AWS Professional Services

Network Firewall Alerts – Slack Integration

Prepared for: 22-June-2021

Security AOD 10 Version 0.2 Draft

Prepared by:

Venki Srivatsav, Professional Services Consultant
vramaam@amazon.com



Amazon Web Services makes no warranties, express or implied, in this document.

Amazon Web Services (AWS) may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from AWS, our provision of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

The descriptions of other companies’ products in this document, if any, are provided only as a convenience to you. Any such references should not be considered an endorsement or support by AWS. AWS cannot guarantee their accuracy, and the products may change over time. Also, the descriptions are intended as brief highlights to aid understanding, rather than as thorough coverage. For authoritative descriptions of these products, please consult their respective manufacturers.

Copyright © 2021 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

The following are trademarks of Amazon Web Services, Inc.: Amazon, Amazon Web Services Design, AWS, Amazon CloudFront, Cloudfront, Amazon DevPay, DynamoDB, ElastiCache, Amazon EC2, Amazon Elastic Compute Cloud, Amazon Glacier, Kindle, Kindle Fire, AWS Marketplace Design, Mechanical Turk, Amazon Redshift, Amazon Route 53, Amazon S3, Amazon VPC. In addition, Amazon.com graphics, logos, page headers, button icons, scripts, and service names are trademarks, or trade dress of Amazon in the U.S. and/or other countries. Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon.

All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

# Revision and Signoff Sheet

## Change Record

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Author | Version | Change Reference |
| June/03/2021 | Venki Srivatsav | .1 | First Draft |
| June/22/2021 | Venki Srivatsav | .2 | Added Filters |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Reviewers

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Version Approved | Position | Date |
| Solution Architect |  |  |  |
| Account Rep |  |  |  |
| Practice Manager |  |  |  |

# Table of Contents

[Revision and Signoff Sheet 2](#_Toc75282087)

[Change Record 2](#_Toc75282088)

[Reviewers 2](#_Toc75282089)

[Table of Contents 3](#_Toc75282090)

[Summary 4](#_Toc75282091)

[Key Elements and Challenges 4](#_Toc75282092)

[Elements discussed 4](#_Toc75282093)

[Challenges 4](#_Toc75282094)

[Solution 4](#_Toc75282095)

[Prerequisites: 4](#_Toc75282096)

[Architecture Overview 5](#_Toc75282097)

[What is Included in the solution? 6](#_Toc75282098)

[What is not Included in the solution? 6](#_Toc75282099)

[How does this solution work? 6](#_Toc75282100)

[Components of this Solution: 6](#_Toc75282101)

[How to Deploy: 7](#_Toc75282102)

[Filter Behavior: 8](#_Toc75282103)

[Appendix 9](#_Toc75282104)

# Summary

The Objective of this document is to deploy an AWS Network firewall using distributed deployment model and propagate the alerts generated by firewall manger to a configurable Slack channel.

# Key Elements and Challenges

## Elements discussed

* Integrating Network firewall generated alerts to Slack for further action.
* Network firewall Distributed deployment model

## Challenges

VPCs are considered same as a physical network when it comes to compliance regimes such as PCI-DSS. Workloads runs on VPCs that are governed by a compliance regime can be protected using AWS Network Firewall, any unauthorized access from the other VPCs of the same account can be blocked or alerted. AWS Network firewall support limited number of destinations for delivering the generated alerts. However, these destinations act as a log destination, any further action on these alerts requires a post log offline analysis either using Athena or Kinesis. This solution provides a method where NFW generated alert can be propagated to other systems for further action in a near real time basis. This solution also provides a platform to extend the functionality to add other alerting mechanism such as Pager duty / Jira etc.,

## Solution

Network firewall can be used monitor and control network traffic between VPCs, ingress and egress flow. Network Firewall manager generated alerts can be delivered to S3 or Cloud watch log group or Kinesis data firehose. There are no out of the box options to deliver the alerts other than the above three destinations. This solution provides an option to deliver the alerts to a configurable Slack channel.

The solution can also be used as a template or a platform to extend the alert delivery to other platforms such as JIRA, text or email etc.,

### Prerequisites:

This solution assumes that user already

1. Has a slack channel
2. Has the required privileges to send a slack message
3. In possession of the slack endpoint url with the token.

[Refer this link for creating new slack workspace](https://slack.com/help/articles/206845317-Create-a-Slack-workspace)

### Architecture Overview

Decentralized Network Firewall deployment with Slack Integration



For details on the deployment model options please refer [here](https://aws.amazon.com/blogs/networking-and-content-delivery/deployment-models-for-aws-network-firewall/)

## ****Target technology stack****

1. Vpc
2. Subnets
3. Internet gateway
4. Route tables with rules
5. S3 bucket for Network firewall alert destination
6. S3 – Event configuration to Trigger Lambda
7. S3 – Bucket policy
8. Lambda Execution role
9. Lambda function to send Slack notifications
10. Secret manager Secret for storing Slack url
11. Network Firewall with alert configuration
12. Slack channel

### What is Included in the solution?

All the resources that fall under the “Region” box in the architecture diagram are included in this solution. These resources will be automatically provisioned  through the CloudFormation templates and  source code required for the Lambda also included in the bundle.

At the end of the CloudFormation execution the following AWS resources will be provisioned.

1. Vpc
2. Subnets – 4 ( 2 subnets dedicated for FW and 2 for workloads)
3. Internet gateway
4. Route tables with rules – 4
5. S3 bucket for Network firewall alert destination
6. S3 – Event configuration to Trigger Lambda
7. S3 – Bucket policy
8. Lambda Execution role
9. Lambda function to send Slack notifications
10. Secret manager Secret for storing Slack url
11. Network Firewall with alert configuration

### What is not Included in the solution?

1. Creating Slack channel
2. Test EC2 instance in the workload subnets
3. Test rules in Network Firewall
4. Actual or simulated traffic to trigger the test rules.
5. S3 bucket to hold the source files to be deployed.

### How does this solution work?

This solution provides two protected subnets (Protected Subnet), two Network Firewall endpoint will be provisioned on the  dedicated subnets (Firewall Subnet), all traffic going in and out of the protected subnets can be monitored by [creating FW policies](https://docs.aws.amazon.com/waf/latest/developerguide/network-firewall-policies.html) and rules. The Network firewall is configured to place all alerts to a S3 bucket. This S3 bucket is configured to invoke a Lambda function upon any ‘put’ event in the bucket. This Lambda fetches the configured slack url from Secret Manager and sends the notification message when invoked from S3.

### Components of this Solution:

1. A set of CloudFormation files in Yaml format.
2. Python source file for Lambda function.
3. Compressed Python source file.

### How to Deploy:

1. Create new or use an existing s3 bucket
2. Copy the following files to the above bucket (prefix recommended) from [source location](https://gitlab.aws.dev/vramaam/security-aod10-vramaam.git)
	1. base.yml
	2. igw-ingress-route.yml
	3. slack-lambda.py
	4. slackLambda.yml
	5. decentralized-deployment.yml
	6. protected-subnet-route.yml
	7. slack-lambda.py.zip
3. Using console create a stack by choosing base.yml.

#### Deployment Parameters:

|  |  |  |
| --- | --- | --- |
| Parameter Name | Description | Sample value |
| pAWSSecretName4Slack | Name of the secret that holds slack url | SlackEnpoint-Cfn |
| pAlertS3Bucket | Name of the s3 buckets to be created, this S3 bucket will be used as the Network firewall alert destination | usw2-05-some\_name-nfw-alerts |
| pAvailabilityZone1 | Pick an aws AZ  | us-west-2a  |
| pAvailabilityZone2 | Pick another aws AZ | us-west-2b |
| pNetworkFirewallSubnet1Cidr | Choose a Subnet CIDR for FW Subnet (minimum /28) | 10.0.1.0/24 |
| pNetworkFirewallSubnet2Cidr | Choose a Subnet CIDR for FW Subnet (minimum /28) | 10.0.2.0/24 |
| pProtectedSubnet1Cidr | Choose a Subnet CIDR for workload ( Protected Subnet) | 10.0.3.0/24 |
| pProtectedSubnet2Cidr | Choose a Subnet CIDR for workload ( Protected Subnet) | 10.0.4.0/24 |
| pS3BucketName | Name of the existing bucket where the source code is located | Us-w2-yourname-lambda-functions |
| pS3KeyPrefix | Name of the bucket prefix where the source code is located |  folder name  |
| pSecretKey | Can be any key, use the default - recommended | webhookUrl |
| pSecretTagName | Tag name for the Secret | AppName |
| pSecretTagValue | Tag value for the above tag name | LambdaSlackIntegration |
| pSlackChannelName | Pre created Slack channel name | nfw-alert-notifications ( Name of your slack channel) |
| pSlackUserName | Slack User name  | userName |
| pVpcCidr | A CIDR range for VPC to be creted | 10.0.0.0/16 |
| pVpcInstanceTenancy | default | default |
| pVpcName | Name for the VPC to be created | Inspection |
| pWebHookUrl | Value of the Slack url | https://hooks.slack.com/services/T???9T??/A031885JRM7/9D4Y?????? |
| plambdaSrcS3  | Name of the existing bucket where the lambda code is located | Us-w2-yourname-lambda-functions ( Same as  pS3BucketName Value) |
| pdestCidr | Filter for Destination CIDR Range  | 10.0.0.0/16 |
| pdestCondition | Flag to indicate to exclude or include the Destination match | valid valid values are "include" or "exclude" |
| psrcCidr | Filter for Source CIDR Range  | 118.2.0.0/16 |
| psrcCondition | Flag to indicate to exclude or include the Source match | valid valid values are "include" or "exclude" |

### Filter Behavior:

If there are no filters configured in lambda then all the generated alerts will be sent to Slack.

Source and destination ips of the generated alerts will be matched against the configured cidrs.

If a match found then the condition will be applied. An alert will be generated either source or destination falls in filter criteria.

Alert will be generated with just one row matches the criteria

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Configured CIDR | Alert IP | Configured Condition | Alert |
| Source | 10.0.0.0/16 | 10.0.0.25 | include | Yes |
| Destination | 100.0.0/16 | 202.0.0.13 | include |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Configured CIDR | Alert IP | Configured Condition | Alert |
| Source | 10.0.0.0/16 | 10.0.0.25 | exclude | No |
| Destination | 100.0.0/16 | 202.0.0.13 | include |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Configured CIDR | Alert IP | Configured Condition | Alert |
| Source | 10.0.0.0/16 | 10.0.0.25 | include | Yes |
| Destination | 100.0.0.0/16 | 100.0.0.13 | include |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Configured CIDR | Alert IP | Configured Condition | Alert |
| Source | 10.0.0.0/16 | 90.0.0.25 | include | Yes |
| Destination | Null | 202.0.0.13 | include |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Configured CIDR | Alert IP | Configured Condition | Alert |
| Source | 10.0.0.0/16 | 90.0.0.25 | include | No |
| Destination | 100.0.0.0/16 | 202.0.0.13 | include |

#

## ****Limitations:****

Currently this solution supports only a single CIDR range as filter for source and destination ips.

The following are not included in the solution.

1. Creating Slack channel
2. Test EC2 instance in the workload subnets
3. Test rules in Network Firewall
4. Actual or simulated traffic to trigger the test rules.
5. S3 bucket to hold the source files to be deployed.

# Appendix

References:

|  |  |
| --- | --- |
| Network Firewall Deployment Options | <https://aws.amazon.com/blogs/networking-and-content-delivery/deployment-models-for-aws-network-firewall/> |
| Network Firewall Policies | <https://docs.aws.amazon.com/waf/latest/developerguide/network-firewall-policies.html> |
| Solution source  | <https://gitlab.aws.dev/vramaam/security-aod10-vramaam.git> |
| Creating workspaces in Slack | <https://slack.com/help/articles/206845317-Create-a-Slack-workspace> |
| Credits  | <https://gitlab.aws.dev/minkimm/network-firewall-deployment-models/-/tree/master/decentralized-deployment-model/model-1> |