

4. Prevent untrusted principals from using an AWS service as a confused deputy

Use case

The [confused deputy problem](#) is a security issue where an entity that doesn't have permission to perform an action can coerce a more-privileged entity to perform the action. In this particular scenario, the more-privileged entity is an AWS service. Confused deputy controls allow you to help prevent someone in another AWS account that you don't own from using an AWS service to read from or write to your AWS resources.

Policy type - [Resource-based policy](#)

Values to use in policy:

Service: s3.amazonaws.com

Action: sns:Publish

Region: us-east-1

Topic name: MyTopic

Account that the SNS topic is in: 111111111111

Account that you want to receive notifications from: 111111111111

S3 bucket that you want to receive notifications from: MY-BUCKET

The Approach

1. Setting Up the Principal and Action:

Principal: Specified as the AWS service s3.amazonaws.com to directly identify the service allowed to interact with our SNS topic.

Action: Limited to sns:Publish, focusing the policy on the ability to publish messages to the SNS topic.

2. Defining Resource and Conditions:

Resource: The [SNS topic](#) arn:aws:sns:us-east-1:111111111111:MyTopic, which ensures that the policy applies specifically to our topic.

Conditions:

ArnEquals: Ensures that the S3 bucket acting as the source is specifically "MY-BUCKET."

StringEquals: Restricts the allowable source account to our own AWS account ID (111111111111), mitigating the risk of the confused deputy problem by tying the permission to our controlled environments.

The Execution

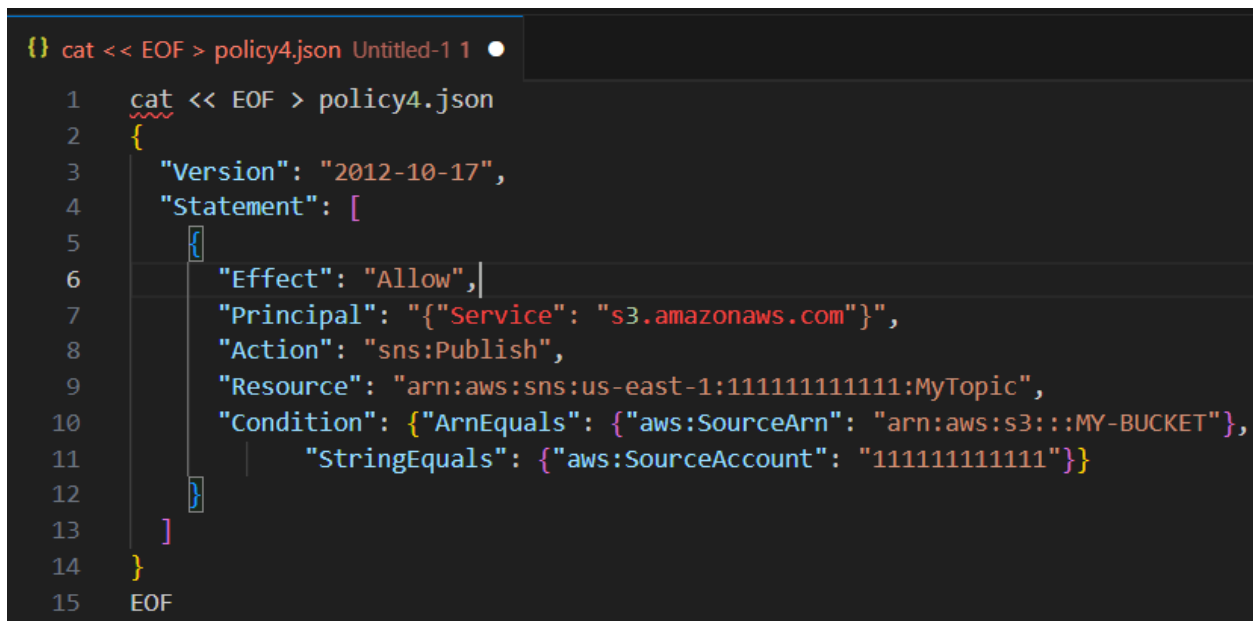
Addressing Potential Security Gaps:

The initial setup included a check for the source S3 bucket ARN. However, considering the possibility of ARN spoofing or reuse (if the bucket were to be deleted and recreated by

another user), the `aws:SourceAccount` condition was added. This ensures that even if the bucket name is duplicated in another account, the policy does not allow unauthorized publishing.

```
cat << EOF > policy4.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {"Service": "s3.amazonaws.com"},
      "Action": "sns:Publish",
      "Resource": "arn:aws:sns:us-east-1:111111111111:MyTopic",
      "Condition": {"ArnEquals": {"aws:SourceArn": "arn:aws:s3:::MY-BUCKET"},
        "StringEquals": {"aws:SourceAccount": "111111111111"}}
    }
  ]
}
EOF
```

As you have already noticed, there was a JSON syntax error for the Principal line of code. I used VS Code tool to check where the error was - it didn't highlighted exactly the issue, but I noticed that's something odd there: the brackets and the quotes were extra:

A screenshot of a VS Code editor window titled 'Untitled-1 1'. The editor shows a JSON policy file being created with the command 'cat << EOF > policy4.json'. The JSON content is as follows:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {"Service": "s3.amazonaws.com"},
      "Action": "sns:Publish",
      "Resource": "arn:aws:sns:us-east-1:111111111111:MyTopic",
      "Condition": {"ArnEquals": {"aws:SourceArn": "arn:aws:s3:::MY-BUCKET"},
        "StringEquals": {"aws:SourceAccount": "111111111111"}}
    }
  ]
}
```

 The editor highlights several syntax errors with red squiggly lines: an extra closing brace '}' at the end of line 1, an extra closing brace '}' at the end of line 12, and an extra closing brace '}' at the end of line 14. The 'EOF' keyword is on line 15.

Corrected the policy and moved forward to the next step.

Testing and Validation

This command was crucial for confirming the policy's operational effectiveness under simulated AWS environment conditions:

```
eval-policy --step 4 --policy policy4.json
```

```
Policy being evaluated: {
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "s3.amazonaws.com"
      },
      "Action": "sns:Publish",
      "Resource": "arn:aws:sns:us-east-1:111111111111:MyTopic",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": "arn:aws:s3:::MY-BUCKET"
        },
        "StringEquals": {
          "aws:SourceAccount": "111111111111"
        }
      }
    }
  ]
}
```

	Expected	Actual	Test details
Pass	Allowed	Allowed	S3 service principal should be allowed to publish to my SNS topic on my behalf
Pass	Denied	Denied	S3 service principal should be denied access when publishing to my SNS topic on behalf of someone else's account

The Outcome

The policy effectively secures the SNS topic from potential misuse through the "confused deputy" problem. By tying the publish permissions explicitly to our S3 bucket and our AWS account, the policy ensures that only legitimate notifications from our specific S3 bucket reach our SNS topic, thereby maintaining data integrity and security across our AWS services. This approach not only mitigates security risks but also aligns with best practices for cross-service AWS configurations.