部署 • 运维 • Agent 工作流

私有网络中的 AI 中转站搭建

软件依赖安装、中转链路搭建、验收与 Matt Pocock skills 驱动的实施方法

洪煜 • 微信：hongyuatbj

依据 docs/final-spec.md · 2 核 4 GB 单台服务器 · WireGuard 私网

00 · WHAT & WHY

我们要搭建什么，为什么要搭建？

先用一句话说清楚

搭建一座只在私有网络里开放的 AI 中转站

同事先连接 WireGuard，再通过一个统一地址调用 OpenAI 等模型。供应商密钥只保存在服务器上，不必分发给每个人。

私网连接　统一调用地址　密钥集中保管

为什么值得搭建

01 更安全

AI 接口、管理界面和数据库不直接暴露在公网。

02 更好管理

每个人使用自己的调用密钥，可以单独停用和限制用量。

03 更省心

模型地址和供应商密钥集中配置，客户端不必逐台修改。

04 可恢复

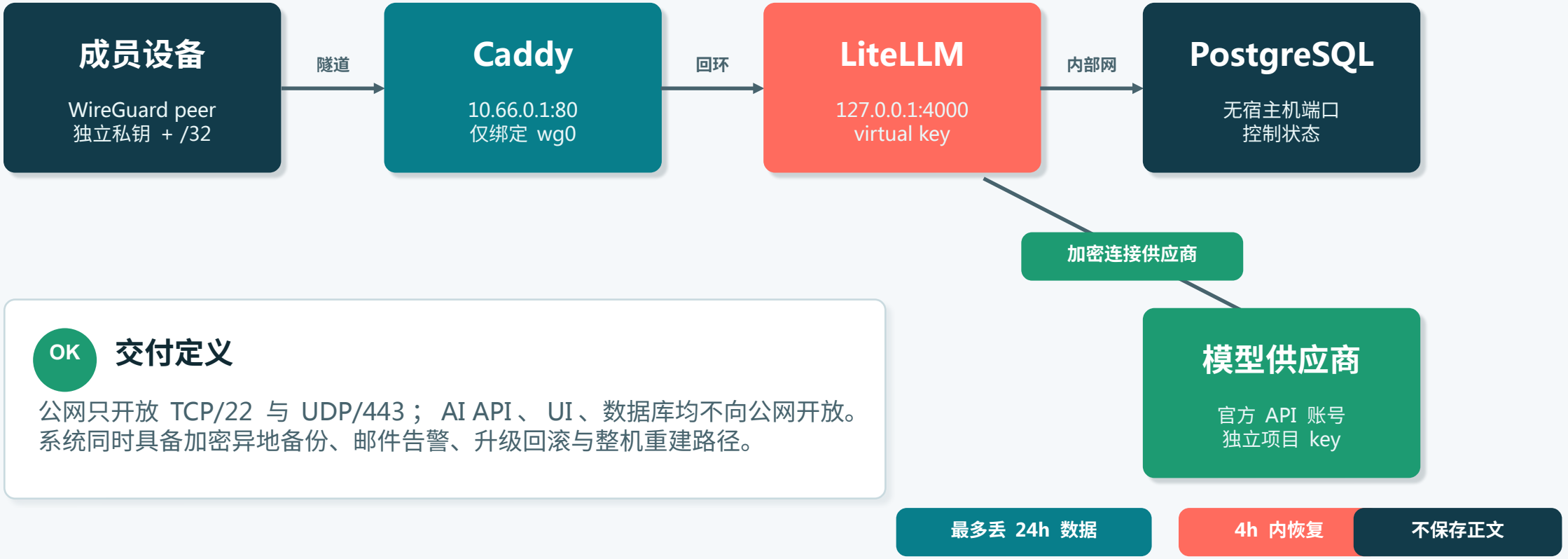
有异地备份、邮件告警和重建说明，故障后可以照步骤恢复。

使用者最终只需要两样东西：自己的 WireGuard 配置，以及自己的 AI 调用密钥。

01 · TARGET STATE

我们最终要交付什么？

一条只有两层身份都通过才可访问的中转链路



OK **交付定义**

公网只开放 TCP/22 与 UDP/443； AI API、 UI、数据库均不向公网开放。
系统同时具备加密异地备份、邮件告警、升级回滚与整机重建路径。

02 · PREREQUISITES

开工前：六类先决条件必须齐备

01 VPS

2 vCPU / 4 GB / 60 GB SSD
Ubuntu 24.04 · 公网 IPv4

02 恢复路径

root 密钥 SSH
厂商 Interactive Console 已实测

03 模型账号

官方 API 账号、计费、预算告警
少量真实调用预算

04 异地备份

私有 R2 Bucket
最小权限 token + 异地 restic 密码

05 外部监控

Healthchecks.io 两个 Check
管理员邮箱已验证

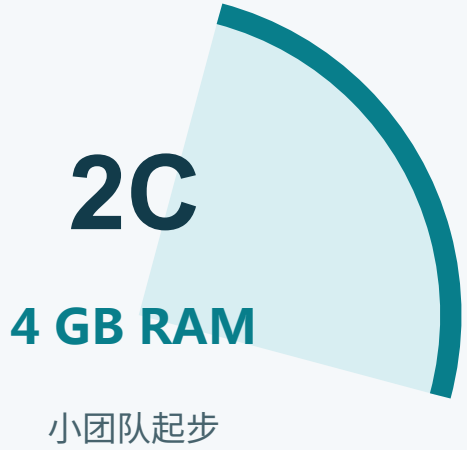
06 管理员设备

SSH / WireGuard / Git
密码管理器与仓库权限

域名不是先决条件： HTTP 只在 WireGuard 隧道内； R2 不需要关联网站或开启公开访问。

03 · CAPACITY

为什么 2 核 4 GB 通常够用



C CPU 做什么

检查身份、转换请求格式、持续转发流式回复；模型推理由供应商完成。

M 内存更关键

LiteLLM 约 800 MiB，PostgreSQL 约 55 MiB；还需留系统与峰值余量。

+ 何时升到 4 核

CPU 长时间满载、请求开始排队、最慢一批请求明显变慢，或启用了更多插件。

D 磁盘基线

60 GB 可以起步；日志、备份临时文件和 Docker 镜像都要限制大小。

够不够用，要看首字出现时间、最慢一批请求和剩余内存，不能只看 CPU 核数。

04 · DEPENDENCY MAP

软件依赖：宿主机与容器各司其职

UBUNTU HOST

W WireGuard
设备准入 · UDP/443

C Caddy
wg0 HTTP 入口

N nftables
默认拒绝 · 精确放行

S systemd/restic
心跳 · 备份 · 恢复

DOCKER COMPOSE

LiteLLM
API · 路由 · 治理
固定 digest

内部网络 →

PostgreSQL
保存配置和用量
使用持久数据卷

= 为什么混合部署
网络和恢复入口放在宿主机，应用及数据库交给 Compose 固定版本、隔离网络并重复部署。

05 · HOST INSTALL

宿主机依赖：按恢复边界安装

UBUNTU 24.04 · 示例命令

```
apt update
apt install -y wireguard caddy nftables restic \
  curl ca-certificates gnupg jq
```

! **禁止动作**

不要执行 nft flush ruleset 。 Docker 自己维护 NAT/filter 表；全部清空会让容器看似 healthy ，却无法连接模型供应商。

实际部署优先运行仓库 deployment/*/deploy.sh ，而不是复制演示命令逐行拼装。

1 **先盘点**

记录 OS 、端口、服务、磁盘；验证 SSH 与厂商控制台。

2 **安装但不立即切流**

先放置配置、校验语法，再准备定时回滚。

3 **保持两条恢复路径**

网络 /SSH 变更期间保留旧会话和 Interactive Console 。

4 **从外部验收**

不要只看 systemctl ；用真实 peer 与非 peer 验证边界。

06 · CONTAINER RUNTIME

Docker CE ：只从官方 Ubuntu 仓库安装

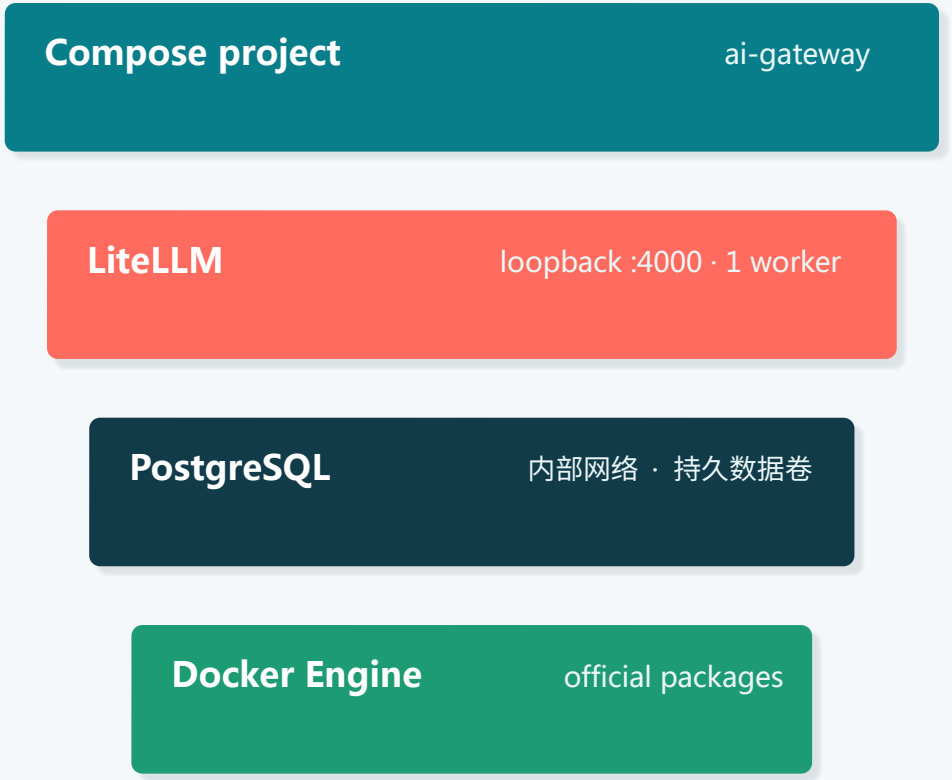
RECOMMENDED

```
# 仓库已提供幂等安装脚本
sudo deployment/docker-ce/install.sh

# 验证
docker version
docker compose version
```

**固定镜像**

禁止 latest。生产 Compose 同时固定 tag 与 sha256 digest，升级前必须验证备份与回滚。



容器不拥有公网入口；网络入口始终由宿主机控制。

07 · PRIVATE ENTRY

中转站步骤 1 ：建立 WireGuard 私网入口

客户端生成 key	服务端分配 /32	监听 UDP/443	Caddy 绑定 wg0	负向验收
私钥不离开设备	一设备一 peer	PersistentKeepalive 25	10.66.0.1:80	撤销 peer + 公网扫描

Y 通过

peer 可访问 /health/private-entry ；管理员 peer 不受测试撤销影响。

N 拒绝

非 peer 无法访问 AI ；公网 TCP/80 、 443 、 4000 、 5432 均不可达。

R 保底

公网 root key SSH 与厂商控制台继续可用。

08 · PRODUCTION GATEWAY

中转站步骤 2 ：部署 LiteLLM + PostgreSQL

DEPLOY & HEALTH

```
docker compose \
  --project-directory /opt/ai-gateway/production-gateway \
  up -d

curl -f http://127.0.0.1:4000/health/liveliness
```

DB **PostgreSQL**

持久数据卷
不开放主机端口
仅内部网络

AI **LiteLLM**

单 worker
loopback :4000
正文日志关闭

1 **创建 root-only secrets**

Postgres 、 master/salt 、独立 UI 凭据；目录 0700 、文件 0600 。

2 **经 Admin UI 添加官方模型**

连接管理员 peer 后访问 http://10.66.0.1/ui/ ， API key 不进 Git 。

3 **签发 virtual key**

一人 / 一应用一 key ；模型 allowlist 、预算、 RPM/TPM 、到期。

4 **真实调用与持久化**

批准后测试普通回复和逐字流式回复；重建容器后配置仍要保留。

09 · SECURITY BOUNDARIES

中转站步骤 3 ：秘密、身份与日志边界

10 · RECOVERY

中转站步骤 4 ：备份、恢复与外部心跳

7 保留策略

保留 7 份每日备份、4 份每周备份，以及每次重要变更前的备份。

Q 季度恢复演练

恢复到空数据库，再用临时 LiteLLM 检查密钥权限、预算、模型和审计记录。

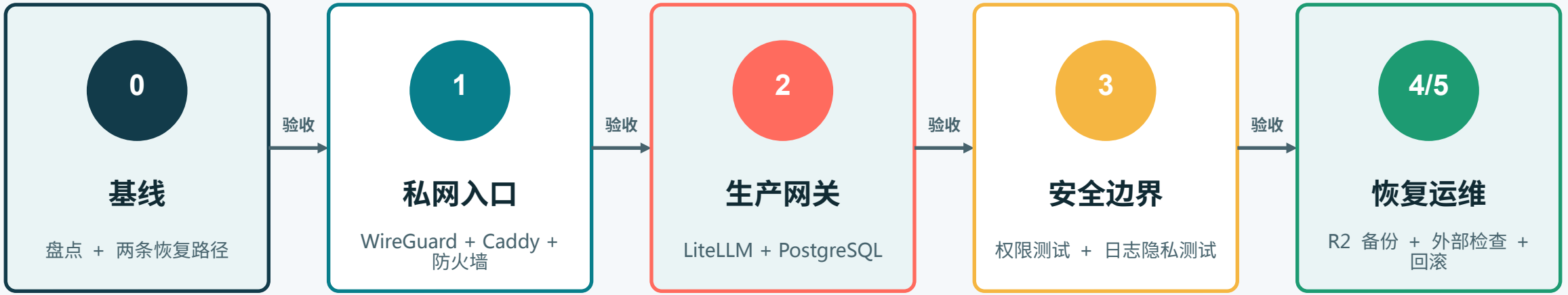
H 两个外部检查

核心服务每 5 分钟检查一次；备份每天检查一次。失败和恢复都发邮件。

最多丢 24h 数据　**4h 内恢复**　**密码异地保存**

11 · EXECUTION GATES

中转站步骤 5 ：五阶段交付，阶段间设门



A **Agent 可以直接做**

查看现状、根据模板生成配置、检查语法、运行测试、记录不含密码的结果。

H **先征得管理员同意**

产生费用的调用、撤销真实账号、故意停止核心服务、修改云防火墙、升级生产系统。

12 · ACCEPTANCE

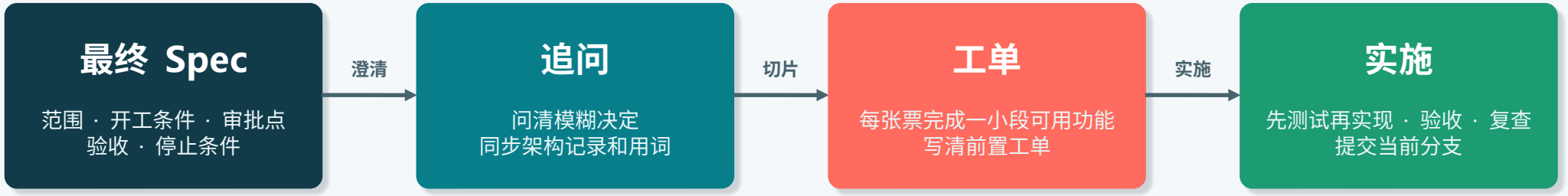
容器显示正常，还不算验收完成

✓	**网络**	已授权设备可访问；未接入私网的设备不可达； UDP/443 能建立连接
✓	**鉴权**	合法调用密钥返回 200 ；无效或撤销密钥返回 401 ；普通设备访问管理页返回 403
✓	**权限**	模型范围、预算、有效期和每分钟次数限制都能实际拒绝越权请求
✓	**隐私**	日志里找不到对话正文、回复、身份头和供应商密钥
✓	**恢复**	R2 备份可以完整读回；空数据库恢复后能从接口核对状态

验收记录只写状态、时间、模型名称和非秘密指标；不要复制调用密钥、对话正文或回复。

13 · SPEC-DRIVEN DELIVERY

如何把 final-spec.md 交给 Agent 使用



S Spec 规定边界

Agent 还要查看仓库、现状记录、架构决定和操作手册；发现服务器情况不符就停止。

T 工单要能单独验收

每张票完成一小段可用功能，并写明依赖哪些前置工单。

G 审批仍由人负责

涉及费用、撤销权限、故意制造故障和升级生产系统时， Agent 必须先询问管理员。

14 · SKILL INSTALLATION

安装 Matt Pocock 工程技能

GLOBAL INSTALL · ACCURATE PACKAGE NAMES

```
# 需要 Node.js / npm； npx 会运行 Skills CLI
npx skills add mattpocock/skills@setup-matt-pocock-skills -g -y
npx skills add mattpocock/skills@grilling -g -y
npx skills add mattpocock/skills@domain-modeling -g -y
npx skills add mattpocock/skills@grill-with-docs -g -y
npx skills add mattpocock/skills@to-tickets -g -y
npx skills add mattpocock/skills@tdd -g -y
npx skills add mattpocock/skills@code-review -g -y
npx skills add mattpocock/skills@implement -g -y
```

! **注意名称**

官方名称是 to-tickets （复数），不是 to-ticket 。

1 **首次配置**

进入目标 repo 后运行 setup-matt-pocock-skills ，选择 GitHub Issues 或本地 Markdown 。

U **更新**

npx skills update
技能来源： mattpocock/skills

15 · SHARPEN THE SPEC

先 setup ，再 grill-with-docs

RUN ONCE PER REPO

```
/setup-matt-pocock-skills

选择：GitHub Issues
领域文档：单 context
CONTEXT.md + docs/adr/
```

可直接复制的提示词

```
/grill-with-docs docs/final-spec.md

请重点追问：
- VPS 与网络条件能否实际检查
- 哪些操作必须由管理员批准
- 改坏某个服务后，能否从别的路径恢复
- 每项验收能否从系统外部完成
```

1

连续追问

先不动服务器，把含糊、矛盾和没有决定的地方问清楚。

2

统一用词

明确“设备”“使用者”“调用密钥”分别指什么，避免同词不同义。

3

记录架构决定

把重要选择和理由写进 ADR，例如为什么 HTTP 只在 WireGuard 内使用。

4

何时结束追问

Spec 已能回答：谁批准、怎样验收、出错后怎样恢复。

16 · PLAN THE WORK

to-tickets ：把大任务拆成可验收的小工单

可直接复制的提示词

```
/to-tickets docs/final-spec.md

请准备发布到已配置的 GitHub Issues。
先展示工单拆分和前后依赖，
等我确认大小合适后再创建工单。
```

T

每张票必须写清

交付什么可用结果、怎样验收、依赖哪些工单。工作量应控制在一次 Agent 会话内。

推荐的中转站工单图

先让负责人确认工单大小和前后顺序，再发布；不要擅自关闭原始需求。

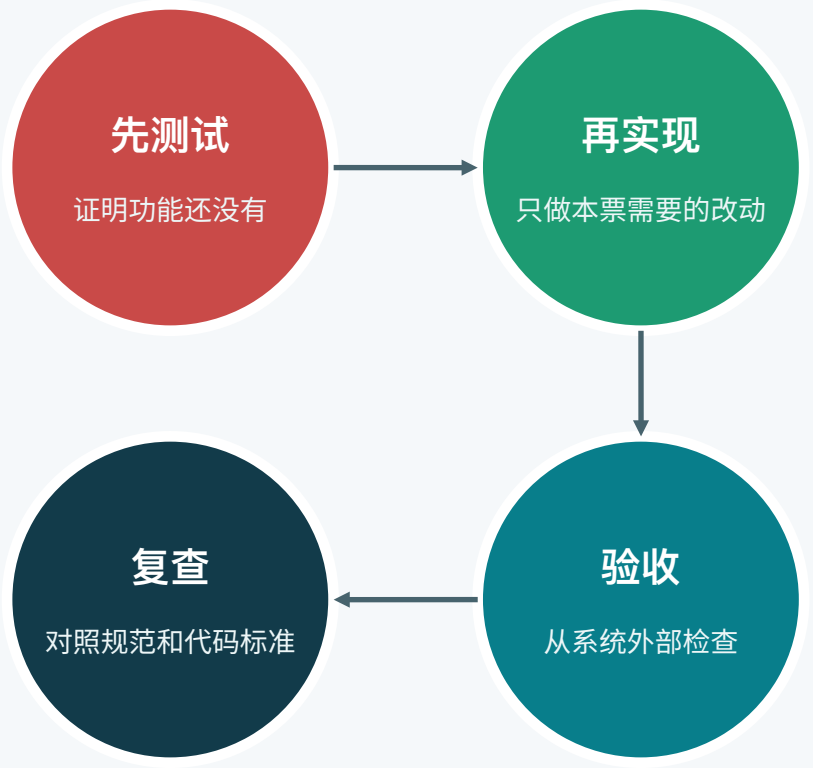
17 · BUILD & VERIFY

implement ：按依赖顺序逐张完成工单

可直接复制的提示词

```
/implement #2

遵循 docs/final-spec.md 和架构决定。
能先写测试的地方，先写失败测试再实现；
修改网络时保留 SSH、厂商控制台和定时回滚；
最后运行全部验收，做代码审查并提交当前分支。
```



安全停止条件

环境与 inventory 冲突、控制台不可用、秘密出现在输出、镜像无法验证、数据库迁移无恢复路径——立即停止。

改完代码后，还要跑完整测试、复查改动、提交代码，并留下不含密码的验收记录。

18 · HANDOFF

给同事的最短可执行路径

1. **准备账号** — VPS 、模型供应商、 R2 、 Healthchecks
2. **克隆仓库** — 先读项目总结和最终 Spec
3. **安装技能** — 仓库设置 → 追问 Spec → 拆工单
4. **逐票实施** — 先测试再实现；遇到审批点就暂停询问
5. **恢复验收** — 从外部检查、恢复到空数据库、确认邮件告警

怎样才算做完

私网可用
公网不可达
身份可撤销
数据可恢复
故障会告警

只看到容器启动，还不够。

先把条件和责任问清楚，再动服务器。

每完成一张工单，都要能说明系统多了什么可用能力，以及如何确认它真的可用。

P 项目入口

docs/project-summary.md
docs/final-spec.md
docs/runbooks/

S 技能来源

github.com/mattpocock/skills
skills.sh/mattpocock/skills

4 四个动作

设置仓库 · 追问规范
拆分工单 · 按票实施

输出：可编辑 PPTX + PDF