Compliance in the Cloud

IIA/ISACA Joint Seminar
April 13\textsuperscript{th}, 2016
Aaron Wilson

- San Diegan (Missouri Export)
- Destroyer of Homes
Aaron Wilson

- Founder & Principal, Scalable Security
- Senior Consultant, AWS
- Chief Security Architect, ArcSight
- CTO, SAIC Managed Security Services
- @GrokJAaron
- scalesec.com

https://github.com/ScaleSec
Cloud: What’s the Big Deal?

• Novartis
  – Challenge: Screen 10 million compounds
  – Estimates: $40,000,000 USD for 50,000 cores
  – Reality on AWS: $4,232 USD for 87,000 cores
    • Completed in 9 hours
  – Savings: $39,995,768 USD
    • Or 9,451x
Customer Data

Platform, Applications, Identity & Access Management

Operating System, Network & Firewall Configuration

Client-side Data Encryption & Data Integrity Authentication

Server-side Encryption (File System and/or Data)

Network Traffic Protection (Encryption / Integrity / Identity)

Customer

This Talk

AWS

Responsible for security ‘in’ the Cloud

Responsible for security ‘of’ the Cloud

Compute

Storage

Database

Networking

AWS Global Infrastructure

Regions

Edge Locations

Availability Zones
How to use the Cloud Management Console
How to use the Cloud

$ aws s3 ls s3://mybucket

<table>
<thead>
<tr>
<th>LastWriteTime</th>
<th>Length</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>2013-09-03 10:00:00</td>
<td>1234</td>
<td>myfile.txt</td>
</tr>
</tbody>
</table>

Management Console

CLI
How to use the Cloud

Create

```python
#!/usr/bin/env python
import boto3

s3 = boto3.resource('s3')
bucket = s3.Bucket('mybucket')
for obj in bucket.objects.all():
    print(obj.key)
```

Run

```
$ python list_s3_bucket.py
myfolder/myfile.txt
$
```
How to use the Cloud

... or use interactively

```
$ python
>>> import boto3
>>> s3 = boto3.resource('s3')
>>> bucket = s3.Bucket('mybucket')
>>> for obj in bucket.objects.all():
...     print(obj.key)
...
myfolder/myfile.txt
```
Five ways our world is changing.

SOFTWARE INDUSTRY TRENDS
The New IT Landscape

• Smaller Deployments
Smaller Deployments

huge & monolithic

compontent changes
Deployments: The Old Way

- Insert media
- Boot system
- Press any key
- Click continue
- Click continue
- Click next
- Reboot
- Click next
- Insert media
- Double click
- Click next
- Reboot
- Click next
- Patch
- Reboot
- Click next
- Patch
- Insert media
- Double click
- Click next
- Reboot
- Click next
- Complete!
Continuous Deployment Pipeline*

Develop → Commit → Build → Test → Deploy → Operate

Develop → Commit → Build → [AuthN, Compliance, Scan, OWASP, ...] → Deploy → Operate

*simplified example
The New IT Landscape

• Smaller Deployments

• More Frequent Deployments
More Frequent Deployments

• In 2014, Amazon pushed 50 million code deployments
• ~136,986 / day
• ~95 / minute
More Frequent Deployments

• Google: 20,000+ / day
• Facebook: “Every engineer deploys every day”
• Netflix: Hundreds / day
• Etsy: 50 / day (150 engineers)
• Flickr: 10 / day (18 engineers)
The New IT Landscape

• Smaller Deployments
• More Frequent Deployments
• Elastic Infrastructure
Internet Load Balancer

Master DB

Standby DB

users
Internet Load Balancer

users

Master DB

Standby DB

SCALABLE SECURITY
The New IT Landscape

• Smaller Deployments
• More Frequent Deployments
• Elastic Infrastructure
• Smaller Compute Slices
Smaller Compute Slices

- Functions
- Containers
- IaaS
- Virtualized
- Bare Metal
The New IT Landscape

- Smaller Deployments
- More Frequent Deployments
- Elastic Infrastructure
- Smaller Compute Slices
- Toward a Serverless World
Immutability & Serverless Concepts

• Phase 1: Immutable Infrastructure
  – Compute layer is ephemeral
  – Changes introduced through the pipeline or configuration management software
  – No user SSH access to hosts
  – Offboard, centralized logging
Immutability & Serverless Concepts

• Phase 2: No Servers
  – Stitching together cloud services
  – Underlying servers managed by cloud provider
  – No hosts for you to manage
COMMON CLOUD ARCHITECTURE
AWS Account region

VPC

Internet Gateway

Internet

AWS

Jump NAT

department

public subnet

private subnet

Availability Zone

Master DB

private subnet

Standby DB

private subnet

Availability Zone

region

public subnet

private subnet

Jump NAT

SCALABLE SECURITY
AWS Account
region
VPC
Internet Gateway
Jump NAT
public subnet
Load Balancer
Jump NAT
public subnet
Auto Scaling group
Master DB
private subnet
Standby DB
private subnet
Availability Zone
Availability Zone
AWS Account region

VPC

Internet Gateway

users

Load Balancer

Jump NAT

custom subnet

Auto Scaling group

private subnet

Master DB

private subnet

Availability Zone

Standby DB

private subnet

Availability Zone

Internet

AWS

region

Internet

jump NAT

custom subnet

users

AWS Account region

VPC

Internet Gateway

users

Load Balancer

Jump NAT

custom subnet

Auto Scaling group

private subnet

Master DB

private subnet

Availability Zone

Standby DB

private subnet

Availability Zone

Internet
AWS Account region
VPC
Internet Gateway
Auto Scaling group
Master DB
Standby DB
Availability Zone
private subnet
public subnet
Jump NAT
Load Balancer
Time Passes...

...and Business Happens...
VISIBILITY
Cloud Event Sources

- Familiar
  - Syslog / Event Log
  - Server Logs
  - App Logs
  - DB Logs
  - AuthN Logs
  - Remote Access Logs

- New
  - Credential Reports
  - API Audit Logs
  - Service Logs
  - State Changes
  - Service Access Logs

Network Flow Logs
AWS IAM: Credential Report

- User creation time
- Password enabled?
- Password last changed
- Password next rotation
- MFA active?
- API access keys?
- Access keys last rotated?
API Audit Logs

- AWS CloudTrail
- Google Stackdriver Logging
- Microsoft?
- Records API call activity
- Ingest and analyze with common logging tools

Example AWS CloudTrail Log

```
"userIdentity": {
  "type": "IAMUser",
  "principalId": "AIEXAMPLE987ZKLALD3HS",
  "arn": "arn:aws:iam::123456789012:user/Bob",
  "accountId": "123456789012",
  "accessKeyId": "ASEXAMPLE1234WTROX8F",
  "userName": "Bob",
  "sessionContext": {
    "attributes": {
      "mfaAuthenticated": "true",
      "creationDate": "2015-02-16T13:35:20Z"
    }
  },
  "invokedBy": "signin.amazonaws.com"
}
```
Network Logging: AWS VPC Flow Logs

- NetFlow/Sflow-like network data
- Enable for a VPC, a subnet, or a network interface
- Delivered to CloudWatch Logs
- Takes a few minutes to deliver results
VPC Flow Logs

2 123456789010 eni-abc123de 172.168.1.12 172.168.1.11 20641 22 6 20 4249 1418530010 1418530070 ACCEPT OK
S3 and ELB Access Logs

• S3 Access Logging
  – Enabled per bucket
  – Flexible delivery options
  – Best effort
  – Delivered to S3 bucket of your choice

• ELB Access Logging
  – Requests sent to your load balancer
  – Delivered to S3 bucket of your choice
S3 Access Logs

79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be mybucket [06/Feb/2014:00:01:57 +0000] 192.0.2.3 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be DD6CC733AEXAMPLE REST.PUT.OBJECT s3-dg.pdf "PUT /mybucket/s3-dg.pdf HTTP/1.1" 200 - - 4406583 41754 28 "-" "S3Console/0.4" -
### ELB Access Logs

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>ELB Name</th>
<th>Client:Port</th>
<th>Backend:Port</th>
<th>Request Processing Time (s)</th>
<th>Backend Processing Time (s)</th>
<th>Response Processing Time (s)</th>
<th>ELB Status Code</th>
<th>Backend Status Code</th>
<th>Sent Bytes</th>
<th>Request</th>
<th>User Agent</th>
<th>SSL Cipher</th>
<th>SSL Protocol</th>
<th>Received Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>2015-05-13T23:39:43.945958Z</td>
<td>my-loadbalancer</td>
<td>192.168.131.39:2817</td>
<td>10.0.0.1:80</td>
<td>0.000086</td>
<td>0.001048</td>
<td>0.001337</td>
<td>200</td>
<td>200</td>
<td>57</td>
<td>&quot;GET <a href="https://www.example.com:443/">https://www.example.com:443/</a> HTTP/1.1&quot;</td>
<td>&quot;curl/7.38.0&quot;</td>
<td>DHE-RSA-AES128-SHA</td>
<td>TLSv1.2</td>
<td>0</td>
</tr>
</tbody>
</table>
Extensible Examples

AUDITING CLOUD SECURITY
Example: AWS Password Policy

Minimum password length:

- Require at least one uppercase letter
- Require at least one lowercase letter
- Require at least one number
- Require at least one non-alphanumeric character
- Allow users to change their own password
- Enable password expiration

Password expiration period (in days): 90

- Prevent password reuse
- Number of passwords to remember: 5

Password expiration requires administrator reset

Apply password policy
Delete password policy
aws.amazon.com/cli

The AWS Command Line Interface (CLI) is a unified tool to manage your AWS services. With just one tool to download and configure, you can control multiple AWS services from the command line and automate them through scripts.

The AWS CLI introduces a new set of simple file commands for efficient file transfers to and from Amazon S3.
Example: Password Policy

$ aws iam get-account-password-policy

{  
    "PasswordPolicy": {  
        "AllowUsersToChangePassword": true,  
        "RequireLowercaseCharacters": true,  
        "RequireUppercaseCharacters": true,  
        "MinimumPasswordLength": 15,  
        "RequireNumbers": true,  
        "PasswordReusePrevention": 24,  
        "HardExpiry": false,  
        "RequireSymbols": true,  
        "MaxPasswordAge": 60,  
        "ExpirePasswords": true  
    }  
}
Example: Password Policy

$ aws iam get-account-password-policy
Example: Password Policy

```bash
$ aws --profile prod-acct1 iam get-account-password-policy
```

```json
{
    "PasswordPolicy": {
        "AllowUsersToChangePassword": true,
        "RequireLowercaseCharacters": true,
        "RequireUppercaseCharacters": true,
        "MinimumPasswordLength": 15,
        "RequireNumbers": true,
        "PasswordReusePrevention": 24,
        "HardExpiry": false,
        "RequireSymbols": true,
        "MaxPasswordAge": 60,
        "ExpirePasswords": true
    }
}
```
Example: Password Policy

```bash
$ aws --profile prod-acct2 iam get-account-password-policy

{
  "PasswordPolicy": {
    "AllowUsersToChangePassword": true,
    "RequireLowercaseCharacters": true,
    "RequireUppercaseCharacters": true,
    "MinimumPasswordLength": 15,
    "RequireNumbers": true,
    "RequireSymbols": true,
    "HardExpiry": false,
    "ExpirePasswords": false
  }
}
```
Example: Password Policy

```bash
#!/bin/bash
cli_profiles="prod-acct1 prod-acct-2"

# loop through multiple accounts
for profile in ${cli_profiles}; do
    echo "\n\n>> Getting current policy for ${profile}..."
    aws --profile ${profile} iam get-account-password-policy
done
```
Example: Password Policy

>> Getting CURRENT policy for prod-acct1... >> Getting CURRENT policy for prod-acct2...

```
>> Getting CURRENT policy for prod-acct1...
{
    "PasswordPolicy": {
        "AllowUsersToChangePassword": true,
        "RequireLowercaseCharacters": true,
        "RequireUppercaseCharacters": true,
        "MinimumPasswordLength": 15,
        "RequireNumbers": true,
        "PasswordReusePrevention": 24,
        "HardExpiry": false,
        "RequireSymbols": true,
        "MaxPasswordAge": 60,
        "ExpirePasswords": true
    }

>> Getting CURRENT policy for prod-acct2...
{
    "PasswordPolicy": {
        "AllowUsersToChangePassword": true,
        "RequireLowercaseCharacters": true,
        "RequireUppercaseCharacters": true,
        "MinimumPasswordLength": 15,
        "RequireNumbers": true,
        "RequireSymbols": true,
        "HardExpiry": false,
        "ExpirePasswords": true
    }
}
```
Example: Password Policy

#!/bin/bash

cli_profiles="prod-acct prod-acct-2"

# Get existing policies, if any
for profile in ${cli_profiles}; do
    echo "\n\n>> Getting CURRENT policy for ${profile}..."
    aws --profile ${profile} iam get-account-password-policy --output=table
done
Example: Password Policy

<table>
<thead>
<tr>
<th>AllowUsersToChangePassword</th>
<th>True</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExpirePasswords</td>
<td>True</td>
</tr>
<tr>
<td>HardExpiry</td>
<td>False</td>
</tr>
<tr>
<td>MaxPasswordAge</td>
<td>60</td>
</tr>
<tr>
<td>MinimumPasswordLength</td>
<td>15</td>
</tr>
<tr>
<td>PasswordReusePrevention</td>
<td>24</td>
</tr>
<tr>
<td>RequireLowercaseCharacters</td>
<td>True</td>
</tr>
<tr>
<td>RequireNumbers</td>
<td>True</td>
</tr>
<tr>
<td>RequireSymbols</td>
<td>True</td>
</tr>
<tr>
<td>RequireUppercaseCharacters</td>
<td>True</td>
</tr>
</tbody>
</table>

| AllowUsersToChangePassword | True    |
| ExpirePasswords            | False   |
| HardExpiry                 | False   |
| MinimumPasswordLength      | 15      |
| RequireLowercaseCharacters | True    |
| RequireNumbers             | True    |
| RequireSymbols             | True    |
| RequireUppercaseCharacters | True    |
AWS IAM: Credential Report

- User creation time
- Password enabled?
- Password last changed
- Password next rotation
- MFA active?
- API access keys?
- Access keys last rotated?
Example: Credential Report

```bash
$ aws iam generate-credential-report
{
    "State": "STARTED",
    "Description": "No report exists. Starting a new report generation task"
}
```
Example: Credential Report

$ aws iam generate-credential-report
{
  "State": "STARTED",
  "Description": "No report exists. Starting a new report generation task"
}

Wait...
Example: Credential Report

$ aws iam generate-credential-report
{
   "State": "STARTED",
   "Description": "No report exists. Starting a new report generation task"
}

Wait...

$ aws iam get-credential-report
{
   "Content": "XzFfbGFzdF91c2VkX3JlZ2lvbixhY2Nlc3Nfa2V5XzFfbGJkX2VuYWJsZWQscGFzc3dvcmRfbGFzdF91c2VkLHBhc3N3b3JkX2xhc3RfY2hhbGFzdF9yb3RhdGVkLGFnY2Vzc19rZXlfMV9sYXN0X3VzZWRfZGF0ZXJhxY2Nlc3Nfa2V5X3Nmd1ZCwxYXNzd29yZGF9uZXh0X3JvdGF0aW9uLG1mYV9hY3RpdmUsYWNjZXNzX2tleV8xX2FjdGl2ZSxhY2Nlc3Nfa2V5XzFfbGFzdF91c2VkX3JlZ2lvbixhY2Nlc3Nfa2V5XzFfbGFzdF91c2VkX3NlcnZpY2UsYWNjZXNzX2tleV8yX2FjdGl2ZSxhY2Nlc3Nfa2V5XzJfbGFzdF9yb3RhdGVkLGFnY2Vzc19rZXlfM19sYXN0X3VzZWRfZGF0ZXJhxY2Nlc3Nfa2V5XzJfbGFzdF91c2V...
   [truncated],
   "GeneratedTime": "2016-04-10T21:35:49Z",
   "ReportFormat": "text/csv"
Example: Credential Report

$ aws --output=json iam get-credential-report | grep Content | awk -F" '{print $4}' | base64 -D > iam_credential_report.csv
Example: Credential Report

#!/bin/bash

cli_profiles="prod-acct1 prod-acct2"

# loop through multiple accounts
for profile in ${cli_profiles}; do
    reportStatus=
    echo "\n\n>> Creating credential report for ${profile}..."
    until [ "${reportStatus}" == 'COMPLETE' ]; do
        reportStatus=`aws --profile ${profile} --output=json iam generate-credential-report | grep State | awk -F" '{print $4}'`
        if [ "${reportStatus}" != 'COMPLETE' ]; then
            echo "Waiting on report generation...(${reportStatus})"
            sleep 10
        fi
    done
    echo "\n\n>> Retrieving credential report for ${profile}..."
    aws --profile ${profile} --output=json iam get-credential-report | grep Content | awk -F" '{print $4}' | base64 -D > iam_credential_report_${profile}.csv
    echo "...report iam_credential_report_${profile}.csv created."
done
Now What?

• Sign up for a Cloud Account
  – Free tiers
• Download examples, modify to your needs
  – https://github.com/ScaleSec
• Homework:
  – Discover network security settings
  – Modify password policy
  – Find IAM users with no MFA
Thank You!

• https://github.com/ScaleSec
• @GrokAaron
• scalesec.com