THE ASH COLE TRADING SERIES

BOOK 5 — COMPANION 3

VPS Setup Guide

Deploying the AI bot on a Linux virtual private server

Ash Cole

This guide lives outside the book on purpose. Hosting prices, provider capacity, and the specifics of broker bridges change faster than the publishing cycle, and a guide that is baked into the book will be out of date the day it ships. This companion is reviewed every six months and the latest version is always available at agenticedgelab.ai.

Last reviewed: April 2026.

Why Linux, not Windows

Book 5 Chapter 11 specifies a Linux VPS with systemd as the process supervisor, and this companion follows the same architecture. The reason is not ideology — it is reliability. A Linux VPS with systemd restarts a crashed process within ten seconds, survives unattended reboots without user interaction, runs the bot as a background service with no login shell attached, and costs less than the equivalent Windows VPS because it does not carry a Windows license fee. For a bot that must run twenty-four hours a day, five days a week, unattended, for months at a time, the Linux stack is the correct choice.

If you already have a MetaTrader 5 workflow built around a Windows VPS and you are not willing to migrate, the core ideas in this guide still apply — every step has a Windows equivalent — but the exact commands and file paths will differ. The Windows-equivalent commands are noted inline in the step list below for readers who need them.

Minimum and Recommended Specifications

Resource	Minimum	Recommended	Notes
vCPU	1 core	2 cores	Bot loop is CPU-light; training is the heavy operation
RAM	2 GB	4 GB	Model fits in memory; 4 GB leaves headroom for pandas
Disk	20 GB SSD	40 GB SSD	Data, logs, model candidates, journal
Bandwidth	1 TB / month	Unmetered	MT5 is quiet; log shipping is the usage
OS	Ubuntu 22.04 LTS	Ubuntu 24.04 LTS	Both carry long support windows
Region	Near broker server	Near broker server	Latency matters more than distance to home

Latency to the broker matters more than latency to your home. If your broker's trade server is in London, you want a VPS in London or Amsterdam, not in your own country. For most retail forex brokers this means Europe (LD4, AM3) or US East Coast (NY4) — the two

data center clusters that host the majority of retail MT5 server infrastructure.

Recommended Providers

The providers below were reviewed in April 2026 and are the five most commonly used by retail traders running this kind of stack. Pricing and features change; verify before purchasing, and do not treat this list as an endorsement that will stay accurate forever.

Provider	Positioning	Typical price	Best for
FXVM	Forex-specific, latency-optimized	$20–35 / month	Traders on London or NY4 broker servers
Beeks	Forex-specific, enterprise-grade	$40–80 / month	Traders who want the institutional tier
ForexVPS	Forex-specific, budget	$15–30 / month	Smaller accounts, low-complexity bots
Contabo	General-purpose, very cheap	$7–15 / month	Development, demo, and testing
Hetzner	General-purpose, strong EU presence	$5–12 / month	European traders, professional setups

For a first deployment, Contabo or Hetzner on a 2 vCPU / 4 GB plan is the most cost-effective option, and both provide systemd-ready Ubuntu images out of the box. Upgrade to FXVM or Beeks later if broker latency becomes the bottleneck — but verify with the Chapter 9

walk-forward that latency is actually affecting fills before paying the premium.

Step-by-Step Install

The fourteen steps below take a fresh Ubuntu VPS to a running, systemd-supervised, dashboard-monitored bot in approximately ninety minutes for a first-time setup and approximately twenty minutes once you have done it once. Every command can be copied verbatim. Every step produces a verification output you should eyeball before moving to the next.

Where a step differs on Windows, the Windows equivalent is noted inline in parentheses; the core ideas are identical across both platforms.

- 1. Provision a 2 vCPU / 4 GB Ubuntu 24.04 VPS in the region closest to your broker's trade server.
- 2. SSH in as root, create a non-root user named trader with sudo rights, and disable root password login. (Windows: equivalent is creating a non-admin Windows user and disabling RDP for Administrator.)

```
adduser trader && usermod -aG sudo trader
sed -i 's/^#\?PermitRootLogin.*/PermitRootLogin no/'
/etc/ssh/sshd_config
systemctl restart ssh
```

- 3. Update the system, add the deadsnakes PPA — Ubuntu's default repos do not carry Python 3.11 — and install the essentials. (Windows: Windows Update + Python 3.11 64-bit installer.)

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

```
sudo add-apt-repository -y ppa:deadsnakes/ppa &&
sudo apt update
sudo apt install -y python3.11 python3.11-venv
python3-pip git ufw
```

- 4. Enable the firewall — allow SSH, dashboard port, and nothing else. (Windows: Windows Defender Firewall inbound rules for RDP + port 5000.)

```
sudo ufw allow 22/tcp && sudo ufw allow 5000/tcp &&
sudo ufw enable
```

- 5. Put the book5_code package you built following Companion 1 onto the VPS at /home/trader/book5_code (upload it with scp or rsync, or pull it from your own private git remote). (Windows: copy your built package into C:\book5_code.)

```
scp -r book5_code trader@<vps-
ip>:/home/trader/book5_code
```

- 6. Create a Python virtual environment and install the requirements. (Windows: py -3.11 -m venv venv plus .\\venv\\Scripts\\activate.)

```
cd /home/trader/book5_code
python3.11 -m venv venv && source venv/bin/activate
pip install -r requirements.txt
```

- 7. Install the MetaTrader 5 bridge. On Linux the MT5 terminal runs under Wine — follow MetaQuotes' official Linux/Wine instructions to install it. The MetaTrader5 Python package ships Windows-only wheels, so it will not pip-install into the Linux venv: install a Windows build of Python 3.11 inside the same Wine prefix, pip-install the MetaTrader5 wheel there, and either run the bot from that Wine-side Python or keep the Linux venv and bridge the two with a small

RPC helper such as the community mt5linux package. Verify the bridge with a one-line connection test before moving on. (Windows: native MetaTrader5 install, no Wine required.)

- 8. Edit /home/trader/book5_code/config.yaml with your broker account number, server name, symbols, risk percentage, and magic number. Do not edit the features section — those parameters are set by the training run, not by hand. (Windows: identical step, path C:\\book5_code\\config.yaml.)
- 9. Run the preflight checklist from Chapter 8 — the one that verifies config validity, model loadability, MT5 connection, symbol info, DST-safe clock output, and log directory permissions. Every check must return yes before proceeding. (Windows: identical, run from the activated venv.)

```
python preflight.py --config config.yaml
```

- 10. Install the systemd unit file from /book5_code/ch11_deployment/ai_bot.service into /etc/systemd/system/ and enable it. (Windows: NSSM service wrapper or Task Scheduler entry with "run whether user is logged on or not".)

```
sudo cp ch11_deployment/ai_bot.service
/etc/systemd/system/
sudo systemctl daemon-reload
sudo systemctl enable ai_bot && sudo systemctl start
ai_bot
```

- 11. Install the dashboard as a second systemd service, enable it, and verify that http://<vps-ip>:5000 returns the traffic-light page from a browser on your laptop. If you prefer not to

expose port 5000 publicly, skip its ufw rule in step 4 and reach the dashboard through an SSH tunnel instead — ssh -L 5000:localhost:5000 trader@<vps-ip>, then browse http://localhost:5000. (Windows: NSSM wraps the Flask app the same way.)

```
sudo systemctl enable ai_dashboard && sudo systemctl start ai_dashboard
```

- 12. Install the alert queue worker as a third systemd service. Configure the Telegram bot token, Twilio SID, and SMTP credentials in /home/trader/book5_code/alerts.env (600 permissions, owned by trader). (Windows: store credentials in a restricted-ACL alerts.env file.)
- 13. Install the heartbeat monitor as a fourth systemd service — note that it runs as a separate process from the bot, as Chapter 11 specifies, so that a hung bot cannot silence its own heartbeat check. (Windows: fourth NSSM-wrapped service.)
- 14. Run in demo mode for four weeks and at least one hundred closed trades before switching the config to a live account, and verify the demo Sharpe is within 30% of the walk-forward combined Sharpe before flipping the switch. Both conditions must hold. (Windows: identical gate.)

When all fourteen steps are complete, you have a supervised bot, a dashboard, an alert pipeline, and a heartbeat monitor — all running under systemd, all restarting automatically on crash, all surviving unattended reboots. That is the full Chapter 11 stack.

The systemd Unit File

This is the exact unit file the package ships. It is fifteen lines and it is the single most important operational file in the entire deployment.

```
[Unit]
Description=Ash Cole Book 5 AI Trading Bot
After=network-online.target
Wants=network-online.target

[Service]
Type=simple
User=trader
WorkingDirectory=/home/trader/book5_code
ExecStart=/home/trader/book5_code/venv/bin/python
ch08_bot/ai_bot.py --config config.yaml
Restart=always
RestartSec=10
StandardOutput=append:/home/trader/book5_code/logs/s
tdout.log
StandardError=append:/home/trader/book5_code/logs/st
derr.log

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

Three things to notice. First, Restart=always with RestartSec=10 means any crash is automatically recovered within ten seconds, no manual intervention required. Second, StandardOutput and StandardError are appended to files inside the bot's own log directory, so every print statement and every traceback ends up somewhere greppable without needing journalctl. Third, the service depends on network-online.target,

which means systemd will not launch the bot until the VPS has a working network connection on boot — eliminating the most common source of spurious startup failures. One adjustment: if your MetaTrader5 import lives in the Wine-side Python from step 7, point ExecStart at that interpreter — or at your RPC bridge launcher — instead of the venv path shown here.

Top Five Setup Failures

MT5 connects in development but not on the VPS

The usual cause is that the broker's server hostname has not been added to the MT5 client on the VPS, or the VPS firewall is blocking the MT5 outbound port (typically 443 for modern brokers, sometimes 1950 or 1951 for legacy ones). Fix: open the MT5 client in the VPS session, add the broker server manually via File → Login → add new broker, and confirm ufw allows the outbound port. The bot's preflight checklist will catch this before the first trade if you run it at every deploy.

Python cannot import MetaTrader5

On Linux this almost always means the Wine-based bridge has not been installed or has installed the wrong architecture. Fix: follow MetaQuotes' official Wine installation steps exactly, verify the 64-bit MetaTrader5 wheel installed into the Wine-side Python with pip list, and run the one-line connection test before moving to the next step. If you see a 32-bit error on a 64-bit system, the Wine prefix or its Python is the wrong architecture — rebuild the prefix as 64-bit, reinstall the 64-bit Windows Python, and repeat.

The bot runs but no orders are placed

The usual cause is either a magic-number collision with another EA on the same account, or a symbol-name

mismatch between the config (EURUSD) and the broker's actual symbol in Market Watch (EURUSD.m, EURUSD+, EURUSDecn). Fix: open MT5, check the exact symbol name, and update config.yaml to match. The preflight checklist prints every configured symbol's broker-side name at startup and refuses to start if any symbol is missing.

DST misfire — the bot trades the wrong session

The cause is that the VPS clock is set to local time instead of UTC, which causes the killzone scheduler to return the wrong session during the week of a DST switch. Fix: set the VPS timezone to UTC with timedatectl set-timezone UTC, reboot, and verify with date that the output is in UTC. The dst_safe_clock.py module from Chapter 5 handles the broker's local time correctly only if the VPS clock itself is in UTC as the baseline.

The bot does not restart after a VPS reboot

The cause is that the systemd service was not enabled at install time, only started. Started means running now; enabled means running at boot. Fix: sudo systemctl enable ai_bot (and the same for ai_dashboard, ai_alerts, ai_heartbeat). Verify with systemctl is-enabled ai_bot which should return enabled. The Chapter 11 operational discipline is that you reboot the VPS deliberately, once a month, to prove the boot

sequence still works — an untested boot sequence is the same as an untested kill switch.

Maintenance Cadence

Running a deployed bot has a rhythm. The rhythm below is the one from Chapter 11 of the book, reproduced here as a single page you can pin next to the dashboard.

Cadence	Task
Daily (1 min)	Glance at dashboard, confirm heartbeat fresh, scan for critical alerts
Weekly (15 min)	Read digest emails, grep warnings, confirm feature stability, compare trade count to trailing 4-week average
Monthly (30 min)	Test the kill switch on demo, test a deliberate VPS reboot, review the decay journal
Quarterly (2 hours)	Run the retraining job, evaluate against the stopping rule, record the result in Companion 2 before the shadow period begins, start the 30-day shadow A/B if the candidate passes
Every 6 months	Review this guide for updated provider pricing; review the 3-tier alert thresholds; rotate SSH keys

Most of these checks will be uneventful most of the time. The value of doing them on a schedule is not what they catch but what they confirm: that the bot is behaving the way it is supposed to, and that the trader has eyes on it even when nothing is wrong.

When to Upgrade

The default 2 vCPU / 4 GB Ubuntu VPS described in this guide will run the Book 5 bot across a single symbol comfortably, and across three or four symbols with acceptable headroom. Upgrade conditions are specific and measurable: (1) if the bot's bar-processing time climbs above one second consistently, the CPU is the bottleneck and you need more cores; (2) if free memory drops below 25% with the bot running, pandas is the bottleneck and you need more RAM; (3) if the disk usage graph exceeds 80%, log rotation or retention is the issue, not the size of the disk; (4) if latency to broker fills measurably exceeds your backtest assumptions, you need a VPS physically closer to the broker's trade server, not a bigger VPS in the same place.

Do not upgrade on intuition. Upgrade on measurement. Every one of the four conditions above is visible on the Chapter 11 dashboard if it is configured correctly, and none of them require guesswork.

A VPS that runs the bot quietly for months is the most boring piece of hardware a trader will ever own. Boring is the goal.