

WireGuard: Connection, Encryption, and Management

Connection

- WireGuard peers communicate over **UDP** (default port `51820`).
- Each peer has a **static keypair**:
 - Private key (kept secret)
 - Public key (shared with peers)
- A peer is identified *solely* by its **public key**.
- Config defines **AllowedIPs**, which act as both:
 - A routing table (what subnets should go through which peer)
 - An access control list (what traffic each peer is allowed to send).

Handshake

- Based on the **NoiseIK pattern** (from the Noise Protocol Framework).
- Initiator sends an ephemeral key + its static public key (encrypted).
- Responder verifies, replies with its ephemeral key.
- Both sides derive a **shared session key**.

Result: a secure tunnel established in a single round trip.

Encryption & Security

- **Curve25519** for key exchange (fast, secure elliptic curve).
- **ChaCha20-Poly1305** for authenticated encryption:
 - ChaCha20 for confidentiality
 - Poly1305 for message authentication
- **BLAKE2s** for hashing & key derivation.
- **SipHash24** for hashtable keys (resists DoS).

- **HKDF** (HMAC-based key derivation function) for key material.
-

Key Rotation

- Session keys are **ephemeral**:
 - Re-keying every ~120 seconds or every 2^{24} messages.
 - Ensures forward secrecy.
 - Even if a session key leaks, it expires quickly.
-

Data Transport

- Once handshake completes:
 - Packets matching `AllowedIPs` are encrypted into UDP datagrams.
 - UDP encapsulation means NAT traversal works well.
 - The remote peer decapsulates and routes into its local stack.
 - WireGuard runs in **kernel space** (Linux), making it very fast.
-

Management Model

- WireGuard has **no central control plane**:
 - Config is static: just keys, endpoints, and AllowedIPs.
 - No built-in certificates, CRLs, or renegotiation logic.
 - This makes it:
 - Simple to audit (very small codebase ~4,000 lines).
 - Resistant to misconfiguration.
 - Easier to secure than large, complex VPN stacks (e.g., IPsec, OpenVPN).
-

Peer Management

- To add a new client:
 1. Generate a keypair on the client.

2. Add client's public key to server config with its AllowedIPs.
 3. Give client server's public key and endpoint.
- Removal is just deleting the peer from config.



Performance Characteristics

- Kernel-level implementation (Linux) → very low overhead.
- Extremely fast crypto primitives (ChaCha20, Curve25519).
- Roaming-friendly: if client's IP changes, tunnel persists.
- MTU-friendly: runs over UDP, less overhead than IPsec/OpenVPN.



Summary

WireGuard is:

- **Cryptographically modern:** uses the latest safe primitives.
- **Operationally simple:** config = keypairs + AllowedIPs.
- **High performance:** fast handshakes, high throughput.
- **Secure by design:** small codebase, forward secrecy, minimal attack surface.

In practice: once configured, WireGuard feels “invisible”—traffic flows securely as if peers were directly connected on the same LAN.