Clearing the Clouds: Incident Response in AWS (Isn’t as Bad as You Thought)

Kristy Westphal
VP, CSIRT
Disclaimer

- The views, opinions, and material presented by Kristy Westphal at this conference are solely based on her experience and opinions related to incident response.

- The content of this presentation does not reflect the views or opinions of MUFG Union Bank.
Agenda

- Why am I Here?
- AWS Architecture 101
- Incident Response Use Cases
- Acquiring Amazon Web Services (AWS) Skills
- 90 Day Plan for AWS Incident Response
Why am I here?

- Information security leader specializing in security assessments, operational risk, and program development
- Security is painful all around; hopefully I can help
- Let’s share knowledge and make it less painful for all of us!
- Props to Pete Ehlke for helping make this preso really come to life
Why AWS incident response is important...

YOU'RE PRETTY NEW TO CLOUD STORAGE, AREN'T YOU?

LYNCH
What this session is...and isn’t...

- Think about cloud incident response differently
  - But not as the impossible mountain to climb

- Yes, this is only Amazon Web Services
  - But the approach can be applied to other providers

- We won’t be doing in-depth AWS training
  - But you will have resources to do this yourself
Poll the Audience

- CSV-R01
- Are you doing security incident response in AWS now?
  - A. Yes
  - B. No
  - C. I Don’t Know

https://rsa1-live.eventbase.com/polls?event=rsa2020&session=1454108104
A Peek Into AWS Architecture
**Pizza as a Service**

<table>
<thead>
<tr>
<th>Traditional On-Premises (Legacy)</th>
<th>Infrastructure as a service (IaaS)</th>
<th>Platform as a service (PaaS)</th>
<th>Software as a service (SaaS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dining Table</td>
<td>Dining Table</td>
<td>Dining Table</td>
<td>Dining Table</td>
</tr>
<tr>
<td>Drinks</td>
<td>Drinks</td>
<td>Drinks</td>
<td>Drinks</td>
</tr>
<tr>
<td>Electric / Gas</td>
<td>Electric / Gas</td>
<td>Electric / Gas</td>
<td>Electric / Gas</td>
</tr>
<tr>
<td>Oven</td>
<td>Oven</td>
<td>Oven</td>
<td>Oven</td>
</tr>
<tr>
<td>Fire</td>
<td>Fire</td>
<td>Fire</td>
<td>Fire</td>
</tr>
<tr>
<td>Pizza Dough</td>
<td>Pizza Dough</td>
<td>Pizza Dough</td>
<td>Pizza Dough</td>
</tr>
<tr>
<td>Tomato Sauce</td>
<td>Tomato Sauce</td>
<td>Tomato Sauce</td>
<td>Tomato Sauce</td>
</tr>
<tr>
<td>Toppings</td>
<td>Toppings</td>
<td>Toppings</td>
<td>Toppings</td>
</tr>
<tr>
<td>Cheese</td>
<td>Cheese</td>
<td>Cheese</td>
<td>Cheese</td>
</tr>
</tbody>
</table>

*Thank you Albert Barron for this example.*
Pizza as a Service

- **Traditional On-Premises (Legacy)**
  - Dining Table
  - Drinks
  - Electric / Gas
  - Oven
  - Fire
  - Pizza Dough
  - Tomato Sauce
  - Toppings
  - Cheese

- **Infrastructure as a Service (IaaS)**
  - Dining Table
  - Drinks
  - Electric / Gas
  - Oven
  - Fire
  - Pizza Dough
  - Tomato Sauce
  - Toppings
  - Cheese

- **Platform as a Service (PaaS)**
  - Dining Table
  - Drinks
  - Electric / Gas
  - Oven
  - Fire
  - Pizza Dough
  - Tomato Sauce
  - Toppings
  - Cheese

- **Software as a Service (SaaS)**
  - Dining Table
  - Drinks
  - Electric / Gas
  - Oven
  - Fire
  - Pizza Dough
  - Tomato Sauce
  - Toppings
  - Cheese

**Made at Home**
- You Manage

**Take and Bake**
- Vendor Manages

**Pizza Delivery**
- You Manage

**Dining Out**
- Vendor Manages
In reality....
The five pillars of the AWS framework

- Operational Excellence
- Security
- Reliability
- Performance Efficiency
- Cost Optimization

Where to focus IR efforts?

- Host
- IAM (a.k.a. role-based access)
- Data storage (e.g., S3)
- Persistence (e.g., when odd things change)
  - S3 bucket permissions
  - Security groups
  - Network gateways
  - EC2 instance ownership
  - Authorization failures
Breaking it down

Bad guy

Leverages

IAM

Attacks

EC2

S3

We detect via

CloudWatch

CloudTrail
Poll the Audience

- CSV-R01

What of the following do you see as roadblock to AWS Incident Response?

A. Not enough authority
B. Knowledge of landscape
C. Lack of skills

https://rsa1-live.eventbase.com/polls?event=rsa2020&session=1454108104
AWS Incident Response Use Cases
Use Case 1: What really happened at Capital One?

- An old vulnerability called Server Side Request Forgery (SSRF)
  - On the Web Application Firewall (WAF) - ModSecurity

- The WAF was misconfigured
  - Able to make a metadata service request
  - Which is how the attacker gained access to credentials
  - Credentials had access to whatever resource requested them
  - WAF assigned too much privilege
  - Could list information contained within S3 storage buckets

- AWS has since added additional authentication to the service
A visual
Why should you care?

- Cloud misconfigurations can have greater impact if exploited
  - Versus on premises misconfigurations
- A bad actor much more likely to access if internet-facing
Use Case 2: Oh no, not more!

- Ever heard of Code Spaces?
  - Maybe not since it’s been dead since 2014
  - It was a site that hosted source code repositories and offered project management services
  - Mostly hosted on AWS

- An attacker gained access to their AWS console
  - Held it for ransom
  - When no payment, started deleting...everything
  - Elastic Block Storage (EBS) Snapshots, S3 data, some server instances
All eggs in one basket

- All the data that was deleted included backups
- Effectively put Code Spaces out of business
So how do I respond to that?

- Follow the breadcrumbs
  - Read the logs (Cloud Trail, Cloud Watch)
  - Understand the flow
  - Understand what your scope of response is
    - An internal shared responsibility model?
  - Understand how things are configured
    - And did they behave as expected?
aws

Description

The AWS Command Line Interface is a unified tool to manage your AWS resources.

Synopsis

```
aws [options] <command> <subcommand> [parameters]
```

Use `aws command help` for information on a specific command. Use `aws help` for a list of available help topics. The synopsis for each command shows its parameters. Examples are shown in square brackets.

Options

- `--debug` (boolean)
  
  Turn on debug logging.
Command line interface tips

- You’ll have to install the environment
  - Looks like DOS! (seriously!)

- Run `aws-configure`
  - Consider – debug to get all the interactive detail
  - Do you have a proxy? May need to pass through/tunnel.
  - If you do tunnel, may need to import certs
    - `set REQUESTS_CA_BUNDLE=full-path-to\[name of].pem`

- Potential for automation
Some examples

- Want to copy a snapshot?
  
  ```bash
  aws ec2 copy-snapshot \ 
  - region us-east-1 \ 
  - source-region us-west-2 \ 
  - source-snapshot-id snap-066877671789bd71b \ 
  - description "This is my copied snapshot."
  ```
Another example

- Change a security group?

```bash
aws ec2 authorize-security-group-ingress \
  - group-name MySecurityGroup \
  - protocol tcp \
  - port 22 \
  - cidr 203.0.113.0/24
```
Other options

- **Crash cart**
  - What tools might you want to have available?
    - Depends upon incident response scope
    - Remnux for malware analysis
    - Other gems?

- **Breakglass**
  - May be more palatable for Incident Response to have emergency only access

- **Lambda**
  - What can you automate through scripting?

- **Who contacts Amazon in case of an incident?**
ThreatResponse
A Free Open Source Security Suite for Hardening and Responding in AWS

- Ashland, OR
- http://www.threatresponse.cloud
- @threatresponse.cloud

Repositories 39

**margaritashotgun**
Remote Memory Acquisition Tool
- Python
- MIT
- 25
- 133
- 11 issues need help
- Updated on Oct 22

**aws_ir**
Python installable command line utility for mitigation of host and key compromises.
- Python
- MIT
- 43
- 214
- 12 issues need help
- Updated on Feb 28

Top languages
- Python
- CSS
- Shell
- JavaScript
- HTML

People
Use Case 3: Let’s Run Through An Incident End To End

- Your Macie service has identified odd behavior via a key pair
  - Key pair is accessing an account it’s never logged into before
  - Is this a security incident?
- Work through four stages of an Incident:
  - Prepare
  - Detect
  - Contain
  - Post-Incident
Run It Down...

- Detect
  - How would you determine what IAM User the access keys belonged to?
    - How were the access keys used?
  - Which log would access to keys show up in? (access, use)
    - What would the log tell you?
    - How do you sift through all those events?
  - Would you have an alert any where when they were accessed?
    - Depends on the action
  - What regions were the keys used in?
    - Were they limited to specific regions?
A quick demo....
Wrapping Up…

● Contain
  – What actions need to be taken to mitigate?
  – Who has the permission to do it?
    ○ Incident Response Team?
    ○ IAM Team?
  – Who needs to be notified?
    ○ Application Team?
    ○ End User?

● Post-Incident
  – How did the credentials get posted?
    ○ Was it posted from within your network?
    ○ Code for the demo available here: https://github.com/kameenan/RSAC2020
Acquiring AWS Skills
Start of available resources

- https://www.aws.training/ (need to set up an account)
- https://aws.amazon.com/free (get your own hands on!)
- https://www.coursera.org/ (search for AWS [topic])
- https://github.com/jlevy/og-aws (awesome find)
- https://aws.amazon.com/
AWS training

Recommended

Getting Started

Amazon SageMaker: Build an Object Detection Model Using
1. INTERMEDIATE 2. 30 MINUTES

AWS Cloud Practitioner Essential (Second Edition) (Traditional
1. FUNDAMENTAL
AWS Security Fundamentals (Second Edition)

ABOUT

Description

In this self-paced course, you will learn fundamental AWS cloud security concepts, including AWS access control, data encryption methods, and how network access to your AWS infrastructure can be secured. We will address and your security responsibility in the AWS cloud and the different security-oriented services available.

Intended Audience

This course is intended for:

- IT business-level professionals interested in cloud security practices
Youtube. OMG.

- Go there
- Search for Amazon Web Service
- Find the very latest from re:Invent
- Learn a ton of stuff!
Apply – 90-day AWS incident response plan

- **30 days:**
  - Identify gaps between existing plan and what we’ve discussed
  - Begin acquiring needed skills

- **60 days:**
  - Confirm architecture of AWS (or other cloud provider) environment
  - Acquire sandbox environment
  - Document processes to support

- **90 days:**
  - Consider automation for certain tasks
  - Test out processes
    - Table tops!
Thank you!

Keep the conversation going!

kmwestphal@cox.net