We have set up hundreds of Linux servers for businesses across Kenya. Banks in Upper Hill E-commerce platforms in Westlands. Logistics companies with offices in Mombasa and Nairobi. Every single time, we run through the same checklist before we hand the server over.

CloudSpinx is a Linux infrastructure and managed services provider based in Nairobi. We deploy, harden and manage production Linux servers for over 40 businesses across East Africa, running everything from simple web apps to complex microservice architectures.

This is that checklist. Every command. Every config change. Copy it, bookmark it, use it. If you are inheriting a server from another provider or spinning up something new on AWS, Azure or any local hosting provider, this is where you start.

Picking a Distro

We default to Ubuntu 24.04 LTS ↗ for most clients. Fight us.

Here is why:

- Five years of security updates without paying anyone
- Largest package repository of any Linux distro
- Every tutorial, Stack Overflow answer and vendor doc assumes Ubuntu
- Our junior engineers can hit the ground running on day one

When do we use something else? Rocky Linux 9 ↗ gets picked when the client needs RHEL binary compatibility, usually because they run commercial software that only certifies again RHEL. Debian gets picked when we want maximum stability and the client does not care about having the latest packages.

We stopped recommending CentOS after Red Hat killed CentOS 8. That decision burned a lo of people in Kenya who had built their entire stack on it. If you are still running CentOS 7, its end-of-life was June 2024. You need to migrate now. Talk to our Linux infrastructure team if you need help with that.

What about Arch, Fedora, openSUSE for production? No.

The First 15 Minutes on a Fresh Server

Here is exactly what we run when we first SSH into a new box. This assumes Ubuntu 24.04, but the concepts apply everywhere.

Update everything first

```
sudo apt update && sudo apt upgrade -y
sudo reboot
```

Boring. Essential. The number of servers we inherit that are 200+ days behind on security patches is alarming. Not "mildly concerning." Alarming.

Timezone and locale

```
sudo timedatectl set-timezone Africa/Nairobi
sudo localectl set-locale LANG=en_US.UTF-8
```

Africa/Nairobi is UTC+3 with no daylight saving changes. If your cron jobs are running at weird times, check this first. We have debugged more "mysterious timing issues" that turne out to be a server set to UTC than we would like to admit.

Create your admin user and get off root

```
sudo adduser deployer
sudo usermod -aG sudo deployer
```

Now set up SSH key access for this user. From your **local machine**, copy your public key to the new user:

```
# Run this from your laptop, not the server
```

```
ssh-copy-id -i ~/.ssh/id_ed25519.pub deployer@your-server-ip
```

If `ssh-copy-id` is not available, do it manually on the server:

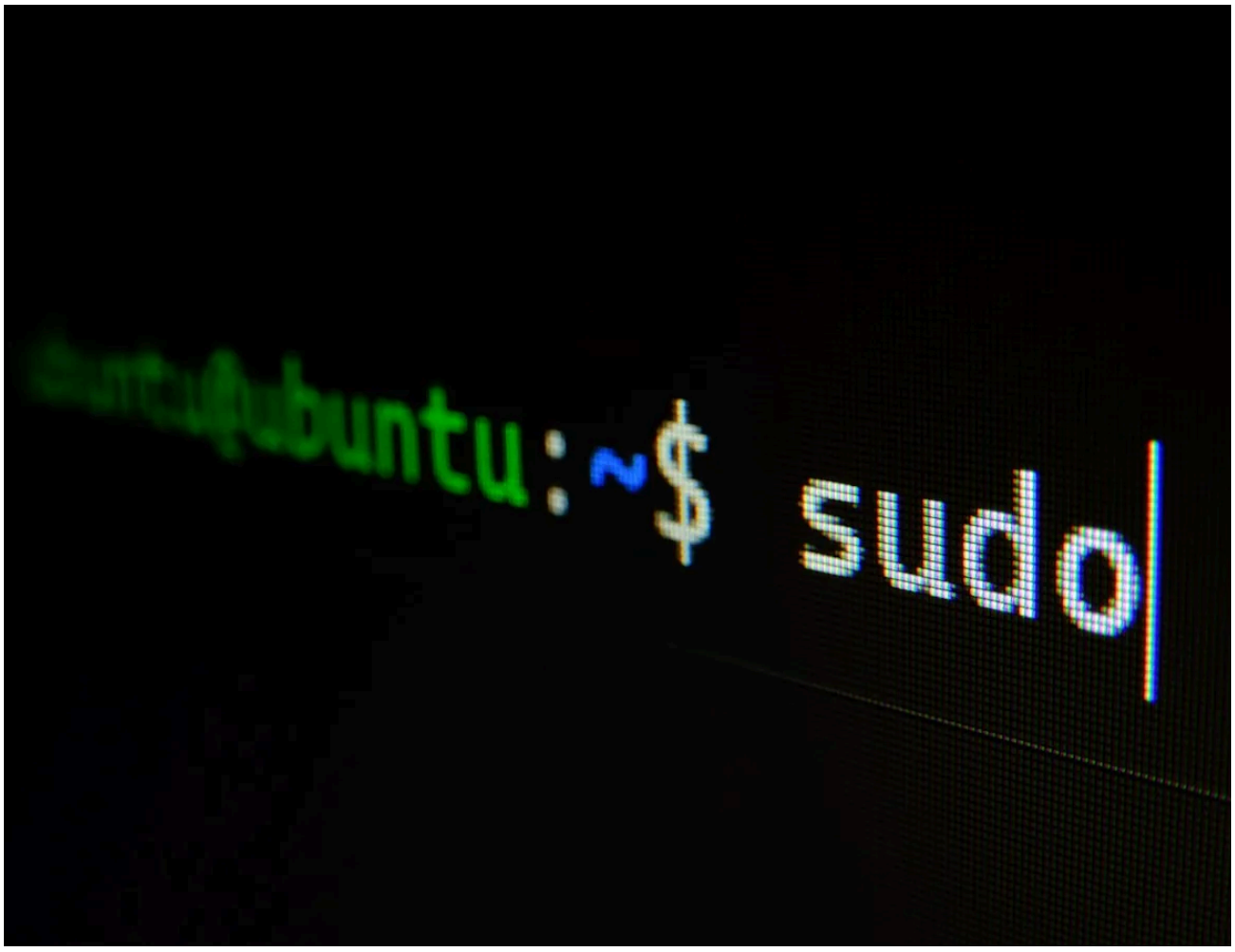
```
sudo mkdir -p /home/deployer/.ssh
sudo cp ~/.ssh/authorized_keys /home/deployer/.ssh/
sudo chown -R deployer:deployer /home/deployer/.ssh
sudo chmod 700 /home/deployer/.ssh
sudo chmod 600 /home/deployer/.ssh/authorized_keys
```

Before moving on, test that you can SSH in as the new user with your key. Open a second terminal and run:

```
ssh deployer@your-server-ip
```

If that works, you are safe to proceed. If it does not, fix it now while you still have your current session open. Do not skip this step.

Never run applications as root. We inherited a server last year from a Mombasa logistics company where everything ran as root. The web app, the database, the cron jobs. One compromised PHP script and an attacker would have had complete system access. It took u a full day to untangle that mess.



SSH Hardening

Only do this after you have confirmed SSH key access works for your new user (the step above). If you disable password authentication before your key is set up, you will lock yourself out.

Edit `/etc/ssh/sshd_config`:

```
Port 2222
PermitRootLogin no
PasswordAuthentication no
PubkeyAuthentication yes
MaxAuthTries 3
ClientAliveInterval 300
ClientAliveCountMax 2
AllowUsers deployer
```

```
sudo systemctl restart sshd
```

Changing the default port is not real security, we know that. But it cuts automated scanning noise by about 95%, which makes your logs actually readable. The real protection is key-on authentication with root login disabled.

Do not close your current session yet. Open a new terminal and test with the new port:

```
ssh -p 2222 deployer@your-server-ip
```

If that works, you are done. If not, your original session is still open and you can fix the config. We have locked ourselves out exactly once. Once was enough.

Firewall Configuration

Ubuntu with ufw

```
sudo ufw default deny incoming
sudo ufw default allow outgoing
sudo ufw allow 2222/tcp comment 'SSH'
sudo ufw allow 80/tcp comment 'HTTP'
sudo ufw allow 443/tcp comment 'HTTPS'
sudo ufw enable
sudo ufw status verbose
```

Rocky Linux with firewalld

```
sudo firewall-cmd --permanent --add-port=2222/tcp
sudo firewall-cmd --permanent --add-service=http
sudo firewall-cmd --permanent --add-service=https
sudo firewall-cmd --permanent --remove-service=ssh
sudo firewall-cmd --reload
```

Only open what you need. We audit servers regularly where ports 3306 (MySQL), 5432 (PostgreSQL) and 6379 (Redis) are wide open to the internet. That is how data breaches happen. If your app server needs to talk to your database, use private networking or SSH tunnels. Our cybersecurity team covers this in depth during infrastructure audits.

Fail2ban

```
sudo apt install fail2ban -y
```

Create `/etc/fail2ban/jail.local`:

```
[DEFAULT]
bantime = 3600
findtime = 600
maxretry = 3
banaction = ufw

[sshd]
enabled = true
port = 2222
filter = sshd
logpath = /var/log/auth.log
maxretry = 3
bantime = 86400
```

```
sudo systemctl enable fail2ban
sudo systemctl start fail2ban
```

Fail2ban ↗ watches your logs and bans IPs that show malicious patterns. On a typical Kenya server with a public IP, we see 500 to 2,000 brute force SSH attempts per day. Fail2ban handles this quietly in the background.

Check who is getting banned:

```
sudo fail2ban-client status sshd
```

Automatic Security Updates

You cannot rely on someone remembering to patch servers. We configure unattended-upgrades on every Ubuntu server:

```
sudo apt install unattended-upgrades -y
sudo dpkg-reconfigure -plow unattended-upgrades
```

Edit `/etc/apt/apt.conf.d/50unattended-upgrades`:

```
Unattended-Upgrade::Allowed-Origins {
    "${distro_id}:${distro_codename}-security";
};
Unattended-Upgrade::Automatic-Reboot "true";
Unattended-Upgrade::Automatic-Reboot-Time "03:00";
Unattended-Upgrade::Mail "alerts@yourcompany.co.ke";
```

We set the reboot time to 3am EAT. Most Kenyan businesses have their lowest traffic between 2am and 5am. For Rocky Linux, the equivalent is `dnf-automatic`:

```
sudo dnf install dnf-automatic -y
sudo systemctl enable dnf-automatic-install.timer
```

Monitoring Setup

A server without monitoring is a server you will find out about from your customers. That phone call at 7am on Monday morning: "The system is down." Nobody wants that call.

We use Prometheus ↗ with node_exporter on every server we manage through our managed IT services.

Install node_exporter

```
wget https://github.com/prometheus/node_exporter/releases/download/v1.7.0/node
tar xzf node_exporter-1.7.0.linux-amd64.tar.gz
sudo mv node_exporter-1.7.0.linux-amd64/node_exporter /usr/local/bin/
sudo useradd --no-create-home --shell /bin/false node_exporter
```

Create `/etc/systemd/system/node_exporter.service`:

```
[Unit]
Description=Prometheus Node Exporter
```

```
After=network.target

[Service]
User=node_exporter
ExecStart=/usr/local/bin/node_exporter
Restart=on-failure
RestartSec=5

[Install]
WantedBy=multi-user.target
```

```
sudo systemctl daemon-reload
sudo systemctl enable node_exporter
sudo systemctl start node_exporter
```

If Prometheus feels like overkill for a single server, start simpler. A bash script that checks disk space, memory and CPU, then sends an alert to a Slack webhook or an SMS via the Africa's Talking API ↗. We have clients who started with a 20-line script and graduated to fu Prometheus plus Grafana as they grew.

What to monitor at minimum

- Disk usage (alert at 80%, panic at 90%)
- Memory usage and swap activity
- CPU load average over 5 and 15 minutes
- SSH login attempts, both successful and failed
- SSL certificate expiry dates
- Service status: is nginx/postgres/your-app actually running

Log Management

Default journald settings on Ubuntu will eventually eat your disk. Configure `/etc/systemd/journald.conf`:

```
[Journal]
SystemMaxUse=500M
MaxRetentionSec=30day
Compress=yes
```

```
sudo systemctl restart systemd-journald
```

For application logs, make sure logrotate is configured. Most packages set this up automatically, but custom apps often do not. We have found 40GB log files on servers wher nobody set up rotation. On a server with a 50GB disk, that turns into a very bad afternoon.

If you need centralized logging across multiple servers, we ship logs to a central Grafana Loki ↗ instance. Our cloud infrastructure team can set this up as part of a managed monitoring stack.

Backup Strategy

No backups, no sympathy. That is the rule in our team.

At minimum:

```
# Database backup - PostgreSQL example
pg_dump -U postgres mydb | gzip > /backups/mydb-$(date +%Y%m%d).sql.gz

# Sync to remote storage
aws s3 sync /backups/ s3://your-backup-bucket/$(hostname)/
```

Put this in a cron job:

```
# /etc/cron.d/backups
0 2 * * * deployer /opt/scripts/backup.sh >> /var/log/backups.log 2>&1
```

But here is the part most people skip: test your restores. A backup you have never restored from is a backup that might not work. We run restore tests monthly for every client on our

managed hosting plans.

For businesses handling sensitive data, the Kenya Data Protection Act requires documented data backup and recovery procedures. This is not optional.

Performance Baseline

Before handing the server to developers, record what "normal" looks like:

```
free -h
nproc
cat /proc/cpuinfo | grep "model name" | head -1

# Disk performance
sudo hdparm -Tt /dev/sda

# Current resource usage
vmstat 1 5
iostat -x 1 5
```

Save this output. When someone says "the server is slow" three months from now, you will want to compare against the baseline.

The 15-Minute Audit for Inherited Servers

When a client moves to CloudSpinx from another provider, we audit first. Here is the quick version:

```
# Who can log in?
cat /etc/passwd | grep -v nologin | grep -v /bin/false

# What is listening on the network?
sudo ss -tlnp

# When was the last update?
stat /var/cache/apt/pkgcache.bin
```

```
# Any unexpected cron jobs?
for user in $(cut -f1 -d: /etc/passwd); do crontab -l -u $user 2>/dev/null; d

# Check for rootkits
sudo apt install rkhunter -y && sudo rkhunter --check --sk

# Disk usage by directory
du -sh /* 2>/dev/null | sort -rh | head -20

# Failed login attempts
sudo journalctl -u sshd | grep "Failed" | tail -20

# Automatic updates configured?
systemctl status unattended-upgrades 2>/dev/null || echo "No automatic update
```

This takes about 15 minutes and gives us a clear picture. Most of the time, we find at least three things that need immediate attention. Old user accounts never removed. Services exposed to the internet that should not be. Backups that stopped running six months ago ar nobody noticed.

Kernel Tuning for Web Servers

If you are running a busy web application, these sysctl tweaks make a measurable difference:

```
# /etc/sysctl.d/99-performance.conf
net.core.somaxconn = 65535
net.ipv4.tcp_max_syn_backlog = 65535
net.ipv4.ip_local_port_range = 1024 65535
net.ipv4.tcp_tw_reuse = 1
vm.swappiness = 10
fs.file-max = 2097152
```

```
sudo sysctl -p /etc/sysctl.d/99-performance.conf
```

The `swappiness = 10` setting is important. Default is 60, which means Linux starts swappir to disk way too aggressively. On a server with sufficient RAM, you want to stay out of swap as much as possible. Disk I/O in cloud environments, especially on lower-tier instances, is orders of magnitude slower than RAM.

When to Call for Help

This checklist covers the essentials. It is what every production server should have before going live. But there are scenarios where you need more:

- PCI DSS compliance for payment processing (M-Pesa integrations, card payments)
- Multi-server architectures with load balancing
- Container orchestration with Kubernetes
- Automated CI/CD pipelines for deployment
- CIS Benchmark ↗ Level 2 hardening

That is where our Linux infrastructure team earns its keep. We handle the complex configurations so your developers focus on building the product, not wrestling with server configs at midnight.

Reach out at hello@cloudspinx.co.ke or WhatsApp 0713 403 044.

Frequently Asked Questions

Which Linux distro is best for production servers in Kenya?

Ubuntu 24.04 LTS for most use cases. Five years of free security updates, the largest community, and every hosting provider in Kenya supports it out of the box. Rocky Linux 9 if you need RHEL compatibility for commercial software certification.

How much does managed Linux server support cost in Kenya?

It depends on the number of servers, the complexity of your stack, and what level of suppo you need. We offer plans that cover monitoring, patching, backups and incident response. Contact our team at hello@cloudspinx.co.ke for a quote specific to your environment.

Should we use cloud servers or dedicated hardware?

Cloud for most businesses. The flexibility to scale during peak times saves money compare to paying for idle capacity. Dedicated servers make sense for consistent, predictable workloads that need guaranteed performance, like database servers processing thousands of queries per second.

How often should we patch our Linux servers?

Security patches should be applied automatically via unattended-upgrades. Major version upgrades need planning and staging environment testing first. We schedule a monthly maintenance window for non-critical updates and apply security patches within 24 hours of release.

Is Ubuntu secure enough for financial applications?

Yes, when properly hardened. Ubuntu with CIS Benchmark hardening, fail2ban, proper firewall rules, and regular patching is used by banks and fintech companies worldwide. The distro choice matters less than the hardening applied to it.

What monitoring tool should a small team start with?

Prometheus with node_exporter if you have more than two servers. For a single server, a bash script that checks disk, memory, and service status, then alerts via email or SMS, is perfectly fine. Graduate to Prometheus and Grafana as your infrastructure grows.

How do we comply with Kenya's Data Protection Act for server hosting?

Document your data backup and recovery procedures. Encrypt sensitive data at rest and in transit. Maintain access logs showing who accessed what data and when. Conduct regular security audits. Our cybersecurity team runs compliance assessments tailored to Kenyan regulations.

Can we manage Linux servers ourselves or should we outsource?

If you have a dedicated sysadmin with genuine Linux experience, in-house works. If your developers are splitting time between code and server maintenance, outsourcing to a managed provider is almost always more cost-effective. Developer time is better spent writing features than chasing down why the disk filled up at 2am.

Read next

All articles →

EMAIL 9 min

Why Your Business Email Still Runs on Free Gmail (and Why That Is Costing You)

Most Kenyan SMEs send invoices and tender responses from personal Gmail...

Amina Hassan 23 March 2026

DEVOPS 11 min

Your Developers Are Deploying Code Over SSH: A CI/CD Wake-Up Call for Kenyan Tech Companies

Most Kenyan dev teams still deploy by SSHing into production. We break down...

Josphat Mutai 23 March 2026

NETWORKING 11 min

Office Network Disasters We Have Fixed in Nairobi (and How to Avoid

Them)

Seven real network disasters from Nairobi offices - switching loops, rogue DHCP...

Peter Ochieng 23 March 2026