PROXIES AND PORT REDIRECTION FOR WEB APPLICATION PENTESTING

by William Marchand*

*Team lead and pentester in Open-Sec. He holds security certifications such as OSCP, eWPT, and eCPPT, as well as serving as an instructor for various IT and security programs. Speaker ar various national and international events. University teacher. (https://www.linkedin.com/in/wmarchand/)

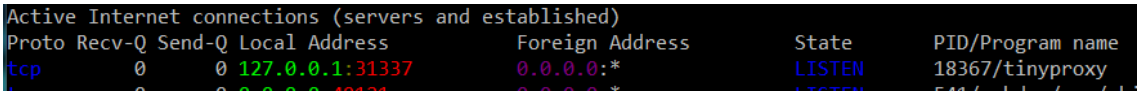
During a pentest of web applications, a good practice is to keep records (logs, snapshots, etc.) of the tests that are performed for later checking, clarifications or because the client eventually requests them, but in many cases it is also requested that the tests can be "monitored in real time" by the client's team. For this purpose, the point source (public) IP addresses must be permanent (at least for the duration of the service).

The pentester, on the other hand, usually keeps his necessary tools arsenal that are configured on his local machine (not only default installations, but environment parameters; licenses, if any, plug-ins, libraries, etc.) connected to a network with non-dedicated commercial Internet access (therefore, no permanent static IP).

So, you could think about setting up, for example, a VPS or instance in the cloud with all the necessary tools and the advantage of the static IP address; however, it is often not the best option because of: data security issues (logs and tool outputs, for example), for the economic factor (more computational resources are required, therefore, more cost) , because having an instance/VPS in the cloud does not have the GUI environment enabled, because it is a shared resource for several services or because it simply has the utility of maintaining static public IP addresses and behave as a proxy.

Considering that this type of testing is 100% remote, then what should we do to meet the requirements for monitoring, security and availability of the tools? Here are the basic steps (it is not the only solution):

1. Have a VPS or cloud instance with a static public IP address.
2. Configure a light http proxy service, such as Tinyproxy, listening on a TCP port such as 31337 and on the localhost interface (127.0.0.1). Why on the localhost interface? When a computer is exposed to the Internet, it is likely to suffer attack attempts, at least port scans, and it is best to avoid or minimize them. Rules or filters can also be applied at the network level.

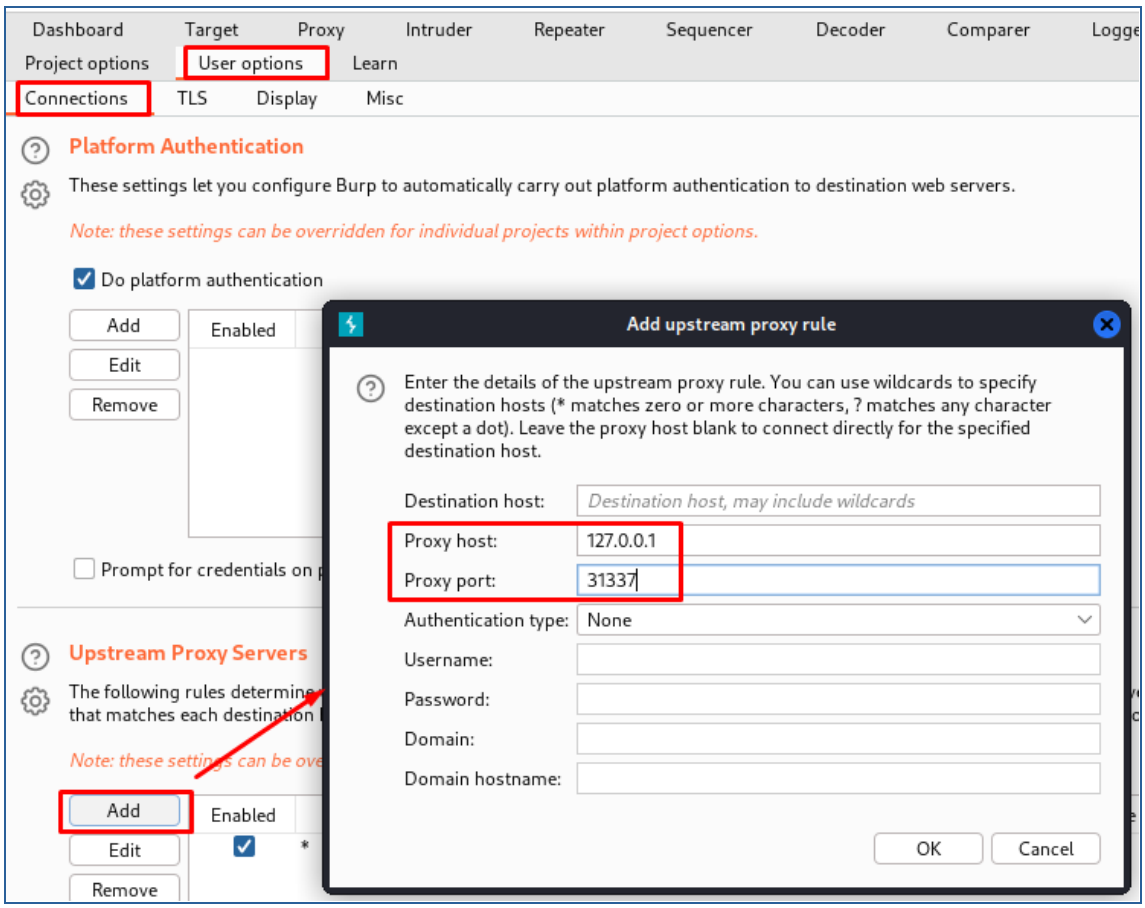


3. As the port of the http proxy is listening on the localhost interface, it is necessary to perform a port forwarding to our local computer. On our computer run:

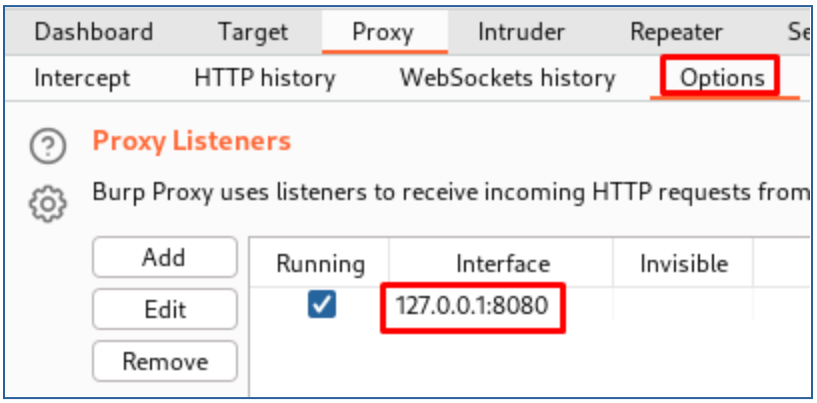
ssh **-L** 127.0.0.1:31337:127.0.0.1:31337 user@vps-open-sec.com

The parameter can be interpreted as: "bring to local port 31337, the remote port 31337 of the vps-open-sec.com server".

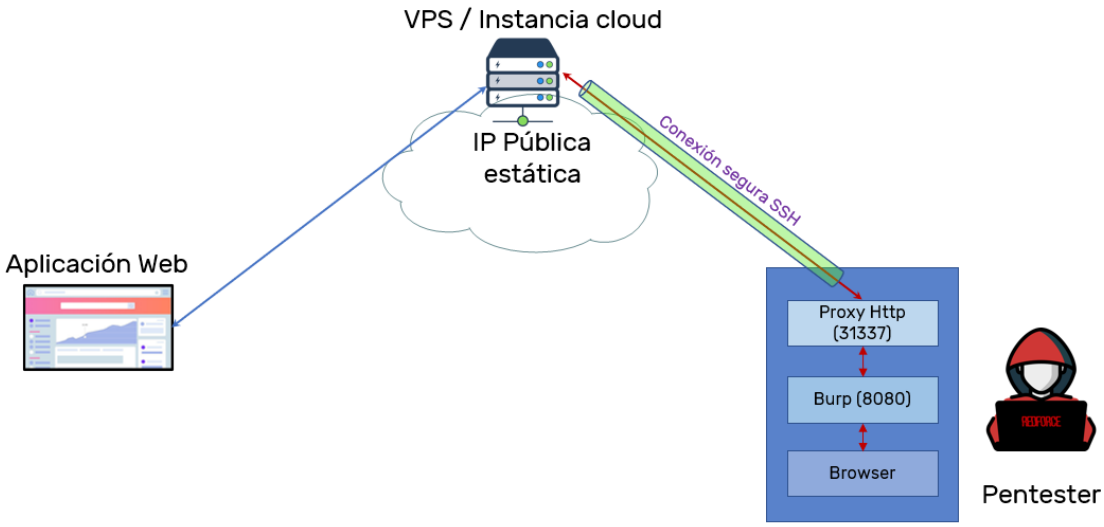
4. In our Burp you must specify in “Upstream Proxy Servers” (User Options --> Connections) the proxy server configured on port 31337



5. The Burp Proxy Listener configuration kept the one we normally use (default or not).



6. Finally, in our browser, configure the proxy to navigate over the evaluation target application. All the traffic will be redirected through the secure SSH connection and the VPS (configured as an http proxy server) and our local computer, with the characteristic that the origin visible traffic for the client will be from the IP address of our VPS or cloud instance.



This type of environment can be applied for different tools or cases with some variations; for example, ZAP, Nessus, nmap or others. What other utilities do you find? Useful for a red team exercise? Protection from the pentester side?

The possibilities offered by this type of techniques and tools (forward ports, tunnels, proxy) are multiple, these include remote pentesting environments for internal network infrastructure, post-exploitation, data exfiltration, etc. In future installments we will expand with more cases.