Engineer

SSRF

an attack that makes a server fetch a URL the attacker chose

What it is

SSRF, short for server-side request forgery, is the attack in which someone hands a server a URL and the server goes and fetches it on their behalf, from wherever the server happens to sit. The Common Weakness Enumeration catalogues it as CWE-918 and describes the flaw by what the code fails to do: it takes a URL from an upstream component, retrieves whatever the URL points at, and never makes sure the request is going to the destination it was meant for. The attacker changes nothing in the server's code. They supply an address, and the server's position in the network does the rest.

That position is the whole point. A server in a data centre or a cloud account can usually reach things a visitor on the internet cannot: an admin panel bound to an internal address, a database port, a monitoring dashboard, a metadata service. The CWE entry spells out what follows. The server becomes a proxy for the attacker, it can be made to scan ports on hosts in the internal network, and URL schemes such as `file://`, `gopher://` and `tftp://` can read documents on the machine itself or give the attacker more control over what the request contains. CWE places SSRF under a broader weakness, CWE-441, the unintended proxy or "confused deputy": a component with privileges doing work for someone who has none.

The feature that opens the door is usually an ordinary one. The OWASP prevention cheat sheet names two typical cases: an avatar image the application downloads from a URL the user typed, and webhook or callback URLs that users register so the application can call them later. Link previews and import-from-URL buttons belong to the same family, as does any feature in which the server follows an address that someone else wrote. CWE's own title spells the term Server-Side Request Forgery, with capitals, and the entry lists an older name for the same thing, XSPA, short for Cross Site Port Attack.

Use the concept in a decision

Put “What is SSRF?” to work in a real decision. We can help relate the concept to the system you are building or buying. Accounts, passwords and permissions arranged so people hold exactly the access their job needs, and leavers lose it that day.

Explore our services Get in touch →

Test the assumptions behind the term

Use the explanation and linked work to frame questions for your project. We can examine the assumptions with you before a tool is chosen. Wrote 1,000 RBAC rules to secure Linux servers — 1,000 access rules written to close a security gap

Wrote 1,000 RBAC rules to secure Linux servers Get in touch →

Why it matters

OWASP added SSRF to its Top 10 in 2021, as A10, and not because of volume. Its data showed a relatively low incidence rate, but the category was the top pick of the community survey, and its exploit and impact ratings were above average. OWASP’s explanation is that fetching a URL has become a routine feature of web applications, and that cloud services and more complex architectures make the consequences more severe. In the 2025 edition SSRF no longer has a category of its own: CWE-918 is named among the notable weaknesses of A01:2025, Broken Access Control, the category at the top of the list. MITRE’s ranking points the same way, with CWE-918 on the CWE Top 25 Most Dangerous Software Weaknesses in every year from 2021 to 2025.

The reason is that SSRF walks straight through the defences teams lean on most. OWASP’s description stresses that the attack works even when the application sits behind a firewall, a VPN or a network access control list, because the request leaves from inside them. Its example scenarios are a tour of what sits behind that perimeter: mapping internal hosts and open ports from connection results or the time a connection takes, reading local files such as `file:///etc/passwd`, talking to internal services such as `http://localhost:28017/`, and abusing those services to go further, as far as remote code execution or denial of service.

Cloud metadata is the scenario that turns a nuisance into a breach. OWASP notes that most cloud providers keep instance metadata at `http://169.254.169.254/`, an address in the 169.254.0.0/16 block that IANA’s special-purpose registry lists as link local. On Amazon EC2, the instance metadata service at that address is also where software on the instance retrieves AWS security credentials for the role attached to it. A server that will fetch any URL for anyone can be asked to fetch those.

The pattern keeps finding new homes. The CWE entry's recent examples include an LLM toolkit that fetched arbitrary image URLs, a flaw that attackers were already using to scan ports in April 2026 (CVE-2026-33626), and an LLM application framework whose URL retriever could be steered to local addresses with a crafted `Location` header (CVE-2024-3095). An agent that browses the web for a user is, from the network's point of view, a server fetching URLs that someone else chose.

What it looks like in practice

Take a chat application at `app.example.com` that shows a preview card when someone pastes a link. The browser sends the URL to the server, and the server fetches the page to read its title and 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"}
```

That is the feature working. Now the same request with another URL, the one the AWS documentation uses to list instance metadata:

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

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

If the server runs on EC2 with the older IMDSv1 still allowed, it sends that request from inside the instance, the metadata service answers, and whatever the preview code passes back lands with the attacker. Nothing was injected and nothing crashed. The server did exactly what it was built to do, for the wrong address.

A destination check starts with where the request would actually go, after the name has been resolved, rather than at the text of the URL. In Python, the standard library's `ipaddress` module uses IANA's special-purpose registries to classify whether an address is global, excluding direct loopback, private, link-

local, shared and documentation addresses. The following helper illustrates that classification, not a complete SSRF defence. A deployment must also limit requests to eligible unicast destinations and account for address translation such as NAT64: on a network with a translating gateway, an address classified as global may still reach a blocked embedded IPv4 destination. The helper's classification test is:

```
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
```

Every address the name resolves to has to pass, not just the first. The fetch then connects to one of those validated addresses, while still sending the original host name in the `Host` header and using it for TLS certificate checks, with redirects switched off. Letting the HTTP client resolve the name a second time would undo the check. That is the shape the OWASP cheat sheet describes for the hard case, webhooks, where users choose the destinations and no allow list can name them in advance. Where the destinations are known, an allow list of exact host names narrows things further, though the cheat sheet is clear that the resolved addresses still need the same test.

Application checks are one layer. Our write-up on building email with failover and delivery logging notes that the send path goes through an SSRF-protected client with a 30-second timeout. Behind the application, OWASP's network-layer advice still applies: put the code that fetches remote resources in its own network segment and give it firewall rules that deny by default, so a missed check meets a closed door.

Where it goes wrong

Deny lists and pattern matching. A filter that rejects URLs containing `127.0.0.1`, `localhost` or `169.254.169.254` has already lost. POSIX `inet_addr()` accepts an IPv4 address in one to four parts, each in decimal, octal or hexadecimal, so to any parser that follows those rules `2852039166` and `0xa9.0xfe.0xa9.0xfe` are both the metadata address. The CWE entry lists a CVE in which octal and hexadecimal forms slipped past a check written for decimal and defeated an SSRF protection (CVE-2016-4029). OWASP's Top 10 entry is blunt about it: "Do not mitigate SSRF via the use of a deny list or regular expression." Decide on the resolved address, as above, and allow rather than deny.

Checking once and connecting later. Validating the host name and then letting the HTTP client look it up again leaves a gap. With DNS rebinding, the attacker's name answers with a public address for the check and an internal one for the connection. Connect to the exact address you validated, and apply the same policy to retries and fallback connections, which the OWASP cheat sheet calls out by name.

Following redirects. The validated URL answers with a redirect to `http://169.254.169.254/` or to a `file://` URL, and the client follows it without asking. The CWE entry's examples include a URL-downloading library that followed redirects to `file://` and `scp://` URLs (CVE-2009-0037). Turn redirects off, or treat every hop as a new URL that goes through the whole check again.

Allowing every scheme. A client that also speaks `file`, `gopher` or `dict` hands an attacker local files, or far more control over what reaches an internal service than an HTTP request allows. The OWASP cheat sheet lists those schemes among the ones a second request can use. Allow `http` and `https` only, on the ports the feature needs.

Two parsers, two hosts. The OWASP cheat sheet shows how one string can name two different hosts. In `http://good.example\@bad.example`, a parser that follows the WHATWG URL Standard handles the backslash like a forward slash, so the path starts there and the host is `good.example`, while Python's `urlsplit` takes everything before the last `@` as user information and returns `bad.example`. When one component validates and another fetches, they can disagree about where the request goes. Parse once, and reject any URL whose host is not read the same way by every parser it passes through.

Returning the raw response. A preview endpoint that passes the fetched body back to the caller turns an attack that could only infer what sits inside, from errors and timings, into one that reads it. Return only what the feature needs, such as a title and an image reference, never the raw response, as OWASP's list advises.

Leaving IMDSv1 on. The AWS documentation says that by default both versions of the instance metadata service can be used. IMDSv2 first requires a session token, requested with a `PUT` to `/latest/api/token` and then sent in the `X-aws-ec2-metadata-token` header, and once IMDSv2 is required (`httpTokens=required`), IMDSv1 stops working. The OWASP cheat sheet calls IMDSv2 a defence-in-depth mechanism that mitigates some instances of SSRF, not all of them. Require IMDSv2 on every instance, and fix the application anyway.

A flat network. When the fetching service can reach the database, the admin panel and the metadata service, every missed check is total. Apply OWASP's network-layer controls: segmentation, "deny by default" firewall rules for internal traffic, and logs of accepted and blocked flows so a probe shows up.

Further reading

- CWE-918: Server-Side Request Forgery (SSRF) — the weakness entry: consequences, the confused-deputy parent and observed CVEs.
- OWASP Top 10:2021: A10 Server Side Request Forgery (SSRF) — why the category was added, network and application controls, and four attack scenarios.
- OWASP Cheat Sheet Series: Server Side Request Forgery Prevention — allow lists, connecting only to validated addresses, parser disagreement and the webhook case.
- OWASP Top 10:2025: A01 Broken Access Control — where CWE-918 sits in the 2025 edition.
- Amazon EC2: Access instance metadata for an EC2 instance — IMDSv1 and IMDSv2, the token request and how to tell whether IMDSv2 is required.
- Python documentation: ipaddress — `is_global` and the IANA registries it follows.

Glossary

Choose for your actual constraints

Bring the deployment path, the operational risks and the people on call. We can scope a change with checks, a recovery route and clear ownership.

See what it costs Get in touch →

Make the terminology practical

Ask how “What is SSRF?” applies to your organisation. A concrete example from your current system is enough to start the discussion.

Explore our services

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

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

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/glossary/ssrf/