NAS over VPN

I built a fully functional, self-hosted VPN using WireGuard with the wg-easy management interface and deployed it on an Oracle Cloud Infrastructure compute instance. The goal of the project was to create a secure, always-available remote access solution that I could use from multiple devices while keeping administration simple and maintaining a small security footprint. Oracle Cloud provided a stable public endpoint, and WireGuard provided the underlying VPN protocol with modern, key-based authentication. To simplify deployment and make the service easy to manage and restart, I ran the WireGuard/wg-easy stack using Docker, which allowed me to containerize the VPN service, keep configuration persistent, and update or redeploy the system without manually rebuilding the server environment. Using wg-easy on top of WireGuard significantly reduced the complexity of provisioning and managing peers by generating client configurations automatically and allowing quick onboarding through downloadable config files or QR codes.

On the server side, I configured the instance so it could accept incoming VPN connections over UDP on the WireGuard port and route client traffic correctly once a tunnel was established. This required aligning both layers of firewalling: Oracle Cloud network rules (security lists or network security groups) and the operating system's firewall configuration. After opening only the required ports and confirming the service was reachable externally, I enabled the networking behavior needed for a cloud-hosted VPN, including forwarding and network address translation so VPN clients could either reach the broader internet through the instance or reach specific resources behind the VPN depending on the routing policy. I also limited access to the wg-easy web UI since it is a management surface; while it is password protected, restricting access further reduced the overall attack surface. After deployment, I validated the VPN by connecting from a client device, confirming successful handshakes, verifying that the client received an internal VPN address, and checking that network traffic flowed through the tunnel as expected.

In addition to secure connectivity, I extended the project by creating a network-attached storage (NAS) setup using SMB. The intent was to provide persistent file storage that would remain consistent across devices and sessions, rather than relying on files stored locally on whichever client happened to be connected. I configured SMB sharing on the server-side storage and ensured it was accessible only through the private network path provided by the VPN. This design meant the storage service was not broadly exposed to the public internet; instead, access depended on first establishing the WireGuard tunnel, which provided an additional security boundary. After the SMB share was operational, I mapped it as a network drive on my client machine, which allowed the storage to behave like a persistent mounted drive for saving and retrieving files. With the drive mapping in place, I was able to maintain continuity of files across reboots and across different client devices, effectively creating a centralized storage location that I could reach securely from anywhere.

Overall, this project delivered a practical remote-access environment that combined secure VPN tunneling with persistent network storage. WireGuard and wg-easy provided a reliable and efficient VPN with straightforward peer management, Docker provided a clean and repeatable way to run and maintain the VPN service, and SMB-based NAS functionality added real utility by enabling centralized persistence through a mapped network drive. Future improvements could include adding monitoring to track uptime and connection attempts, further tightening management access paths (such as UI access only via VPN or SSH tunneling), and automating the entire deployment using infrastructure-as-code so the environment could be recreated quickly and consistently.

Sources:

This project report is based on the official documentation for both wg-easy and WireGuard. wg-easy's documentation and repository were used to reference the recommended installation approaches (e.g., container-based deployment) and general usage/administration of the web UI for managing WireGuard peers. (wg-easy.github.io) WireGuard's official site and technical materials were used to reference protocol behavior and design goals (secure, modern VPN tunneling) as well as the formal specification details described in the technical whitepaper and protocol documentation. (wireguard.com)