Tom Curie
tomcurie.fr

DOCUMENT D'ARCHITECTURE TECHNIQUE

Infrastructure homelab

Services auto-hébergés sur Proxmox : DNS filtré, reverse proxy HTTPS, accès distant chiffré et supervision.

AUTEUR	Tom Curie
VERSION	1.0
DATE	1er novembre 2026
PORTFOLIO	tomcurie.fr
DIFFUSION	Publique — données anonymisées

Proxmox VE · Debian 12 · Pi-hole · Nginx Proxy Manager · Tailscale · Docker Compose · Homarr · Uptime Kuma · step-ca

Sommaire

01 Contexte et objectifs

02 Vue d'ensemble

03 Plan d'adressage et inventaire

04 Services

05 Flux principaux

06 Décisions d'architecture

07 Sécurité

08 Exploitation et déploiement

09 Retours d'expérience, limites et évolutions

01

Contexte et objectifs

Ce homelab transforme un LAN domestique « tout sur la box » en une petite infrastructure de services auto-hébergés : DNS filtré, reverse proxy HTTPS, accès distant chiffré et portail d'applications. Il sert de terrain d'expérimentation pour des pratiques d'entreprise (séparation des rôles, nommage, supervision, PKI) à l'échelle d'une maison.

Objectifs

- Filtrer les publicités et le tracking pour tous les équipements du LAN, sans configuration poste par poste.
- Publier chaque service sous un nom lisible en `*.lan`, en HTTPS, derrière un point d'entrée unique.
- Accéder à l'ensemble du LAN depuis l'extérieur sans ouvrir de port sur la box.
- Industrialiser la création de VM et le déploiement de services (Debian 12, cloud-init, Docker Compose).
- Superviser les services et le réseau avec des outils 100 % locaux.

Périmètre

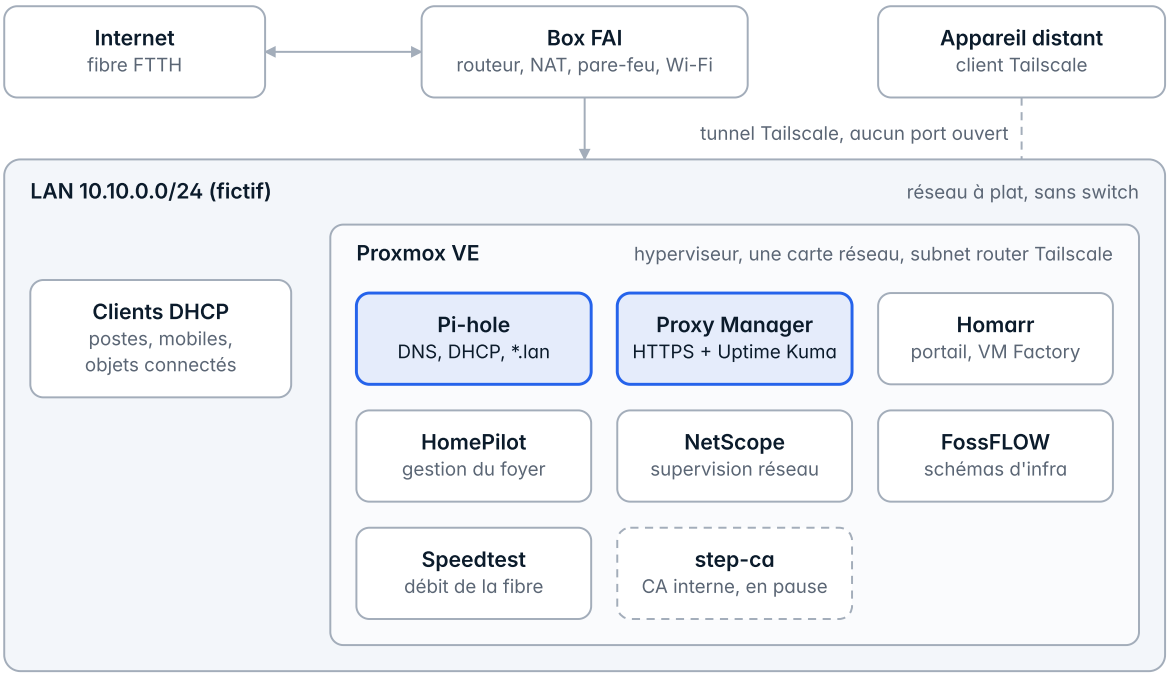
- **Inclus** : réseau local, hyperviseur Proxmox, VM de services, accès distant, outils d'exploitation.
- **Exclus** : objets connectés grand public, postes personnels, projet domotique, pilotage de la box par API.

> Toutes les adresses IP, identifiants et noms de machines de ce document sont **fictifs**. Le plan d'adressage présenté reproduit la logique de l'infrastructure réelle, pas ses valeurs.

02

Vue d'ensemble

Toute l'infrastructure repose sur un seul serveur Proxmox branché sur la box FAI. Deux VM d'infrastructure (Pi-hole, Nginx Proxy Manager) rendent les services utilisables ; les autres VM hébergent les applications.



En couleur : services d'infrastructure. Pointillés : service en pause ou tunnel chiffré.

Figure 1 Un seul serveur Proxmox héberge tous les services, derrière la box FAI

La box ne fait plus que router et filtrer en périphérie ; le DNS, le DHCP et la publication des services sont repris par les VM, et l'accès distant arrive directement sur Proxmox par Tailscale.

03

Plan d'adressage et inventaire

Le LAN tient dans un seul sous-réseau privé /24 (ici `10.10.0.0/24`, fictif), découpé par rôle : infrastructure en bas de plage, VM de services ensuite, clients DHCP dans une plage dédiée. Chaque service a une IP fixe et un nom `*.lan` déclaré dans Pi-hole.

PLAGE (FICTIVE)	USAGE
.1	Passerelle : box FAI
.2 – .9	Services d'infrastructure réseau (DNS/DHCP, reverse proxy)
.10 – .19	Hyperviseurs
.20 – .49	VM de services, IP fixes
.100 – .199	Clients DHCP (postes, mobiles, objets connectés)

COMPOSANT	IP (FICTIVE)	NOM	HÉBERGEMENT	RÔLE	ÉTAT
Box FAI	10.10.0.1	—	Matériel FAI	Routeur, NAT, pare-feu, Wi-Fi 6E, DHCP désactivé	En service
Pi-hole	10.10.0.2	`pihole.lan`	VM Debian 12	DNS filtrant, DHCP du LAN, zone locale `*.lan`	En service
Nginx Proxy Manager	10.10.0.3	`npm.lan`	VM Debian 12 + Docker	Reverse proxy HTTPS, wildcard `*.lan` ; Uptime Kuma	En service
Proxmox VE	10.10.0.10	`proxmox.lan`	Serveur physique	Hyperviseur, subnet router Tailscale	En service
HomePilot	10.10.0.20	`homepilot.lan`	VM	Gestion du foyer (Flask)	En service
NetScope	10.10.0.21	`netscope.lan`	VM + Docker	Supervision et diagnostic réseau	En développement
Homarr + VM Factory	10.10.0.22	`portail.lan`	VM + Docker	Portail d'applications, création de VM	En service
step-ca	10.10.0.23	`step-ca.lan`	VM	Autorité de certification « LAN CA »	En pause
FossFLOW	10.10.0.24	`fossflow.lan`	VM + Docker	Schémas d'infrastructure isométriques	En service
Speedtest Tracker	10.10.0.25	`speedtest.lan`	VM + Docker	Mesure périodique du débit fibre	En cours

04

Services

Chaque brique a un rôle unique et tourne dans sa propre VM, ce qui permet de redémarrer, sauvegarder ou remplacer un service sans toucher aux autres.

4.1 Connectivité — box FAI

- Fibre FTTH, box Wi-Fi 6E (8 Gbit/s descendant, 1 Gbit/s montant).
- Rôles conservés : routage, NAT, pare-feu périmétrique, Wi-Fi.
- Rôle retiré : serveur DHCP IPv4, désactivé au profit de Pi-hole.
- Topologie physique à plat : pas de switch, tous les équipements filaires sont branchés directement sur la box.

4.2 DNS et DHCP — Pi-hole

- Résolveur DNS de tout le LAN, avec listes de blocage publicité et tracking.
- Serveur DHCP IPv4 : distribue l'adresse, la passerelle (box) et **lui-même comme DNS**.
- Zone locale `*.lan` : un enregistrement par service, pointant vers le reverse proxy ou directement vers la VM.
- Serveur DNS des clients distants via le split DNS Tailscale.

4.3 Reverse proxy — Nginx Proxy Manager

- Point d'entrée HTTP/HTTPS unique des applications web.
- Un « proxy host » par service : `homepilot.lan`, `netscope.lan`, `portail.lan`, etc.
- Certificat wildcard `*.lan` auto-signé, appliqué à tous les hôtes.
- Administré par interface web, donc délégable à une personne non spécialiste.

4.4 Accès distant — Tailscale

- Le nœud Proxmox joue le rôle de *subnet router* et annonce le /24 du LAN au tailnet.
- Split DNS : seules les requêtes du domaine `lan` sont envoyées à Pi-hole ; le reste suit le DNS habituel de l'appareil distant.
- Aucun port ouvert ni redirigé sur la box.

4.5 Portail — Homarr et VM Factory

- **Homarr** : page d'accueil unique qui liste et ouvre toutes les applications du LAN, à la manière d'un portail SSO d'entreprise. Il affiche aussi des widgets (ex. résultats Speedtest Tracker).
- **VM Factory** : application web maison qui encapsule le script de création de VM et appelle l'API Proxmox avec un jeton dédié à privilèges limités.

4.6 Supervision

- **Uptime Kuma** : sondes de disponibilité des services du LAN.

- **NetScope** : plateforme maison de découverte, santé réseau, flux et diagnostic, 100 % locale (React/TypeScript, FastAPI, PostgreSQL, Docker Compose), développée en 6 phases.
- **Speedtest Tracker** : historique du débit de la fibre.

4.7 Applications et outillage

- **HomePilot** : gestion du foyer (charges, maintenance, tâches, énergie, coffre de documents), Flask + Gunicorn + Nginx + SQLite.
- **FossFLOW** : réalisation des schémas d'infrastructure isométriques.
- **step-ca** : autorité de certification interne, déployée et validée puis mise en pause (voir D7).

05

Flux principaux

L'accès à un service suit le même chemin en trois étapes, que l'utilisateur soit chez lui ou à l'extérieur.

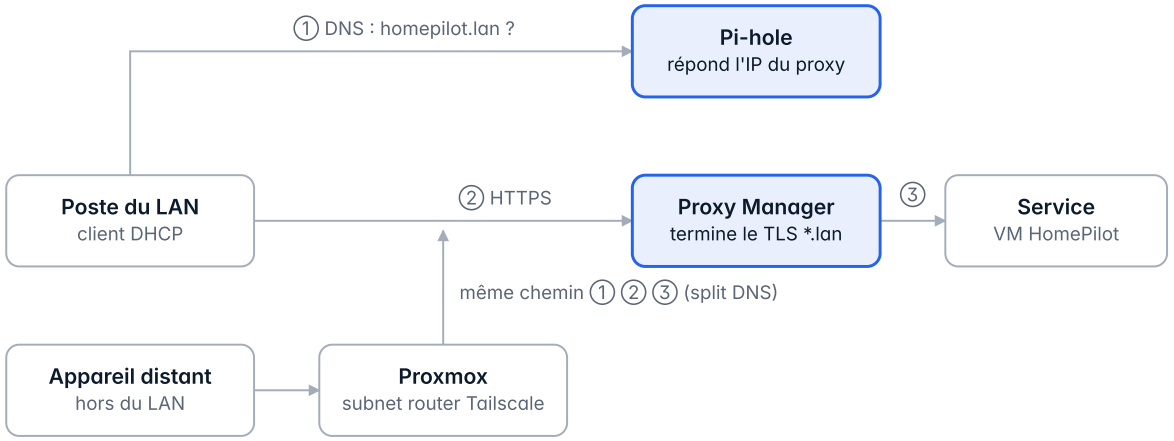


Figure 2 Un nom *.lan passe toujours par Pi-hole puis par le proxy, en local comme à distance

1. **Résolution DNS** : le client demande `homepilot.lan` à Pi-hole, qui répond avec l'IP du reverse proxy. Les autres noms sont résolus vers Internet après filtrage.
2. **HTTPS** : le navigateur se connecte au proxy, qui présente le certificat wildcard `*.lan`.
3. **Relais** : le proxy transmet la requête à la VM du service, selon le nom demandé.

À distance, Tailscale annonce la route du LAN et envoie les requêtes DNS du domaine `lan` à Pi-hole : l'appareil suit ensuite exactement les mêmes étapes, sans aucun port ouvert sur la box.

06

Décisions d'architecture

Les choix ont été guidés par trois contraintes : une box FAI peu paramétrable, un matériel minimal (un serveur, une carte réseau, pas de switch) et la volonté de garder une infra administrable sans ligne de commande au quotidien.

#	DÉCISION	POURQUOI	ALTERNATIVE ÉCARTÉE
D1	**Pi-hole devient aussi serveur DHCP du LAN**	La box n'autorise pas à changer le DNS distribué (option DHCP 6 verrouillée sur elle-même). Seul le serveur DHCP peut imposer Pi-hole comme DNS à tous les clients.	DNS réglé poste par poste : non tenable pour les mobiles et les objets connectés.
D2	**Renumérotation du LAN hors de la plage par défaut de la box**	Sortir de la plage `192.168.1.0/24` partagée par la plupart des box grand public, pour éviter les conflits de routes quand on accède au LAN via Tailscale depuis un autre réseau. Migration faite en une fois : box, Pi-hole, reverse proxy, Proxmox, VM, routes Tailscale.	Garder la plage par défaut : simple, mais collisions probables en itinérance.
D3	**Nginx Proxy Manager à la place de Traefik**	Traefik a été déployé en premier. NPM l'a remplacé sur la même VM pour que la gestion du reverse proxy puisse être déléguée à une personne qui préfère une interface graphique.	Traefik : configuration déclarative et découverte automatique, mais administration par fichiers et labels.
D4	**Accès distant par Tailscale en subnet router**	Aucun port à ouvrir sur la box, chiffrement WireGuard de bout en bout, un seul nœud à maintenir. Le split DNS limite à `*.lan` ce qui passe par Pi-hole.	VPN WireGuard ou OpenVPN auto-hébergé : impose une redirection de port et l'exposition d'un service sur Internet.
D5	**Une VM Debian 12 par service, applications en Docker Compose**	Isolation des pannes et des mises à jour, sauvegarde et restauration par VM, déploiement reproductible à partir d'un template cloud-init.	Un seul hôte Docker pour tout : plus économe, mais un incident touche tous les services.
D6	**Pare-feu pfSense mis en pause**	Une seule carte réseau sur le serveur et aucun switch : impossible de placer proprement un pare-feu entre la box et le LAN sans matériel supplémentaire. pfSense avait été préféré à OPNsense.	Pare-feu virtuel sur une seule interface : complexité et risque de coupure sans bénéfice réel.
D7	**CA interne step-ca mise en pause, retour au certificat auto-signé de NPM**	La CA « LAN CA » a été déployée et son certificat importé dans NPM avec succès. Elle est mise de côté en attendant de la coupler à un SSO (Authentik envisagé).	Garder la CA en production : impose de distribuer la racine sur chaque appareil dès maintenant.
D8	**Homarr comme portail plutôt qu'une page sur mesure**	Outil existant, maintenu, avec ajout d'applications et widgets en quelques clics.	Page HTML maison : entièrement contrôlée, mais à maintenir soi-même.
D9	**Supervision : Uptime Kuma + NetScope, Beszel abandonné**	Besoin apparu après une panne DNS/DHCP de Pi-hole. Kuma couvre la disponibilité ; NetScope couvre la vue réseau. Beszel n'apportait pas assez pour une VM de plus.	Stack Prometheus/Grafana complète : surdimensionnée pour la taille du LAN.

07

Sécurité

Aucun service du homelab n'est exposé sur Internet : la seule porte d'entrée distante est le tailnet, authentifié et chiffré.

MESURE	MISE EN ŒUVRE
Surface d'exposition nulle	Aucune redirection de port ni DMZ sur la box ; accès distant uniquement via Tailscale.
Filtrage DNS	Tous les clients DHCP résolvent via Pi-hole (blocage publicité et tracking).
Chiffrement des accès web	HTTPS sur tous les services via NPM (wildcard `*.lan`).
Moindre privilège	VM Factory utilise un jeton API Proxmox dédié, pas le compte root.
Isolation	Une VM par service : la compromission d'une application ne donne pas la main sur les autres.
Secrets	Mots de passe et jetons dans des fichiers `.env` en `chmod 600`, jamais dans les images ni le code.

Risques connus

- Le certificat auto-signé génère des alertes navigateur et habitue à les ignorer ; la CA interne (D7) est la réponse prévue.
- Réseau à plat : objets connectés, postes et serveurs partagent le même segment, faute de switch et de pare-feu interne (D6).
- Le DNS IPv6 annoncé par la box peut contourner Pi-hole (voir section 9).

08

Exploitation et déploiement

Un nouveau service passe en production en cinq gestes, toujours dans le même ordre :

1. **Créer la VM** depuis VM Factory (ou le script `new-debian.sh`) : clone d'un template Debian 12, configuration cloud-init (utilisateur, clé SSH, IP fixe).
2. **Déployer l'application** en Docker Compose dans la VM.
3. **Déclarer le nom** `service.lan` dans la zone locale de Pi-hole.
4. **Publier le service** dans NPM : proxy host vers la VM, certificat wildcard `*.lan`.
5. **L'exposer aux utilisateurs** : tuile dans Homarr, sonde dans Uptime Kuma.

Conventions

- **Nommage** : un service = une VM = un nom `*.lan`.
- **Adressage** : IP fixe par VM dans la plage des services, hors de la plage DHCP.
- **Configuration** : fichiers `docker-compose.yml` + `.env`, sans secret en clair dans le code.
- **Administration** : SSH sur le LAN ou via Tailscale, jamais depuis Internet.

09

Retours d'expérience, limites et évolutions

Le principal enseignement : chaque service centralisé (DNS, DHCP, proxy) simplifie l'usage mais devient un point unique de défaillance qu'il faut superviser.

SUJET	CONSTAT	RÉPONSE APPORTÉE
Panne de Pi-hole	DNS et DHCP tombent ensemble : plus de résolution ni de nouveaux baux sur tout le LAN.	Mise en place de la supervision (Uptime Kuma, puis NetScope).
Fuite DNS IPv6	La box annonce son propre DNS en IPv6, indépendamment du DHCP IPv4 : les clients peuvent contourner Pi-hole.	DHCPv6 de la box désactivé ; l'annonce DNS restante par Router Advertisement reste à traiter.
Conflit de port	Le port 80 de la VM Pi-hole était occupé par Apache installé par défaut.	Service Apache arrêté et désactivé avant l'installation.
Renumérotation	Changer de sous-réseau touche chaque IP fixe, chaque enregistrement DNS et les routes VPN.	Migration planifiée et exécutée en une passe, routes Tailscale et split DNS mis à jour.

Limites actuelles

- Pi-hole est un point unique de défaillance pour le DNS et le DHCP.
- Réseau à plat, sans segmentation entre serveurs, postes et objets connectés.
- Un seul hyperviseur, une seule carte réseau.
- Certificat auto-signé sur les services web.

Évolutions envisagées

- Supprimer l'annonce du DNS IPv6 par la box, ou annoncer Pi-hole en IPv6.
- Second résolveur Pi-hole pour la redondance DNS.
- Switch manageable + VLAN, puis reprise du projet pfSense (segmentation serveurs / clients / IoT).
- Réactivation de la CA interne et SSO avec Authentik.
- Phases 2 à 6 de NetScope : découverte, santé réseau, flux, alertes, durcissement.