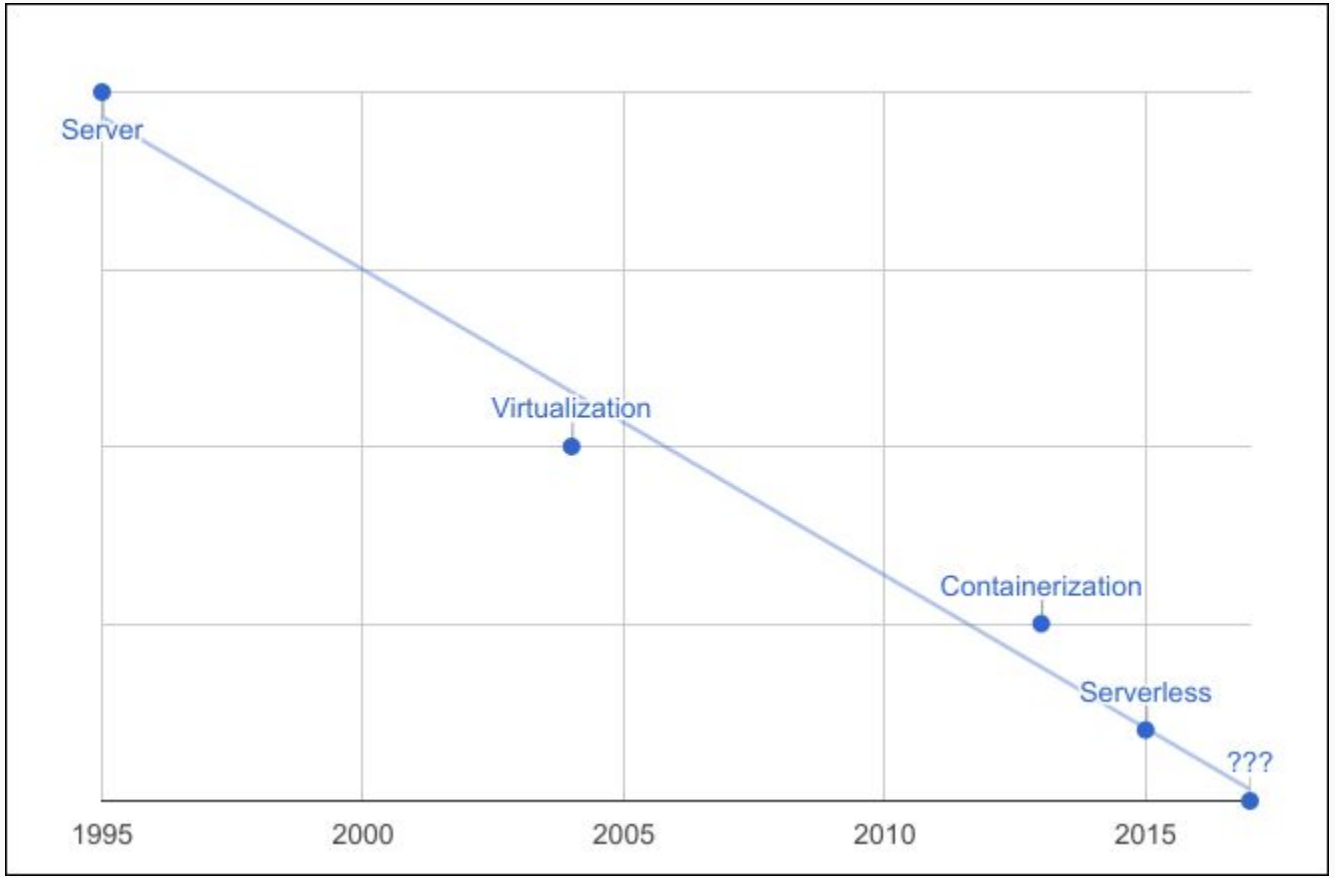
Serverless and security

with Bo Bayles (Observable Networks)

The arc of history

Simplification helps with security:

- Real servers (since forever)
- Virtual machines (since the mid 2000s)
- Containers (since about 2013)
- Serverless (since about 2014)



Extrapolating past serverless computing...

Functionless computing

Taking the trend to its logical conclusion…

- The next generation should do literally nothing
- It will be maximally secure

VCs, call me.

Rarely asked questions about Lambda security

How can we **audit** and **monitor** each of these areas?

- Function access - who is allowed to modify or invoke the Lambda?
- Resource permissions - what is the Lambda is allowed to interact with?
- Resource abuse - can malicious actors waste resources with the function?
- Container security - can code escape its sandbox?
- Code injection - can the function be made to do something unexpected?

Monitoring by category

Function access	CloudTrail, Lambda API, IAM API
Resource permissions	IAM API, CloudTrail
Resource abuse	CloudWatch, ?
Container security	?
Code injection	?

Auditing function access

Policies determine who/what can invoke the function:

- The Lambda API (*GetPolicy*) shows what can invoke the function (e.g. S3, SNS)
- The IAM API could be used to find Roles that have Lambda access (ugh)

SourceArn

This is optional; however, when granting Amazon S3 permission to invoke your function, you should specify this field with the Amazon Resource Name (ARN) as its value. This ensures that only events generated from the specified source can invoke the function.

Important

If you add a permission for the Amazon S3 principal without providing the source ARN, any AWS account that creates a mapping to your function ARN can send events to invoke your Lambda function from Amazon S3.

Why should we audit this? Oh, that's why...

TOP DEFINITION

footgun

A podiatric penetration purposed pistol. A gun which is apparently designed for shooting yourself in the foot.

My favorite footgun: ***C++***
Your favorite footgun: icon fonts
Everyone's favorite footgun: ***representative democracy***

#pain #suffering #failure #gun #foot #shoot

by brb, doll January 16, 2014

Source: http://www.urbandictionary.com/define.php?term=footgun

Monitoring function access

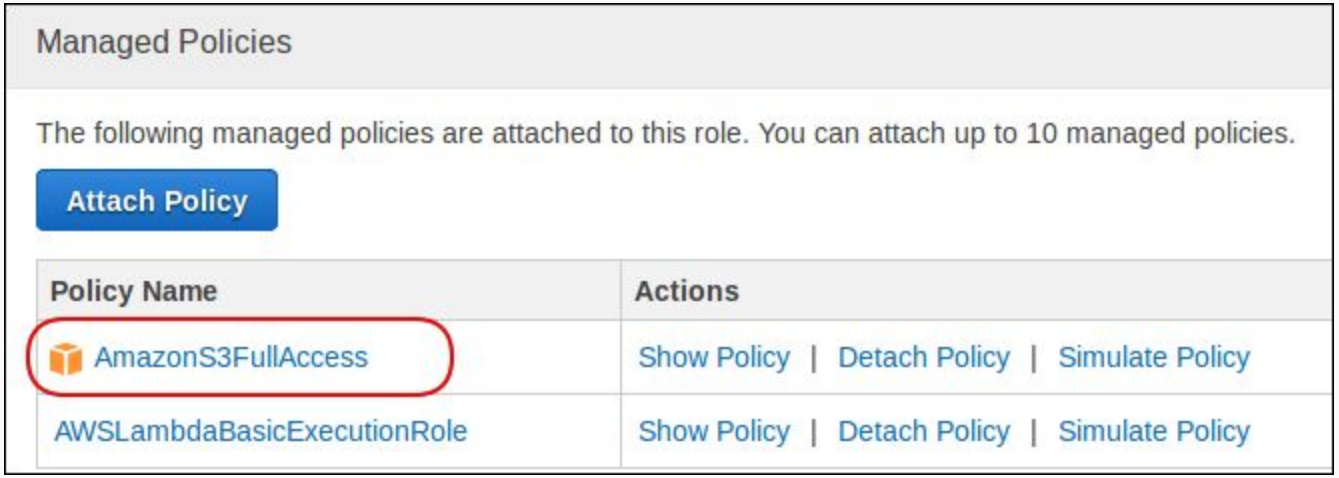
CloudTrail logs “write” events:

- UpdateFunctionCode
- UpdateFunctionConfiguration
- AddPermission
- ...and others

Auditing resource permissions

The IAM API can read the policies associated with the Lambda function's execution role (e.g. *logs:CreateLogGroup* for logging).

This determines what the Lambda *can* do.

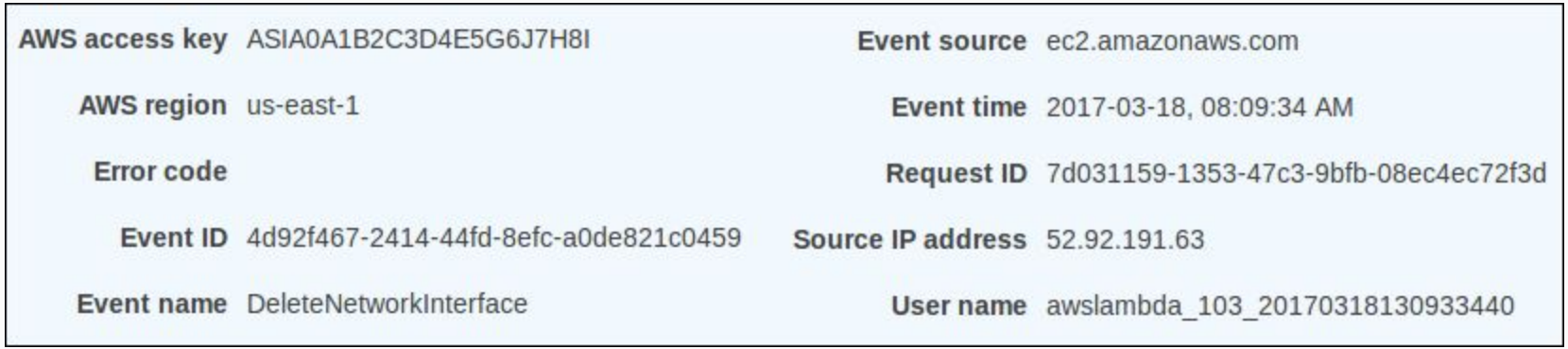


I don’t see what could possibly go wrong here...

Monitoring resource permissions

CloudTrail makes an entry with an *awslambda* user when the function makes some API calls.

This shows (some of) what the Lambda function *actually did*.



AWS access key	ASIA0A1B2C3D4E5G6J7H8I	**Event source**	ec2.amazonaws.com
AWS region	us-east-1	**Event time**	2017-03-18, 08:09:34 AM
Error code		**Request ID**	7d031159-1353-47c3-9bfb-08ec4ec72f3d
Event ID	4d92f467-2414-44fd-8efc-a0de821c0459	**Source IP address**	52.92.191.63
Event name	DeleteNetworkInterface	**User name**	awslambda_103_20170318130933440

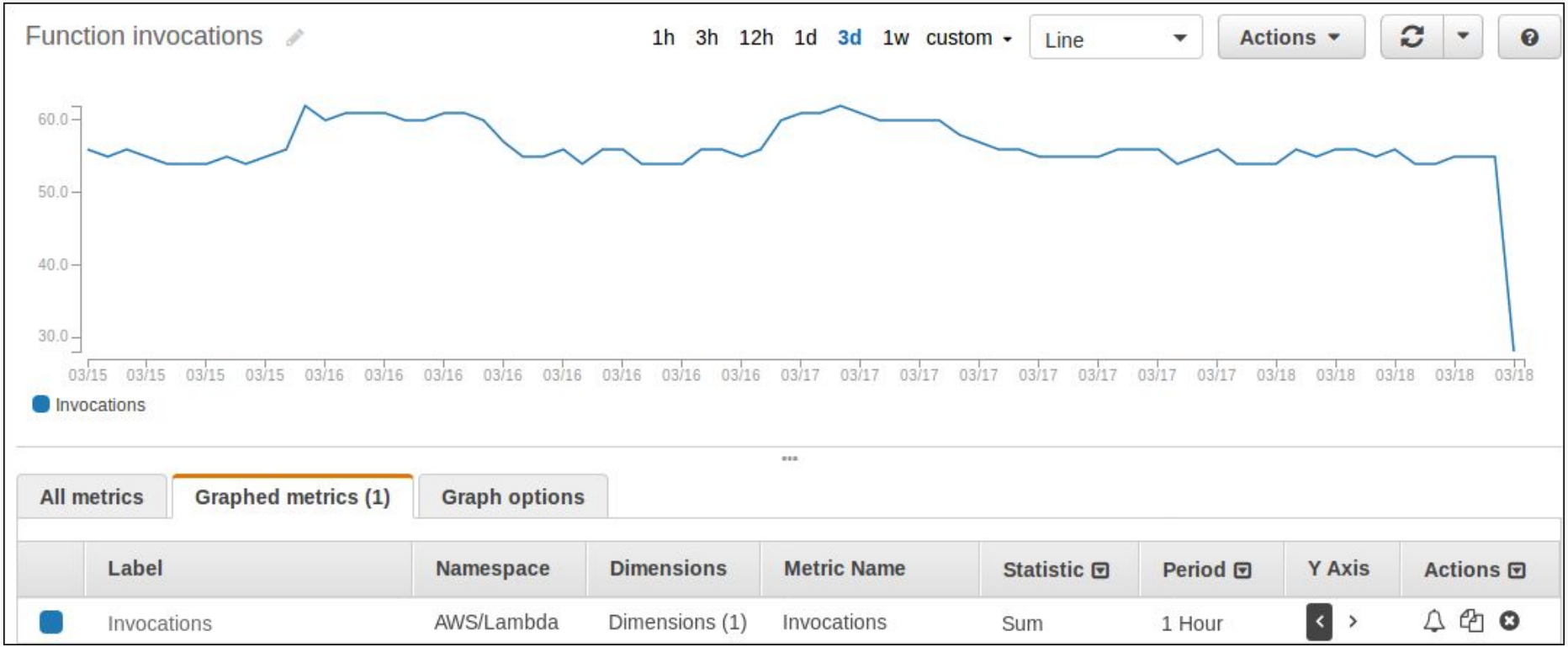
CloudTrail entry for Lambda-initiated API call

Resource abuse

Can the Lambda function be invoked from the outside? If so, would you know if its activity increased 10x?

- CloudWatch Logs will have a record of invocations for billing
- CloudWatch Metrics can display and alarm when things go haywire

What about other potential abuses? Can we know if the Lambda reads too much from RDS? ElastiCache? Redshift?



CloudWatch Metrics can track, e.g., how often a function is invoked and set alarms

Lambda access to VPC resources

How does a Lambda function access resources in a VPC, anyway?

- Through the EC2 API, obviously
- Functions create temporary network interfaces (ENIs), get assigned IP addresses, and can then communicate with the associated subnet
- Then they can do…. lots of things.

How can this activity be tracked?

Caution

If your VPC does not have sufficient ENIs or subnet IPs, your Lambda function will not scale as requests increase, and you will see an increase in function failures. AWS Lambda currently does not log errors to CloudWatch Logs that are caused by insufficient ENIs or IP addresses. If you see an increase in errors without corresponding CloudWatch Logs, you can invoke the Lambda function synchronously to get the error responses (for example, test your Lambda function in the AWS Lambda console because the console invokes your Lambda function synchronously and displays errors).

If there aren't enough IP addresses or ENIs, good luck figuring out why your function is failing

Finding Lambda IP addresses

- The EC2 *DescribeNetworkInterfaces* call lists ENIs, including temporary Lambda ones
- A single Lambda may have several ENIs
- The ENI identifier and IP address can be matched with VPC Flow Logs

						Bytes		Packets	
Time	IP	Connected IP	Port	Connected Port	Protocol	To	From	To	From
3/18/17 2:12 AM	10.0.11.233	10.0.10.193	34548	3306 (mysql)	TCP	52,511	738	38	11
3/18/17 2:02 AM	10.0.11.233	10.0.10.193	34548	3306 (mysql)	TCP	663,461	6,970	462	106
3/18/17 1:52 AM	10.0.11.233	10.0.10.193	34548	3306 (mysql)	TCP	893,886	7,844	617	121
3/18/17 1:42 AM	10.0.11.233	10.0.10.193	34548	3306 (mysql)	TCP	733,965	7,376	504	112
3/18/17 1:32 AM	10.0.11.233	10.0.10.193	34548	3306 (mysql)	TCP	654,609	6,880	466	102

Example entries from VPC Flow logs - Lambda function accessing RDS database

Monitoring by category, revisited

Function access	CloudTrail, Lambda API, IAM API
Resource permissions	IAM API, CloudTrail
Resource abuse	CloudWatch, **VPC Flow Logs**
Container security	**VPC Flow Logs**
Code injection	**VPC Flow Logs**

Resource abuse, revisited

- VPC Flow logs allow for close monitoring of activity - when the function accesses something in the VPC the flow is recorded
- Lambda functions access the Internet via NAT instances / gateways in the VPC, so their external traffic is recorded also
- Application can track expected connections and data amounts, send notifications when there are deviations
- Useful for unexpected behavior of the function (coding mistake), or abuse (function can be invoked by malicious user)

Code injection, container security breaches

- Injection attacks that use the function’s existing code in an unexpected way can be tracked if they generate new VPC traffic.
- Container escape is so far a speculative issue for AWS, but has affected other providers (e.g. VENOM vulnerability).
- Behavior profiling can show a break from established traffic patterns and produce alerts.

Historical Outlier Observation

One of the source's metrics deviated significantly from its historical baseline.

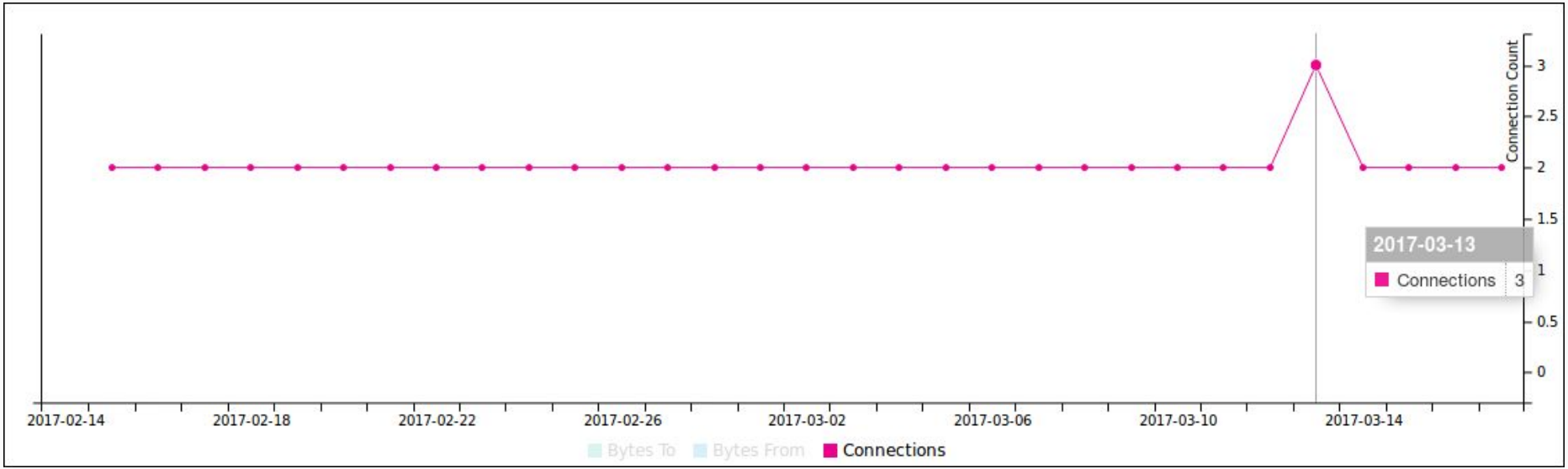
Time	Source	Time Window	Type	Metric	Expected Value	Outlier	Probability	Sample Size	
3/13/17 12:00 AM	lambda:RDSQueryLogger	1d	device	Bytes Out	12,097,460.313	142,117,719	0.37%	42	
3/13/17 12:00 AM	lambda:RDSQueryLogger	1d	device	Internal Bytes Out	12,097,460.313	142,117,719	0.37%	42	

Static Connection Set Deviation Observation

Device normally talks to a static set of (internal/external) devices, but has recently started/stopped talking to new/normal devices.

			Normal Connections		New Connections		Lost Connections			
Time	Source	Type	Set	Count	Set	Count	Set	Count	History Length (Days)	
3/13/17 12:00 AM	lambda:RDSQueryLogger	internal	10.0.10.193 , 10.0.12.134	2	10.0.255.29	1	-	0	35	

Observable Networks alert for Lambda function changing its connectivity pattern

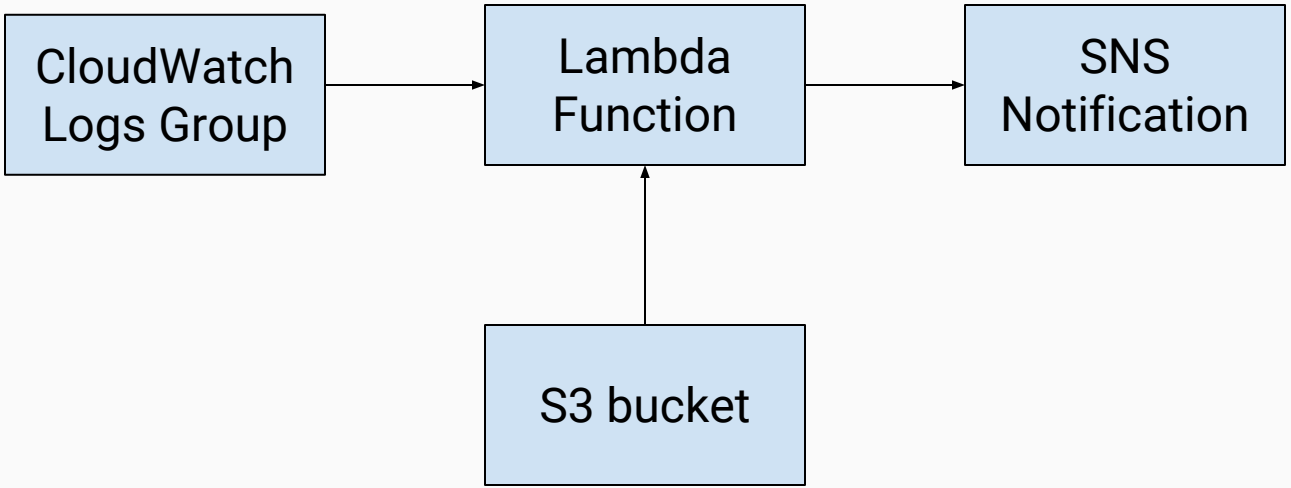


Spotting a change in where the Lambda function connects in the VPC

Serverless monitoring pipeline

With VPC Flow Logs and Lambda, we have a way to monitor everything in our VPC without any servers

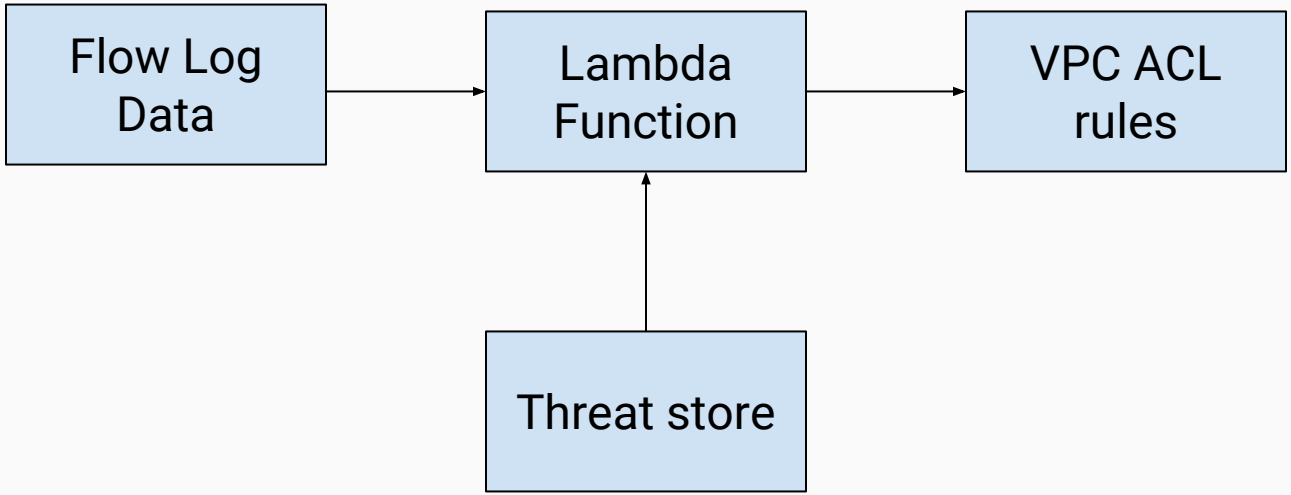
- CloudWatch logs stores VPC FLow Logs
- S3 (or some other persistent store) can provide threat intelligence or behavior specifications
- Lambda function can be triggered by CWL, read from S3, and send notifications to SNS


CloudWatch Logs Group
Lambda Function
SNS Notification
S3 bucket

Extending to the pipeline

This basic framework can be extended in various ways

- Add EC2 authentication logs from CWL to match VPC traffic to remote user activity
- Read API usage from CloudTrail to record behaviors that don’t generate VPC traffic
- Put Kinesis between CWL and Lambda for high-volume VPCs
- Automatically take action for certain conditions (e.g. block or shun a malicious IP address)



Automatically block a malicious user using VPC ACLs, sketch

Rule #	Type	Protocol	Port Range	Source	Allow / Deny
99	ALL Traffic	ALL	ALL	198.51.100.1/32	DENY
100	ALL Traffic	ALL	ALL	0.0.0.0/0	ALLOW
101	ALL Traffic	ALL	ALL	::/0	ALLOW
*	ALL Traffic	ALL	ALL	::/0	DENY
*	ALL Traffic	ALL	ALL	0.0.0.0/0	DENY

Automatically block a malicious user using VPC ACLs, result

Plug for Observable Networks

- Observable has Lambda function monitoring built in.
- (And also other stuff)
- Get alerts for security-relevant behavior changes without all the wolf-crying
- Set up a trial at observable.net - setup for AWS takes a few minutes

The end

Bo Bayles:
bbayles@obsrvbl.com

Observable Networks:
https://observable.net

Open-source AWS tools and more:
https://github.com/obsrvbl