Incident Response Playbook Template

Incident Type

GuardDuty Finding: Discovery:Kubernetes/SuccessfulAnonymousAccess

Introduction

This playbook is provided as a template to customers using AWS products and who are building their incident response capability. You should customize this template to suit your particular needs, risks, available tools and work processes.

Security and Compliance is a shared responsibility between you and AWS. AWS is responsible for “Security of the Cloud”, while you are responsible for “Security in the Cloud”. For more information on the shared responsibility model, please review our documentation (https://aws.amazon.com/compliance/shared-responsibility-model/).

You are responsible for making your own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) references current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. This document is provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

Summary

This Playbook

This playbook outlines response steps for incidents involving an API operation that was successfully invoked by the `system:anonymous user` in a Kubernetes cluster. These steps are based on the NIST Computer Security Incident Handling Guide (https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-61r2.pdf) (Special Publication 800-61 Revision 2) that can be used to:

- Gather evidence
- Contain and then eradicate the incident
- Recover from the incident
- Conduct post-incident activities, including post-mortem and feedback processes

Interested readers may also refer to the AWS Security Incident Response Guide (https://docs.aws.amazon.com/whitepapers/latest/aws-security-incident-response-guide/welcome.html) which contains additional resources.

Once you have customized this playbook to meet your needs, it is important that you test the playbook (e.g., Game Days) and any automation (functional tests), update as necessary to achieve the desired results, and then publish to your knowledge management system and train all responders.

Note that some of the incident response steps noted in each scenario may incur costs in your AWS account(s) for services used in either preparing for, or responding to incidents. Customizing these scenarios and testing them will help you to determine if additional costs will be incurred. You can use AWS Cost Explorer (https://aws.amazon.com/aws-cost-management/aws-cost-explorer/) and look at costs incurred over a particular time frame (such as when running Game Days) to establish what the possible impact might be.

In reviewing this playbook, you will find steps that involve processes that you may not have in place today. Proactively preparing for incidents means you need the right resource configurations, tools and services in place that allow you to respond to an incident. The next section will provide a summary of this incident type, and then cover the five steps (parts 1 - 5) for handling privileged containers.

This Incident Type

An Amazon GuardDuty finding represents a potential security issue detected within your network. GuardDuty generates a finding whenever it detects unexpected and potentially malicious activity in your AWS environment. All GuardDuty finding references in this playbook will be related to the GuardDuty finding JSON that can be seen in the GuardDuty console, downloaded from the GuardDuty or Security Hub console, or exported to S3.

Discovery:Kubernetes/SuccessfulAnonymousAccess (https://docs.aws.amazon.com/guardduty/latest/ug/guardduty_finding-types-kubernetes.html#discovery-kubernetes-successfulanonymousaccess)

An API commonly used to discover resources in a Kubernetes cluster was invoked by an unauthenticated user.

This finding informs you that an API operation was successfully invoked by the system:anonymous user. API calls made by system:anonymous are unauthenticated. The observed API is commonly associated with the discovery stage of an attack when an adversary is gathering information on your Kubernetes cluster. This activity indicates that anonymous or unauthenticated access is permitted on the API action reported in the finding and may be permitted on other actions. If this behavior is not expected, it may indicate a configuration mistake or that your credentials are compromised.

Details on what resources are involved with this activity can be found in the finding details (https://docs.aws.amazon.com/guardduty/latest/ug/guardduty_findings-summary.html#findings-resource-affected).

Incident Response Process

Part 1: Acquire, Preserve, Document Evidence

For Any Incident

1. You become aware of potential indicators of compromise (IoCs). These could come in various forms, but the original source is a GuardDuty finding:
    - An internal ticketing system (the sources of the ticket are varied and could include any of the means below)
    - From an alert in one of your monitoring systems either inside or external to AWS (that are ingesting GuardDuty Findings, in AWS, this could include AWS Security Hub)
    - Alarms or observations that resources have been created or deleted in your account that cannot be accounted for in your CMDB, exist in regions that you do not operate infrastructure in, or themselves have generated alerts (Amazon Detective (https://aws.amazon.com/detective/getting-started/) is a useful tool for understanding these relationships)
2. Confirm a ticket/case has been raised for the incident. If not, manually raise one
3. Determine and begin to document any end-user impact/experience of the issue. Findings should be documented in the ticket/case related to the incident
4. Open an incident war room using your standard communication tools, and page in the relevant stakeholders
5. In the case of automatically created tickets/cases, verify the finding in GuardDuty (what caused the ticket to be created?)

For This Incident Type

1. Identify the specific EKS cluster impacted. In GuardDuty, this will be in the Resource.EksClusterDetails.Name section of the finding.
2. Identify Pod, Node, and User information to be used in **Part 2**.
    - Pod information can be found in the Resource.KubernetesWorkloadDetails section of the GuardDuty finding.
    - Node information can be found after determining the pod name. Once you have the pod name you can run the command provided below to determine the node name.

        - kubectl get pods --namespace -o=jsonpath='{"\n"}'
    - User information can be found in the Resource.KubernetesDetails.KubernetesUserDetails section of the GuardDuty Finding.
3. Identify the origination information from which the Kubernetes APIs were called. In GuardDuty, Look at the Resource.Service.Action section of the finding
    - Verify that the IP address is valid for your enterprise users
    - Verify that the Location and/or ASN/Organization is known and valid for this request
4. Contact the owner of the cluster where the API operation was invoked by the `system:anonymous user` to verify this was valid and intended.
    - If the person states that they did invoke the API using anonymous access, verify that there is a valid requirement for enabling anonymous access:
        - Once verified, move on to Part 5 to review the incident, and propose improvements that would either automatically archive findings for valid use of anonymous access within your organization, or eliminate the need to enable anonymous access entirely.
        - If the requirement for anonymous access cannot be verified, communicate to stakeholders your intent to remediate and proceed to Part 2.
    - If the person states they did not invoke the API operation or that the `system:anonymous user` access is not valid, or you or the owner of the cluster are unable to verify who invoked the API operation, proceed to **Part 2**.

Part 2: Contain the Incident

If you determine that the activity is unauthorized, or decide it is prudent to assume so, the first priority is to prevent further compromise without impact to production workloads.

1. Verify that there is not a valid business requirement for use of the `system:anonymous user` that will cause an impact to production workloads if removed.

2. The first step is to examine the permissions that have been granted to the system:anonymous user, and determine what permissions are needed. To accomplish this, you need to first understand what permissions the `system:anonymous user` has. You can use an rbac-lookup tool (https://github.com/FairwindsOps/rbac-lookup) to list the Kubernetes roles and cluster roles bound to users, service accounts, and groups. An alternative method can be found at this GitHub page (https://github.com/mhausenblas/rbac.dev).

```
./rbac-lookup | grep -P 'system:(anonymous)|(unauthenticated)'
system:anonymous                cluster-wide        ClusterRole/system:discovery
system:unauthenticated          cluster-wide        ClusterRole/system:discovery
system:unauthenticated          cluster-wide        ClusterRole/system:public-info-viewer
```

1. Next, disassociate the system:unauthenticated group from system:discovery and system:basic-user ClusterRoles, by editing the ClusterRoleBinding.

- **Note:** Make sure to not remove system:unauthenticated from the system:public-info-viewer cluster role binding, because that will prevent the Network Load Balancer from performing health checks against the API server

- Run the command `kubectl edit clusterrolebindings system:discovery`. This command will open the current definition of system:discovery ClusterRoleBinding in your editor as shown in the sample .yaml configuration file below:

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this
# file will be reopened with the relevant failures.
#
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
annotations:
  rbac.authorization.kubernetes.io/autoupdate: "true"
creationTimestamp: "2021-06-17T20:50:49Z"
labels:
  kubernetes.io/bootstrapping: rbac-defaults
name: system:discovery
resourceVersion: "24502985"
selfLink: /apis/rbac.authorization.k8s.io/v1/clusterrolebindings/system%3Adiscovery
uid: b7936268-5043-431a-a0e1-171a423abeb6
roleRef:
apiGroup: rbac.authorization.k8s.io
kind: ClusterRole
name: system:discovery
subjects:
- apiGroup: rbac.authorization.k8s.io
kind: Group
name: system:authenticated
- apiGroup: rbac.authorization.k8s.io
kind: Group
name: system:unauthenticated
```

- Delete the entry for system:unauthenticated group
- Repeat the same steps for system:basic-user ClusterRoleBinding.

1. Continue to provide updates to incident stakeholders on the current state of the incident response.

Part 3: Eradicate the Incident

This is the stage for taking remedial action to minimize the impact of the unintended activities.

1. Using your preferred monitoring tool, access CloudTrail and search for all API actions performed by the AWS EKS resource in the last 90 days:

    1. If this tool is a third-party tool such as Splunk, Sumo Logic or others, follow the normal procedure for obtaining log information from that tool
    2. If you do not ingest CloudTrail logs into a third-party tool, but do send those logs to Amazon Simple Storage Service (Amazon S3), you will be able to use Amazon Athena to query the logs. The remaining steps will focus on AWS tools to retrieve the necessary information

2. Create an Athena table referencing the bucket containing your CloudTrail logs (https://aws.amazon.com/premiumsupport/knowledge-center/athena-tables-search-cloudtrail-logs/) that link also includes example queries that can be run in Athena

3. In the Athena console, run a query that shows all API actions taken by the compromised resources post-compromise date/time. From the resulting list, determine which API calls:

    - Accessed sensitive data (such as S3 GetObject)

NOTE to get S3 data events CloudTrail must be configured to collect S3 data events prior to the incident. More information about CloudTrail data events can be found here (https://docs.aws.amazon.com/awscloudtrail/latest/userguide/logging-data-events-with-cloudtrail.html)

- Created new AWS resources, such as databases, Amazon Elastic Compute Cloud (Amazon EC2) instances, AWS Lambda functions or S3 buckets, etc.
- Services that create resources should also be carefully checked; for example, CloudFormation Stacks/StackSets, AWS Firewall Manager Security Policies, AWS Elastic Beanstalk resources, etc.
- Created or modified AWS identity resources that could be used to extend a foothold into the account (or other accounts, for example with AWS Security Token Service (AWS STS) API methods such as AssumeRole). Within an account, API methods including (but not limited to) the following should also be investigated:
  - CreateUser
  - CreateRole
  - AssumeRole*
  - Get*Token
  - Attach*Policy
  - RunInstances (especially with PassRole)
  - *Image*
  - *Provider
  - Tag*
  - Untag*
  - Create*
  - Delete*
  - Update*
  - etc.
- Deleted existing AWS resources
- Modified existing AWS resources

Example queries

Get a list of total calls a user has made and whether they were successful

```
select eventname, count(*) as total, eventsource, awsregion, errormessage
from cloudtrail_logs
where useridentity.principalid = 'AROA…'
group by eventname, eventsource, awsregion, errormessage
order by total desc;
```

Find all events by a particular user after a given date range (not including Describe/List events):

```
select eventname, useridentity.username, sourceIPAddress, eventtime, requestparameters from cloudtrail_logs
where (useridentity.username like '%ExampleUser%')
and eventtime > '2017-06-01T00:00:00Z'
and (eventname not like '%Describe%' AND eventname not like '%List%')
order by eventtime asc;
```

For an IAM user:

```
select eventname, useridentity.username, sourceIPAddress, eventtime, requestparameters from cloudtrail_logs
where (useridentity.username like '%ExampleUser%')
and eventtime > '2017-06-01T00:00:00Z'
and (eventname not like '%Describe%' OR eventname not like '%List%')
order by eventtime asc;
```

1. Based on the results of the previous step, determine if any applications are potentially impacted:
    - Obtain the ARN and/or tag information for each resource impacted (from step 4, above)
    - Go back to your CMDB and determine which application that resource belongs to
    - Notify the application owner based on the results of the above steps
2. At this point you will need to follow the appropriate playbook based on the type of application impact or further compromised resources to determine full impact and contain any further compromise.

Part 4: Recover from the Incident

1. After identifying, containing, and eradicating any involved pods, nodes, users, and any other AWS resources follow the steps below to complete recovery (note that some steps below could also be considered as eradication).
2. For resources that were modified during the compromise:
    1. If the resource can be destroyed and replaced, do so. For example, Identify the vulnerability that compromised the pods, implement the fix for that vulnerability and start new replacement pods, and then delete the vulnerable pods.
    2. If the resource cannot be replaced, either:
        1. Restore the resource from a known good backup, or;
        2. Prepare a new resource and configure it into the application's infrastructure, while isolating the compromised resource and removing it from the application's infrastructure. Update your CMDB accordingly
        3. Either destroy the compromised resource, or continue to leave it isolated for post-incident forensics
3. For resources that were deleted during the compromise:
    1. Determine what (if any) application the resource belonged to, by checking your CMDB, or confirming the resource's tag(s) (Check AWS Config if the tags aren't listed in the CloudTrail entry and the resource is supported by AWS Config (https://docs.aws.amazon.com/config/latest/developerguide/resource-config-reference.html))
    2. If the deleted resource can be restored from backup, commence the restore procedure
    3. If the deleted resource cannot be restored from backup, consult your CMDB to obtain the resource's configuration, and recreate the resource and configure it into the application's infrastructure
4. For resources that were created during the compromise:
    1. Confirm these have been deleted or isolated for further analysis, per the steps in **Part 2**.

Part 5: Post-Incident Activity

This activity contains two parts. Firstly, some compromised resources may require forensic analysis, either to fulfill regulatory obligations or improve incident handling, both taking input from the root cause analysis that will result from forensic investigation. The second part is a "sharpen the saw" activity which helps teams to assess their response to the actual incident, determine what worked and what didn't, update the process based on that information and record these findings.

Firstly, perform any required forensic investigation to determine (for compromised resources) what methods the actors may have used and to determine if additional risks and risk mitigations are required for the resources and/or applications in question.

1. For any compromised resources that have been isolated for further analysis, perform the forensic activity on those resources and incorporate the findings into the post-incident report.
2. Ensure that the CMDB is correctly updated to reflect the current status of all resources and applications impacted
3. Once the analysis is completed, ensure relevant resources are terminated and the resulting data and any artifacts from the analysis maintained and/or entered into the relevant system for record-keeping

Secondly, review the incident itself and the response to it, to determine if anything needs to be changed for handling any similar incidents in the future.

1. Review the incident handling and the incident handling process with key stakeholders identified in Part 1, Step 8
2. Document lessons learned, including attack vector(s) mitigation(s), misconfiguration, etc.

3. Store the artifacts from this process with the application information in the CMDB entry for the application and also in the CMDB entry for the credential compromise response process.

Additional Resources

Incident Response

https://aws.amazon.com/blogs/security/how-get-started-security-response-automation-aws/ (https://aws.amazon.com/blogs/security/how-get-started-security-response-automation-aws/)

https://docs.aws.amazon.com/whitepapers/latest/https://docs.aws.amazon.com/whitepapers/latest/aws-security-incident-response-guide/welcome.html (https://docs.aws.amazon.com/whitepapers/latest/https://docs.aws.amazon.com/whitepapers/latest/aws-security-incident-response-guide/welcome.html)

https://aws.amazon.com/blogs/security/how-to-automate-incident-response-in-aws-cloud-for-ec2-instances/ (https://aws.amazon.com/blogs/security/how-to-automate-incident-response-in-aws-cloud-for-ec2-instances/)

https://aws.amazon.com/blogs/security/forensic-investigation-environment-strategies-in-the-aws-cloud/ (https://aws.amazon.com/blogs/security/forensic-investigation-environment-strategies-in-the-aws-cloud/)

https://aws.github.io/aws-eks-best-practices/security/docs/incidents/ (https://aws.github.io/aws-eks-best-practices/security/docs/incidents/)

https://docs.aws.amazon.com/guardduty/latest/ug/guardduty-remediate-kubernetes.html (https://docs.aws.amazon.com/guardduty/latest/ug/guardduty-remediate-kubernetes.html)

GuardDuty

https://docs.aws.amazon.com/guardduty/latest/ug/guardduty_findings-summary.html (https://docs.aws.amazon.com/guardduty/latest/ug/guardduty_findings-summary.html)

Other

https://docs.aws.amazon.com/quicksight/latest/APIReference/API_Reference.html (https://docs.aws.amazon.com/quicksight/latest/APIReference/API_Reference.html)

https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html (https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html)

https://docs.aws.amazon.com/pdfs/eks/latest/APIReference/eks-api.pdf (https://docs.aws.amazon.com/pdfs/eks/latest/APIReference/eks-api.pdf)