Google™ Apps

APIs for Domain Admins

by warden@geneseo.edu

The Lay of the Land

$$\begin{tikzcd}
 & \text{Google Apps Platform} \arrow[dl, "\text{Contains}"'] \arrow[dr, "\text{Contains}"] & \\
\text{Applications} \arrow[dr, "\text{Each with their own}"'] & & \text{Admin SDK} \arrow[dl, "\text{Contains many}"] \\
 & \text{API} \arrow[dl, "\text{Accessed by client libraries}"'] \arrow[dr] & \\
\text{GData} & & \text{“API Client”*}
\end{tikzcd}$$

*I do not like the name “API Client”

Imagine if someone tried name a car “The Car”...

...Oh, wait.

APIs

Application

Calendar API

Tasks API

Gmail APIs

Contacts API

Apps Activity API

Drive API

Spreadsheets APIs

Sites API

Admin SDK

Admin Settings API

Calendar Resource API

★ Directory API

Domain Shared Contacts API

Email Audit API

Email Migration API

Email Settings API

Groups Migration API

Groups Settings API

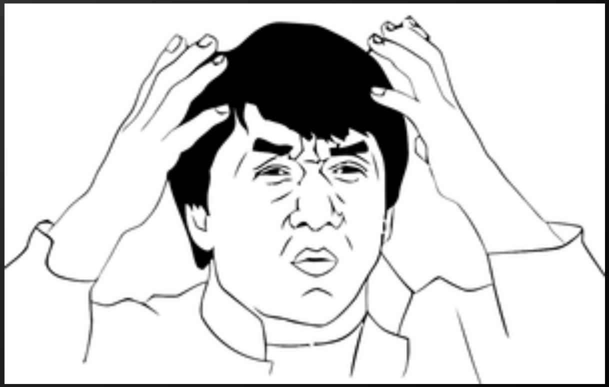
Enterprise License Manager API

★ Reports API

Reseller API

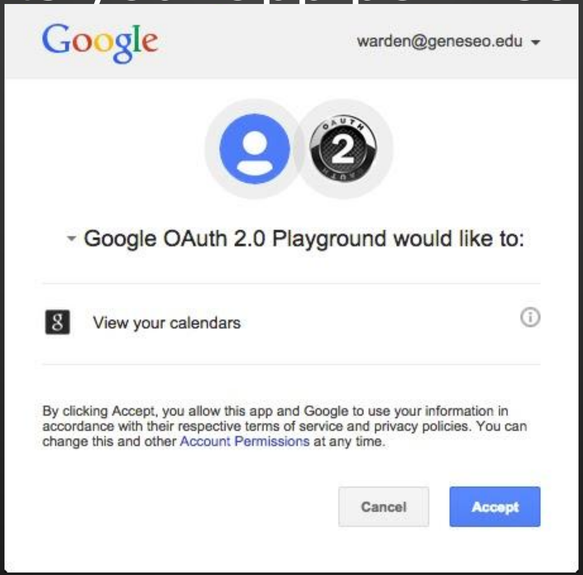
GData vs “API Client”

- Generally, “API Client” is newer and preferred
- Some GData APIs have been deprecated and migrated to “API Client”
  - Calendar - Shutdown Nov 17, 2014
  - YouTube Data - Deprecated March 2014
  - Provisioning - Deprecated May 2013
  - Documents List - Deprecated September 2012
- But some things are ONLY available in GData!
  - Contacts
  - Sites
  - Spreadsheets



Authentication

- OAuth 2.0 is your only option in many APIs
  - 2 Legged OAuth 2.0 (2LO)
    - Administratively access your GApps domain users
    - “domain-wide delegation of authority”
  - 3 Legged OAuth 2.0
    - User grants your app permission at a Google screen


Google warden@geneseo.edu

Google OAuth 2.0 Playground would like to:

View your calendars

By clicking Accept, you allow this app and Google to use your information in accordance with their respective terms of service and privacy policies. You can change this and other Account Permissions at any time.

2LO Overview

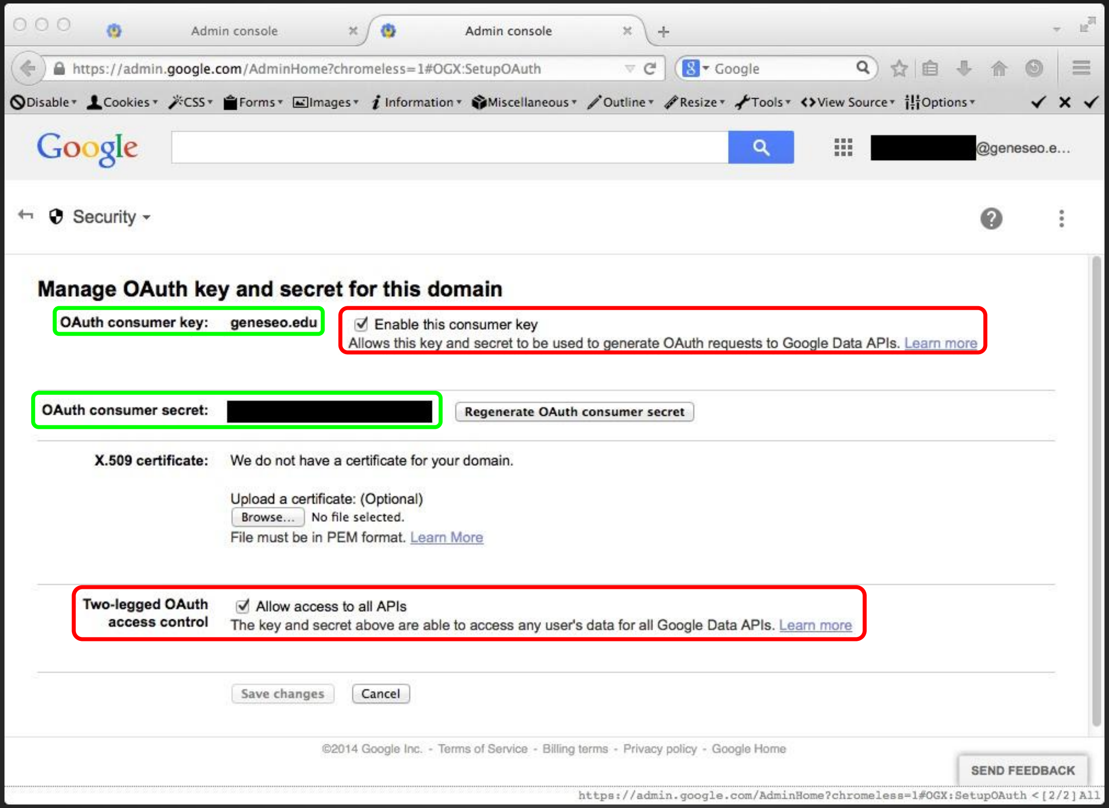
Did you think it would be easy?

GData and “API Client” have different 2LO implementations!

GData 2LO Overview

1. Get Consumer Key and Consumer Secret
2. Ensure 2LO is enabled
3. Use Consumer Key and Secret to authenticate to GData API

GDatа 2LO Steps 1 & 2


https://admin.google.com/AdminHome?chromeless=1#OGX:SetupOAuth

[redacted]@geneseo.e...

Security

Manage OAuth key and secret for this domain

OAuth consumer key: geneseo.edu

☑ Enable this consumer key
Allows this key and secret to be used to generate OAuth requests to Google Data APIs. Learn more

OAuth consumer secret: [redacted]

Regenerate OAuth consumer secret

X.509 certificate: We do not have a certificate for your domain.

Upload a certificate: (Optional)
Browse... No file selected.
File must be in PEM format. Learn More

Two-legged OAuth access control: ☑ Allow access to all APIs
The key and secret above are able to access any user's data for all Google Data APIs. Learn more

Save changes | Cancel

©2014 Google Inc. - Terms of Service - Billing terms - Privacy policy - Google Home

Get here via

1. Google Admin Console
2. Security
3. Advanced settings
4. Manage OAuth domain key

GData 2LO Step 3 - Meat & Potatoes

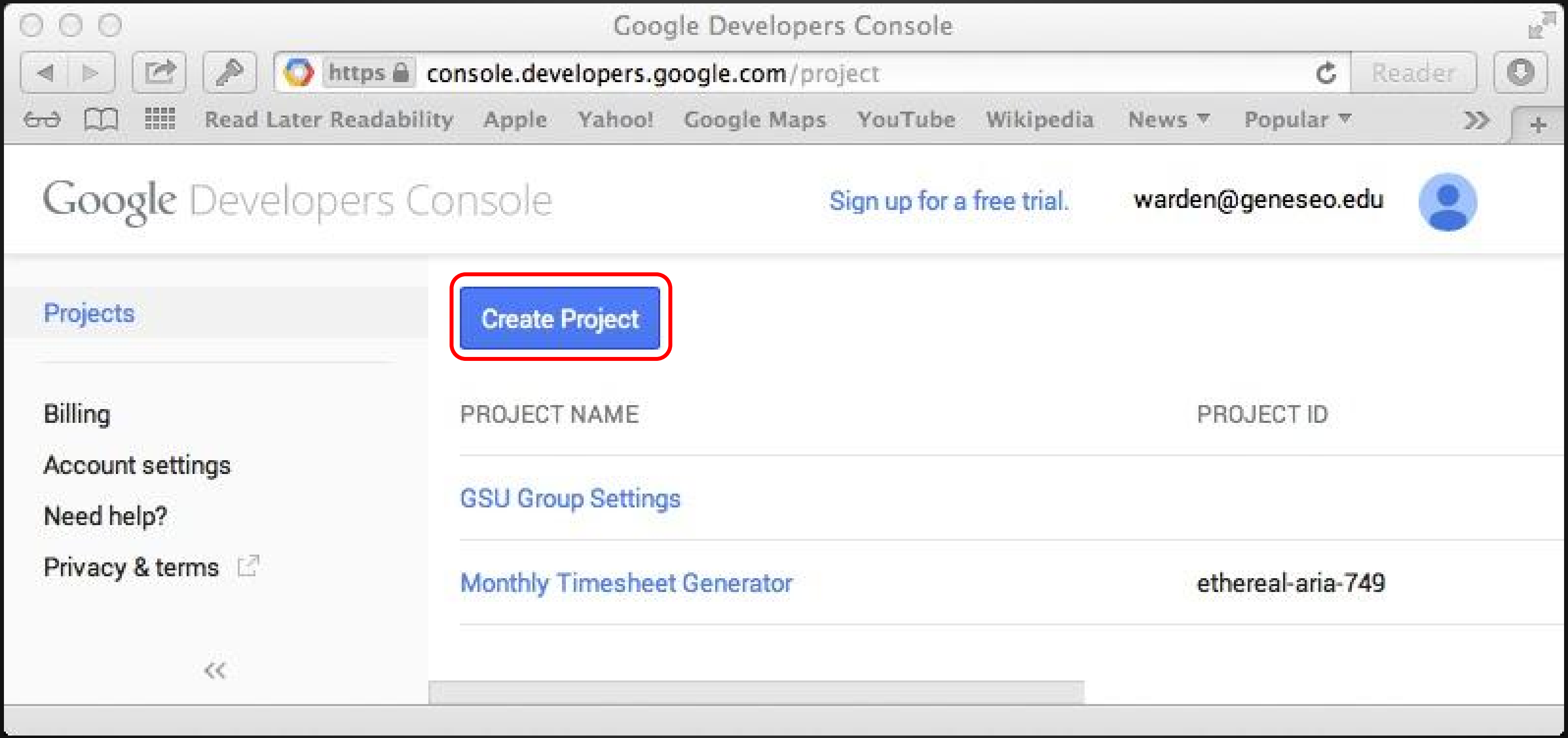
```
import gdata.gauth
import gdata.contacts.client

SOURCE_APP_NAME = 'anything-you-want'
CONSUMER_KEY = 'your-consumer-key'  # Generally, your GApps domain
CONSUMER_SECRET = 'your-consumer-secret'
requestor_id = 'gapps-user-email-address'  # User you wish to access

two_legged_oauth_token = gdata.gauth.TwoLeggedOAuthHmacToken(
    CONSUMER_KEY, CONSUMER_SECRET, requestor_id)

contacts_client = gdata.contacts.client.ContactsClient
(source=SOURCE_APP_NAME)

contacts_client.auth_token = two_legged_oauth_token

contacts_list = contacts_client.GetContacts()
for entry in contacts_list.entry:
    print entry.title.text
```

“API Client” 2LO Overview

1. Create new project on Google Developers Console (GDC)
2. Enable APIs in GDC
3. Create service account in GDC
4. Enable scopes for service account in Google Admin Console
5. Use service account .p12 key to authenticate to “API Client” API

“API Client” 2LO Step 1: Create Project

https://console.developers.google.com


Google Developers Console

https console.developers.google.com/project

Google Developers Console

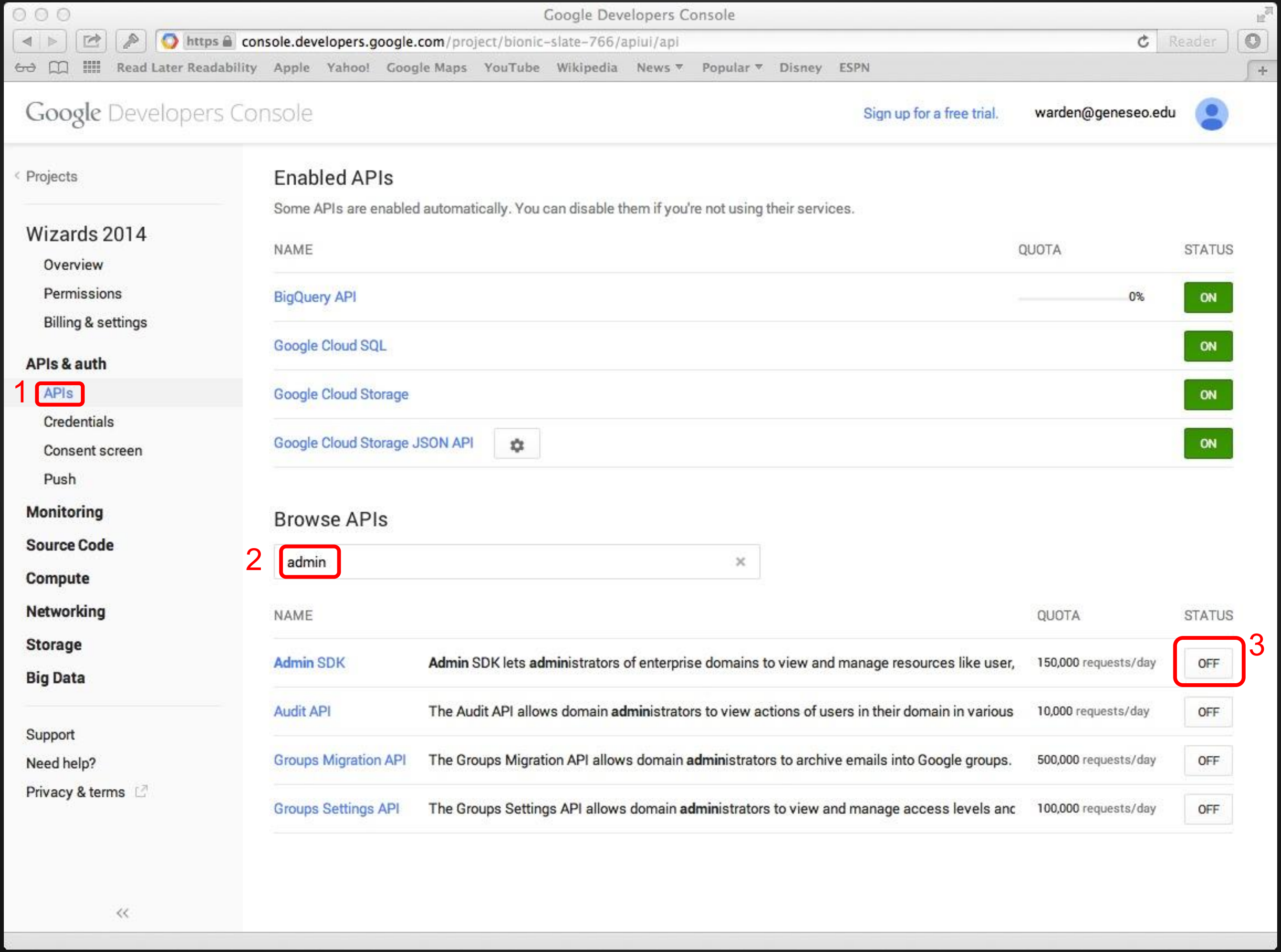
Sign up for a free trial.

warden@geneseo.edu

PROJECT NAME	PROJECT ID
GSU Group Settings	
Monthly Timesheet Generator	ethereal-aria-749

Privileged account not required!

“API Client” 2LO Step 2: Enable API(s)


Google Developers Console
https console.developers.google.com/project/bionic-slate-766/apiui/api
Read Later Readability Apple Yahoo! Google Maps YouTube Wikipedia News Popular Disney ESPN
Google Developers Console
Sign up for a free trial.
warden@geneseo.edu
Projects
Wizards 2014
Overview
Permissions
Billing & settings
APIs & auth
1 APIs
Credentials
Consent screen
Push
Monitoring
Source Code
Compute
Networking
Storage
Big Data
Support
Need help?
Privacy & terms
2
3

Enabled APIs

Some APIs are enabled automatically. You can disable them if you're not using their services.

NAME	QUOTA	STATUS
BigQuery API	0%	ON
Google Cloud SQL		ON
Google Cloud Storage		ON
Google Cloud Storage JSON API		ON

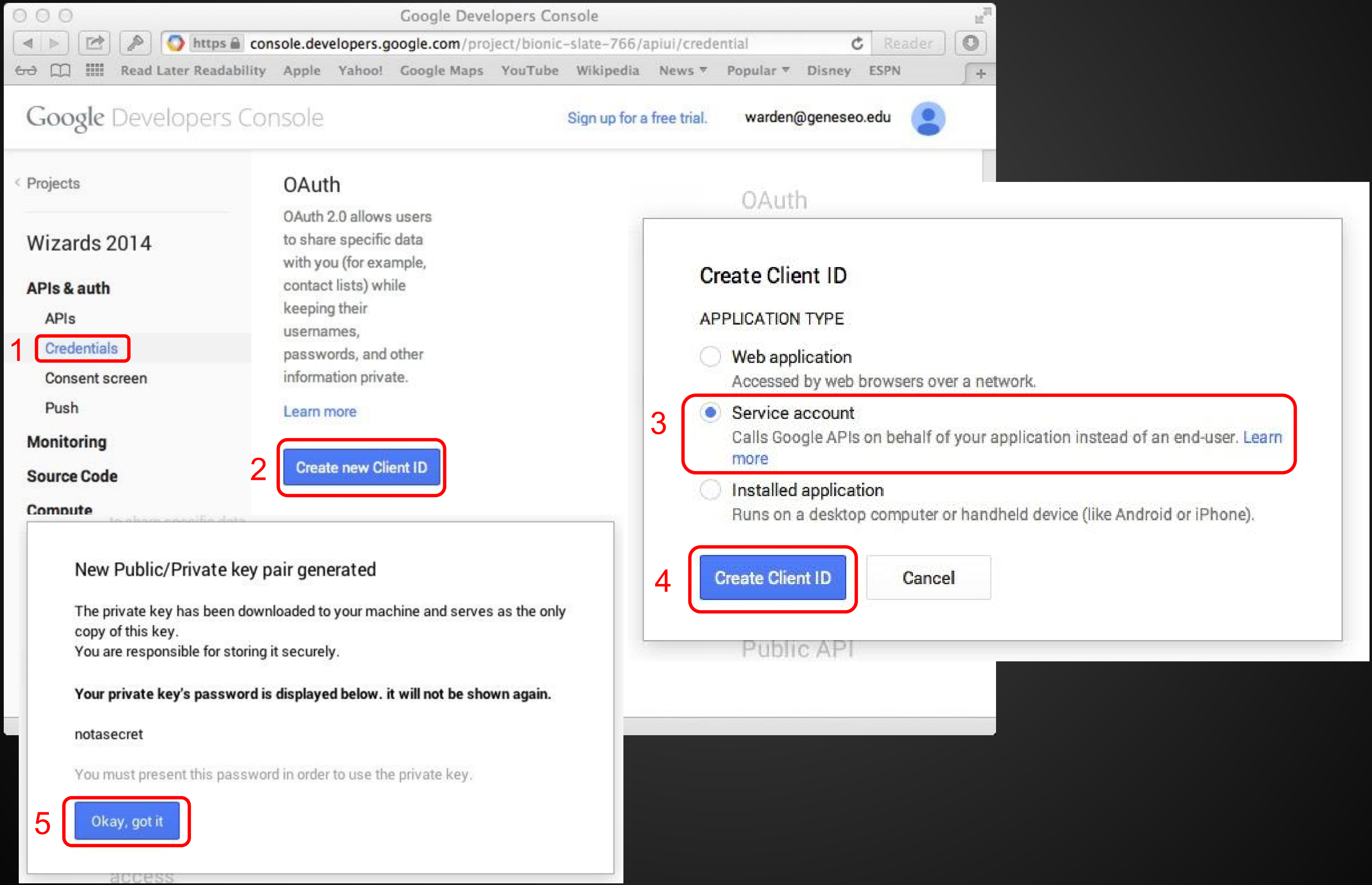
Browse APIs

admin

NAME		QUOTA	STATUS
Admin SDK	Admin SDK lets administrators of enterprise domains to view and manage resources like user,	150,000 requests/day	OFF
Audit API	The Audit API allows domain administrators to view actions of users in their domain in various	10,000 requests/day	OFF
Groups Migration API	The Groups Migration API allows domain administrators to archive emails into Google groups.	500,000 requests/day	OFF
Groups Settings API	The Groups Settings API allows domain administrators to view and manage access levels anc	100,000 requests/day	OFF

There are other interesting APIs besides Admin SDK!

"API Client" 2LO Step 3: Create Service Account


Google Developers Console
https console.developers.google.com/project/bionic-slate-766/apiui/credential
Reader
Read Later Readability Apple Yahoo! Google Maps YouTube Wikipedia News Popular Disney ESPN
Google Developers Console
Sign up for a free trial.
warden@geneseo.edu
Projects
Wizards 2014
APIs & auth
APIs
1 Credentials
Consent screen
Push
Monitoring
Source Code
Compute
2 Create new Client ID
3
4 Create Client ID
Cancel
5 Okay, got it

OAuth

OAuth 2.0 allows users to share specific data with you (for example, contact lists) while keeping their usernames, passwords, and other information private.

Learn more

Create Client ID

APPLICATION TYPE

- Web application
  Accessed by web browsers over a network.
- [✓ Service account]
  Calls Google APIs on behalf of your application instead of an end-user. Learn more
- Installed application
  Runs on a desktop computer or handheld device (like Android or iPhone).

New Public/Private key pair generated

The private key has been downloaded to your machine and serves as the only copy of this key.
You are responsible for storing it securely.

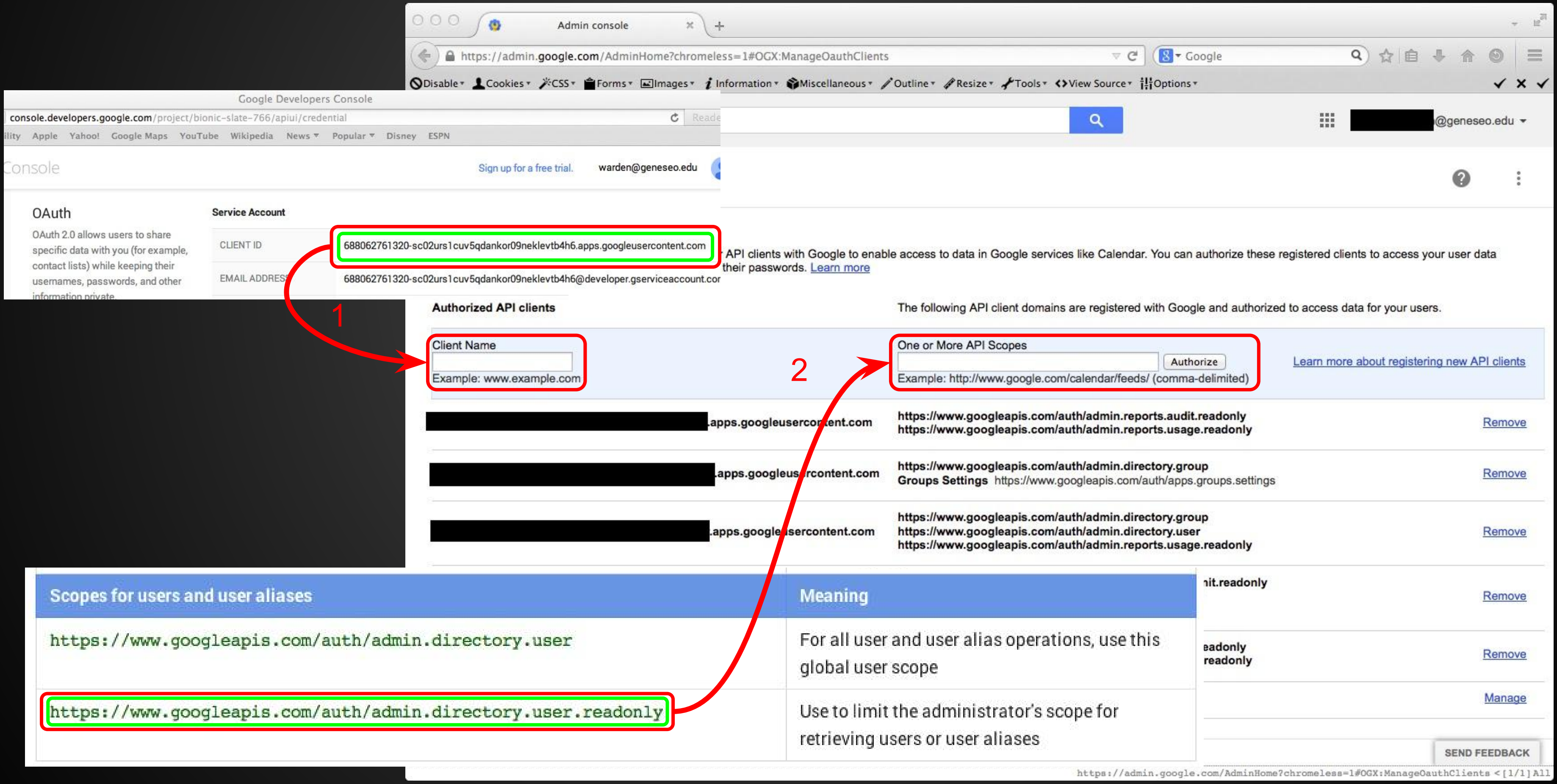
Your private key's password is displayed below. it will not be shown again.

notasecret

You must present this password in order to use the private key.

“API Client” 2LO Step 4: Enable Scopes

Google Admin Console > Security > Advanced settings > Manage API client access


Admin console
https://admin.google.com/AdminHome?chromeless=1#OGX:ManageOauthClients
Google
Disable Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools View Source Options
[redacted]@geneseo.edu
Google Developers Console
console.developers.google.com/project/bionic-slate-766/apiui/credential
Apple Yahoo! Google Maps YouTube Wikipedia News Popular Disney ESPN
Console
Sign up for a free trial.
warden@geneseo.edu
1
2
SEND FEEDBACK

OAuth

OAuth 2.0 allows users to share specific data with you (for example, contact lists) while keeping their usernames, passwords, and other information private

Service Account

CLIENT ID	688062761320-sc02urs1cuv5qdankor09neklevtb4h6.apps.googleusercontent.com
EMAIL ADDRESS	688062761320-sc02urs1cuv5qdankor09neklevtb4h6@developer.gserviceaccount.com

API clients with Google to enable access to data in Google services like Calendar. You can authorize these registered clients to access your user data their passwords. Learn more

Authorized API clients

The following API client domains are registered with Google and authorized to access data for your users.

Client Name
Example: www.example.com

One or More API Scopes [Authorize]
Example: http://www.google.com/calendar/feeds/ (comma-delimited)

Learn more about registering new API clients

[redacted].apps.googleusercontent.com	https://www.googleapis.com/auth/admin.reports.audit.readonly https://www.googleapis.com/auth/admin.reports.usage.readonly	Remove
[redacted].apps.googleusercontent.com	https://www.googleapis.com/auth/admin.directory.group Groups Settings https://www.googleapis.com/auth/apps.groups.settings	Remove
[redacted].apps.googleusercontent.com	https://www.googleapis.com/auth/admin.directory.group https://www.googleapis.com/auth/admin.directory.user https://www.googleapis.com/auth/admin.reports.usage.readonly	Remove
	it.readonly	Remove
	eadonly readonly	Remove
		Manage

Scopes for users and user aliases	Meaning
`https://www.googleapis.com/auth/admin.directory.user`	For all user and user alias operations, use this global user scope
`https://www.googleapis.com/auth/admin.directory.user.readonly`	Use to limit the administrator's scope for retrieving users or user aliases

https://admin.google.com/AdminHome?chromeless=1#OGX:ManageOauthClients <[1/1] All

“Authorize requests” in API documentation usually has available scopes

“API Client” 2LO Step 5: Return of the Meat & Potatoes

```
import httplib2
import oauth2client.client
from apiclient.discovery import build

f = file('path-to-key-file', 'rb')
serviceacct_key = f.read()
f.close()

credentials = oauth2client.client.SignedJwtAssertionCredentials(
    'service-account-email-address', serviceacct_key,
    scope = [ 'scope-previously-authorized' ],
    sub = 'gapps-privileged-user-email-address' )

http = credentials.authorize(httplib2.Http())
directory = build('admin', 'directory_v1', http=http)

print directory.users().get(userKey='gapps-user-email-address').execute()
```

GData and “API Client” Summary

Client Libraries Bookmarks:

- GData: https://developers.google.com/gdata/docs/client-libraries
- “API Client”: https://developers.google.com/discovery/libraries

For Python:

- GData: http://gdata-python-client.googlecode.com/hg/pydocs/
    - Monolithic documentation
- “API Client”: https://developers.google.com/api-client-library/python/apis/
    - Per-API documentation. Click the version link on an API in the link above, then “PyDoc reference for the X API”

That’s nice.

How about something useful?

How about pushed Google Apps logs?

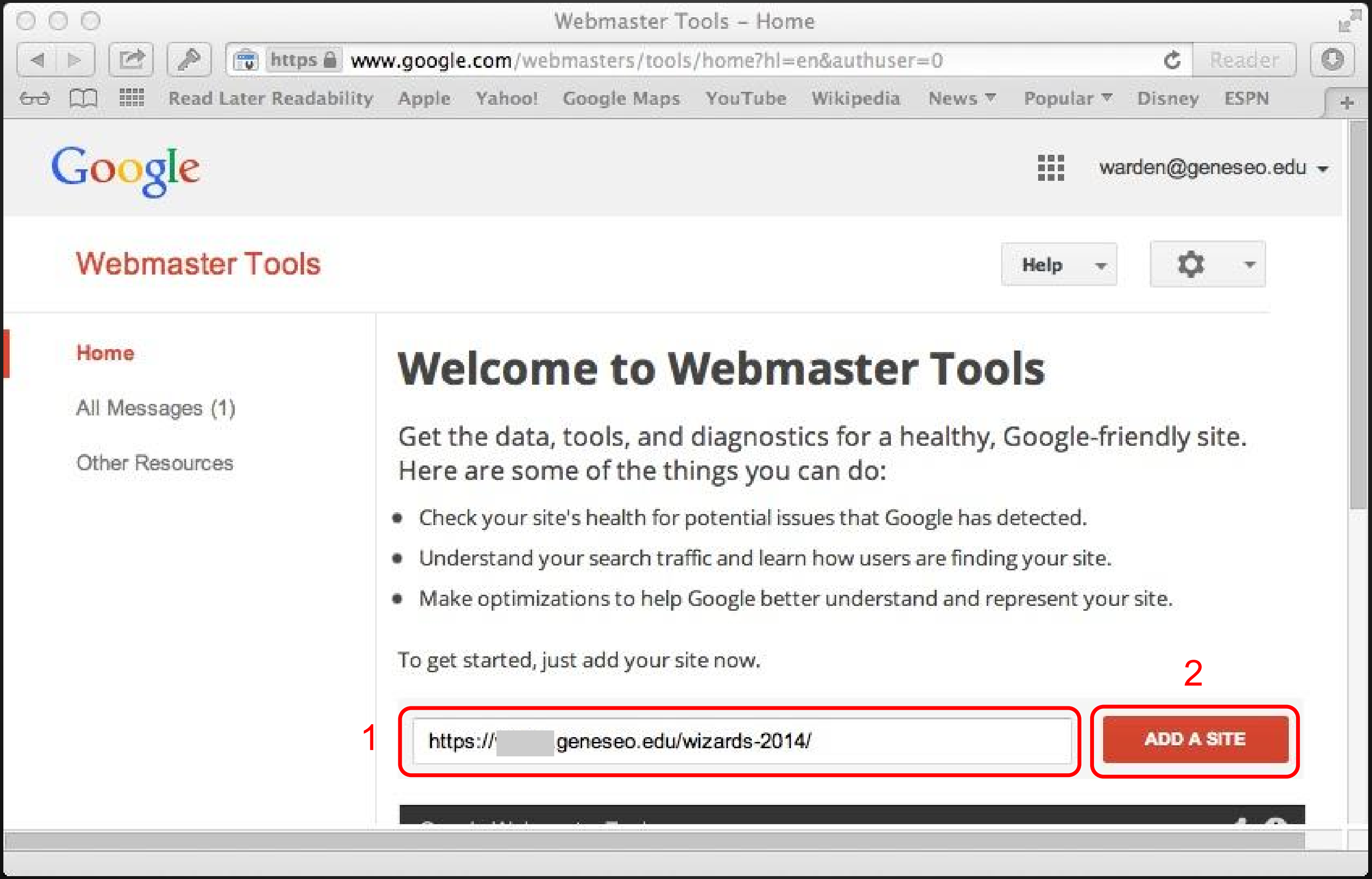
https://developers.google.com/admin-sdk/reports/v1/guides/push

Push Logs Overview

0. Follow “API Client” 2LO steps for both scopes of the Reports API in Admin SDK.
1. Verify ownership of a URL in Google Webmaster
2. Enable Push in Google Developers Console (GDC)
3. Create HTTP endpoint under URL
4. Initiate log push channel(s)
5. PROFIT

Push Logs Step 1a: Add a URL in Webmaster Tools

https://www.google.com/webmasters/tools/


Webmaster Tools – Home
https www.google.com/webmasters/tools/home?hl=en&authuser=0
Reader
Read Later Readability Apple Yahoo! Google Maps YouTube Wikipedia News Popular Disney ESPN
Google
warden@geneseo.edu
Webmaster Tools
Help
Home
All Messages (1)
Other Resources
1
2
ADD A SITE

Welcome to Webmaster Tools

Get the data, tools, and diagnostics for a healthy, Google-friendly site. Here are some of the things you can do:

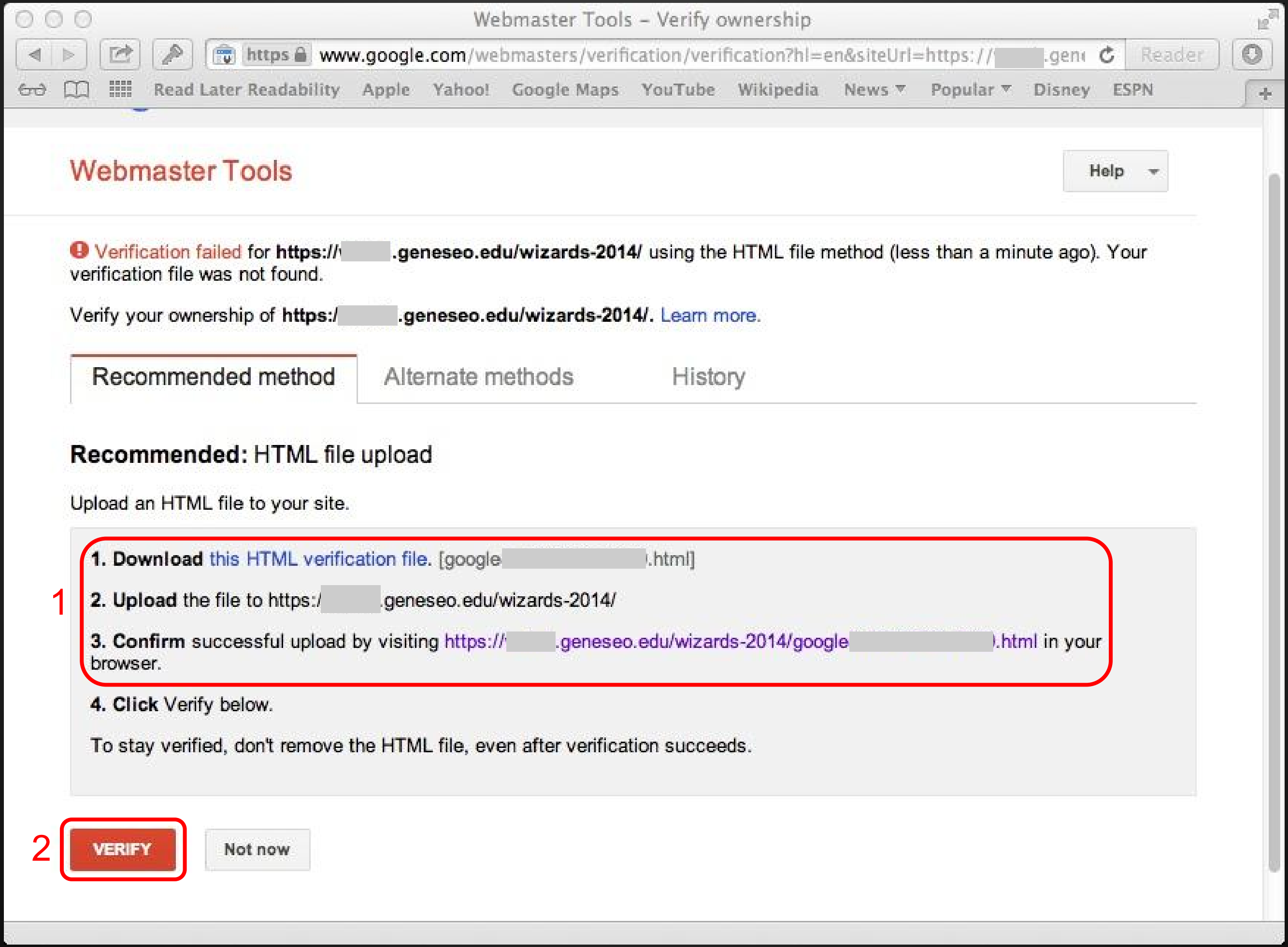
- Check your site's health for potential issues that Google has detected.
- Understand your search traffic and learn how users are finding your site.
- Make optimizations to help Google better understand and represent your site.

To get started, just add your site now.

https://[redacted]geneseo.edu/wizards-2014/

Push Logs Step 1b: Verify a URL in Webmaster Tools

https://www.google.com/webmasters/tools/


Webmaster Tools – Verify ownership
https www.google.com/webmasters/verification/verification?hl=en&siteUrl=https://[redacted].gen
Reader
Read Later Readability Apple Yahoo! Google Maps YouTube Wikipedia News Popular Disney ESPN
Help
1
2

Webmaster Tools

Verification failed for **https://[redacted].geneseo.edu/wizards-2014/** using the HTML file method (less than a minute ago). Your verification file was not found.

Verify your ownership of **https:/[redacted].geneseo.edu/wizards-2014/.** Learn more.

Recommended method | Alternate methods | History

Recommended: HTML file upload

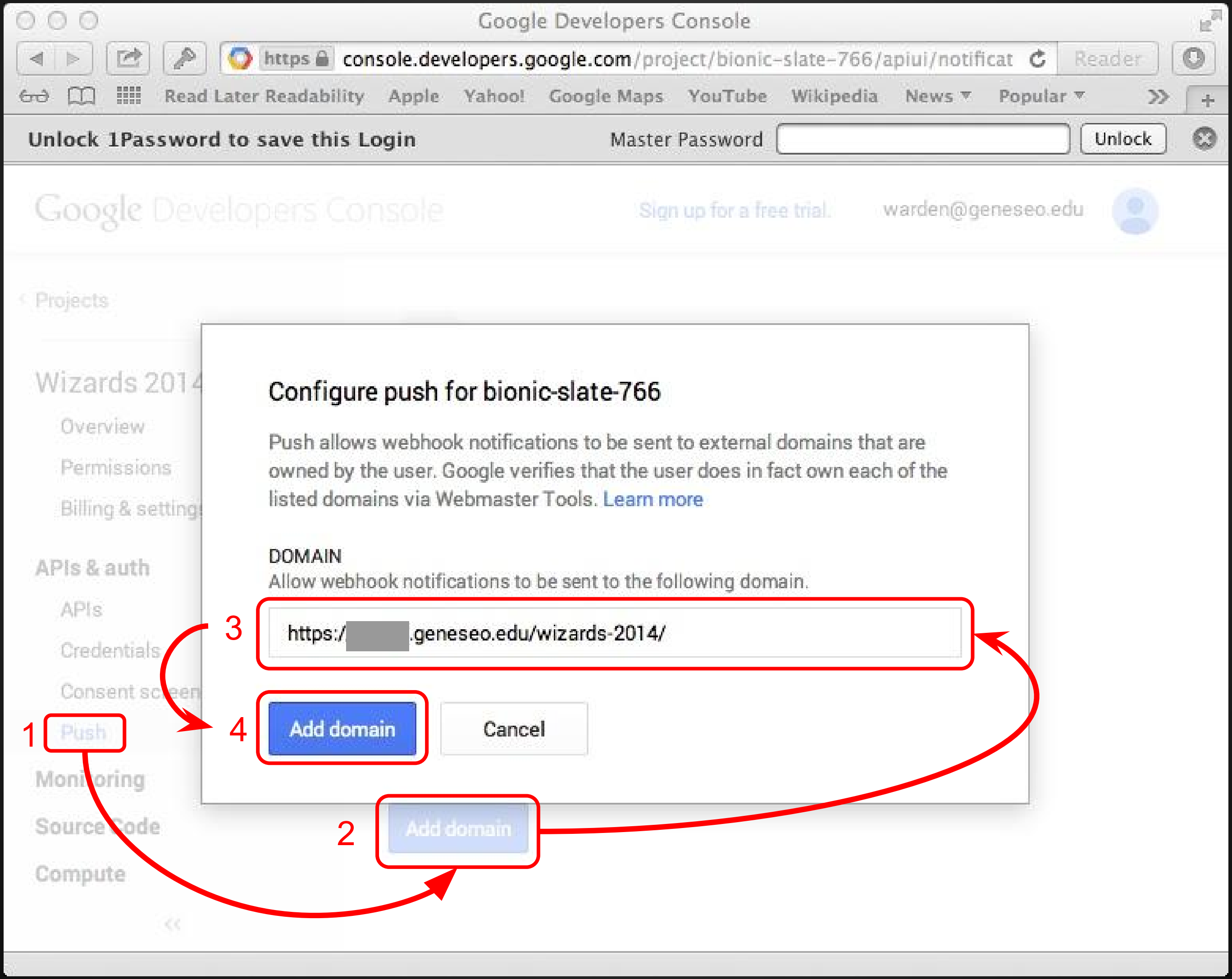
Upload an HTML file to your site.

1. **Download** this HTML verification file. [google[redacted].html]
2. **Upload** the file to https:/[redacted].geneseo.edu/wizards-2014/
3. **Confirm** successful upload by visiting https://[redacted].geneseo.edu/wizards-2014/google[redacted].html in your browser.
4. **Click** Verify below.

To stay verified, don't remove the HTML file, even after verification succeeds.

VERIFY | Not now

Push Logs Step 2: Enable Push for URL



Configure push for bionic-slate-766

Push allows webhook notifications to be sent to external domains that are owned by the user. Google verifies that the user does in fact own each of the listed domains via Webmaster Tools. Learn more

DOMAIN
Allow webhook notifications to be sent to the following domain.

https:/[redacted].geneseo.edu/wizards-2014/

Push Logs Step 3:
Script to dump HTTP requests to JSON

```php
$log_file = 'path-to-log-file';

$headers = apache_request_headers();
$body = file_get_contents('php://input');
$payload = json_decode($body, $assoc=true);
if (json_last_error() != JSON_ERROR_NONE)
     exit;

# Headers and payload have been successfully loaded in associative arrays.
# Merge, append newline and append to log file.
if (is_array($payload))
    $event = array_merge($headers, $payload);
else
    $event = $headers;
$event_string = json_encode($event) . "\n";
file_put_contents($log_file, $event_string, FILE_APPEND);
```

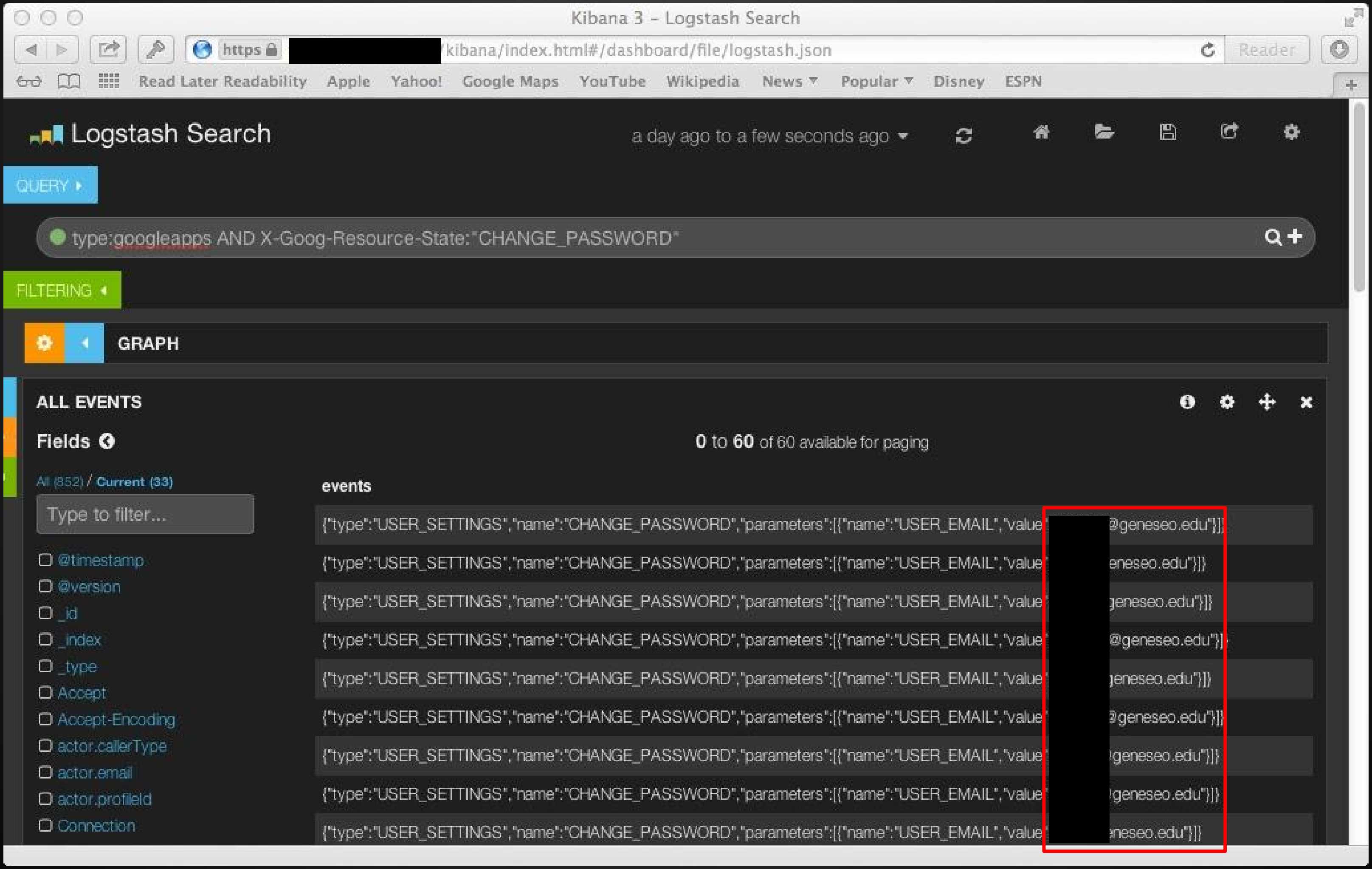
Push Logs Step 4: Init Push Channels Meat & Potatoes

```
# Not shown: “API Client 2LO” setup of http variable
reports = discovery.build('admin', 'reports_v1', http=http)
import time
from datetime import datetime, timedelta
expire = datetime.now() + timedelta(hours=6)  # Max allowed by google
expire_unix = int(time.mktime(expire.timetuple()))
expire_milliseconds = expire_unix * 1000
reports.activities().watch(userKey='all', applicationName='admin', body=dict(
     type='web_hook',
     address='https://example.geneseo.edu/wizards-2014/pushtolog.php',
     id='gsu-admin-{0}'.format(expire_unix), expiration=expire_milliseconds)).execute()
reports.activities().watch(userKey='all', applicationName='login', body=dict(
     type='web_hook',
     address='https://example.geneseo.edu/wizards-2014/pushtolog.php', id='gsu-login-
     {0}'.format(expire_unix), expiration=expire_milliseconds)).execute()
reports.activities().watch(userKey='all', applicationName='drive', body=dict(
     type='web_hook',
     address='https://example.geneseo.edu/wizards-2014/pushtolog.php', id='gsu-drive-
     {0}'.format(expire_unix), expiration=expire_milliseconds)).execute()
```

PROFIT

Sometimes Google Apps Password Sync fails.

Google password changes as they happen:


Kibana 3 - Logstash Search
https [redacted] kibana/index.html#/dashboard/file/logstash.json
Read Later Readability Apple Yahoo! Google Maps YouTube Wikipedia News Popular Disney ESPN
Logstash Search
a day ago to a few seconds ago
QUERY
FILTERING
GRAPH
ALL EVENTS
Fields
All (852) / Current (33)
Type to filter...
@timestamp
@version
_id
_index
_type
Accept
Accept-Encoding
actor.callerType
actor.email
actor.profileId
Connection

```
type:googleapps AND X-Goog-Resource-State:"CHANGE_PASSWORD"
```

0 to 60 of 60 available for paging

events

```
{"type":"USER_SETTINGS","name":"CHANGE_PASSWORD","parameters":[{"name":"USER_EMAIL","value [redacted] @geneseo.edu"}]}
{"type":"USER_SETTINGS","name":"CHANGE_PASSWORD","parameters":[{"name":"USER_EMAIL","value [redacted] eneseo.edu"}]}
{"type":"USER_SETTINGS","name":"CHANGE_PASSWORD","parameters":[{"name":"USER_EMAIL","value [redacted] geneseo.edu"}]}
{"type":"USER_SETTINGS","name":"CHANGE_PASSWORD","parameters":[{"name":"USER_EMAIL","value [redacted] @geneseo.edu"}]}
{"type":"USER_SETTINGS","name":"CHANGE_PASSWORD","parameters":[{"name":"USER_EMAIL","value [redacted] geneseo.edu"}]}
{"type":"USER_SETTINGS","name":"CHANGE_PASSWORD","parameters":[{"name":"USER_EMAIL","value [redacted] @geneseo.edu"}]}
{"type":"USER_SETTINGS","name":"CHANGE_PASSWORD","parameters":[{"name":"USER_EMAIL","value [redacted] geneseo.edu"}]}
{"type":"USER_SETTINGS","name":"CHANGE_PASSWORD","parameters":[{"name":"USER_EMAIL","value [redacted] geneseo.edu"}]}
{"type":"USER_SETTINGS","name":"CHANGE_PASSWORD","parameters":[{"name":"USER_EMAIL","value [redacted] eneseo.edu"}]}
```

It’s Not All Rainbows and Unicorns

- You need to renew those Push channels every 6 hours, which sometimes fails
- Logstash has problems fully parsing some of the deep JSON fields

One last thing...

https://github.com/dfwarden/SUNYWizards2014-GoogleAppsAPIs

Pull requests welcome!