

Blind SSRF

Server-Side Request Forgery vulnerabilities can be "blind." In these cases, even though the request is processed, we can't see the backend server's response. For this reason, blind SSRF vulnerabilities are more difficult to detect and exploit.

We can detect blind SSRF vulnerabilities via out-of-band techniques, making the server issue a request to an external service under our control. To detect if a backend service is processing our requests, we can either use a server with a public IP address that we own or services such as:

- [Burp Collaborator](#) (Part of Burp Suite professional. Not Available in the community edition)
- <http://pingb.in>

Blind SSRF vulnerabilities could exist in PDF Document generators and HTTP Headers, among other locations.

[◀ Previous](#)[Next ➔](#)[Mark Complete & Next](#)

Table of Contents

[Introduction to Server-Side Attacks](#)[Abusing Intermediary Applications](#)[AJP Proxy](#)[Nginx Reverse Proxy & AJP](#)[Apache Reverse Proxy & AJP](#)

Server-Side Request Forgery (SSRF)

[Server-Side Request Forgery \(SSRF\) Overview](#)[SSRF Exploitation Example](#)[Blind SSRF](#)[Blind SSRF Exploitation Example](#)[Time-Based SSRF](#)

Server-Side Includes (SSI) Injection

[Server-Side Includes Overview](#)[SSI Injection Exploitation Example](#)

Edge-Side Includes (ESI) Injection

[Edge-Side Includes \(ESI\)](#)

Server-Side Template Injections

[Introduction to Template Engines](#)[SSTI Identification](#)[SSTI Exploitation Example 1](#)[SSTI Exploitation Example 2](#)[SSTI Exploitation Example 3](#)

Extensible Stylesheet Language Transformations Server-Side Injections

[Attacking XSLT](#)

Skills Assessment

[Server-Side Attacks - Skills Assessment](#)

My Workstation

OFFLINE

[Start Instance](#)

00 / 1 spawns left

