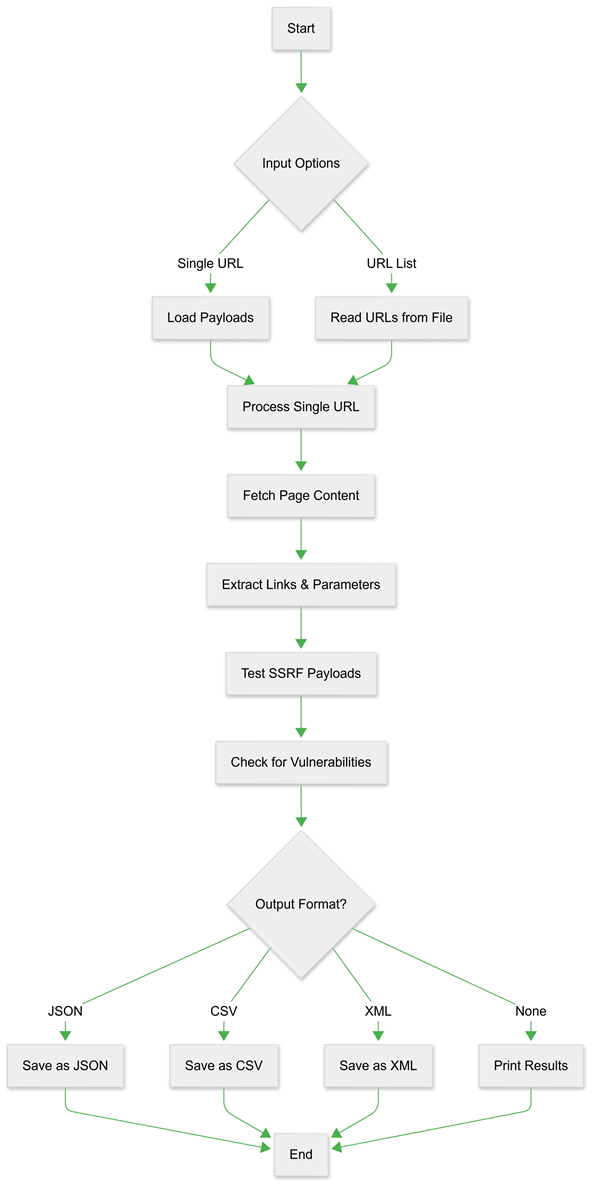
This script implements a sophisticated tool for detecting Server-Side Request Forgery (SSRF) vulnerabilities in web applications. SSRF is a security issue where an attacker can abuse server-side functionality to access internal resources or services.

The scanner automates the process of discovering SSRF flaws by injecting crafted payloads into various input parameters and analyzing the responses. It also supports integration with out-of-band interaction platforms (like Burp Collaborator) to detect blind SSRF.



Key Features

1. Multi-Payload Testing

The tool tests targeted URLs by injecting various malicious payloads using the following techniques:

Manipulating URL Scheme: Modifying the URL format and integrating it into the payload  
Internal IP/Port Combinations: Attempts to access internal network resources (e.g., http://192.168.0.1:8080).

URL Encoded Paths: Uses encoded URLs to bypass input sanitization (e.g., http%3A%2F%2Fattacker.com).

Payload upload :Payloads are loaded from external files (payload.txt, pathpayload.txt), allowing users to add or modify entries.

2. Comprehensive Parameter Discovery

The tool automatically discovers and tests all potential injection points by:

Extracting form actions and parameters from HTML content

Identifying URL-like values in form fields

Discovering hidden parameters that accept URL inputs

Analyzing select elements and their options for URL values

3. Advanced Detection Techniques

The scanner implements sophisticated detection methods:

Response length analysis to identify successful internal resource access

Error message pattern detection to filter false positives

Operating system fingerprinting through server header analysis

Time-based detection for blind SSRF vulnerabilities

4. Collaborative Testing

The tool integrates with external collaboration services to:

Detect out-of-band vulnerabilities

Test for blind SSRF scenarios

Validate DNS-based vulnerabilities

Identify shellshock vulnerabilities through HTTP headers

Scanning Methodology

1. Initial Reconnaissance Phase

The scanner begins by crawling the target application to:

Map all accessible endpoints and forms

Identify parameters that accept URL inputs

Discover relationships between different application components

Build a comprehensive attack surface model

2. Payload Injection Phase

The tool systematically tests discovered parameters with:

Basic SSRF payloads targeting common internal services

Advanced obfuscation techniques to bypass filters

Protocol scheme manipulations

Internal IP range scanning (192.168.0.0/24)

Localhost bypass techniques

3. Response Analysis Phase

Each injection attempt is carefully analyzed by:

Comparing response lengths against baseline

Checking for error messages indicating successful exploitation

Detecting subtle differences in response content

Fingerprinting accessed resources through response characteristics

4. Reporting Phase

Vulnerability findings are presented through:

Detailed console output

Structured report generation (JSON, CSV, XML)

Comprehensive metadata including timestamps and response characteristics

Technical Implementation Details

Concurrency Model

The scanner employs a sophisticated threading model that:

Utilizes configurable thread pools for efficient scanning

Implements graceful error handling for network issues

Maintains session state across related requests

Optimizes resource usage during large-scale scans

Proxy Support

The tool can operate through proxy servers with:

HTTP/HTTPS proxy configuration

Consistent session management

Proper header injection through proxies

Input Handling

The scanner processes various input formats:

Single URL targets

Files containing lists of URLs

Custom payload lists for specialized testing

Path-based payloads for specific attack scenarios

Output Formats

1. JSON Output

Provides machine-readable output with complete vulnerability details including:

Target URLs and vulnerable parameters

Successful payloads

Response metadata (status codes, lengths)

Timestamps and OS detection results

2. CSV Output

Delivers spreadsheet-compatible results focusing on:

Essential vulnerability information

Clean formatting for analysis

Easy integration with other tools

3. XML Output

Structured XML format suitable for:

Enterprise vulnerability management systems

Automated processing pipelines

Long-term archiving of scan results