Analyzing and Managing Risk from 3rd-party OAuth Application Access

FINAL REVISION: November 6, 2023

Jenko Hwong
Principal Threat Researcher, Netskope Inc., USA

Jenko Hwong is a Principal Researcher on Netskope's Threat Research Labs, analyzing emerging cloud threats. He has over 20 years of experience in research, product management, and engineering at companies such as Cisco and TIBCO, as well as security start-ups in markets such as vulnerability scanning, anti-virus/anti-spam appliances, penetration-testing, threat intelligence, and Active Directory security. He has successfully founded a start-up in the enterprise monitoring market and has led production deployments at enterprise customers including Walmart, Microsoft, Lucent, and Chase. He holds a B.S. in computer systems engineering from Stanford University.

Threat Research Labs
Netskope, Inc.
2445 Augustine Dr.
Santa Clara CA 95054
Phone: +1-800-979-6988
Email: jhwong@netskope.com

Abstract

Modern cloud application architectures allow users to dynamically grant 3rd-party apps access to their cloud resources. When looking at a large, anonymized dataset of over 600,000 users and their approvals of over 43,000 applications, we found that organizations often are unaware of the magnitude of the access problems:

- On average, an organization will grant 440 unique 3rd-party apps access to Google data and resources
- One organization's users approved 12,330 unique applications
- Out of all the approved applications, over 44% have been granted access to either sensitive data or all data on the user's Google Drive

The use of OAuth 2.0 has resulted in users creating numerous, unknown access paths from external 3rd-party apps to organizations' data and resources, in a way that is often hidden and unmanaged by organizations' IT or security departments. In addition, the credentials granted to access users' cloud data and resources often can be refreshed indefinitely. Security operations face challenges such as minimal to no tracking or management of these paths or credentials, along with immature or incomplete technical controls.

This paper discusses an approach to understanding, assessing, analyzing, and managing 3rd-party OAuth application risk, using anonymized real-world data as relevant examples, to provide prescriptive guidance on reducing risk associated with 3rd-party applications.

Keywords: OAuth, identity, data access, 3rd-party cloud applications

1. Introduction

In the early 2000s, Internet architectures involved users utilizing web browsers and HTML/HTTP to interact with website content and functionality. The access model involved two primary parties: the human/user agent and the website containing data/resources, pictured in Fig. 1.

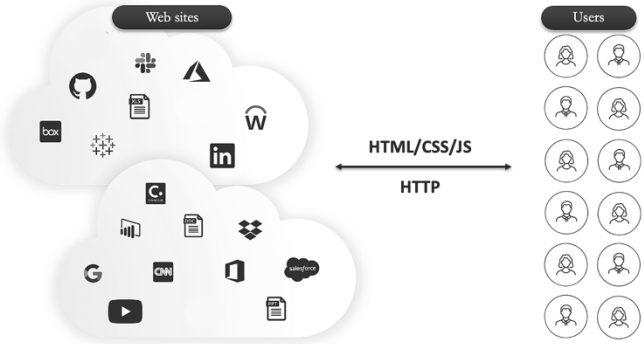


Figure 1: Early cloud / Internet architecture

Today, cloud architectures support access to cloud resources by 3rd-party apps using REST APIs and OAuth 2.0[1], which allows users to securely grant 3rd-party applications restricted access to their resources, pictured in Fig. 2.

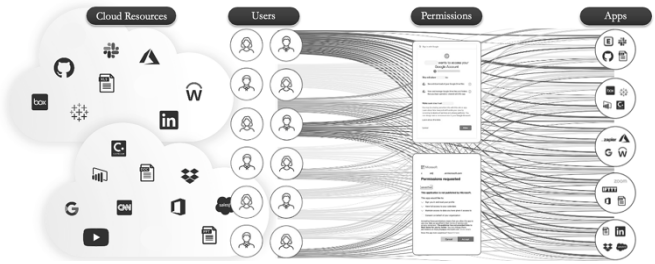


Figure 2: Modern cloud app architecture access grant model

This authorization flow results in the runtime access architecture pictured in Fig. 3.

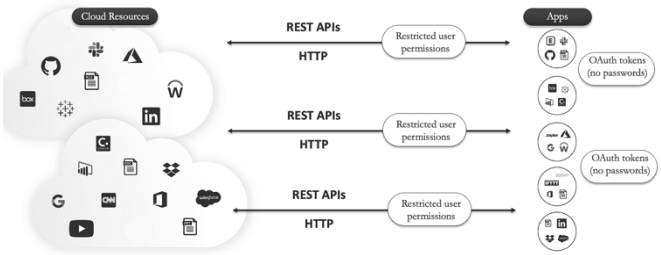


Figure 3: Modern cloud app architecture

Cloud resources can be user data (e.g., Google Drive files) or functionality (e.g., sending an Outlook email, starting an AWS EC2 instance). While access is within the scope of a user's permissions on her resources, this could include broader data e.g., "shared files" in a file-sharing cloud service. There is typically a high number of different access paths between the number of 3rd-party apps (RHS) accessing the cloud resources (LHS), with each access path reflecting some permissions to a user's resources for a particular application. There is a many-to-many relationship on the order of #Resources x #Users x #Apps.

These differences in cloud architectures are summarized in Table 1.

	Old architecture	New architecture
Backend	Website	Web application
Client	User (browser)	3rd-party application
Authorization	Full access, upon authentication	Delegated to apps with OAuth 2.0
Communication	HTML/CSS/JS (HTTP)	REST APIs/JSON (HTTP)

Table 1: Old vs new cloud architectures

The problem with allowing users to dynamically grant 3rd-party apps access to their cloud resources, is that this access is typically hidden from and unmanaged by corporate IT/security operations. When looking at a large, anonymized dataset of over 600,000 users' approvals of over 43,000 apps, we find the magnitude of this unexpected data access can be sizeable:

- Organizations granted on average 440 3rd-party apps access to their Google data and resources.
- One organization's users approved 12,330 unique applications.
- Of all approved applications, over 44% have access to either sensitive data or all data on the user's Google Drive.

Cloud architectures have changed, but security practices are lagging, and users face multiple challenges. Organizations are often unaware of the extent that their users have granted access to sensitive data. Unfortunately, application and identity vendors do not provide enough visibility into which 3rd-party application access paths have been granted, with which permissions, and to which resources. Further, security controls, best practices, and processes to manage risk from 3rd-party apps are immature or non-existent.

This paper will look to illustrate the relevance and severity of the problem through real-world, anonymized data; describe an approach to assess, measure, and prioritize risk related to 3rd-party application access; and provide guidance on how to implement and improve technical controls to reduce and mitigate risk.

2. Risk Framework

Third-party application access risk is a relatively new field and not well understood, so we will describe a basic risk analysis and measurement framework specific to this domain, particularly the use of the OAuth 2.0 protocol. We'll start with key concepts, how they map to risk factors related to threat, vulnerability, and impact, and identify metrics for those risk factors. This framework will be used in our approach to analyze, reduce, and manage 3rd-party application risk later in this paper.

2.1 Risk Measurement

Risk measurement can be effective when it is specifically adapted to a particular security domain and allows for differences in user environments. CVSS[2] is an example risk measurement framework that has been successfully used for assessing risk and prioritizing remediation efforts for vulnerability management. CVSS has metrics to reflect different threat, vulnerability, and impact factors and allows for differing user environments with metrics in its Environmental and Supplemental Metric Groups. Based on these metrics, the overall CVSS numeric score helps to prioritize remediation efforts and support CVE risk measurement and vulnerability management KPIs.

2.2 Third-Party Application Concepts

The key components of the authorization process for 3rd-party applications shown in Fig 2 are: applications, cloud resources, users, and access paths between applications and resources. The relationships are explained in more detail in Appendix B and involve users granting or approving 3rd-party application access to their cloud resources with a defined set of permissions on behalf of or as the user.

The 3rd-party applications are the "clients" that interact with the user's resources and represent the "threat" factor in a risk equation. The more applications that have access, the higher the risk and they represent a range of potential threats including malicious apps, stolen credentials from an authorized app, malicious actors who have compromised an app, and phishing attacks that spoof app identities. The cloud resources are the assets at risk or impact. Authorized access paths means that an application has access to user resources, which is indicative of vulnerability, but they can have varied permissions that have a large effect on the impact. Finally, users are the primary actors in granting access during the authorization stage, and they represent both the impact involved since it is their data and functionality at risk, as well as a threat factor since they grant or create access in the first place.

2.3 Risk Factors

With 3rd-party application access, there are multiple aspects that affect threat, vulnerability, and impact.

A high number of access paths may be due to a large number of approved applications, which increases the threat factor. The high number of access paths may also be caused by a large number of users, implying more data at risk, which reflects impact. More access paths also increase the user attack surface, which increases vulnerability.

The level of permissions associated with an access path is crucial to assessing risk. An application with write permissions on Google drive can have a *deeper* impact because it can destroy data, a concept of blast depth. This is different from an application that has access to all data on a user's Google Drive where a large *breadth* of data is accessible, conceptually the blast radius. The type of permission might also result in access to sensitive data such as PII, which results in a more damaging or *costly* impact.

In Table 2, we explicitly list the primary risk factors, risk implications, and metrics associated with 3rd-party application access. The ids were created to support easier cross-referencing throughout this article.

Id	Risk Factor	Implication	Metrics
1 NUM APPS	High number of access paths due to apps	More attack sources or vectors (↑threat)	Counts/enumeration of unique apps requesting access
2 UNKN APP	Unknown or unexpected apps	Additional attack sources (↑threat)	Counts/enumeration of unknown apps and permissions
3 NUM USERS	High number of access paths due to users	More users increase threat and attack surface (↑threat, ↑vulnerability)	Counts/enumeration of users granting access
4 ELEV PERM	Access with higher permissions	Deeper damage, blast depth (↑impact depth)	Counts/enumeration of access paths that have sensitive permission levels
5 WIDE DATA	Access to more data or functionality	Wider assets at risk, wider damage, blast radius (↑impact breadth)	Counts/enumeration of access paths with broad data access permissions
6 SENS DATA	Access to sensitive data or functionality	More valuable assets at risk, higher cost (↑impact cost)	Counts/enumeration of access paths with permissions to sensitive data or functionality
7 SENS ROLE	Access granted by users with sensitive roles or data	More valuable assets at risk, higher cost (↑impact cost)	Counts/enumeration of access paths granted by users with roles in senior management, finance, or access to PII

Table 2: Risk Factors, Implications, and Metrics

This granularity in risk factors and metrics helps us be more precise in measuring and mitigating risk. For example, if we have a compliance objective surrounding access controls for PII data, then we can focus on counting and reducing the number of scopes that grant permissions to sensitive data. On the other hand, a minimal privileges access objective might have us implement a control that reduces the scopes that grant wide access to user data such as "access to a user's Google Drive." This should improve the efficacy of security operations and support internal and compliance audits.

We now look in detail at applications, permissions, and users and their relationship to the 3rd-party application risk factors above.

2.4 Applications

We noted earlier some interesting application statistics in our dataset that we can review to see how we use application data within risk analysis:

- Organizations on average granted more than **440** total unique 3rd-party apps access to their Google data and apps
- **12,330** different apps were granted access in one organization

When measuring risk, we can start with prioritizing, sorting, and filtering based on application counts. Counts will allow us to easily create a baseline and lend themselves to prioritization and identification of outliers. This will help us with the risk factor, **1_NUM_APPS**, from Table 2.

In Table 3 we drill-down further on the top 20 organizations with >5,000 users and the number of unique 3rd-party apps that are given any kind of permissions to Google Drive, sorted by the highest apps-to-user ratio:

# Apps	# Users	Apps / User
1,218	10,243	.12
2,666	35,551	.07
422	7,597	.06
1,011	22,599	.04
430	12,068	.04
339	10,271	.03
451	16,186	.03
166	6,905	.02
353	16,200	.02
520	23,600	.02
243	15,133	.02
98	5,860	.02
331	18,804	.02
59	7,822	.01
88	8.043	.01
129	19,621	.01
253	31,529	.01
105	8,428	.01
202	13,990	.01
53	7,481	.01

Table 3: Number of Google Drive apps for large enterprises

Are the number of apps too high or too low? That depends on an organization's size, users, functions, and security policies and should be reviewed in that context.

The more important point is to understand what you have in your environment by counting the number of unique applications from your OAuth audit logs. Then steps can be taken to make improvements, e.g.

- "Last year, we had 12,330 unique apps, and we have been able to lock down the approval process and now have 2,037 unique apps."
- Creating a goal or KPI based on your organization's risk tolerance and policies based on the expected number of standard corporate apps and other approved apps.

Application counts are simple and useful to identify anomalies or common apps, but to continue to reduce risk significantly, we need to enumerate the actual apps in the environment and identify which are known, unexpected, and unknown. This will allow us to address the risk factor: **2_UNKN_APP**. Table 4 shows the application data, sorting by how often the application has been granted access i.e., the number of users.

Application	Category	%
Google Chrome	Browser	16.1
iOS Account Manager	Mobile	6.6
Zoom	Video Conferencing	5.9
Slack	Messaging	5.1
Android device	Mobile	3.9
Virtru	Encryption	3.0
Google Drive for desktop	Storage	2.9
iOS	Mobile	2.5
Atlassian	Workflow/Collaboration	2.5
macOS	Desktop	1.3
LinkedIn	Jobs	1.1
Adobe	Design	0.9
Zoom for G Suite	Video Conferencing	0.8
diagrams.net (draw.io)	Graphics	0.8
Figma	Graphics	0.7
Miro	Workflow/Collaboration	0.7
LucidChart	Graphics	0.6
Google Data Studio	Analytics	0.6
Canva	Graphics	0.5
Grammarly	Office	0.5
Confluence	Workflow	0.5
Trello	Task	0.5
Dropbox	Storage	0.5
Postman	Developer	0.4
Microsoft apps & services	Workflow/Collaboration	0.4
Pinterest	Social Media	0.4
Medium	Social Media	0.4
Windows	Desktop	0.3
Glassdoor	Jobs	0.3
Google Drive LTI by Canvas	Storage	0.3
Zendesk	Service Desk	0.3
Twitter	Social Media	0.3

Table 4: Top 3rd-Party Applications

With this application list, additional analysis can be performed to identify anomalies while categorizing apps as: expected vs unexpected vs unknown. By looking at the top-N most-approved apps, one can focus on those apps with potentially large impact.

Known/Expected Applications

Some apps may be expected and well-known, such as corporate standards like Chrome, Zoom, Slack, Atlassian. These apps can be whitelisted or pre-approved, as part of your

policy on accepted apps. Other apps may not be corporate standards, but would be expected and acceptable, such as Adobe being approved by the marketing department.

Unexpected Applications

Other apps may be recognizable but unexpected e.g., reflect more personal usage such as: Pinterest, Medium, Glassdoor, Twitter. Or perhaps may be business/job-related such as specialized, technical, or security-related apps: Postman, Virtru, Figma.

The apps may not be expected, but can be researched, understood, and either accepted or rejected. Unexpected apps should be triaged immediately into the expected category (approved), possibly be marked as follow-up with the approving users for review (in-process) or be removed (unknown/risky).

Unknown/Risky Applications

Unknowns apps are a challenge as you may only have an obscure application name and application id from your audit logs. Identity vendors do not expose any application identity metadata such as developer name, website, or verification status. If an Internet search on the unknown application name provides little to no useful information, then remove the application by removing its grant or access path i.e., invalidate OAuth access/refresh token as mentioned in Section 4.3.1. Then, if a user reauthorizes it, you can find and ask the user more information about the application and make a decision at that point.

When initially attempting an application audit, reviewing several thousand approved applications may be daunting, but it is a one-time task to categorize your current app list and typically will not take more than 1-2 days. For future app audits and review, only new apps approved since the last audit need review, which will be a much smaller list, especially if technical controls are tightened. Analyzing application counts and enumerating and categorizing applications has helped us manage the risk factors **1_NUM_APPS** and **2_UNKN_APP** from Table 2.

2.5 Permissions

Permissions or OAuth scopes are defined by the identity provider (Google in our dataset). They apply to resources and are composed of unique ids (URIs), a name, and sometimes a long description[3]:

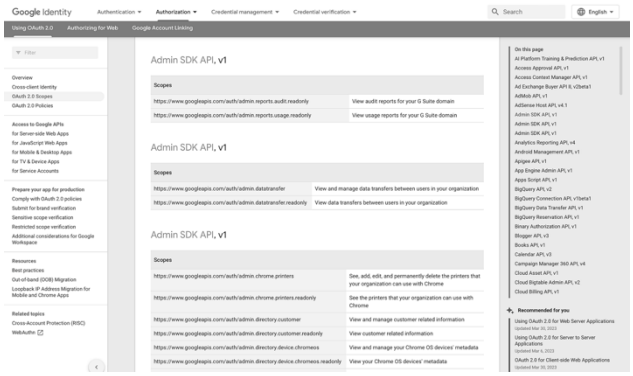


Figure 4: Google Scopes

Table 5 shows the most common scopes approved by users and requested by applications in our dataset.

Scope	# Users	# Apps
View and manage the files in your Google Drive	168,369	3,605
Manager your calendars	113,798	1,152
View and manage your mail	107,402	3,145
View and manage Google Drive files and folders that you have opened or created with this app	84,529	755
View and manage your spreadsheets in Google Drive	34,591	11,124
View and manage its own configuration data in your Google Drive	26,949	458
Manage your contacts	14,665	220
Manage mailbox labels	14,300	55
View and manage the provisioning of users on your domain	12,341	146
Manage your tasks	11,996	78

Table 5: Top permissions granted by users

Of particular interest are the permissions that are more sensitive or that grant higher levels of access. This includes scopes that involve:

- "Manage" permissions that include write access
- All files on the user's Google Drive
- Domain-wide administrator functionality on other user's data
- PII like health/student data

We'll talk about permission categorization in the next few sections, but it's important to look at the top-N lists and identify broad or administrative privileges since they pose more risk. We may find outliers of sensitive permissions that become a starting point for drill-down and further analysis e.g., the most popular permission in Table 5 is "View and manage [all] your spreadsheets in Google Drive." You certainly may want to identify which of the 11,124 applications are really required in your environment and whether they are known or not.

Permission Categories

There are often hundreds of scopes defined for large resource systems like Google or Microsoft. Analyzing risk one permission at a time is inefficient as it's often too granular. By grouping related scopes with similar risk impact into categories, we can be more effective in analysis without losing fidelity on what is causing the most risk.

The resource vendor may provide some categorization, and in Google's case, they identify scopes as "sensitive" or "restricted." They use it in the context of developing and publishing public 3rd-party apps whereby developers must pass more prerequisite checks if their app is requesting more sensitive permissions. Analyzing Google sensitive or restricted permissions will help us assess the risk factors **4_ELEV_PERM** and **6_SENS_DATA** from Table 2.

For this analysis, we propose three additional categorizations to assist in more granularity during analysis:

1) Google API/Service[3]

   Google Drive, Chat, Gmail, Fitness, Google Cloud Platform, etc.

2) Breadth of data access for a particular scope, as defined in Table 6, which helps us analyze the risk factor **5_WIDE_DATA**.

Data Access	Implication for 3rd-party app access
Application Data Only	can only read/write data it has created. Most restrictive.
All Data of Type X	can access common file formats e.g., Google Docs or image files, even if created by other apps
Sensitive Data	can access health data, data about minors/students, or other PII
All Data on Storage	can access all data on Google Drive, Google Cloud Bucket, Database, etc.

Table 6: Permission Categorization by Data Access

3) Level of elevated access provided by the permission, as defined in Table 7, which helps us analyze the risk factor **6_SENS_DATA**.

Permission Level	Implication for 3rd-party app access
Read-only	can only read/list data
Write	can update/modify/delete data
Administrator	can read/write/update/delete data or configurations for other users, across the domain, subscription, project, etc.

Table 7: Permission Categorization by Permission Level

To use the above in risk analysis, it would be nice if Google or identity providers exposed a machine-readable form of scopes and their definitions sensitive/restricted, but they do not. Their documentation is incomplete as well (no listing of sensitive scopes).

By testing and through manual efforts, we have created a listing of all Google scopes, their Google API/service mappings, their Google sensitive/restricted categorization, as well as our additional data access and permission level mappings at:

https://github.com/netskopeoss/oauth/google/google_oauth_scopes.csv
https://github.com/netskopeoss/oauth/google/google_oauth_api2scope.csv

Google API/Service Category

There are hundreds of permissions or OAuth scopes provided by most identity/resource providers, including Google. Google documents the APIs/services but most can also be derived from the URI id namespace , as listed in Table 8.

Scope	Description
https://www.googleapis.com/auth/drive	See, edit, create, and delete all of your Google Drive files
https://www.googleapis.com/auth/drive.appdata	See, create, and delete its own configuration data in your Google Drive
https://www.googleapis.com/auth/drive.file	See, edit, create, and delete only the specific Google Drive files you use with this app
https://www.googleapis.com/auth/drive.metadata	View and manage metadata of files in your Google Drive
https://www.googleapis.com/auth/drive.metadata.readonly	See information about your Google Drive files
https://www.googleapis.com/auth/drive.photos.readonly	View the photos, videos and albums in your Google Photos
https://www.googleapis.com/auth/drive.readonly	See and download all your Google Drive files
https://www.googleapis.com/auth/drive.scripts	Modify your Google Apps Script scripts' behavior

Table 8: Google Drive API, v3 OAuth Scopes[3]

When analyzing permission data, it makes sense to first break-down datasets by API/service or similar grouping. For Google, this means we would look at apps with access to: Google Drive, Gmail, Fitness, etc. one at a time. Sometimes, there is one specific OAuth scope that maps to an API or service, such as https://www.googleapis.com/auth/cloud-platform mapping to Google Cloud Platform access. After looking at these subsets one-by-one, it is then useful to drill down further using the two additional categories: data scope and permission level.

Data Scope Category

Table 9 lists the 43,549 unique applications in our dataset, broken down by data scope:

Data Scope	# Apps	%
Application Data Only	684	1.57
All Data of Type X	27,986	64.26
Sensitive Data	10,764	24.72
All Data on Storage Device	9,202	21.13

Table 9: Permission Data Access Breakdown

All Data of Type X is the most popular data access category and includes apps that access an open shared file format such as read/write of Google Sheets files e.g., budgeting or financial planning utilities.

What may be more surprising and warrant attention are the Sensitive Data (health data, student data, PII) and All Data on

Storage Device (e.g., access to all user files on Google Drive), which have been requested by 21% to 25% of the apps.

The fewest apps are found in the most restrictive category of Application Data Only, which means the 3rd-party app can only access data it generates, like the mobile app sandboxes on the iPhone.

How does this all help? We have an ability to quickly assess overall impact and prioritize efforts and determine whether it's useful to spend time on specific apps within any of these categories. The answer from the above data is: a lot of elevated privileges are granted to apps, we should start with All Data and Sensitive Data access, form baselines and determine our current exposure. Over time, we can set KPIs and work on incremental improvements in reducing the number of apps with sensitive/broad permissions.

Permission Level Category

Let's look at the same permission data but categorized by the permission level i.e., read-only, write, or admin-level, in Table 10.

Permission Level	# Apps	%
Read-only	400	0.92
Write	43,005	98.75
Administrator	1,113	2.56

Table 10: Application Permission Levels

What immediately stands out is that almost 99% of apps that have been approved in this environment have write privileges of some kind. In any security context, this would be a priority for further analysis. On the one hand, this might include more expected functionality such as creating files on Google Drive or sending emails via Gmail. On the other hand, it means that most 3rd-party applications can modify data in some way in the environment.

Category Intersections

Application counts using only one category may not be granular enough, so it is usually more useful to use the intersection of multiple categories to zero in on more specific permissions. Th ere are several examples from our dataset:

1) Apps that have write permissions (level) on Google Drive (API/service) only, show in Table 11.

Data Scope	# Apps with Write Permission Level	%
All Data of Type X	27,915	64.10
Sensitive Data	7,752	17.80
All Data on Google Drive	1,113	17.49

Table 11: Data Scope Breakdown of Apps with Write on GDrive

2) 25% of apps have write permissions (level) on spreadsheets (subset of All Data of Type X) on Google Drive (API/service)

Permission Details

We can now drill-down further and look at the specific permissions and applications within these categories or category intersections.

Table 12 shows the apps that have write permissions (level) on spreadsheets (subset of All Data of Type X) on Google Drive (API/service):

Scope	Application
View and manage your spreadsheets in Google Drive	Slack
View and manage your spreadsheets in Google Drive	Draw.io
View and manage your spreadsheets in Google Drive	LucidChart
View and manage your spreadsheets in Google Drive	Zendesk
View and manage your spreadsheets in Google Drive	Asana
View and manage your spreadsheets in Google Drive	Doordash
View and manage your spreadsheets in Google Drive	Untitled project
View and manage your spreadsheets in Google Drive	Zapier
View and manage your spreadsheets in Google Drive	Project Default Service Account
View and manage your spreadsheets in Google Drive	Titus
View and manage your spreadsheets in Google Drive	JIRA
View and manage your spreadsheets in Google Drive	Awesome Table
View and manage your spreadsheets in Google Drive	SurveyMonkey
View and manage your spreadsheets in Google Drive	MITRA
View and manage your spreadsheets in Google Drive	Yet Another Mail Merge
View and manage your spreadsheets in Google Drive	Google APIs Explorer
View and manage your spreadsheets in Google Drive	Google Marketing Platform
View and manage your spreadsheets in Google Drive	Quickstart
View and manage your spreadsheets in Google Drive	Remove Duplicates
View and manage your spreadsheets in Google Drive	HubSpot

Table 12: Apps with Write Permissions on All Spreadsheets in GDrive

Some apps might be unexpected or unknown such as Doordash, Untitled project, Zapier, Quickstart, Remove Duplicates, and we could take action to remove access paths or whitelist the apps.

2.6 Users

Using a user's role, department, or job function can help categorize their data as more or less valuable. For example, a financial employee, a senior executive, or a customer service rep may have access to highly sensitive PII or company-confidential information. This involves the risk factor, **7_SENS_ROLE**, an impact factor. Typically, we can analyze this risk factor by using directory service role/group/department attributes with the user in the authorization audit events to group and aggregate the data by user role, department, etc.

We should also determine if there are users who are introducing more risk i.e., approving apps that are riskier. We can review top-N users sorted by approved app counts and see if any users standout. We might choose to apply this to user groups (as defined within the corporate directory service) to see if job function drives approval of certain apps. If there are any anomalies, we want to determine if any risk can be reduced (training, restriction of privileges) or if it is part of job function (administrators), and at least make the risk known and managed. For example, we might decide on one corporate

standard app for job role function XYZ and whitelist it. This affects the threat risk factor, **3_NUM_USERS** from Table 2.

The more users approving access paths for the same app would intuitively imply that those apps carry higher risk as more user data is vulnerable. However, when it is a very high majority of an organization's users, these apps tend to be well-known apps that are part of corporate standards. The more nefarious or harder to find risk areas are the small one-off apps used by one or very small percentage of users. This relates to the risk factor: **2_UNKN_APP**.

Here are two real examples from our dataset, discovered through review of unrecognized apps with highly sensitive permissions but in use and approved by a very small number of users.

1) GSuite Alerting Tool

This application had administrator/domain permissions (level) on sensitive data (data scope) across multiple APIs/services. This is a combination of multiple risk factors: **2_UNKN_APP**, **3_NUM_USERS** (low not high), **4_ELEV_PERM**, **5_WIDE_DATA**, and **6_SENS_DATA**.

Manage data access permissions for users on your domain
Manage delegated admin roles for your domain
Manage messages in groups on your domain
Manage the list of sites and domains you control
Manage your Google Classroom class rosters
Manage your Google Classroom classes
Manage your calendars
Manage your contacts
View and manage Google Apps licenses for your domain
View and manage customer related information
View and manage data transfers between users in your organization
View and manage organization units on your domain
View and manage the provisioning of calendar resources on your domain
View and manage the provisioning of domains for your customers
View and manage the provisioning of groups on your domain
View and manage the provisioning of user schemas on your domain
View and manage the provisioning of users on your domain
View and manage the settings of a Google Apps Group
View and manage your Chrome OS devices' metadata
View and manage your mobile devices' metadata
View audit reports of Google Apps for your domain
View the email addresses of people in your classes
View the profile photos of people in your classes
View usage reports of Google Apps for your domain
View your basic profile info
View your data in Google Cloud Storage
View your email address

Figure 5: Unknown Administrative Tool

The app name hinted at administrative alerting functionality and that it was an internally developed app. It would warrant further investigation to determine if it's a valid internal tool and whether controls have been implemented to ensure only approved administrators can use it.

2) CamScanner

This application requested write permissions on all files on Google Drive. This is common; however, it was used by only a used by 0.04% of users and is a public app (but not so common in our dataset). This was a combination of the risk factors: **2_UNKN_APP**, **3_NUM_USERS** (low not high), **4_ELEV_PERM**, **5_WIDE_DATA**. It turns out that CamScanner is a scanning/image app that was:

- Found by Kaspersky in August 2019 to contain malware[4]
- Banned by the Indian government over security concerns in June 2020[5]

Not only does CamScanner have broad privileges, it has been compromised with malware and banned by a government. In this case, we would remove all of its access, and if the business purpose (scanning/image editing) was legitimate, find alternatives and set a corporate standard that can be whitelisted/pre-approved. This is an example where surgical analysis, looking at highly sensitive apps and permissions that are not widely known or used, can identity high risk applications that need remediation.

3. Risk Analysis

We now formalize an approach to applying the above concepts in a structured manner to analyze risk in your environment.

3.1 Understand Risk

An application inventory or a baseline of 3rd-party apps that have access to your environment is crucial to support investigations/forensics, provide software inventory of 3rd-party apps (compliance), and to alert on 3rd-party app approvals and changes (new app, different app, different app id, etc.).

To get started, export OAuth log events from the last 6 months via the Admin Console (e.g., Google Workspace) in .CSV format. Alternatively, you can use API scripts and a more structured data store (RDBMS) (see Appendix A.2.).

Using the OAuth log events, you can now calculate basic application statistics including counts, averages, and aggregations.

With these basic stats, review your application inventory using the counts to sort and rank to prioritize efforts, starting with the most-approved apps. You'll generally be looking to assess the inventory for any strange anomalies that jump out to you. Our goal is to create two deliverables:

- Create a baseline of known 3rd-party applications and their metadata (id, name, scopes). This could be as simple as an Excel spreadsheet with appropriate columns for application metadata and various count metrics. This addresses the risk factor, **1_NUM_APPS**.

- Track unknown apps within this spreadsheet, for further analysis. Your goal with the unknown apps will be to reduce them to zero (accept them as known and valid or reject them as suspicious or unknown) by investigating and iterating in later steps. This addresses the risk factor: **2_UNKN_APP**.

3.2 Reduce Risk

The primary goal will be to reduce risk by both reducing the number of 3rd-party apps and number of unknown apps that have access. The priority of efforts will be driven by analysis based on their permissions.

We will proceed with several steps: categorize apps to reduce the unknowns, measure/analyze the app data from a permission-based viewpoint, then reduce the size of the app list. The steps in this section will continue to help mitigate risk factors: **1_NUM_APPS** and **2_UNKN_APP**.

3.2.1 Categorize Apps

We will start by categorizing the app inventory into known, unknown, and unexpected buckets. Some of this becomes apparent simply by review of the application list, starting with obvious expected apps, then iterating on the remaining unknowns.

Prioritize which apps to start with by categorizing apps into known/valid, known/unexpected, unknown categories (Section 2.4). By counting applications, we can look at the top-N apps and top-N app categories.

If the number of apps is too high to work through or you have a lot of unused legacy apps access paths in existence remove all 3rd-party app connections and start over; nothing will be lost, and users will be forced to reauthenticate and reauthorize app access. However, you should consider locking down or restricting the approval process and set session timeouts to prevent # apps from increasing while you assess. This is described in Section 4.1 Prevention in more detail.

Our end-goal for this step is to revise our baseline of 3rd-party applications by categorizing each app as:

- Known (expected, approved corporate standards, and allowed user apps, low risk)
- Unexpected (app creator is known, but unexpected usage, medium risk)
- Unknown (app and creator are not known or trusted, high risk)

3.2.2 Measure/Analyze

We can further refine the application list by using permissions requested by the applications. This helps focus efforts on the "riskiest" or most potentially damaging apps.

Using some of the techniques in Section 2.5 Permissions, we can focus on application subsets based on their granted permissions. It will generally be more useful to look at the top-N apps based on intersection of permission categories rather than individual permissions. This helps us with the risk factors: **4_ELEV_PERM**, **5_WIDE_DATA**, and **6_SENS_DATA.**

We can start with individual APIs/services e.g., Google Drive and drill down into each API/service to see if there is a concentration of risk or unexpected/unknowns. We do this by looking at which apps have more sensitive data scopes e.g., Read/Write All Files in Google Drive coupled with permission levels e.g., Write or Admin.

We can also look at which applications have been granted by the majority or just a few users (Section 2.6 Users). We want to find the users that are introducing disproportionate amount of risk from an approval of a high # of apps or apps with highly sensitive permissions (data scope x level). We also want to identify apps used by a low number of users that have an unknown/high risk with sensitive permissions.

Our goals are to:

- Refine the application baseline using sub-groupings based on application metrics: for each API/service category, we look at the intersections of sensitive data scopes and permission levels.
- Create risk reports that identify the top-N apps w/ sensitive permissions, top-N sensitive data scopes, and the top-N sensitive permissions

3.2.3 Reduce

Reducing risk primarily involves reducing the total number of apps as well as reducing the unknown apps. Continue iterating using techniques from the previous sections to analyze the remaining unknown apps to categorize them as known (ok) or unexpected (needs further investigation).

In addition to categorization, we must enforce the categorization with technical controls, which will depend upon the identity system (e.g., Google Workspace). Generally, major identity systems support whitelisting or pre-approving/pre-installing known apps or removing access if they are disallowed.

Recategorizing the unknown apps will reduce the total number of apps. Be sure to also remove duplicate apps that have similar/overlapping functionality, by standardizing apps.

By the end of this step, you should have a revised application baseline with a reduced # of total apps, reduced # unknown apps, enforced with an application whitelist or pre-installed list within the identity system.

4. Risk Management

To manage ongoing 3rd-party app risk, we cover prescriptive guidance for operational control areas to prevent, detect, and remediate abuse. These guidelines are recommendations that should be customized based on the risk tolerance of an organization as well as what is feasible based on budget, resources, and end-user requirements.

4.1 Prevention

4.1.1 Restrict Approval Process

If possible, restrict the ability for users to grant 3rd-party access, as it addresses the root cause of 3rd-party application risk i.e., allowing end-users to control access. Most identity systems, including Google Workspace and Microsoft Azure, provide multiple options depending upon how controlled of a process you want.

The most restrictive option would be to only allow approvals by administrators. Less restrictive would be an approval process that allows users to approve but requires an administrator for final approval; some vendors support an approval workflow.

These approaches could be supplemented by requiring users to file a ticket justifying the use of the 3rd-party application. Continuing to audit approvals from the audit logs and alerting the administrators about new app approvals is another technique to not block users but add an extra layer of security oversight.

Restricting and controlling the approval process should reduce the number of 3rd-party apps overall as well as the unknown and unexpected, helping address the risk factors: **1_NUM_APPS** and **2_UNKN_APP**.

4.1.2 Implement Approved App List / Pre-Installed Apps

An approved app list (whitelist) or pre-approved applications can complement the approval process. An organization could pre-approve/install all corporate standards or approved apps to make it easy but have a tighter approval process for apps not on the list e.g., require administrator approval or a ticket. This app list will usually match the expected applications from the application inventory baseline discussed in Section 3.1 Understand Risk. Some identity vendors have you specify applications explicitly or allow you to restrict it to verified or market-place apps, both help. A pre-approved or pre-installed app list will help reduce the number of 3rd-party apps overall as well as the unknown and unexpected, helping address the risk factors: **1_NUM_APPS** and **2_UNKN_APP**.

4.1.3 Set Session / Credential Timeouts

One general risk factor is lost or stolen credentials. With OAuth 2.0, 3rd-party applications receive temporary OAuth tokens for access and these tokens can be compromised by malware, malicious apps, or stolen with phishing attacks. This risk factor should be addressed by using available controls to set session or credential timeouts. This will mitigate compromised credentials by forcing an expiration.

This may cause pain (users must reauthenticate/reauthorize 3rd-party apps), so set these based on environment, sensitivity of accessed data and user role. Example: production environments might have a short time of 1-2 hours. Corporate/office environments might have a longer timeout of 8 hours.

Not all vendors support a timeout. For example, Google does, Microsoft does not. Finding the setting can also be difficult e.g., the Google Cloud session timeout setting is in the Workspace Admin Console NOT the GCP Admin Console.

4.1.4 Mitigate Compromised Credentials

There are other measures that mitigate compromised credentials such as client policies that constrain the use of credentials e.g., only approved IP addresses can use credentials or credentials can only be used from managed such as IT-approved devices running anti-virus software. To this end, Google has VPC Service Controls and Microsoft has conditional access policies.

4.2 Detection

4.2.1 Vendor Behavioral Detection (M/L)

Detection is hard. If a badly behaving app (or bad actor) accesses corporate data via one of these approved OAuth access grants, then it can be a behavioral detection problem. Microsoft is starting to provide some OAuth app detection use cases in Cloud Defender. Enable a vendor's monitoring solution, if available.

4.2.2 Detect Unknown Apps

In Section 4.1.2, we used the known apps from the application inventory as a preventative control. There is additional value in using the same list as a detection control which will help with the risk factor: **2_UNKN_APP**. We can continuously check the approved 3rd-party applications from our identity audit logs against the known app list from our application inventory and alert on anything that is unexpected. This allows us to double-check our preventative controls and ensure that nothing bypasses it.

If real-time detection is too difficult, implement it as a daily or weekly audit job that compares recent application approvals (that have occurred since the last check) against the expected, known applications from the application inventory.

4.2.3 Correlate App Authorizations and User Activity

So far, we have been assessing the risk from the granting or approval process, which reflects authorization of access but not necessarily whether any application uses that access. There is value in correlating real application activity because it allows us to refine our risk assessment and to identify unnecessary or unused access paths.

If we look at real user activity events from application audit logs, we can identify actual API calls from 3rd-party applications to cloud resources. You will likely need to look at the src IP, since the client app is usually not logged during usage. Assess a higher risk when actual activity occurs with an app grant, like when a vulnerability is actual exploited, or an open port has traffic. Any real abuse activity of access paths would be related to the risk factors, **3_NUM_USERS** and **7_SENS_ROLE**, identifying additional risk with the users that authorized the access in the first place.

We can also revoke unused apps which have no activity and can revoke/delete their access (this becomes a preventative measure).

4.3 Remediation

For remediation, the goal is to have clear and standardized operational runbooks. Vendor measures for revoking access can be complicated and vary by vendor.

4.3.1 Revoke Access when credentials are compromised

Revoking access is the most important remediation action to define, review, test, and operationalize, and this involves some combination of managing the primary user credentials, the OAuth access and refresh tokens, and potentially other user accounts if SSO is being used.

Vendor functionality differs greatly with respect to revoking the two OAuth tokens. Google provides a per user API, or a user can self-serve via the Workspace Console. Microsoft provides an API to revoke the OAuth refresh token but not the access token. When the OAuth token revocation is incomplete in the case of Microsoft, the guidance is to revoke the refresh token then wait for the access token to timeout (60 minutes by default).

Additionally, primary user credentials will need to be reset and changed and ensure to revoke credentials or accounts that are synchronized across systems (SSO, external directory services).

4.3.2 Test, Test, and Retest

OAuth token revocation functionality varies greatly across vendors, so testing is paramount. Update ops runbooks after you test and retest all procedures every six months because vendors update or change capabilities.

5. Conclusion

We have discussed aspects of 3rd-party app risk as well as techniques to understand and analyze the risk: collect an app inventory for an initial baseline, categorize it to reduce the number of apps and unknowns, and use permissions-analysis to further refine the app list. The inventory can be formed by using your identity provider console, APIs, or example scripts. This baseline of applications and associated permissions, and user data can be kept in a spreadsheet or in a reporting tool or database (at a minimum), using the audited data.

To manage and reduce risk, we described some control areas to improve or implement for prevention, detection, and remediation. Implement better preventative controls by enforcing session timeouts, using domain-wide/admin-created approved app lists, and by moving away from a user-only approval process. Detection measures can be improved by whitelisting known apps and alerting on unrecognized apps. If available, enable vendor-supplied behavioral detection (M/L), and start to correlate app authorizations/approvals (OAuth) with user activity with those apps to identify active access paths and to revoke unused application access.

This is a base plan that should be adjusted to meet an organization's risk tolerance, security priorities, and timelines. Regardless of the pace, continuous iteration will result in incremental improvement.

Appendix A: Datasets and Methodology

A.1 Referenced Datasets

Example data referenced in this paper is from anonymized and aggregated real-world data about OAuth approvals, collected by the Netskope Security platform with customer approval. Data was composed of OAuth app approvals for 3rd-party app access to Google resources as gathered from Google Workspace Admin audit logs. This involved production data, covering over 600,000 users in thousands of organizations of all segments, verticals, geographies over a 3-month period collected in 1H of 2023.

A.2 Collecting Data from your Environment

For readers who wish to apply some of the practices to their environment, similar data can be collected by accessing the audit logs of the identity provider used by the resource manager. Typically, audit events can be access from both the administrative console, as well as via API:

For example, in the Google Workspace Console, OAuth audit events can be found under Admin > Reporting > Audit and investigation > OAuth log events:

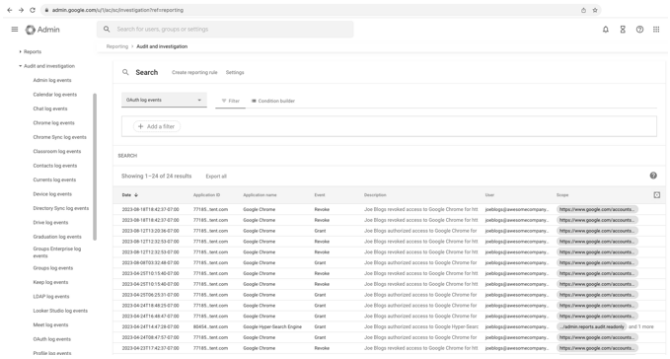
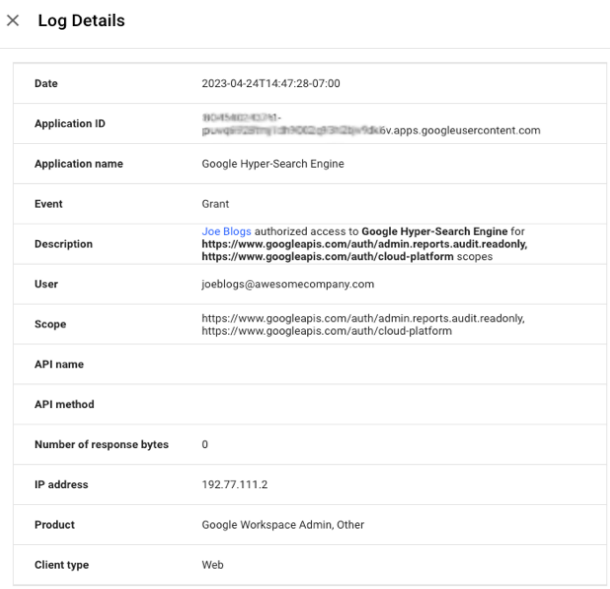


Figure 6: Google Workspace Console (OAuth Logs)



Log Details

Date	2023-04-24T14:47:28-07:00
Application ID	[redacted]-puvq6928tmj1dh9002q93h2bjv9dki6v.apps.googleusercontent.com
Application name	Google Hyper-Search Engine
Event	Grant
Description	Joe Blogs authorized access to **Google Hyper-Search Engine** for **https://www.googleapis.com/auth/admin.reports.audit.readonly, https://www.googleapis.com/auth/cloud-platform** scopes
User	joeblogs@awesomecompany.com
Scope	https://www.googleapis.com/auth/admin.reports.audit.readonly, https://www.googleapis.com/auth/cloud-platform
API name	
API method	
Number of response bytes	0
IP address	192.77.111.2
Product	Google Workspace Admin, Other
Client type	Web

Figure 7: Google Workspace Console (OAuth Log Details)

Conveniently, the Workspace Console can export the log events to Google Sheets or .CSV format for analysis.

There is a corresponding Workspace API call that retrieves the same information from the audit using the API endpoint, https://admin.googleapis.com/admin/reports/v1/activity/users/all /applications/token:

```
$ curl -s -H "Authorization: Bearer ya29.a0Ael9sCPyjjVW4G0KngGDC7C4erWrvNIgnmuptaES7cP-dR9U0aooM5-
CGG0llLFvH3LS5foRPTQJ3zxJAWnn4DWK4_nMG_BzuvfE3a_YhelC1lY1wB49sBupXQfhS-xYWuB9ivtpLy9n-
DzfPEJH9siLbxuxaCgYKATgSARESFQF4udJhf67qCxyvkeEdxRWX6TGAPA0163"
https://admin.googleapis.com/admin/reports/v1/activity/users/all/applications/token?maxResults=25

{ "kind": "admin#reports#activities",
  "etag": "\"P4UZt3E47YE1EedEgRGeNMKVdI4cxfy8A0aTCl4A_NM/RFrt6Zz39_-vvHTeIkk-S0BNqtc\"",
  "items": [
    { "kind": "admin#reports#activity",
      "id": {
        "time": "2023-04-24T21:47:28.605Z",
        "uniqueQualifier": "-7690708801800961903",
        "applicationName": "token",
        "customerId": "C01lxktk6"},
      "etag": "\"P4UZt3E47YE1EedEgRGeNMKVdI4cxfy8A0aTCl4A_NM/1QpOzCwQYYr2jo2HxWYK9nP9lEs\"",
      "actor": {
        "email": "joeblogs@awsomecompany.com",
        "profileId": "117226486721886598480"},
      "ipAddress": "192.77.111.2",
  "events": [
        { "name": "authorize",
          "parameters": [
            { "name": "client_id",
              "value": "804540243761-puvq6928tmj1dh9002q93h2bjv9dki6v.apps.googleusercontent.com"},
            { "name": "app_name",
              "value": "Google Hyper-Search Engine"},
```

Figure 8: Google Workspace API (OAuth Log Details)

Appendix B: OAuth 2.0 Essentials

Term	Meaning
user, user agent	The human user and browser client used in OAuth authorization
3rd-party app	OAuth client application requesting access to user resources e.g., PDF Image editor, PayPal
resources	User's cloud resources data, functionality e.g., Google Drive files
resource manager	Party (product/service) that manages the user resources e.g., Google or Google Drive
Identity provider (iDP)	The authentication and authorization provider for the cloud users, typically including or calling a directory service e.g., Google Workspace
scopes	Specific permissions for access to user resources e.g., View and manage the files in your Google Drive
approval, authorization	The user's action to approve or authorise an application to access user resources

Table 13: OAuth Terms and Concepts

OAuth 2.0[1] was designed to allow users to grant 3rd-party applications access to their cloud data and resources. It is the de-facto standard for cloud identity access and authorization and is ubiquitous. Since 2012, it has grown to handle a multitude of use cases including authorization of public web apps, desktop apps, mobile apps, smart TV apps, as well as single sign-on. It is not an interoperability protocol but rather a looser framework for authorization with multiple optional or custom extensions.

OAuth 2.0 was designed specifically to address several security concerns with the then-broad practice of storing user passwords in 3rd-party applications:

- Storage of a user's credentials (typically, a password for cloud resources) within a 3rd-party application
- Compromise of a 3rd-party application resulting in compromise of the user 's credential
- A weak password-based authentication system
- 3rd-party applications gaining overly broad access via the user's credentials (same privileges as the user)
- Inability to revoke one 3rd-party application access without revoking access to all 3rd-party applications

OAuth 2.0 is widely implemented for both consumer and enterprise authorization use cases, and most users encounter it every day, in cases such as:

- Granting access to a website (3rd-party app) to use one's electronic payment resource (e.g., Venmo, PayPal)
- Granting a utility (e.g., 3rd-party app such as Slack, Jira, or image editor) access to one's cloud file-sharing resource (e.g., OneDrive)

The granting process is user-driven or controlled by the user and is done dynamically, as needed whenever a 3rd-party app requests access or permissions. Users interact with the identity provider of the resource manager to authenticate themselves and authorize the app request within the user's browser. After the approval is granted, the 3rd-party application receives access (via OAuth tokens) to access the user's resources[1]:

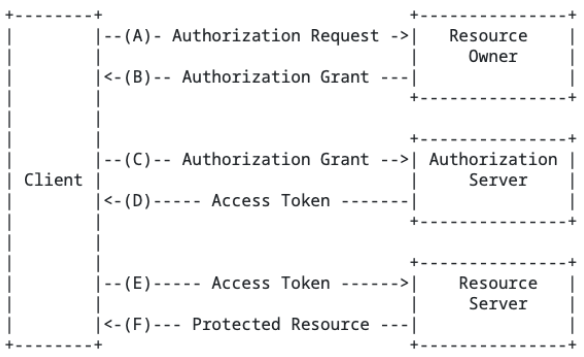


Figure 9: Generic OAuth 2.0 authorization flow

OAuth access tokens are temporary, typically with an hour lifespan, however OAuth provides a refresh token that allow applications to refresh the access tokens as required, for near-permanent access: [1]

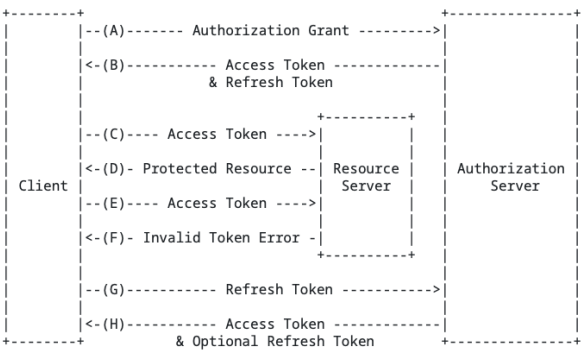


Figure 10: OAuth 2.0 access token refresh flow

Because OAuth is a user-controlled approval process, where the access is granted dynamically, as needed, by the users, this has several implications:

- Users by default can create numerous **access paths unknown** to the IT or security departments.
- There is **minimal tracking and management** of access paths (or generated OAuth tokens)

The OAuth protocol is complex due the wide range of use cases it addresses, the number of subsequent RFCs published to evolve the protocol framework, and the optional or custom extension areas of the spec. This has led to widely varying vendor implementations that have a:

- **Lack of visibility** into OAuth activity including the granting of permissions and use of OAuth credentials. e.g.,
  - the refresh token process by a 3rd-party application is not typically logged
  - when a 3rd-party app performs an API call, which OAuth token being used is not logged
- **Lack of controls** and manageability. i.e., incomplete or immature controls in the operational areas of: prevention, detection, remediation.
- **Unreliable 3rd-party app identity**. All vendor implementations have weak to no application identity mechanisms in place, other than obscure ids and developer-chosen names
- **No application validation**. Development of 3rd-party typically involves a validation/verification process for publishing public apps. The application status ("validated" or "verified") is not available to users via an API or feed.

Measuring OAuth application risk, risk prioritization, and mitigating impact is still immature with a lack of processes, methodologies related to risk; there are little to no best practices around OAuth risk metrics and evaluation.

References

[1] Hardt, D. (2012, October). *The OAuth 2.0 Authorization Framework*. IETF. https://datatracker.ietf.org/doc/html/rfc6749

[2] First.org. (n.d.). *Common Vulnerability Scoring System SIG*. Retrieved September 21, 2023, from https://www.first.org/cvss

[3] Google. (n.d.). *OAuth 2.0 Scopes for Google APIs*. Retrieved September 21, 2023, from https://developers.google.com/identity/protocols/oauth2/scopes

[4] Kaspersky. (2019, August 27). *Malicious Android app had more than 100 million downloads in Google Play*. https://usa.kaspersky.com/blog/camscanner-malicious-android-app/18535/

[5] Sinha, B. (2021, January 9). The Print. https://theprint.in/india/delhi-police-admits-it-used-banned-chinese-app-camscanner-apologises-on-twitter/582767/