Rapport de clôture — Tor sécurisé v3.1

Projet : tor-securise (rotation automatique des bridges d'entrée obfs4) **Date** : 15 octobre 2026 **Machine** : Kali One (poste c3b) **Statut** : **INSTALLÉ ET EN PRODUCTION** — vérifié 12 OK / 0 problème **Publication** : mémoire centralisée kvm8 `/srv/memoire-projets/infra/tor-securise/publications/`

---

1. Objet du présent rapport

Ce document clôt le chantier « Tor sécurisé » ouvert le 24/09/2026 et corrigé les 14 et 15/10. Il consigne trois choses :

1. **ce qui est réellement installé** sur la machine (mesuré, pas déclaré) ;
2. **la preuve que la correction fonctionne** (rotation réelle en root, bridges morts écartés) ;
3. **les limites** qui subsistent — pour qu'aucun lecteur ne prenne une hypothèse pour un fait.

Il corrige aussi deux entrées de journal devenues fausses : celles qui annonçaient encore « installation Tor v3.1 attend le mot de passe » alors que l'installation a eu lieu.

---

2. Le problème d'origine

La v1 du mécanisme de rotation (installée le 24/09/2026) présentait deux défauts **mesurés** :

Défaut	Mesure
Elle installait des bridges **morts**	`51.222.13.177:80` écrit en production, sondé **FERMÉ** ; `146.57.248.225:22` idem
Elle **ne les retirait jamais**	liste moat brute, prise dans l'ordre de l'API, sans test de vivacité
Elle ne tournait **que quotidiennement**	`OnCalendar=daily`, aucun déclenchement au démarrage

Conséquence : Tor fonctionnait **malgré** sa configuration, parce que 2 bridges sur 4 restaient vivants. Ce n'était pas une panne visible, c'était une dégradation silencieuse — le cas le plus coûteux à diagnostiquer.

Cause racine du défaut « bridges morts » : l'API moat **ne filtre pas** les bridges décédés. Elle renvoie 7 entrées obfs4 fixes, dont les 2 morts, et **seul l'ordre change d'un tirage à l'autre** — la v1 prenant « les N premiers », elle tombait aléatoirement sur les morts. Mesuré : sha256 de l'ensemble trié **identique sur 4 tirages** (`bf72eed4bc7cbfc6`), ordre variable.

---

3. Ce qui est installé — état vérifié

3.1 Binaires et empreintes

Fichier	sha256 (tronqué)	Statut
`/usr/local/bin/tor-securise-rotation`	`7ab312bbde5f59ffee…`	= empreinte v3 attendue (v1 était `b4835135474a10d0…`)
`/usr/local/lib/tor-securise/moat.sh`	`84134dcd391f28737a66…`	= lib v3 attendue

3.2 Unité de service

```
/etc/systemd/system/tor-securise-rotate.service Type=oneshot
ExecStart=/usr/local/bin/tor-securise-rotation -q
Environment=TOR_SECURISE_BRIDGE_COUNT=2 ← 2 bridges visés (choix
de Pierre) Environment=TOR_MIN_ALIVE=2 ← minimum 2 vivants exigés
Environment=TOR_PRUNE_DEAD=1 ← élagage des morts en service
Nice=10 / IOSchedulingClass=idle TimeoutStartSec=420
ProtectHome=read-only / PrivateTmp=true / NoNewPrivileges=true ⚠
AUCUNE directive ProtectSystem= (voir §5, correctif v3.1)
```

Ni `User=` ni `Group=` : systemd lance donc le service **en root** (défaut). C'est un point décisif : `sudoers` **n'entre pas en jeu** quand c'est systemd qui démarre le service — PID 1 est déjà root et n'appelle pas `sudo`. L'automatisme ne dépend donc d'**aucun** droit NOPASSWD.

3.3 Timer

```
/etc/systemd/system/tor-securise-rotate.timer OnBootSec=2min ←
rotation 2 minutes après CHAQUE démarrage OnCalendar=daily ← puis
une fois par jour RandomizedDelaySec=6h ← évite les pics de
charge Persistent=true ← rattrapage si la machine était éteinte
WantedBy=timers.target
```

Mesures : `is-enabled` → **enabled** · `is-active` → **active** · NEXT **Fri 2026-10-16 01:48:31 CEST**.

3.4 Bridges en production

`/etc/tor/torrc.d/50-bridges.conf` — md5 `59e299bcbc2b84862d090d67eea783f3` :

```
Bridge obfs4 209.148.46.65:443
74FAD13168806246602538555B5521A0383A1875 … Bridge obfs4
212.83.43.95:443 BFE712113A72899AD685764B211FACD30FF52C31 …
```

Deux bridges, **tous deux vivants**, générés par le script v3 (`source : moat/live`).

---

4. Preuve que ça fonctionne

4.1 Rotation réelle en root — trois passages, tous réussis

`journalctl -u tor-securise-rotate.service` :

``` 11:09:17 moat: [51.222.13.177:80] liste noire (mort recurrent) — rejete 11:09:17 moat: [146.57.248.225:22] liste noire (mort recurrent) — rejete 11:09:43 Finished … status=0

11:11:42 (les 2 mêmes rejets) 11:12:00 Finished … status=0

11:12:45 (les 2 mêmes rejets) 11:13:18 Finished … status=0 ```

**Lecture** : la v3 rejette **à chaque passage** exactement les deux bridges que la v1 installait. C'est le correctif « filtre vivants » à l'œuvre, en production, pas en simulation.

### 4.2 Vérificateur officiel du livrable

```
bash VERIFIER-apres-installation.sh → BILAN : 12 OK / 0
problème(s) rc=0 [+] bridge mort 51.222.13.177 écarté [+] bridge
mort 146.57.248.225 écarté [+] timer : armed / enabled (survit au
redémarrage) [+] rotation prévue 2 min après chaque démarrage [+]
sortie Tor : {"IsTor":true,"IP":"171.25.193.20"}
```

Intégrité du livrable : `sha256sum -c MANIFESTE.sha256` → **7/7 OK** (contrôle au moment de la rédaction de ce rapport).
```

4.3 Tor opérationnel

Contrôle	Résultat
`tor@default`	active, `NRestarts=0`
Bootstrap	`Bootstrapped 100%`
SOCKS 9050	écoute
Sortie mesurée (2×)	`{"IsTor":true,"IP":"171.25.193.20"}` rc=0
`UseBridges`	1
Sortie	aléatoire (conforme)

---

5. Le correctif v3.1 — la panne qui aurait été invisible

La v3 portait `ProtectSystem=full`. Cette option monte `/etc` en **lecture seule** pour le service. Or la rotation **écrit** dans `/etc/tor/torrc.d/50-bridges.conf` et sauvegarde dans `/etc/tor/bridges-backup/`.

Mesure faite : `systemd-run -p ProtectSystem=full` → `/etc` monté `ro` → toute écriture renvoie **« Read-only file system »**.

Pourquoi c'était grave : la rotation aurait échoué **dans le timer**, alors qu'un test manuel (hors namespacing) aurait réussi. Panne invisible, découverte des semaines plus tard sous forme de « Tor ne démarre plus ». La v3.1 **retire** cette directive et revient au comportement éprouvé de la v1, qui n'avait aucune protection de ce type.

Alternative écartée volontairement : `ProtectSystem=strict` + `ReadWritePaths=/etc/tor/torrc.d /etc/tor/bridges-backup` — plus élégant, mais **un seul chemin oublié** = **rotation morte silencieusement**. Le risque d'échec silencieux a été jugé pire que l'absence de protection.

---

6. Ce qui n'est PAS possible : « 1 bridge en sortie »

Point de protocole, pas de configuration. **Un bridge ne peut pas être un nœud de sortie.**

- Un bridge fournit obfs4 (camouflage) **uniquement pour la première étape**. Son descripteur n'est **pas publié** dans le consensus Tor, donc Tor ne peut pas le sélectionner comme exit.
- Il n'existe **aucune** directive `ExitBridge`.

- Vérifié dans `/etc/tor/torrc` : **aucune** ligne `ExitNodes` / `EntryNodes` / `StrictNodes`.

Sur Onionoo (l'intuition était partiellement juste) : Onionoo publie bien des entrées `type=bridge`, mais leurs adresses sont **caviardées en IP privées** (`10.x.x.x`) — par conception. Onionoo publie en revanche les **relais de sortie** avec adresse publique, fingerprint et flags, et l'API est exploitable (mesuré : 8 relais Exit+Fast, ORPort 8/8 ouverts, latence 0,6 s).

Ce qui serait faisable : remplir `ExitNodes` avec des fingerprints **vivants et vérifiés**, régénéré automatiquement comme les bridges d'entrée — « N relais de sortie choisis », pas « 1 bridge de sortie ». **Le prix, dit franchement** : forcer un petit ensemble de sorties identifiables **réduit** l'anonymat — un adversaire qui observe un exit sait que le trafic vient d'un des N relais de la configuration. C'est mécaniquement plus corrélé que la sortie aléatoire actuelle. **Non installé, non recommandé en l'état.**

Et le réglage « 1 seul bridge en entrée » a été **écarté** : avec un bridge unique, s'il tombe, Tor ne bootstrappe **plus du tout**. Le choix retenu est **2** (`COUNT=2` / `MIN_ALIVE=2`), ce qui donne une redondance minimale sans diluer la maîtrise de l'entrée.

---

7. Décision A vs B — état réel

Deux options étaient posées :

- **A** — conserver le NOPASSWD ciblé sur `tor-securise-rotation` + `tor-securise-verifier` ;
- **B** — s'appuyer uniquement sur le service systemd (lancé en root par PID 1, sans sudoers).

B est démontré en pratique : le timer tourne, le service se lance en root, la rotation de 11:12 a réussi `status=0`, **sans** que les droits NOPASSWD aient servi. L'automatisme est intégralement couvert par B.

Mais : `/etc/sudoers.d/tor-rotation` **est toujours en place** (`root:root`, mtime 14/10 00:56). Son retrait n'a **pas** été effectué. Raison : **un cycle timer complet n'a pas encore été observé** (`NEXT Fri 2026-10-16 01:48:31 CEST`). Retirer le filet avant d'avoir vu un cycle passer serait une imprudence.

Coût de A qu'il faut connaître : A est le seul élément qui permet à un agent de lancer la rotation **en root depuis une session utilisateur** — exactement le vecteur que le registre d'autorisations signale. B ne le supprime pas tout seul : il faut retirer le fichier sudoers.

Procédure de retrait, quand ce sera le moment (jamais avant un cycle réussi) :

```
bash sudo cp -p /etc/sudoers.d/tor-rotation /root/tor-rotation.sudoers.bak-$(date +%Y%m%dT%H%M%S) sudo visudo -c # doit répondre « parsed OK » AVANT toute modification sudo visudo -f /etc/sudoers.d/tor-rotation
```

⚠ **Ne jamais** éditer `/etc/sudoers.d/` avec `>` ou `sed -i` : une erreur de syntaxe **casse sudo pour tout le système**. D'où `visudo` obligatoire.

Conséquence à accepter : après retrait, `sudo /usr/local/bin/tor-securise-rotation` échouera ; seul `systemctl start tor-securise-rotate.service` (avec mot de passe) déclenchera une rotation à la main.

---

8. Correction des entrées de journal périmées

Deux entrées de `ACTIONS-EN-COURS-20260907.md` annonçaient encore une installation **en attente** :

Ligne	Ancien statut	Nouveau statut
705 (15/10 10:40)	`installation Tor v3.1 attend le mot de passe`	`PÉRIMÉ — v3.1 INSTALLÉE`
706 (15/10 10:42–10:48)	`sudo toujours en attente du mot de passe`	`PÉRIMÉ — v3.1 INSTALLÉE 15/10 11:12, rotation réelle status=0`

Méthode : sauvegarde préalable `…md.bak-avant-cloture-tor-20261015T092521Z` (sha256 `118c5ebd…`), puis **édition ciblée** des deux seuls marqueurs de statut — pas de réécriture du fichier. Contrôles : `grep -c "PÉRIMÉ"` → **2** ; `grep -c` des anciens marqueurs → **0**. Taille : 455 424 → 459 480 o (ajout de l'entrée de clôture comprise).

---

9. Limites assumées

1. **L'installation n'est pas de mon fait.** Mes `sudo -n` renvoient `a password is required` (rc=1, mesuré à chaque tour). Les mtime 11:09–11:12 correspondent à une **exécution root humaine** — vraisemblablement Pierre, ou la session DSH parallèle détectée le matin même. Je **constate** l'état, je ne le revendique pas.
2. **La sauvegarde `/root/tor-securise-sauvegarde-*/` n'a pas pu être listée** : `/root` est en `drwx------ root root`, `ls` → `Permission denied`. Son existence est **inférée** de `INSTALLER-v3.1.sh:88` (`NSAVED -ne 5` ⇒

arrêt immédiat), **pas constatée**. Pour la vérifier : `sudo ls -la /root/tor-securise-sauvegarde-*/`

3. **La sonde TCP prouve une socket, pas un handshake obfs4.** Le test dit « le port répond », pas « le bridge fonctionne avec ce certificat ». Un test plus fort exigerait de lire les circuits — or `ControlPort` **n'est pas configuré** (`9051` → `Connection refused`).
4. **Point non élucidé** : `journalctl -u tor@default` sur la fenêtre 11:12+ ne montre **pas** de ligne `Bootstrapped 100%`, alors que le vérificateur la rapporte. Écart relevé, non expliqué.
5. **Un cycle timer complet n'a pas encore été observé.** La prochaine échéance (`Fri 2026-10-16 01:48:31 CEST`) sera la première preuve d'endurance.
6. **Le retrait du NOPASSWD n'est pas fait** (§7).

---

10. Journal des actions de ce tour

- Sauvegarde du journal : `ACTIONS-EN-COURS-20260907.md.bak-avant-cloture-tor-20261015T092521Z`
- Correction des 2 entrées périmées + ajout de l'entrée de clôture
- Rejeu du vérificateur officiel : **12 OK** / **0 problème**, rc=0
- Contrôle du manifeste : **7/7 OK**
- Mesure de sortie Tor : `{"IsTor":true,"IP":"171.25.193.20"}` rc=0
- Dépôt sur kvm8 : `/srv/memoire-projets/infra/tor-securise/publications/`

Aucune écriture dans `/etc/` ce tour. **Aucun secret** dans ce document (contrôle `grep -iE` sur clés, jetons, `password=`, clés privées avant publication).

Rapport rédigé par DSH (agent c3b) — 15 octobre 2026. Les valeurs d'empreintes sont tronquées pour la lisibilité ; les valeurs complètes figurent dans `MANIFESTE.sha256` et `.sha256-complet.txt`.