Engineer

SSRF

атака, що змушує сервер надіслати запит на URL зловмисника

Що це таке

SSRF, або підробка запитів на боці сервера (англійською server-side request forgery), — це атака, за якої хтось дає серверу URL, а сервер іде й завантажує його від імені цієї людини, з того місця в мережі, де стоїть сам. Common Weakness Enumeration веде цю ваду як CWE-918 і описує її через те, чого код не робить: він отримує URL від компонента вище за ланцюжком, завантажує те, на що URL вказує, і ніколи не перевіряє, що запит іде саме туди, куди мав іти. Зловмисник не змінює в коді сервера жодного рядка. Він лише постачає адресу, а решту робить місце сервера в мережі.

У цьому місці й уся суть. Сервер у дата-центрі чи в хмарному обліковому записі зазвичай дістає до того, до чого відвідувач з інтернету не дістане: панелі адміністратора на внутрішній адресі, порту бази даних, панелі моніторингу, служби метаданих. Запис CWE прямо каже, що з цього випливає. Сервер стає проксі для зловмисника, його можна змусити сканувати порти на хостах внутрішньої мережі, а URL, що починаються з `file://`, `gopher://` чи `tftp://`, дають змогу читати документи на самій машині або краще керувати тим, що містить запит. CWE відносить SSRF до ширшої вади, CWE-441, ненавмисного проксі, відомого як «confused deputy»: компонент із привілеями виконує роботу для того, хто привілеїв не має.

Двері зазвичай відчиняє цілком звичайна функція. Шпаргалка OWASP із запобігання SSRF (cheat sheet) наводить два типові випадки: аватар, який застосунок завантажує за URL, що його ввів користувач, і URL вебхуків чи зворотних викликів (callback), які користувачі реєструють, щоб застосунок згодом їх викликав. Попередній перегляд посилань і кнопки імпорту за URL належать до тієї самої родини, як і будь-яка функція, де сервер іде за адресою, яку написав хтось інший. У назві запису CWE термін пишеться з великих літер, Server-Side Request Forgery, а сам запис згадує давнішу назву того самого явища, XSPA, скорочення від Cross Site Port Attack.

Застосуйте поняття до рішення

Застосуйте «Що таке SSRF?» до конкретного рішення. Ми можемо пов'язати поняття із системою, яку ви створюєте або купуєте. Облікові записи, паролі й права налаштовані так, щоб доступ відповідав роботі, а звільнені втрачали його того ж дня.

Переглянути наші послуги Зв'язатися з нами →

Перевірте припущення за терміном

Використайте пояснення та пов'язані роботи, щоб сформулювати запитання для проєкту. Ми можемо перевірити припущення разом до вибору інструмента. Написала 1 000 правил RBAC для Linux-серверів — 1,000 правил доступу написано, щоб усунути вразливість

Написала 1 000 правил RBAC для Linux-серверів Зв'язатися з нами →

Чому це важливо

OWASP додав SSRF до свого Top 10 у 2021 році як A10, і не через поширеність. Дані показували відносно низьку частоту, але категорію найчастіше обирали учасники опитування спільноти, а її оцінки експлуатованості та впливу були вищими за середні. OWASP пояснює це тим, що завантаження URL стало звичайною функцією вебзастосунків, а хмарні сервіси й складніші архітектури роблять наслідки серйознішими. У редакції 2025 року SSRF уже не має окремої категорії: CWE-918 названо серед помітних вад категорії A01:2025, Broken Access Control, яка стоїть на першому місці списку. Рейтинг MITRE вказує в той самий бік: CWE-918 входить до CWE Top 25 Most Dangerous Software Weaknesses щороку з 2021 до 2025.

Причина в тому, що SSRF проходить крізь той захист, на який команди покладаються найбільше. OWASP наголошує, що атака працює, навіть коли застосунок стоїть за брандмауером, VPN чи мережевим списком контролю доступу, бо запит виходить зсередини. Приклади сценаріїв від OWASP — це екскурсія тим, що ховається за периметром: скласти карту внутрішніх хостів і відкритих портів за результатами з'єднань або за тим, скільки триває з'єднання, прочитати локальні файли на кшталт `file:///etc/passwd`, звернутися до внутрішніх сервісів на кшталт `http://localhost:28017/` і використати ці сервіси, щоб піти далі, аж до віддаленого виконання коду чи відмови в обслуговуванні.

Хмарні метадані — це сценарій, що перетворює неприємність на злам. OWASP зазначає, що більшість хмарних провайдерів тримають метадані екземпляра за адресою `http://169.254.169.254/`, у блоці 169.254.0.0/16, який реєстр адрес спеціального призначення IANA позначає як link-local. На Amazon EC2 служба метаданих екземпляра за цією адресою — ще й місце,

звідки програми на екземплярі отримують облікові дані безпеки AWS для ролі, прив'язаної до нього. Сервер, який завантажить будь-який URL для будь-кого, можна попросити завантажити й їх.

Цей шаблон постійно знаходить нові місця. Серед найсвіжіших прикладів у записі CWE — набір інструментів для LLM, що завантажував довільні URL зображень і в квітні 2026 року зазнав реальних атак, де його використовували для сканування портів (CVE-2026-33626), і фреймворк для LLM-застосунків, чий завантажувач URL можна було спрямувати на локальні адреси підробленим заголовком `Location` (CVE-2024-3095). Агент, що ходить вебом для користувача, з погляду мережі — це сервер, який завантажує URL, обрані кимось іншим.

Як це виглядає на практиці

Візьмімо чат-застосунок на `app.example.com`, який показує картку попереднього перегляду, коли хтось вставляє посилання. Браузер надсилає URL серверу, а сервер завантажує сторінку, щоб прочитати її заголовок і зображення:

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

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

Так функція працює за задумом. А тепер той самий запит з іншим URL, тим, яким документація AWS показує список метаданих екземпляра:

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

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

Якщо сервер працює на EC2 і старіший IMDSv1 досі дозволено, він надсилає цей запит зсередини екземпляра, служба метаданих відповідає, і все, що код попереднього перегляду повертає, потрапляє до зловмисника. Нічого не

впровадили, нічого не впало. Сервер зробив саме те, для чого його збудували, тільки для чужої адреси.

Перевірка адреси призначення починається з того, куди запит насправді піде після розв'язання імені, а не з тексту URL. У Python модуль `ipaddress` зі стандартної бібліотеки використовує реєстри адрес спеціального призначення IANA, щоб класифікувати адресу як глобальну чи ні, відсікаючи адреси, що безпосередньо належать до loopback, приватних, link-local і спільних діапазонів, а також діапазонів для документації. Наведена нижче допоміжна функція ілюструє цю класифікацію, а не повний захист від SSRF. Розгорнута система має також обмежувати запити дозволеними адресами призначення для одноадресної передачі й ураховувати трансляцію адрес, як-от NAT64: у мережі зі шлюзом трансляції адреса, класифікована як глобальна, все одно може вести до заблокованої вбудованої IPv4-адреси призначення. Тест класифікації в допоміжній функції такий:

```
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
```

Пройти перевірку мусить кожна адреса, на яку розв'язується ім'я, а не лише перша. Далі завантаження підключається до однієї з перевірених адрес, але надсилає початкове ім'я хоста в заголовку `Host` і перевіряє за ним сертифікат TLS, а переспрямування вимкнено. Якщо дозволити HTTP-клієнту розв'язати ім'я вдруге, перевірка втрачає сенс. Саме такий підхід шпаргалка OWASP описує для складного випадку, вебхуків, де адресатів обирають користувачі й жоден список дозволених адрес не може назвати їх

заздалегідь. Там, де адресати відомі, список точних імен хостів звужує коло ще більше, але шпаргалка чітко каже, що розв'язані адреси все одно мають пройти той самий тест.

Перевірки в застосунку — лише один шар. У портфоліо є опис того, як побудовано пошту з failover і журналом доставки, і там сказано, що сам шлях надсилання проходить через клієнт із захистом від SSRF і тайм-аутом 30 секунд. За межами застосунку й далі діють мережеві поради OWASP: винести код, що завантажує віддалені ресурси, в окремий сегмент мережі й дати йому правила брандмауера, які за замовчуванням усе забороняють, щоб пропущена перевірка наткнулася на зачинені двері.

Де виникають проблеми

Списки заборон і пошук за шаблоном. Фільтр, що відкидає URL із `127.0.0.1`, `localhost` чи `169.254.169.254`, уже програв. Функція POSIX `inet_addr()` приймає адресу IPv4 з однієї, двох, трьох або чотирьох частин, кожна з яких може бути десятковою, вісімковою чи шістнадцятковою, тож для будь-якого парсера, що дотримується цих правил, і `2852039166`, і `0xa9.0xfe.0xa9.0xfe` — це адреса служби метаданих. Запис CWE наводить CVE, у якій вісімкові й шістнадцяткові форми проскочили повз перевірку, написану для десяткових, і обійшли захист від SSRF (CVE-2016-4029). Top 10 від OWASP каже про це прямо: SSRF не слід стримувати списком заборон чи регулярним виразом. Вирішуйте за розв'язаною адресою, як показано вище, і дозволяйте, а не забороняйте.

Перевірка зараз, з'єднання потім. Якщо перевірити ім'я хоста, а потім дати HTTP-клієнту розв'язати його знову, лишається шпарина. За DNS rebinding ім'я зловмисника відповідає публічною адресою під час перевірки та внутрішньою під час з'єднання. Підключайтеся саме до тієї адреси, яку перевірили, і застосовуйте ту саму політику до повторних спроб і резервних з'єднань, які шпаргалка OWASP згадує окремо.

Переспрямування, за якими йде клієнт. Перевірений URL відповідає переспрямуванням на `http://169.254.169.254/` або на URL `file://`, і клієнт іде за ним, нічого не питаючи. Серед прикладів у записі CWE є бібліотека для завантаження URL, яка йшла за переспрямуваннями на URL `file://` і `scp://` (CVE-2009-0037). Вимкніть переспрямування або розглядайте кожен перехід як новий URL, що знову проходить усю перевірку.

Клієнт говорить не лише HTTP. Якщо він розуміє ще й `file`, `gopher` чи `dict`, зловмисник може прочитати локальні файли або дістати значно більше контролю над тим, що потрапляє до внутрішнього сервісу, ніж дає HTTP-запит. Шпаргалка OWASP називає саме ці типи URL серед тих, якими може скористатися другий запит. Дозволяйте лише `http` і `https` і лише ті порти, які потрібні функції.

Два парсери, два хости. Шпаргалка OWASP показує, як один рядок може вказувати на два різні хости. У `http://good.example\@bad.example` парсер, що дотримується стандарту URL від WHATWG, вважає зворотну скісну риску роздільником шляху й читає хост як `good.example`, тоді як `urlsplit` у Python бере все до останнього `@` як дані користувача й повертає `bad.example`. Коли один компонент перевіряє, а інший завантажує, вони можуть розійтися в тому, куди йде запит. Розбирайте URL один раз і відкидайте будь-який URL, хост якого не однаково читають усі парсери на його шляху.

Повернення сирої відповіді. Ендпоінт попереднього перегляду, що віддає завантажений вміст тому, хто його викликав, перетворює атаку, яка могла лише здогадуватися про те, що всередині, з помилок і часу відповіді, на атаку, яка це читає. Повертайте лише те, що потрібно функції, наприклад заголовок і посилання на зображення, і ніколи сиру відповідь, як і радить список OWASP.

Увімкнений IMDSv1. Документація AWS каже, що за замовчуванням можна користуватися обома версіями служби метаданих екземпляра. IMDSv2 спершу вимагає токен сесії, який запитують через `PUT` на `/latest/api/token` і потім надсилають у заголовку `X-aws-ec2-metadata-token`, а коли IMDSv2 стає обов'язковим (`httpTokens=required`), IMDSv1 перестає працювати. Шпаргалка OWASP називає IMDSv2 додатковим рівнем захисту, що пом'якшує деякі випадки SSRF, але не всі. Зробіть IMDSv2 обов'язковим на кожному екземплярі й однаково виправте застосунок.

Пласка мережа. Коли сервіс, що завантажує URL, може дістатися до бази даних, панелі адміністратора й служби метаданих, кожна пропущена перевірка стає фатальною. Застосуйте мережеві заходи OWASP: сегментацію, правила брандмауера, що за замовчуванням забороняють внутрішній трафік, і журнали дозволених і заблокованих з'єднань, щоб спроба розвідки стала помітною.

Що почитати далі

- CWE-918: Server-Side Request Forgery (SSRF) — запис про ваду: наслідки, батьківська вада «confused deputy» і зафіксовані CVE.
- OWASP Top 10:2021: A10 Server Side Request Forgery (SSRF) — чому категорію додали, заходи на рівні мережі й застосунку та чотири сценарії атак.
- OWASP Cheat Sheet Series: Server Side Request Forgery Prevention — списки дозволених адрес, підключення лише до перевірених адрес, розбіжності парсерів і випадок вебхуків.
- OWASP Top 10:2025: A01 Broken Access Control — де CWE-918 стоїть у редакції 2025 року.
- Amazon EC2: Access instance metadata for an EC2 instance — IMDSv1 та IMDSv2, запит токена і як дізнатися, чи IMDSv2 обов'язковий.
- Python documentation: ipaddress — `is_global` і реєстри IANA, на які він спирається.

Словник

Обирайте за фактичними умовами

Опишіть шлях розгортання, експлуатаційні ризики та чергову команду. Ми можемо визначити зміну з перевірками, шляхом відновлення та чіткою відповідальністю.

Дивитися, скільки це коштує Зв'язатися з нами →

Зробіть термінологію практичною

Запитайте, як «Що таке SSRF?» стосується вашої організації. Конкретного прикладу з поточної системи достатньо для початку розмови.

Переглянути наші послуги

welcome@engineer.company +45 50 10 61 84 Копенгаген, Данія

Візьміть цю сторінку із собою

Візитівка

Дбайливо зроблено з любов'ю — D.

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

Авторське право © 2025 – дотепер · Engineer ApS · Усі права захищено.

Надруковано з https://engineer.company/uk/glossary/ssrf/