orchestration</u>
    - Help you operate at cloud scale
    - You don't have to wait for more hardware you just tell cloud that you want more infrastructure
    - To manage 10,000 100,000 containers and how they are working together
    - Deals with containers that are already spun up
    - Like Kubernetes

4. <u>Serverless / Functions as a Service</u>
    - Focus on code instead of infrastructure
    - Like saying here's a piece of code to the cloud service provider "go run this"
    - Run this for 5 minutes at a time everytime ___ (so you are not paying for it while it is running)

5. <u>Infrastructure as Code</u>

- Manage your infrastructure as code
- DevOps - manage your operations (infrastructure) like you manage your dev
- Define what you want your infrastructure to be in code and then it makes it happen

6. Service Mesh
    - A bunch of tools that you need once you start running kubernetes
    - Combination of a bunch of things that you need in one package
        - Like monitoring tools
    - Use sidecar container to monitor applications
        - So don't have to add code to application to monitor it

    Observability Tools
    - Can run these on your own without service mesh
    - Monitoring, logging, tracing, and chaos engineering
    - Concept: know your system and now how your business is running

7. Observability - Monitoring
    - Know what's happening when it happens
    - Putting tools in place to know when things are going wrong or right
    - Instrument an application then analyzing the metrics

8. Observability - Logging
    - Know what happened before in your application
    - Go back in logs

9. Observability - Tracing
    - If something goes wrong follow the trail back to see where it leads/ something was messed up

10. Security and Compliance
    - Policy - make sure policies are in place and enforces those policies
    - Help you have security in cloud native environment
        - Security is different than in non-cloud native because you are hosting things in a more distributed way
            - Pieces of one application spread out with containers

11. Streaming and Messaging
    - Consume data at scale
    - Ex. usecase - Image going into object storage (event) that triggers serverless application to run

12. Remote Procedure Call

- Distributed Systems, Distributed Communication
- When you have huge distributed systems, sometimes you need to call a procedure from somewhere else than where the trigger is
- Help you do remote procedure calling

13. Cloud Native Storage
    - If the system is distributed, where is the data?
        - Sometimes need data closer to the application that is using it then the traditional storage space
    - Way to get data closer to the applications that need it
    - Container does not include the storage - data must be stored separately