Google
Developers

Keys to the Kingdom: OAuth in Google Cloud Platform

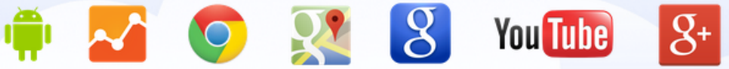
Adam Eijdenberg
Product Manager
eijdenberg@google.com

Ben Sittler
Software Engineer
bsittler@google.com

Google I/O 13

Google™ Cloud Platform

Google Services



Compute

- Compute Engine (IaaS)
- App Engine (PaaS)

Storage

- Cloud Storage (Blob)
- Cloud SQL (Relational)
- Cloud DataStore (NoSQL)
- Persistent Disk (Block)

App Services

- BigQuery
- Cloud EndPoints
- Caching
- Queues
- and more...

Google Infrastructure

- Global Data Centers
- 99.95% Uptime SLA
- Performance
- Redundancy
- Disaster Recovery
- Audits & Certifications
- Security
- Energy Efficient

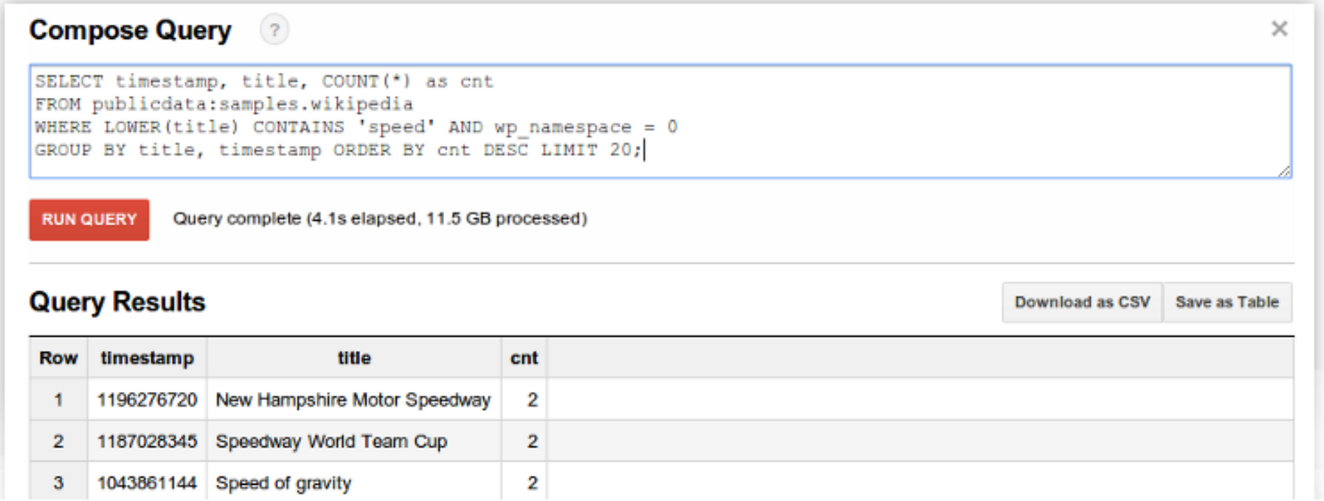
What are we covering today?

- **Authentication**
  - From first principles, using Google Client Libraries
  - From anywhere, Google App Engine, Google Compute Engine
- **Authorization**

Let's make an API call

Analyze terabytes of data with just a click of a button

Use Google BigQuery to interactively analyze massive datasets — up to billions of rows.



What is Google BigQuery?

Google BigQuery is a web service that lets you do interactive analysis of massive datasets—up to billions of rows. Scalable and easy to use, BigQuery lets developers and businesses tap into powerful data analytics on demand.

Learn More

Sign Up for BigQuery

Ready to try out BigQuery? BigQuery is now available to the public! Sign up for the service now, or learn more.

Need enterprise level support? Contact a sales representative.

Sign Up

All I want to do is make an API call!

- What is the minimum I need to do to make a successful API call against a Google API?

```
GET /bigquery/v2/projects/1234/datasets HTTP/1.1
Host: www.googleapis.com
Authorization: Bearer yaXXXXXXXXXX
```

- You need a token - This presentation will show you how to get one - painlessly!

What is a token?

- Called an "Access Token" - identifies 3 things to Google:

1. The application (called **Client**) making the request
2. The **User** (if applicable) authorizing the request
3. What class of actions the user has authorized the application to perform (called **Scope**)

How do I get one?

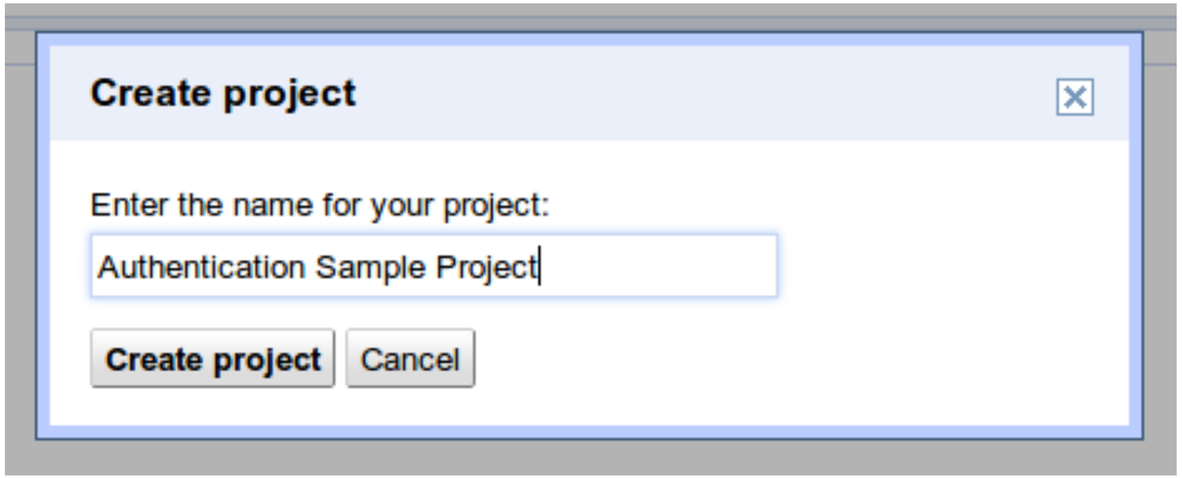
- This depends on *whose* data is being accessed
- Data owned by a Google **user**
  - e.g. show consent screen for your application to access data on behalf of a user, e.g. Google Drive, Google Calendar
- Data owned by my **application** or provided by Google
  - e.g. Google Cloud Storage, Google BigQuery API, Google Compute Engine
- This presentation will focus on accessing data belonging to your **application** - typical pattern for Cloud Platform APIs

Configuring the project

Accessing data owned by my application

- Let's get started with BigQuery

https://developers.google.com/console/



Project

- Container for:
  - Resources
    - Google Cloud Storage buckets
    - Google Compute Engine virtual machines
    - BigQuery datasets
  - Registered applications and credentials

- Holds configuration for:
  - Authorization
  - Billing / quota management
  - Services enabled

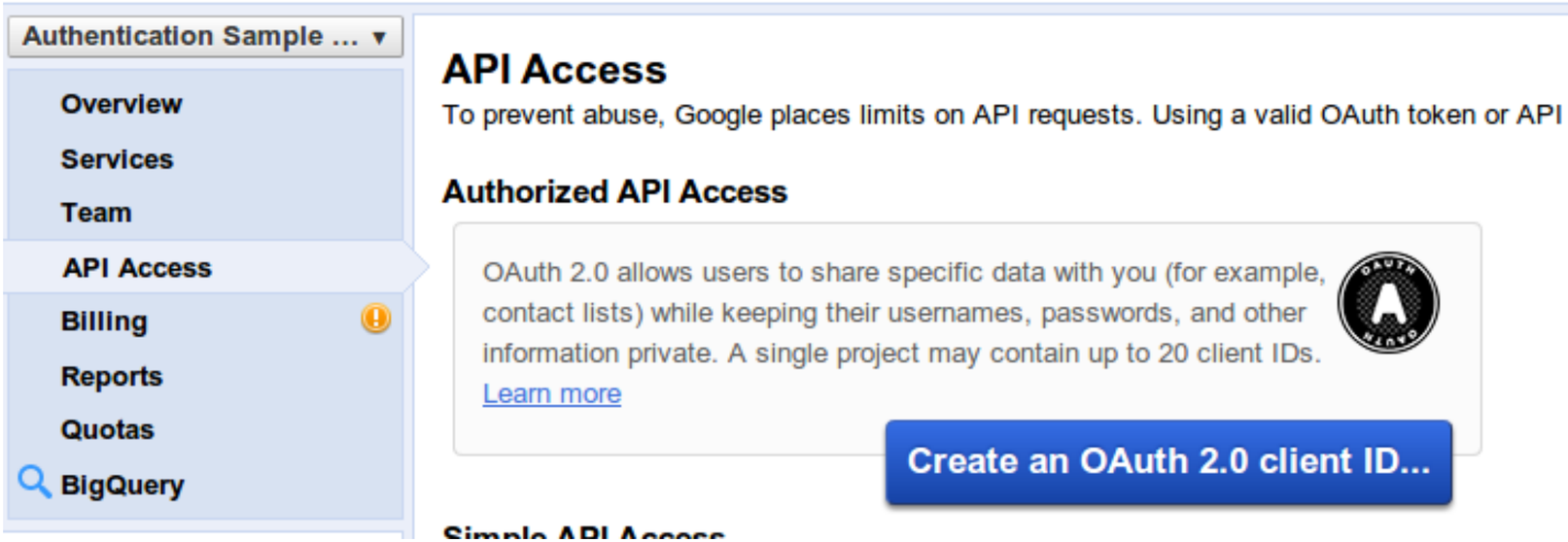
Enable the service

- Remember the access token identifies the application making the request?

- The API you are calling must be enabled within the project where the application is registered (next step)

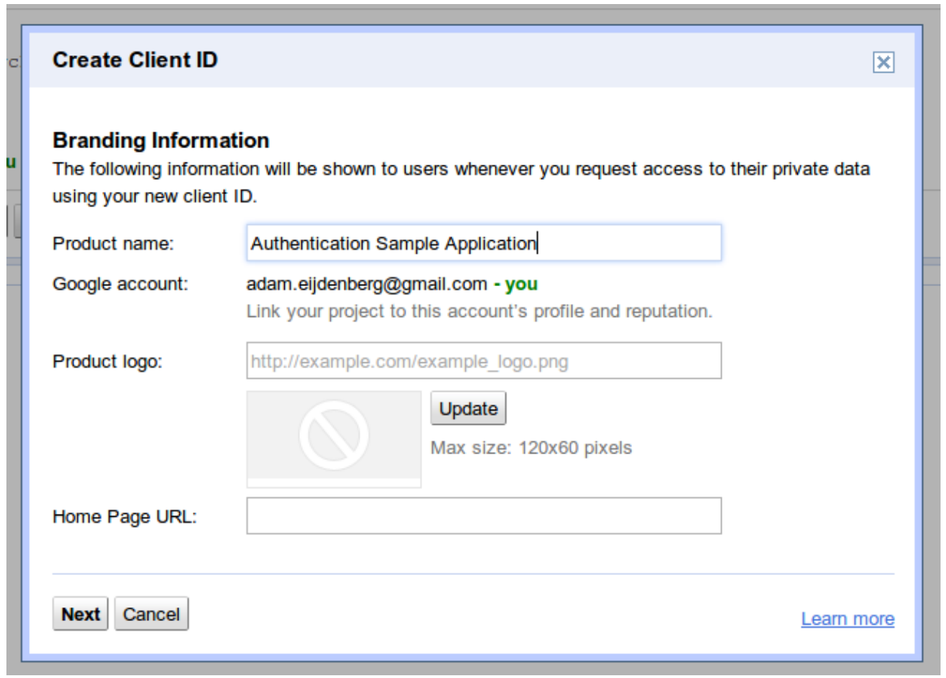
Create a service account

- A service account is a non-human user
- Allows your application to make API calls on behalf of itself
- Identity for your application - like a role account

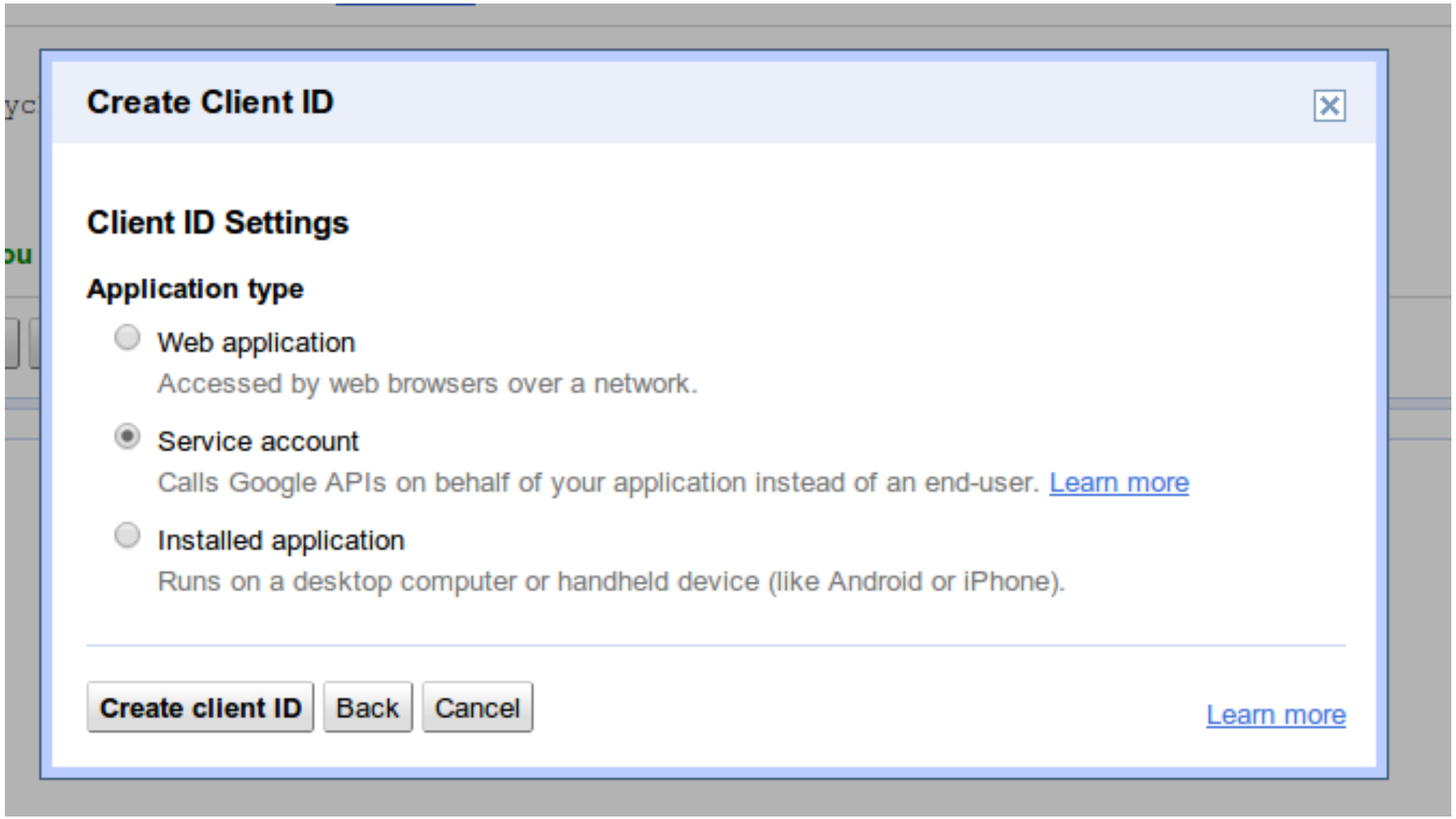


Branding Information screen

- The first time a Client is created you will be prompted for the following - this is not used for application authentication.



Select "Service Account"



Download key

Take note of the email address

Service account

Use service accounts to call Google APIs on behalf of your application instead of an end-user. Learn more

Client ID:	587080245102.apps.googleusercontent.com
Email address:	587080245102@developer.gserviceaccount.com
Public key fingerprints:	b4becc0154b4511aa1916a6e3008b45c401f1a84 – Delete...

Create another client ID...

Let's get an access token

Create an assertion

```
import time, json, base64

now = long(time.time())
assertion_input = "%s.%s" % (
    base64.urlsafe_b64encode(
        json.dumps({"alg": "RS256", "typ": "JWT"}).encode("UTF-8")
    ).rstrip("="),
    base64.urlsafe_b64encode(json.dumps({
        "iss": "yyyyyyy@developer.gserviceaccount.com",
        "scope": "https://www.googleapis.com/auth/bigquery",
        "aud": "https://accounts.google.com/o/oauth2/token",
        "exp": now + (60 * 60), "iat": now
    }).encode("UTF-8")).rstrip("="))
```

Sign it - and swap it

```
import urllib2, urllib
from OpenSSL import crypto

access_token = json.loads(urllib2.urlopen("https://accounts.google.com/o/oauth2/token",
  urllib.urlencode({
      "grant_type": "urn:ietf:params:oauth:grant-type:jwt-bearer",
      "assertion": "%s.%s" % (assertion_input,
          base64.urlsafe_b64encode(crypto.sign(crypto.load_pkcs12(
              file("xxxxxxxx-privatekey.p12", "rb").read(),
              "notasecret").get_privatekey(), assertion_input, "sha256")).rstrip("="))
  })).read())["access_token"]
```

Result


```
{
  "access_token" : "yaXXXXXXXXXXXXXXXXXXXXXXXXX",
  "token_type" : "Bearer",
  "expires_in" : 3600
}
```


Try using it

```
print urllib2.urlopen(urllib2.Request(
    "https://www.googleapis.com/bigquery/v2/projects/%s/datasets"
        % "my-test-project-id",
    headers={'Authorization': 'Bearer %s' % access_token}
)).read()
```

Result:

```
{
  "kind": "bigquery#datasetList",
  "etag": "..."
}
```

Using Google Client libraries

```
import httplib2
from oauth2client.client import SignedJwtAssertionCredentials

http = SignedJwtAssertionCredentials(
    "yyyyyyyyyyy@developer.gserviceaccount.com",
    file("xxxxxxxxxxx-privatekey.p12", "rb").read(),
    scope="https://www.googleapis.com/auth/bigquery"
).authorize(httplib2.Http())
```

Use authorized HTTP object directly

```
print http.request("https://www.googleapis.com/bigquery/v2/projects/%s/datasets"
    % "my-test-project-id")[1]
```

Result:

```
{
  "kind": "bigquery#datasetList",
  "etag": "..."
}
```

Or the full client library stack

```
from apiclient.discovery import build

service = build('bigquery', 'v2')
print service.datasets().list(projectId="my-test-project-id").execute(http)
```

Result:

```
{
  "kind": "bigquery#datasetList",
  "etag": "..."
}
```

Google hosted environments

Google hosted environments?

- Built-in service account support provided for:
  - Google Compute Engine
  - Google App Engine
- You need to add the Google App Engine service account email address to the team for the project that contains the data you wish to access.
- For some APIs you will also need to pass an API key in the request

Use built-in identity in Google App Engine

```
# Built-in:
from google.appengine.api import app_identity

access_token, ttl = app_identity.get_access_token(
    "https://www.googleapis.com/auth/bigquery")

# With client library:
from oauth2client.appengine import AppAssertionCredentials

http = AppAssertionCredentials(
    "https://www.googleapis.com/auth/bigquery").authorize(httplib2.Http())
```

Use built-in identity in Google Compute Engine

```
# When starting your VM - authorize the VM to use the scope you need:
gcutil --project=my-test-project-id addinstance foobar \
    --service_account_scopes=https://www.googleapis.com/auth/bigquery

# Inside the VM:
curl "http://metadata/computeMetadata/v1beta1/instance/service-accounts/default/token"

# Result:
{
  "access_token" : "yaXXXXXXXXXXXXXXXXXXXXXXX",
  "token_type" : "Bearer",
  "expires_in" : 3600
}
```

Use client library on Google Compute Engine

```
from oauth2client.gce import AppAssertionCredentials

http = AppAssertionCredentials(
    "https://www.googleapis.com/auth/bigquery").authorize(httplib2.Http())
```

Authentication - summary

- Your applications need an identity (like a role account) when they access Google APIs
- When running in Google App Engine or Google Compute Engine use built-in credentials
- When running elsewhere (including local Google App Engine development) create a service account and download a private key

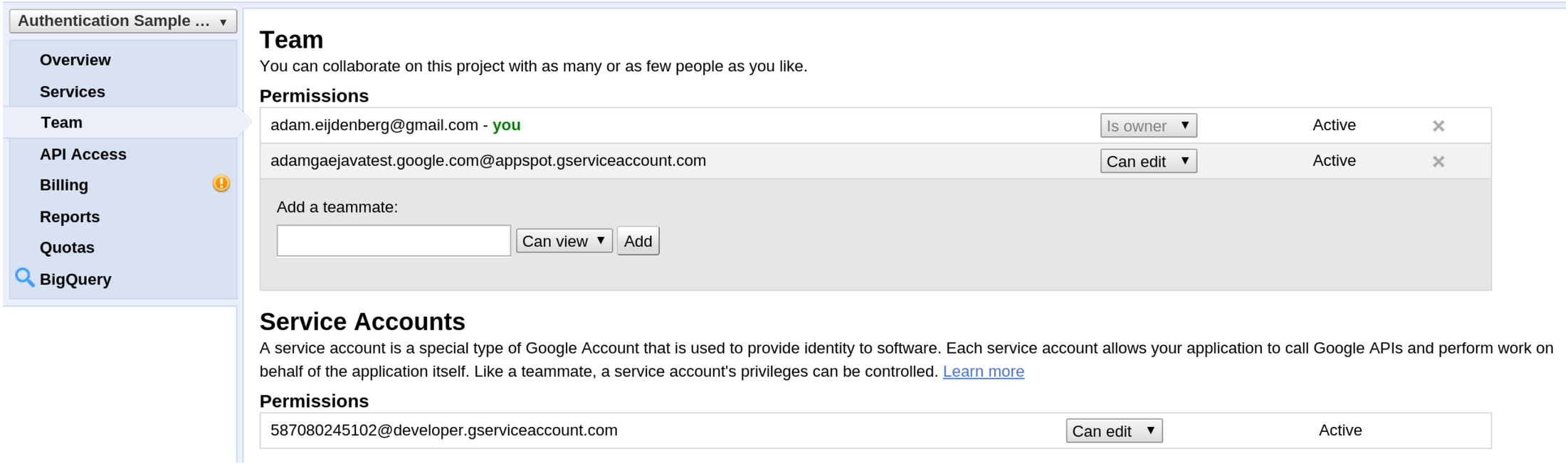
What is a token?

- Called an "Access Token" - identifies 3 things to Google:

1. The application (called **Client**) making the request
2. The **User** (if applicable) authorizing the request
3. What class of actions the user has authorized the application to perform (called **Scope**)

Authorization is based on the user

- Ensure the email address associated with the user is added as a "Team" member on the project associated with the resource being accessed
- Correct scope associated with access token



Billing/quota based on client

- Validate against the project associated with client ID:
  - Service is enabled
  - Billing state is active (where applicable)
  - Quota limit (for API call) has not been exceeded
    - Per project (e.g. courtesy limit)
    - Per user - configurable by developers
- For APIs that include accessing resources belonging to a project (e.g. most Cloud Platform APIs), the above is also validated against the project owning that resource (if different)

Summary

- Access token must be presented
  - Recommend client libraries to acquire token
- Authorize user by adding to appropriate Team
- Now, go and use our APIs to change the world!
  - (or at least solve your problems!)
- Ask questions if you are stuck!
  - StackOverflow (google-oauth)

Thank You!

Adam Eijdenberg - eijdenberg@google.com
Ben Sittler - bsittler@google.com

Google Cloud Platform Resources

cloud.google.com - get started today

developers.google.com - docs and developer guidance

cloud.google.com/newsletter - stay up to date

googlecloudplatform.blogspot.com - our blog

https://developers.google.com/api-client-library/ - client libraries

https://developers.google.com/accounts/docs/OAuth2ServiceAccount - protocol details

Get questions answered on StackOverflow (google-oauth)

Google
Developers