Engineer

SSRF

攻撃者が選んだURLをサーバーに取得させる攻撃

どのようなものか

SSRFはserver-side request forgeryの略で、サーバーにURLを渡し、そのサーバーがいるネットワーク上の位置から、代わりに内容を取得させる攻撃です。Common Weakness Enumerationは、これをCWE-918として分類しています。その説明は、コードが何を確認していないかに着目しています。上流のコンポーネントからURLを受け取り、そのURLが指すものを取得する一方で、リクエストが本来の宛先へ向かっていることを確かめない、という欠陥です。攻撃者はサーバーのコードを一切変更しません。攻撃者が渡すのはアドレスだけで、あとはサーバーのネットワーク上の位置を利用します。

その位置こそが、この攻撃の要点です。データセンターやクラウドアカウント内のサーバーは、通常、インターネット上の訪問者には届かないものへ到達できます。内部アドレスだけで待ち受ける管理画面、データベースのポート、監視ダッシュボード、メタデータサービスなどです。CWEの項目は、その結果を具体的に示しています。サーバーが攻撃者のプロキシとなり、内部ネットワーク上のホストのポートを走査させられます。また、`file://`、`gopher://`、`tftp://`などのURLスキームを使うと、マシン自体の文書を読んだり、リクエストの内容を攻撃者がより細かく制御したりできます。CWEは、SSRFを、より広い弱点であるCWE-441の下に位置付けています。これは、権限を持つコンポーネントが、権限のない相手のために働く、意図しないプロキシ、または「混乱した代理人」の問題です。

入り口を作るのは、たいてい普通の機能です。OWASPの防止策チートシートは、典型例を二つ挙げています。利用者が入力したURLからアプリがダウンロードするアバター画像と、後からアプリに呼び出させるため、利用者が登録するWebhookやコールバックのURLです。リンクのプレビューやURLから取り込むボタンも、同じ仲間です。他者が書いたアドレスをサーバーがたどる機能なら、どれも該当します。CWEの項目名ではServer-Side Request Forgeryと大文字を交えて表記され、同じ攻撃の古い呼び名として、Cross Site Port Attackの略であるXSPAも掲載されています。

概念を判断に生かす

「SSRFとは？」を実際の判断に生かしましょう。その概念を、開発または購入するシステムに結び付けて考えるお手伝いをします。 アカウント、パスワード、権限を整理し、業務に必要なアクセスだけを付与し、退職者の権限は当日に解除します。

サービスを見る お問い合わせ →

用語の背後にある前提を検証する

解説とリンク先の実績を参考に、プロジェクトについての質問を整理してください。ツールを選ぶ前に、前提を一緒に検討できます。 Linuxサーバーを守る1,000件のRBACルールを作成 — 1,000 セキュリティ上の穴を塞ぐアクセスルール

Linuxサーバーを守る1,000件のRBACルールを作成 お問い合わせ →

なぜ重要なのか

OWASPがSSRFをTop 10のA10として追加したのは2021年ですが、発生件数の多さが理由ではありません。データ上の発生率は比較的低かった一方、コミュニティー調査では最も多く選ばれ、悪用のしやすさと影響の評価も平均を上回っていました。OWASPは、URLの取得がウェブアプリの日常的な機能になり、クラウドサービスや複雑なアーキテクチャーが結果をより深刻にしていると説明しています。2025年版では、SSRFは独立した分類ではなくなりました。CWE-918は、最上位の分類であるA01:2025のBroken Access Controlで、代表的な弱点の一つに挙げられています。MITREの順位も同じ方向を示しています。CWE-918は、2021年から2025年まで毎年、CWE Top 25 Most Dangerous Software Weaknessesに入っています。

理由は、SSRFが、チームの頼りにする主要な防御をそのまま通り抜けることにあります。OWASPは、アプリがファイアウォール、VPN、ネットワークアクセス制御リストの背後にあっても、リクエストはその内側から出るため攻撃が成立すると強調しています。その攻撃例をたどると、境界の内側に何があるかが分かります。接続の結果や所要時間から内部ホストや開いているポートを調べ、`file:///etc/passwd` などのローカルファイルを読み、`http://localhost:28017/` などの内部サービスと通信します。そのサービスをさらに悪用し、リモートコード実行やサービス拒否にまで至る場合があります。

クラウドのメタデータが絡むと、単なる厄介事がセキュリティ侵害に発展します。OWASPによると、多くのクラウド事業者は、インスタンスのメタデータを `http://169.254.169.254/` で提供しています。このアドレスは、IANAの特殊用途レジストリーでリンクローカルとされる169.254.0.0/16の範囲にあります。Amazon EC2では、このアドレスにあるインスタンスメタデータサービスから、インスタンス上のソフトウェアが、そのインスタンスに付与されたロールのAWSセキュリティ認証情報も取得します。誰のためにでも任意のURLを取得するサーバーには、それらを取得するよう要求できてしまいます。

同じパターンの攻撃は、新しい用途にも広がり続けています。CWEの最近の事例には、任意の画像URLを取得していたLLMツールキットの欠陥が含まれます。これは、2026年4月にはすでにポート走査に悪用されていました（CVE-2026-33626）。別のLLMアプリフレームワークでは、細工した `Location` ヘッダーによってURL取得機能をローカルアドレスへ向けられました（CVE-2024-3095）。利用者のためにウェブを閲覧するエージェントは、ネットワークの視点では、他者が選んだURLを取得するサーバーです。

実際の活用例

誰かがリンクを貼るとプレビューカードを表示する、 `app.example.com` 上のチャットアプリを考えます。ブラウザーがURLをサーバーへ送り、サーバーがそのページを取得して、タイトルと画像を読みます。

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

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

これが通常の機能です。次に、AWSのドキュメントがインスタンスメタデータの一覧取得に使っているURLへ、同じリクエストの宛先を変えます。

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

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

サーバーがEC2上で動き、古いIMDSv1が引き続き許可されていれば、リクエストはインスタンス内から送信され、メタデータサービスが応答します。プレビューのコードが返す内容は、そのまま攻撃者へ届きます。何かが注入されたわけでも、クラッシュしたわけでもありません。サーバーは、作られたとおりの処理を、違う宛先に対して行ったのです。

宛先の確認は、URLの文字列ではなく、名前解決の後でリクエストが実際にどこへ向かうかから始めます。Pythonの標準ライブラリーの `ipaddress` モジュールは、IANAの特殊用途レジストリーを使い、アドレスがグローバルかどうかを分類します。アドレス自体がループバック、プライベート、リンクローカル、共

有、文書用の範囲に属する場合は除外します。以下のヘルパーは、その分類を示す例であり、完全なSSRF対策ではありません。実際の導入では、リクエストを許容されるユニキャスト宛先に限定し、NAT64などのアドレス変換も考慮する必要があります。変換ゲートウェイを持つネットワークでは、グローバルと分類されたアドレスが、アドレス内に埋め込まれた、遮断すべきIPv4宛先へ到達する場合があります。このヘルパーが行う分類テストは、次のとおりです。

```
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
```

名前解決で返るすべてのアドレスが、最初の一つだけでなく、検査を通らなければなりません。その後の取得処理は、検証済みアドレスの一つへ接続しつつ、`Host` ヘッダーでは元のホスト名を送り、TLS証明書の検証にもその名前を使います。リダイレクトは無効にします。HTTPクライアントにもう一度名前解決させると、検査が無意味になります。これは、OWASPのチートシートが、難しいケースであるWebhookに対して説明している構成です。利用者が宛先を選ぶため、許可リストで事前にすべてを列挙できません。宛先が分かっている場合は、完全一致のホスト名の許可リストでさらに絞り込めます。ただし、チートシートは、解決されたアドレスにも同じ検査が必要だと明記しています。

アプリ内の検査は、一つの層にすぎません。フェイルオーバーと配信ログを備えたメール機能の構築記事では、送信経路にSSRF対策付きクライアントを使い、タイムアウトを30秒に設定していると記載しています。アプリの背後でも、OWASPのネットワーク層の助言が当てはまります。外部リソースを取得するコードを独立したネットワーク区画に置き、ファイアウォールのルールを既定で拒否する形にします。そうすれば、検査に漏れがあっても、その先の扉は閉じています。

問題が起きるところ

拒否リストとパターン照合。 `127.0.0.1`、`localhost`、`169.254.169.254` を含むURLを拒否するフィルターでは、すでに防御が破られています。POSIXの `inet_addr()` は、IPv4アドレスを一つから四つの部分として受け取り、それぞれを十進数、八進数、十六進数で記述できます。そのルールに従うパーサーにとって、`2852039166` と `0xa9.0xfe.0xa9.0xfe` は、どちらもメタデータのアドレスです。CWEには、十進数を想定した検査を八進数と十六進数の表記がすり抜け、SSRF対策を破った事例が掲載されています（CVE-2016-4029）。OWASPのTop 10の項目は、「拒否リストや正規表現を使ってSSRFを緩和しないでください」と明言しています。前述のように、名前解決後のアドレスで判断し、拒否するものではなく、許可するものを決めます。

一度確認しただけで、後の接続を任せる。 ホスト名を検証した後でHTTPクライアントに再度検索させると、隙間ができます。DNSリバインディングでは、攻撃者が管理するホスト名が、検査時には公開アドレスを、接続時には内部アドレスを返します。検証したそのアドレスへ接続し、再試行やフォールバック接続にも同じポリシーを適用します。OWASPのチートシートも、それらを明示しています。

リダイレクトをたどる。 検証済みのURLが、`http://169.254.169.254/` や `file://` のURLへの転送を返し、クライアントが確認せずにたどってしまいます。CWEの事例には、`file://` や `scp://` のURLへのリダイレクトをたどったURLダウンロードライブラリーがあります（CVE-2009-0037）。リダイレクトを無効にするか、一段ごとに新しいURLとして扱い、検査全体をやり直します。

すべてのスキームを許可する。 `file`、`gopher`、`dict` にも対応するクライアントは、攻撃者へローカルファイルを渡したり、内部サービスへ届く内容を、HTTPリクエストよりはるかに細かく制御させたりします。OWASPのチートシートは、二つ目のリクエストが使えるスキームとして、これらを挙げています。その機能に必要なポート上の `http` と `https` だけを許可します。

二つのパーサーが、二つのホストを読む。 OWASPのチートシートは、一つの文字列が異なる二つのホストを指し得ることを示しています。`http://good.example\@bad.example` では、WHATWG URL Standardに従うパーサーは、バックスラッシュをスラッシュと同様に扱うため、そこでパスが始まり、ホストは `good.example` となります。一方、Pythonの `urlsplit` は、最後の `@` より前をすべてユーザー情報として扱い、`bad.example` を返します。検証するコンポ

ーネントと取得するコンポーネントが別だと、宛先の解釈が食い違うことがあります。一度だけ解析し、通過するすべてのパーサーがホストを同じように読めないURLは拒否します。

生の応答を返してしまう。 取得した本文をそのまま呼び出し側へ返すプレビュー用エンドポイントは、エラーや所要時間から内部を推測するだけだった攻撃を、内容を読める攻撃へ変えてしまいます。OWASPの助言のとおり、タイトルや画像への参照など、その機能に必要なものだけを返し、生の応答は決して返しません。

IMDSv1を有効にしたままにする。 AWSのドキュメントによると、既定では両バージョンのインスタンスメタデータサービスを使えます。IMDSv2では、まず `PUT` で `/latest/api/token` へセッショントークンを要求し、その後は `X-aws-ec2-metadata-token` ヘッダーでそのトークンを送る必要があります。IMDSv2を必須にすると（ `httpTokens=required` ）、IMDSv1は使えなくなります。OWASPのチートシートは、IMDSv2を、SSRFの一部を緩和する多層防御の仕組みと位置付け、すべてを防ぐものとはしていません。すべてのインスタンスでIMDSv2を必須にし、それでもアプリの欠陥は修正します。

ネットワークが分割されていない。 取得サービスが、データベース、管理画面、メタデータサービスのすべてへ到達できると、一つの検査漏れが全面的な影響を持ちます。ネットワークの区画分け、内部通信を既定で拒否するファイアウォールのルール、許可した通信と遮断した通信のログという、OWASPのネットワーク層の制御を適用します。そうすれば、探りを入れる通信も見えるようになります。

参考資料

- CWE-918: Server-Side Request Forgery (SSRF) — この弱点の項目です。影響、親分類である混乱した代理人の問題、観測されたCVEを説明しています。
- OWASP Top 10:2021: A10 Server Side Request Forgery (SSRF) — 分類が追加された理由、ネットワークとアプリの制御、四つの攻撃例を説明しています。
- OWASP Cheat Sheet Series: Server Side Request Forgery Prevention — 許可リスト、検証済みアドレスだけへの接続、パーサー間の解釈の違い、Webhookのケースを説明しています。
- OWASP Top 10:2025: A01 Broken Access Control — 2025年版でCWE-918がどこに位置付けられるかを示しています。

- Amazon EC2: Access instance metadata for an EC2 instance — IMDSv1とIMDSv2、トークンの要求、IMDSv2が必須かどうかを確認する方法を説明しています。
- Python documentation: ipaddress — `is_global` と、その判断に使うIANAレジストリーを説明しています。

用語集

実際の制約に合わせて選ぶ

デプロイの流れ、運用上のリスク、緊急対応の担当者をお知らせください。検査、復旧手段、明確な責任の所在を伴う変更の範囲を定められます。

料金を見る お問い合わせ →

用語を実務に結び付ける

「SSRFとは？」がお客様の組織にどう当てはまるかをご相談ください。現在のシステムの具体例が一つあれば、話を始められます。

サービスを見る

welcome@engineer.company +45 50 10 61 84 デンマーク、コペンハーゲン

このページを持ち歩く

連絡先カード

心を込めて丁寧に制作： D.

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

著作権 © 2025 – 現在 · Engineer ApS · すべての権利を留保します。

印刷元： https://engineer.company/ja/glossary/ssrf/