the complete n00b's guide to Cloud Security

Cassandra Young aka muteki

@muteki_rtw

Who's "muteki"?

- Azure/O365 SysAdmin with a focus on Security
- Computer Science Grad Student
  (aka glutton for punishment)
- Blue Team Village Organizer
- Jack of All Trades
- Lives for international travel, scuba diving, woodworking, jigsaw puzzles and baking
- Not entirely sure how she got here?

Cassandra Young

Pronouns: she/her

Twitter: @muteki_rtw

LinkedIn: linkedin.com/in/cassandray

GitHub: github.com/muteki-apps

Fundamentals

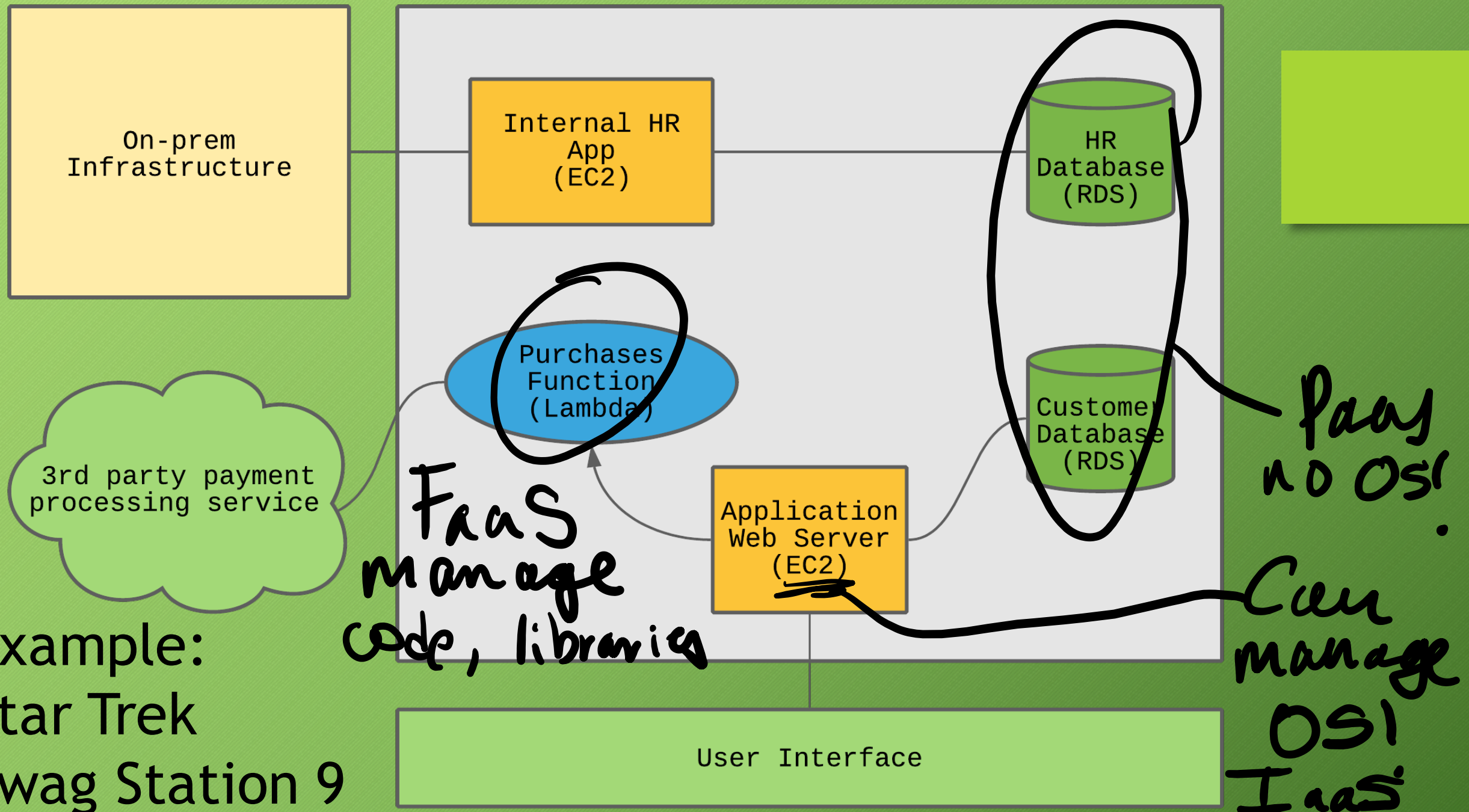
Fundamentals | Take Note!

THINK COMPONENTS

LEARN CODE

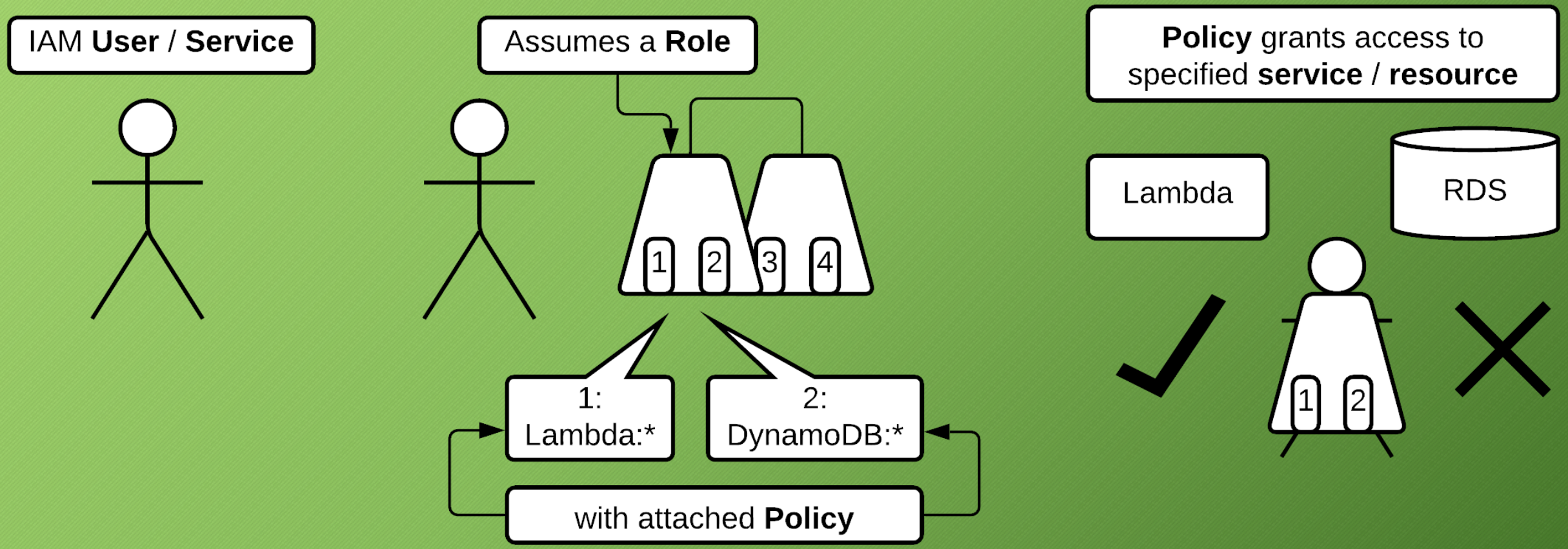
BE FLEXIBLE

Fundamentals | The Shared Responsibility Model

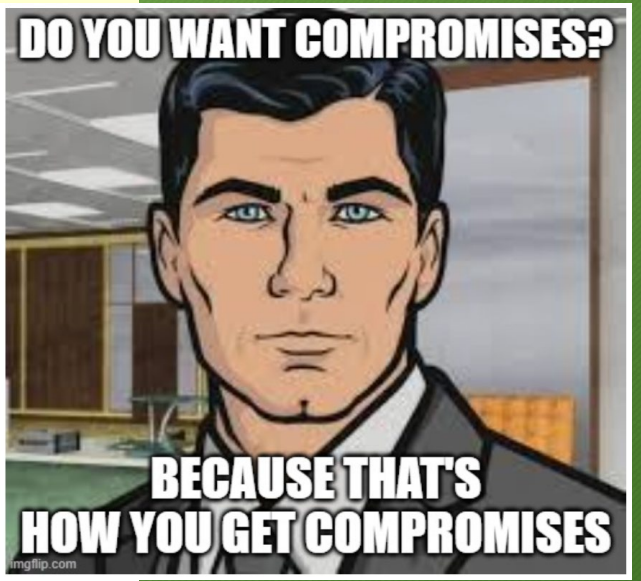


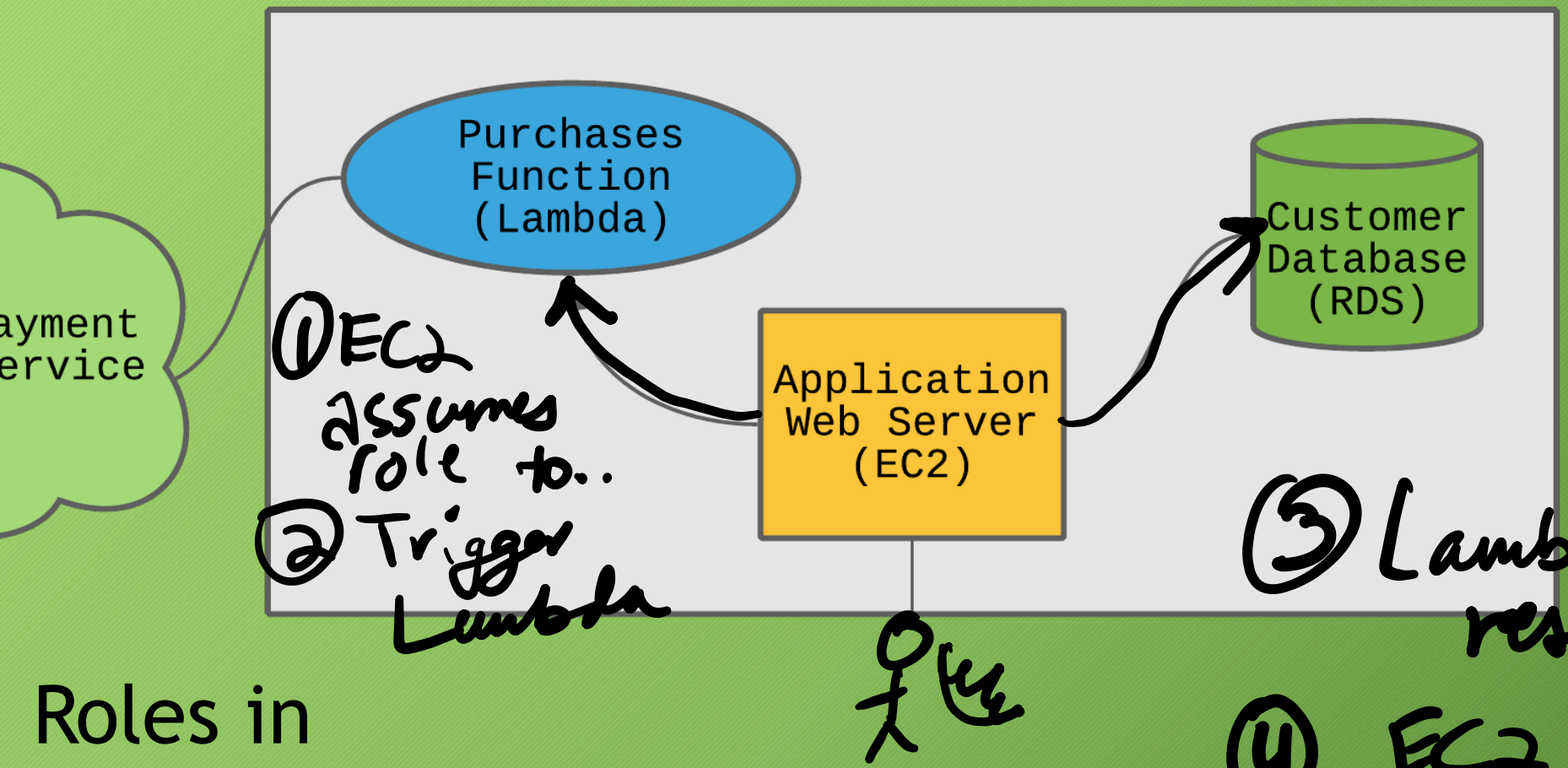
Example:
Star Trek
Swag Station 9

Key Areas | Identity & Access Management

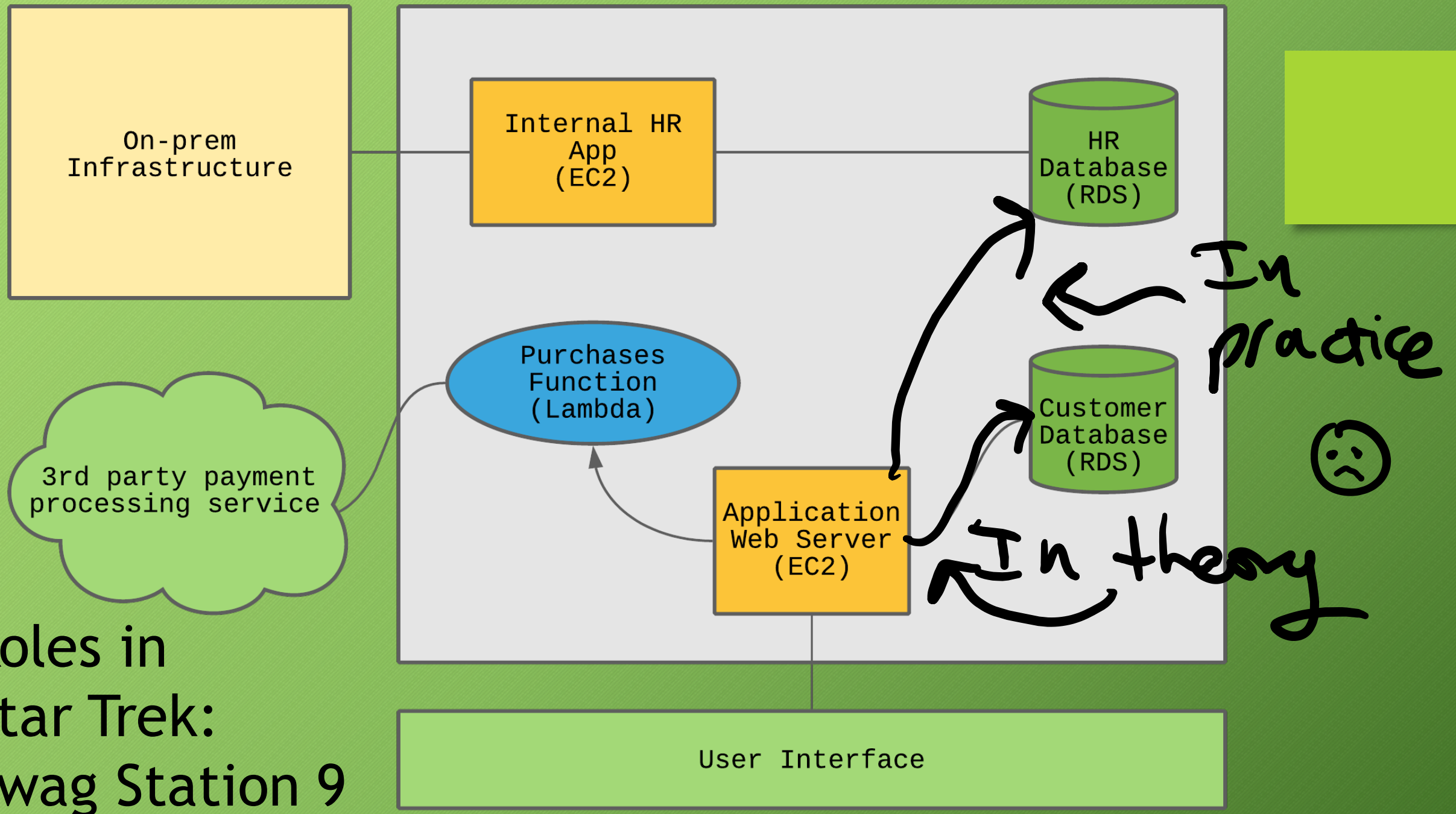


```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "lambda:CreateFunction",
                "lambda:InvokeFunction",
                "lambda:GetFunction",
                "lambda:UpdateFunctionConfiguration",
                .........
                "lambda:UpdateAlias",
                "lambda:UpdateCodeSigningConfig",
                "lambda:UpdateFunctionCode",
                "lambda:CreateAlias"
            ],
            "Resource": "*"
        }
    ]
```



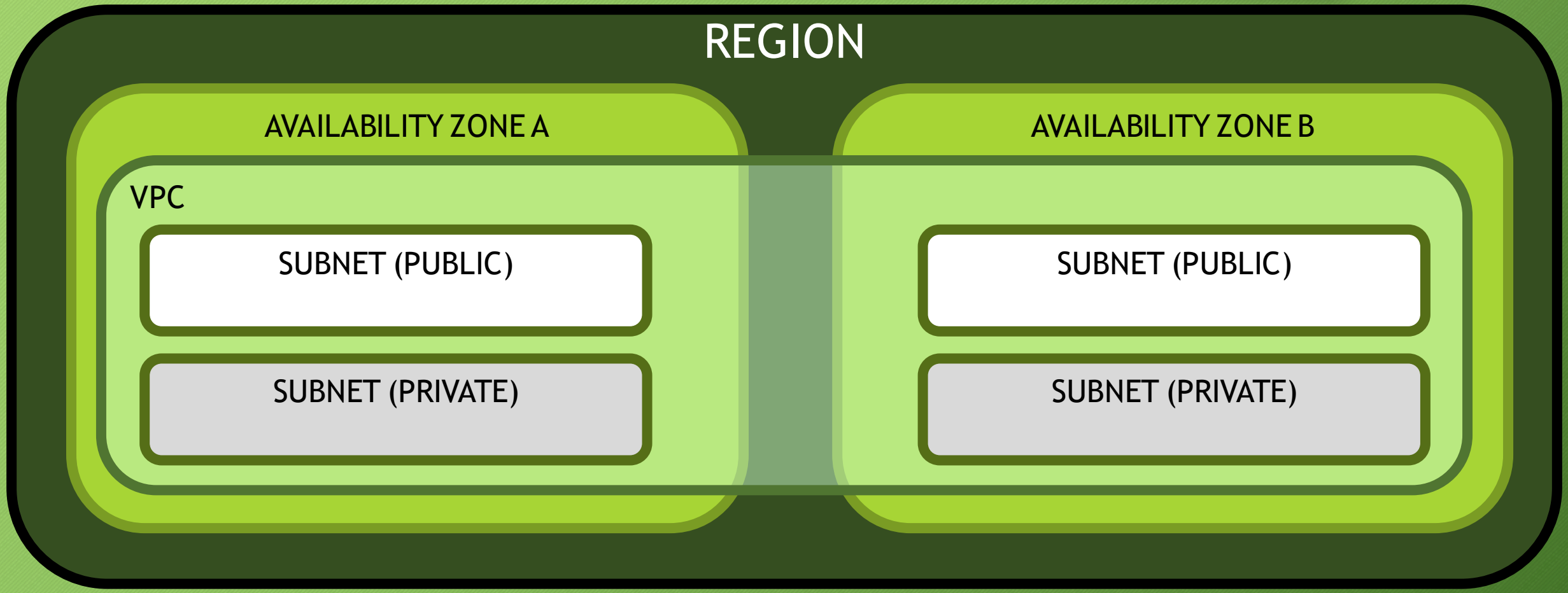


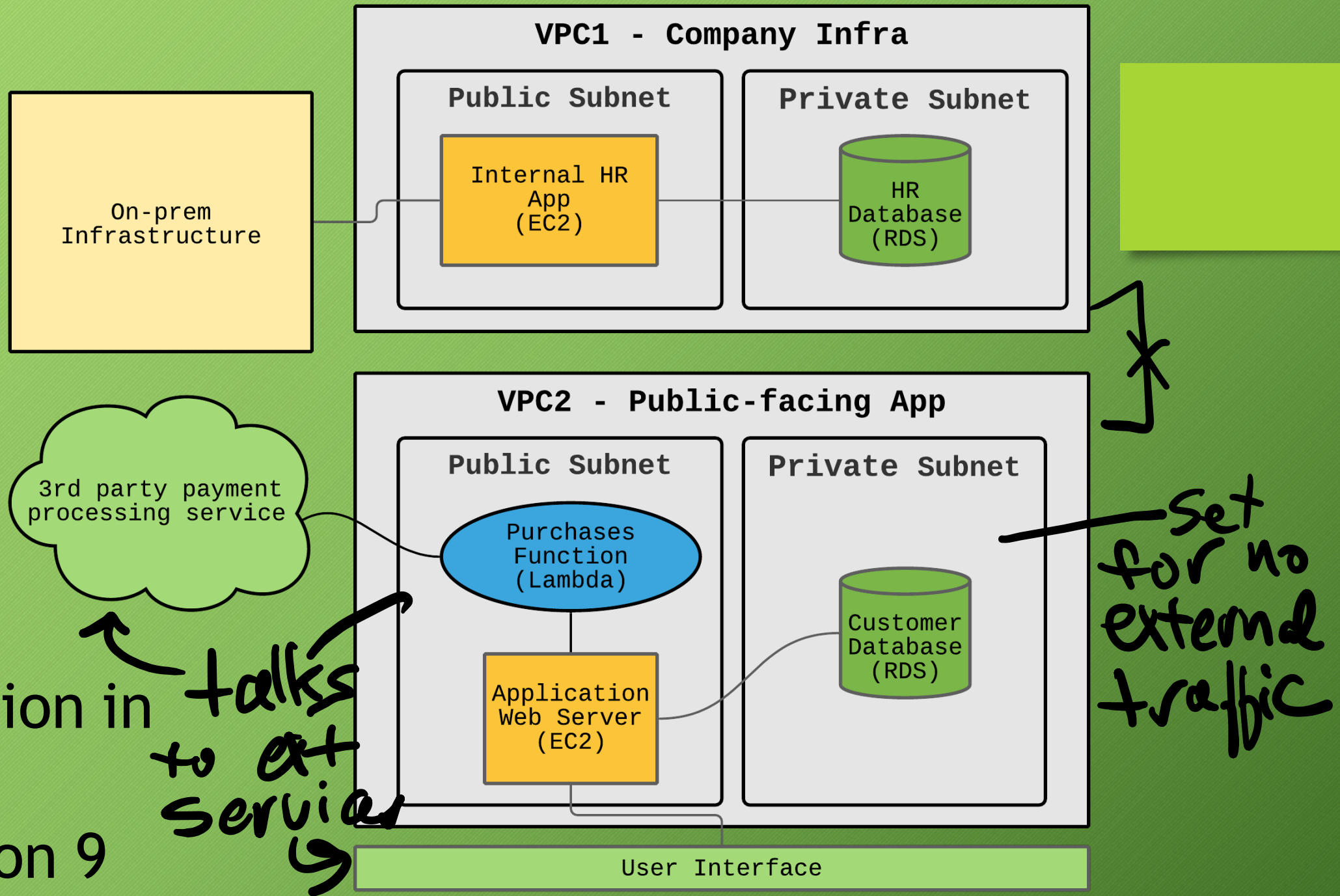
Roles in Star Trek: Swag Station 9



Roles in Star Trek: Swag Station 9

Key Areas | Resource Segmentation | AWS Networking





Segmentation in Star Trek Swag Station 9


Inbound rules
Outbound rules
Tags
Inbound rules
Type
Protocol
Port range
Source
MYSQL/Aurora
TCP
3306
sg-03ecf9b16a40ebef8 (DBGroup)
AVAILABILITY ZONE A
VPC
Security Group A
(DBGroup)
EC2
EC2
Security Group B
RDS DB
RDS DB

Key Areas | Data Protection | AWS Examples

AWS documentation details service endpoints and the protocols they accept:

Layer	Service	Encryption in Transit	Encryption at Rest
FaaS	AWS Lambda	HTTPS only	Env. variables encrypted at rest (RAM only, no permanent storage)
PaaS	AWS DynamoDB	HTTPS only	Default: encrypted, AWS owned key AWS managed
IaaS	AWS EC2	HTTP and HTTPS	Default: not encrypted, options available if configured

https://docs.aws.amazon.com/general/latest/gr/aws-service-information.html

Key Areas | Data Protection | AWS S3 Example

SSE-S3:

Server-Side Encryption
with Amazon S3-Managed Keys

SSE-KMS:

Server-Side Encryption with
Customer Master Keys (CMKs)
Stored in AWS Key Management Service

SSE-C:

Server-Side Encryption with
Customer-Provided Keys

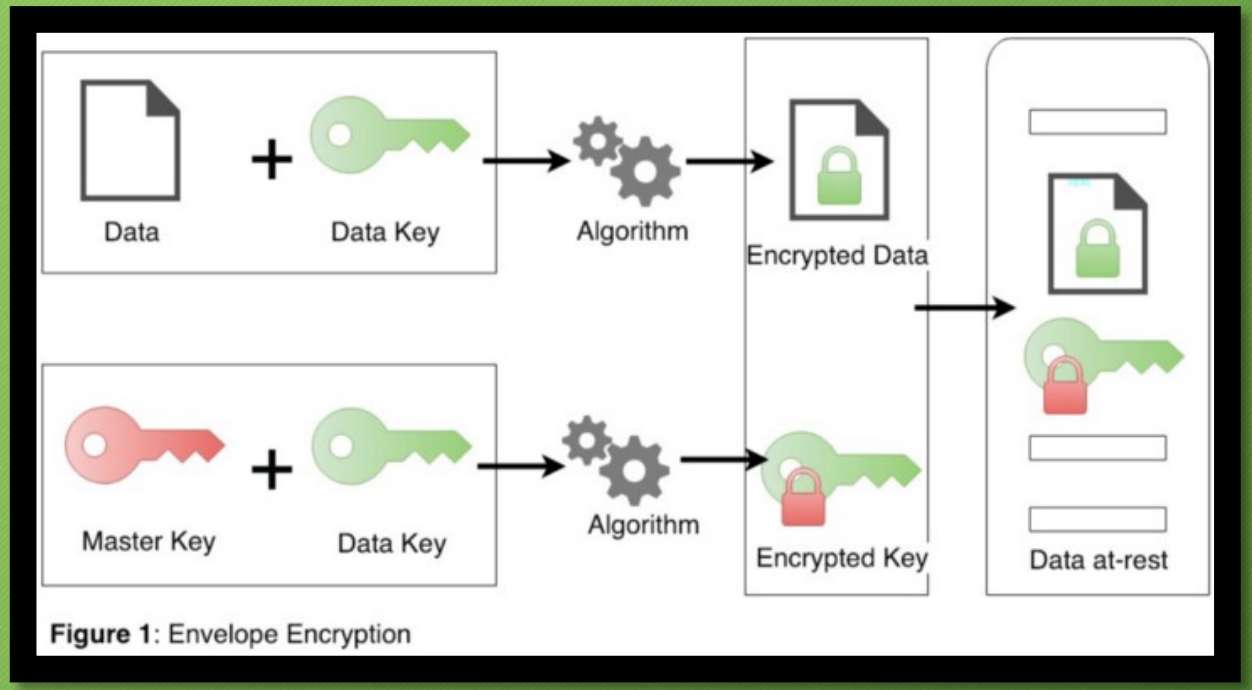


Image Source: https://medium.com/@thandototo/securing-encryption-keys-in-the-cloud-part-1-aws-kms-726b5c3a2083

Key Areas | Data Protection | HIPAA

- Customers who execute an AWS BAA may use any AWS service in an account designated as a HIPAA Account, but they may only process, store and transmit PHI using the HIPAA-eligible services defined in the AWS BAA.
- AWS maintains a standards-based risk management program to ensure that the HIPAA-eligible services specifically support HIPAA administrative, technical, and physical safeguards.
- AWS's BAA requires customers to encrypt PHI stored in or transmitted using HIPAA-eligible services in accordance with guidance from the Secretary of Health and Human Services (HHS).
- AWS offers a comprehensive set of features and services to make key management and encryption of PHI easy to manage and simpler to audit, including the AWS Key Management Service (AWS KMS). Customers with HIPAA compliance requirements have a great deal of flexibility in how they meet encryption requirements for PHI.

https://docs.aws.amazon.com/whitepapers/latest/architecting-hipaa-security-and-compliance-on-aws/introduction.html

Key Areas | Visibility

Challenges

- Format
- Completeness
- Correlation

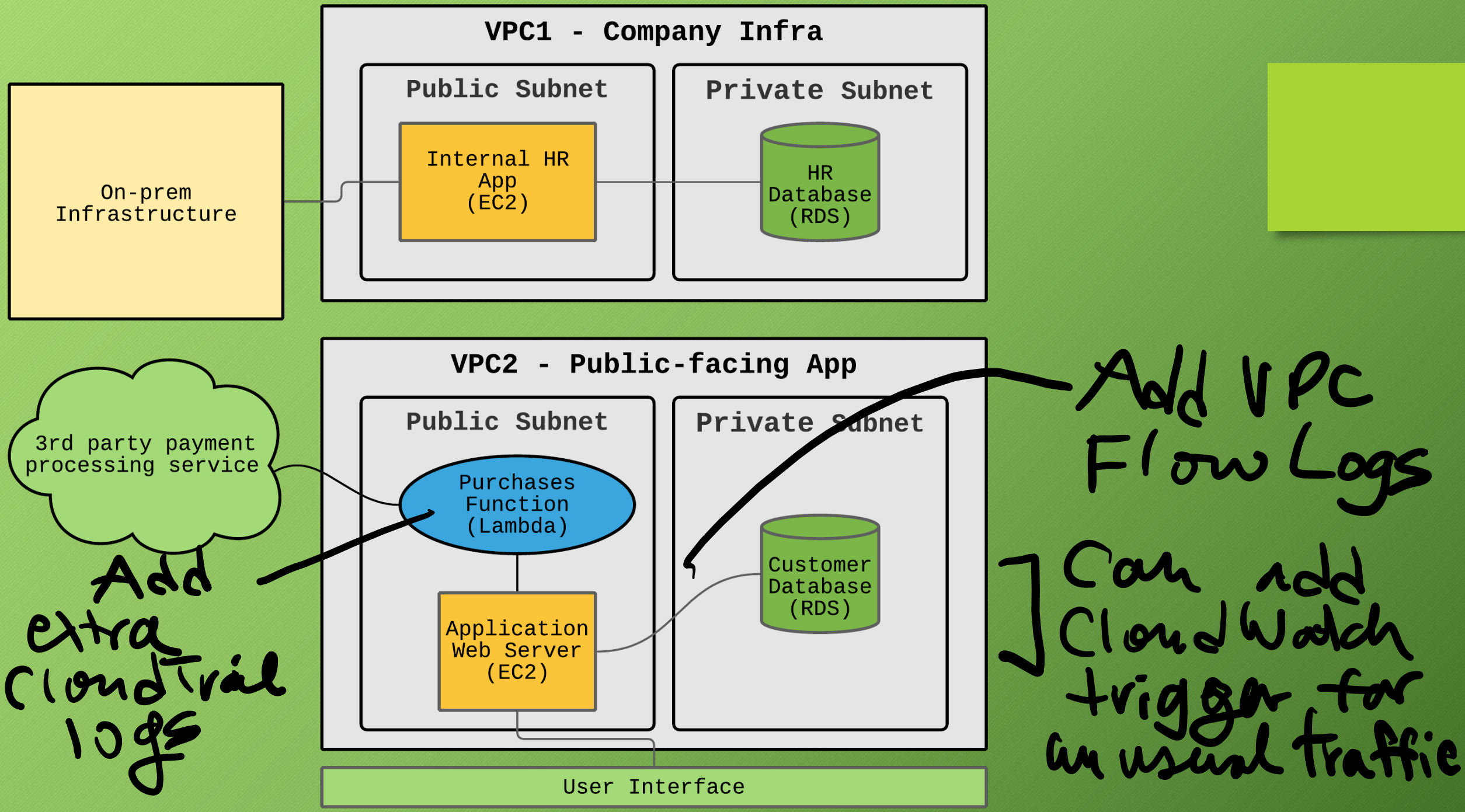
Approaches

- Analyze
- Expand
- Reassess

Key Areas | Logging | AWS CloudTrail, VPC Flow Logs

- Track API calls with **CloudTrail**
  - Defaults ‘on’ for many services
  - Boto3 library support
  - Integrate with CloudWatch Logs, or Splunk, SIEMs
- Log network interface traffic with **VPC Flow Logs**:
  - Log by VPC, subnet, or network interface
  - Use with other AWS services
- Monitor with **CloudWatch**:
  - Collect and analyze with dashboards
  - Build alerts, trigger other AWS services

https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-aws-service-specific-topics.html


VPC1 - Company Infra
Public Subnet
Private Subnet
Internal HR App (EC2)
HR Database (RDS)
On-prem Infrastructure
VPC2 - Public-facing App
Public Subnet
Private Subnet
3rd party payment processing service
Purchases Function (Lambda)
Application Web Server (EC2)
Customer Database (RDS)
User Interface
Add VPC Flow Logs
Can add CloudWatch trigger for unusual traffic
Add extra CloudTrail logs

Key Areas | Logging & Monitoring | More AWS Tools?

- Key Management Service (KMS) - Encryption
- Amazon Macie – Data Classification
- CloudFormation – Templatized Deployment
- Config – Inventory / Config Tracking
- Control Tower – Multi-Account Security
- GuardDuty – Threat Detection
- WAF & Shield – DDoS, Web Traffic Protection
- Security Hub – Security & Compliance
- Inspector – App Security
- Any of the tons of 3rd party and opensource tools out there?

…but which services are best for ***your*** organization?

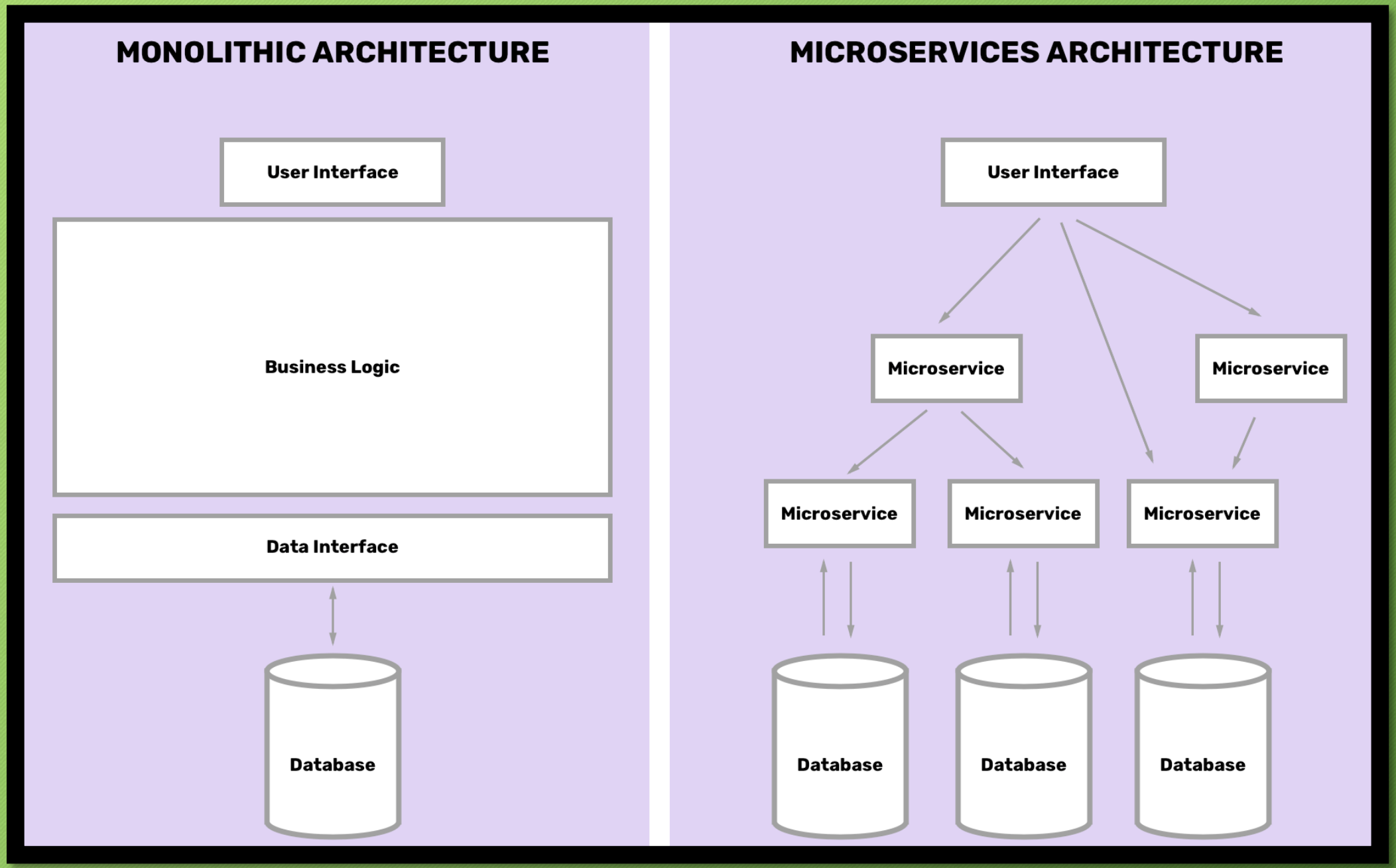
Leveling Up

Leveling Up | Serverless, Microservices, Containers… Oh My!

What does Microservices mean?

- Microservices Architecture is a **design pattern**
- **Collection** of loosely-coupled components
- Development and upgrade focus shifted to **component** level

"The microservices architecture is defined as collection of smaller services arranged together in a workflow, often utilizing a serverless platform. Microservices architecture … [connects] these services or functions in a loosely coupled manner, often hosted separately and communicating via REST APIs or other lightweight protocols."



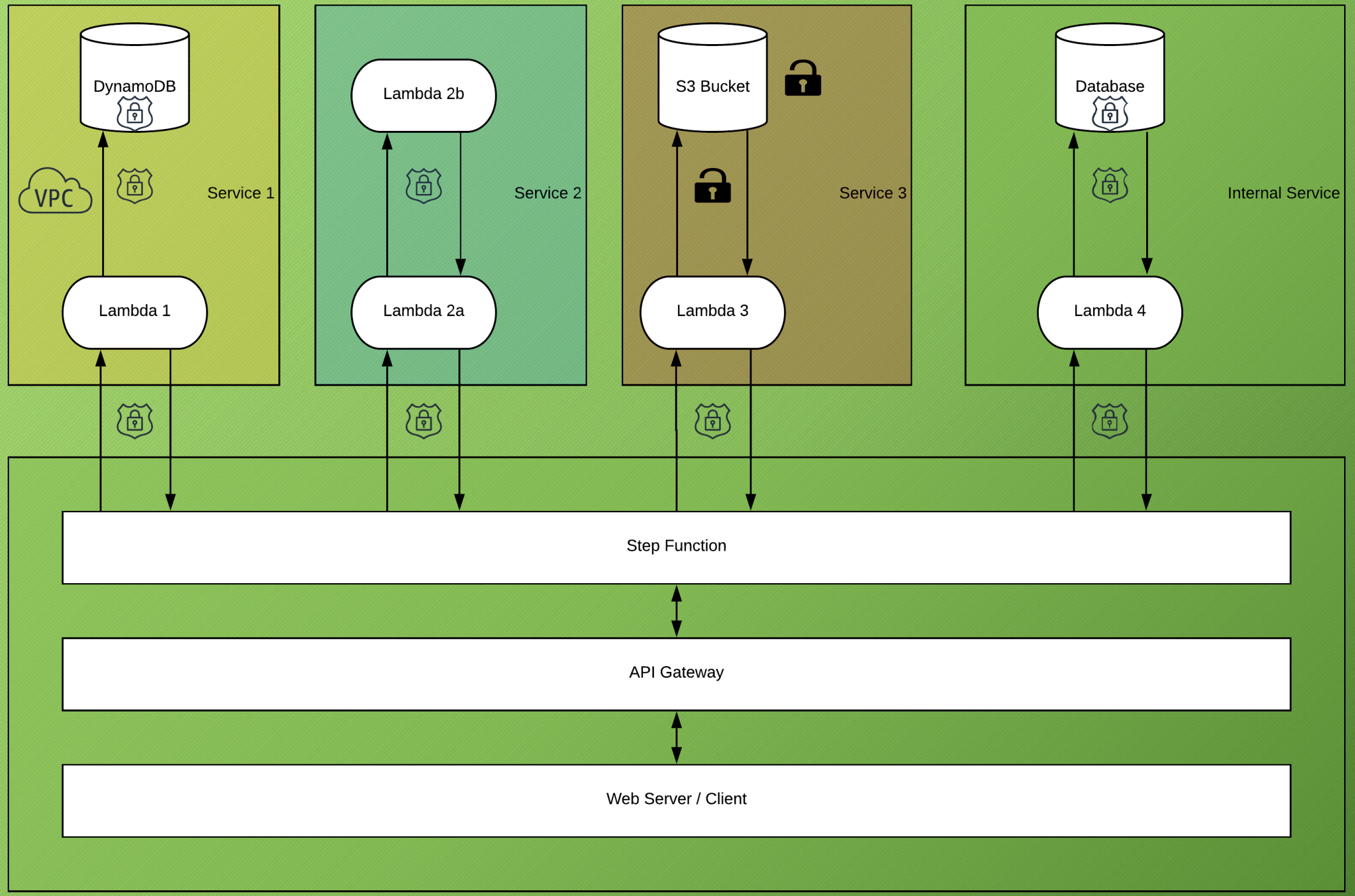
- Image: algorithmia.com/blog/introduction-to-serverless-microservices

Leveling Up | Serverless, Microservices, Containers… Oh My!

What is Serverless?

- Execution model which abstracts away infrastructure
- Code-focused, obfuscating OS & dependencies
- Simplified development process
- Event-driven, primarily accessed via APIs

AWS Lambda is Serverless Function-as-a-Service!


DynamoDB
VPC
Service 1
Lambda 1
Lambda 2b
Service 2
Lambda 2a
S3 Bucket
Service 3
Lambda 3
Database
Internal Service
Lambda 4
Step Function
API Gateway
Web Server / Client

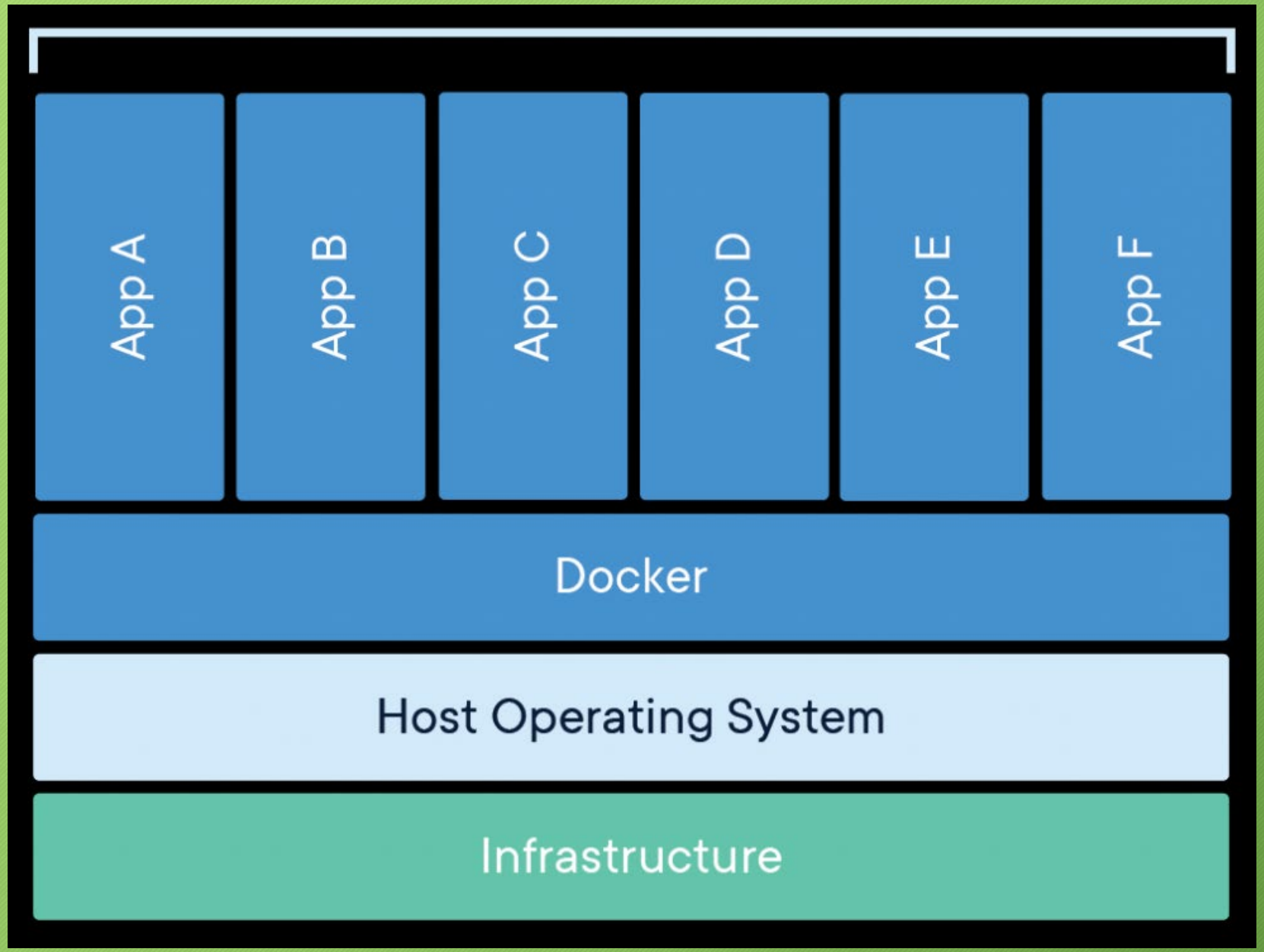
Leveling Up | Serverless, Microservices, Containers… Oh My!

What is a Container?

- Abstraction of the application layer
- Isolates processes from the kernel
- Lightweight, easy to orchestrate

"A container is [packaged] up code and all its dependencies [design to run] quickly and reliably from one computing environment to another. … Containers isolate software from its environment and ensure that it works uniformly despite differences for instance between development and staging."

https://www.docker.com/resources/what-container



How's that different from a Virtual Machine?

No guest OS or hypervisor!

…virtualize the OS instead of the hardware.

https://www.docker.com/resources/what-container

Leveling Up | Serverless, Microservices, Containers… Oh My!

- Container images are open source; proceed with caution
- Yes, you can break out of a container!

- Microservices' multiple components == multiple permission sets
- Yes, you can break out of serverless functions!

…a microservices application
is as secure as its least secure code,
& its most lenient permissions boundary.

Julia Evans
@b0rk

seccomp-bpf

21

all programs use system calls

rarely-used system calls can help an attacker

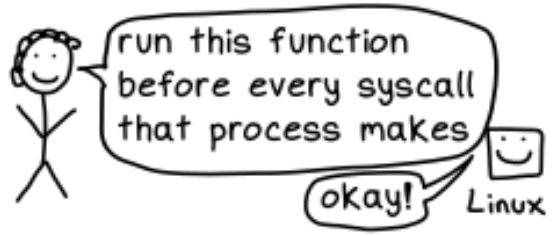
request_key

reboot

process_vm_readv

read memory from another process

seccomp-BPF lets you run a function before every system call



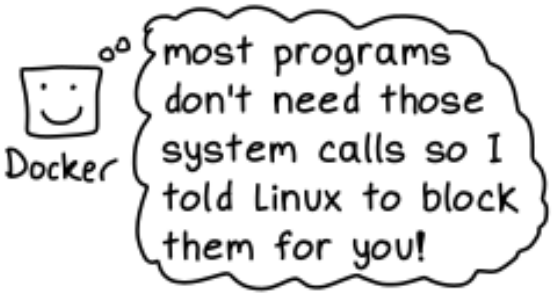
the function decides if that syscall is allowed

example function:

```
if name in allowed_list {
    return true;
}
return false;
```

this means the syscall doesn't happen!

Docker blocks dozens of syscalls by default



2 ways to block scary system calls

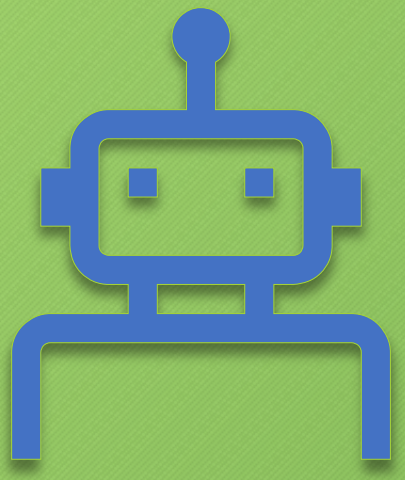
1. limit the container's capabilities
2. set a seccomp-bpf whitelist

You should do both!

TL;DR: Permissions, permissions, permissions!!

https://wizardzines.com - programming zines by Julia Evans(@b0rk)

Leveling Up | Orchestration & Automation



- Replaces repetitive **tasks (automate)**
  & manual management of **systems (orchestrate)**
  to code & config files
- Enables **flexible, scalable** apps & security infra
- When does it makes sense to automate & orchestrate?
- Examples: Terraform & Ansible, CloudFormation (AWS)

Infrastructure as Code!

Are we there yet??

Lessons Learned

- You can easily go 0-60, from simple logging to orchestration!
- Cloud services offers varying levels of access & control
- Design your Cloud architecture based on connected components
- IAM and permissions are the new perimeter

Putting It All Together

START SIMPLE

RE-EVALUATE

LEVEL UP

tips for n00bs

- Lay Groundwork
- Learn by Doing
- Collaborate

so long, and thanks for all the fish!

-Cassandra aka muteki

@muteki_rtw