AWS Security Training

Sudarshan Narayanan

we45

Agenda

- AWS Intro and Basics
- A Gentle Introduction to Terraform and Boto3
- AWS Security Foundations - IAM, Security Groups
- S3, Dreaded S3!
- AWS CloudTrail and Cloudwatch => Logging and Monitoring
- AWS - Vulnerability Assessment using Inspector

Things I have to say

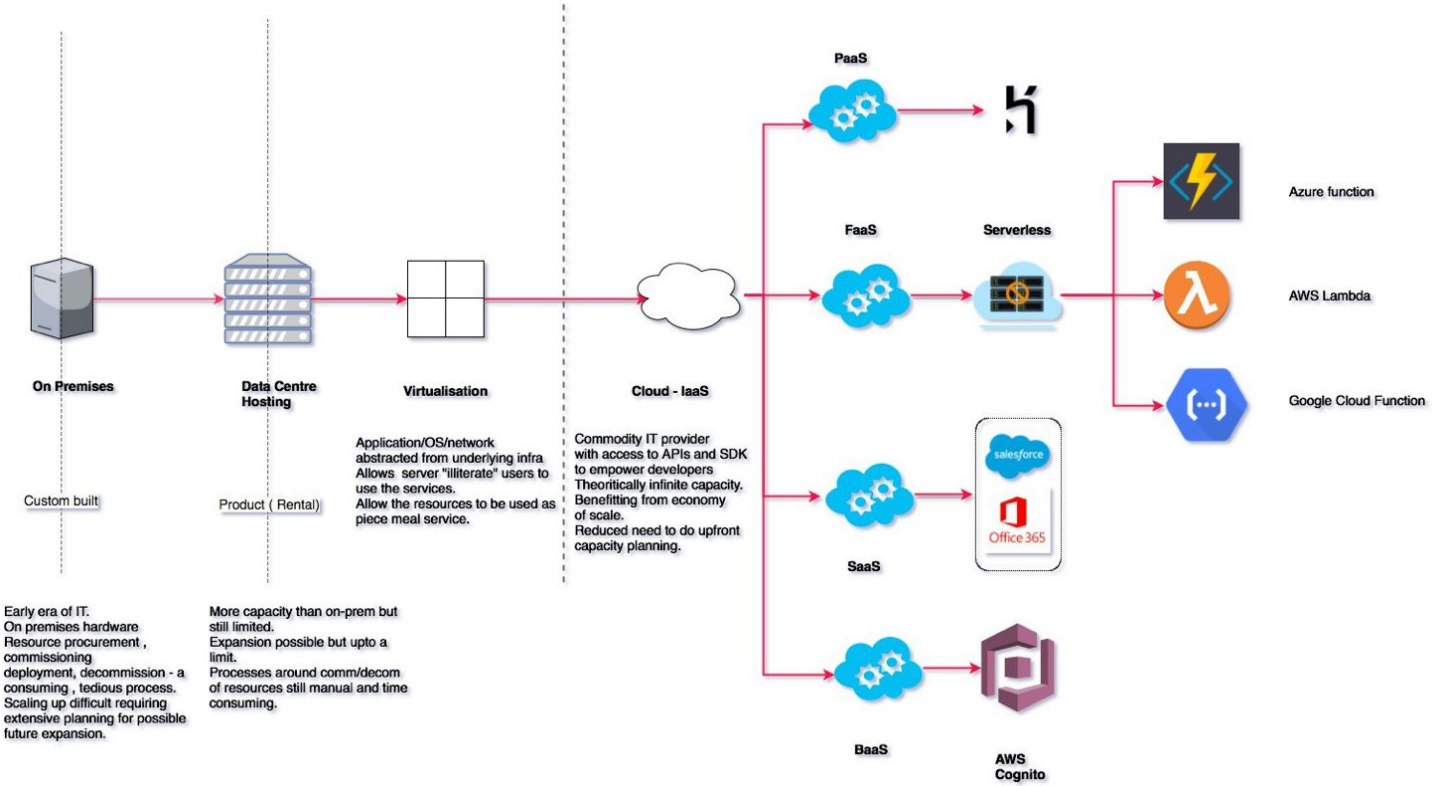
- Please DON’T use this code in production….For purposes of training only…
- The code is given as is, no warranties, etc…
- Please feel free to use the code, play around with it and hack it as you would like.

we45

Before we start....

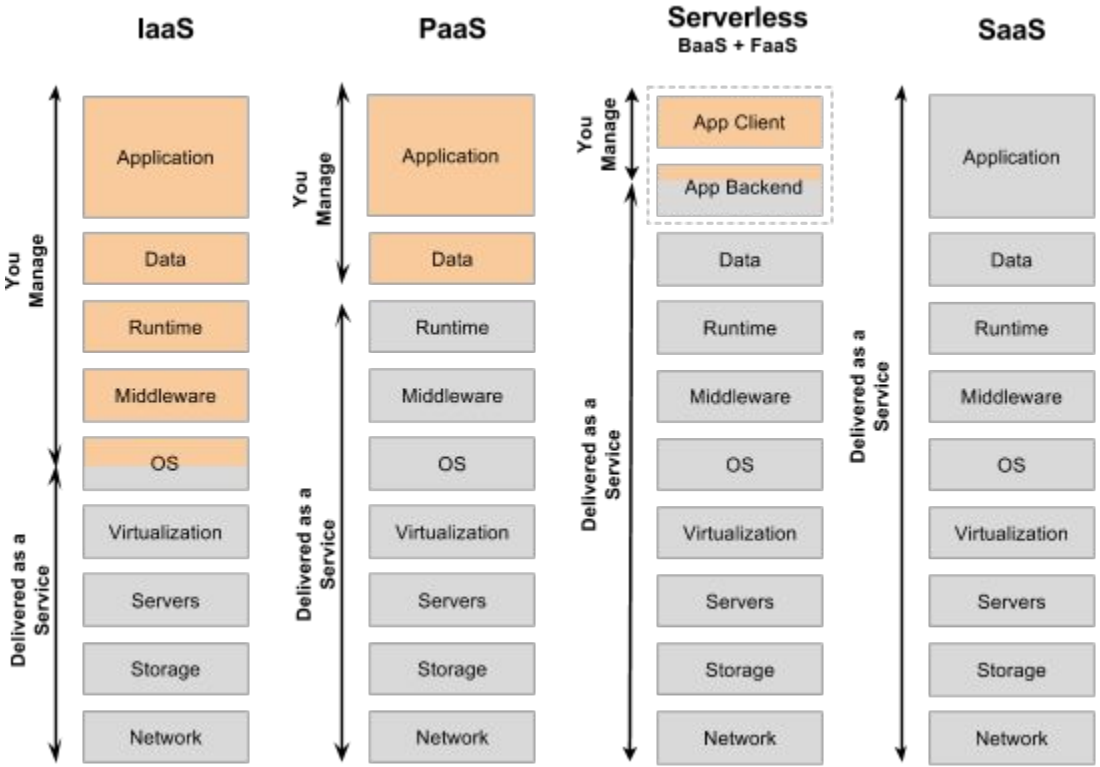
we45

Cloud and Cloud Services - A History



we45

Types of Cloud Computing - Service

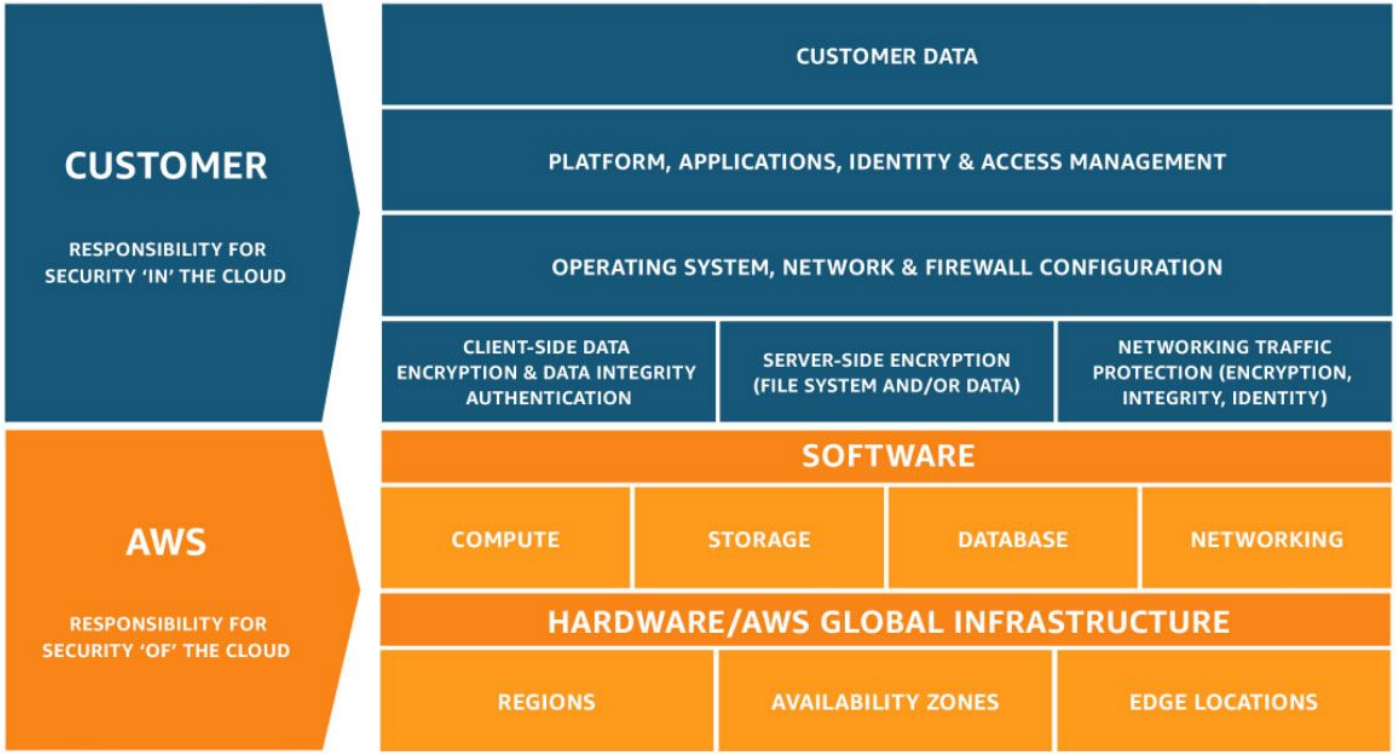


we45

Accelerators of Cloud Computing

- Desired state systems
- Abstracting infrastructure layer of computing
- Fast, Reliable and Scalable

Shared Responsibility Model



Lab Setup

- VM => preconfigured with required tools
- You will be quickly setting up and destroying resources on AWS
- You should choose a region where you don’t have production workloads
- The deployment will not touch existing resources
- Each lab has detailed explanations and steps for execution

Lab: AWS CLI Setup

we45

A Gentle intro to Terraform

we45

we45

What is Terraform?

- Create, Change and Destroy Infrastructure
- Concept of “Providers” => Different Cloud Providers and Infrastructure as Code solutions
- Preview/Plan changes before applying
- Resource Graph and Parallelization

Terraform - Operation and Config

- Terraform reads configurations from *.tf files
- Resources to be provisioned are denoted with the `resource` component
  - Example `resource “aws_instance” “web-server”`
- Data Sources => Allow data to be fetched or computed for use elsewhere in Terraform
  - Example `data “aws_ami” “web-ubuntu”`
  - Used as `${data.aws_ami.web-ubuntu}`
- Providers => Modules that allow you to provision/manipulate/deprovision resources on cloud/compute environments:
  - Example: aws, digitalocean, azure are providers

Lab: Terraform Basics

AWS - IAM and Security Model

we45

Identity and Access Management - A Primer

- Definition: “Enables the RIGHT **individuals** to access the RIGHT **resources** at the RIGHT **times** for the RIGHT **reasons**”
- Drives the entire security model for your public Cloud environment
- IAM controls access and dictates access privileges to the your entire AWS instance

AWS IAM Terminology

- Policy: A set of rules that dictate access to resources within the AWS account.
  A Policy is governed based on a **`Statement`** that is described with:
  - Resources => Resources that the someone has access to
  - Actions => The type of actions that are bound to the access to said resources
  - Effect => Allow or disallow access to the resources with the specific actions
  - Version => For version controlling the policy as required
- Group: Aggregation of Users who can be managed with policies
  - Examples: Developers (Group) with users, who can only deploy AWS Lambda Functions into a Dev Namespace
- NOTE: IAM is not bound to region. It works across AWS Regions

AWS IAM Terminology - Contd

- **ARN** (Amazon Resource Name): Globally Unique Identifier:
  - `arn:aws:serviceregionaccount_IDrelative_ID`
  - Access to most resources within AWS is defined based on the resource's ARN value
- IAM **User**: Refers to a user to your AWS instance. Access can be provided programmatically or through the console OR both
- IAM Role: Similar to service accounts, for applications or instances, where you can assign a role to an instance/application and attach policies governing the access to resources for those instances or applications
  - For example, Role: **`functionRole`** with limited access to DynamoDB can be assigned to a function that you deploy on AWS Lambda
  - With Roles, you don't need to hardcode/provide credentials as credentials are provided by AWS to the instance

AWS Policy Example

```
{
    “Version”: “2012-10-17”,
    “Statement”: {
        “Effect”: “Allow”,
        “Action”: [
        “iam:AddUserToGroup”,
        “iam:RemoveUserFromGroup”,
        “iam:GetGroup”
        ],
        “Resource”: [
        “arn:aws:iam::609103258633:group/Developers”,
        “arn:aws:iam::609103258633:group/Operators”
        ]
    }
}
```

AWS Multi-Step Breaches

we45

- EC2 => Steal IAM creds => Access RDS => Access S3 => Delete Logs
- Lambda => Steal IAM creds => Access DynamoDB (steal info) => Delete Logs

we45

AWS IAM Security Hygiene

- Never store AWS Keys and Secrets in Git repos, etc
- Always use Strong password policies and configure them as mandatory for ALL users in AWS
- Consider using Multi-Factor Authentication
- Consider “throwing away root”
- Use on HTTPS for accessing AWS through API

Lab: Create and Attach Role

we45

we45

Amazon AWS S3

Vulnerabilities and Best Practices

we45

Quick Tour of Object Storage

- Data is stored as Objects, as opposed to Files in a File System
- Or as Blocks in a Block Storage System (Sectors and Tracks)
- Data (Objects) have:
  - Metadata
  - Unique Identifiers
- Idea is to abstract the lower layers of storage:
  - Increase speed of Developers,
  - Reduce Effort of Administrators

Common Use-Cases: Amazon S3

- S3 = “Simple Storage Service”
- Backup and Storage
  - Backup for Application Data, Objects and Files
  - Storage for Application Data, Objects and Files
- Media Hosting
  - Video Streaming
  - File Hosting/Sharing
- Software Delivery
  - Downloadable Software Applications
- Application Hosting
  - Static Websites/Apps
- Distributed Delivery of Content - CloudFront + S3

S3 and Security Issues

- Most Issues related to S3 - Largely relate to Access Control
- According to statistics by SkyHigh Networks, 7% of S3 Buckets have unrestricted public access
    - Users can read/download content from the buckets
    - Users might even be able write data into the buckets
-

UpGuard Breach Analysis Blog What are Cloud Leaks? Our Product What is Cyber Resilience? About Us

Data of 14 Million Verizon Customers Exposed in Server Snafu

By Catalin Cimpanu July 12, 2017 04:25 PM 0

Sensitive data for around 14 million Verizon customers was exposed online because a third-party contractor forgot to limit external access to an Amazon S3 server.

Threats/Attacks against S3 Deployments

- Subdomain Takeover
- Public Read/Write on Amazon S3 Buckets - Sensitive Information
- Overly-Permissive Access Control Rules on S3 Deployments

Subdomain Takeover

- Attack is based on obsolete/deleted S3 Resources, which can be enumerated through DNS
- Attacker identifies CNAMES in the target domain's zone file and looks for deleted S3 Resources, that correspond to a particular sub-domain
- Attacker creates S3 Bucket with the deleted subdomain and now effectively controls the content on the specific subdomain of the organization
- Attack is quite popular simply because of the number of deleted S3 resources, for which DNS content is not removed at the earliest

Subdomain Takeover - Diagram

Public Read/Write on Amazon S3 Buckets

- Administrators often leave Public Read/Write Access on S3 Buckets
- The “Any Authenticated AWS User” problem - assumption that its an AWS Authenticated User for that organization
- Often, highly sensitive information from apps are stored in S3 Buckets

Overly-Permissive Access Control Rules

- Administrators often provide overly permissive access to users to S3 and other services in the AWS ecosystem
- Largely a failure to implement appropriate IAM permissions

we45

Amazon S3 - Security Best Practices

we45

we45

Amazon S3 - Defense-in-Depth

- Access Control
- Secure Configuration
- Encryption and Key Management - Assets in S3
- Logging and Monitoring
- Tools
- Examples and API

we45

Lab: Setup S3 + Access Controls

AWS - Host and Network Security

we45

we45

Terminologies

- Instance
  - A Virtual Server = Typically referred to as an “EC2 Instance”
- AMI
  - Amazon Machine Image = Prebuilt images that can be used to run applications
- Availability Zone:
  - Regions
  - Availability Zone = Zone in a Physically separate DataCenter within its region
- VPC = Virtual Private Cloud
  - Private Network in an Availability Zone

Compute Services - AWS

- Elastic Cloud Compute (EC2)
- Elastic Beanstalk (EB) => Deploy and Scale Web Applications
- ECS and EKS => Elastic Containers and Elastic Kubernetes
- AWS Lambda => Serverless
- AWS Batch => Run Batch Jobs on EC2 or EC2 Spot Instances
- LightSail => Lightweight Servers for Web Hosting

Running Instances in AWS

- EC2 == Elastic Cloud Compute
- Multiple Variants and Types based on Usage and Resource Requirements
  - We will typically use micro or nano instances in our labs
- This is Amazon's IaaS offering => Manage your server and everything upwards, yourself!

Security Parameters - EC2

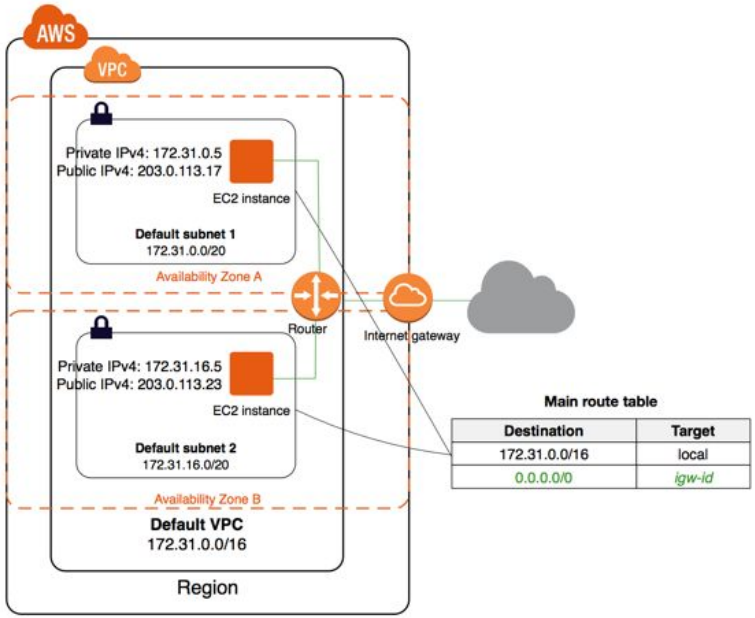
- **Host and OS Security** => Inspector, CIS Benchmarks, Lynis, Microsoft Baseline Security Standards, HIDS and HIPS (as required)
- **Network Security** => Security Groups + VPC
- **Logging** => Centralized Logging to Cloudwatch and S3
- **Protect Data at Rest** => Disk Encryption for EBS/Custom
- **Access Control** => Host Access Control (SSH, etc) + IAM

Network Security

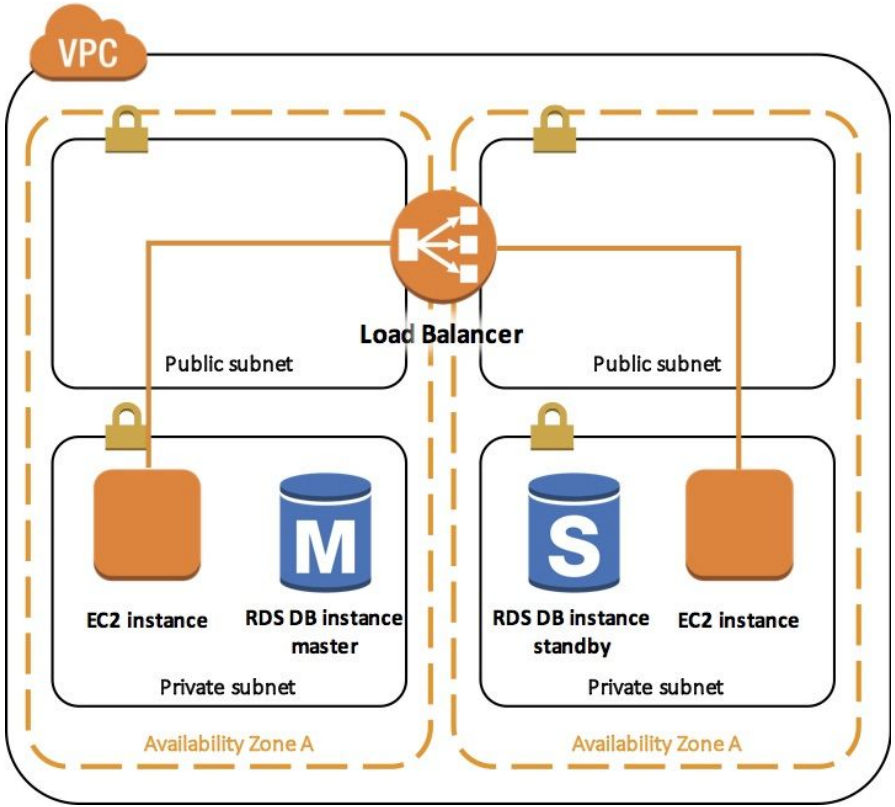
- VPC
- VPC + Security Groups
- Private and Public Resources
- Access to Public and Private Subnets in a VPC
- Gateways and VPC Peering

Virtual Private Cloud

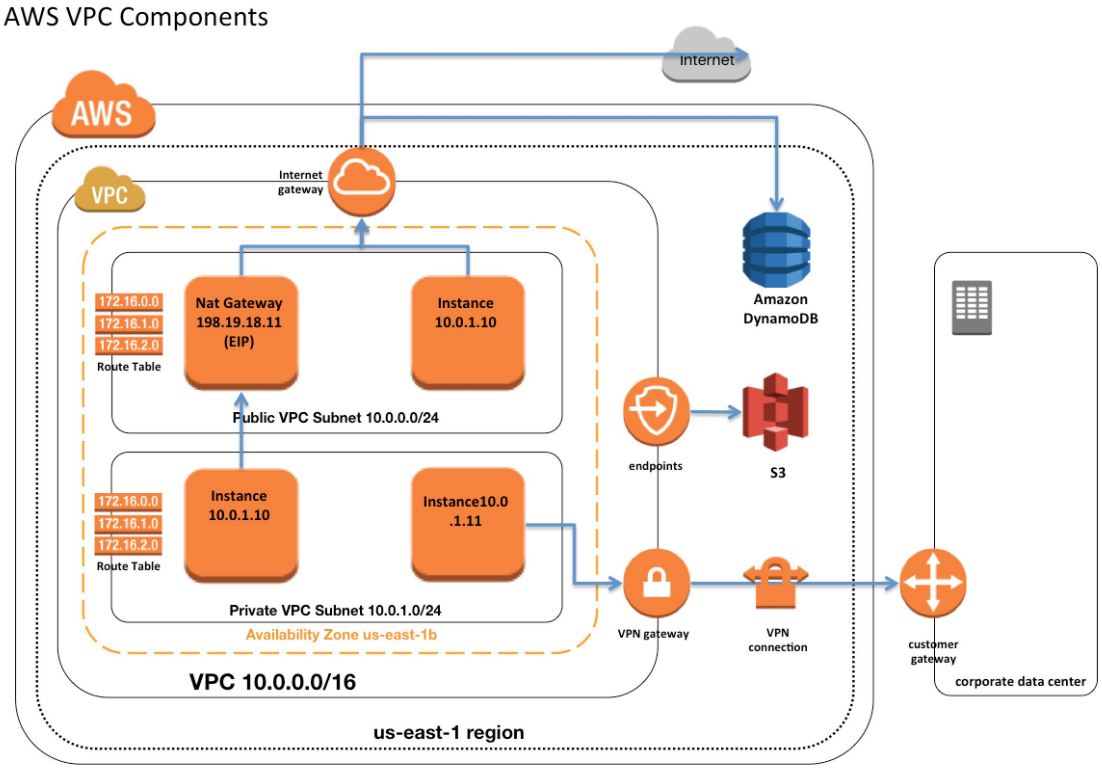
- VPC = Virtual Network dedicated to your AWS Account
- A VPC spans all availability zones in a region
- Subnet = Range of IPs within your VPC
  - Create Multiple Subnets in each availability zone
- Your account comes with a default VPC
- Control based on how your VPC can access the gateway



VPC Models - Private & Public Subnet



VPC - Private and Public Subnet



we45

Security Groups

- Access Control feature akin to a Firewall
- Security Groups can be applied on a VPC
- Default Deny => Doesn’t allow any communication with any hosts in the VPC
- Control Ingress (Inbound) and Egress (Outbound) traffic to the VPC (hosts)
- Security Groups are ALLOW only, there’s no Deny.
    - If its not allowed. Its denied
- Security Groups are Stateful
- Limits on number of rules and number of VPCs

Lab: VPC with Public and Private Subnet

we45

Compute Services - AWS

- Elastic Cloud Compute (EC2)
- Elastic Beanstalk (EB) => Deploy and Scale Web Applications
- ECS and EKS => Elastic Containers and Elastic Kubernetes
- AWS Lambda => Serverless
- AWS Batch => Run Batch Jobs on EC2 or EC2 Spot Instances
- LightSail => Lightweight Servers for Web Hosting

Recent Problems

Microsoft Windows SMB Server (MS17-010) Vulnerability

Description: Microsoft Windows SMB Server is prone to a remote code-execution vulnerability. Successful exploits will allow an attacker to execute arbitrary code on the target system. Failed attacks will cause denial of service conditions.

Related CVE's: CVE-2017-0145, CVE-2017-0146, CVE-2017-0147, CVE-2017-0148

Vulnerable Versions:

- Microsoft Windows Vista x64 Edition Service Pack 2
- Microsoft Windows Vista Service Pack 2
- Microsoft Windows Server 2012 R2
- Microsoft Windows Server 2012

Host Security Practices

- Host Access Control
  - Authentication
  - Authorization
  - IAM Role
- Host - Encryption
  - Disk Encryption
  - Data Encryption
- Host - Logging and Monitoring
  - Cloudwatch
  - CloudTrail
- Host - Vulnerability Management
  - AWS Inspector
  - Lynis and MBSA
- Host - Host-Based Intrusion Detection
  - OSSEC
  - Commercial Solutions
- Host - Network Security
  - VPC
  - Security Groups

Host Access Control

we45

Access Control - Hosts on AWS

- Default SSH on Linux hosts with Key required to be provided by the user, or AWS-provided Key
- EC2 Hosts can be assigned to the IAM as well => IAM Role
  - Must be Least-Privilege
- Gotchas:
  - Metadata Object within the EC2 Instance has Temporary creds to access
  - Compromise of Instance == Compromise of linked services on the cloud

EC2 - Metadata

AWS EC2 metadata. Check attached IAM role from EC2 instance. Get temporary credentials.

gistfile1.txt — Raw

```
# Get IAM Role name from Instance Profile Id
curl http://169.254.169.254/latest/meta-data/iam/info

# Get credentials
curl http://169.254.169.254/latest/meta-data/iam/security-credentials/<role-name>

# More info
http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-metadata.html
```

matheusoliveira commented on Jul 21, 2017

I found a simpler way (no need to process `InstanceProfileArn` to get the role name):

```
role_name=$( curl -s http://169.254.169.254/latest/meta-data/iam/security-credentials/ )
curl -s http://169.254.169.254/latest/meta-data/iam/security-credentials/${role_name}
```

Lab: Compromise EC2 and linked services

we45

Hardening your Host

- Access Control => Authentication and Authorization
- Securing Vulnerable Configurations on the Host OS
- Reducing Attack Surface by running only required services
- Running Services with Secure Defaults/Configurations
- Continuously Monitoring Host for signs of attack/compromise
- Hardening Guides
    - CIS Benchmarks for <Insert Name Here> Operating System / Services
    - Vendor Security Best Practices => Operating System and Services
    - CVE Database from NIST and NVD
    - Security Alerts from Vendor

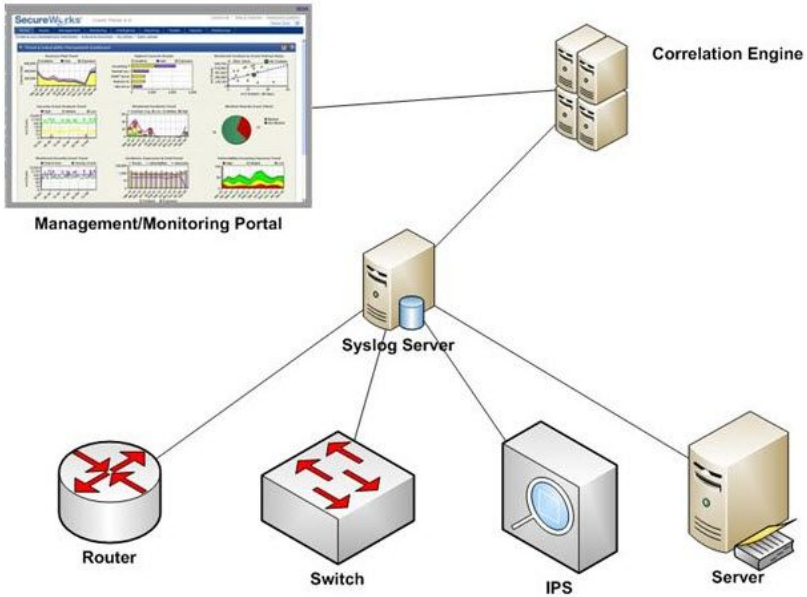
AWS - Logging and Monitoring

we45

Why Log and Monitor?

- Essential Detective Control => Detect Attacks in progress, post-mortem
- Good way to understand potential security exceptions or minor incidents
- Compliance and Regulatory Requirements
- Required in Forensic investigations

Enterprise Logging and Monitoring



Security Monitoring Viewpoints

- Host/OS:
  - Operating System Logs
  - Service Logs
  - HIDS Logs
  - Anti-Virus/Malware Prevention Logs
  - File System Event Logs
- Application
  - Component Logs (Service Logs)
  - In-Application Logs (Events, APIs, etc)
  - Access Logs
  - Exception Logs
- Network
  - Flow Logs
  - Deep-Packet Inspection Logs
  - Intrusion Detection/Prevention Logs
  - Content Filtering Logs
  - Firewall Logs
  - Access Control Logs
  - VPN logs

Challenges with Logging & Monitoring

- Monitoring ONLY works when there's logging
- Logging is turned OFF most of the time
- Perceived as unnecessary overhead and storage
- Applications don't log meaningful information => Architecture
- (Previously) Hard to do without really expensive tools
- Expectation that logs "manage themselves"

Lab - Logging for EC2

we45

Amazon CloudTrail

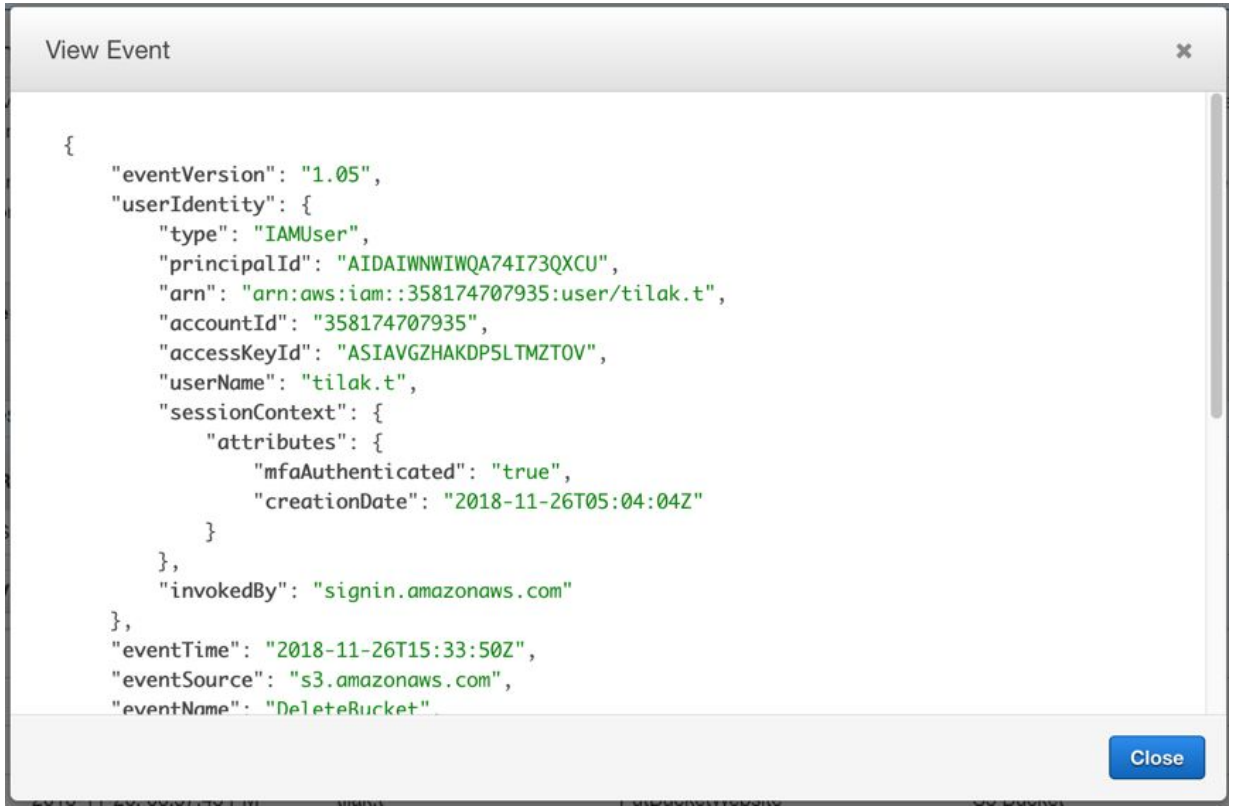
we45

- Service that is meant to track activity within the AWS Account to enable:
  - Governance
  - Compliance
  - Operational Activity
- Near-Realtime activity of AWS APIs on AWS Services being invoked
- View based on Event History OR
- Based on a Trail that you create that capture ongoing activities in your AWS Account

CloudTrail Security Considerations

- By default, Event History is stored for 90 days
- Be default, Cloudtrail is encrypted with KMS SSE
- Trail delivers logs within 15 mins of activity
- Notifications can be setup with Amazon Simple Notification Service (SNS)
- Captures Actions done on behalf of the user. Example: KMS for S3 Encryption

CloudTrail


View Event
{
"eventVersion": "1.05",
"userIdentity": {
"type": "IAMUser",
"principalId": "AIDAIWNWIWQA74I73QXCU",
"arn": "arn:aws:iam::358174707935:user/tilak.t",
"accountId": "358174707935",
"accessKeyId": "ASIAVGZHAKDP5LTMZTOV",
"userName": "tilak.t",
"sessionContext": {
"attributes": {
"mfaAuthenticated": "true",
"creationDate": "2018-11-26T05:04:04Z"
}
},
"invokedBy": "signin.amazonaws.com"
},
"eventTime": "2018-11-26T15:33:50Z",
"eventSource": "s3.amazonaws.com",
"eventName": "DeleteBucket",
Close

CloudTrail Workflows

- View Logs
- Create Trail
- Monitor with Cloudwatch Logs
- Enable Encryption
- Enable File Integrity

Lab: Cloudtrail Basics

we45

Cloudwatch

we45

Cloudwatch

- Monitoring AWS Services and Applications
- Collect Metrics and Events from Services
- Create Log Streams and capture logs in those streams
- Obtain Alerts and Manage Alerts based on Metrics

we45

Things you can do with Cloudwatch

- Cloudwatch Logs => Agent to collect and push logs from EC2
- Collate results from Cloudtrail (trails) and analyze logs from it
- Setup Alarms (Alerts) for exceptional events

Lab: Cloudwatch

we45

Security Features - AWS

- User Access:
  - Access Keys and Secrets
  - Passwords with Login Profiles
  - Multi-Factor Authentication
  - Comprehensive IAM System - Access Control
  - Access Control Policies/Lists for each Component within AWS
- Protect Data at Rest
  - Amazon Key Management System (KMS)
  - Amazon Secrets Manager
  - Amazon System Manager Parameter Store (SSM Param Store)
- Network Security
  - VPC => Virtual Private Cloud
  - Security Groups
  - Access Control Lists

Security Features - AWS

- Logging and Monitoring
  - Cloudwatch
  - Cloudtrail
  - AWS Config
- Advanced Security Controls
  - Managed Web Application Firewall
  - Amazon Macie => Information Discovery, Classification and Protection
  - Amazon GuardDuty => Threat Intelligence
  - Amazon Inspector => Host Security Scanning
  - Amazon Cognito => Backend-as-a-Service for SSO and Access Management (Apps)

Security Concepts - AWS

Region:

- Independent Collection of AWS Resources in a Geography

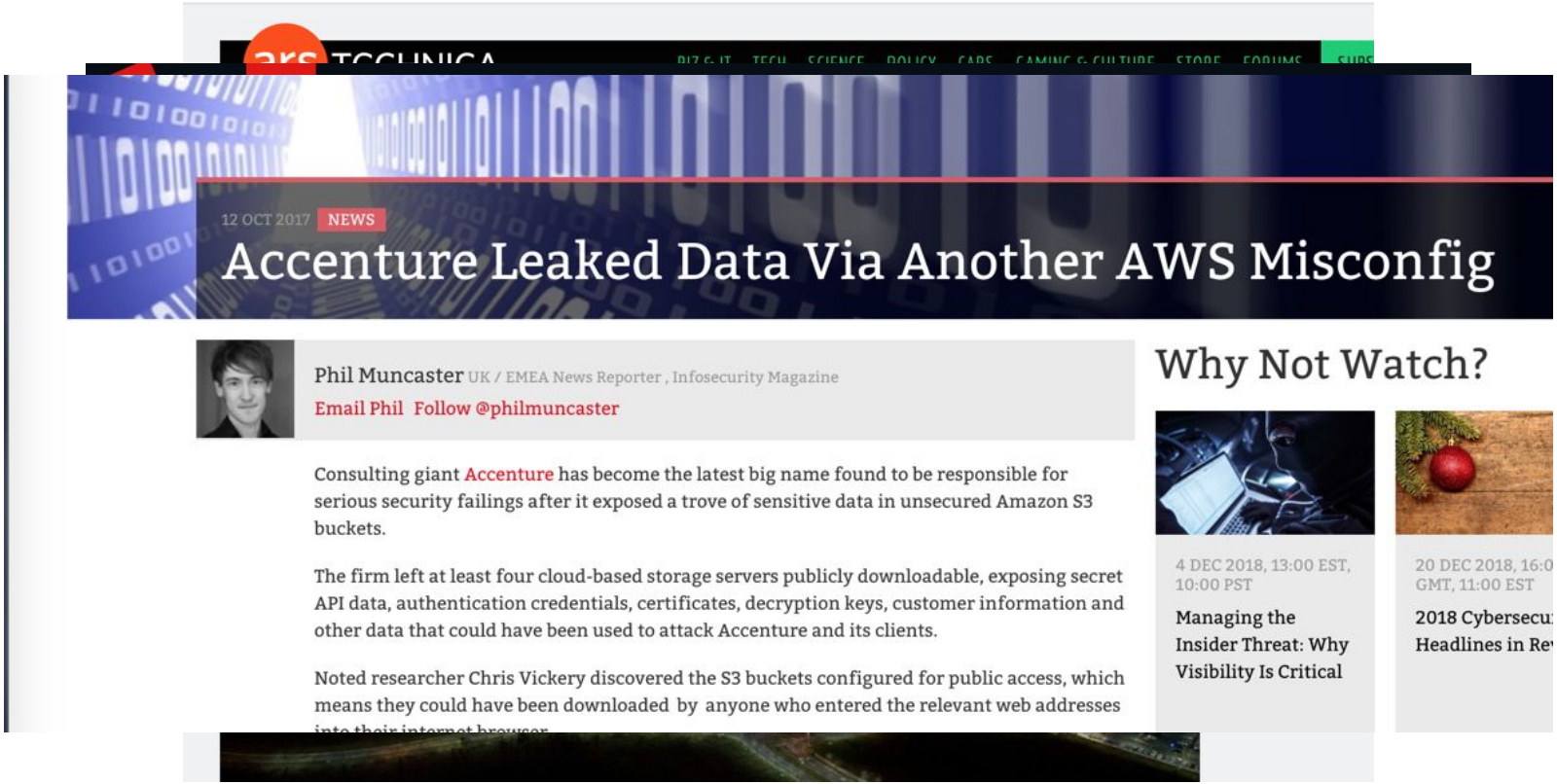
Availability Zone:

- Collection of resources of a region in a particular datacenter

Common AWS Security Mistakes

we45

Amazon S3



12 OCT 2017 NEWS

Accenture Leaked Data Via Another AWS Misconfig

Phil Muncaster UK / EMEA News Reporter , Infosecurity Magazine
Email Phil Follow @philmuncaster

Consulting giant Accenture has become the latest big name found to be responsible for serious security failings after it exposed a trove of sensitive data in unsecured Amazon S3 buckets.

The firm left at least four cloud-based storage servers publicly downloadable, exposing secret API data, authentication credentials, certificates, decryption keys, customer information and other data that could have been used to attack Accenture and its clients.

Noted researcher Chris Vickery discovered the S3 buckets configured for public access, which means they could have been downloaded by anyone who entered the relevant web addresses into their internet browser

Why Not Watch?

4 DEC 2018, 13:00 EST, 10:00 PST

Managing the Insider Threat: Why Visibility Is Critical

20 DEC 2018, 16:0 GMT, 11:00 EST

2018 Cybersecu Headlines in Re

Amazon IAM - Elevation of Privs

Abusing the AWS metadata service using SSRF vulnerabilities

18 June 2017

I recently worked on a small toy project to execute untrusted Python code in Docker containers. This lead me to test several online code execution engines to see how they reacted to various attacks. While doing so, I found several interesting vulnerabilities in the code execution engine developed by Qualified, which is quite widely used including by websites like CodeWars or InterviewCake. The combination of being able to run code with network access and the fact that the infrastructure was running in Amazon Web Services lead to an interesting set of vulnerabilities which we present in this post.

We start by presenting several vulnerabilities I found in the Qualified code execution engine via one of its customers, InterviewCake. Then, we talk about the implications of a specific one: a SSRF vulnerability in a service running on AWS. I won't cover the basics of what is a SSRF vulnerability, as there are already great resources available about it (here, here or here). In one sentence, it's a vulnerability that allows you to have an application initiate a network connection on its behalf.

AWS Lack of Encryption

- Encryption is a little-used feature within the AWS ecosystem
- Encryption is rarely leveraged
- Account access == Complete plaintext access
- Differences in Encryption types, not well understood

AWS - Vulnerability Assessment

we45

An Intro to Vulnerability Assessment

we45

- Vulnerability Assessment => Practice of identifying vulnerabilities against a target system
- The assessment focuses on identifying vulnerabilities with automated tools and manual techniques
- Different Types of Assessments:
  - Whitebox
  - Greybox
  - BlackBox
- Typically stops with identifying flaws. I.E. No Exploitation

we45

Why Vulnerability Assessment?

- To identify vulnerabilities against targets on an ongoing basis
- Compliance Requirements:
  - ISO-27001
  - PCI-DSS
  - HIPAA
  - GDPR
- Contract Obligations

Common Vulnerabilities with AWS

- IAM Policies being too open, or by principle of least privilege
- Lack of encryption of data at rest
- Lack of User Security Features
- Application Security Flaws

Lab: Vulnerability Scanning with Amazon Inspector

we45