AYI NEDJIMI Consultants

Abus OAuth/OIDC : Consent - Guide Pratique Cybersecurite

Catégorie : Articles Techniques Lecture : 9 min Publié le : 07/12/2025 Auteur : Ayi NEDJIMI

Guide expert sur les attaques OAuth/OIDC : Illicit Consent Grant, Device Code Phishing, Token Replay. Détections SIEM, règles de corrélation et...

Cette analyse detaillee de Abus OAuth/OIDC : Consent - Guide Pratique Cybersecurite s'appuie sur les retours d'experience d'equipes de securite confrontees quotidiennement aux menaces actuelles. Les methodologies presentees couvrent l'ensemble du cycle de vie de la securite, de la detection initiale a la remediation complete, en passant par l'investigation forensique et le durcissement des configurations. Les recommandations sont directement applicables dans les environnements de production et tiennent compte des contraintes operationnelles rencontrees par les equipes techniques sur le terrain. Les outils et techniques presentes ont ete valides dans des contextes reels d'incidents et de tests d'intrusion. La mise en oeuvre d'une strategie de defense en profondeur reste essentielle face a l'evolution constante du paysage des menaces, en combinant prevention, detection et capacite de reponse rapide aux incidents de securite.

Cette analyse technique de Abus OAuth/OIDC : Consent - Guide Pratique Cybersecurite s'appuie sur les retours d'experience d'equipes confrontees quotidiennement aux defis operationnels du domaine. Les methodologies presentees couvrent l'ensemble du cycle de vie, de la conception initiale au deploiement en production, en passant par les phases de test et de validation. Les recommandations sont directement applicables dans les environnements professionnels.

Table des matières

Auteur : Ayi NEDJIMI **Date :** 23 octobre 2025

Notre avis d'expert
La documentation technique de sécurité est le parent pauvre de la plupart des organisations. Pourtant, un playbook de réponse à incident bien rédigé peut faire la différence entre une résolution en heures et une crise qui s'étend sur des semaines.

Introduction

OAuth 2.0 et OpenID Connect (OIDC) sont devenus les protocoles de facto pour l'authentification et l'autorisation dans les environnements cloud modernes. Adoptés massivement par les entreprises utilisant Microsoft 365, Google Workspace, Salesforce et d'innombrables applications SaaS, ces protocoles offrent une expérience utilisateur fluide et une sécurité théoriquement robuste. Cependant, leur complexité inhérente et leur omniprésence en font des cibles privilégiées pour les attaquants aboutis.

Cet article examine en profondeur les principales techniques d'abus d'OAuth/OIDC exploitées par les acteurs malveillants : les attaques par consentement illégitime (Illicit Consent Grant), l'exploitation du flux Device Code, le rejeu de tokens (Token Replay), ainsi que d'autres vecteurs d'attaque émergents. Nous explorerons comment détecter ces activités malveillantes via des solutions SIEM et présenterons des stratégies complètes de durcissement des Identity Providers (IdP).

Destiné aux architectes de sécurité, aux analystes SOC, aux administrateurs d'identité et aux RSSI, ce document fournit une compréhension technique approfondie des menaces, des indicateurs de compromission, des règles de détection pratiques et des recommandations de configuration pour réduire significativement la surface d'attaque.

Element	Description	Priorite
Prevention	Mesures proactives de reduction de la surface d'attaque	Haute
Detection	Surveillance et alerting en temps reel	Haute
Reponse	Procedures d'incident response et remediation	Critique
Recovery	Plan de reprise et continuite d'activite	Moyenne

Avez-vous automatisé les tâches de sécurité répétitives qui consomment le temps de vos équipes ?

Comprendre OAuth 2.0 et OpenID Connect

Architecture et acteurs

OAuth 2.0 est un framework d'autorisation qui permet à des applications tierces d'obtenir un accès limité aux ressources d'un utilisateur sans exposer ses identifiants. OpenID Connect (OIDC) est une couche d'identité construite au-dessus d'OAuth 2.0, ajoutant des capacités d'authentification.

Acteurs principaux :

- **Resource Owner** : L'utilisateur final qui possède les données et peut accorder l'autorisation d'y accéder.
- **Client** : L'application qui souhaite accéder aux ressources protégées au nom de l'utilisateur.
- **Authorization Server** : Le serveur qui authentifie l'utilisateur et émet les tokens après obtention du consentement (Azure AD, Okta, Auth0, etc.).
- **Resource Server** : Le serveur hébergeant les ressources protégées (API Microsoft Graph, Google APIs, etc.).

Flux OAuth 2.0 courants :

- **Authorization Code Flow** : Le flux le plus sécurisé, recommandé pour les applications web avec backend.
- **Authorization Code Flow with PKCE** : Extension avec Proof Key for Code Exchange, essentielle pour les applications publiques (mobiles, SPA).
- **Implicit Flow** : Flux historique pour les applications JavaScript, désormais déprécié en raison de vulnérabilités.
- **Client Credentials Flow** : Pour l'authentification service-to-service sans utilisateur interactif.
- **Device Code Flow** : Pour les appareils avec interface limitée (smart TVs, IoT), particulièrement vulnérable aux abus.
- **Resource Owner Password Credentials** : Flux legacy où l'application gère directement les identifiants, fortement déconseillé.

Tokens et permissions

Access Token : Jeton donnant accès aux ressources protégées. Format généralement JWT (JSON Web Token) ou opaque. Durée de vie courte (minutes à heures).

Refresh Token : Jeton longue durée permettant d'obtenir de nouveaux access tokens sans réauthentification. Hautement sensible.

ID Token (OIDC) : Jeton contenant les informations d'identité de l'utilisateur (claims). Toujours au format JWT signé.

Scopes et Permissions : Les scopes définissent les permissions accordées. Exemples dans l'écosystème Microsoft 365 :

- `User.Read` : Lire le profil de l'utilisateur
- `Mail.Read` : Lire les emails

- `Files.ReadWrite.All` : Accès complet aux fichiers
- `Directory.AccessAsUser.All` : Accéder à l'annuaire au nom de l'utilisateur

Les scopes se divisent en deux catégories critiques :

- **Delegated Permissions** : L'application agit au nom d'un utilisateur connecté. Permissions limitées par celles de l'utilisateur.
- **Application Permissions** : L'application agit avec sa propre identité, sans utilisateur. Permissions potentiellement illimitées (nécessite consentement administrateur).

Mécanisme de consentement

Le consentement OAuth est le moment où l'utilisateur autorise explicitement une application à accéder à ses ressources avec des permissions spécifiques. Cette interface de consentement affiche le nom de l'application, l'éditeur/développeur, les permissions demandées et la portée de l'accès.

Consentement utilisateur : Pour les permissions déléguées non sensibles, n'importe quel utilisateur peut consentir.

Consentement administrateur : Pour les permissions élevées ou les permissions d'application, seul un administrateur peut accorder le consentement.

Cette dichotomie crée une surface d'attaque : les attaquants exploitent la confiance des utilisateurs pour obtenir des consentements illégitimes, ou compromettent des comptes administrateurs pour obtenir des permissions étendues.

Cas concret
L'exploitation massive des vulnérabilités ProxyShell sur Microsoft Exchange en 2021 a démontré l'importance du patch management rapide. Les organisations ayant tardé à appliquer les correctifs ont vu leurs serveurs compromis et utilisés comme points de pivot pour des attaques ransomware.

Attaques par consentement illégitime

Description de l'attaque

L'attaque par consentement illégitime (Illicit Consent Grant) exploite le mécanisme de consentement OAuth pour obtenir un accès persistant aux ressources d'un utilisateur sans compromettre directement ses identifiants. Cette technique, popularisée par des groupes APT, est particulièrement insidieuse car elle contourne les protections traditionnelles (MFA, mot de passe complexe, etc.).

Déroulement typique :

1. **Phase 1 - Préparation** : L'attaquant enregistre une application malveillante auprès de l'IdP. Cette application demande des permissions sensibles comme Mail.Read, Files.ReadWrite.All, ou Contacts.Read.
2. **Phase 2 - Phishing** : La victime reçoit un email de phishing avec un lien vers l'URL d'autorisation OAuth de l'application malveillante.

3. **Phase 3 - Consentement** : La victime clique sur le lien, est redirigée vers la véritable page de consentement de Microsoft/Google, s'authentifie (avec MFA si configuré), et voit une interface demandant le consentement pour l'application.
4. **Phase 4 - Exploitation** : Une fois le consentement accordé, l'attaquant obtient un refresh token longue durée lui permettant d'accéder aux ressources de la victime sans nouvelle authentification.
5. **Phase 5 - Actions malveillantes** : L'attaquant peut lire les emails, exfiltrer des fichiers, accéder aux contacts, ou utiliser l'accès comme point d'entrée pour une compromission plus large.

Indicateurs de compromission

Caractéristiques suspectes à surveiller :

- Application créée récemment (moins de 30 jours)
- Éditeur non vérifié (absence de badge vérifié)
- Demande de permissions sensibles : Mail.Read, Files.ReadWrite.All, Contacts.Read
- Consentement depuis une adresse IP inhabituelle ou un appareil non géré
- Horaire inhabituel (3h du matin pour un utilisateur normalement actif en journée)
- Multiple consentements pour la même application en peu de temps (campagne)

Détection via Microsoft Sentinel

Règle KQL (Kusto Query Language) :

```
AuditLogs
| where OperationName == "Consent to application"
| extend AppDisplayName = tostring(TargetResources[0].displayName)
| extend AppId = tostring(TargetResources[0].id)
| extend UserPrincipalName = tostring(InitiatedBy.user.userPrincipalName)
| extend IPAddress = tostring(InitiatedBy.user.ipAddress)
| extend Permissions = tostring(TargetResources[0].modifiedProperties)
| where Permissions contains "Mail.Read"
    or Permissions contains "Files.ReadWrite"
    or Permissions contains "Contacts.Read"
| where AppDisplayName !startswith "Microsoft"
| project TimeGenerated, UserPrincipalName, AppDisplayName, AppId,
          IPAddress, Permissions
| order by TimeGenerated desc
```

Attaque par Device Code Flow

Fonctionnement et exploitation

Le Device Code Flow est conçu pour les appareils avec interface limitée (smart TV, imprimantes, CLI tools). Un attaquant peut créer une application malveillante utilisant Device Code Flow et envoyer le code à une victime via phishing :

- "Pour accéder au document partagé, allez sur microsoft.com/devicelogin et entrez le code : WXYZ-1234"
- La victime, croyant à une demande légitime, s'authentifie et consent
- L'attaquant obtient immédiatement les tokens d'accès

Avantages pour l'attaquant :

- URL légitime Microsoft/Google (pas de domaine de phishing à héberger)
- Contourne les solutions anti-phishing basées sur l'URL
- Utilisable même si les applications tierces sont restreintes

Détection et mitigation

Règle de détection Azure Sentinel :

```
SigninLogs
| where AuthenticationProtocol == "deviceCode"
| extend AppDisplayName = tostring(AppDisplayName)
| extend UserPrincipalName = tostring(UserPrincipalName)
| extend IPAddress = tostring(IPAddress)
| where AppDisplayName !startswith "Microsoft"
    and AppDisplayName !startswith "Azure"
| summarize
    FirstSeen = min(TimeGenerated),
    LastSeen = max(TimeGenerated),
    LoginCount = count(),
    UniqueUsers = dcount(UserPrincipalName)
    by AppDisplayName
| where UniqueUsers > 5
| order by UniqueUsers desc
```

Mitigation :

- Désactiver le Device Code Flow si non nécessaire via Conditional Access
- Créer une politique d'accès conditionnel bloquant Device Code Flow pour les applications non approuvées
- Former les utilisateurs à ne JAMAIS entrer de codes reçus par email
- Implémenter une validation manuelle pour les nouvelles applications utilisant Device Code Flow

---

Token Replay et Token Theft

Vecteurs de vol de tokens

Les tokens OAuth (Access Token, Refresh Token) peuvent être volés via plusieurs vecteurs :

- **Malware sur endpoint** : Extraction depuis le cache navigateur, keylogger, memory dump
- **Man-in-the-Middle (MitM)** : Interception sur réseau non sécurisé (rare avec HTTPS strict)
- **XSS (Cross-Site Scripting)** : Vol de tokens stockés dans localStorage ou sessionStorage
- **Application compromise** : Exfiltration depuis les logs, bases de données, ou code malveillant injecté
- **Phishing poussé** : Proxy inverse capturant les tokens en temps réel (evilginx2, Modlishka)

Détection d'anomalies

Anomalies à détecter :

- Utilisation du même token depuis deux géolocalisations incompatibles temporellement (impossible travel)
- Changement de User-Agent pour le même token
- Accès depuis IP/ASN inhabituelle
- Utilisation du token en dehors des heures normales de l'utilisateur
- Volume d'appels API anormal (exfiltration massive)

```
SigninLogs
| where ResultType == "0"
| extend Location1 = tostring(LocationDetails.city)
| extend Country1 = tostring(LocationDetails.countryOrRegion)
| join kind=inner (
    SigninLogs
    | where ResultType == "0"
    | extend Location2 = tostring(LocationDetails.city)
    | extend Country2 = tostring(LocationDetails.countryOrRegion)
) on UserPrincipalName
| where TimeGenerated1 < TimeGenerated
| where Country1 != Country2
| extend TimeDiff = datetime_diff('hour', TimeGenerated, TimeGenerated1)
| where TimeDiff < 1
| project UserPrincipalName, Location1, Location2, Country1, Country2,
          TimeGenerated1, TimeGenerated, TimeDiff
```

Durcissement et recommandations

Configuration Azure AD / Entra ID

1. Restreindre le consentement utilisateur :

- Azure AD > Enterprise Applications > Consent and permissions
- Définir "Users can consent to apps from verified publishers for selected permissions"

- Limiter les permissions pouvant être consenties par les utilisateurs (ex: autoriser User.Read mais bloquer Mail.Read)
- Activer "Admin consent workflow" pour que les utilisateurs puissent demander l'approbation d'un admin

2. Implémenter Conditional Access pour applications OAuth :

- Créer une politique exigeant MFA pour toute nouvelle demande de consentement
- Bloquer les applications non approuvées via "Cloud apps > Select apps"
- Restreindre Device Code Flow aux appareils gérés uniquement

3. Activer Continuous Access Evaluation (CAE) :

- Permet la révocation instantanée des tokens en cas d'événement critique
- Événements déclencheurs : changement de mot de passe, désactivation de compte, changement de localisation
- Réduit la fenêtre d'exploitation d'un token volé

4. Utiliser Token Protection (Certificate-bound tokens) :

- Lie les tokens à un certificat d'appareil spécifique
- Empêche le rejeu de tokens volés depuis un autre appareil
- Disponible sur Windows avec Azure AD Join/Hybrid Join

Stratégies de monitoring

Métriques clés à surveiller :

- Nombre de consentements par jour (spike = campagne possible)
- Applications avec consentements de plusieurs utilisateurs en peu de temps
- Applications non vérifiées recevant des consentements
- Utilisation de Device Code Flow
- Refresh Token usage anormal (refresh excessifs)
- Impossible travel (connexions géographiquement impossibles)

Formation et sensibilisation

Points clés à communiquer aux utilisateurs :

- Toujours vérifier le nom de l'application et l'éditeur avant de consentir
- Être méfiant des applications demandant des permissions excessives
- Ne JAMAIS entrer de codes Device Code reçus par email ou chat
- Vérifier régulièrement les applications autorisées : myapps.microsoft.com
- Signaler toute demande de consentement suspecte à l'équipe sécurité

Questions frequentes

Comment ce sujet impacte-t-il la securite des organisations ?

Ce sujet a un impact significatif sur la securite des organisations car il touche aux fondamentaux de la protection des systemes d'information. Les entreprises doivent evaluer leur exposition, mettre en place des mesures preventives adaptees et former leurs equipes pour faire face aux risques associes a cette problematique.

Quelles sont les bonnes pratiques recommandees par les experts ?

Les experts recommandent une approche basee sur les risques, incluant l'evaluation reguliere de la posture de securite, la mise en place de controles techniques et organisationnels, la formation continue des equipes et l'adoption des referentiels de securite reconnus comme ceux du NIST, de l'ANSSI et de l'OWASP.

Pourquoi est-il important de se former sur ce sujet en 2026 ?

En 2026, la maitrise de ce sujet est devenue incontournable face a l'evolution constante des menaces et des exigences reglementaires. Les professionnels de la cybersecurite doivent maintenir leurs competences a jour pour proteger efficacement les actifs numeriques de leur organisation et repondre aux obligations de conformite.

Pour approfondir ce sujet, consultez notre outil open-source security-automation-framework qui facilite l'automatisation des workflows de sécurité.

Conclusion

Les attaques ciblant OAuth 2.0 et OpenID Connect représentent une menace croissante et avancée dans les environnements cloud modernes. Contrairement aux attaques traditionnelles visant directement les identifiants, les abus OAuth exploitent les mécanismes de confiance inhérents aux flux d'autorisation légitimes, rendant leur détection particulièrement délicate.

Une défense efficace repose sur une approche multi-couches combinant :

- **Prévention** : Politiques de consentement restrictives, Conditional Access, Token Protection
- **Détection** : Monitoring SIEM en temps réel des événements OAuth, corrélation avec les comportements utilisateurs
- **Réponse** : Playbooks automatisés de révocation et investigation
- **Formation** : Sensibilisation continue des utilisateurs aux risques OAuth

Les organisations utilisant Microsoft 365, Google Workspace ou d'autres plateformes cloud doivent impérativement auditer leurs configurations OAuth, implémenter les bonnes pratiques présentées dans cet article, et établir des capacités de détection robustes. La complexité croissante des menaces nécessite une vigilance constante et une adaptation continue des stratégies de défense.

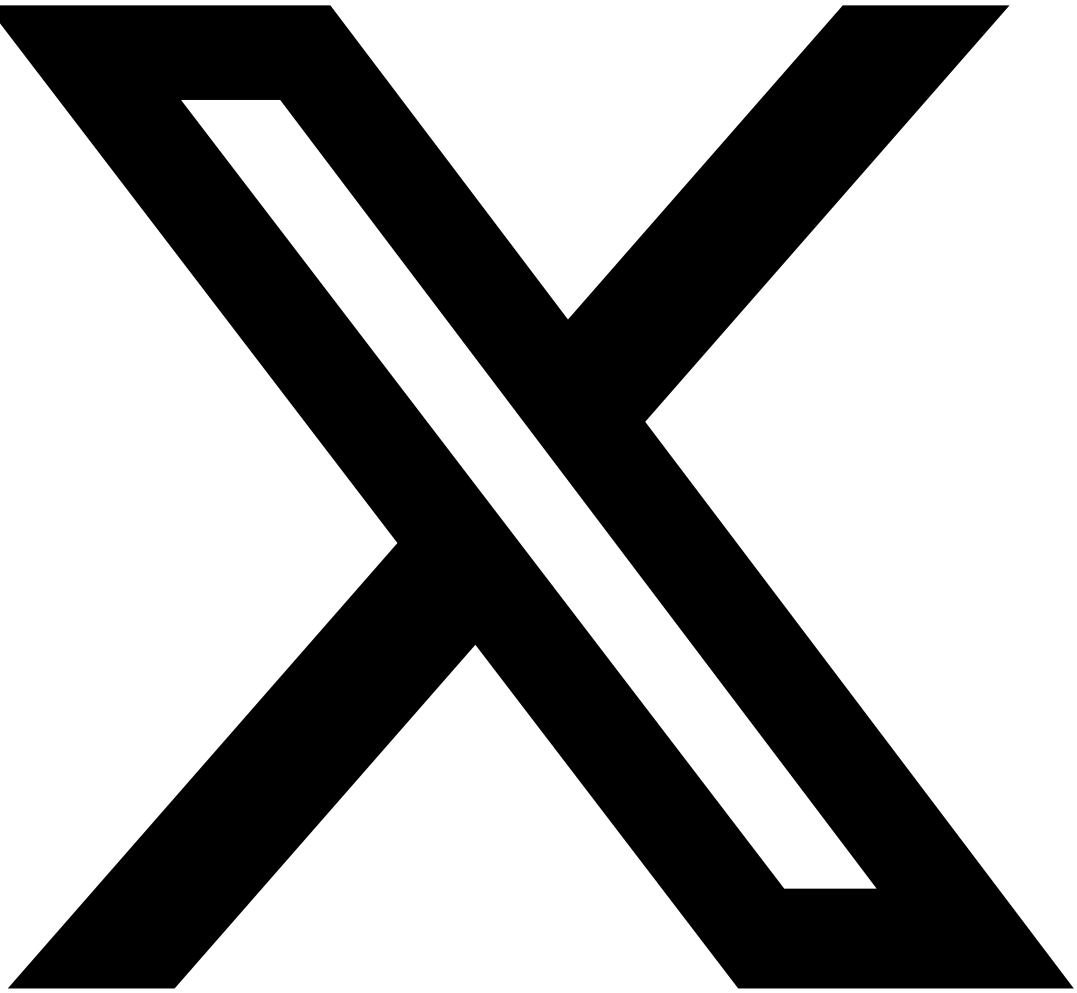
Sources et références : MITRE ATT&CK · CERT-FR

Ressources et références

- Microsoft 365 / Azure AD : Détection des attaques et compromission d'identités
- Threat Hunting dans Microsoft 365 avec Defender et Sentinel
- Audit avancé Microsoft 365 : Corrélation de journaux et logs
- Automatiser l'audit de sécurité Microsoft 365 avec PowerShell et Graph API
- Chaîne d'exploitation Kerberos en Active Directory
- Meilleures pratiques de sécurité Microsoft 365 en 2025

💬 Partagez cet Article

Cet article vous a été utile ? Partagez-le avec votre réseau professionnel !

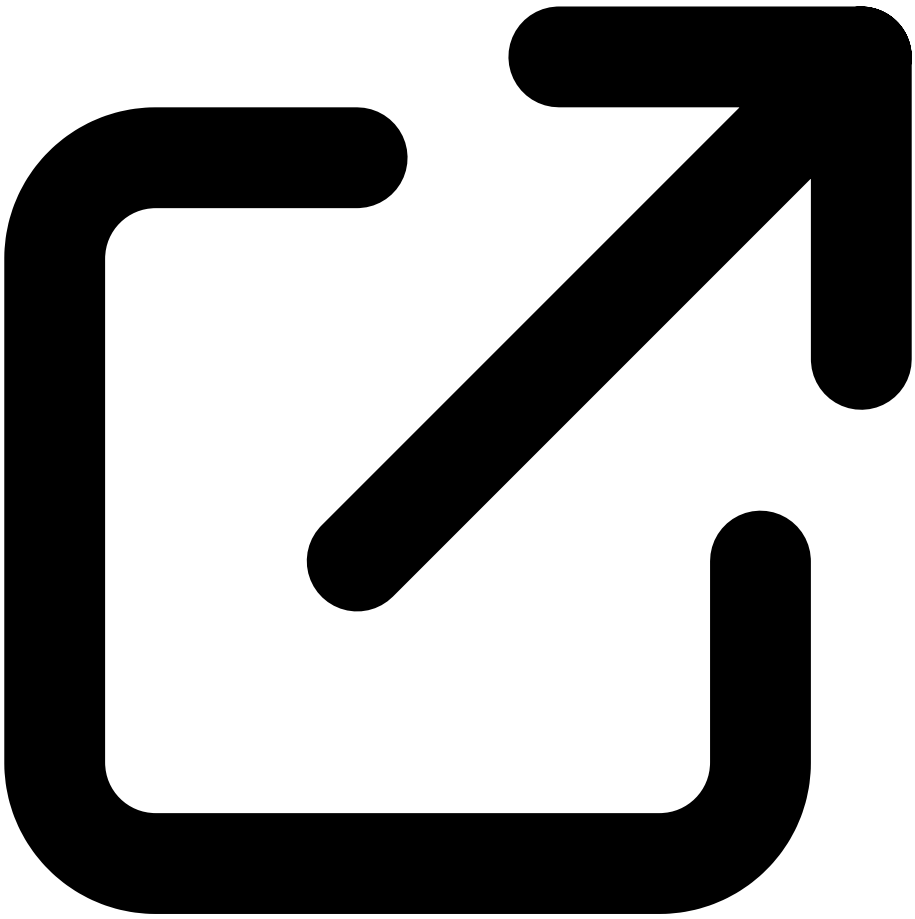


Partager sur X

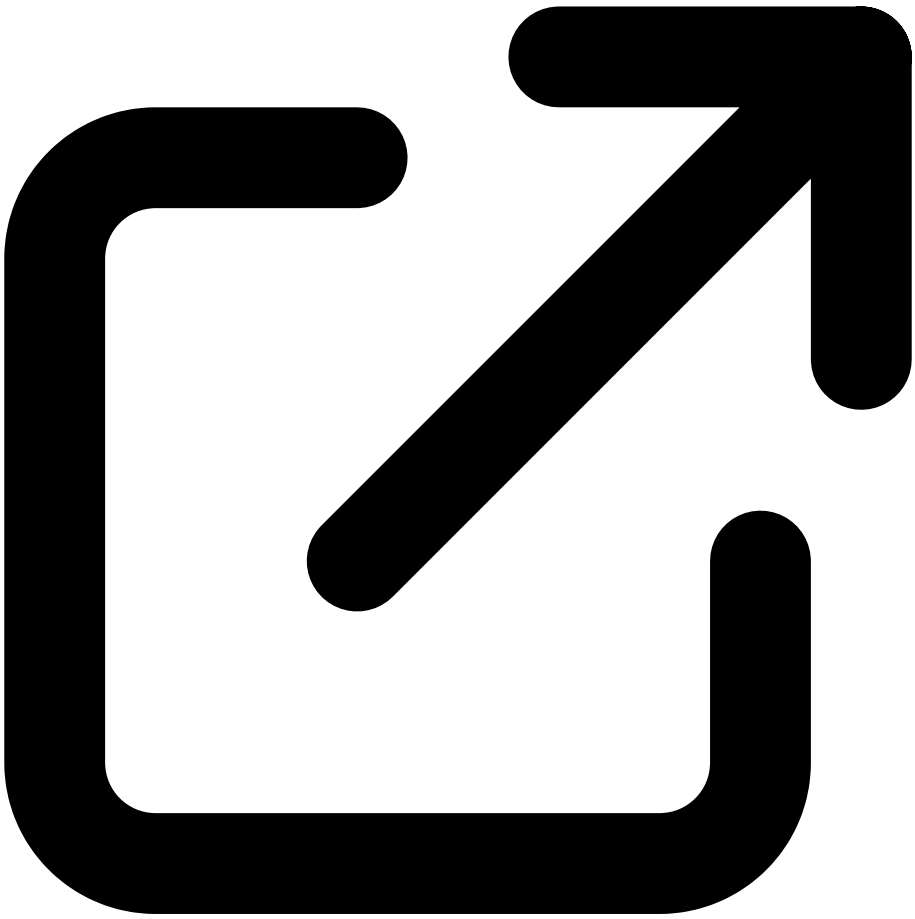
Partager sur LinkedIn

📚 Ressources & Références Officielles

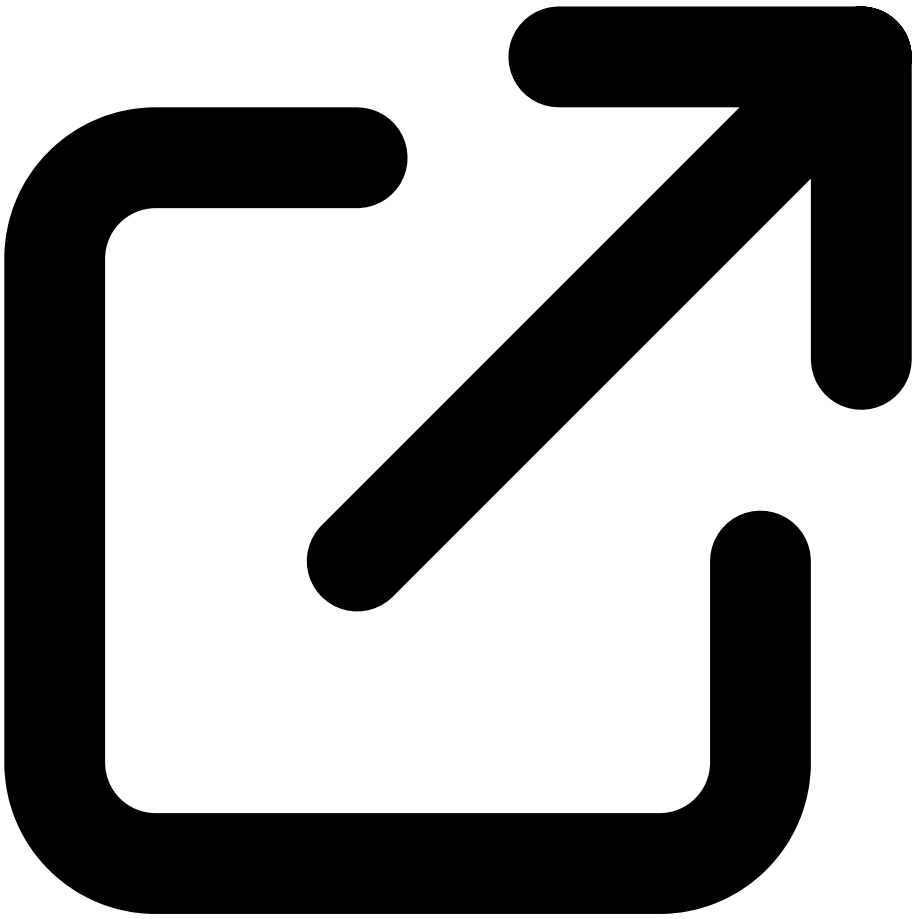
Documentations officielles, outils reconnus et ressources de la communauté



OAuth 2.0 Security Best Practices
ietf.org



OWASP - OAuth 2.0 Security Cheat Sheet
owasp.org



OAuth2 Proxy (GitHub)
github.com

Ressources open source associées :

- oauth-api-security-fr — Dataset sécurité OAuth/API (HuggingFace)
- owasp-top10-fr — Dataset OWASP Top 10 (HuggingFace)

Ayi NEDJIMI Consultants — Expert cybersécurité offensive & intelligence artificielle

ayinedjimi-consultants.fr · ayi@ayinedjimi-consultants.fr