



Securosis

Pragmatic Cloud Security Automation

Rich Mogull/Crash/@rmogull
Securosis and DisruptOps

Cloud is Fundamentally Different

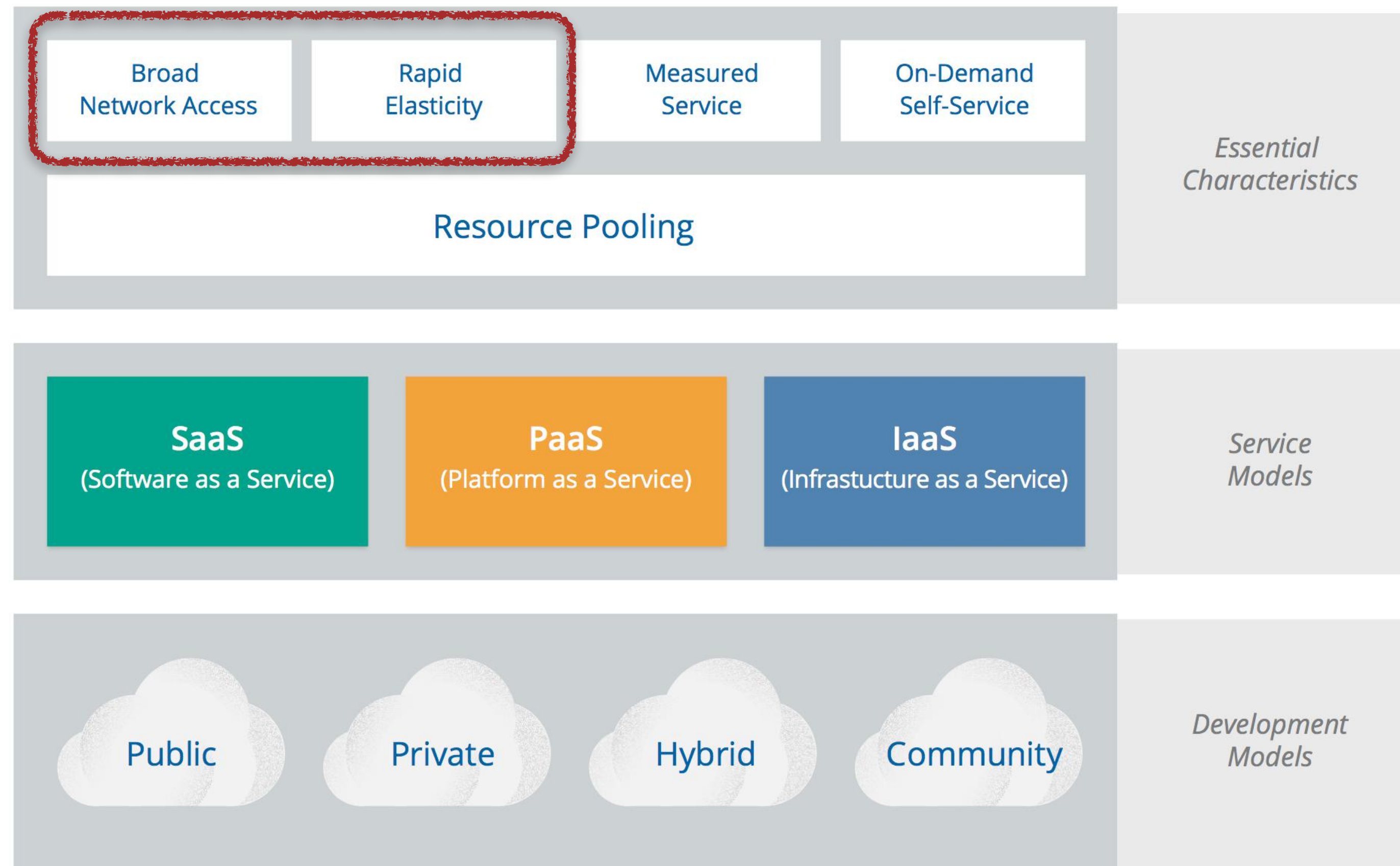


Abstraction



Automation

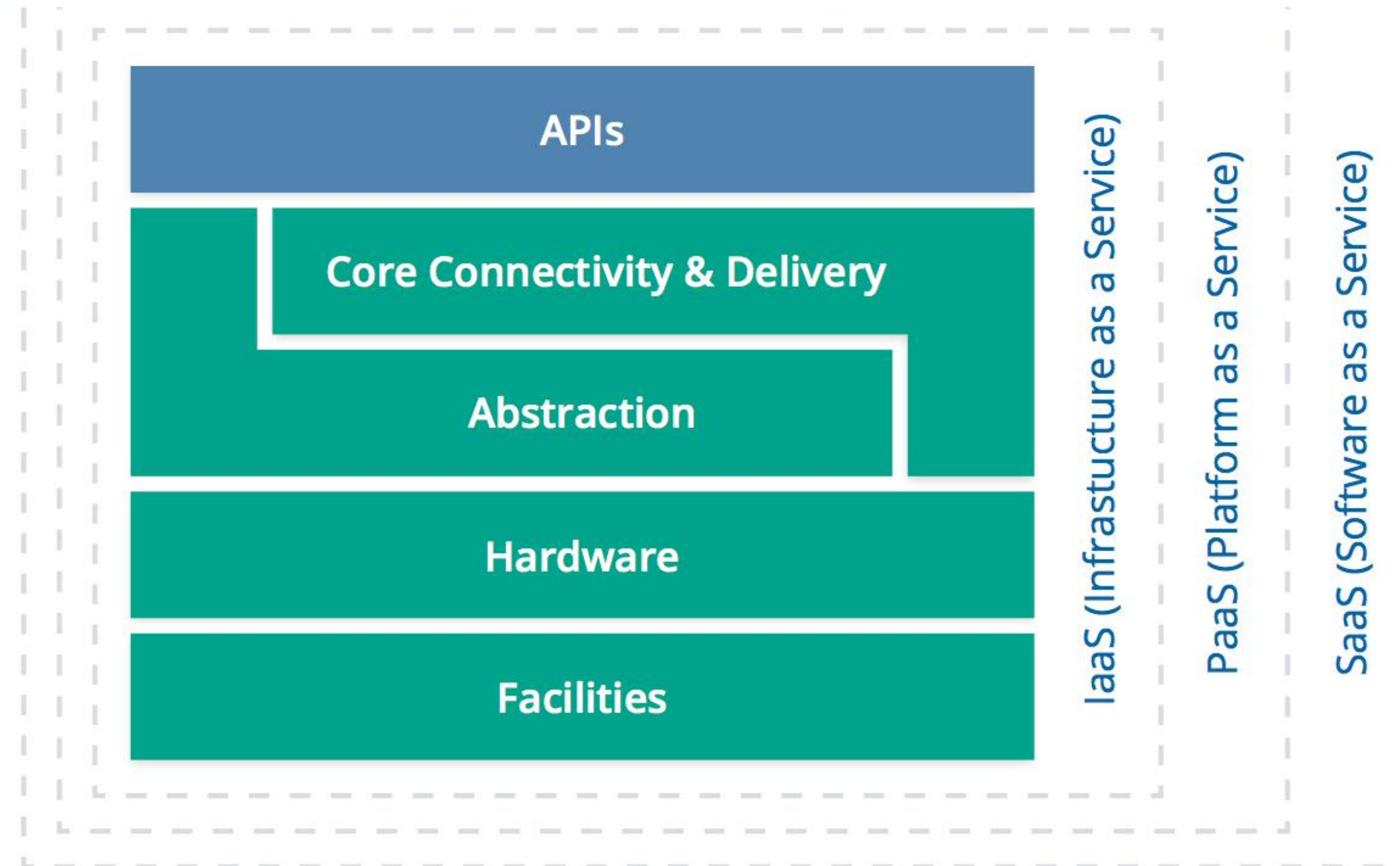
Automation is Inherent



The NIST Model (courtesy the CSA)

APIs are Ubiquitous

Cloud Security Alliance
IaaS Reference Model }



Cloud Security Must Be Cloud Native

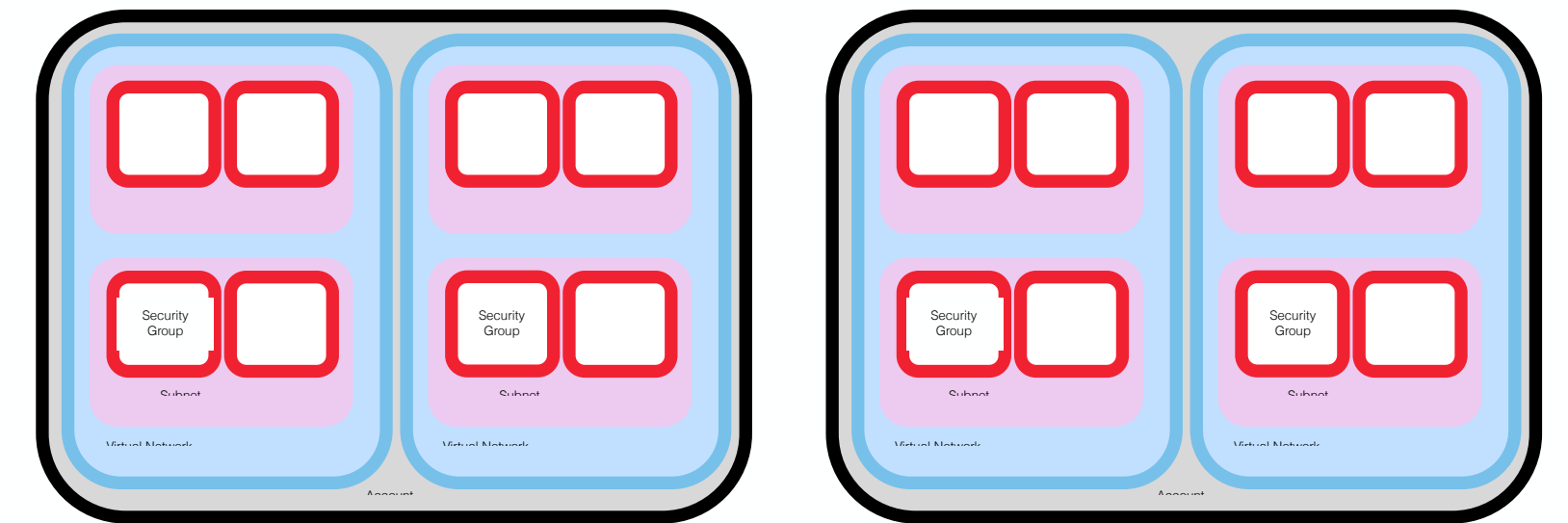
Management Plane



Volatility/Velocity



Distribution/Segregation



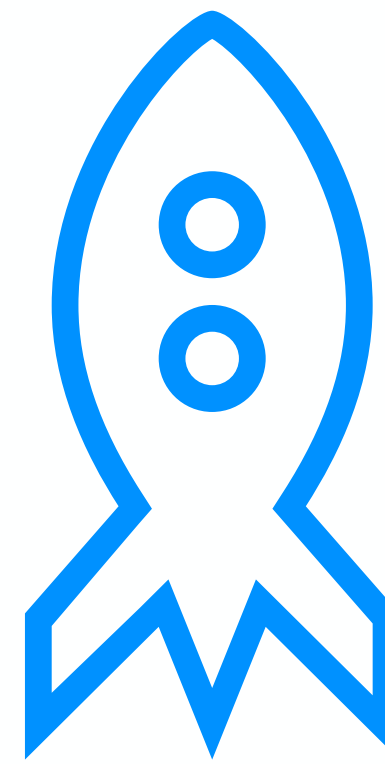
The Categories



Guardrails

Continuously assess and enforce operational and security policies

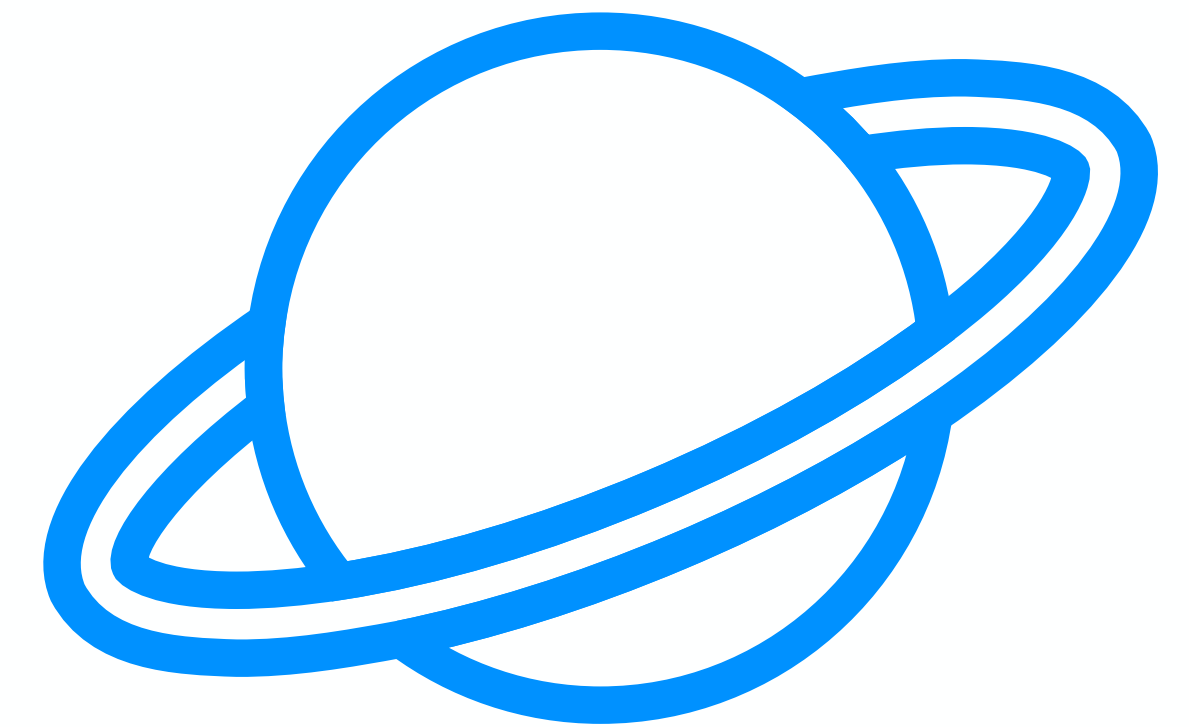
Fix security group or S3 misconfigurations



Workflows

Streamline and accelerate IT operations and security through automated workflows

Incident response

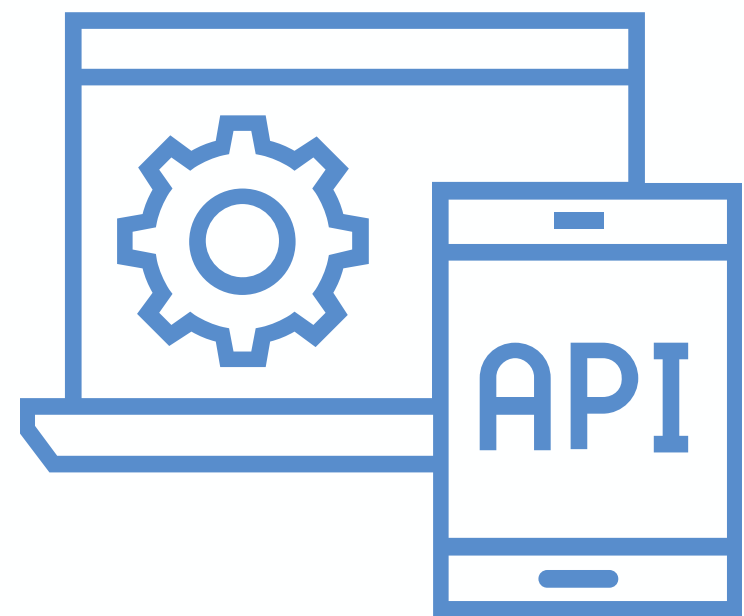


Orchestrations

Empower new capabilities through advanced orchestration of infrastructure, operations, and security

Automatic WAF insertion and configuration

The Principles



Software
Defined
Security



Stateless
Security



Event Driven
Security



Continuous
Feedback
Loops

The Foundation

Cloud Service Provider

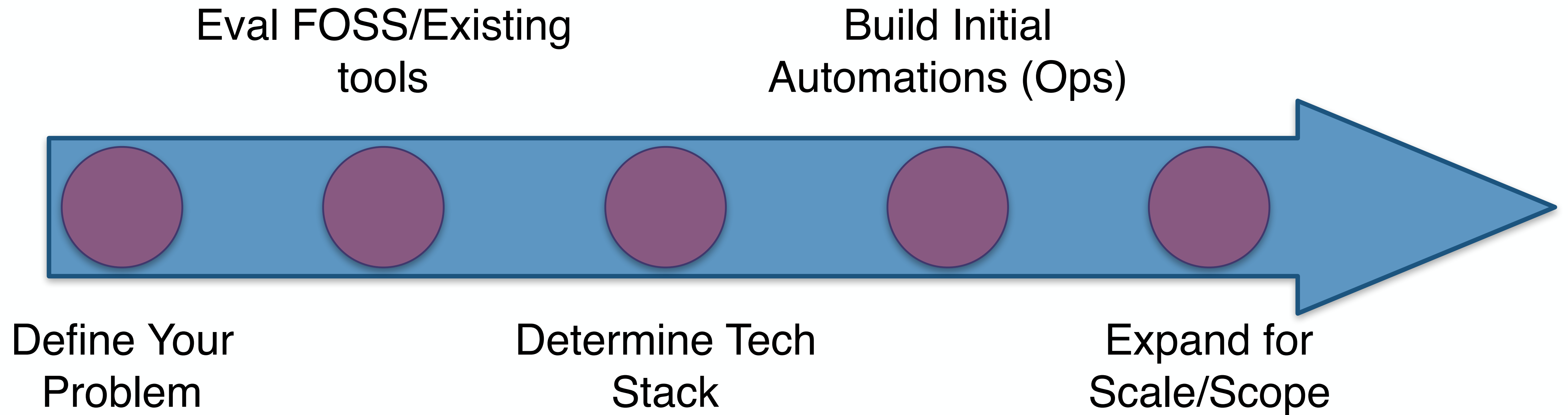
- ▶ API and full administrative activity logging
- ▶ Events/triggers/rules
- ▶ Function as a Service (Serverless)
- ▶ Notification service

Critical Capabilities

Cloud Consumer (you)

- ▶ Continuous Integration Pipeline
- ▶ Version control repository
- ▶ Full IAM access to accounts/subscriptions/projects
- ▶ Security development team (person)

The Process



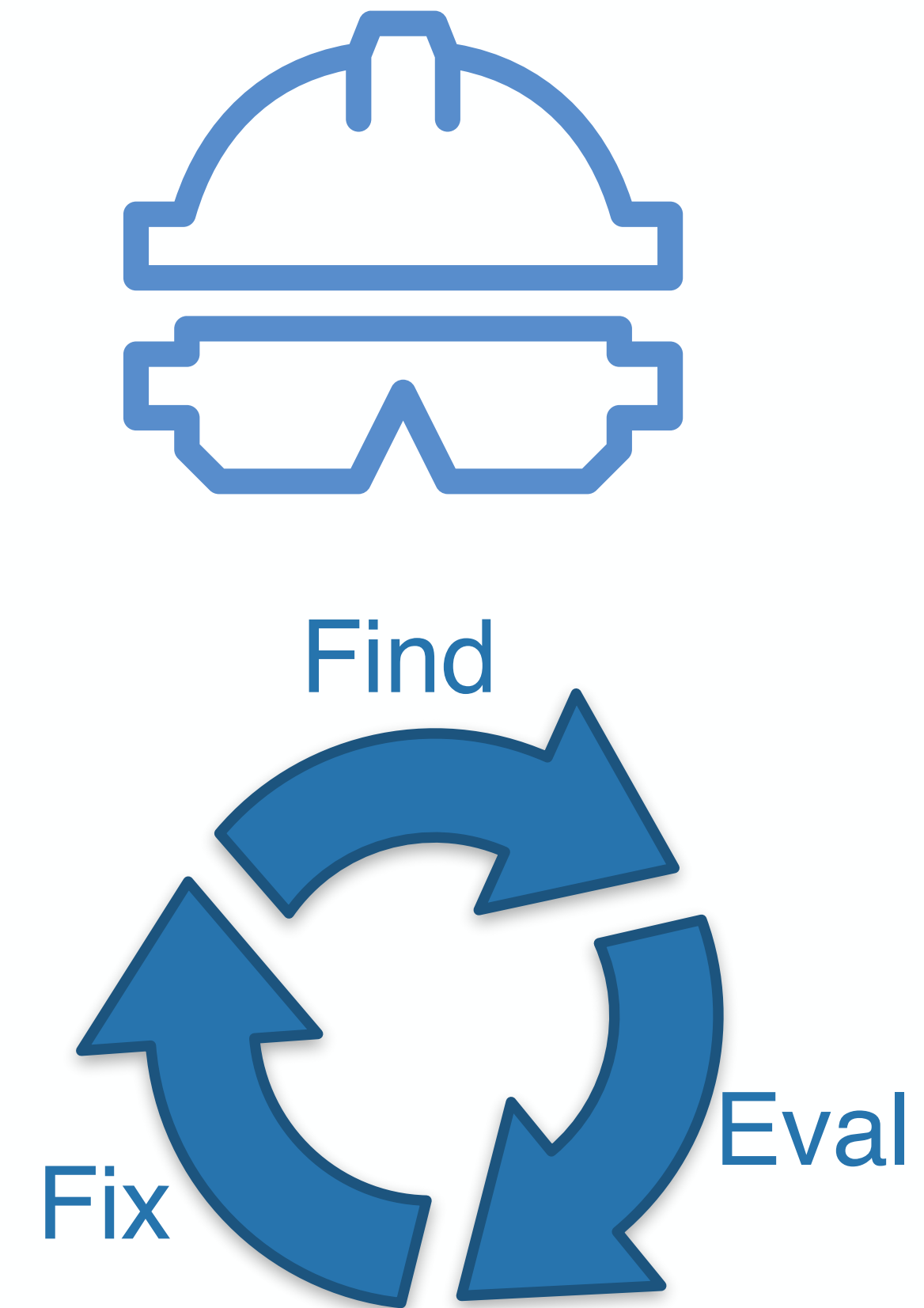
Things We Are Skipping (for time)



- ▶ How to configure all the core monitoring/logging
- ▶ Setting up IAM and permissions
- ▶ The details of implementation on Azure and GCP
- ▶ We will list the core capabilities, but can't cover all 3 with real examples in 45 minutes

What's a Guardrail?

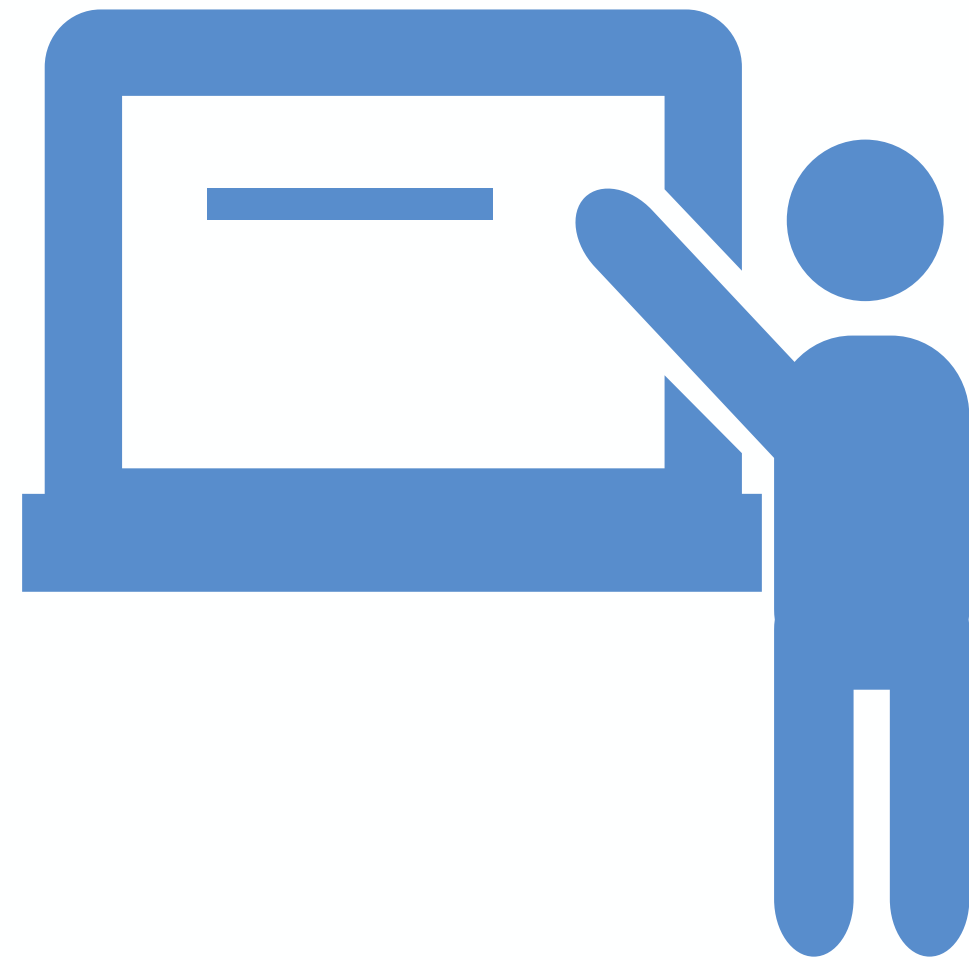
- Define and set limits
 - Can be “allow” or “deny”
- Find deviations
 - Assessment or event based
- Evaluate the issue
- Fix/remediate
 - Automatically or manually depending on rules



Example Guardrails

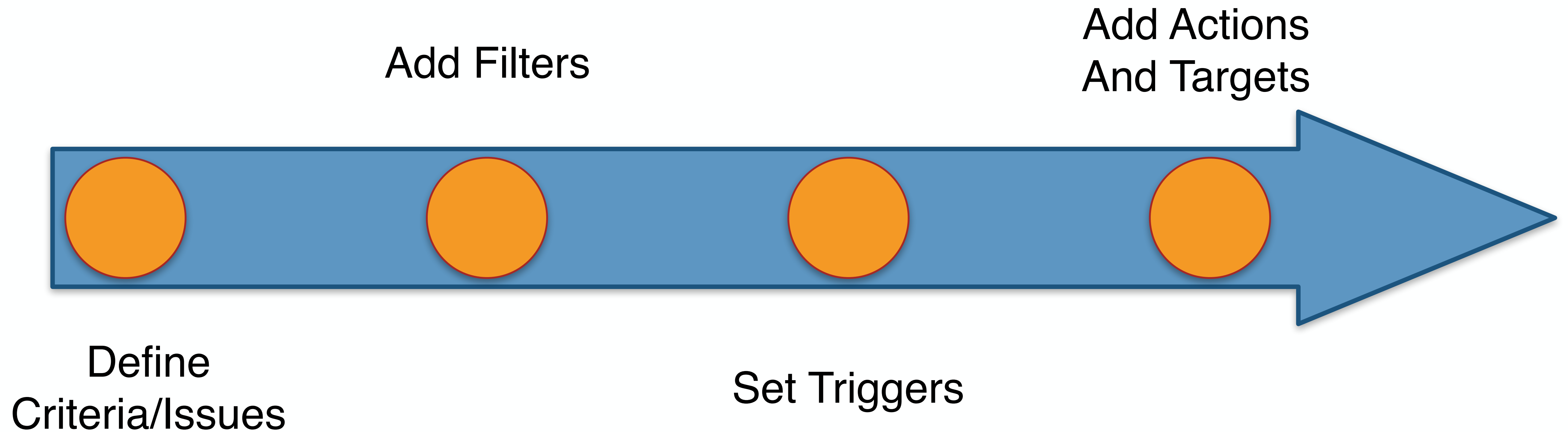
- If you find a public S3 bucket, restrict it to our known network addresses
 - Unless it is approved or tagged
 - Don't allow internal security groups with all ports and protocols open in Prod
 - But allow in Dev
 - Require MFA for API access for any user that needs MFA for console access
 - Create our baseline IAM policies and roles for all new accounts
 - Based on the environment
- Validate that monitoring and alerting is properly configured
 - And fix if not
 - Disable access keys that haven't been used in 90 days
 - Find instances with an IAM role that allows power user or greater access via API
 - Restrict the privileges
 - Identify all cross-network peering from accounts we don't own
 - Then check the security group permissions

What Makes a *Good* Guardrail?



- Accounts for different environments
 - At least Dev vs. Prod
- Handles exceptions
 - And is capable of remembering them
- Understands state and context
- Doesn't bog down the alert queue
- Can remediate automatically
 - Either completely, or after manual approval
- Ops communications/notifications
- Education, not Blamification

Building a Guardrail



Our Guardrail

- ▶ Criteria/Issues
 - ▶ All instances with port 22 open to the 0.0.0.0/0 (the Internet)
- ▶ Filters
 - ▶ Region is us-west-2 (could be VPC/tag/etc)
- ▶ Trigger
 - ▶ Time = every 5 minutes
- ▶ Action
 - ▶ Restrict to known IP range

Step 1: Create rule

Create rules to invoke Targets based on Events happening in your AWS environment

Event Source

Build or customize an Event Pattern or set a Schedule to invoke Targets.

☐ Event Pattern ⓘ ☒ Schedule ⓘ

☒ Fixed rate of

☐ Cron expression

[Learn more](#) about CloudWatch Events schedules.

▶ Show sample event(s)

* Required



Code Walk Through

- Key aspects:
 - Authentication/authorization via Roles
 - Initializing clients
 - Understanding method and variable scope
 - AWS SDK/JSON navigation
 - Structs > hash > arrays
 - Hidden complexities (e.g. ENIs and security groups)
- Tips
 - Waiters
 - Managing API limits
 - CLI vs. SDK (`--query`)

Whiteboard

Coding Recommendations

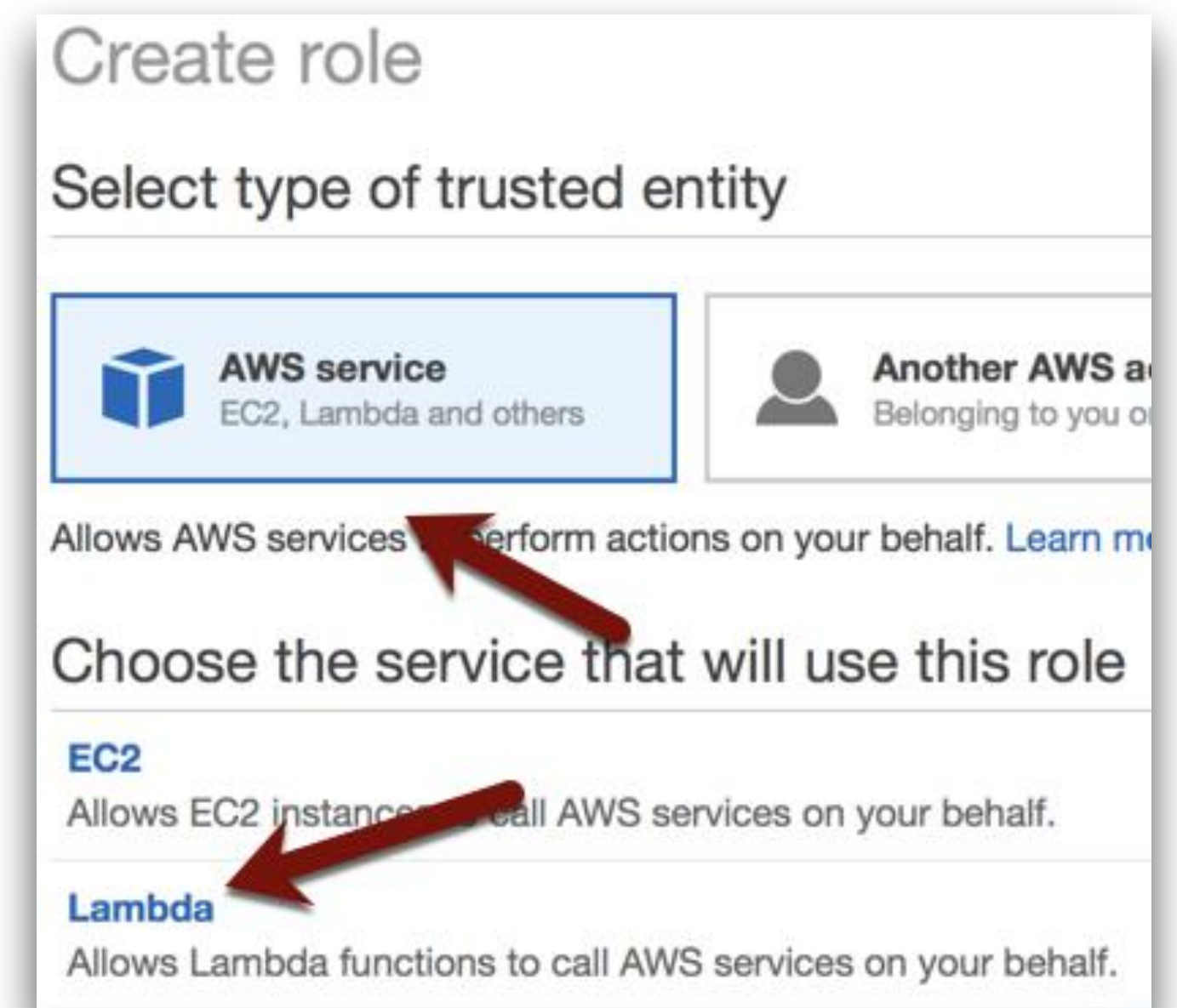
- ▶ Language doesn't matter... as long as it supports Lambda
- ▶ Understand the AWS credentials hierarchy
 - ▶ Hard coded > specified credentials file > default config and credentials files > role
- ▶ API limits are a thing. They suck
 - ▶ Paginators are your friend when available
- ▶ Make sure you understand how to use server side filtering and when it hurts more than it helps

Lab: Create Time-Based Guardrail

22

- ▶ Create a new IAM role to run the Lambda functions for today
 - ▶ Give it AdministratorAccess policy only to speed things up
 - ▶ NEVER EVER DO THIS IN REAL LIFE!!!
 - ▶ (Yes, I've found it in evaluations)
- ▶ Name it lambda_admin

**Use the
SharedServices
account**



Lab: Create Notification (or use one from an earlier lab)

- ▶ Create a new topic, or pick your existing topic, from SNS
- ▶ Make sure you have an active subscription (e.g. SMS or email) to receive the notifications
- ▶ Copy and paste the topic ARN to your cheat sheet

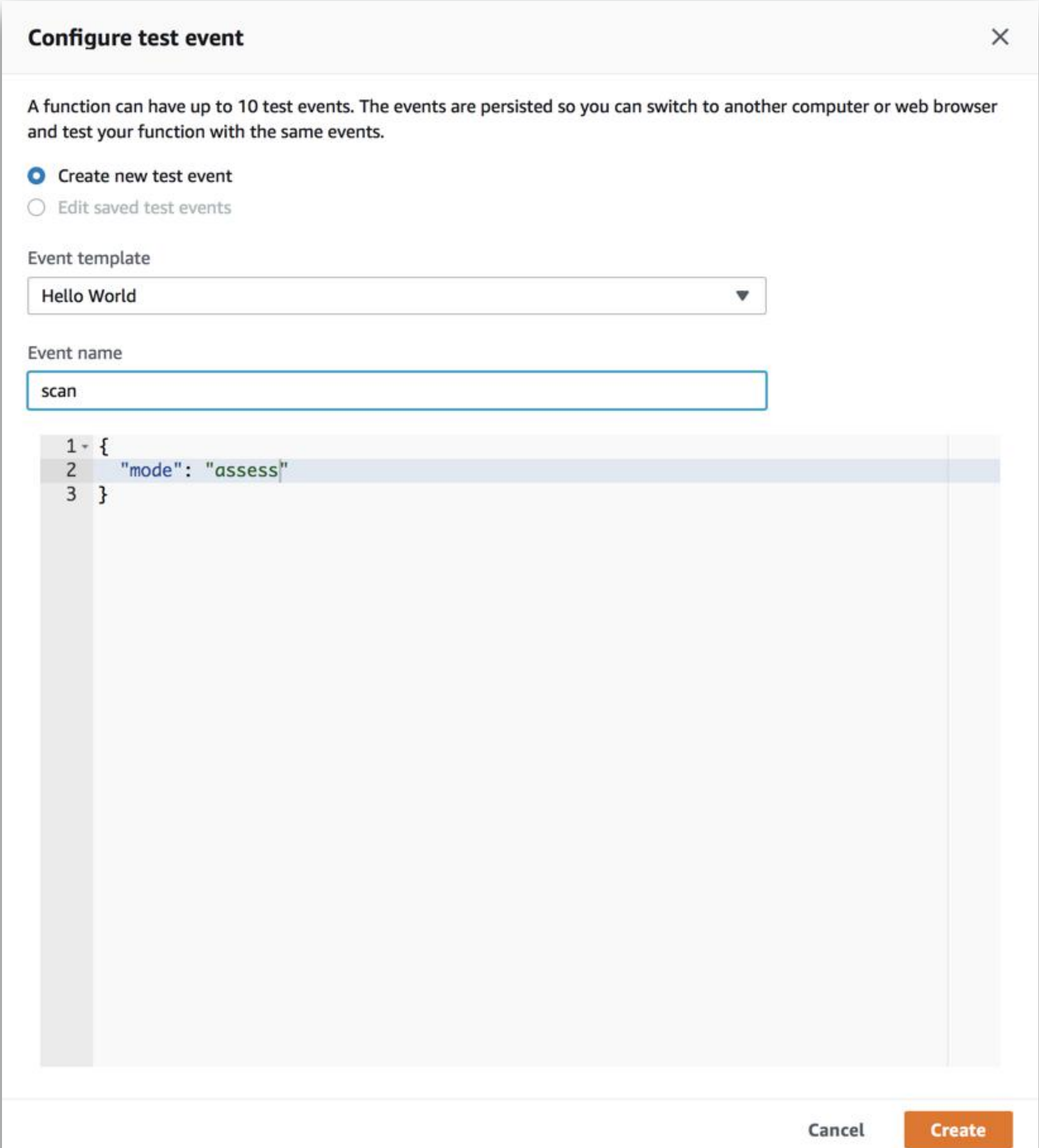
Lab: Create the Function

- ▶ Create a new Lambda function
 - ▶ Name it identify_internet_facing_servers
 - ▶ Choose Python 3.x
 - ▶ Choose the lambda_admin role
- ▶ Paste in the sample code from your student directory
 - ▶ If you are a hacker, or ever wanted to be a hacker, figure out how to change to dark mode.
- ▶ If you hit an error wait 1-2 minutes and try again, sometimes IAM is slow. Welcome to the cloud!

```
lambda_function x
1 import boto3
2 import json
3
4 # Updates needed- IPV6, an IP range that includes port 22
5 # Also add fix in case requested IP range is already in there
6
7 def update_security_group(event, group, existingIpPermission, port, new_range):
8     if "clickType" not in event:
9         ec2 = boto3.client('ec2', region_name=event)
10    else:
11        ec2 = boto3.client('ec2')
12    sub_range = []
13    for curIP in existingIpPermission['IpRanges']:
14        if curIP['CidrIp'] != '0.0.0.0/0':
15            sub_range.append(curIP)
16    sub_range.append(new_range)
17    remove_rule = ec2.revoke_security_group_ingress(
18        GroupId=group,
19        IpPermissions=[
20            {'IpProtocol': 'tcp',
21             'FromPort': port,
22             'ToPort': port,
23             'IpRanges': existingIpPermission['IpRanges']}
24        ])
25    add_rule = ec2.authorize_security_group_ingress(
26        GroupId=group,
```

Lab: Create Test Event

- ▶ Create the test event (it's on the top of the Lambda page)
- ▶ Paste in the sample JSON from your cheat sheet (it's under ### Guardrail)
- ▶ Replace with the ARN of your SNS topic
- ▶ Update the SNS ARN in the Lambda function: "TargetArn" around line 110



The screenshot shows the 'Configure test event' dialog box in the AWS Lambda console. It includes a close button (X) in the top right corner. Below the title, there is a descriptive text: 'A function can have up to 10 test events. The events are persisted so you can switch to another computer or web browser and test your function with the same events.' There are two radio buttons: 'Create new test event' (selected) and 'Edit saved test events'. Below this is an 'Event template' dropdown menu currently set to 'Hello World'. An 'Event name' text input field contains the word 'scan'. A large text area for the event payload shows a JSON snippet:

```
1 {  
2   "mode": "assess"  
3 }
```

 At the bottom right, there are 'Cancel' and 'Create' buttons.

Configure test event

A function can have up to 10 test events. The events are persisted so you can switch to another computer or web browser and test your function with the same events.

☒ Create new test event
☐ Edit saved test events

Event template
Hello World

Event name
scan

```
1 {  
2   "mode": "assess"  
3 }
```

Cancel Create

Test

Test

✓

Execution result: succeeded (logs)

✕

▼

Details

The area below shows the result returned by your function execution.

null

Summary

Code SHA-256	jE6w+hOYjz5ZiIYuHQ7tc7Tj7ErAN3KBIATR7CO7C5w=	Request ID	f4beabf1-83a1-11e8-addc-79cb7b7572de
Duration	1584.43 ms	Billed duration	1600 ms
Resources configured	128 MB	Max memory used	37 MB

Log output

The area below shows the logging calls in your code. These correspond to a single row within the CloudWatch log group corresponding to this Lambda function. [Click here](#) to view the CloudWatch log group.

```
[{"cidrIp": "0.0.0.0/0"}, {"ipV6Ranges": [], "prefixListIds": [], "userIdGroupPairs": [], "vpcId": "vpc-fbc4a793"}]
SGs [{"Description": "Temporary group for Packer", "GroupName": "packer_5afcc818-8aae-faa1-8d95-1edc92810fad", "IpPermissions": [{"FromPort": 22, "IpProtocol": "tcp", "IpRanges": [{"CidrIp": "0.0.0.0/0"}], "Ipv6Ranges": [], "PrefixListIds": [], "ToPort": 22, "UserIdGroupPairs": []}], "OwnerId": "935440313651", "GroupId": "sg-1a40aa6b", "IpPermissionsEgress": [{"IpProtocol": "-1", "IpRanges": [{"CidrIp": "0.0.0.0/0"}], "Ipv6Ranges": [], "PrefixListIds": [], "UserIdGroupPairs": []}], "VpcId": "vpc-fbc4a793"}, {"Description": "Temporary group for Packer", "GroupName": "packer_5b19c495-c23d-0d03-e1e9-501aab880c0c", "IpPermissions": [{"FromPort": 22, "IpProtocol": "tcp", "IpRanges": [{"CidrIp": "0.0.0.0/0"}], "Ipv6Ranges": [], "PrefixListIds": [], "ToPort": 22, "UserIdGroupPairs": []}], "OwnerId": "935440313651", "GroupId": "sg-893e50f8", "IpPermissionsEgress": [{"IpProtocol": "-1", "IpRanges": [{"CidrIp": "0.0.0.0/0"}], "Ipv6Ranges": [], "PrefixListIds": [], "UserIdGroupPairs": []}], "VpcId": "vpc-fbc4a793"}]
END RequestId: f4beabf1-83a1-11e8-addc-79cb7b7572de
REPORT RequestId: f4beabf1-83a1-11e8-addc-79cb7b7572de  Duration: 1584.43 ms    Billed Duration: 1600 ms    Memory Size: 128 MB    Max Memory Used: 37 MB
```

Lab: Set Schedule

Step 1: Create rule

Create rules to invoke Targets based on Events happening in your AWS environment.

Event Source

Build or customize an Event Pattern or set a Schedule to invoke Targets.

☐ Event Pattern ⓘ ☒ Schedule ⓘ

☒ Fixed rate of

☐ Cron expression

[Learn more about CloudWatch Events schedules.](#)

▶ Show sample event(s)

Targets

Select Target to invoke when an event matches your Event Pattern or when schedule is triggered.

Lambda function

Function*

▶ Configure version/alias

▼ Configure input

☐ Matched event ⓘ

☐ Part of the matched event ⓘ

☒ Constant (JSON text) ⓘ

☐ Input Transformer ⓘ

+ Add target*

* Required

Cancel **Configure details**

- ▶ Create a CloudWatch Rule to run the lambda function every 5 minutes (or sooner if you want)
- ▶ Provide the configuration details by pasting in the JSON from your test (e.g. “mode”: “assess”)

Now try putting it into
remediation mode

Our Event-Driven Guardrail

- ▶ Criteria/Issues
 - ▶ New inbound security group rule added
- ▶ Filters
 - ▶ IAM user, VPC, Tag
- ▶ Trigger
 - ▶ API event (CloudTrail)
- ▶ Action
 - ▶ Reverse + Notify



Environment

fixSecurityGroup
lambda_function.py

lambda_function.py x +
1 from __future__ import print_function
2
3 # A demonstration AWS Lambda function to revert a security group change based on a CloudWatch event.
4 # By Rich Mogull and Securosis, released under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International license.
5 # http://creativecommons.org/licenses/by-nc-sa/4.0/ ***** https://securosis.com
6
7 # This lambda function will reverse any security group changes when triggered by a CloudWatch event
8 # that shows an ingress change. It *does not* work for outbound changes, but as you can see
9 # if you review the code it could easily be modified for that.
10 # The demonstration code also includes various conditional options. If you don't use one of these
11 # this will apply to every change in your account. To use them, modify the parameters and then
12 # move the function call into the conditional block.
13
14 import json
15 import urllib
16 import boto3
17
18 print('Loading function')
19
20 ec2 = boto3.client('ec2')
21
22
23 def lambda_handler(event, context):
24 # dump the raw event for log purposes
25 print("An unauthorized security group change was detected and will be remediated. The event details are:")
26 print("-----")
27 print("Received event: " + json.dumps(event, indent=5))
28 print("-----")
29
30 # Only execute if the change was in a certain region
31 if event["detail"]["awsRegion"] == "us-west-2":
32 print("Region is us-west-2")
33
34 # Only execute if the change was not from a designated admin account
35 # Only execute if the change was not from a designated admin account
36
37 # Only execute if the change was not from a designated admin account
38
39 # Only execute if the change was not from a designated admin account
40
41 # Only execute if the change was not from a designated admin account
42
43 # Only execute if the change was not from a designated admin account
44
45 # Only execute if the change was not from a designated admin account
46
47 # Only execute if the change was not from a designated admin account
48
49 # Only execute if the change was not from a designated admin account
50
51 # Only execute if the change was not from a designated admin account
52
53 # Only execute if the change was not from a designated admin account
54
55 # Only execute if the change was not from a designated admin account
56
57 # Only execute if the change was not from a designated admin account
58
59 # Only execute if the change was not from a designated admin account
60
61 # Only execute if the change was not from a designated admin account
62
63 # Only execute if the change was not from a designated admin account
64
65 # Only execute if the change was not from a designated admin account
66
67 # Only execute if the change was not from a designated admin account
68
69 # Only execute if the change was not from a designated admin account
70
71 # Only execute if the change was not from a designated admin account
72
73 # Only execute if the change was not from a designated admin account
74
75 # Only execute if the change was not from a designated admin account
76
77 # Only execute if the change was not from a designated admin account
78
79 # Only execute if the change was not from a designated admin account
80
81 # Only execute if the change was not from a designated admin account
82
83 # Only execute if the change was not from a designated admin account
84
85 # Only execute if the change was not from a designated admin account
86
87 # Only execute if the change was not from a designated admin account
88
89 # Only execute if the change was not from a designated admin account
90
91 # Only execute if the change was not from a designated admin account
92
93 # Only execute if the change was not from a designated admin account
94
95 # Only execute if the change was not from a designated admin account
96
97 # Only execute if the change was not from a designated admin account
98
99 # Only execute if the change was not from a designated admin account
100
101 # Only execute if the change was not from a designated admin account
102
103 # Only execute if the change was not from a designated admin account
104
105 # Only execute if the change was not from a designated admin account
106
107 # Only execute if the change was not from a designated admin account
108
109 # Only execute if the change was not from a designated admin account
110
111 # Only execute if the change was not from a designated admin account
112
113 # Only execute if the change was not from a designated admin account
114
115 # Only execute if the change was not from a designated admin account
116
117 # Only execute if the change was not from a designated admin account
118
119 # Only execute if the change was not from a designated admin account
120
121 # Only execute if the change was not from a designated admin account
122
123 # Only execute if the change was not from a designated admin account
124
125 # Only execute if the change was not from a designated admin account
126
127 # Only execute if the change was not from a designated admin account
128
129 # Only execute if the change was not from a designated admin account
130
131 # Only execute if the change was not from a designated admin account
132
133 # Only execute if the change was not from a designated admin account
134
135 # Only execute if the change was not from a designated admin account
136
137 # Only execute if the change was not from a designated admin account
138
139 # Only execute if the change was not from a designated admin account
140
141 # Only execute if the change was not from a designated admin account
142
143 # Only execute if the change was not from a designated admin account
144
145 # Only execute if the change was not from a designated admin account
146
147 # Only execute if the change was not from a designated admin account
148
149 # Only execute if the change was not from a designated admin account
150
151 # Only execute if the change was not from a designated admin account
152
153 # Only execute if the change was not from a designated admin account
154
155 # Only execute if the change was not from a designated admin account
156
157 # Only execute if the change was not from a designated admin account
158
159 # Only execute if the change was not from a designated admin account
160
161 # Only execute if the change was not from a designated admin account
162
163 # Only execute if the change was not from a designated admin account
164
165 # Only execute if the change was not from a designated admin account
166
167 # Only execute if the change was not from a designated admin account
168
169 # Only execute if the change was not from a designated admin account
170
171 # Only execute if the change was not from a designated admin account
172
173 # Only execute if the change was not from a designated admin account
174
175 # Only execute if the change was not from a designated admin account
176
177 # Only execute if the change was not from a designated admin account
178
179 # Only execute if the change was not from a designated admin account
180
181 # Only execute if the change was not from a designated admin account
182
183 # Only execute if the change was not from a designated admin account
184
185 # Only execute if the change was not from a designated admin account
186
187 # Only execute if the change was not from a designated admin account
188
189 # Only execute if the change was not from a designated admin account
190
191 # Only execute if the change was not from a designated admin account
192
193 # Only execute if the change was not from a designated admin account
194
195 # Only execute if the change was not from a designated admin account
196
197 # Only execute if the change was not from a designated admin account
198
199 # Only execute if the change was not from a designated admin account
200
201 # Only execute if the change was not from a designated admin account
202
203 # Only execute if the change was not from a designated admin account
204
205 # Only execute if the change was not from a designated admin account
206
207 # Only execute if the change was not from a designated admin account
208
209 # Only execute if the change was not from a designated admin account
210
211 # Only execute if the change was not from a designated admin account
212
213 # Only execute if the change was not from a designated admin account
214
215 # Only execute if the change was not from a designated admin account
216
217 # Only execute if the change was not from a designated admin account
218
219 # Only execute if the change was not from a designated admin account
220
221 # Only execute if the change was not from a designated admin account
222
223 # Only execute if the change was not from a designated admin account
224
225 # Only execute if the change was not from a designated admin account
226
227 # Only execute if the change was not from a designated admin account
228
229 # Only execute if the change was not from a designated admin account
230
231 # Only execute if the change was not from a designated admin account
232
233 # Only execute if the change was not from a designated admin account
234
235 # Only execute if the change was not from a designated admin account
236
237 # Only execute if the change was not from a designated admin account
238
239 # Only execute if the change was not from a designated admin account
240
241 # Only execute if the change was not from a designated admin account
242
243 # Only execute if the change was not from a designated admin account
244
245 # Only execute if the change was not from a designated admin account
246
247 # Only execute if the change was not from a designated admin account
248
249 # Only execute if the change was not from a designated admin account
250
251 # Only execute if the change was not from a designated admin account
252
253 # Only execute if the change was not from a designated admin account
254
255 # Only execute if the change was not from a designated admin account
256
257 # Only execute if the change was not from a designated admin account
258
259 # Only execute if the change was not from a designated admin account
260
261 # Only execute if the change was not from a designated admin account
262
263 # Only execute if the change was not from a designated admin account
264
265 # Only execute if the change was not from a designated admin account
266
267 # Only execute if the change was not from a designated admin account
268
269 # Only execute if the change was not from a designated admin account
270
271 # Only execute if the change was not from a designated admin account
272
273 # Only execute if the change was not from a designated admin account
274
275 # Only execute if the change was not from a designated admin account
276
277 # Only execute if the change was not from a designated admin account
278
279 # Only execute if the change was not from a designated admin account
280
281 # Only execute if the change was not from a designated admin account
282
283 # Only execute if the change was not from a designated admin account
284
285 # Only execute if the change was not from a designated admin account
286
287 # Only execute if the change was not from a designated admin account
288
289 # Only execute if the change was not from a designated admin account
290
291 # Only execute if the change was not from a designated admin account
292
293 # Only execute if the change was not from a designated admin account
294
295 # Only execute if the change was not from a designated admin account
296
297 # Only execute if the change was not from a designated admin account
298
299 # Only execute if the change was not from a designated admin account
300
301 # Only execute if the change was not from a designated admin account
302
303 # Only execute if the change was not from a designated admin account
304
305 # Only execute if the change was not from a designated admin account
306
307 # Only execute if the change was not from a designated admin account
308
309 # Only execute if the change was not from a designated admin account
310
311 # Only execute if the change was not from a designated admin account
312
313 # Only execute if the change was not from a designated admin account
314
315 # Only execute if the change was not from a designated admin account
316
317 # Only execute if the change was not from a designated admin account
318
319 # Only execute if the change was not from a designated admin account
320
321 # Only execute if the change was not from a designated admin account
322
323 # Only execute if the change was not from a designated admin account
324
325 # Only execute if the change was not from a designated admin account
326
327 # Only execute if the change was not from a designated admin account
328
329 # Only execute if the change was not from a designated admin account
330
331 # Only execute if the change was not from a designated admin account
332
333 # Only execute if the change was not from a designated admin account
334
335 # Only execute if the change was not from a designated admin account
336
337 # Only execute if the change was not from a designated admin account
338

Self-Healing Infrastructure (yes, for real)

Type ⓘ	Protocol ⓘ	Port Range ⓘ	Source ⓘ
SSH	TCP	22	68.2.174.98/32
HTTP	TCP	80	0.0.0.0/0

Change a security group



Event Recorded to CloudTrail



Passed to CloudWatch Log Stream



Triggers an CloudWatch Event



Lambda Function analyzes and reverses

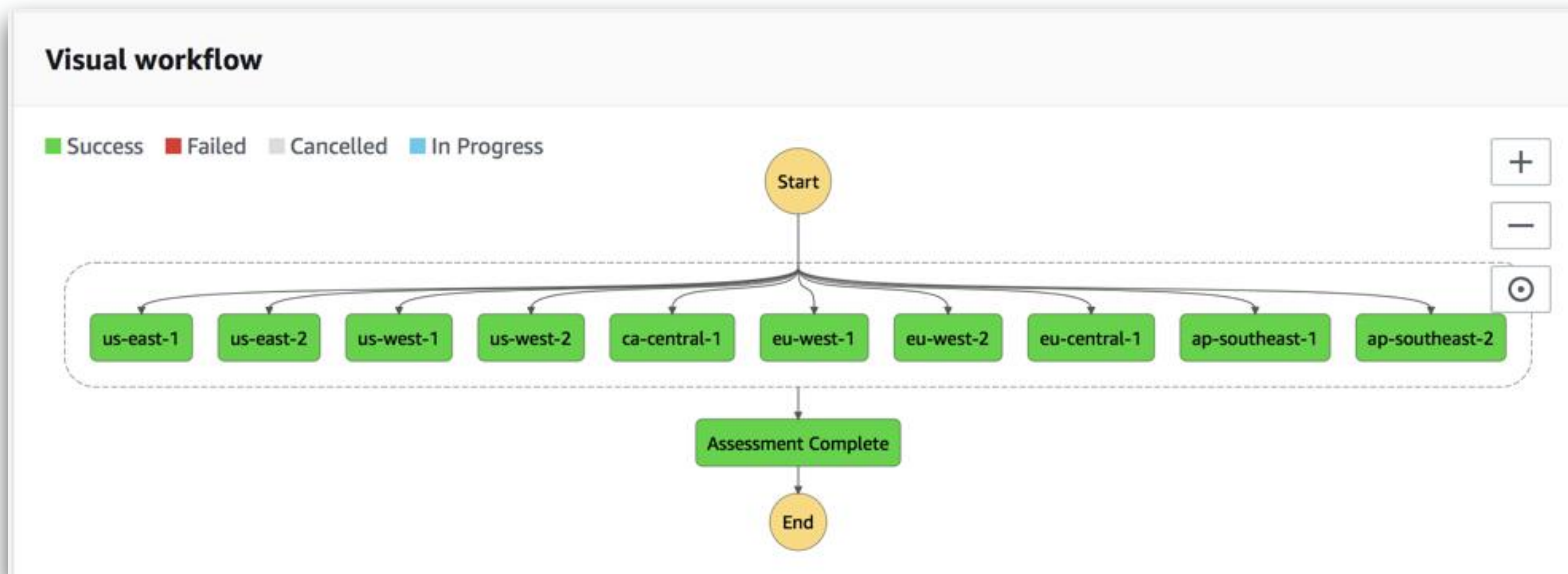


Lab: Event-Driven Guardrail

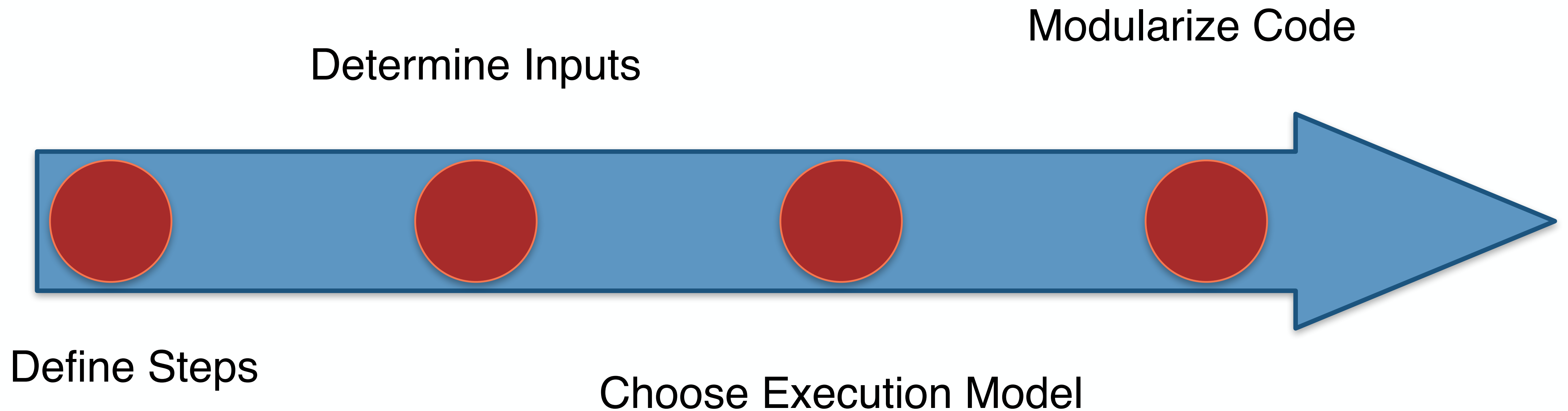
- ▶ Create a new lambda function using Python 2.7 and use the same role
- ▶ Paste in the content from the `revert_security_groups.py` file
- ▶ Either add the lambda as a second target to your existing alert or create a new CloudWatch rule to trigger this event anytime there is the API call “AuthorizeSecurityGroupIngress”
 - ▶ At this point, you should be able to figure this out
 - ▶ Pass in the raw event source to the Lambda
- ▶ Change a security group to test it
 - ▶ This version of the demo code only reverts an ingress authorization. It may also miss certain change operations
 - ▶ It does **not** revert IPV6 permissions if your VPC supports it

Expanding to Enterprise Scale

- Hitting all 14 regions simultaneously
- Multiplex
- Central event stream
- Queues/SNS
- AuthN/AuthZ



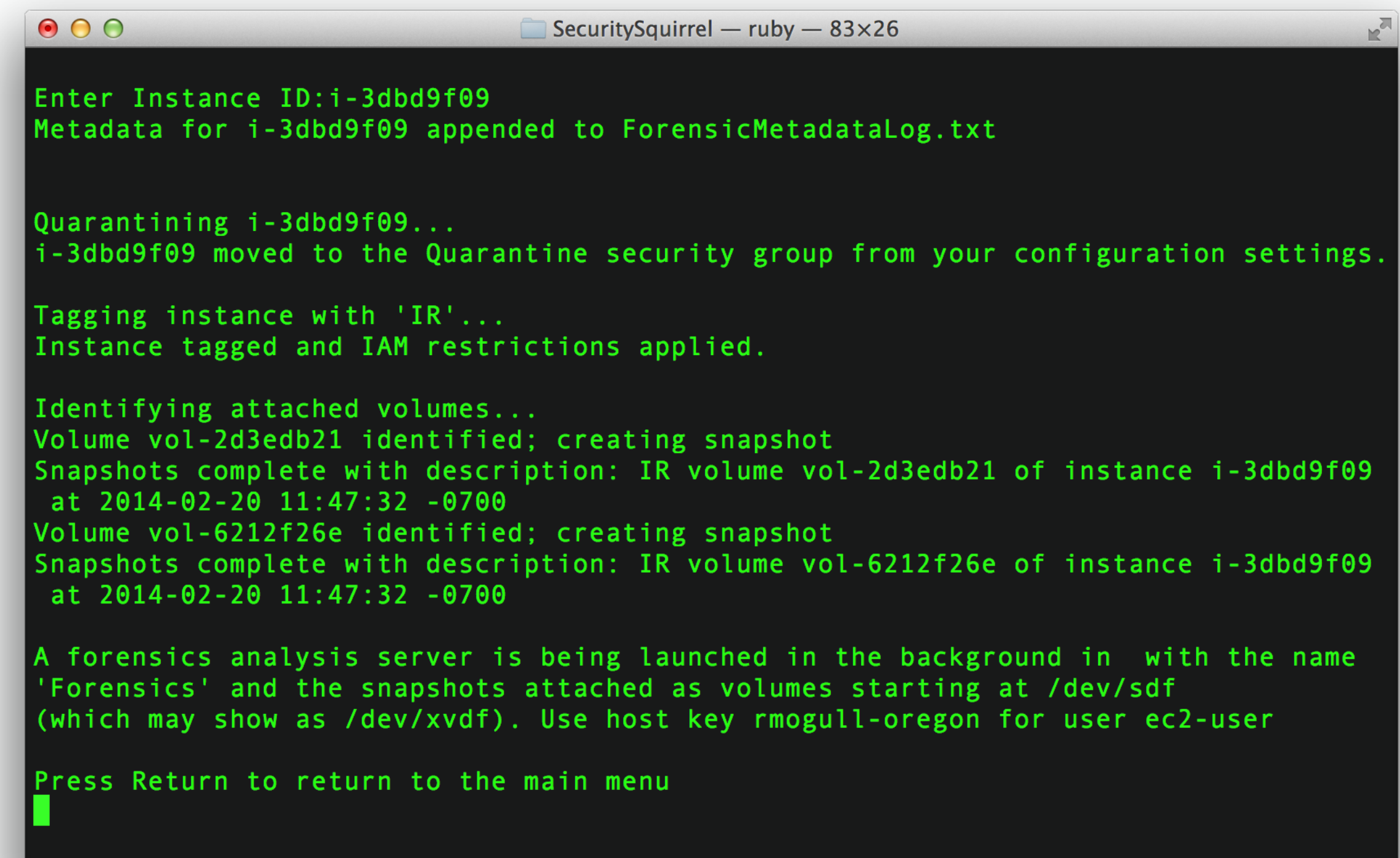
Building a Workflow



Can be built on Guardrails and support Orchestrations

Our Workflow

- Steps (Incident Response)
 - Collect metadata (before we change it)
 - Quarantine on the network and in AWS
 - Snapshot all storage and attach for forensics
 - Analyze
- Inputs
 - Instance ID
- Execution Model
 - Command line (container or remote)
- Modularize Code
 - Classes for analyze vs. respond
 - All methods reusable



```
SecuritySquirrel — ruby — 83x26

Enter Instance ID:i-3dbd9f09
Metadata for i-3dbd9f09 appended to ForensicMetadataLog.txt

Quarantining i-3dbd9f09...
i-3dbd9f09 moved to the Quarantine security group from your configuration settings.

Tagging instance with 'IR'...
Instance tagged and IAM restrictions applied.

Identifying attached volumes...
Volume vol-2d3edb21 identified; creating snapshot
Snapshots complete with description: IR volume vol-2d3edb21 of instance i-3dbd9f09
at 2014-02-20 11:47:32 -0700
Volume vol-6212f26e identified; creating snapshot
Snapshots complete with description: IR volume vol-6212f26e of instance i-3dbd9f09
at 2014-02-20 11:47:32 -0700

A forensics analysis server is being launched in the background in with the name
'Forensics' and the snapshots attached as volumes starting at /dev/sdf
(which may show as /dev/xvdf). Use host key rmogull-oregon for user ec2-user

Press Return to return to the main menu
█
```

Demo

Lab: Run the Incident Response Workflow

24

- This is pre-loaded in Admin
 - Launch an instance you can quarantine in your **default VPC**
 - If you want to use your SecOps VPC you will need to update the code
 - Create a new security group named “quarantine” without any permissions *in the same VPC as your target instance*
 - Log in and `cd ir`
 - `nano config.json`
 - Modify settings for us-west-2 as indicated then save
 - Change the security groups
 - User your SSH key name
 - Update the AMI to ami-082b5a644766e0e6f
 - `ruby ir.rb`

ir.rb Current Limitations

- ▶ This is older code we haven't fully updated as better-supported tools are emerging
 - ▶ <https://threatresponse.cloud>
- ▶ Everything has to be in the same VPC (target + security groups)
- ▶ Requires hard-coding of various IDs
 - ▶ These days we code automations to look for required resources, like security groups, then create them if they don't exist
- ▶ There is a bunch of in-development code in there that isn't fully functional yet

Lab: Add code to stop the instance

- ▶ <https://docs.aws.amazon.com/sdkforruby/api/index.html>

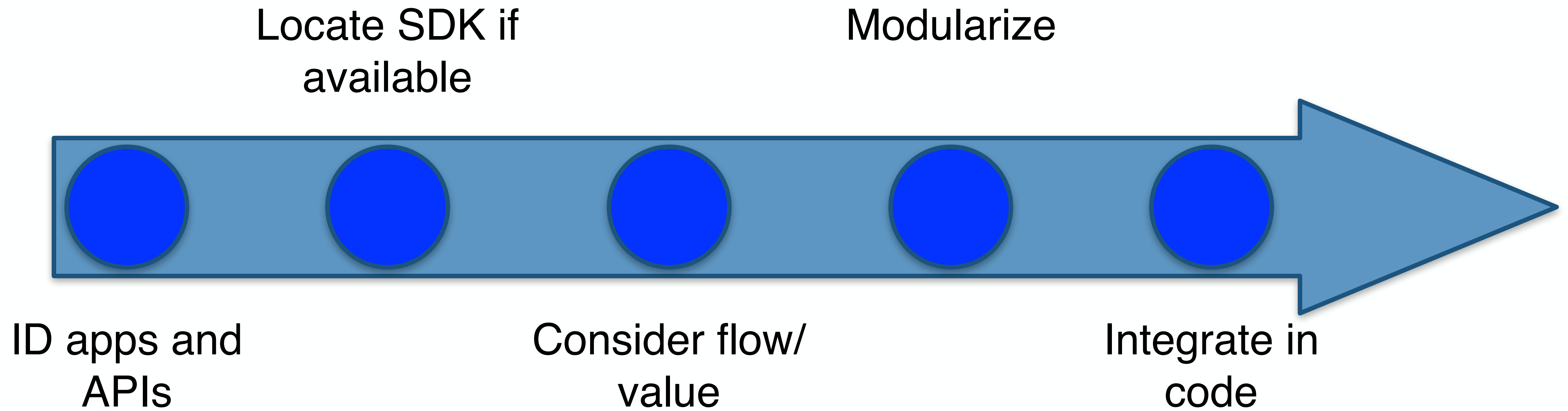


Workflows Advice

- Workflows are to speed up common, manual tasks
 - Guardrails are for automated enforcement
 - The line between a guardrail action and an Workflows is often thin
- Execution environment matters
 - Lambda vs. containers vs. your laptop
- Use your pipeline
 - Continuous integration servers (Jenkins) make great platforms for repeat automation, not just security testing
- Make a static console
 - E.g. S3 + API Gateway + SQS

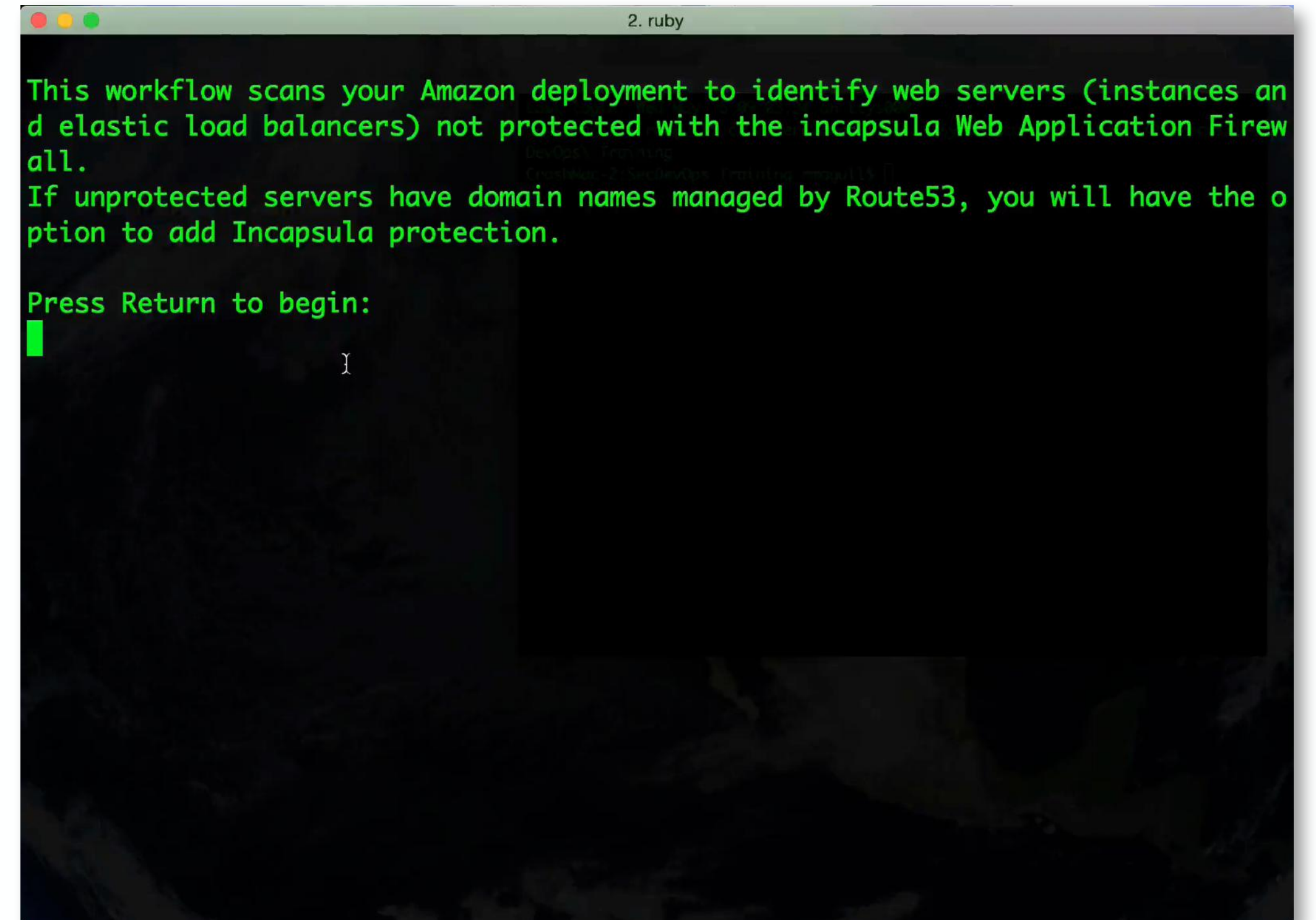


Building an Orchestration



Our Orchestration Demo

- Apps/API
 - EC2 + Route 53 + Incapsula
- SDK
 - AWS Ruby + REST client
- Flow/Value
 - ID public web servers -> determine DNS -> check WAF -> add WAF
 - Limit: default AWS domain names
- Modularize
 - Find web instances, ELBs
 - Change DNS, add Incapsula
- Integrate into code
 - See video



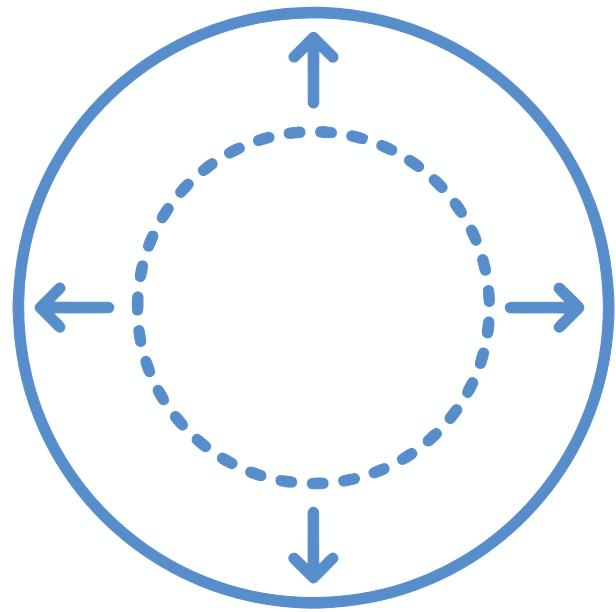
```
2. ruby
This workflow scans your Amazon deployment to identify web servers (instances and elastic load balancers) not protected with the incapsula Web Application Firewall.
If unprotected servers have domain names managed by Route53, you will have the option to add Incapsula protection.

Press Return to begin:
█
```

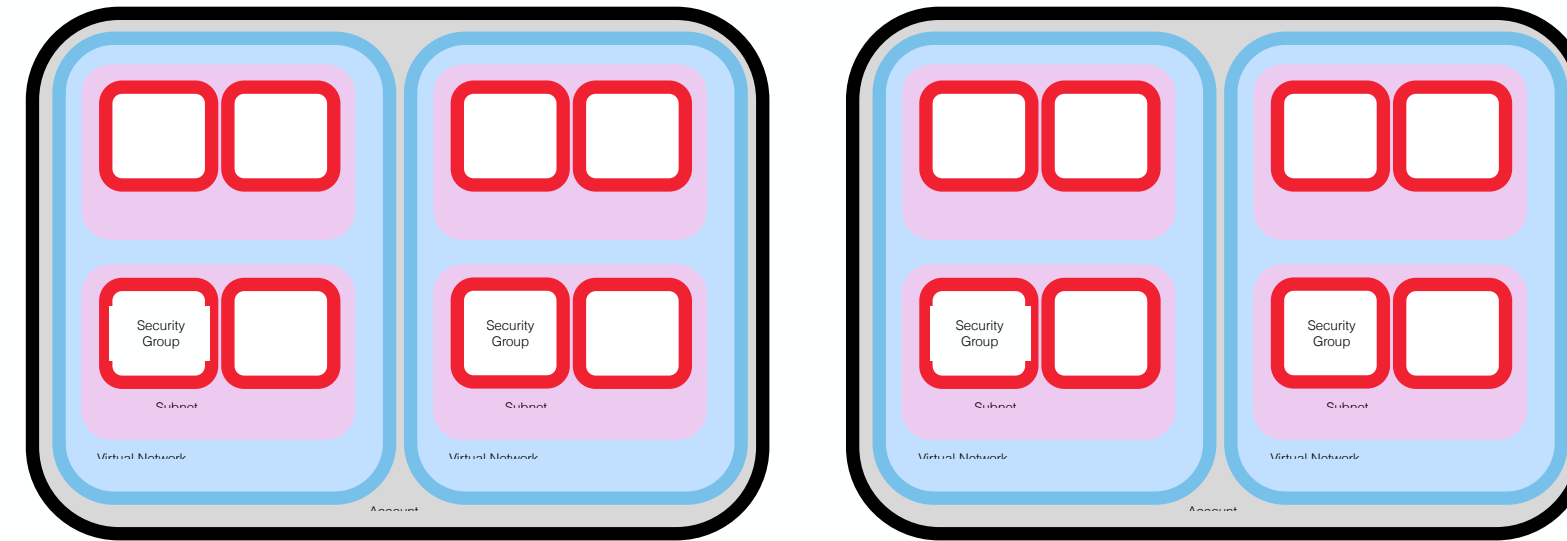
Demo

Your Student Share directory includes multiple sample lambdas for you to experiment with and modify if you have the time

Complexities



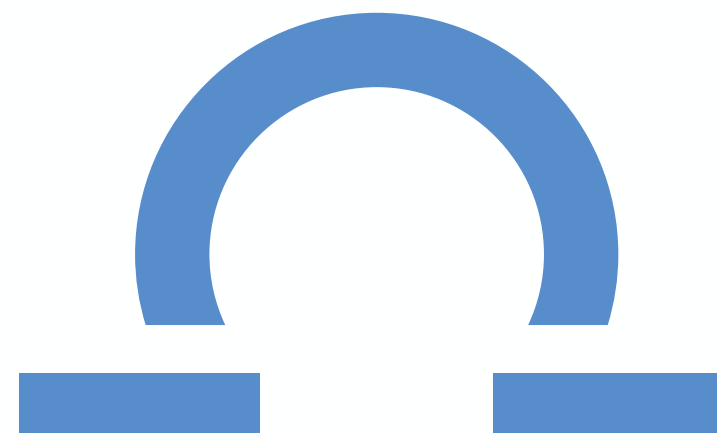
Scaling



Multiple Accounts

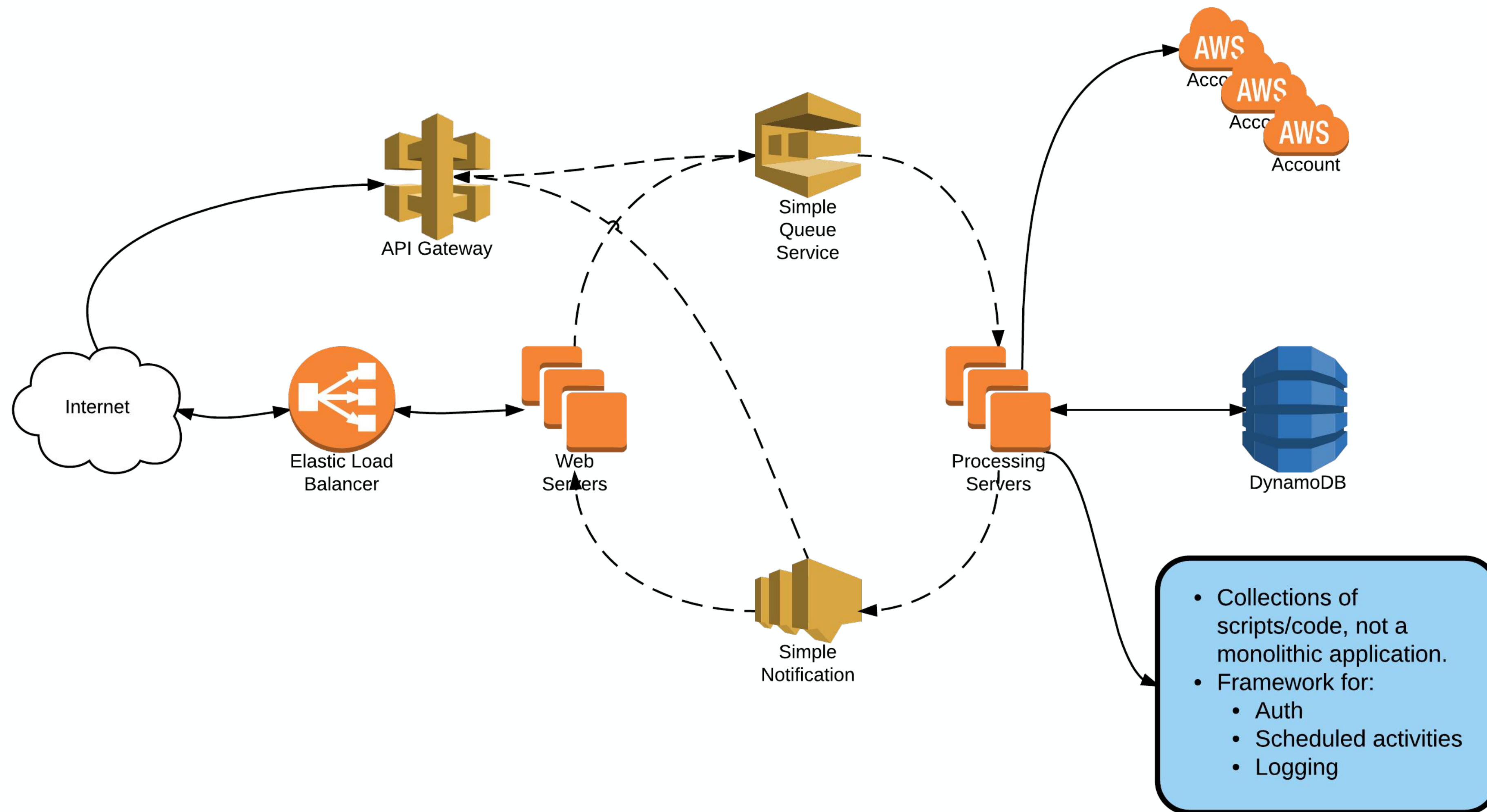


Multiple Providers



Circuit Breakers

Architecting For Enterprise Scale



Where to Start

- Start with something simple
 - Build it in one account/subscription/project
 - Event + Notification is super easy to start
 - Then go with your first FaaS
 - Desktop first, then FaaS for execution environment
- Build a library
 - Experiment with execution environments, but standardize quickly
- Add enterprise scaling capabilities
 - Will depend on your execution environment/model
 - Build it in the cloud and leverage PaaS options
- Make sure you use CI/CD for long term management

Incident Response

Key Incident Response Issues

- ▶ Real world cloud IR is both better and worse than traditional infrastructure:
 - ▶ You still need to manage compromised resources (e.g. instances).
 - ▶ You also need to add the cloud management plane to the scope.
 - ▶ The cloud provider and you will have different priorities.
 - ▶ You may have more or less control, depending on your governance and SaaS vs. IaaS.
 - ▶ E.g. you can totally manage the infrastructure remotely with automation, which is an advantage. But in SaaS you might not control much of anything.
 - ▶ You have to rely less on network packet capture.
 - ▶ Immutable infrastructure is a powerful recovery option.
 - ▶ Containment can be much easier.



Key Principles

- ▶ Know who to call
- ▶ Train on your providers of choice
- ▶ Write your response procedures and automation code ahead of time
- ▶ Don't rely on manual response
- ▶ Use immutable for recovery as often as possible
- ▶ **Kill IAM/metastructure access first**
 - ▶ Don't forget that on both the network and with IAM/management plane you may need to **kill active sessions**, not merely revoke access

Background: How to Image an Instance

- ▶ Get the instance ID from the EC2 console
- ▶ Click on volumes and filter on the instance ID
- ▶ Snapshot the volume(s) and record the snapshot ID
- ▶ Create a new volume based on the snapshot
 - ▶ When you create a new volume you can base it on the snapshot ID
- ▶ Attach the new volume to a running instance (and remember the device mapping)
- ▶ Log into the running instance and start your forensics



Lab: Incident Response

- ▶ This is the capstone lab for this training, leveraging multiple skills.
- ▶ You will launch a CloudFormation template to set everything up and launch an attack simulator in 2 accounts
 - ▶ That instance will simulate a cloud-native attack on your accounts
 - ▶ The activities are all constrained, but represent techniques a real attacker would use
 - ▶ It is also designed to be easy to clean up and allow you to perform a response in the allotted time.
- ▶ You must follow all the normal steps in an IR process.

IR Lab Prep

In the SharedServices account

- ▶ Do not modify the current account security
 - ▶ However, this is where you will deploy any analysis tools to complement the tools already installed
 - ▶ Consider using those tools to assess and harden the WebappProduction account

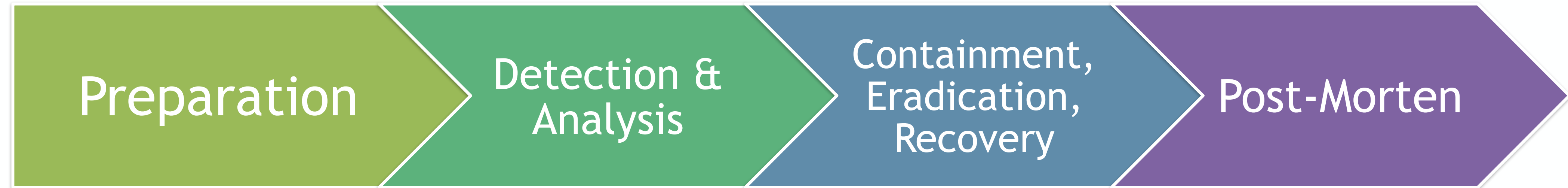
Use the WebappProduction account

- ▶ Your instructor will give you a time window to harden the account
- ▶ Your objective is to take everything you have learned to prepare the account for the upcoming attack

IR Lab Prep Part 2

- ▶ Consider writing an SCP for the Incident Response OU
 - ▶ What would you put into an SCP that would help in an incident?
 - ▶ Would those changes break the application and is this acceptable?
 - ▶ How can you use the SCPs to contain the attack without destroying needed forensics?
- ▶ Then, when your instructor tells you
 - ▶ Follow the instructions on the next page to start the simulation
 - ▶ Run the CloudFormation template in both SharedServices **and** WebappProduction
- ▶ Using both accounts will help you better understand the role of your defenses

Lab: Incident Response



- ▶ Launch the CloudFormation template on your cheat sheet:
 - ▶ us-west-2 as usual
 - ▶ Wait 5-ish minutes for it to settle
- ▶ **You must!**
 - ▶ Follow the IR steps above
 - ▶ Contain the attack
 - ▶ Determine what happened
- ▶ We will provide full cleanup instructions separately

IR Lab Constraints and Reality

- ▶ This attack simulation is deliberately constrained:
 - ▶ It relies on provided admin credentials and skips the hard part of exploitation.
 - ▶ It uses all pre-determined resources to ensure we can clean it up.
 - ▶ It purposely doesn't attack certain resources that could either violate terms of service or damage your account.
 - ▶ It is designed to fit within our classroom time constraints.
- ▶ However:
 - ▶ It does demonstrate multiple real-world techniques used by cloud native attackers.
 - ▶ It forces you to think in cloud-native response terms.

IR Discussion

- ▶ How could an attacker compromise credentials to carry out this attack?
- ▶ How could they escalate privileges if they only gain access to lower-level credentials?
- ▶ What inherent tools and techniques would prevent the various attacks demonstrated in this lab?
- ▶ How could you use automation? Do you think it's required?

Whiteboard

What We Covered

- ▶ Baseline security, from the account architecture and root account through IAM, monitoring, and network security
- ▶ Real-world network architectures and security
- ▶ Leveraging DevOps techniques and deployment pipelines for security
- ▶ A primer on leveraging cloud-native options for building secure application architectures
- ▶ Security automation
- ▶ Incident response for cloud

