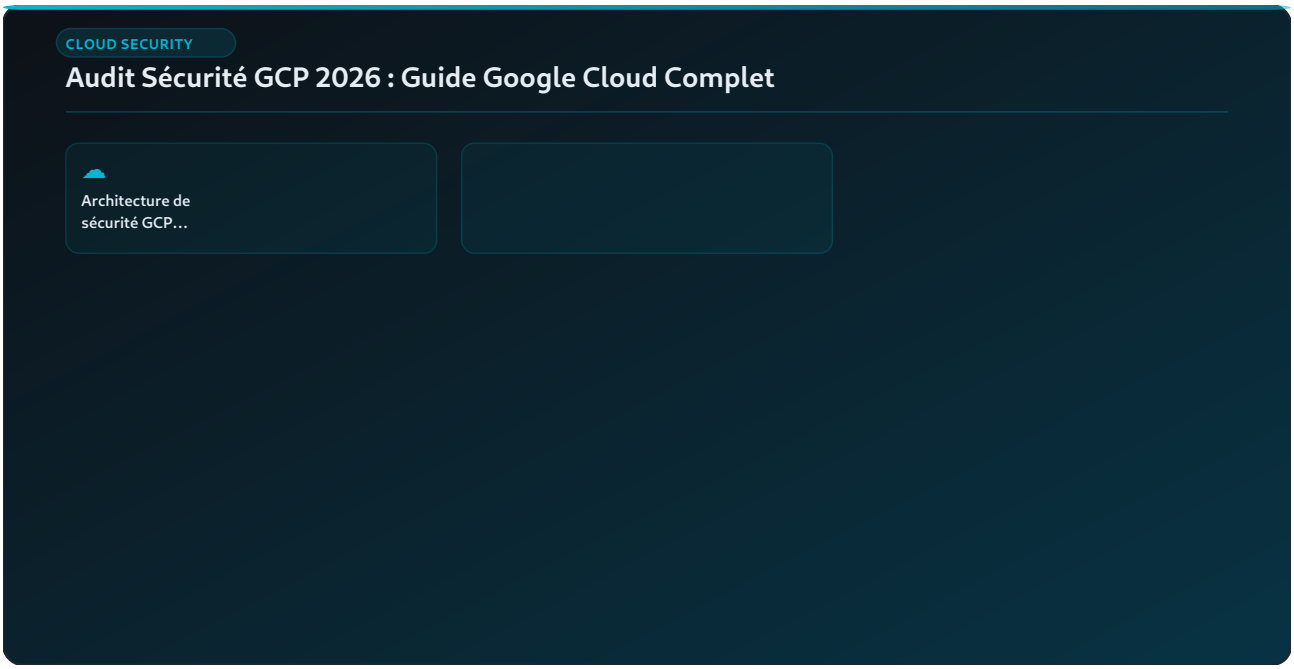
Audit Sécurité GCP 2026 : Guide Complet Google Cloud Platform

> Auditez votre GCP 2026 : IAM least privilege, VPC firewall, Cloud Logging, BigQuery, Cloud Run. Checklist RSSI, outils ScoutSuite et Forseti Security.

Google Cloud Platform (GCP) s'impose progressivement comme le troisième hyperscaler mondial et gagne du terrain dans les entreprises françaises, notamment dans les secteurs de la data, du retail et de l'industrie. Pourtant, la sécurité de GCP reste moins bien documentée que celle d'AWS ou Azure, et les RSSI se retrouvent souvent démunis face à l'audit de leur infrastructure Google Cloud. GCP présente une architecture de sécurité sophistiquée — avec une hiérarchie de ressources en trois niveaux, un modèle IAM basé sur les rôles et les bindings, et des services de logging et de monitoring natifs — mais cette sophistication se traduit aussi par une complexité accrue pour l'audit. Une mauvaise configuration de la hiérarchie Organisation/Folder/Project, des service accounts avec des clés téléchargeables et des permissions excessives, des buckets Cloud Storage publics, ou des règles firewall VPC trop permissives peuvent exposer des données sensibles ou ouvrir la voie à une compromission complète du tenant GCP. Ce guide d'audit sécurité GCP 2026 fournit une méthodologie structurée, des commandes gcloud concrètes et une checklist opérationnelle pour les RSSI, architectes cloud et consultants en sécurité qui doivent évaluer la posture de sécurité d'un environnement Google Cloud Platform.



Avant d'auditer un environnement GCP, il est indispensable de comprendre l'architecture de sécurité native de Google Cloud. GCP repose sur plusieurs piliers de sécurité : la hiérarchie de ressources, le modèle IAM, la sécurité réseau VPC, et les services de logging et monitoring. Maîtriser ces fondamentaux est la condition préalable à tout audit efficace.

Resource hierarchy : Organisation, Folders et Projects

La hiérarchie de ressources GCP est le fondement de la gouvernance sécurité. Elle s'organise en trois niveaux :

Organisation : le niveau racine, lié au domaine Google Workspace ou Cloud Identity de l'entreprise. Les politiques appliquées à l'organisation héritent vers tous les dossiers et projets. Le rôle `roles/resourcemanager.organizationAdmin` est l'équivalent du Global Administrator Azure — il faut le protéger avec une attention maximale.

Folders : regroupements logiques de projets, permettant de refléter la structure organisationnelle (par BU, par environnement, par équipe). Les Organization Policies et les IAM bindings

appliqués au niveau Folder héritent vers tous les projets fils. Une mauvaise configuration à ce niveau peut affecter des dizaines de projets.

Projects : l'unité de base de GCP. Chaque projet possède ses propres ressources (VMs, buckets, bases de données), sa facturation, et ses IAM bindings. L'audit doit vérifier que les projets sont correctement isolés et que les permissions ne "remontent" pas au niveau organisation sans justification.

La commande fondamentale pour auditer la hiérarchie est :

```
 # Lister toute la hiérarchie Organisation/Folders/Projects
gcloud organizations list
gcloud resource-manager folders list --organization=ORG_ID
gcloud projects list --filter="parent.id=FOLDER_ID"

# Vérifier les Organization Policies (contraintes imposées à l'org)
gcloud org-policies list --organization=ORG_ID
gcloud org-policies describe constraints/compute.vmExternalIpAccess --organization=ORG_ID
```

Identity and Access Management GCP

Le modèle IAM GCP repose sur trois concepts : les **membres** (Google Accounts, Service Accounts, Google Groups, domaines Cloud Identity), les **rôles** (collections de permissions — primitifs, prédéfinis, ou personnalisés), et les **bindings** (associations membre-rôle sur une ressource).

Les rôles primitifs (`roles/owner` , `roles/editor` , `roles/viewer`) sont à bannir dans un environnement de production : ils accordent des permissions extrêmement larges et ne respectent pas le principe de least privilege. Leur présence dans les IAM bindings d'un projet est un finding critique lors d'un audit.

Les **Service Accounts** sont des identités pour les applications et les workloads. Ils présentent plusieurs risques spécifiques : les clés téléchargeables (clés JSON) constituent des credentials permanents qui ne peuvent pas être révoqués facilement ; les services accounts avec des rôles trop étendus peuvent être abusés si le service qu'ils desservent est compromis ; et le binding `roles/iam.serviceAccountUser` sur un service account avec des droits élevés permet d'impersonnifier ce compte.

Checklist audit IAM GCP

L'audit IAM est la section la plus critique d'un audit GCP. Voici les vérifications essentielles avec les commandes associées.

> ⚡ **Retour terrain**
>
> J'ai accompagné la mise en sécurité d'un environnement GCP pour un groupe média. Le principal risque identifié : des comptes de service avec des rôles Owner ou Editor au niveau projet, utilisés par des applications en production. L'inventaire des comptes de service et la migration vers des rôles personnalisés à moindres privilèges a pris 6 semaines — mais lors d'un incident ultérieur impliquant un secret volé, le rayon d'impact était limité à un seul microservice.

1. Identifier les bindings avec des rôles primitifs dangereux :

```
 # Lister tous les bindings owner/editor sur un projet
gcloud projects get-iam-policy PROJECT_ID --format=json |
  python3 -c "
import sys, json
policy = json.load(sys.stdin)
for binding in policy.get('bindings', []):
    if binding['role'] in ['roles/owner', 'roles/editor']:
        print(f"CRITIQUE - Rôle: {binding['role']}")
        for member in binding['members']:
            print(f"  Membre: {member}")
"
```

2. Identifier les service accounts avec des clés téléchargeables :

```
# Lister toutes les clés de service accounts (hors clés managées GCP)
for SA in $(gcloud iam service-accounts list --project=PROJECT_ID --format="value(email)"); do
  KEYS=$(gcloud iam service-accounts keys list --iam-account=$SA --managed-by=user --format="valu
  if [ -n "$KEYS" ]; then
    echo "ATTENTION - Service Account avec clé utilisateur: $SA"
    echo "$KEYS"
  fi
done
```

3. Détecter les comptes allUsers ou allAuthenticatedUsers dans les IAM bindings :

```
# Détecter les accès publics au niveau projet (finding critique)
gcloud projects get-iam-policy PROJECT_ID --format=json |
  python3 -c "
import sys, json
policy = json.load(sys.stdin)
for binding in policy.get('bindings', []):
    for member in binding['members']:
        if member in ['allUsers', 'allAuthenticatedUsers']:
            print(f'CRITIQUE - Accès public: {member} avec rôle {binding["role"]}')
"
```

4. Vérifier les rôles personnalisés sur-privilegiés :

```
# Lister les rôles personnalisés et leurs permissions
gcloud iam roles list --project=PROJECT_ID --format="value(name)"
gcloud iam roles describe CUSTOM_ROLE_ID --project=PROJECT_ID
```

5. Auditer les Service Accounts avec impersonation permissive :

```
# Vérifier qui peut impersonnifier les service accounts critiques
gcloud iam service-accounts get-iam-policy SA_EMAIL --project=PROJECT_ID
```

Les findings IAM doivent être priorisés selon leur impact : un rôle Owner accordé à un utilisateur externe est critique, un service account editor sans clé téléchargeable est moyen. L'objectif est d'atteindre un modèle IAM où chaque identité n'a que les permissions strictement nécessaires à sa fonction — le principe de least privilege que nous détaillons dans notre guide sur l'architecture Zero Trust.

Audit réseau GCP : VPC, Firewall Rules, Cloud NAT

Le réseau dans GCP est Software-Defined : les VPC (Virtual Private Cloud) sont globaux par défaut, les sous-réseaux sont régionaux, et les règles firewall s'appliquent au niveau VPC. L'audit réseau GCP se concentre sur quatre points critiques.

Règles firewall dangereuses : les règles autorisant l'accès depuis `0.0.0.0/0` vers des ports d'administration (22/SSH, 3389/RDP, 5432/PostgreSQL, 3306/MySQL) sont particulièrement risquées. GCP dispose d'un VPC par défaut avec des règles permissives (`default-allow-ssh` , `default-allow-rdp`) qui doivent être supprimées ou restreintes.

```
 # Identifier les règles firewall autorisant du trafic depuis Internet sur des ports sensibles
gcloud compute firewall-rules list --project=PROJECT_ID --format=json |
  python3 -c "
import sys, json
rules = json.load(sys.stdin)
sensitive_ports = ['22', '3389', '3306', '5432', '6379', '27017', '9200']
for rule in rules:
    if rule.get('direction') == 'INGRESS' and rule.get('disabled') == False:
        ranges = rule.get('sourceRanges', [])
        if '0.0.0.0/0' in ranges or '::/0' in ranges:
            allowed = rule.get('allowed', [])
            for a in allowed:
                ports = a.get('ports', [])
                protocol = a.get('IPProtocol', '')
                for port in ports:
                    if any(sp in str(port) for sp in sensitive_ports) or protocol == 'all':
                        print(f'CRITIQUE - Règle: {rule["name"]}, Protocole: {protocol}, Ports: {
"
```

VPC Flow Logs : les VPC Flow Logs enregistrent les métadonnées des connexions réseau au niveau des sous-réseaux. Leur absence prive l'équipe sécurité d'une source de détection précieuse pour identifier les connexions anormales, les exfiltrations de données et les scans réseau internes.

```
 # Vérifier si les VPC Flow Logs sont activés sur les sous-réseaux
gcloud compute networks subnets list --project=PROJECT_ID --format=json |
  python3 -c "
import sys, json
subnets = json.load(sys.stdin)
for subnet in subnets:
    flow_logs = subnet.get('logConfig', {}).get('enable', False)
    if not flow_logs:
        print(f'MOYEN - VPC Flow Logs désactivés: {subnet["name"]} ({subnet["region"]})')
"
```

Cloud NAT et accès Internet : les instances GCE sans adresse IP externe devraient utiliser Cloud NAT pour accéder à Internet — cela évite d'exposer des IPs publiques. Vérifier que les instances sensibles (bases de données, services internes) n'ont pas d'IP externe associée.

Private Service Connect et Private Google Access : l'accès aux APIs Google (Cloud Storage, BigQuery, etc.) devrait se faire via des endpoints privés pour éviter que le trafic transite par Internet. Vérifier que Private Google Access est activé sur les sous-réseaux contenant des workloads sensibles.

Audit Cloud Logging et monitoring

La visibilité sur les actions réalisées dans GCP est assurée par Cloud Logging (anciennement Stackdriver Logging) et Security Command Center. L'audit de la configuration de ces services est crucial : un environnement sans logging adéquat est un environnement aveugle face aux incidents de sécurité.

GCP propose trois types de logs d'audit dans Cloud Audit Logs :

Admin Activity logs : enregistrent les modifications de configuration des ressources (création/suppression de VM, modification d'IAM policies). Activés par défaut, non désactivables, sans coût.

Data Access logs : enregistrent les lectures et écritures de données (accès aux objets Cloud Storage, requêtes BigQuery). Désactivés par défaut et peuvent générer des volumes importants — ils doivent être activés sur les ressources contenant des données sensibles.

System Event logs : enregistrent les actions automatiques de GCP. Activés par défaut.

```
 # Vérifier la configuration des Data Access logs au niveau projet
gcloud projects get-iam-policy PROJECT_ID --format=json |
  python3 -c "
import sys, json
policy = json.load(sys.stdin)
audit_configs = policy.get('auditConfigs', [])
if not audit_configs:
    print('ATTENTION - Aucun Data Access Log configuré au niveau projet')
for config in audit_configs:
    print(f'Service: {config["service"]}')
    for log_type in config.get('auditLogConfigs', []):
        print(f'  Type: {log_type["logType"]}')
"

# Vérifier l'export des logs vers un bucket ou BigQuery (retention longue durée)
gcloud logging sinks list --project=PROJECT_ID
```

Security Command Center (SCC) est le SIEM natif GCP. Son tier Premium offre des détections comportementales (Event Threat Detection), des analyses de configuration (Security Health Analytics) et une intégration avec Google Threat Intelligence. L'audit doit vérifier que SCC Premium est activé au niveau organisation et que les findings critiques sont routés vers un système d'alerte (Cloud Monitoring, PubSub, SIEM externe).

```
 # Lister les findings actifs Security Command Center (nécessite gcloud SCC CLI)
gcloud scc findings list ORGANIZATION_ID
  --filter="state=ACTIVE AND severity=CRITICAL"
  --format="table(finding.name,finding.category,finding.resourceName)"
```

Audit des services managés à risque

GCP propose des dizaines de services managés, chacun avec ses propres vecteurs de mauvaise configuration. Les trois services qui concentrent le plus de findings lors des audits sont BigQuery, Cloud Storage et Cloud Run/Cloud Functions.

BigQuery : exposition de données

BigQuery est le service d'analyse de données massivement utilisé par les entreprises qui adoptent GCP. Il contient souvent les données les plus sensibles de l'organisation — logs applicatifs, données clients, données financières. Les risques principaux sont :

Datasets publics : un dataset BigQuery avec `allUsers` en lecture est une exposition de données catastrophique. À vérifier systématiquement.

Authorized Views : mal configurées, elles peuvent exposer des données à des projets non autorisés.

Partition expiry et data retention : les données sensibles doivent avoir une durée de rétention définie, sinon elles s'accumulent indéfiniment.

```
 # Lister tous les datasets et vérifier leurs ACLs
bq ls --project_id=PROJECT_ID --format=json | python3 -c "import sys,json; [print(d['datasetRefer

# Pour chaque dataset, vérifier les ACLs
bq show --format=prettyjson PROJECT_ID:DATASET_ID | python3 -c "
import sys, json
info = json.load(sys.stdin)
for acl in info.get('access', []):
    if acl.get('specialGroup') in ['allUsers', 'allAuthenticatedUsers']:
        print(f'CRITIQUE - Dataset public: {acl}')
"
```

Cloud Storage : buckets publics

Les buckets Cloud Storage mal configurés sont l'une des causes les plus fréquentes de fuites de données dans GCP. Les vérifications essentielles sont la présence de `allUsers` ou `allAuthenticatedUsers` dans les ACLs des buckets, l'absence de Uniform Bucket-Level Access (qui simplifie la gestion des permissions et évite les ACLs au niveau objet), et la présence de données sensibles dans des buckets dont le nom est prévisible ou déjà connu publiquement.

```
 # Lister tous les buckets du projet
gsutil ls -p PROJECT_ID

# Vérifier les permissions IAM de chaque bucket
gsutil iam get gs://BUCKET_NAME | python3 -c "
import sys, json
policy = json.load(sys.stdin)
for binding in policy.get('bindings', []):
    for member in binding.get('members', []):
        if member in ['allUsers', 'allAuthenticatedUsers']:
            print(f'CRITIQUE - Bucket public: {member} avec rôle {binding["role"]}')
"

# Vérifier si Uniform Bucket-Level Access est activé
gsutil uniformbucketlevelaccess get gs://BUCKET_NAME
```

La configuration de la prévention de l'accès public au niveau organisation via Organization Policy (`constraints/storage.publicAccessPrevention`) est la meilleure protection systémique contre ce type de misconfiguration. Un audit GCP mature vérifie que cette contrainte est appliquée au niveau Organisation ou au minimum au niveau des Folders contenant des données sensibles.

Cloud Run / Cloud Functions : permissions excessives

Les services serverless GCP (Cloud Run, Cloud Functions) s'exécutent sous l'identité d'un service account. Si ce service account est trop permissif, toute vulnérabilité dans le code applicatif (injection, SSRF) peut être amplifiée en une compromission IAM. Les vérifications clés sont :

```
 # Lister les Cloud Run services et leurs service accounts
gcloud run services list --project=PROJECT_ID --format=json |
  python3 -c "
import sys, json
services = json.load(sys.stdin)
for svc in services:
    sa = svc.get('spec', {}).get('template', {}).get('spec', {}).get('serviceAccountName', 'defau
    public = any(b['members'] for b in svc.get('status', {}).get('conditions', []) if 'allUsers'
    print(f'Service: {svc["metadata"]["name"]}, SA: {sa}')
"

# Vérifier si un Cloud Run service est accessible publiquement (allUsers invoker)
gcloud run services get-iam-policy SERVICE_NAME --region=REGION --project=PROJECT_ID
```

Les services Cloud Run exposés publiquement sans authentification (`allUsers` avec le rôle `roles/run.invoker`) ne sont pas nécessairement un problème si le service est conçu pour être public — mais ils doivent être explicitement justifiés. La présence d'un service account avec des permissions élevées sur un service public est systématiquement un finding critique. Cette dimension s'inscrit dans les pratiques de sécurité cloud que nous abordons dans notre guide sur la conformité cloud HDS et SecNumCloud.

Outils d'audit GCP automatisé

L'audit manuel avec gcloud est exhaustif mais chronophage. Des outils automatisés permettent d'accélérer la phase de collecte et d'identifier rapidement les findings les plus courants.

Outil	Type	Points forts	Limitations
ScoutSuite	Open source	Multi-cloud, rapport HTML interactif, > 100 règles GCP	Pas de remédiation automatique
Forseti Security	Open source (Google)	Inventaire continu, règles personnalisables, notifications	Déprécié au profit de SCC, setup complexe
Prowler	Open source	Checks CIS GCP, NIST, PCI-DSS, output JSON/CSV/HTML	Moins de règles GCP qu'AWS
Security Command Center	Natif GCP (payant Premium)	Intégré, Event Threat Detection, remédiation guidée	Coût Premium, moins flexible que open source
Checkov	Open source	Analyse IaC Terraform/Deployment Manager, shift-left	Analyse statique uniquement (pas de runtime)

```
 # Audit GCP complet avec ScoutSuite
pip3 install scoutsuite
scout gcp --service-account /path/to/key.json --project-id PROJECT_ID
# Rapport dans ./scoutsuite-report/

# Audit avec Prowler (checks CIS GCP Benchmark)
pip3 install prowler
prowler gcp --project-ids PROJECT_ID -c cis_gcp

# Prowler avec output JSON pour intégration SIEM
prowler gcp --project-ids PROJECT_ID --output-formats json-asff
```

Pour les organisations qui souhaitent intégrer l'audit GCP dans leur pipeline DevSecOps, Checkov analyse les fichiers Terraform avant le déploiement et bloque les ressources qui violent les politiques de sécurité. C'est l'approche shift-left appliquée à la configuration cloud, complémentaire aux audits runtime. Ce type d'approche s'intègre naturellement dans une démarche de reconnaissance et d'évaluation continue de la posture de sécurité.

La combinaison recommandée pour un audit GCP complet est : ScoutSuite pour la couverture globale et le rapport HTML client, Prowler pour la conformité CIS/PCI/NIST avec export JSON, et Security Command Center pour la surveillance continue post-audit. Pour les aspects organisationnels et la gestion des risques, notre guide EBIOS RM ANSSI 2026 détaille comment structurer et prioriser les findings d'un audit cloud dans une démarche formelle de gestion des risques cyber.

Questions fréquentes sur l'audit GCP

Quelle est la fréquence recommandée pour un audit GCP ?

Un audit GCP complet (manuel + automatisé) devrait être réalisé au minimum une fois par an, avec des scans automatisés hebdomadaires via Prowler ou ScoutSuite. Les changements d'architecture majeurs (nouveau projet, nouvelle région, adoption d'un service managé critique) doivent déclencher un audit ciblé. Security Command Center en mode Premium offre une surveillance continue qui peut remplacer les scans hebdomadaires manuels. Pour les environnements soumis à des réglementations (HDS, PCI-DSS, NIS 2), des audits trimestriels sont généralement requis par les référentiels de conformité.

Comment gérer les service accounts dans un environnement GCP sécurisé ?

Les bonnes pratiques pour les service accounts GCP suivent plusieurs principes : désactiver les clés téléchargeables via Organization Policy (`constraints/iam.disableServiceAccountKeyCreation`), utiliser Workload Identity Federation pour les workloads externes (GitHub Actions, Jenkins) plutôt que des clés JSON, appliquer le least privilege en attribuant des rôles prédéfinis ciblés plutôt que `roles/editor`, éviter d'utiliser le service account de calcul par défaut (souvent trop permissif), et activer l'audit des clés existantes avec des alertes sur la création de nouvelles clés. Le Managed Service Account est préférable aux User-managed accounts pour les services GCP natifs.

Quels sont les findings GCP les plus fréquents lors d'un audit ?

D'après notre expérience d'audit GCP, les cinq findings les plus fréquents sont : (1) des service accounts avec le rôle primitif Editor ou Owner au lieu de rôles prédéfinis ciblés ; (2) des clés de service account téléchargeables qui n'ont pas été rotées depuis plus de 90 jours ; (3) des règles firewall VPC autorisant SSH/RDP depuis 0.0.0.0/0 sur des instances de production ; (4) des Data Access Logs non configurés sur les ressources contenant des données sensibles (BigQuery, Cloud Storage) ; et (5) des buckets Cloud Storage sans Uniform Bucket-Level Access avec des

ACLs héritées au niveau objet. Ces findings reflètent le décalage classique entre la vitesse d'adoption du cloud et la maturité des pratiques de sécurité.

Comment prioriser les remédations après un audit GCP ?

La priorisation des remédiations doit combiner deux critères : la criticité technique du finding (probabilité d'exploitation et impact) et la facilité de remédiation. Les quick wins — findings critiques avec une remédiation simple comme la suppression d'une règle firewall ou la désactivation d'un bucket public — doivent être traités en priorité absolue, idéalement dans les 24 à 72 heures. Les findings structurels (refonte du modèle IAM, mise en place d'une Organisation Policy, activation de Security Command Center Premium) nécessitent un projet dédié avec une roadmap. Un tableau de bord des findings (Security Command Center ou export Prowler vers Google Sheets) aide le RSSI à suivre la progression. Cette approche de priorisation s'applique également dans le cadre d'une analyse de risques formelle : notre guide Cyber Threat Landscape France 2026 contextualise les menaces qui pèsent sur les environnements cloud français.

À RETENIR

Points clés à retenir

L'audit IAM est la section la plus critique : eliminer les rôles primitifs (Owner/Editor) et les service accounts avec clés téléchargeables trop permissifs.

Activer les Data Access Logs sur toutes les ressources contenant des données sensibles (BigQuery, Cloud Storage) — ils sont désactivés par défaut.

Les règles firewall VPC autorisant 0.0.0.0/0 sur des ports d'administration sont des findings critiques à corriger en priorité.

ScoutSuite + Prowler couvrent 90% des checks d'un audit GCP standard ; Security Command Center Premium apporte la surveillance continue.

Les Organization Policies sont la protection systémique la plus efficace : `iam.disableServiceAccountKeyCreation` et `storage.publicAccessPrevention` bloquent les classes entières de misconfigurations.

Vérifier systématiquement BigQuery, Cloud Storage et Cloud Run : ces trois services concentrent la majorité des expositions de données dans GCP.

Un audit GCP complet nécessite à la fois des outils automatisés (ScoutSuite, Prowler) et une analyse manuelle de la hiérarchie organisationnelle et des configurations IAM complexes.

Sources

ANSSI CERT-FR — Publications et alertes

MITRE ATT&CK — Framework des techniques d'attaque

CISA — Advisories de cybersécurité

