Engineer

SSRF

en attack som får en server att hämta angriparens URL

Vad det är

SSRF, kort för server-side request forgery, är attacken där någon ger en server en URL och servern hämtar den åt dem, från den plats i nätverket där servern råkar stå. Common Weakness Enumeration för upp den som CWE-918 och beskriver felet utifrån vad koden underlåter att göra: den tar emot en URL från en komponent längre upp i kedjan, hämtar det som URL:en pekar på och försäkrar sig aldrig om att anropet går till det mål det var avsett för. Angriparen ändrar inte en rad i serverns kod. Angriparen lämnar en adress, och serverns plats i nätverket sköter resten.

Den platsen är hela poängen. En server i ett datacenter eller ett molnkonto når oftast saker som en besökare på internet inte når: en adminpanel bunden till en intern adress, en databasport, en övervakningspanel, en metadatatjänst. CWE-posten säger rakt ut vad som följer. Servern blir proxy åt angriparen, den kan fås att skanna portar på maskiner i det interna nätverket, och URL:er som börjar med `file://`, `gopher://` eller `tftp://` kan läsa dokument på själva maskinen eller ge angriparen mer kontroll över vad anropet innehåller. CWE placerar SSRF under en bredare svaghet, CWE-441, den oavsiktliga proxyn eller ”confused deputy”: en komponent med behörigheter som utför arbete åt någon som saknar dem.

Det som öppnar dörren är oftast en helt vanlig funktion. OWASP:s fusklapp om förebyggande (cheat sheet) nämner två typiska fall: en profilbild som applikationen laddar ned från en URL som användaren skrivit in, och webhook- eller callback-URL:er som användarna registrerar så att applikationen kan anropa dem senare. Förhandsvisningar av länkar och knappar för import från URL hör till samma familj, liksom varje funktion där servern följer en adress som någon annan har skrivit. CWE stavar själv beteckningen Server-Side Request Forgery med versaler, och posten nämner ett äldre namn på samma sak, XSPA, kort för Cross Site Port Attack.

Använd begreppet i ett beslut

Använd ”Vad är SSRF?” i ett konkret beslut. Vi kan knyta begreppet till systemet ni bygger eller köper. Konton, lösenord och behörigheter ordnade så att alla har exakt den åtkomst deras arbete kräver, och de som slutar förlorar den samma dag.

Se våra tjänster Hör av er →

Granska antagandena bakom ordet

Använd förklaringen och det länkade arbetet för att formulera frågor till projektet. Vi kan undersöka antagandena tillsammans innan ett verktyg väljs. Skrev 1 000 RBAC-regler som säkrar Linux-servrar — 1,000 åtkomstregler, skrivna för att täppa till en säkerhetslucka

Skrev 1 000 RBAC-regler som säkrar Linux-servrar Hör av er →

Varför det spelar roll

OWASP tog med SSRF på sin Topp 10 år 2021, som A10, och inte på grund av mängden. Datan visade en relativt låg förekomst, men kategorin var förstahandsvalet i OWASP:s enkät bland gemenskapen, och betygen för utnyttjande och konsekvens låg över genomsnittet. OWASP förklarar det med att hämtning av URL:er har blivit en rutinfunktion i webbapplikationer, och att molntjänster och mer komplexa arkitekturer gör följderna allvarligare. I 2025 års utgåva har SSRF ingen egen kategori längre: CWE-918 finns bland de anmärkningsvärda svagheterna under A01:2025, Broken Access Control, kategorin som ligger överst på listan. MITRE:s rangordning pekar åt samma håll, med CWE-918 på CWE Top 25 Most Dangerous Software Weaknesses varje år från 2021 till 2025.

Skälet är att SSRF går rakt igenom de försvar som team lutar sig mest mot. OWASP betonar att attacken fungerar även när applikationen står bakom en brandvägg, ett VPN eller en åtkomstlista i nätverket, eftersom anropet går ut inifrån. OWASP:s exempelscenarier är en rundtur i det som döljer sig bakom den perimetern: att kartlägga interna maskiner och öppna portar utifrån om anslutningar lyckas eller hur lång tid de tar, att läsa lokala filer som `file:///etc/passwd`, att tala med interna tjänster som `http://localhost:28017/` och att missbruka de tjänsterna för att komma vidare, ända till fjärrkörning av kod eller överbelastning.

Molnets metadata är scenariot som gör en olägenhet till ett intrång. OWASP påpekar att de flesta molnleverantörer har instansens metadata på `http://169.254.169.254/`, en adress i blocket 169.254.0.0/16 som IANA:s register över adresser för särskilda ändamål för upp som link-local. På Amazon EC2 är instansens metadatatjänst på den adressen också platsen där programvara på instansen hämtar AWS-autentiseringsuppgifter för den roll som är kopplad till den. En server som hämtar vilken URL som helst åt vem som helst kan ombes hämta dem också.

Mönstret dyker hela tiden upp på nya ställen. Bland de senaste exemplen i CWE-posten finns en verktygslåda för LLM:er som hämtade godtyckliga bild-URL:er och utnyttjades aktivt i april 2026 för portskanning (CVE-2026-33626), och ett ramverk för LLM-applikationer där URL-hämtaren kunde styras mot lokala adresser med en konstruerad `Location` -header (CVE-2024-3095). En agent som surfar på webben åt en användare är, sett från nätverkets håll, bara en server som hämtar URL:er som någon annan har valt.

Så ser det ut i praktiken

Ta en chattapplikation på `app.example.com` som visar ett förhandsvisningskort när någon klistrar in en länk. Webbläsaren skickar URL:en till servern, och servern hämtar sidan för att läsa titeln och bilden:

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

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

Så fungerar funktionen som avsett. Se nu samma anrop med en annan URL, den som AWS-dokumentationen använder för att lista instansens 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/"}
```

Om servern körs på EC2 och den äldre IMDSv1 fortfarande är tillåten skickar den anropet inifrån instansen, metadatatjänsten svarar, och det som förhandsvisningskoden skickar tillbaka hamnar hos angriparen. Ingenting injicerades och ingenting kraschade. Servern gjorde precis det den var byggd för, bara för fel adress.

En kontroll av destinationen börjar med vart anropet faktiskt skulle gå när namnet har slagits upp, inte med texten i URL:en. I Python använder standardbibliotekets modul `ipaddress` IANA:s register över adresser för särskilda ändamål för att klassificera om en adress är global. Det utesluter adresser som

direkt tillhör loopback-, privata, link-local-, delade eller dokumentationsområden. Hjälpfunktionen nedan illustrerar denna klassificering, inte ett fullständigt skydd mot SSRF. En driftsättning måste också begränsa anrop till tillåtna unicastdestinationer och ta hänsyn till adressöversättning som NAT64: i ett nätverk med en översättande gateway kan en adress som klassificerats som global fortfarande nå en blockerad, inbäddad IPv4-destination. Hjälpfunktionens klassificeringstest är:

```
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
```

Varje adress som namnet slås upp till måste klara testet, inte bara den första. Själva hämtningen ansluter sedan till en av de godkända adresserna, men skickar fortfarande det ursprungliga värdnamnet i `Host`-headern och använder det för att kontrollera TLS-certifikatet, och omdirigeringar är avstängda. Låter man HTTP-klienten slå upp namnet en gång till är kontrollen förgäves. Det är den formen som OWASP:s fusklapp beskriver för det svåra fallet, webhooks, där användarna själva väljer målen och ingen lista över tillåtna adresser kan räkna upp dem i förväg. Där målen är kända snävar en lista över exakta värdnamn in fältet ytterligare, men fusklappen är tydlig med att de uppslagna adresserna ändå måste igenom samma test.

Kontrollerna i applikationen är ett lager. Vår beskrivning av e-post med failover och leveransloggning nämner att själva sändningsvägen går genom en SSRF-skyddad klient med en timeout på 30 sekunder. Bakom applikationen gäller OWASP:s råd för nätverkslagret fortfarande: lägg koden som hämtar externa

resurser i ett eget nätverkssegment och ge den brandväggsregler som nekar allt som inte uttryckligen är tillåtet, så att en förbisedd kontroll möter en stängd dörr.

Var det går fel

Spärrlistor och mönstermatchning. Ett filter som avvisar URL:er med `127.0.0.1`, `localhost` eller `169.254.169.254` har redan förlorat. POSIX-funktionen `inet_addr()` godtar en IPv4-adress i en till fyra delar, var och en decimal, oktal eller hexadecimal, så för varje parser som följer de reglerna är både `2852039166` och `0xa9.0xfe.0xa9.0xfe` metadataadressen. CWE-posten nämner en CVE där oktala och hexadecimala former slank förbi en kontroll skriven för decimaltal och slog ut ett SSRF-skydd (CVE-2016-4029). OWASP:s Topp 10 säger det rakt ut: SSRF ska inte motverkas med en spärrlista eller ett reguljärt uttryck. Avgör saken på den uppslagna adressen, som ovan, och tillåt hellre än förbjud.

Kontroll nu, anslutning senare. Att validera värdnamnet och sedan låta HTTP-klienten slå upp det igen lämnar en lucka. Med DNS rebinding svarar angriparens namn med en publik adress vid kontrollen och en intern vid anslutningen. Anslut till exakt den adress du validerade, och tillämpa samma policy på nya försök och reservanslutningar, som OWASP:s fusklapp nämner vid namn.

Omdirigeringar som följs. Den godkända URL:en svarar med en omdirigering till `http://169.254.169.254/` eller till en `file://`-URL, och klienten följer den utan att fråga. Bland exemplen i CWE-posten finns ett bibliotek för nedladdning av URL:er som följde omdirigeringar till `file://`- och `scp://`-URL:er (CVE-2009-0037). Stäng av omdirigeringar, eller behandla varje hopp som en ny URL som går igenom hela kontrollen igen.

Klienten talar mer än HTTP. Förstår den också `file`, `gopher` eller `dict` kan en angripare läsa lokala filer eller få långt mer kontroll över vad som når fram till en intern tjänst än ett HTTP-anrop ger. OWASP:s fusklapp nämner just dessa bland de URL-typer som ett nytt anrop kan använda. Tillåt bara `http` och `https`, och bara de portar som funktionen behöver.

Två parsrar, två mål. OWASP:s fusklapp visar hur en och samma sträng kan peka på två olika värdnamn. I `http://good.example\@bad.example` läser en parser som följer WHATWG:s URL-standard bakåtsnedstrecket som avgränsare i sökvägen och värdnamnet som `good.example`, medan Pythons `urlsplit` tar allt före det

sista `@` som användaruppgifter och returnerar `bad.example`. När en komponent validerar och en annan hämtar kan de vara oense om vart anropet går. Tolka URL:en en gång, och avvisa varje URL vars värdnamn inte läses likadant av alla parsrar den passerar.

Råsvaret skickas tillbaka. En slutpunkt för förhandsvisningar som skickar tillbaka det hämtade innehållet till anroparen gör en attack som bara kunde gissa sig till vad som finns innanför, utifrån fel och svarstider, till en som kan läsa det. Returnera bara det som funktionen behöver, till exempel en titel och en hänvisning till en bild, och aldrig råsvaret, som OWASP:s lista råder.

IMDSv1 är fortfarande på. AWS-dokumentationen säger att båda versionerna av instansens metadatatjänst kan användas som standard. IMDSv2 kräver först en sessionstoken, som hämtas med en `PUT` till `/latest/api/token` och sedan skickas i headern `X-aws-ec2-metadata-token`, och när IMDSv2 krävs (`httpTokens=required`) slutar IMDSv1 att fungera. OWASP:s fusklapp kallar IMDSv2 ett extra försvarslager som dämpar vissa fall av SSRF, inte alla. Kräv IMDSv2 på alla instanser, och rätta applikationen ändå.

Ett platt nätverk. När tjänsten som hämtar URL:er kan nå databasen, adminpanelen och metadatatjänsten blir varje förbisedd kontroll ödesdiger. Använd OWASP:s kontroller på nätverkslagret: segmentering, brandväggsregler som i grunden nekar intern trafik och loggning av tillåtna och blockerade anslutningar, så att ett sökande efter öppningar syns.

Läs mer

- CWE-918: Server-Side Request Forgery (SSRF) — svaghetsposten: följderna, föräldern ”confused deputy” och de observerade CVE:erna.
- OWASP Top 10:2021: A10 Server Side Request Forgery (SSRF) — varför kategorin kom med, kontroller i nätverk och applikation och fyra angreppsscenarier.
- OWASP Cheat Sheet Series: Server Side Request Forgery Prevention — listor över tillåtna adresser, anslutning bara till validerade adresser, oense parsrar och webhook-fallet.
- OWASP Top 10:2025: A01 Broken Access Control — var CWE-918 står i 2025 års utgåva.
- Amazon EC2: Access instance metadata for an EC2 instance — IMDSv1 och IMDSv2, tokenanropet och hur man ser om IMDSv2 krävs.

- Python documentation: ipaddress — `is_global` och IANA-registren som den följer.

Ordlista

Välj utifrån era faktiska förutsättningar

Ta med driftsättningsvägen, driftsriskerna och dem som har jour. Vi kan avgränsa en förändring med kontroller, en väg tillbaka och tydligt ansvar.

Se vad det kostar Hör av er →

Gör begreppen praktiska

Fråga hur ”Vad är SSRF?” gäller i er organisation. Ett konkret exempel från ert nuvarande system räcker för att börja samtalet.

Se våra tjänster

welcome@engineer.company +45 50 10 61 84 Köpenhamn, Danmark

Ta med dig den här sidan

Visitkort

Omsorgsfullt skapad med kärlek av D.

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

Upphovsrätt © 2025 – nu · Engineer ApS · Alla rättigheter förbehållna.

Utskrivet från https://engineer.company/sv/glossary/ssrf/