**Engineer**

# SSRF

attaque où le serveur charge l'URL choisie par l'attaquant

## Ce que c’est

La SSRF, abréviation de server-side request forgery, est une attaque qui consiste à fournir une URL à un serveur pour qu’il en récupère le contenu au nom de l’attaquant, depuis l’emplacement réseau où il se trouve. La Common Weakness Enumeration la répertorie sous CWE-918 et décrit la faille par ce que le code omet de faire : il reçoit une URL d’un composant en amont, récupère ce qu’elle désigne et ne vérifie jamais que la requête atteint bien la destination prévue. L’attaquant ne change rien au code du serveur. Il fournit une adresse, et la position du serveur dans le réseau fait le reste.

Cette position est tout l’enjeu. Un serveur dans un centre de données ou un compte cloud peut généralement atteindre des ressources inaccessibles à un visiteur d’Internet : un panneau d’administration lié à une adresse interne, un port de base de données, un tableau de bord de supervision, un service de métadonnées. La fiche CWE détaille les conséquences. Le serveur devient un relais pour l’attaquant ; il peut être amené à scanner les ports des machines du réseau interne, et des schémas d’URL tels que `file://` , `gopher://` et `tftp://` permettent de lire des documents sur la machine elle-même ou de donner à l’attaquant davantage de contrôle sur le contenu de la requête. CWE rattache la SSRF à une faiblesse plus générale, CWE-441, le relais involontaire ou « confused deputy » : un composant privilégié qui travaille pour quelqu’un qui n’a aucun de ces privilèges.

La fonctionnalité qui ouvre la porte est généralement banale. L’aide-mémoire de prévention d’OWASP cite deux cas typiques : une image d’avatar que l’application télécharge depuis une URL saisie par l’utilisateur, et des URL de webhook ou de rappel que les utilisateurs enregistrent pour que l’application puisse les appeler plus tard. Les aperçus de liens et les boutons d’import depuis une URL appartiennent à la même famille, comme toute fonctionnalité qui fait suivre au serveur une adresse écrite par quelqu’un d’autre. Le titre de la fiche CWE écrit

le terme Server-Side Request Forgery avec des majuscules, et la fiche donne aussi un ancien nom pour la même attaque : XSPA, abréviation de Cross Site Port Attack.

Utilisez le concept pour prendre une décision

Utilisez « Qu'est-ce que la SSRF ? » pour éclairer une décision réelle. Nous pouvons vous aider à relier ce concept au système que vous construisez ou achetez. Des comptes, mots de passe et droits attribués selon les besoins exacts de chaque poste, puis retirés le jour du départ.

**Découvrir nos prestations** Contactez-nous →

Vérifiez les hypothèses derrière le terme

Appuyez-vous sur l'explication et les réalisations liées pour formuler les questions de votre projet. Nous pouvons examiner les hypothèses avec vous avant le choix d'un outil. 1 000 règles RBAC pour sécuriser Linux — 1,000 règles d'accès écrites pour combler une faille de sécurité

**1 000 règles RBAC pour sécuriser Linux** Contactez-nous →

## Pourquoi c’est important

OWASP a ajouté la SSRF à son Top 10 en 2021, à la place A10, sans que ce soit une question de volume. Ses données montraient une incidence relativement faible, mais cette catégorie arrivait en tête du sondage communautaire, avec des notes d’exploitabilité et d’impact supérieures à la moyenne. OWASP explique que récupérer le contenu d’une URL est devenu une fonctionnalité courante des applications web, et que les services cloud et les architectures plus complexes en aggravent les conséquences. Dans l’édition 2025, la SSRF n’a plus sa propre catégorie : CWE-918 figure parmi les faiblesses notables d’A01:2025, Broken Access Control, la catégorie en tête de liste. Le classement de MITRE va dans le même sens : CWE-918 figure au CWE Top 25 Most Dangerous Software Weaknesses chaque année de 2021 à 2025.

La raison tient à ce que la SSRF traverse directement les défenses sur lesquelles les équipes s’appuient le plus. La description d’OWASP souligne que l’attaque fonctionne même lorsque l’application se trouve derrière un pare-feu, un VPN ou une liste de contrôle d’accès réseau : la requête part de l’intérieur. Ses scénarios parcourent ce qui se trouve derrière ce périmètre : cartographier les hôtes internes et les ports ouverts d’après le résultat ou la durée d’une connexion, lire des fichiers locaux tels que `file:///etc/passwd`, dialoguer avec des services internes tels que `http://localhost:28017/`, puis exploiter ces services pour aller plus loin, jusqu’à l’exécution de code à distance ou au déni de service.

Les métadonnées cloud sont le scénario qui transforme une nuisance en intrusion. OWASP note que la plupart des fournisseurs cloud exposent les métadonnées d’instance à `http://169.254.169.254/`, une adresse du bloc

169.254.0.0/16 que le registre des adresses à usage particulier de l'IANA classe parmi les adresses de lien local. Sur Amazon EC2, le service de métadonnées d'instance à cette adresse est aussi celui où les logiciels de l'instance récupèrent les informations d'identification de sécurité AWS du rôle qui lui est associé. Il suffit de demander à un serveur prêt à récupérer toute URL pour n'importe qui de récupérer celles-ci.

Le mécanisme trouve sans cesse de nouveaux terrains. Parmi les exemples récents de la fiche CWE figurent une boîte à outils pour LLM qui récupérait des images à des URL arbitraires, faille déjà exploitée pour scanner des ports en avril 2026 (CVE-2026-33626), et un framework d'applications LLM dont le récupérateur d'URL pouvait être dirigé vers des adresses locales au moyen d'un en-tête `Location` forgé (CVE-2024-3095). Du point de vue du réseau, un agent qui parcourt le web pour un utilisateur est un serveur qui récupère des URL choisies par quelqu'un d'autre.

## À quoi cela ressemble en pratique

Considérer une application de discussion à `app.example.com` qui affiche une carte d'aperçu lorsqu'un lien est collé. Le navigateur envoie l'URL au serveur, puis celui-ci récupère la page pour en lire le titre et l'image :

```
POST /api/link-preview HTTP/1.1
Host: app.example.com
Content-Type: application/json

{"url": "https://news.example.org/2026/10/harbour-reopens"}
```

C'est le fonctionnement attendu. Voici maintenant la même requête avec une autre URL, celle qu'utilise la documentation AWS pour énumérer les métadonnées d'instance :

```
POST /api/link-preview HTTP/1.1
Host: app.example.com
Content-Type: application/json

{"url": "http://169.254.169.254/latest/meta-data/"}
```

Si le serveur tourne sur EC2 et que l'ancien IMDSv1 reste autorisé, il émet cette requête depuis l'intérieur de l'instance, le service de métadonnées répond, et tout ce que le code d'aperçu renvoie parvient à l'attaquant. Rien n'a été injecté et rien n'a planté. Le serveur a fait exactement ce pour quoi il était conçu, à la mauvaise adresse.

Le contrôle d'une destination commence par l'endroit où la requête arriverait réellement, après résolution du nom, plutôt que par le texte de l'URL. En Python, le module `ipaddress` de la bibliothèque standard utilise les registres d'adresses à usage particulier de l'IANA pour déterminer si une adresse est globale, en excluant les adresses de boucle locale, privées, de lien local, partagées et de documentation lorsqu'elles sont utilisées directement. La fonction auxiliaire suivante illustre ce classement, pas une défense complète contre la SSRF. Un déploiement doit aussi limiter les requêtes aux destinations unicast admissibles et tenir compte des traductions d'adresses telles que NAT64 : dans un réseau doté d'une passerelle de traduction, une adresse classée globale peut encore atteindre une destination IPv4 interdite qui est encodée dans cette adresse. Voici le test de classement de la fonction :

```
import ipaddress
import socket
from urllib.parse import urlsplit

ALLOWED_SCHEMES = {"http", "https"}


def resolve_public(url: str) -> tuple[str, int, list[str]]:
    parts = urlsplit(url)
    if parts.scheme not in ALLOWED_SCHEMES or not parts.hostname:
        raise ValueError("scheme or host not allowed")
    port = parts.port or (443 if parts.scheme == "https" else 80)
    infos = socket.getaddrinfo(parts.hostname, port, type=socket.SOCK_STREAM)
    addresses = sorted({info[4][0] for info in infos})
    for address in addresses:
        if not ipaddress.ip_address(address).is_global:
            raise ValueError(f"{address} is not a public address")
    return parts.hostname, port, addresses
```

Toutes les adresses obtenues par résolution du nom doivent réussir le test, pas seulement la première. Le client se connecte ensuite à l'une de ces adresses validées, tout en envoyant le nom d'hôte d'origine dans l'en-tête `Host` et en l'utilisant pour vérifier le certificat TLS, avec les redirections désactivées. Laisser le client HTTP résoudre une seconde fois le nom annulerait le contrôle. C'est la

démarche décrite par l'aide-mémoire d'OWASP pour le cas difficile des webhooks, où les utilisateurs choisissent les destinations et où aucune liste d'autorisation ne peut les nommer à l'avance. Lorsque les destinations sont connues, une liste de noms d'hôte exacts réduit encore le champ, mais l'aide-mémoire précise que les adresses résolues doivent toujours subir le même test.

Les contrôles applicatifs constituent une couche. Le récit de l'entreprise sur la messagerie avec bascule et journalisation des envois indique que le chemin d'envoi passe par un client protégé contre la SSRF, avec un délai maximal de 30 secondes. Derrière l'application, les conseils réseau d'OWASP restent valables : placer le code qui récupère les ressources distantes dans son propre segment réseau et lui appliquer des règles de pare-feu qui refusent par défaut, pour qu'un contrôle manqué rencontre une porte fermée.

## Où les problèmes surviennent

**Listes de refus et recherche de motifs.** Un filtre qui rejette les URL contenant `127.0.0.1`, `localhost` ou `169.254.169.254` a déjà perdu. La fonction POSIX `inet_addr()` accepte une adresse IPv4 en une à quatre parties, chacune en décimal, octal ou hexadécimal. Pour tout parseur qui suit ces règles, `2852039166` et `0xa9.0xfe.0xa9.0xfe` désignent donc tous deux l'adresse des métadonnées. La fiche CWE répertorie une CVE où des formes octales et hexadécimales ont contourné un contrôle conçu pour le décimal et déjoué une protection SSRF (CVE-2016-4029). La fiche du Top 10 d'OWASP est catégorique : « Ne pas atténuer la SSRF au moyen d'une liste de refus ou d'une expression régulière. » Décider d'après l'adresse résolue, comme ci-dessus, et autoriser plutôt que refuser.

**Vérifier une fois et se connecter plus tard.** Valider le nom d'hôte, puis laisser le client HTTP le résoudre à nouveau laisse une brèche. Avec le DNS rebinding, le nom de l'attaquant renvoie une adresse publique pour le contrôle et une adresse interne pour la connexion. Se connecter exactement à l'adresse validée et appliquer la même politique aux nouvelles tentatives et aux connexions de repli, explicitement citées par l'aide-mémoire d'OWASP.

**Suivre les redirections.** L'URL validée répond par une redirection vers `http://169.254.169.254/` ou une URL `file://`, et le client la suit sans rien demander. Parmi les exemples de CWE figure une bibliothèque de téléchargement par URL qui suivait les redirections vers des URL `file://` et `scp://` (CVE-2009-0037). Désactiver les redirections, ou traiter chaque saut comme une nouvelle URL soumise à l'intégralité du contrôle.

**Autoriser tous les schémas d'URL.** Un client qui comprend aussi `file`, `gopher` ou `dict` livre à l'attaquant des fichiers locaux ou un contrôle sur ce qui atteint un service interne bien supérieur à celui d'une requête HTTP. L'aide-mémoire d'OWASP cite ces schémas parmi ceux qu'une seconde requête peut utiliser. N'autoriser que `http` et `https`, sur les ports nécessaires à la fonctionnalité.

**Deux parseurs, deux hôtes.** L'aide-mémoire d'OWASP montre comment une seule chaîne peut désigner deux hôtes différents. Dans `http://good.example\@bad.example`, un parseur qui suit le WHATWG URL Standard traite la barre oblique inverse comme une barre oblique : le chemin commence donc là et l'hôte est `good.example`. En revanche, `urlsplit` de Python prend tout ce qui précède le dernier `@` pour les informations utilisateur et renvoie `bad.example`. Quand un composant valide et qu'un autre récupère, ils peuvent diverger sur la destination de la requête. Analyser une seule fois et rejeter toute URL dont l'hôte n'est pas interprété de la même façon par tous les parseurs traversés.

**Renvoyer la réponse brute.** Un point d'accès d'aperçu qui transmet le corps récupéré à l'appelant transforme une attaque qui ne pouvait que déduire le contenu interne d'après les erreurs et les délais en une attaque qui le lit. Ne renvoyer que ce dont la fonctionnalité a besoin, par exemple un titre et une référence d'image, jamais la réponse brute, comme le conseille la liste d'OWASP.

**Laisser IMDSv1 activé.** La documentation AWS indique que, par défaut, les deux versions du service de métadonnées d'instance sont utilisables. IMDSv2 exige d'abord un jeton de session, demandé avec un `PUT` vers `/latest/api/token`, puis envoyé dans l'en-tête `X-aws-ec2-metadata-token`. Dès qu'IMDSv2 devient obligatoire (`httpTokens=required`), IMDSv1 cesse de fonctionner. L'aide-mémoire d'OWASP présente IMDSv2 comme une mesure de défense en profondeur qui atténue certains cas de SSRF, pas tous. Exiger IMDSv2 sur chaque instance et corriger malgré tout l'application.

**Un réseau plat.** Lorsque le service de récupération peut atteindre la base de données, le panneau d'administration et le service de métadonnées, chaque contrôle manqué expose tout. Appliquer les protections réseau d'OWASP : segmentation, règles de pare-feu « refuser par défaut » pour le trafic interne et journaux des flux acceptés comme bloqués pour rendre les tentatives visibles.

# Pour aller plus loin

- CWE-918 : Server-Side Request Forgery (SSRF) — la fiche de la faiblesse : conséquences, faiblesse parente dite « confused deputy » et CVE observées.
- OWASP Top 10:2021 : A10 Server Side Request Forgery (SSRF) — les raisons de l'ajout de la catégorie, les protections réseau et applicatives et quatre scénarios d'attaque.
- OWASP Cheat Sheet Series : Server Side Request Forgery Prevention — les listes d'autorisation, la connexion aux seules adresses validées, les divergences entre parseurs et le cas des webhooks.
- OWASP Top 10:2025 : A01 Broken Access Control — la place de CWE-918 dans l'édition 2025.
- Amazon EC2 : Access instance metadata for an EC2 instance — IMDSv1 et IMDSv2, la demande de jeton et la manière de savoir si IMDSv2 est obligatoire.
- Documentation Python : ipaddress — `is_global` et les registres de l'IANA qu'il suit.

Glossaire

Choisissez selon vos contraintes réelles

Présentez le parcours de déploiement, les risques opérationnels et les personnes d'astreinte. Nous pouvons définir un changement avec des vérifications, une procédure de reprise et une responsabilité claire.

**Consulter les tarifs** Contactez-nous →

# Rendez la terminologie concrète

Demandez comment « Qu'est-ce que la SSRF ? » s'applique à votre organisation. Un exemple concret tiré de votre système actuel suffit pour engager la discussion.

**Découvrir nos prestations**

welcome@engineer.company +45 50 10 61 84 Copenhague, Danemark

Emportez cette page avec vous

Fiche de contact

Créé avec soin et amour par D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Droit d'auteur © 2025 – aujourd'hui · Engineer ApS · Tous droits réservés.

Imprimé depuis https://engineer.company/fr/glossary/ssrf/