How to Enable Root Login on Your VM (BYOVM)

Elestio's **BYOVM (Bring Your Own VM)** workflow requires **direct root-level SSH access** to the target host. Many cloud providers disable or restrict root login by default; this document outlines how to safely re-enable it while maintaining a strong security posture.

Architectural Requirement for Root Access

Elestio performs **low-level system provisioning and lifecycle management** directly on the host. This includes:

- Writing configuration and state under `/root/` other privileged paths
- Installing and managing system packages and daemons
- Modifying network configuration, firewall rules, and kernel-adjacent settings
- Executing privileged deployment and orchestration tasks

Relying on a non-root user with `sudo` is intentionally avoided due to

- **Inconsistent `sudo` availability and behavior** across distributions and cloud images
- **Hard dependencies on root-owned paths** (e.g., `/root`, system-level configs)
- **Increased operational complexity** from privilege escalation (TTY requirements, environment differences, edge cases)
- **Higher failure surface** during automated provisioning

From an automation and reliability standpoint, **direct root access is deterministic and reduces ambiguity**.

Security Model: Root Access with SSH Keys

Enabling root login is **not inherently insecure** when implemented correctly.

Elestio enforces the following security guarantees:

- **Password authentication is permanently disabled**
- **Only SSH key-based authentication is permitted**
- **Root login is restricted to key-based access (`prohibit-password`)**

Implications

- No password-based login is possible under any circumstances
- Attack vectors such as brute-force or credential stuffing are eliminated
- Access is strictly limited to holders of the corresponding private key

Blocking the root while allowing SSH access to an enabled user does **not materially improve security**. Any compromise of such a user trivially escalates to root (`sudo su`). The **true security boundary is the SSH private key**, not the username.

Step 1: Remove Provider-Imposed Restrictions

Some providers (notably AWS and GCP) inject a forced command `/root/.ssh/authorized_keys` to prevent root login:

```
no-port-forwarding,no-agent-forwarding,no-X11-forwarding,command="echo 'Please login as the user \"ubuntu\" rather than the user \"root\".';echo;sleep 10;exit 142"
```

This must be removed.

Execute as a privileged non-root user (e.g., `ubuntu`, `ec2-user`):

```
sudo sed -e "s/echo;sleep 10;exit 142//g" -i /root/.ssh/authorized_keys
```

Validate:

```
sudo cat /root/.ssh/authorized_keys
```

Expected state: **only raw public key entries, no `command="..."` prefix**.

Step 2: Configure SSH Daemon

Inspect the current configuration:

```
sudo grep -i PermitRootLogin /etc/ssh/sshd_config
```

Enforce the correct policy:

```
sudo sed -i 's/^.*PermitRootLogin.*/PermitRootLogin prohibit-password/' /etc/ssh/sshd_config
```

Semantics

- `prohibit-password`
  - ✅ (allows) SSH key authentication
  - ❌ (blocks) password authentication

This is the **recommended and secure baseline configuration**.

Step 3: Reload SSH Service

Apply changes without terminating the active session:

```
sudo systemctl reload sshd || sudo service ssh reload
```

Step 4: Validate Root Access

From your local environment:

```
ssh -i /path/to/private_key root@<server_ip>
```

A successful connection should yield a root shell immediately.

Provider Behavior Matrix

Provider	Root Access Default	Notes
AWS EC2 (Ubuntu)	Blocked	Requires `authorized_keys` fix
AWS EC2 (Amazon Linux)	Blocked	Requires `authorized_keys` fix
Google Cloud	Blocked	Requires `authorized_keys` fix
Azure	Blocked	Controlled via `sshd_config`
DigitalOcean	Enabled	No action required
Hetzner	Enabled	No action required

Provider	Root Access Default	Notes
Vultr	Enabled	No action required
Linode (Akamai)	Enabled	No action required

For AWS and GCP, **Step 1 is mandatory** in most cases.

Troubleshooting

Root login still denied

Check for override files:

```
sudo grep -r PermitRootLogin /etc/ssh/sshd_config.d/
```

Ensure all instances are set to:

```
PermitRootLogin prohibit-password
```

Forced “login as ubuntu” message

Indicates the provider-injected restriction is still present.

Re-run Step 1 and re-validate `/root/.ssh/authorized_keys`.

SSH key not accepted

Ensure the key exists and permissions are correct:

```
sudo cat /root/.ssh/authorized_keys
```

If missing:

```
sudo cp ~/.ssh/authorized_keys /root/.ssh/authorized_keys
sudo chmod 600 /root/.ssh/authorized_keys
sudo chown root:root /root/.ssh/authorized_keys
```

Consolidated Execution Block

```
# Remove provider-enforced root restriction
sudo sed -e "s/echo;sleep 10;exit 142//g" -i /root/.ssh/authorized_keys

# Enable root login via SSH key only
sudo sed -i 's/^.*PermitRootLogin.*/PermitRootLogin prohibit-password/' /etc/ssh/sshd_config

# Normalize any drop-in overrides
sudo sed -i 's/^.*PermitRootLogin.*/PermitRootLogin prohibit-password/'
/etc/ssh/sshd_config.d/*.conf 2>/dev/null || true

# Reload SSH daemon
sudo systemctl reload sshd || sudo service ssh reload
```

Revision #2
Created 2026-04-06 13:34:53 UTC by Amit Shukla
Updated 2026-04-06 15:37:12 UTC by Amit Shukla