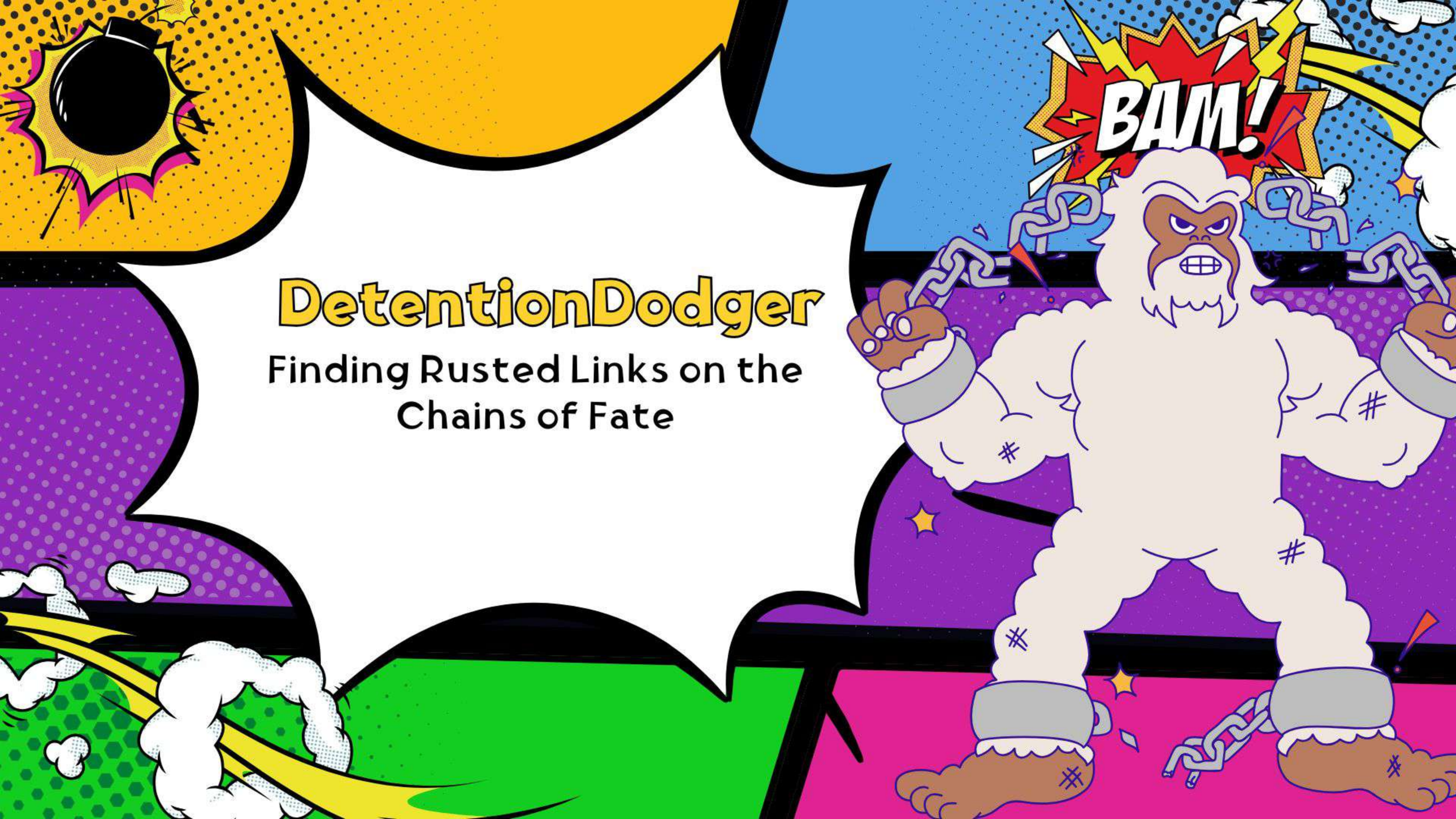
# DetentionDodger

## Finding Rusted Links on the Chains of Fate




LOOK! UP IN THE SKY!
IT'S A BIRD!
IT'S A PLANE!
IT'S Permiso !

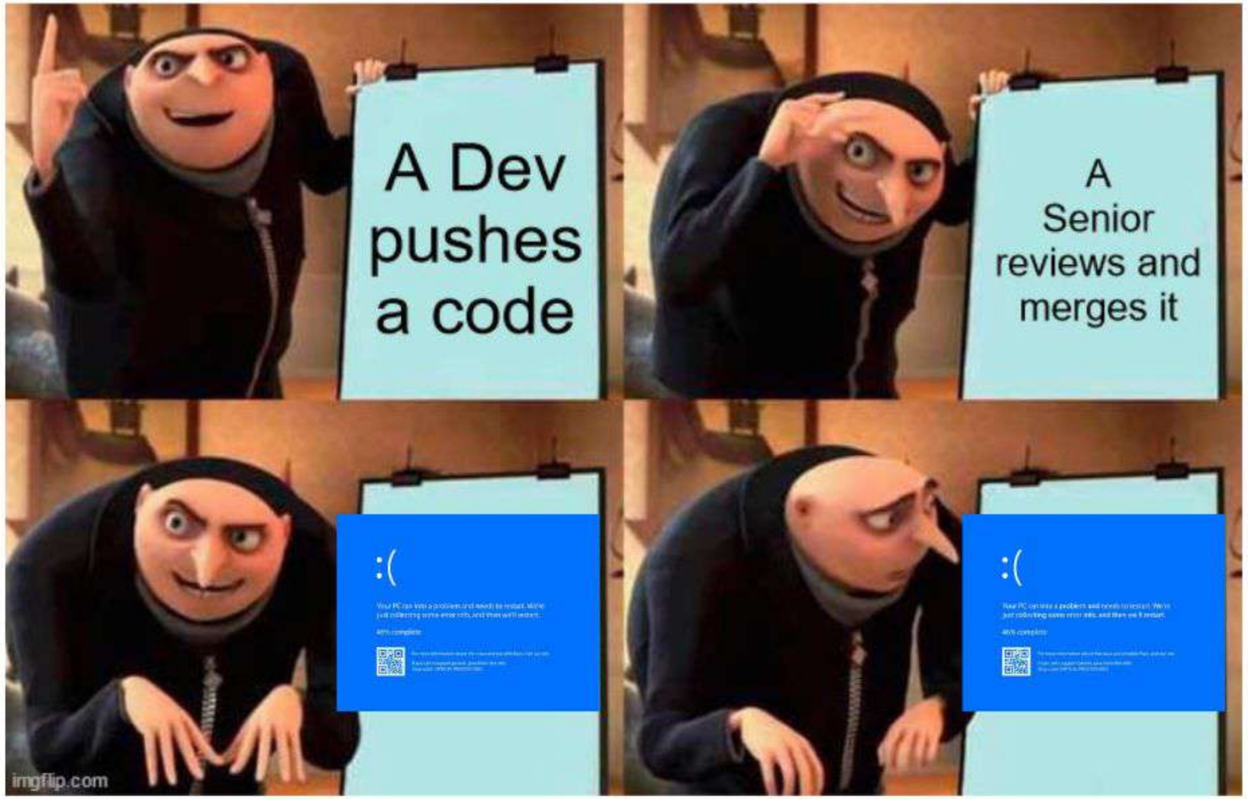

Andrew Kraut
aka Opie


Bleon Proko
aka gl4ssesbol

# Summary

- What is AWSCompromisedKeyQuarantineV2 Policy
- What privileges does the policy not restrict
- How we can bypass it
- How we can bypass it with style
- Now what?

# Hypothetically speaking...



Hello,

We have become aware that the AWS access key AKIAZBGIGYA4N7A6KFUO, belonging to IAM User s3admin, along with the corresponding secret key is publicly available online at https://github.com/bleonproko/Somescripts/blob/53cfad58eae9a9e3d89c0f89b637883ed999e410/.aws.

To protect your account, we have temporarily limited your ability to use some AWS services.

To protect your account from unwanted activity, we have applied the "AWSCompromisedKeyQuarantineV2" AWS Managed Policy ("Quarantine Policy") to the IAM User [user_id]. The Quarantine Policy applied to the IAM User [user_id] protects your account by denying access to high risk actions like iam:CreateAccessKey and ec2:RunInstances. Please refer to tehe "AWSCompromisedKeyQuarantineV2 Permissions" [1] to review all of the actions denied by the policy.

Please do not remove the Quarantine Policy before following the instructions below. However, in cases where the Quarantine Policy is causing production issues you may choose to detach the policy from the user. Only users with admin privileges or with access to iam:DetachUserPolicy may remove the policy. Refer to the IAM User Guide [2] on how to remove managed policies.

Hello,

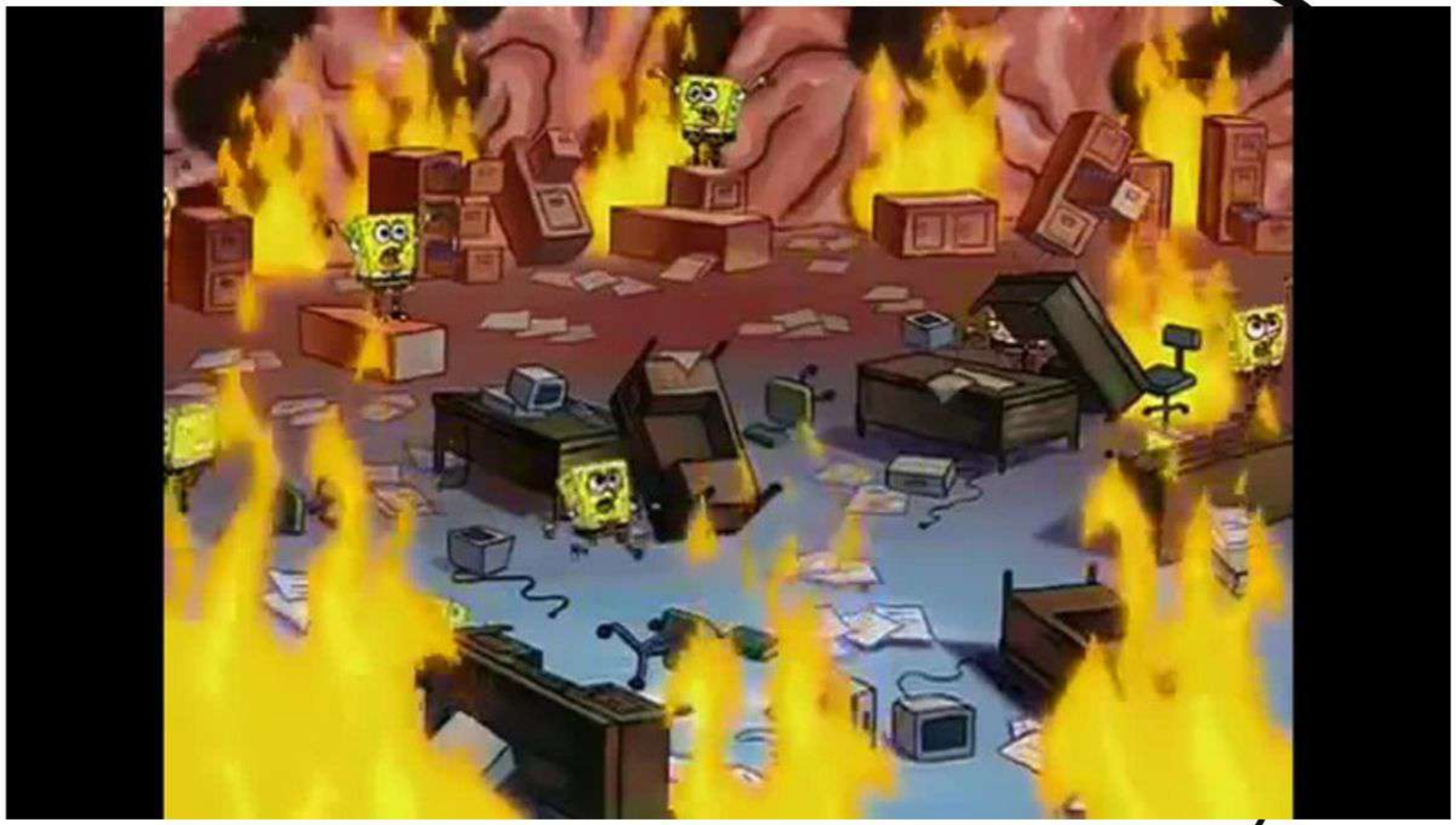
We have become aware that the AWS access key AKIAZBGIGYA4EDFA2S57, belonging to User administrator along with the corresponding secret key is publicly available online at https://github.com/bleonproko/Somescripts/blob/df6f2a967b451001aeca14aaabffc124dc94c05c/.aws.

To protect your account, we have temporarily limited your ability to use some AWS services.

To restore access, you must contact AWS by and follow the instructions below. If you do not contact AWS by , we may suspend your account. We may terminate any suspicious resources on your account, and some resources may not be recoverable once terminated.

As a security best practice, we recommend that you enable multi-factor authentication (MFA) [1].

Follow the instructions below and for further detailed instructions, please refer to the “What do I do if I notice unwanted activity in my AWS account?” user guide [2].



# My user is leaked? Luckily my infrastructure is protected by AWSCompromisedKeyQuarantineV2

AWSCompromisedKeyQuarantineV2 is an AWS policy that attaches to identities whose credentials are leaked. It denies access to certain actions, applied by the AWS team in the event that an IAM credentials have been compromised or exposed publicly.

# Policy Definition

```
{"Version" : "2012-10-17",
  "Statement" : [
    {"Effect" : "Deny",
      "Action" : [
        "cloudtrail:LookupEvents",
        "ec2:RequestSpotInstances",
        "ec2:RunInstances",
        "ec2:StartInstances",
        "ec2:PurchaseReservedInstancesOffering",
        "ec2:AcceptReservedInstancesExchangeQuote",
        "ec2:CreateReservedInstancesListing",
        "iam:AddUserToGroup",
        "iam:AttachGroupPolicy",
        "iam:AttachRolePolicy",
        "iam:AttachUserPolicy",
        "iam:ChangePassword",
        "iam:CreateAccessKey",
        "iam:CreateInstanceProfile",
        "iam:CreateLoginProfile",
        "iam:CreatePolicyVersion",
        "iam:CreateRole",
        "iam:CreateUser",
        "iam:DetachUserPolicy",
        "iam:PassRole",
        "iam:PutGroupPolicy",
        "iam:PutRolePolicy",
        "iam:PutUserPermissionsBoundary",
        "iam:PutUserPolicy",
        "iam:SetDefaultPolicyVersion",
        "iam:UpdateAccessKey",
        "iam:UpdateAccountPasswordPolicy",
        "iam:UpdateAssumeRolePolicy",
        "iam:UpdateLoginProfile",
        "iam:UpdateUser",
        "lambda:AddLayerVersionPermission",
        "lambda:AddPermission",
        "lambda:CreateFunction",
        "lambda:GetPolicy",
        "lambda:ListTags",
        "lambda:PutProvisionedConcurrencyConfig",
        "lambda:TagResource",
        "lambda:UntagResource",
        "lambda:UpdateFunctionCode",
        "lightsail:Create*",
        "lightsail:Delete*",
        "lightsail:DownloadDefaultKeyPair",
        "lightsail:GetInstanceAccessDetails",
        "lightsail:Start*",
        "lightsail:Update*",
        "organizations:CreateAccount",
        "organizations:CreateOrganization",
        "organizations:InviteAccountToOrganization",
        "s3:DeleteBucket",
        "s3:DeleteObject",
        "s3:DeleteObjectVersion",
        "s3:PutLifecycleConfiguration",
        "s3:PutBucketAcl",
        "s3:PutBucketOwnershipControls",
        "s3:DeleteBucketPolicy",
        "s3:ObjectOwnerOverrideToBucketOwner",
        "s3:PutAccountPublicAccessBlock",
        "s3:PutBucketPolicy",
        "s3:ListAllMyBuckets",
        "savingsplans:CreateSavingsPlan"
      ],
      "Resource" : [
        "*"
      ]
    }
  ]
}
```

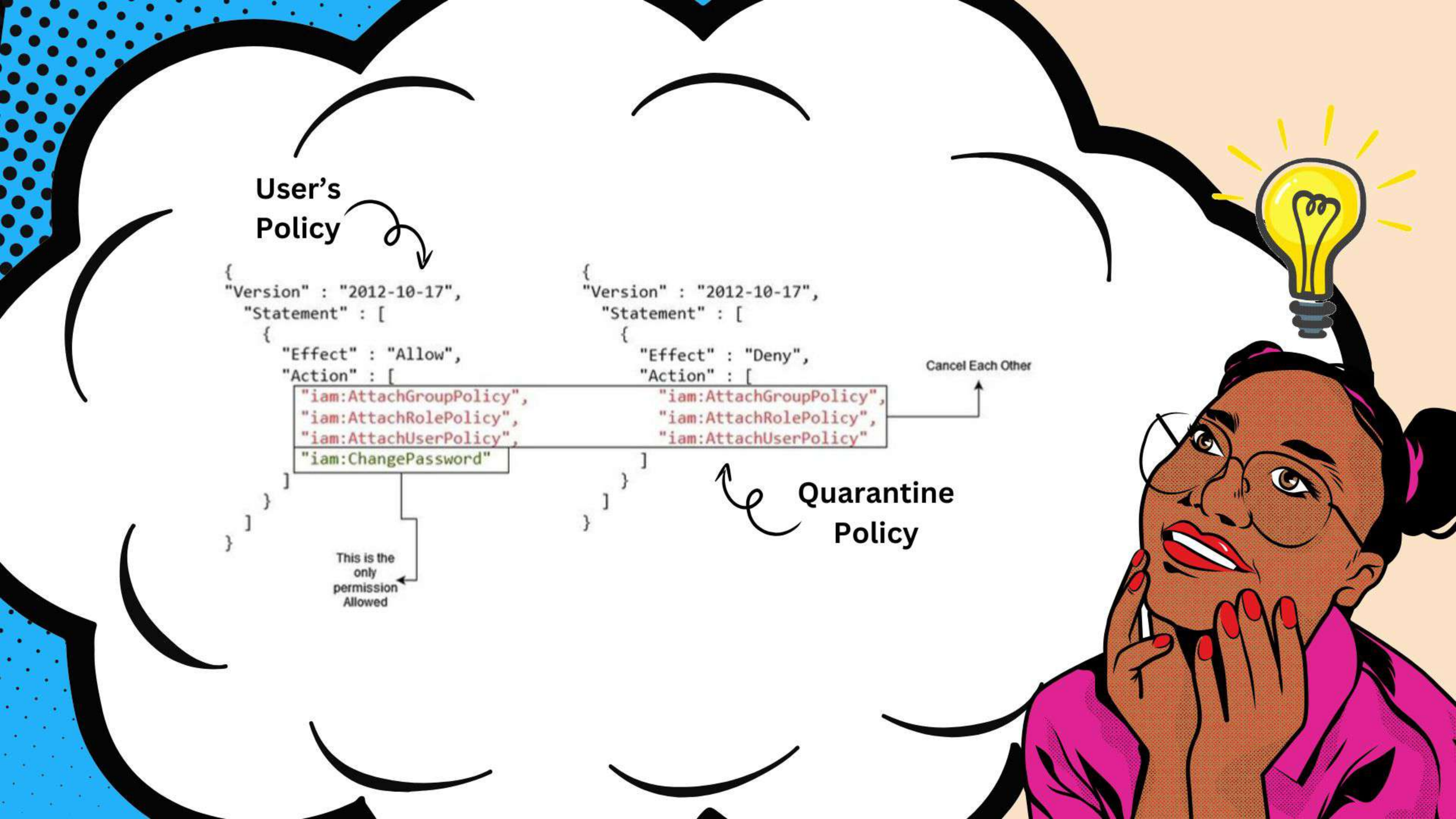
# Ok, But how?

```
{
"Version" : "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "iam:AttachGroupPolicy",
        "iam:AttachRolePolicy",
        "iam:AttachUserPolicy",
        "iam:ChangePassword"
      ]
    }
  ]
}
```

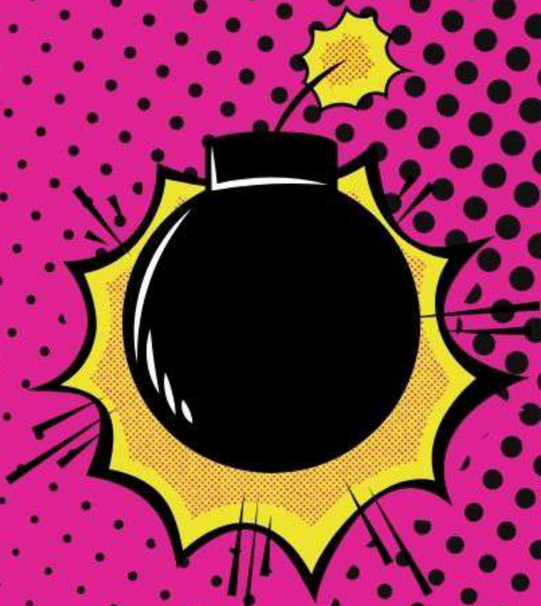
This is the only permission Allowed

```
{
"Version" : "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Deny",
      "Action" : [
        "iam:AttachGroupPolicy",
        "iam:AttachRolePolicy",
        "iam:AttachUserPolicy"
      ]
    }
  ]
}
```

Cancel Each Other


User's Policy
{
"Version" : "2012-10-17",
"Statement" : [
{
"Effect" : "Allow",
"Action" : [
"iam:AttachGroupPolicy",
"iam:AttachRolePolicy",
"iam:AttachUserPolicy",
"iam:ChangePassword"
]
}
]
}
This is the only permission Allowed
{
"Version" : "2012-10-17",
"Statement" : [
{
"Effect" : "Deny",
"Action" : [
"iam:AttachGroupPolicy",
"iam:AttachRolePolicy",
"iam:AttachUserPolicy"
]
}
]
}
Cancel Each Other
Quarantine Policy

How much of a security feature can the Quarantine Policy be considered as?



# Quarantine policy once more...

```
{"Version" : "2012-10-17",
  "Statement" : [
    {"Effect" : "Deny",
      "Action" : [
        "cloudtrail:LookupEvents",
        "ec2:RequestSpotInstances",
        "ec2:RunInstances",
        "ec2:StartInstances",
        "ec2:PurchaseReservedInstancesOffering",
        "ec2:AcceptReservedInstancesExchangeQuote",
        "ec2:CreateReservedInstancesListing",
        "iam:AddUserToGroup",
        "iam:AttachGroupPolicy",
        "iam:AttachRolePolicy",
        "iam:AttachUserPolicy",
        "iam:ChangePassword",
        "iam:CreateAccessKey",
        "iam:CreateInstanceProfile",
        "iam:CreateLoginProfile",
        "iam:CreatePolicyVersion",
        "iam:CreateRole",
        "iam:CreateUser",
        "iam:DetachUserPolicy",
        "iam:PassRole",
        "iam:PutGroupPolicy",
        "iam:PutRolePolicy",
        "iam:PutUserPermissionsBoundary",
        "iam:PutUserPolicy",
        "iam:SetDefaultPolicyVersion",
        "iam:UpdateAccessKey",
        "iam:UpdateAccountPasswordPolicy",
        "iam:UpdateAssumeRolePolicy",
        "iam:UpdateLoginProfile",
        "iam:UpdateUser",
        "lambda:AddLayerVersionPermission",
        "lambda:AddPermission",
        "lambda:CreateFunction",
        "lambda:GetPolicy",
        "lambda:ListTags",
        "lambda:PutProvisionedConcurrencyConfig",
        "lambda:TagResource",
        "lambda:UntagResource",
        "lambda:UpdateFunctionCode",
        "lightsail:Create*",
        "lightsail:Delete*",
        "lightsail:DownloadDefaultKeyPair",
        "lightsail:GetInstanceAccessDetails",
        "lightsail:Start*",
        "lightsail:Update*",
        "organizations:CreateAccount",
        "organizations:CreateOrganization",
        "organizations:InviteAccountToOrganization",
        "s3:DeleteBucket",
        "s3:DeleteObject",
        "s3:DeleteObjectVersion",
        "s3:PutLifecycleConfiguration",
        "s3:PutBucketAcl",
        "s3:PutBucketOwnershipControls",
        "s3:DeleteBucketPolicy",
        "s3:ObjectOwnerOverrideToBucketOwner",
        "s3:PutAccountPublicAccessBlock",
        "s3:PutBucketPolicy",
        "s3:ListAllMyBuckets",
        "savingsplans:CreateSavingsPlan"
      ],
      "Resource" : [
        "*"
      ]
    }
  ]
}
```

# Enumeration not Allowed

- List S3 Buckets
- Lightsail Get Instance Access
- cloudtrail:LookupEvents
- lambda:ListTags
- lambda:GetPolicy

# Enumeration

- Enumerate IAM
- List and Download S3 Objects
- Lambda Get Function Code and Env Vars
- SecretsManager no Restrictions
- No KMS Restrictions
- CloudFormation Read*
- All EC2, LightSail, EKS, ECR, Fargate, etc read

And basically any enumeration

# Enumeration Using Policy Simulation

## Identity Policy

```
$ aws iam simulate-custom-policy --policy-input-list '{"Version": "2012-10-17", "Statement": [{"Sid": "VisualEditor0", "Effect": "Allow", "Action":
"*", "Resource": "*"}]}' --action-names iam:ListUsers --profile quarantinedUser
{
    "EvaluationResults": [
        {
            "EvalActionName": "iam:ListUsers",
            "EvalResourceName": "arn:aws:iam::[redacted]:user/",
            "EvalDecision": "allowed",
            "MatchedStatements": [
                {
                    "SourcePolicyId": "PolicyInputList.1",
                    "SourcePolicyType": "IAM Policy",
                    "StartPosition": {
                        "Line": 1,
                        "Column": 41
                    },
                    "EndPosition": {
                        "Line": 1,
                        "Column": 116
                    }
                }
            ],
            "MissingContextValues": []
        }
    ]
}
```

iam:SimulateCustomPolicy

## Principal Policy

```
$ aws iam simulate-principal-policy --policy-source-arn arn:aws:iam::[redacted]:role/Role1 --action-names sts:AssumeRole --profile quarantinedUser
{
    "EvaluationResults": [
        {
            "EvalActionName": "sts:AssumeRole",
            "EvalResourceName": "*",
            "EvalDecision": "allowed",
            "MatchedStatements": [
                {
                    "SourcePolicyId": "AdministratorAccess",
                    "SourcePolicyType": "IAM Policy",
                    "StartPosition": {
                        "Line": 3,
                        "Column": 17
                    },
                    "EndPosition": {
                        "Line": 8,
                        "Column": 6
                    }
                }
            ],
            "MissingContextValues": [],
            "OrganizationsDecisionDetail": {
                "AllowedByOrganizations": true
            }
        }
    ]
}
```

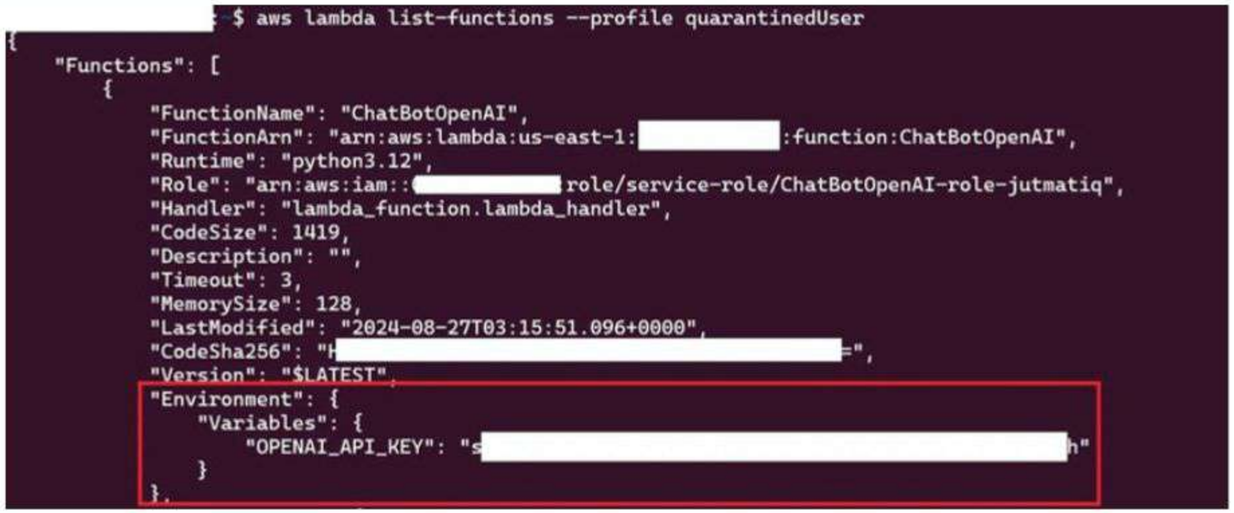
iam:SimulatePrincipalPolicy

# Lambda Function Download

## Download Function


lambda:GetFunction

## Get Function Environment Variables




lambda:ListFunctions

lambda:GetFunction


# SecretsManager Secrets Read

sm:ListSecrets

```
$ aws secretsmanager list-secrets --profile quarantinedUser
{
    "SecretList": [
        {
            "ARN": "arn:aws:secretsmanager:us-east-1:[redacted]:secret:best_albanian_dance-Mqcglo",
            "Name": "best_albanian_dance",
            "LastChangedDate": "2023-11-10T10:16:21.224000-05:00",
            "LastAccessedDate": "2024-08-29T20:00:00-04:00",
            "Tags": [],
            "SecretVersionsToStages": {
                "8[redacted]2": [
                    "AWSCURRENT"
                ]
            },
            "CreatedDate": "2023-11-10T10:16:20.673000-05:00"
        },
[redacted]$ aws secretsmanager get-secret-value --secret-id arn:aws:secretsmanager:us-east-1:[redacted]:secret:best_city_in_the_balkans-wkBVyG --profile qua
rantinedUser
{
    "ARN": "arn:aws:secretsmanager:us-east-1:[redacted]:secret:best_city_in_the_balkans-wkBVyG",
    "Name": "best_city_in_the_balkans",
    "VersionId": "4[redacted]9",
    "SecretString": "{\"best_city_in_the_balkans\":\"gjakova\"}",
    "VersionStages": [
        "AWSCURRENT"
    ],
    "CreatedDate": "2023-11-10T10:17:08.208000-05:00"
}
```

sm:GetSecretValue

# Privilege Escalation

## Not Allowed

- Creating a new policy version
- Setting the default policy version to an existing version
- Creating an EC2 instance with an existing instance profile
- Creating a new user access key
- Creating a new login profile
- Updating an existing login profile
- Attaching a policy to a user
- Attaching a policy to a group
- Attaching a policy to a role
- Creating/updating an inline policy for a user
- Creating/updating an inline policy for a group
- Creating/updating an inline policy for a role
- Adding a user to a group
- Passing a role to a new Lambda function, then invoking it cross-account
- Updating the code of an existing Lambda function
- Passing a role to CloudFormation
- Passing a role to a new CodeStar project
- Passing a role to a new SageMaker Jupyter notebook
- Passing a role to a Glue Development Endpoint
- Creating a CodeStar project from a template (no longer possible)

## Partially Allowed

- Assume Role (Updating the Assume Role Policy Document of a role not allowed)
- Invoke Lambda Function (Passing a role to a new Lambda function not allowed)
- Invoke Lambda using DynamoDB (Passing a role to a new Lambda function not allowed)
- Update DataPipeline Definition

## Allowed

- Updating an existing Glue Dev Endpoint
- Associating a Codestar team member
- Adding a malicious Lambda layer to an existing Lambda function
- Gaining access to an existing SageMaker Jupyter notebook

# Non Administrative Infrastructure Impact

- SSM Command Execution on Instances
- S3 Bucket pillage
- S3 Ransomware
- Lambda Function Invoking and Deleting
- Shutting down instances in production
- Getting Management Console Access
- Stopping and Deleting CloudTrail
- Tampering with GuardDuty
- Bedrock full access

# Getting Attached to Permissions...

Another issue we noticed with the policy is the fact that the policy uses the current identity's credentials to attach the policy to itself, thus requiring iam:AttachUserPolicy to be executed by the user to itself.

Hello,

We have become aware that the AWS access key AKIAZBGIGYA4EDFA2S57, belonging to User administrator along with the corresponding secret key is publicly available online at https://github.com/bleonproko/Somescripts/blob/df6f2a967b451001aeca14aaabffc124dc94c05c/.aws.

To protect your account, we have temporarily limited your ability to use some AWS services.

No Attach Policy Access

To restore access, you must contact AWS by and follow the instructions below. If you do not contact AWS by , we may suspend your account. We may terminate any suspicious resources on your account, and some resources may not be recoverable once terminated.

As a security best practice, we recommend that you enable multi-factor authentication (MFA) [1].

Follow the instructions below and for further detailed instructions, please refer to the "What do I do if I notice unwanted activity in my AWS account?" user guide [2].

Attach Policy Access

Hello,

We have become aware that the AWS access key AKIAZBGIGYA4N7A6KFUO, belonging to IAM User s3admin, along with the corresponding secret key is publicly available online at https://github.com/bleonproko/Somescripts/blob/53cfad58eae9a9e3d89c0f89b637883ed999e410/.aws.

To protect your account, we have temporarily limited your ability to use some AWS services.

To protect your account from unwanted activity, we have applied the "AWSCompromisedKeyQuarantineV2" AWS Managed Policy ("Quarantine Policy") to the IAM User [user_id]. The Quarantine Policy applied to the IAM User [user_id] protects your account by denying access to high risk actions like iam:CreateAccessKey and ec2:RunInstances. Please refer to tehe "AWSCompromisedKeyQuarantineV2 Permissions" [1] to review all of the actions denied by the policy.

Please do not remove the Quarantine Policy before following the instructions below. However, in cases where the Quarantine Policy is causing production issues you may choose to detach the policy from the user. Only users with admin privileges or with access to iam:DetachUserPolicy may remove the policy. Refer to the IAM User Guide [2] on how to remove managed policies.

# Getting Attached to Permissions...

Another thing we found out is that when the iam:AttachUserPolicy request fails, neither requestParameters or responseElements fields are filled

Unfilled fields

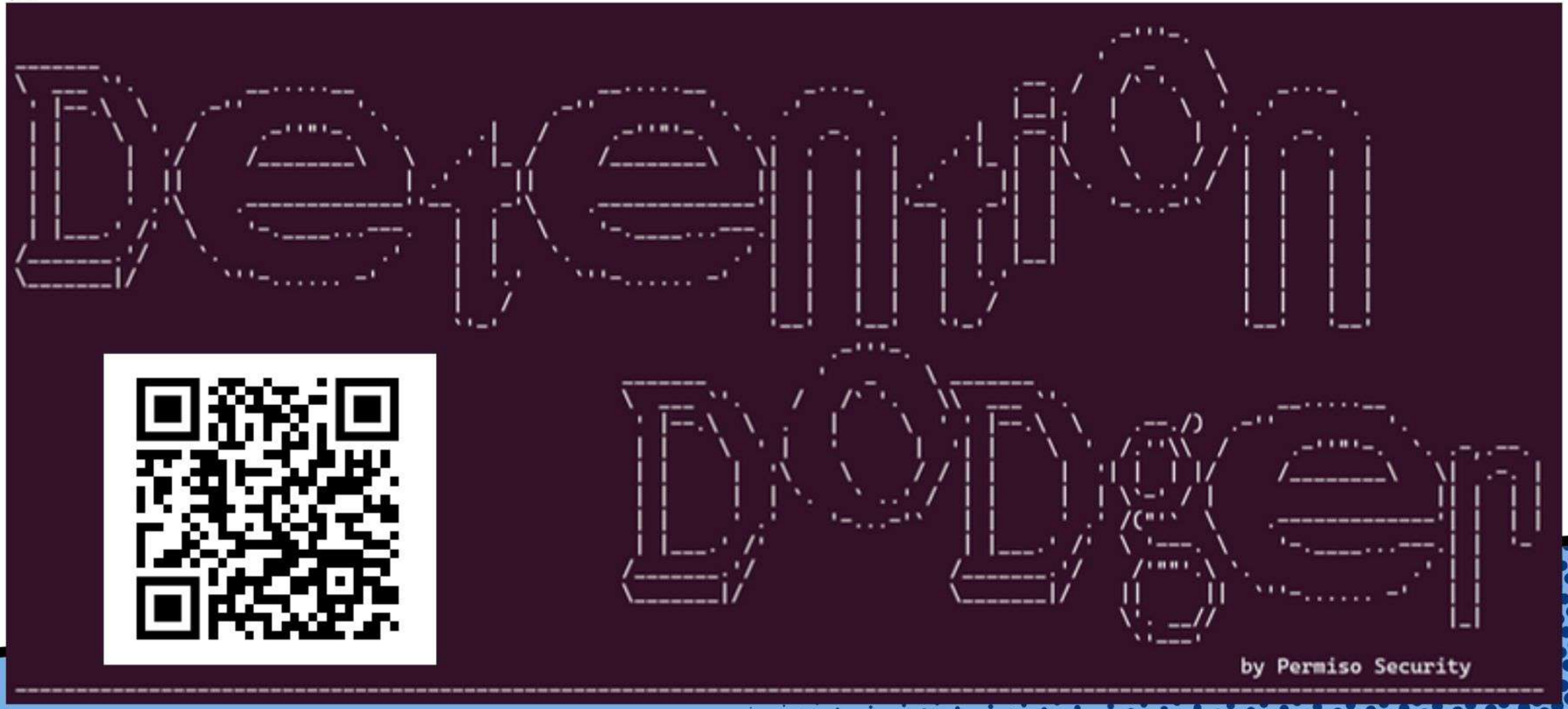

"requestParameters": null,
"responseElements": null,

# AWS Response

- On August 21st 2024 AWS created AWSCompromisedKeyQuarantineV3 policy and on October 2nd 2024 AWS updated AWSCompromisedKeyQuarantineV3 policy with new additions:

- amplify:CreateBackendEnvironment
- amplify:CreateDeployment
- **bedrock:CreateFoundationModelAgreement**
- **bedrock:CreateModelInvocationJob**
- **bedrock:InvokeModel**
- **bedrock:InvokeModelWithResponseStream**
- **bedrock:PutFoundationModelEntitlement**
- codebuild:CreateProject
- ecr:GetAuthorizationToken
- ecs:CreateCluster
- ecs:CreateService
- ecs:RegisterTaskDefinition
- glue:CreateJob
- iam:DeleteAccessKey
- iam:DeleteRole
- iam:ListUsers
- lambda:GetEventSourceMapping
- mediapackagev2:CreateChannel
- s3:CreateBucket
- **s3:GetObject**
- **s3:ListBucket**
- s3:PutBucketCors
- sagemaker:CreateEndpointConfig
- sagemaker:CreateProcessingJob
- ses:GetSendQuota
- ses:ListIdentities
- sns:GetSMSAttributes
- **sts:GetFederationToken**
- sts:GetSessionToken

# DetentionDodger



# How does the tool work?

1) The tool will get a list of Users and their PermissionBoundary using **iam:ListUsers** and **iam:GetUser**
2) For each user, a list of attached user policies will be taken using **iam:ListAttachedUserPolicies** and a list of Inline Policies using **iam:ListUserPolicies**
3) If the user has any of the **AWSCompromisedKeyQuarantineV*** Policies attached, they will be checked. Otherwise they are skipped.
4) A list of **iam:AttachUserPolicy** events of the user on itself with **requestParameters** and **responseElements** being null will be taken to get a more throughout list using **cloudtrail:LookupEvents**
5) Then, a list of user's groups is taken using **iam:ListGroupsForUser** and a list of the group's policies using **iam:ListAttachedUserPolicies** and **iam:ListGroupPolicies**
6) For each policy, the id of the default policy version is taken using **iam:GetPolicy** and a policy document using **iam:GetPolicyVersion**
7) Lastly, all the policies and PermissionBoundary will be valuated for a list of privileges predefined using **iam:SimulateCustomPolicy**

# Demo Time

Thank you!
Any
quersioins?