LIVRE BLANC · CYBERSÉCURITÉ CLOUD

Guide Complet d'Audit de Sécurité Google Workspace

Méthodologie, Outils et Bonnes Pratiques — 10 Niveaux d'Audit
De l'IAM à la conformité réglementaire RGPD, NIS2 et CIS Benchmark

10	75+	12 000+	3
NIVEAUX D'AUDIT	POINTS DE CONTRÔLE	MOTS	PROFILS

CIS Benchmark · RGPD · ISO 27001 · NIS2 · DORA

Ayi NEDJIMI

Expert Cybersécurité & IA

Version 1.0 — Mars 2026

ayinedjimi-consultants.fr | Classification : Usage interne — Équipes Sécurité / GRC

Guide Complet d'Audit de Sécurité Google Workspace

Méthodologie, Outils et Bonnes Pratiques

Classification : Usage interne — Équipes Sécurité / GRC **Version :** 1.0 **Date :** Mars 2026
Périmètre : Google Workspace (anciennement G Suite) — Toutes éditions (Business, Enterprise, Education)

Table des Matières

1. [Introduction & Philosophie de l'Audit](#1-introduction--philosophie-de-laudit)
2. [Périmètre et Niveaux d'Audit](#2-périmètre-et-niveaux-daudit)
3. [Prérequis et Infrastructure d'Audit](#3-prérequis-et-infrastructure-daudit)
4. [Niveau 1 — Identités, Accès et Authentification](#4-niveau-1--identités-accès-et-authentification)
5. [Niveau 2 — Messagerie Gmail](#5-niveau-2--messagerie-gmail)
6. [Niveau 3 — Google Drive et Partages](#6-niveau-3--google-drive-et-partages)
7. [Niveau 4 — DNS et Sécurité Email](#7-niveau-4--dns-et-sécurité-email)
8. [Niveau 5 — Applications OAuth et Intégrations Tierces](#8-niveau-5--applications-oauth-et-intégrations-tierces)
9. [Niveau 6 — Appareils Mobiles et Points de Terminaison](#9-niveau-6--appareils-mobiles-et-points-de-terminaison)
10. [Niveau 7 — Groupes et Politiques de Partage](#10-niveau-7--groupes-et-politiques-de-partage)
11. [Niveau 8 — Journalisation, Alertes et Surveillance](#11-niveau-8--journalisation-alertes-et-surveillance)
12. [Niveau 9 — Reconnaissance Passive et Threat Intelligence](#12-niveau-9--reconnaissance-passive-et-threat-intelligence)
13. [Niveau 10 — Conformité Réglementaire](#13-niveau-10--conformité-réglementaire)
14. [Automatisation et Outillage](#14-automatisation-et-outillage)
15. [Rapport et Feuille de Route](#15-rapport-et-feuille-de-route)
16. [Référentiel CIS Benchmark Google Workspace](#16-référentiel-cis-benchmark-google-workspace)
17. [Annexes Techniques](#17-annexes-techniques)

---

1. Introduction & Philosophie de l'Audit

1.1 Pourquoi auditer Google Workspace ?

Google Workspace est devenu le système nerveux central de millions d'organisations : messagerie, stockage de fichiers, gestion des identités, collaboration en temps réel, et désormais intégrations d'intelligence artificielle. Cette centralisation offre un confort opérationnel indéniable mais crée une surface d'attaque massive et souvent sous-évaluée.

Contrairement à une infrastructure on-premise, la sécurité Google Workspace est une **responsabilité partagée** :

- Google est responsable de la sécurité **de** l'infrastructure (disponibilité, chiffrement au repos et en transit, physique des datacenters)
- L'organisation est entièrement responsable de la sécurité **dans** le tenant (configurations, droits d'accès, partages, applications tierces, politiques)

Cette limite est mal comprise par la majorité des équipes IT. Un audit Google Workspace vise précisément à évaluer **la posture de sécurité dans ce périmètre de responsabilité organisationnelle**.

1.2 Principes directeurs

Lecture seule absolue. Un audit ne doit jamais modifier la configuration d'un tenant de production. Toute action doit être réversible et tracée. Les techniques utilisées dans ce guide s'appuient exclusivement sur des APIs en lecture seule. **Exhaustivité et priorisation.** L'objectif n'est pas de lister des dizaines de findings sans impact, mais d'identifier les risques réels, de les quantifier et de les prioriser selon leur probabilité et leur impact métier. **Reproductibilité.** Chaque vérification décrite dans ce guide doit pouvoir être réalisée de manière identique par différents auditeurs sur différents tenants. **Preuve par les données.** Chaque finding doit être étayé par des données extraites des APIs Google, pas par des suppositions. La chaîne de preuves doit être documentée.

1.3 Types d'audit

TYPE	DURÉE	PROFONDEUR	CAS D'USAGE
Audit flash	2-4h	IAM + Gmail + DNS	Vérification rapide avant incident
Audit standard	1-2 jours	Tous niveaux 1-7	Audit annuel de conformité
Audit approfondi	3-5 jours	Tous niveaux + recon + threat intel	Pré-certification ISO 27001, post-incident
Audit continu	Permanent	Monitoring + alertes	Programme de sécurité mature

1.3 Menaces spécifiques à Google Workspace

Comprendre le modèle de menace propre à Google Workspace est essentiel pour orienter les vérifications. Les vecteurs d'attaque les plus fréquemment observés en environnement Workspace sont :

1. Phishing de credentials Le phishing reste le vecteur initial le plus courant. Les attaquants créent des pages de connexion Google factices, souvent hébergées sur des domaines légitimes compromis ou des services cloud (Firebase, Cloudflare Workers). Le vol de cookies de session via des proxies AITM (Adversary-in-the-Middle) comme EvilGinx permet de contourner la 2FA classique (TOTP/SMS) en capturant à la fois les credentials et le cookie de session valide. **2. OAuth Consent Phishing** L'attaquant enregistre une application OAuth malveillante et envoie à la victime un lien d'autorisation légitime (hébergé sur accounts.google.com). L'utilisateur pense autoriser un outil de productivité et accorde en réalité des droits Gmail ou Drive à une app contrôlée par l'attaquant. Le token OAuth créé persiste même après un changement de mot de passe. **3. Account Takeover via récupération** Si l'email de récupération d'un compte Workspace est compromis (notamment via les milliers de breaches publiques), l'attaquant peut déclencher la procédure de récupération du compte Google et contourner la 2FA. C'est un vecteur sous-estimé mais très efficace, notamment via les stealer logs qui contiennent des mots de passe en clair. **4. Insider threat** Un employé malveillant ou un ex-employé avec accès résiduel peut activer silencieusement un transfert Gmail, exporter massivement des fichiers Drive, ou créer un service account backdoor. La détection repose sur l'analyse comportementale des logs d'administration et d'accès. **5. Compromission de la chaîne logicielle (supply chain)** Des applications OAuth légitimes utilisées par l'organisation peuvent elles-mêmes être compromises. Un attaquant qui prend le contrôle du serveur d'une app tierce hérite de tous les tokens OAuth accordés par les utilisateurs. La validation régulière des applications autorisées est critique. **6. Attaque sur les appareils BYOD non gérés** Les utilisateurs accèdent à Gmail et Drive depuis des appareils personnels non gérés. Un malware info-stealer sur ces appareils capture les cookies de session Google, donnant à l'attaquant un accès authentifié sans besoin de credentials.

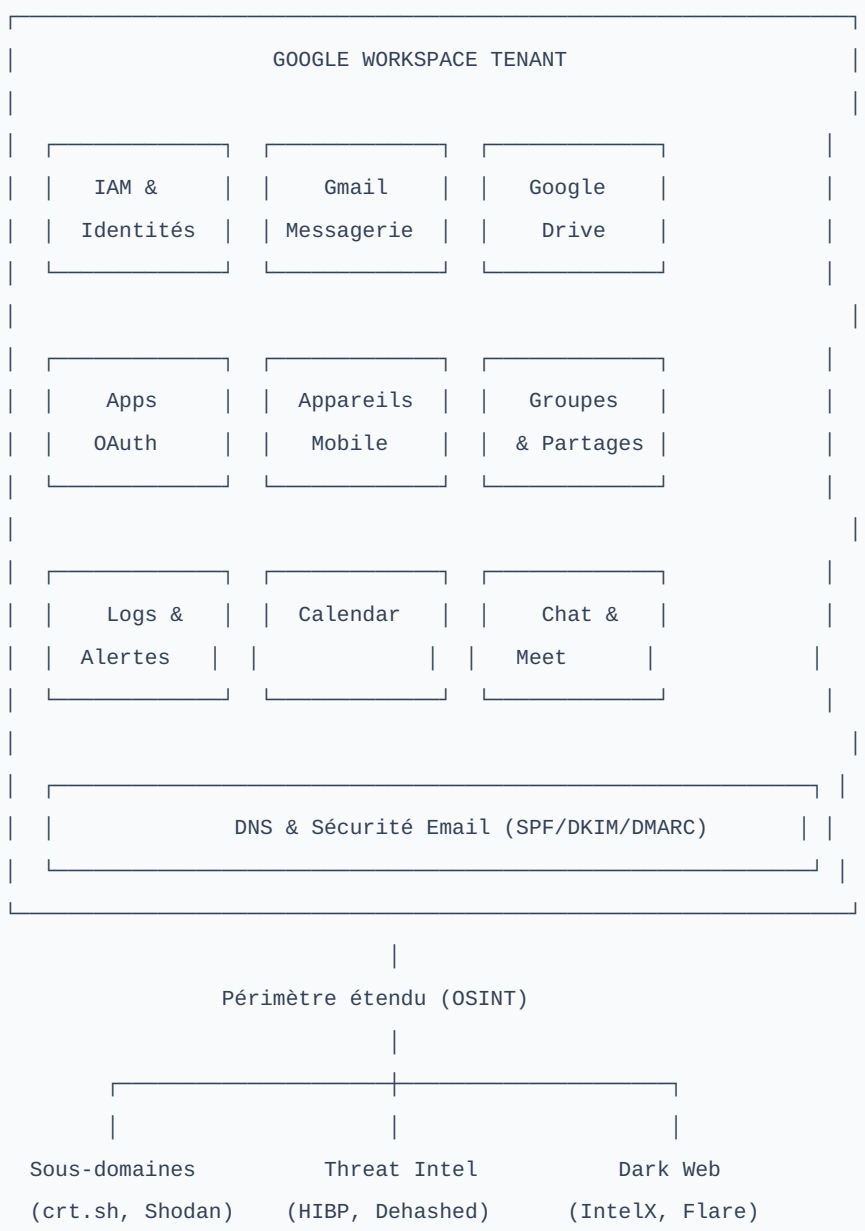
1.4 Rôles et responsabilités

RÔLE	RESPONSABILITÉ
Auditeur principal	Exécution technique, collecte de données, analyse
Super Admin client	Création du compte de service, délégation des droits
DPO	Validation du périmètre RGPD, approbation des accès
RSSI	Validation de la méthodologie, priorisation des risques
Direction	Approbation de l'audit, réception du rapport exécutif

2. Périmètre et Niveaux d'Audit

2.1 Périmètre technique

Un audit Google Workspace complet couvre les composants suivants :



2.2 Matrice de couverture par niveau

NIVEAU	DOMAINE	APIS REQUISES	IMPACT POTENTIEL
1	IAM & Authentification	Directory v1, Reports v1	Critique
2	Gmail & Messagerie	Gmail API, Directory v1	Critique
3	Google Drive	Drive API v3	Critique
4	DNS & Email	DNS (externe)	Élevé
5	Applications OAuth	Directory v1, Reports v1	Élevé
6	Appareils MDM	Directory v1 (devices)	Moyen
7	Groupes	Directory v1, Groups Settings	Moyen
8	Logs & Alertes	Reports v1, Alert Center	Élevé
9	Recon & Threat Intel	APIs externes	Variable
10	Conformité	Synthèse	Réglementaire

3. Prérequis et Infrastructure d'Audit

3.1 Compte requis côté client

Pour réaliser un audit complet via APIs, il faut :

1. **Un compte Super Admin Google Workspace** appartenant à l'organisation auditée — pour créer le Service Account et configurer la délégation
2. **Un projet Google Cloud Platform (GCP)** — pour héberger le Service Account

> *En lecture seule, le risque pour le tenant est nul. La clé de service account doit rester strictement confidentielle et être révoquée à la fin de l'audit.*

3.2 Création du projet GCP et du Service Account

Étape 1 — Créer le projet GCP

```
# Via gcloud CLI (ou depuis console.cloud.google.com)
gcloud projects create audit-workspace-YYYYMMDD \
  --name="Audit Workspace"

gcloud config set project audit-workspace-YYYYMMDD
```

Étape 2 — Créer le Service Account

```
gcloud iam service-accounts create audit-sa \
  --description="Service Account lecture seule pour audit Workspace" \
  --display-name="Audit Service Account"

# Créer la clé JSON
gcloud iam service-accounts keys create audit-key.json \
  --iam-account=audit-sa@audit-workspace-YYYYMMDD.iam.gserviceaccount.com
```

Étape 3 — Activer les APIs nécessaires

```
# APIs essentielles
gcloud services enable admin.googleapis.com
gcloud services enable gmail.googleapis.com
gcloud services enable drive.googleapis.com
gcloud services enable calendar-json.googleapis.com
gcloud services enable alertcenter.googleapis.com
gcloud services enable groupssettings.googleapis.com
```

Étape 4 — Configurer la Domain-Wide Delegation

Dans **admin.google.com → Sécurité → Contrôles et données d'accès → Contrôle des API → Délégation au niveau du domaine** :

Ajouter le Client ID du Service Account avec les scopes suivants (en une seule ligne, séparés par des virgules) :

```
https://www.googleapis.com/auth/admin.reports.audit.readonly,
https://www.googleapis.com/auth/admin.reports.usage.readonly,
https://www.googleapis.com/auth/admin.directory.user.readonly,
https://www.googleapis.com/auth/admin.directory.group.readonly,
https://www.googleapis.com/auth/admin.directory.domain.readonly,
https://www.googleapis.com/auth/admin.directory.device.mobile.readonly,
https://www.googleapis.com/auth/admin.directory.user.security,
https://www.googleapis.com/auth/admin.directory.rolemanagement.readonly,
https://www.googleapis.com/auth/admin.directory.customer.readonly,
https://www.googleapis.com/auth/apps.groups.settings,
https://www.googleapis.com/auth/gmail.settings.basic,
https://www.googleapis.com/auth/drive.readonly,
https://www.googleapis.com/auth/calendar.readonly,
https://www.googleapis.com/auth/apps.alerts
```

> ***Attention :*** *Copier-coller les scopes sans espaces parasites. Une erreur fréquente est l'introduction d'espaces entre les scopes qui empêche la validation silencieusement.*

3.3 Configuration Python de base

```
# config.py
DOMAIN = "exemple.com"
SERVICE_ACCOUNT_KEY = "audit-key.json"
ADMIN_EMAIL = "admin@exemple.com"  # Email d'un Super Admin pour l'impersonation

SCOPES = [
    "https://www.googleapis.com/auth/admin.reports.audit.readonly",
    "https://www.googleapis.com/auth/admin.reports.usage.readonly",
    "https://www.googleapis.com/auth/admin.directory.user.readonly",
    "https://www.googleapis.com/auth/admin.directory.group.readonly",
    "https://www.googleapis.com/auth/admin.directory.user.security",
    "https://www.googleapis.com/auth/admin.directory.rolemanagement.readonly",
    "https://www.googleapis.com/auth/admin.directory.customer.readonly",
    "https://www.googleapis.com/auth/apps.groups.settings",
    "https://www.googleapis.com/auth/gmail.settings.basic",
    "https://www.googleapis.com/auth/drive.readonly",
    "https://www.googleapis.com/auth/calendar.readonly",
    "https://www.googleapis.com/auth/apps.alerts",
    "https://www.googleapis.com/auth/admin.directory.device.mobile.readonly",
]
```

```
# auth.py
from google.oauth2 import service_account
from googleapiclient.discovery import build
from config import SERVICE_ACCOUNT_KEY, ADMIN_EMAIL, SCOPES

def get_credentials(scopes=None):
    creds = service_account.Credentials.from_service_account_file(
        SERVICE_ACCOUNT_KEY,
        scopes=scopes or SCOPES,
        subject=ADMIN_EMAIL,
    )
    return creds

def get_directory_service():
    return build("admin", "directory_v1", credentials=get_credentials())

def get_reports_service():
    return build("admin", "reports_v1", credentials=get_credentials())
```

3.4 Prérequis techniques auditeur

OUTIL	VERSION MIN	USAGE
Python	3.10+	Scripts d'audit
google-api-python-client	2.x	Clients APIs Google
google-auth	2.x	Authentification OAuth2
dnspython	2.x	Vérifications DNS
requests	2.x	APIs externes (HIBP, crt.sh)
rich	13.x	Affichage terminal enrichi

```
pip install google-api-python-client google-auth dnspython requests rich
```

3.5 Synchronisation de l'horloge (critique)

Les tokens OAuth nécessitent une horloge système synchronisée. Une dérive de plus de 5 minutes provoque des erreurs `invalid_grant`. Avant tout audit :

```
# Linux/macOS
sudo ntpdate -u pool.ntp.org

# Windows (PowerShell admin)
w32tm /resync /force

# Ou via Python si accès admin
import ntplib, ctypes, time
c = ntplib.NTPClient()
r = c.request('pool.ntp.org', version=3)
# Appliquer via SetLocalTime sur Windows
```

---

4. Niveau 1 — Identités, Accès et Authentification

4.1 Inventaire des utilisateurs

La première étape est d'obtenir la liste exhaustive des utilisateurs et leurs attributs de sécurité.

```
def get_all_users(service):
    """Retourne tous les utilisateurs du domaine avec leurs attributs de sécurité."""
    users = []
    page_token = None
    while True:
        result = service.users().list(
            domain=DOMAIN,
            maxResults=500,
            pageToken=page_token,
            projection="full",
            orderBy="email",
        ).execute()
        users.extend(result.get("users", []))
        page_token = result.get("nextPageToken")
        if not page_token:
            break
    return users
```

Attributs à extraire et analyser :

ATTRIBUT API	SIGNIFICATION	ALERTE SI
`isAdmin`	Super Admin	> 4 comptes
`isDelegatedAdmin`	Admin délégué	Lister et justifier
`suspended`	Compte suspendu	Présent sans offboarding
`isEnrolledIn2Sv`	2FA enrollée	False = CRITIQUE
`isEnforcedIn2Sv`	2FA obligatoire enforced	False = non conforme CIS
`lastLoginTime`	Dernière connexion	> 90 jours = inactif
`recoveryEmail`	Email de récupération	Externe = risque bypass 2FA
`recoveryPhone`	Téléphone de récupération	Externe = risque SIM swap

4.2 Analyse des comptes Super Admin

Le CIS Benchmark recommande un maximum de 4 Super Admins (idéalement 2-3). Chaque compte Super Admin compromis donne un accès total au tenant.

Points de contrôle :

- Nombre de Super Admins (≤ 4)
- Les Super Admins utilisent-ils des comptes dédiés (non utilisés au quotidien) ?
- Chaque Super Admin dispose-t-il d'une clé de sécurité physique (FIDO2/YubiKey) ?
- Les Super Admins ont-ils des emails de récupération **internes** au domaine ?

```
def audit_super_admins(users):
    super_admins = [u for u in users if u.get("isAdmin")]
    risks = []
    for sa in super_admins:
        recovery = sa.get("recoveryEmail", "")
        if recovery and not recovery.endswith(f"@{DOMAIN}"):
            risks.append({
                "email": sa["primaryEmail"],
                "risk": "Email de récupération externe",
                "recovery": recovery,
                "severity": "CRITIQUE",
            })
    return super_admins, risks
```

4.3 Analyse de la 2FA

Distinction cruciale : Il existe une différence entre 2FA *enrollée* (l'utilisateur a configuré un second facteur) et 2FA *enforced* (la politique admin oblige la 2FA — un utilisateur ne peut pas la désactiver).

```
def audit_2fa(users):
    findings = {
        "no_2fa": [],           # Aucune 2FA configurée
        "enrolled_not_enforced": [],  # 2FA configurée mais pas obligatoire
        "fully_enforced": [],   # 2FA enforced
    }
    for user in users:
        enrolled = user.get("isEnrolledIn2Sv", False)
        enforced = user.get("isEnforcedIn2Sv", False)
        if not enrolled:
            findings["no_2fa"].append(user["primaryEmail"])
        elif enrolled and not enforced:
            findings["enrolled_not_enforced"].append(user["primaryEmail"])
        else:
            findings["fully_enforced"].append(user["primaryEmail"])
    return findings
```

Types de 2FA — Résistance au phishing :

TYPE 2FA	RÉSISTANCE PHISHING	RECOMMANDATION
FIDO2 / Passkey / YubiKey	✅ Totale	**Obligatoire pour admins**
TOTP (Google Authenticator)	⚠️ Partielle	Vulnérable AITM
SMS OTP	❌ Faible	Vulnérable SIM swap
Backup codes	❌ Très faible	À usage unique uniquement

4.4 Emails de récupération — Vecteur d'account takeover

L'email de récupération est le **talon d'Achille** de la 2FA. Si l'email de récupération est compromis, un attaquant peut réinitialiser le mot de passe **sans avoir besoin de la 2FA**. C'est le vecteur d'attaque le plus sous-estimé en environnement Workspace.

```
def audit_recovery_info(service, user_emails):
    findings = {
        "external_recovery_emails": [],
        "external_recovery_phones": [],
        "no_recovery": [],
    }
    for email in user_emails:
        user = service.users().get(userKey=email, projection="full").execute()
        recovery_email = user.get("recoveryEmail", "")
        recovery_phone = user.get("recoveryPhone", "")

        if recovery_email and not recovery_email.endswith(f"@{DOMAIN}"):
            findings["external_recovery_emails"].append({
                "user": email,
                "recovery_email": recovery_email,
                "provider": recovery_email.split("@")[-1],
            })
        if not recovery_email and not recovery_phone:
            findings["no_recovery"].append(email)
    return findings
```

Critères de risque pour les emails de récupération :

TYPE	NIVEAU	RAISON
`@gmail.com`, `@outlook.com`	🔴 ÉLEVÉ	Grand public, souvent peu sécurisé
`@yahoo.fr`, `@hotmail.fr`	🔴 ÉLEVÉ	Breaches historiques massives (Yahoo 2016)
Domaine d'une autre entreprise	🟠 MOYEN	Employé précédent ou freelance
`@domaine.com` interne	✅ OK	Contrôlé par l'organisation

4.5 Comptes inactifs et suspendus

Un compte inactif non supprimé représente un vecteur d'attaque persistant. Les comptes suspendus non traités signalent l'absence d'une procédure d'offboarding.

```
from datetime import datetime, timezone, timedelta

def audit_inactive_accounts(users, inactive_days=90):
    threshold = datetime.now(timezone.utc) - timedelta(days=inactive_days)
    inactive = []
    for user in users:
        last_login = user.get("lastLoginTime")
        if last_login:
            last_login_dt = datetime.fromisoformat(
                last_login.replace("Z", "+00:00")
            )
            if last_login_dt < threshold and not user.get("suspended"):
                inactive.append({
                    "email": user["primaryEmail"],
                    "last_login": last_login,
                    "days_inactive": (datetime.now(timezone.utc) -
last_login_dt).days,
                })
    suspended = [u for u in users if u.get("suspended")]
    return inactive, suspended
```

4.6 Rôles administratifs

Google Workspace permet de déléguer des rôles granulaires (admin messagerie, admin helpdesk, etc.). Un audit doit lister **tous les rôles administratifs** et vérifier le respect du principe du moindre privilège.

```
def audit_admin_roles(service):
    """Liste tous les rôles et leurs assignations."""
    roles = service.roles().list(customer="my_customer").execute()
    role_assignments = service.roleAssignments().list(
        customer="my_customer"
    ).execute()

    findings = {
        "custom_roles": [r for r in roles.get("items", []) if not
r.get("isSystemRole")],
        "assignments": role_assignments.get("items", []),
        "super_admin_assignments": [
            a for a in role_assignments.get("items", [])
            if a.get("roleId") == "SUPER_ADMIN_ROLE_ID"
        ],
    }
    return findings
```

---

5. Niveau 2 — Messagerie Gmail

5.1 Transferts automatiques

Le transfert automatique de messagerie est l'un des risques les plus graves : il crée une exfiltration de données en temps réel, silencieuse et persistante. À auditer sur **tous les comptes**.

```
def audit_gmail_forwarding(gmail_service, user_email):
    """Vérifie si un utilisateur a configuré un transfert automatique."""
    try:
        settings = gmail_service.users().settings().getAutoForwarding(
            userId=user_email
        ).execute()
        if settings.get("enabled"):
            return {
                "email": user_email,
                "forwarding_to": settings.get("emailAddress"),
                "disposition": settings.get("disposition"),
                "severity": "CRITIQUE",
            }
    except Exception as e:
        return {"email": user_email, "error": str(e)}
    return None
```

Contre-mesures :

- Désactiver le transfert automatique pour tous les utilisateurs via la politique admin
- Chemin : `admin.google.com → Apps → Gmail → Paramètres utilisateur → Transfert automatique de courrier`
- Activer des alertes sur toute activation de transfert

5.2 Délégation de boîte mail

La délégation permet à un utilisateur de gérer la boîte d'un autre. C'est une fonctionnalité légitime (assistante/dirigeant) mais qui crée une surface d'attaque si mal configurée.

```
def audit_gmail_delegation(gmail_service, user_email):
    """Liste les délégations actives sur un compte."""
    try:
        delegates = gmail_service.users().settings().delegates().list(
            userId=user_email
        ).execute()
        items = delegates.get("delegates", [])
        if items:
            external = [
                d for d in items
                if not d.get("delegateEmail", "").endswith(f"@{DOMAIN}")
            ]
            return {
                "email": user_email,
                "total_delegates": len(items),
                "external_delegates": external,
                "severity": "CRITIQUE" if external else "INFO",
            }
    except Exception:
        pass
    return None
```

5.3 Filtres Gmail avec actions de redirection

Les filtres Gmail peuvent rediriger silencieusement certains emails vers des adresses externes. Moins visibles que le transfert automatique, ils représentent un risque d'exfiltration ciblée.

```
def audit_gmail_filters(gmail_service, user_email):
    """Détecte les filtres avec redirection vers l'extérieur."""
    risky_filters = []
    try:
        filters = gmail_service.users().settings().filters().list(
            userId=user_email
        ).execute()
        for f in filters.get("filter", []):
            action = f.get("action", {})
            # Vérifier si le filtre redirige vers une adresse externe
            if "addLabelIds" in action or "forward" in str(action).lower():
                risky_filters.append({
                    "filter_id": f.get("id"),
                    "criteria": f.get("criteria", {}),
                    "action": action,
                })
    except Exception:
        pass
    return risky_filters
```

5.4 Tokens de session actifs (App Passwords)

Les App Passwords (anciennement "Less Secure Apps") permettent un accès IMAP/POP3 sans 2FA. Leur présence signale qu'un client mail ancien (Outlook, Thunderbird en mode IMAP basique) accède au compte en contournant la 2FA.

```
def audit_app_passwords(security_service, user_email):
    """Détecte les App Passwords (Less Secure Apps)."""
    try:
        asps = security_service.asps().list(userKey=user_email).execute()
        items = asps.get("items", [])
        if items:
            return {
                "email": user_email,
                "count": len(items),
                "apps": [a.get("name", "") for a in items],
                "oldest": min(a.get("creationTime", "") for a in items),
            }
    except Exception:
        pass
    return None
```

---

6. Niveau 3 — Google Drive et Partages

6.1 Partages publics ("Anyone with link")

Un fichier partagé avec "Tout le monde avec le lien" est accessible sans authentification. N'importe qui possédant le lien peut accéder au contenu, et ce lien peut circuler indéfiniment.

```
def audit_public_files(drive_service, user_email):
    """Trouve les fichiers accessibles publiquement."""
    public_files = []
    page_token = None
    while True:
        results = drive_service.files().list(
            q="'me' in owners",
            fields="nextPageToken, files(id, name, mimeType, permissions, size,
modifiedTime)",
            pageToken=page_token,
            pageSize=1000,
        ).execute()
        for f in results.get("files", []):
            for perm in f.get("permissions", []):
                if perm.get("type") == "anyone":
                    public_files.append({
                        "file_id": f.get("id"),
                        "name": f.get("name"),
                        "owner": user_email,
                        "permission": perm.get("role"),
                        "size": f.get("size", 0),
                        "modified": f.get("modifiedTime"),
                    })
        page_token = results.get("nextPageToken")
        if not page_token:
            break
    return public_files
```

6.2 Partages externes

Les partages avec des utilisateurs hors domaine doivent être inventoriés. Les points d'attention sont :

- Volume total de partages externes
- Destinataires récurrents (identifier les partenaires vs les erreurs)
- Permissions accordées (WRITER est plus risqué que READER)
- Absence de date d'expiration

```
def audit_external_shares(drive_service, domain):
    """Identifie tous les partages vers des utilisateurs extérieurs au domaine."""
    external = []
    page_token = None
    while True:
        results = drive_service.files().list(
            q="'me' in owners and sharedWithMe=false",
            fields="nextPageToken, files(id, name, permissions, mimeType)",
            pageToken=page_token,
            pageSize=1000,
        ).execute()
        for f in results.get("files", []):
            for perm in f.get("permissions", []):
                email = perm.get("emailAddress", "")
                if email and not email.endswith(f"@{domain}"):
                    external.append({
                        "file": f.get("name"),
                        "shared_with": email,
                        "role": perm.get("role"),
                        "type": perm.get("type"),
                        "expiration": perm.get("expirationTime"),
                    })
        page_token = results.get("nextPageToken")
        if not page_token:
            break
    return external
```

6.3 Analyse volumétrique des fichiers

L'analyse volumétrique permet d'identifier des anomalies : dumps de base de données, archives de données sensibles, doublons massifs, fichiers anciens oubliés.

```
SENSITIVE_KEYWORDS = [
    "bulletin", "paie", "salaire", "contrat", "password", "mdp", "credential",
    "dump", "backup", "bak", "export", "confidentiel", "secret", "budget",
    "bilan", "paierol", "rh", "ressources-humaines",
]

def analyze_file_sensitivity(filename):
    """Détecte si un fichier a un nom évocateur de données sensibles."""
    name_lower = filename.lower()
    return any(kw in name_lower for kw in SENSITIVE_KEYWORDS)

def audit_drive_files_deep(drive_service, user_email):
    """Analyse approfondie des fichiers Drive d'un utilisateur."""
    stats = {
        "total_files": 0,
        "total_size_bytes": 0,
        "sensitive_files": [],
        "large_files": [],      # > 100 MB
        "old_files": [],        # > 3 ans sans modification
        "sql_files": [],
        "archive_files": [],
    }
    # ... implémentation complète par pagination
    return stats
```

Indicateurs à surveiller :

INDICATEUR	SEUIL D'ALERTE	RISQUE
Fichiers SQL/BDD	> 0	Données production exposées
Fichiers anciens (> 3 ans)	> 30%	Violation Art. 5(1)(e) RGPD
Fichiers > 1 GB	Tout	Dump de données potentiel
Fichiers ".msg" Outlook	> 100	Emails archivés hors contrôle
Noms évocateurs RH/paie	> 0	Données personnelles sensibles

6.4 Service Accounts avec accès Drive

Des service accounts GCP (souvent créés pour des automatisations ou des applications tierces) peuvent avoir des accès en écriture sur des fichiers Drive. Ce vecteur est souvent oublié.

```
def detect_service_account_shares(permissions):
    """Détecte les partages avec des service accounts GCP."""
    sa_shares = []
    for perm in permissions:
        email = perm.get("emailAddress", "")
        if email.endswith(".gserviceaccount.com"):
            sa_shares.append({
                "service_account": email,
                "role": perm.get("role"),
                "project": email.split("@")[1].replace(".iam.gserviceaccount.com",
"""),
            })
    return sa_shares
```

---

7. Niveau 4 — DNS et Sécurité Email

7.1 SPF — Sender Policy Framework

Le SPF définit quels serveurs sont autorisés à envoyer des emails au nom du domaine.

```
import dns.resolver

def check_spf(domain):
    """Vérifie et analyse l'enregistrement SPF."""
    try:
        answers = dns.resolver.resolve(domain, "TXT")
        for rdata in answers:
            txt = str(rdata).strip('"')
            if txt.startswith("v=spf1"):
                return {
                    "record": txt,
                    "has_spf": True,
                    "hard_fail": txt.endswith("-all"),
                    "soft_fail": "~all" in txt,
                    "neutral": "?all" in txt,
                    "severity": "OK" if "-all" in txt else "PARTIEL" if "~all" in txt
else "CRITIQUE",
                    "recommendation": "Passer de ~all à -all pour le hard fail" if
"~all" in txt else None,
                }
    except Exception as e:
        return {"has_spf": False, "error": str(e), "severity": "CRITIQUE"}
```

Niveaux SPF et impact :

MÉCANISME	COMPORTEMENT	NIVEAU DE PROTECTION
`-all` (hard fail)	Emails non autorisés rejetés	✅ Optimal
`~all` (soft fail)	Emails marqués suspects	⚠️ Insuffisant
`?all` (neutral)	Aucune action	❌ Inutile
`+all`	Tout autorisé	🔴 CRITIQUE — à corriger immédiatement

7.2 DKIM — DomainKeys Identified Mail

DKIM ajoute une signature cryptographique aux emails sortants, permettant au destinataire de vérifier que le message n'a pas été altéré.

```
def check_dkim(domain, selector="google"):
    """Vérifie l'enregistrement DKIM pour le sélecteur spécifié."""
    try:
        dkim_record = f"{selector}._domainkey.{domain}"
        answers = dns.resolver.resolve(dkim_record, "TXT")
        for rdata in answers:
            txt = str(rdata).strip('"')
            if "v=DKIM1" in txt or "k=rsa" in txt:
                key_length = "2048" if len(txt) > 300 else "1024"
                return {
                    "selector": selector,
                    "record": txt[:100] + "...",
                    "key_length": key_length,
                    "severity": "OK" if key_length == "2048" else "PARTIEL",
                    "recommendation": "Clé RSA 1024 obsolète — migrer vers 2048 bits"
if key_length == "1024" else None,
                }
    except Exception:
        return {"selector": selector, "has_dkim": False, "severity": "CRITIQUE"}
```

7.3 DMARC — Domain-based Message Authentication

DMARC définit quoi faire des emails qui échouent SPF et DKIM, et où envoyer les rapports.

```
def check_dmarc(domain):
    """Vérifie et analyse la politique DMARC."""
    try:
        dmarc_record = f"_dmarc.{domain}"
        answers = dns.resolver.resolve(dmarc_record, "TXT")
        for rdata in answers:
            txt = str(rdata).strip('"')
            if txt.startswith("v=DMARC1"):
                policy = "none"
                if "p=reject" in txt:
                    policy = "reject"
                elif "p=quarantine" in txt:
                    policy = "quarantine"
                has_rua = "rua=" in txt
                return {
                    "record": txt,
                    "policy": policy,
                    "has_reporting": has_rua,
                    "severity": "OK" if policy == "reject" else "PARTIEL" if policy ==
"quarantine" else "CRITIQUE",
                    "recommendation": None if policy == "reject" else f"Passer de
p={policy} à p=reject",
                }
    except Exception:
        return {"has_dmarc": False, "severity": "CRITIQUE",
                "recommendation": "Configurer un enregistrement DMARC immédiatement"}
```

Chemin de maturation DMARC recommandé :

```
Phase 1 (Observation) : p=none ; rua=mailto:dmarc@votredomaine.com
    ↓ (2-4 semaines d'analyse des rapports)
Phase 2 (Durcissement) : p=quarantine ; pct=10 → pct=100
    ↓ (2-4 semaines de validation)
Phase 3 (Protection complète) : p=reject
```

7.4 Enregistrements MX et blacklists

```
def check_mx_blacklists(domain):
    """Vérifie si les serveurs MX sont sur des blacklists email."""
    blacklists = [
        "zen.spamhaus.org",
        "bl.spamcop.net",
        "dnsbl.sorbs.net",
        "b.barracudacentral.org",
        "dnsbl-1.uceprotect.net",
    ]
    try:
        mx_records = dns.resolver.resolve(domain, "MX")
        mx_host = str(sorted(mx_records, key=lambda r: r.preference)
[0].exchange).rstrip(".")
        ip = str(dns.resolver.resolve(mx_host, "A")[0])
        reversed_ip = ".".join(reversed(ip.split(".")))

        results = {"mx_ip": ip, "listed_on": [], "clean_on": []}
        for bl in blacklists:
            try:
                dns.resolver.resolve(f"{reversed_ip}.{bl}", "A")
                results["listed_on"].append(bl)
            except dns.resolver.NXDOMAIN:
                results["clean_on"].append(bl)
        return results
    except Exception as e:
        return {"error": str(e)}
```

---

8. Niveau 5 — Applications OAuth et Intégrations Tierces

8.1 Inventaire des tokens OAuth

Chaque fois qu'un utilisateur autorise une application tierce ("Connexion avec Google"), un token OAuth est créé. Ces tokens accordent des droits sur les données de l'utilisateur selon les scopes demandés.

```
SCOPE_RISK_LEVELS = {
    "https://mail.google.com/": ("CRITIQUE", "Accès complet Gmail (lecture + modification + suppression)"),
    "https://www.googleapis.com/auth/gmail.modify": ("ÉLEVÉ", "Modification des emails Gmail"),
    "https://www.googleapis.com/auth/drive": ("CRITIQUE", "Accès complet Google Drive"),
    "https://www.googleapis.com/auth/drive.file": ("MOYEN", "Fichiers Drive créés par l'app"),
    "https://www.googleapis.com/auth/admin.directory.user": ("CRITIQUE", "Gestion complète des utilisateurs"),
    "https://www.googleapis.com/auth/contacts": ("MOYEN", "Accès au carnet d'adresses"),
    "https://www.googleapis.com/auth/calendar": ("MOYEN", "Calendrier complet"),
    "https://www.googleapis.com/auth/admin.directory.group": ("ÉLEVÉ", "Gestion des groupes"),
}

def analyze_token_risks(tokens):
    """Analyse les scopes des tokens OAuth et identifie les risques."""
    risky_tokens = []
    for token in tokens:
        risks = []
        for scope in token.get("scopes", []):
            if scope in SCOPE_RISK_LEVELS:
                level, description = SCOPE_RISK_LEVELS[scope]
                risks.append({"scope": scope, "level": level, "description": description})
        if risks:
            risky_tokens.append({
                "user": token.get("email"),
                "app": token.get("displayText", token.get("clientId", "")),
                "client_id": token.get("clientId"),
                "risks": risks,
                "max_severity": "CRITIQUE" if any(r["level"] == "CRITIQUE" for r in risks) else "ÉLEVÉ",
            })
    return risky_tokens
```

8.2 Gouvernance des applications OAuth

Politique de whitelisting — Sans liste blanche, tout utilisateur peut autoriser n'importe quelle application OAuth. Cette configuration est la source de nombreux incidents de phishing OAuth ("consent phishing"). **Configuration recommandée dans admin.google.com :**

```
Admin Console → Sécurité → Contrôle des API → Contrôle d'accès aux applications Google
→ Activer : "Seules les apps approuvées par l'administrateur peuvent accéder aux
données"
→ Créer une liste blanche des applications autorisées
```

Signaux d'alerte sur un token OAuth :

- Application avec un `clientId` inconnu ou non documenté
- Scope `mail.google.com` accordé à une app non-Google
- Tokens anonymes (apps non identifiables publiquement)
- Nombre de tokens excessif pour un seul utilisateur (> 20)
- Applications de type "AI" avec accès Drive/Gmail complet

8.3 Cas particulier — Outils d'IA et données d'entreprise

Avec la prolifération des outils d'IA (Gemini, Copilot, Claude, ChatGPT via plugins), il est fréquent de trouver des tokens OAuth accordant accès à des volumes importants de données d'entreprise. Ces accès posent des questions :

- Les CGU du fournisseur permettent-elles l'entraînement sur les données ?
- Un DPA (Data Processing Agreement) existe-t-il avec le fournisseur d'IA ?
- Les données personnelles de tiers (clients, employés) transitent-elles par ces services ?

Contrôle à réaliser : Lister toutes les applications OAuth contenant "AI", "GPT", "Gemini", "Claude", "Assistant" dans leur nom, et vérifier les scopes accordés.

---

9. Niveau 6 — Appareils Mobiles et Points de Terminaison

9.1 Inventaire MDM

Google Workspace intègre une gestion basique des appareils mobiles (MDM). L'audit MDM révèle souvent des appareils fantômes ou des appareils BYOD non gérés.

```
def audit_mobile_devices(service):
    """Inventaire complet des appareils mobiles enregistrés."""
    devices = []
    page_token = None
    while True:
        result = service.mobiledevices().list(
            customerId="my_customer",
            pageToken=page_token,
            maxResults=100,
            projection="FULL",
        ).execute()
        devices.extend(result.get("mobiledevices", []))
        page_token = result.get("nextPageToken")
        if not page_token:
            break

    findings = {
        "total": len(devices),
        "by_status": {},
        "old_devices": [],      # Pas de sync depuis > 6 mois
        "risky_devices": [],    # BYOD sur comptes admins
        "unencrypted": [],
    }
    for device in devices:
        last_sync = device.get("lastSync", "")
        owner = device.get("email", "")
        status = device.get("status", "")
        # ... analyse
    return findings
```

Points critiques :

- Appareils non synchronisés depuis > 6 mois : fantômes à supprimer
- Appareils BYOD appartenant à des Super Admins : risque élevé
- Appareils sans chiffrement activé
- Absence de PIN obligatoire
- Absence de remote wipe configuré

9.2 Configuration MDM recommandée

Politique minimale pour les appareils accédant aux données d'entreprise :

CONTRÔLE	STANDARD	POUR ADMINS
PIN/biométrie	Obligatoire (6+ chiffres)	Obligatoire (8+ chiffres)
Chiffrement	Obligatoire	Obligatoire
Mise à jour OS	Recommandée	Forcée sous 30 jours
Remote wipe	Activé	Activé + testé annuellement
Timeout verrouillage	5 minutes	2 minutes
Appareils rootés/jailbreakés	Bloqués	Bloqués

10. Niveau 7 — Groupes et Politiques de Partage

10.1 Audit des groupes

```
def audit_groups(service, groups_settings_service):
    """Analyse la sécurité des groupes Google."""
    findings = {
        "total_groups": 0,
        "external_members": [],
        "orphan_groups": [],    # Sans owner
        "public_groups": [],    # Accessibles par tous
    }

    page_token = None
    while True:
        result = service.groups().list(
            domain=DOMAIN,
            maxResults=200,
            pageToken=page_token,
        ).execute()
        groups = result.get("groups", [])
        findings["total_groups"] += len(groups)

        for group in groups:
            group_email = group.get("email", "")
            members = service.members().list(
                groupKey=group_email
            ).execute().get("members", [])

            # Membres externes
            ext = [m for m in members if not m.get("email",
"").endswith(f"@{DOMAIN}")]
            if ext:
                findings["external_members"].append({
                    "group": group_email,
                    "external": ext,
                })

            # Groupes orphelins (sans owner)
            owners = [m for m in members if m.get("role") == "OWNER"]
            if not owners:
                findings["orphan_groups"].append(group_email)

        page_token = result.get("nextPageToken")
        if not page_token:
            break
    return findings
```

10.2 Politiques de partage global

Paramètres à vérifier dans admin.google.com → Drive → Partage :

PARAMÈTRE	VALEUR RECOMMANDÉE	RISQUE SI NON CONFIGURÉ
Partage hors domaine	Désactivé ou approuvé	Fuite de données non contrôlée
"Anyone with link"	Désactivé pour non-admins	Fichiers publics non maîtrisés
Avertissement partage externe	Activé	Partages accidentels
Expiration des liens	30-90 jours	Liens actifs indéfiniment
Domaines de confiance	Liste explicite	Partages vers n'importe qui

11. Niveau 8 — Journalisation, Alertes et Surveillance

11.1 Alert Center

L'Alert Center Google Workspace centralise les alertes de sécurité du tenant.

```
def audit_alert_center(service, days=90):
    """Récupère et analyse les alertes du Alert Center."""
    from datetime import datetime, timezone, timedelta
    import json

    start = (datetime.now(timezone.utc) - timedelta(days=days)).isoformat()
    alerts = []
    page_token = None

    while True:
        result = service.alerts().list(
            filter=f'createTime > "{start}"',
            pageToken=page_token,
        ).execute()
        alerts.extend(result.get("alerts", []))
        page_token = result.get("nextPageToken")
        if not page_token:
            break

    HIGH_SEVERITY_TYPES = {
        "Suspicious login": "CRITIQUE",
        "Phishing email reported": "ÉLEVÉ",
        "Malware detected": "CRITIQUE",
        "Super admin password reset": "ÉLEVÉ",
        "User suspended (spam)": "ÉLEVÉ",
        "Government-backed attack": "CRITIQUE",
        "Data exfiltration": "CRITIQUE",
    }

    categorized = {}
    for alert in alerts:
        alert_type = alert.get("type", "Unknown")
        severity = HIGH_SEVERITY_TYPES.get(alert_type, "INFO")
        categorized.setdefault(alert_type, []).append({
            "id": alert.get("alertId"),
            "time": alert.get("createTime"),
            "severity": severity,
        })
    return {"total": len(alerts), "by_type": categorized}
```

11.2 Logs de connexion

```
def audit_login_logs(service, days=30):
    """Analyse les logs de connexion pour détecter des comportements suspects."""
    from datetime import datetime, timezone, timedelta

    start = (datetime.now(timezone.utc) - timedelta(days=days)).isoformat() + "Z"
    findings = {
        "failed_logins": {},
        "suspicious_logins": [],
        "login_by_country": {},
        "anomalous_ips": [],
    }

    page_token = None
    while True:
        result = service.activities().list(
            userKey="all",
            applicationName="login",
            startTime=start,
            maxResults=1000,
            pageToken=page_token,
        ).execute()

        for activity in result.get("items", []):
            actor = activity.get("actor", {}).get("email", "unknown")
            ip = activity.get("ipAddress", "")

            for event in activity.get("events", []):
                name = event.get("name", "")
                if name == "login_failure":
                    findings["failed_logins"].setdefault(actor, 0)
                    findings["failed_logins"][actor] += 1
                elif name == "suspicious_login":
                    findings["suspicious_logins"].append({
                        "user": actor,
                        "ip": ip,
                        "time": activity.get("id", {}).get("time"),
                    })

        page_token = result.get("nextPageToken")
        if not page_token:
            break
    return findings
```

Comportements suspects à surveiller :

- Connexions depuis des pays inhabituels (géolocalisation des IPs)
- Connexions à des heures anormales (2h-5h du matin pour une organisation française)
- Pic d'échecs de connexion sur un compte (> 5 en 24h = brute force potentiel)
- Connexions simultanées depuis des localisations géographiquement impossibles
- Super Admins avec des échecs de connexion répétés

11.3 Logs d'administration

```
def audit_admin_logs(service, days=30):
    """Analyse les actions d'administration des 30 derniers jours."""
    SENSITIVE_ACTIONS = [
        "CHANGE_USER_PASSWORD",
        "CHANGE_RECOVERY_EMAIL",
        "CHANGE_RECOVERY_PHONE",
        "GRANT_ADMIN_PRIVILEGE",
        "REVOKE_ADMIN_PRIVILEGE",
        "DELETE_USER",
        "SUSPEND_USER",
        "CREATE_SERVICE_ACCOUNT",
        "CHANGE_APPLICATION_SETTING",
        "TOGGLE_SERVICE_ENABLED",
        "DOMAIN_WIDE_DELEGATION_CHANGE",
    ]
    # Implémenter via Reports API applicationName="admin"
    pass
```

11.4 Alertes recommandées à configurer

Dans admin.google.com → Sécurité → Alertes → Créer des alertes :

TYPE D'ALERTE	DÉCLENCHEUR	PRIORITÉ
Connexion suspecte	Toute connexion suspecte détectée	🔴 Critique
Super Admin modifié	Ajout/suppression de Super Admin	🔴 Critique
Transfert Gmail activé	Activation d'un forwarding	🔴 Critique
Partage "anyone" créé	Fichier partagé publiquement	🟠 Élevé
App OAuth autorisée	Nouvelle autorisation d'app	🟠 Élevé
Compte suspendu	Suspension automatique (spam)	🟠 Élevé
Réinitialisation 2FA admin	Tout admin	🟠 Élevé

12. Niveau 9 — Reconnaissance Passive et Threat Intelligence

12.1 Certificate Transparency (crt.sh)

Les Certificate Transparency Logs enregistrent tous les certificats TLS émis. Ils permettent de découvrir des sous-domaines non documentés.

```
import requests

def check_certificate_transparency(domain):
    """Découverte de sous-domaines via Certificate Transparency Logs."""
    try:
        resp = requests.get(
            f"https://crt.sh/?q=%.{domain}&output=json",
            timeout=15,
            headers={"Accept": "application/json"},
        )
        certs = resp.json()
        subdomains = set()
        wildcards = []

        for cert in certs:
            for name in cert.get("name_value", "").split("\n"):
                name = name.strip().lower()
                if name.endswith(f".{domain}") or name == domain:
                    if name.startswith("*."):
                        wildcards.append(name)
                    else:
                        subdomains.add(name)
        return {
            "subdomains": sorted(subdomains),
            "wildcard_certs": wildcards,
            "total": len(subdomains),
        }
    except Exception as e:
        return {"error": str(e)}
```

Sous-domaines à risque à rechercher :

SOUS-DOMAINE	RISQUE
`dev.`, `staging.`, `test.*`	Environnements non sécurisés, PHP/frameworks obsolètes
`vpn.`, `remote.`	Points d'entrée réseau exposés
`admin.`, `portal.`	Interfaces d'administration exposées
`api.*`	APIs potentiellement non authentifiées
`mail.`, `webmail.`	Clients mail alternatifs

12.2 Google Dorks — Recherche de données exposées

Les Google Dorks permettent de chercher des données spécifiques indexées par Google. À exécuter manuellement dans un navigateur (les requêtes automatiques violent les ToS Google).

```
# Documents confidentiels indexés
site:votredomaine.com filetype:pdf "confidentiel"
site:votredomaine.com filetype:xlsx OR filetype:csv

# Documents Google Drive publics
site:drive.google.com "votredomaine.com"
site:docs.google.com "votredomaine.com"

# Leaks de credentials
"votredomaine.com" "api_key" OR "secret_key" OR "password"
intext:"@votredomaine.com" site:pastebin.com

# Code source exposé
site:github.com "votredomaine.com"
site:gitlab.com "votredomaine.com"

# Dumps et backups
"votredomaine.com" filetype:sql OR filetype:bak OR filetype:dump
```

12.3 Shodan — Exposure Internet

Shodan indexe les services Internet exposés. Sans clé API, l'API gratuite `internetdb.shodan.io` donne des informations de base.

```
def check_shodan_free(ip):
    """Interroge l'API Shodan InternetDB (gratuite, sans clé)."""
    try:
        resp = requests.get(
            f"https://internetdb.shodan.io/{ip}",
            timeout=10,
        )
        if resp.status_code == 200:
            data = resp.json()
            return {
                "ports": data.get("ports", []),
                "hostnames": data.get("hostnames", []),
                "cpes": data.get("cpes", []),  # Technologies détectées
                "vulns": data.get("vulns", []),
                "tags": data.get("tags", []),  # ex: "eol-product"
            }
    except Exception as e:
        return {"error": str(e)}
```

Tags Shodan critiques :

- `eol-product` : logiciel en fin de vie (PHP 7.x, Apache 2.2, etc.)
- `self-signed` : certificat auto-signé
- `starttls` : chiffrement opportuniste (non forcé)
- `vpn` : service VPN exposé

12.4 HIBP — Have I Been Pwned

HIBP permet de vérifier si des emails d'une organisation apparaissent dans des bases de données de credentials compromis.

```
def check_hibp_domain(domain, api_key):
    """Vérifie via l'API HIBP Domain Search tous les emails d'un domaine."""
    headers = {
        "hibp-api-key": api_key,
        "User-Agent": "Security-Audit-Tool",
    }
    try:
        resp = requests.get(
            f"https://haveibeenpwned.com/api/v3/breacheddomain/{domain}",
            headers=headers,
            timeout=15,
        )
        if resp.status_code == 200:
            return resp.json()  # {email: [breach1, breach2, ...]}
        elif resp.status_code == 404:
            return {}  # Aucun email compromis
        elif resp.status_code == 402:
            return {"error": "Clé API requise — https://haveibeenpwned.com/API/Key"}
    except Exception as e:
        return {"error": str(e)}
```

Interprétation des breaches HIBP :

TYPE DE BREACH	RISQUE	ACTION
Stealer Logs (Naz.API, ALIEN TXTBASE)	🔴 **MAXIMAL**	Mots de passe en clair potentiels — rotation immédiate
Credential stuffing lists	🔴 CRITIQUE	Combinaisons email/password testées activement
Collection #1/#2/#3	🔴 CRITIQUE	Compilation massive de credentials
Exploit.in, Cit0day	🔴 CRITIQUE	Bases de credentials actifs
LinkedIn Scraped	🟡 MOYEN	Données publiques — risque spear-phishing
PDL Data Enrichment	🟡 MOYEN	Données de contact — risque phishing
Deezer, Nitro, Canva	🟡 MOYEN	Services grand public — réutilisation mot de passe

Automatisation de la vérification HIBP

Pour les organisations avec une clé API HIBP, la vérification peut être automatisée sur l'ensemble du domaine :

```
import requests, time

def hibp_domain_search(domain, api_key):
    """
    HIBP Domain Search — retourne tous les emails compromis du domaine.
    Nécessite un abonnement HIBP (clé API).
    """
    headers = {
        "hibp-api-key": api_key,
        "User-Agent": "WorkspaceSecurityAudit/1.0",
    }
    resp = requests.get(
        f"https://haveibeenpwned.com/api/v3/breacheddomain/{domain}",
        headers=headers,
        timeout=30,
    )
    if resp.status_code == 200:
        # Retourne {email: [breach_name1, breach_name2, ...]}
        data = resp.json()
        # Classifier par sévérité
        stealer_sources = {"Naz.API", "ALIEN TXTBASE Stealer Logs",
                           "Exploit.in", "Operation Endgame", "Cit0day"}
        critical = {
            email: breaches for email, breaches in data.items()
            if any(s in breaches for s in stealer_sources)
        }
        return {"all_compromised": data, "critical_stealer": critical}
    return {"error": f"HTTP {resp.status_code}"}

def hibp_check_recovery_emails(recovery_emails, api_key):
    """Vérifie les emails de récupération individuellement."""
    results = []
    for entry in recovery_emails:
        resp = requests.get(
            f"https://haveibeenpwned.com/api/v3/breachedaccount/
{entry['recovery_email']}",
            headers={"hibp-api-key": api_key, "User-Agent": "Audit/1.0"},
            timeout=10,
        )
        if resp.status_code == 200:
            breaches = resp.json()
            results.append({
                "cyrias_account": entry["email"],
```

```
                "recovery_email": entry["recovery_email"],
                "breach_count": len(breaches),
                "breaches": [b["Name"] for b in breaches],
                "severity": "CRITIQUE" if len(breaches) > 5 else "MOYEN",
            })
        time.sleep(1.5)  # Rate limiting HIBP : 1 requête/1.5s
    return results
```

Corrélation critique : Toujours croiser les emails compromis HIBP avec :

1. Leur rôle dans l'organisation (Super Admin = risque maximal)
2. La présence d'un email de récupération externe
3. Le type de breach (stealer log vs. scraping public)

---

13. Niveau 10 — Conformité Réglementaire

13.1 Conformité RGPD

Le RGPD impose des obligations précises sur la gestion des données personnelles. Un audit Google Workspace doit évaluer la conformité article par article.

Articles RGPD directement impactés par la configuration Workspace :

ARTICLE	PRINCIPE	CE QU'ON VÉRIFIE
Art. 5(1)(e)	Limitation de la conservation	Fichiers anciens > 3 ans sans politique de purge
Art. 5(1)(f)	Intégrité et confidentialité	Fichiers publics, partages non contrôlés
Art. 9 & 32	Données sensibles	Bulletins de paie, données médicales, RH dans Drive non chiffré
Art. 17	Droit à l'effacement	Procédure offboarding documentée
Art. 25	Privacy by design	Paramètres de partage par défaut restrictifs
Art. 28	Sous-traitants	DPA avec Google, Zendesk, Ivalua et autres SaaS
Art. 32	Sécurité du traitement	2FA, chiffrement, transferts email
Art. 33 & 34	Notification de violation	Procédure de réponse aux incidents documentée
Art. 35	DPIA	Traitements à risque élevé identifiés
Art. 44-49	Transferts internationaux	SaaS US — Privacy Shield remplacé, clauses contractuelles

```
def calculate_gdpr_score(findings):
    """Calcule un score de conformité RGPD basé sur les findings."""
    score = 100
    violations = []

    # Art. 5(1)(e) — Vieux fichiers
    old_ratio = findings.get("drive", {}).get("old_files_ratio", 0)
    if old_ratio > 0.3:
        score -= 10
        violations.append(("Art. 5(1)(e)", "MOYEN", f"{old_ratio:.0%} de fichiers > 3
ans"))

    # Art. 5(1)(f) — Fichiers publics
    public_files = findings.get("drive", {}).get("public_files_count", 0)
    if public_files > 0:
        score -= 15
        violations.append(("Art. 5(1)(f)", "ÉLEVÉ", f"{public_files} fichiers
publics"))

    # Art. 32 — Transfert email
    if findings.get("gmail", {}).get("has_forwarding"):
        score -= 20
        violations.append(("Art. 32", "CRITIQUE", "Transfert automatique email vers
tiers"))

    # Art. 9 & 32 — Données RH
    sensitive_files = findings.get("drive", {}).get("sensitive_files_count", 0)
    if sensitive_files > 100:
        score -= 20
        violations.append(("Art. 9 & 32", "CRITIQUE", f"{sensitive_files} fichiers
sensibles non chiffrés"))

    return max(0, score), violations
```

13.2 Conformité NIS2

La directive NIS2 s'applique aux entités importantes et essentielles, notamment les ESN/prestataires IT. Les obligations clés à auditer :

OBLIGATION NIS2	VÉRIFICATION WORKSPACE	OUTIL D'AUDIT
Politique de sécurité des SI	PSSI documentée et appliquée	Revue documentaire
Gestion des incidents (72h)	Alert Center + procédure d'escalade	Alert Center API
Continuité d'activité	PCA/PRA documenté	Revue documentaire
Sécurité chaîne d'approvisionnement	Apps OAuth validées, sous-traitants DPA	OAuth audit
Contrôle d'accès MFA résistant phishing	FIDO2 pour admins	Directory API
Chiffrement	Données sensibles isolées et chiffrées	Drive API
Journalisation et surveillance	SIEM + rétention logs	Reports API

13.3 CIS Benchmark Google Workspace

Le Center for Internet Security publie un benchmark de sécurité spécifique à Google Workspace. Voir la Section 16 pour le détail complet.

Score CIS = nombre de contrôles conformes / total des contrôles évalués

Un score < 50% indique une posture de sécurité insuffisante.

---

14. Automatisation et Outillage

14.1 Architecture du framework d'audit

```
audit-workspace/
├── config.py              # Configuration centralisée
├── auth.py                # Authentification Service Account
├── main.py                # Orchestrateur principal
├── run_full_audit.py      # Exécution de tous les modules
│
├── audit_users.py         # Niveau 1 — IAM
├── audit_gmail.py         # Niveau 2 — Gmail
├── audit_drive.py         # Niveau 3 — Drive (partages)
├── audit_drive_files.py   # Niveau 3 — Drive (fichiers)
├── audit_dns.py           # Niveau 4 — DNS
├── audit_oauth_apps.py    # Niveau 5 — OAuth
├── audit_less_secure.py   # Niveau 1 — Tokens + Recovery
├── audit_devices.py       # Niveau 6 — MDM
├── audit_groups.py        # Niveau 7 — Groupes
├── audit_alerts.py        # Niveau 8 — Alert Center
├── audit_admin_logs.py    # Niveau 8 — Logs admin
├── audit_login.py         # Niveau 8 — Logs connexion
├── audit_calendar.py      # Niveau 2 — Calendar
├── audit_recon.py         # Niveau 9 — Recon passive
├── audit_hibp.py          # Niveau 9 — HIBP
├── audit_darkweb.py       # Niveau 9 — Dark web
├── audit_compliance.py    # Niveau 10 — RGPD/NIS2
│
└── reports/
    ├── AUDIT_SECURITE_YYYYMMDD.md
    ├── extra_modules.json
    ├── levels_1_4.json
    ├── login_audit.json
    ├── calendar_audit.json
    └── drive_files_analysis.json
```

14.2 Orchestrateur principal

```
# run_full_audit.py
import os, sys, json
os.environ["PYTHONUTF8"] = "1"

from auth import get_directory_service, get_reports_service
from audit_users import audit_all_users
from audit_gmail import audit_gmail_all_users
from audit_drive import audit_drive_shares
from audit_drive_files import audit_drive_files_deep
from audit_dns import check_all_dns
from audit_oauth_apps import audit_oauth_apps
from audit_less_secure import audit_user_security_settings, audit_user_recovery_info
from audit_devices import audit_mobile_devices
from audit_groups import audit_groups
from audit_alerts import audit_alert_center
from audit_admin_logs import audit_admin_logs
from audit_login import audit_login_activity
from audit_calendar import audit_all_calendars
from audit_recon import check_certificate_transparency, check_mx_blacklists
from audit_compliance import assess_rgpd_compliance, assess_nis2_compliance

def run_full_audit():
    """Exécute l'audit complet en lecture seule."""
    print("[*] Démarrage de l'audit Google Workspace...")

    service = get_directory_service()
    reports_service = get_reports_service()

    # Récupération de la liste des utilisateurs
    users = get_all_users(service)
    emails = [u["primaryEmail"] for u in users]
    print(f"[*] {len(emails)} utilisateurs trouvés")

    results = {}

    # Niveaux 1-8 : APIs Google
    results["users"] = audit_all_users(service)
    results["gmail"] = audit_gmail_all_users(emails)
    results["drive"] = audit_drive_shares(emails)
    results["drive_files"] = audit_drive_files_deep(emails[:10])  # Limiter pour les
premiers
    results["dns"] = check_all_dns(DOMAIN)
    results["oauth"] = audit_oauth_apps()
```

```
    results["security"] = audit_user_security_settings(emails)
    results["recovery"] = audit_user_recovery_info(emails)
    results["devices"] = audit_mobile_devices(service)
    results["groups"] = audit_groups(service)
    results["alerts"] = audit_alert_center(alert_service)
    results["admin_logs"] = audit_admin_logs(reports_service)
    results["logins"] = audit_login_activity(reports_service)
    results["calendar"] = audit_all_calendars(emails)

    # Niveau 9 : Recon et Threat Intel
    results["recon"] = {
        "crt": check_certificate_transparency(DOMAIN),
        "blacklists": check_mx_blacklists(DOMAIN),
    }

    # Niveau 10 : Conformité
    results["rgpd"] = assess_rgpd_compliance(results)
    results["nis2"] = assess_nis2_compliance()

    # Sauvegarde
    with open(f"reports/audit_{DOMAIN}_{datetime.now().strftime('%Y%m%d')}.json", "w")
as f:
        json.dump(results, f, ensure_ascii=False, indent=2, default=str)

    print("[+] Audit terminé.")
    return results
```

14.3 Gestion des erreurs courantes

ERREUR	CAUSE	SOLUTION
`invalid_grant`	Horloge désynchronisée	Resynchroniser NTP
`unauthorized_client`	Scope absent de la délégation	Ajouter le scope dans admin.google.com
`insufficientPermissions`	API non activée dans GCP	Activer l'API dans Cloud Console
`notACalendarUser`	Utilisateur sans Calendar	Ignorer — pas une erreur de sécurité
`UnicodeEncodeError`	Terminal Windows cp1252	Utiliser `python -X utf8` ou `PYTHONUTF8=1`
`403 Forbidden`	Service Account sans impersonation	Vérifier la Domain-Wide Delegation

15. Rapport et Feuille de Route

15.1 Structure du rapport

Un rapport d'audit Google Workspace complet doit contenir :

```
1. Synthèse Exécutive (1 page)
   - Score global de sécurité
   - Top 5 risques immédiats
   - Métriques clés

2. Findings par niveau de sévérité
   - CRITIQUE : action sous 48h
   - ÉLEVÉ : action sous 7 jours
   - MOYEN : action sous 30 jours
   - INFORMATIF : à planifier

3. Audits détaillés par domaine
   - IAM, Gmail, Drive, DNS, OAuth, MDM, Groupes, Logs

4. Conformité
   - RGPD (score + violations par article)
   - NIS2 (matrice d'obligations)
   - CIS Benchmark (score détaillé)

5. Chemins d'attaque (Kill Chains)
   - Scénarios réalistes basés sur les findings

6. Feuille de route de sécurisation
   - Matrice priorité / effort / impact
   - Planning par semaine/mois

7. Annexes
   - Données brutes
   - Méthodologie
   - Outils utilisés
```

15.2 Scoring de sécurité

```
def calculate_security_score(findings):
    """Calcule un score de sécurité global de 0 à 100."""
    score = 100
    deductions = {
        "no_2fa_users": (-5, "par utilisateur sans 2FA"),
        "public_files": (-10, "par fichier public"),
        "active_forwarding": (-20, "transfert email actif"),
        "external_sa_writer": (-15, "service account externe WRITER"),
        "super_admin_external_recovery": (-10, "par Super Admin avec récup externe"),
        "php_eol_server": (-8, "serveur PHP EOL exposé"),
    }
    # ... calcul basé sur les findings
    return max(0, score)
```

Grille d'interprétation du score :

SCORE	NIVEAU	SIGNIFICATION
90-100	✅ Excellent	Posture mature, risques résiduels mineurs
75-89	🟡 Bon	Bonne base, quelques gaps à combler
60-74	🟠 Acceptable	Risques notables — roadmap requise
40-59	🔴 Insuffisant	Vulnérabilités significatives — actions urgentes
< 40	🔴 Critique	Posture dangereuse — intervention immédiate

15.3 Matrice de priorisation

```
Priorité = f(Impact, Probabilité, Effort de correction)

P1 (Immédiat) : Impact CRITIQUE + Probabilité HAUTE + Effort FAIBLE
P2 (J+7)      : Impact ÉLEVÉ  + Probabilité MOYENNE + Effort MOYEN
P3 (J+30)     : Impact MOYEN  + Probabilité BASSE   + Effort ÉLEVÉ
P4 (J+90)     : Gouvernance, processus, formation
```

15.4 Chemins d'attaque types

Kill Chain 1 — Exfiltration via forwarding Gmail

```
Phishing ou insider malveillant
→ Activation d'un transfert automatique Gmail
→ Copie en temps réel de tous les emails vers un tiers
→ Fuite continue de données confidentielles
→ Impact : RGPD, concurrentiel, contractuel
```

Kill Chain 2 — Account Takeover via email de récupération

```
Email de récupération présent dans un stealer log
→ Attaquant accède au compte Gmail/Yahoo de récupération
→ Procédure "mot de passe oublié" sur le compte Workspace
→ Bypass de la 2FA via le code envoyé sur l'email compromis
→ Accès complet au compte (emails, Drive, calendrier)
→ Escalade si compte Super Admin
```

Kill Chain 3 — OAuth Consent Phishing

```
Email malveillant : "Autorisez cette app pour accéder à votre planning"
→ Utilisateur clique et autorise l'app OAuth
→ L'app obtient les scopes accordés (Gmail, Drive)
→ Exfiltration silencieuse continue
→ Aucune trace dans les logs Gmail standards
→ Persiste même après changement de mot de passe
```

Kill Chain 4 — Compromission via service account externe

```
Clé de service account externe volée ou fuitée
→ Accès aux fichiers Drive partagés avec ce SA
→ Lecture de données confidentielles
→ Injection de code malveillant dans des scripts partagés
→ Exécution par un employé → RCE sur son poste
→ Pivot vers le réseau interne
```

---

16. Référentiel CIS Benchmark Google Workspace

Le CIS Benchmark for Google Workspace définit 40+ contrôles de sécurité. Voici les contrôles les plus critiques avec leur méthode de vérification :

Section 1 — Gestion des Identités et des Accès

CIS ID	CONTRÔLE	MÉTHODE DE VÉRIFICATION	CRITICITÉ
1.1.1	≤ 4 Super Admins	`users.list` + filtre `isAdmin=true`	Haute
1.1.2	Comptes admin dédiés	Vérifier si les admins ont des emails dédiés non-quotidiens	Haute
1.1.3	2FA obligatoire (enforced)	Attribut `isEnforcedIn2Sv` sur tous les comptes	Haute
1.1.4	Clés FIDO2 pour Super Admins	Absence de clé hardware = non conforme	Haute
1.1.5	Email récupération interne	`recoveryEmail` doit se terminer par `@domaine.com`	Haute
1.1.6	Politique mot de passe min. 12 car.	`admin.google.com → Sécurité → Mots de passe`	Moyenne
1.1.7	Expiration mots de passe désactivée si MFA fort	Dépend de la politique de mot de passe	Faible
1.2.1	Context-Aware Access pour admins	`admin.google.com → Sécurité → Context-Aware Access`	Haute

Section 2 — Google Drive

CIS ID	CONTRÔLE	MÉTHODE DE VÉRIFICATION	CRITICITÉ
2.1	Restreindre partage hors domaine	Paramètres Drive dans admin.google.com	Haute
2.2	Désactiver "Anyone with link"	API Drive, permissions type=anyone	Haute
2.3	Avertissement avant partage externe	Paramètre admin.google.com	Moyenne
2.4	Expiration automatique des liens	Paramètre global Drive	Moyenne
2.5	Audit Drive régulier documenté	Processus organisationnel	Faible
2.6	DLP sur données sensibles	Google DLP ou solution tierce	Haute

Section 3 — Gmail

CIS ID	CONTRÔLE	MÉTHODE DE VÉRIFICATION	CRITICITÉ
3.1	Désactiver transfert automatique	API Gmail, `getAutoForwarding()`	Critique
3.2	Désactiver délégation de boîte	API Gmail, `delegates().list()`	Haute
3.3	Anti-spam/phishing renforcé	Paramètres Gmail avancés	Haute
3.4	SMTP relay authentifié	`admin.google.com → Gmail → Routing`	Moyenne
3.5	Désactiver Less Secure Apps	API, `asps().list()`	Haute

Section 4 — Sécurité Email DNS

CIS ID	CONTRÔLE	MÉTHODE	CRITICITÉ
4.1	SPF `-all` (hard fail)	`dns.resolver.resolve(domain, "TXT")`	Haute
4.2	DKIM configuré (2048 bits)	`dns.resolver.resolve("google._domainkey.domain", "TXT")`	Haute
4.3	DMARC `p=reject`	`dns.resolver.resolve("_dmarc.domain", "TXT")`	Critique
4.4	DMARC reporting (rua)	Présence `rua=` dans l'enregistrement	Moyenne
4.5	MX uniquement Google	Vérifier qu'aucun MX parasite n'existe	Haute

Section 5 — Applications Tierces

CIS ID	CONTRÔLE	MÉTHODE	CRITICITÉ
5.1	Liste blanche d'applications	`admin.google.com → Sécurité → API Controls`	Haute
5.2	Bloquer apps non vérifiées	Paramètre admin.google.com	Haute
5.3	Revue trimestrielle des OAuth	Processus organisationnel	Moyenne
5.4	Alertes nouvelles autorisations OAuth	Alert Center config	Haute

Section 6 — Appareils Mobiles

CIS ID	CONTRÔLE	MÉTHODE	CRITICITÉ
6.1	MDM activé et enforced	`mobiledevices().list()`	Haute
6.2	Chiffrement obligatoire	Attribut `deviceCompromisedStatus`	Haute
6.3	PIN biométrique obligatoire	Politique MDM	Haute
6.4	Remote wipe disponible	Fonctionnalité MDM Workspace	Moyenne

Section 7 — Groupes

CIS ID	CONTRÔLE	MÉTHODE	CRITICITÉ
7.1	Pas de membres externes non justifiés	`members().list()` par groupe	Haute
7.2	Chaque groupe a un owner	Vérifier `role=OWNER` dans membres	Moyenne
7.3	Restriction création groupes aux admins	Paramètre admin.google.com	Faible

Section 8 — Journalisation

CIS ID	CONTRÔLE	MÉTHODE	CRITICITÉ
8.1	Logs d'audit activés	Reports API accessible	Haute
8.2	Rétention ≥ 1 an	Google Vault ou export SIEM	Haute
8.3	Alertes sur événements critiques	Alert Center configuration	Haute
8.4	Export logs vers SIEM	BigQuery, Chronicle, Splunk	Haute
8.5	Procédure réponse incidents	Processus organisationnel documenté	Haute

17. Annexes Techniques

Annexe A — Scopes OAuth par module d'audit

MODULE	SCOPES REQUIS
Utilisateurs & IAM	`admin.directory.user.readonly`, `admin.directory.rolemanagement.readonly`
Gmail forwarding/ delegation	`gmail.settings.basic` (par utilisateur impersonifié)
Google Drive	`drive.readonly`
Tokens OAuth	`admin.directory.user.security`
Appareils MDM	`admin.directory.device.mobile.readonly`
Groupes	`admin.directory.group.readonly`, `apps.groups.settings`
Logs admin/connexion	`admin.reports.audit.readonly`
Alert Center	`apps.alerts`
Calendriers	`calendar.readonly` (par utilisateur impersonifié)

Annexe B — Commandes de référence rapide

```
# Lancer l'audit complet
python -X utf8 run_full_audit.py

# Lancer un module spécifique
python -X utf8 -c "from audit_users import *; print(audit_all_users())"

# Synchroniser l'horloge (Windows)
w32tm /resync /force

# Vérifier DNS manuellement
python -c "import dns.resolver; print(dns.resolver.resolve('cyrias.com','TXT'))"

# Lister les fichiers publics Drive d'un utilisateur
python -X utf8 -c "
from audit_drive import audit_public_files
from auth import get_drive_service
service = get_drive_service('user@domain.com')
print(audit_public_files(service, 'user@domain.com'))
"
```

Annexe C — Checklist d'audit condensée

Avant l'audit :

- [ ] Service Account créé et clé JSON générée
- [ ] APIs GCP activées (Admin SDK, Gmail, Drive, Calendar, Alert Center)
- [ ] Domain-Wide Delegation configurée sans espaces dans les scopes
- [ ] Horloge système synchronisée
- [ ] Environnement Python configuré (PYTHONUTF8=1)

Pendant l'audit — Module IAM :

- [ ] Nombre de Super Admins (≤ 4 recommandé)
- [ ] 2FA enrollée et enforced sur tous les comptes
- [ ] Emails de récupération internes ou absents
- [ ] Comptes inactifs > 90 jours identifiés
- [ ] Comptes suspendus avec offboarding documenté

Pendant l'audit — Module Gmail :

- [ ] Aucun transfert automatique actif
- [ ] Aucune délégation externe
- [ ] Aucun App Password actif

- [ ] Filtres Gmail avec actions de redirection revus

Pendant l'audit — Module Drive :

- [ ] Fichiers publics ("anyone with link") inventoriés
- [ ] Partages externes documentés et justifiés
- [ ] Service accounts tiers avec accès Drive identifiés
- [ ] Fichiers sensibles inventoriés (RH, paie, BDD)

Pendant l'audit — Module DNS :

- [ ] SPF `-all` (hard fail)
- [ ] DKIM actif (2048 bits recommandé)
- [ ] DMARC `p=reject` avec reporting
- [ ] IP MX non listées sur blacklists

Pendant l'audit — Module OAuth :

- [ ] Volume de tokens par utilisateur
- [ ] Scopes accordés (Gmail complet, Drive complet = CRITIQUE)
- [ ] Applications AI avec accès données d'entreprise

Pendant l'audit — Module Recon :

- [ ] Sous-domaines via crt.sh
- [ ] Serveurs exposés (Shodan)
- [ ] Emails compromis (HIBP)
- [ ] Emails de récupération dans stealer logs

Post-audit :

- [ ] Clé Service Account révoquée
- [ ] Délégation domain-wide retirée
- [ ] Rapport livré et discuté
- [ ] Roadmap validée avec le client

Annexe D — Audit Google Chat et Meet

Google Chat est souvent négligé dans les audits alors qu'il constitue un canal de communication sensible et un vecteur de partage de données non maîtrisé.

D.1 Risques spécifiques à Google Chat

RISQUE	DESCRIPTION	GRAVITÉ
Espaces publics	Un espace Google Chat ouvert à tous les utilisateurs peut contenir des informations sensibles	🟠 ÉLEVÉ
Bots non autorisés	Des bots OAuth peuvent accéder aux messages et répondre automatiquement	🔴 CRITIQUE
Partage de fichiers hors DLP	Les pièces jointes partagées dans Chat contournent les règles DLP Drive	🟠 ÉLEVÉ
Historique non maîtrisé	Par défaut, l'historique des messages est conservé indéfiniment	🟡 MOYEN
Espaces avec membres externes	Des espaces peuvent être partagés avec des utilisateurs hors domaine	🟠 ÉLEVÉ

D.2 Vérifications Chat via admin.google.com

```
Admin Console → Apps → Google Workspace → Chat
→ Paramètres Chat :
   - Autoriser les utilisateurs externes dans les espaces : À restreindre
   - Historique par défaut : Configurer la rétention
   - Bots : Vérifier la liste des bots autorisés
   - Imports de fichiers : Vérifier les types autorisés
```

D.3 Audit Google Meet

CONTRÔLE	PARAMÈTRE	RISQUE SI NON CONFIGURÉ
Réunions externes	Qui peut rejoindre sans authentification	Participants non autorisés
Enregistrement	Où sont stockés les enregistrements	Fuites de réunions confidentielles
Transcriptions	Qui peut activer les transcriptions	Données sensibles transcrites et stockées
Partage d'écran	Restrictions	Données affichées par erreur enregistrées
Lobby (salle d'attente)	Activé ou non	Accès direct sans validation hôte

Recommandation : Activer systématiquement le lobby pour les réunions avec des participants externes. Restreindre l'enregistrement aux utilisateurs authentifiés du domaine.

Annexe E — Intégration SIEM et Monitoring Continu

Un audit ponctuel fournit une photographie de la posture de sécurité à un instant T. Pour une protection continue, il est essentiel d'exporter les logs Google Workspace vers un SIEM.

E.1 Options d'export des logs

SOLUTION	MÉCANISME	AVANTAGES	INCONVÉNIENTS
Google Chronicle	Export natif Workspace	Intégration native, UEBA inclus	Coût Chronicle
BigQuery	Export via Cloud Logging	Flexible, SQL	Pas d'alertes natives
Splunk	App Google Workspace for Splunk	Écosystème Splunk	Complexité configuration
Microsoft Sentinel	Connecteur Google Workspace	Unifié si hybrid	Latence
Elastic SIEM	Logstash Google Workspace	Open source	Maintenance
Script Python custom	Reports API en polling	Gratuit, flexible	Pas temps réel

E.2 Logs prioritaires à collecter

```
# Applications prioritaires dans l'API Reports
LOG_APPLICATIONS = [
    "login",          # Connexions : succès, échecs, suspicions
    "admin",          # Actions admin : droits, configurations
    "drive",          # Accès fichiers : téléchargements, partages
    "gmail",          # Actions Gmail : forwarding, délégation
    "token",          # Autorisations OAuth
    "groups_enterprise",  # Modifications de groupes
    "meet",           # Réunions enregistrées, participants externes
    "chat",           # Messages (si activé)
    "calendar",       # Partages calendriers, événements sensibles
]
```

E.3 Règles de détection recommandées

```
# Exemples de règles SIEM pour Google Workspace

Rule: Super Admin Login Outside Business Hours
  Source: login logs
  Condition: actor.isAdmin = true AND hour(timestamp) NOT IN (8..20)
  Severity: HIGH

Rule: New OAuth Grant with Sensitive Scope
  Source: token logs
  Condition: scope CONTAINS "mail.google.com" OR scope CONTAINS "auth/drive"
  Severity: HIGH

Rule: Gmail Forwarding Activation
  Source: admin logs
  Condition: eventName = "CHANGE_MAIL_FORWARDING_SETTING"
  Severity: CRITICAL

Rule: Multiple Failed Logins
  Source: login logs
  Condition: eventName = "login_failure" AND COUNT > 5 WITHIN 1 hour
  Severity: MEDIUM

Rule: File Shared Publicly
  Source: drive logs
  Condition: eventName = "change_user_access" AND visibility = "people_with_link"
  Severity: HIGH

Rule: Super Admin Role Grant
  Source: admin logs
  Condition: eventName = "ASSIGN_ROLE" AND role = "SUPER_ADMIN"
  Severity: CRITICAL

Rule: Login from New Country
  Source: login logs
  Condition: country NOT IN user_historical_countries AND eventName = "login_success"
  Severity: MEDIUM
```

E.4 Métriques de maturité SOC pour Workspace

MATURITÉ	CAPACITÉS	INDICATEURS
Niveau 0	Aucune surveillance	Pas de SIEM, alertes manuelles
Niveau 1	Logs collectés	Rétention > 1 an, accès ponctuel
Niveau 2	Alertes basiques	Alert Center + quelques règles SIEM
Niveau 3	Détection active	Règles UEBA, corrélation inter-sources
Niveau 4	SOC mature	SOAR, réponse automatisée, Threat Hunting

Annexe F — Politique de Mots de Passe et Authentification Avancée

F.1 Vérification de la politique de mots de passe

La politique de mots de passe Workspace se configure dans `admin.google.com → Sécurité → Authentification → Politique de mots de passe`. Elle n'est pas accessible via API, mais les paramètres à vérifier manuellement sont :

PARAMÈTRE	VALEUR RECOMMANDÉE	CIS
Longueur minimale	12 caractères	CIS 1.1.6
Complexité	Activée (majuscule, chiffre, symbole)	Bonne pratique
Réutilisation	Bloquer les 10 derniers	Bonne pratique
Expiration	Désactivée si MFA fort en place	CIS 1.1.7
Force du mot de passe	Indicateur activé	Informationnel

Pourquoi désactiver l'expiration avec MFA fort ? Les recherches du NIST (SP 800-63B) montrent que l'expiration forcée des mots de passe conduit les utilisateurs à choisir des mots de passe prévisibles (ex: `MotDePasse2025!` → `MotDePasse2026!`). Avec un MFA résistant au phishing (FIDO2), la résilience vient du second facteur, pas de la rotation du premier.

F.2 Context-Aware Access

Context-Aware Access permet de conditionner l'accès aux services Google à des critères contextuels (localisation géographique, appareil géré, plage horaire).

Cas d'usage recommandés :

```
Politique 1 : Accès admin depuis appareils gérés uniquement
  Condition : isAdminUser = true
  Exigence : deviceManagementLevel = MANAGED
  Action si non conforme : Bloquer

Politique 2 : Restriction géographique
  Condition : ALL users
  Exigence : country IN [FR, BE, LU, CH, ...pays_autorisés]
  Action si non conforme : Bloquer ou MFA supplémentaire

Politique 3 : Accès données sensibles depuis IP de confiance
  Condition : Access to Vault or Admin Console
  Exigence : ipRange IN [IP_bureau, IP_VPN]
  Action si non conforme : Bloquer

Politique 4 : Horaires de bureau pour accès admin
  Condition : isAdminUser = true
  Exigence : time IN [08:00-20:00 CET] AND dayOfWeek IN [MON-FRI]
  Action : MFA supplémentaire hors plage
```

F.3 Passkeys et FIDO2

Les passkeys remplacent le mot de passe par une paire de clés cryptographiques stockées sur l'appareil. Elles sont résistantes au phishing car liées au domaine lors de la création.

Déploiement progressif recommandé :

```
Phase 1 : Déployer les clés physiques FIDO2 (YubiKey 5 NFC) pour les Super Admins
          → Blocage complet de toute connexion sans clé physique

Phase 2 : Activer les passkeys pour les utilisateurs à haut risque
          → Dirigeants, équipes financières, équipes IT

Phase 3 : Déployer les passkeys à l'ensemble des utilisateurs
          → Migration progressive, support utilisateurs
          → Conserver SMS OTP comme fallback temporaire uniquement

Phase 4 : Désactiver SMS OTP et backup codes comme facteurs principaux
          → Uniquement FIDO2, TOTP app, ou passkey
```

Annexe G — Gestion des Incidents liés à Workspace

G.1 Procédure d'urgence — Compte compromis

```
DÉTECTION
└── Alerte Google, signalement utilisateur, ou détection SOC
        ↓
CONFINEMENT IMMÉDIAT (< 15 minutes)
└── 1. Révoquer toutes les sessions actives
       Admin Console → Utilisateurs → [compte] → Réinitialiser l'authentification
    2. Désactiver le compte temporairement
    3. Révoquer tous les tokens OAuth du compte
    4. Si Super Admin : retirer les privilèges admin temporairement
        ↓
INVESTIGATION (< 2 heures)
└── 1. Analyser les logs Login sur 30 jours
    2. Identifier les IPs, heures, pays de connexion
    3. Vérifier si un forwarding a été activé
    4. Vérifier les nouveaux partages Drive créés
    5. Vérifier les OAuth grants créés
    6. Identifier le vecteur initial (phishing ? credential stuffing ?)
        ↓
REMEDIATION
└── 1. Réinitialiser le mot de passe via canal sécurisé hors-bande
    2. Forcer la reconfiguration de la 2FA
    3. Révoquer définitivement les tokens OAuth suspects
    4. Supprimer les forwardings créés
    5. Vérifier et corriger les partages Drive créés pendant l'incident
        ↓
NOTIFICATION (si violation RGPD avérée)
└── 1. Évaluer si des données personnelles ont été compromises
    2. Si oui : notification CNIL dans les 72h (Art. 33 RGPD)
    3. Si risque élevé pour les personnes : notification individuelle (Art. 34)
    4. Documenter : nature, volume, catégories de données affectées
        ↓
RETOUR D'EXPÉRIENCE
└── 1. Rapport post-incident
    2. Identification de la cause racine
    3. Mesures préventives pour éviter la récurrence
    4. Mise à jour de la politique de sécurité
```

G.2 Indicateurs de Compromission (IoC) spécifiques Workspace

IOC	SIGNAL	PRIORITÉ
Forwarding Gmail activé vers domaine externe	Log admin `CHANGE_MAIL_FORWARDING_SETTING`	🔴 Critique
Connexion depuis pays jamais vu	Log login + geoloc	🔴 Critique
Token OAuth accordé à app inconnue	Log token avec clientId inconnu	🔴 Critique
Téléchargement massif Drive	Logs Drive > N fichiers en < 1h	🔴 Critique
Nouveau Super Admin ajouté sans ticket	Log admin `ASSIGN_ROLE`	🔴 Critique
Réinitialisation 2FA sur compte admin	Alert Center + log admin	🟠 Élevé
Partage "anyone" sur fichier sensible	Log Drive permission change	🟠 Élevé
Service account ajouté au Drive	Permissions Drive type=serviceAccount	🟠 Élevé
Connexion depuis IP Tor ou VPN connu	Log login + threat intel	🟠 Élevé
Volume anormal d'emails envoyés	Reports usage API	🟡 Moyen

G.3 Commandes d'urgence — Réponse rapide

```
# Révoquer toutes les sessions d'un utilisateur (lecture + écriture requise)
def revoke_all_sessions(service, user_email):
    """ATTENTION : Action d'écriture — à utiliser en cas d'incident confirmé."""
    service.users().signOut(userKey=user_email).execute()

# Lister les tokens actifs d'un utilisateur (lecture seule)
def list_active_tokens(security_service, user_email):
    tokens = security_service.tokens().list(userKey=user_email).execute()
    return tokens.get("items", [])

# Vérifier le forwarding actif (lecture seule)
def check_active_forwarding(gmail_service, user_email):
    result = gmail_service.users().settings().getAutoForwarding(
        userId=user_email
    ).execute()
    return result if result.get("enabled") else None
```

Annexe H — Scoring et Benchmarking

H.1 Grille de scoring détaillée

La pondération ci-dessous reflète l'impact réel des contrôles sur la posture de sécurité :

CONTRÔLE	POIDS	RAISON
2FA enforced sur tous les comptes	15 pts	Prévention account takeover massive
Aucun transfert Gmail actif	15 pts	Exfiltration en temps réel
Emails récup Super Admins internes	12 pts	Vecteur takeover compte le plus élevé
DMARC p=reject	10 pts	Protection usurpation identité email
Aucun fichier public ("anyone")	10 pts	Données exposées sans contrôle
Gouvernance OAuth (whitelist)	8 pts	Consent phishing
SPF hard fail (-all)	7 pts	Complémentaire DMARC
DKIM 2048 bits	5 pts	Intégrité emails
Groupes sans membres externes	5 pts	Fuite d'information via groupes
Logs 1 an + SIEM	5 pts	Forensics et détection
MDM actif et enforced	4 pts	Appareils non gérés
Context-Aware Access	4 pts	Contrôle contextuel

H.2 Comparaison sectorielle

SECTEUR	SCORE MOYEN OBSERVÉ	MATURITÉ TYPIQUE
Finance / Banque	72/100	Élevée — régulation stricte
Santé	58/100	Moyenne — effort RGPD récent
ESN / IT	61/100	Variable — souvent focalisé technique mais pas gouvernance
Industrie	49/100	Faible — Workspace outil récent
Associations / ONG	35/100	Très faible — ressources limitées
Startups (< 50 pers.)	42/100	Faible — croissance rapide, sécurité en retard

H.3 Évolution et suivi dans le temps

Un audit unique n'a que peu de valeur sans suivi. Les indicateurs à suivre trimestriellement :

```
Tableau de bord trimestriel Google Workspace

┌──────────────────────────────────────────────┐
│ Score global de sécurité         : [  /100] │
│ Conformité CIS Benchmark         : [  %   ] │
│ Emails compromis HIBP (domaine)  : [  /55 ] │
│ Tokens OAuth non revus (> 90j)   : [  #   ] │
│ Partages externes actifs         : [  #   ] │
│ Utilisateurs sans 2FA            : [  #   ] │
│ Super Admins                     : [  #   ] │
│ Alertes Alert Center (30j)       : [  #   ] │
│ Fichiers publics ("anyone")      : [  #   ] │
└──────────────────────────────────────────────┘
```

Annexe D — Ressources et Références

RESSOURCE	URL	USAGE
CIS Benchmark Google Workspace	cisecurity.org	Référentiel de contrôles
MITRE ATT&CK for Enterprise	attack.mitre.org	Techniques d'attaque
Google Admin SDK Docs	developers.google.com	Documentation API
HIBP API	haveibeenpwned.com/API	Vérification breaches
Have I Been Pwned Domain	haveibeenpwned.com/DomainSearch	Vérification par domaine
crt.sh	crt.sh	Certificate Transparency
Shodan InternetDB	internetdb.shodan.io	Exposition Internet (gratuit)
IntelligenceX	intelx.io	Darknet/leaks
ANSSI — Guide sécurité GSuite	ssi.gouv.fr	Recommandations françaises
Google Workspace Security Checklist	workspace.google.com/security	Checklist officielle Google

Annexe E — Modèle de feuille de route

ID	ACTION	PRIORITÉ	EFFORT	IMPACT	RESPONSABLE	DÉLAI
R-01	Désactiver transfert Gmail actif	🔴 P1	Faible	Critique	Admin IT	J+0
R-02	Retirer emails récup externes (stealer logs)	🔴 P1	Faible	Critique	Admin IT	J+0
R-03	Révoquer service accounts Drive externes	🔴 P1	Moyen	Critique	DSI	J+2
R-04	Supprimer fichiers publics	🔴 P1	Faible	Élevé	Admin IT	J+3
R-05	Désactiver forwarding Gmail globalement	🔴 P1	Faible	Critique	Admin IT	J+3
R-06	Réduire Super Admins à 3 max	🟠 P2	Moyen	Élevé	RSSI + DSI	J+7
R-07	SPF `-all` (hard fail)	🟠 P2	Faible	Élevé	Admin IT	J+7
R-08	Créer comptes admin dédiés	🟠 P2	Moyen	Élevé	DSI	J+14
R-09	Activer approbation OAuth	🟠 P2	Moyen	Élevé	Admin IT	J+14
R-10	Révoquer tokens AI sur comptes sensibles	🟠 P2	Faible	Élevé	Admin IT	J+7
R-11	Assigner owners aux groupes orphelins	🟡 P3	Faible	Moyen	Admin IT	J+14
R-12	Procédure offboarding documentée	🟡 P3	Moyen	Moyen	RH + DSI	J+30
R-13	Déployer YubiKeys pour Super Admins	🟡 P3	Élevé	Critique	RSSI	J+45
R-14	Context-Aware Access (restriction géo)	🟡 P3	Élevé	Élevé	Admin IT	J+45
R-15	Activer Google Vault	🔵 P4	Élevé	Moyen	DSI	J+60
R-16	Configurer export logs SIEM	🔵 P4	Élevé	Élevé	SOC	J+90
R-17	Règles DLP sur données sensibles	🔵 P4	Élevé	Élevé	RSSI	J+90
R-18	Formation sécurité utilisateurs	🔵 P4	Moyen	Élevé	RSSI + RH	J+60
R-19	Revue trimestrielle OAuth	🔵 P4	Faible	Moyen	Admin IT	Récurrent
R-20	Audit Drive annuel documenté	🔵 P4	Moyen	Moyen	DSI	Récurrent

---

Conclusion — Vers une Sécurité Google Workspace Durable

L'audit Google Workspace n'est pas une fin en soi : c'est le point de départ d'un programme de sécurité continu. Les organisations qui traitent cet exercice comme une vérification annuelle ponctuelle resteront exposées entre deux audits. Celles qui intègrent les contrôles automatisés, le monitoring continu et la culture de sécurité dans leur ADN réduiront durablement leur surface d'attaque.

Les cinq leviers de la maturité Workspace

1. L'identité comme périmètre de sécurité Dans un monde où les données résident dans le cloud, le périmètre réseau traditionnel n'existe plus. L'identité — et sa protection via 2FA résistant au phishing, Context-Aware Access, et gestion rigoureuse des comptes — est le nouveau périmètre. Investir prioritairement sur l'IAM est le levier à plus fort retour sur investissement. **2. La gouvernance des données avant la technologie** Les technologies de protection (DLP, CASB, SIEM) sont efficaces uniquement si la gouvernance des données est en place : qui a accès à quoi, pour quels usages, pour quelle durée. Un Drive mal gouverné avec 200 000 fichiers partagés ne sera pas sécurisé par un outil — il faut d'abord définir et appliquer des politiques. **3. La détection comme complément à la prévention** Aucun contrôle préventif n'est infaillible. La capacité à détecter rapidement une compromission — via des logs centralisés, des alertes bien configurées, et des procédures de réponse testées — réduit considérablement l'impact d'un incident. Le délai moyen de détection d'une compromission est encore de plusieurs semaines dans la majorité des organisations. **4. La sensibilisation des utilisateurs** La grande majorité des incidents Workspace commence par une action humaine : cliquer sur un lien de phishing, autoriser une app OAuth malveillante, partager un document par erreur. Les contrôles techniques réduisent la surface d'attaque, mais la sensibilisation régulière des utilisateurs — notamment aux simulations de phishing, aux bonnes pratiques de partage Drive et à la reconnaissance des apps OAuth suspectes — reste indispensable. **5. La continuité de l'amélioration** La sécurité n'est pas un état stable : le tenant évolue, de nouveaux utilisateurs arrivent, de nouvelles apps sont autorisées, de nouvelles vulnérabilités sont découvertes. Un audit annuel minimum, un monitoring continu des indicateurs clés, et une revue trimestrielle des accès (OAuth, partages Drive, Super Admins) constituent le rythme minimal d'un programme de sécurité Workspace sérieux.

Priorités universelles — Quel que soit le tenant

Quelle que soit la taille de l'organisation ou son niveau de maturité initial, trois actions produisent le plus grand effet de protection en un minimum de temps :

```
1. Enforcer la 2FA sur tous les comptes, sans exception
   → Réduit de 99% le risque d'account takeover par credential stuffing

2. Supprimer ou remplacer par des emails internes
   tous les emails de récupération externes sur les comptes admins
   → Élimine le bypass 2FA via récupération de compte

3. Désactiver le transfert automatique de messagerie
   et activer une alerte sur toute future activation
   → Stoppe l'exfiltration silencieuse en temps réel
```

Ces trois contrôles, appliqués en moins d'une heure par un Super Admin, ferment les trois vecteurs d'attaque les plus fréquents et les plus dommageables sur Google Workspace.

Guide d'Audit Google Workspace — Version 1.0 — Mars 2026 Rédigé pour les équipes Sécurité et GRC — Reproduction autorisée à des fins internes