ODYSSEY LINUX · DOCUMENTATION DE RÉFÉRENCE

Odyssey Control Center — Guide Pratique

Chaque panneau, section par section. Ce qu'il fait, ce qu'il change, et sur quoi cliquer — écrit pour qui ne l'a jamais utilisé.

Launch Edition

Don't trust me. Verify me.

— nobody

Odyssey Linux — Odyssey Control Center : Guide Pratique
Launch Edition. Couvre le Control Center tel qu'il est fourni sur l'ISO de la Launch Edition — dix panneaux complets, de Boot à About, plus un chapitre provisoire pour le module du pilote NVIDIA (visible uniquement sur du matériel NVIDIA, en attente de captures d'écran depuis une machine qui en possède).

Copyright © 2026 the Odyssey Linux project.
Publié sous licence Creative Commons Attribution-ShareAlike 4.0 International (CC BY-SA 4.0).

Ce guide est le complément pratique du Security and Hardening Guide, qui couvre les panneaux Firewall et AppArmor en pleine profondeur technique — vous trouverez ici le guide visuel pas à pas et un renvoi pour le reste.

Chaque écran, bouton et comportement décrit ici a été lu à partir du code source réel du Control Center et de captures d'écran réelles, pas de mémoire.

Les libellés des boutons, les noms des panneaux, les chemins des fichiers et les commandes restent en anglais car ce sont l'interface réelle du programme : les chercher traduits dans le Control Center ne les ferait pas apparaître.

TABLE DES MATIÈRES

1 Qu’est-ce que le Control Center 1
1.1 Motifs que tu retrouveras dans chaque panneau 2
2 Boot 3
2.1 Guide visuel — trouver un autre système d’exploitation et définir le défaut 3
2.2 Guide visuel — choisir ce qui démarre par défaut 5
2.3 Guide visuel — l’onglet EFI 8
2.4 Paramètres du noyau 9
2.5 Advanced 9
3 Kernel 11
3.1 Guide visuel — les préréglages 11
3.2 Guide visuel — SYSCTL, groupe par groupe 12
3.3 Guide visuel — CPU governor 14
3.4 Guide visuel — hugepages 15
3.5 Guide visuel — modules 16
3.6 Advanced 17
4 Memory & Swap 19
4.1 Guide visuel 19
4.2 Advanced 20
5 Firewall 22
5.1 Guide visuel 22
5.2 Advanced 23
6 AppArmor 24
6.1 Guide visuel — STATUS 24
6.2 Guide visuel — PROFILES 25
6.3 Guide visuel — COMMUNITY 26
6.4 Guide visuel — VIOLATIONS 28
6.5 Guide visuel — EDITOR 29
6.6 Advanced 30
7 Privacy 32
7.1 Guide visuel — SearXNG, étape par étape 32
7.2 Guide visuel — Tor, étape par étape 35
7.3 Guide visuel — les combiner 37
7.4 Advanced 38
8 Containers 40
8.1 Guide visuel — le moteur 40
8.2 Guide visuel — rootless vs. rootful, et l’état vide 42
8.3 Guide visuel — le catalogue sélectionné, du début à la fin 42
8.4 Guide visuel — la liste des conteneurs et ses actions 46
8.5 Guide visuel — exécuter un conteneur, et l’ouvrir 47
8.6 Usage disque 48
8.7 Advanced 48
9 Gaming Mode 50
9.1 Guide visuel — l’interrupteur principal 50
9.2 Guide visuel — Packages et Steam 51
9.3 GameMode et MangoHud 53
9.4 Proton 54
9.5 Advanced 54
10 Telemetry 56
10.1 Guide visuel 56
10.2 Advanced 58

11 About 60
11.1 Guide visuel 60
11.2 Copy report 61
11.3 Advanced 61
12 NVIDIA driver 63

CHAPTER 1

Qu’est-ce que le Control Center

L’Odyssey Control Center, ce sont dix petits programmes indépendants derrière un seul lanceur — un par domaine du système : Boot, Kernel, Memory & Swap, Firewall, AppArmor, Privacy, Containers, Gaming Mode, Telemetry, About, et (sur du matériel NVIDIA) le module du pilote NVIDIA.

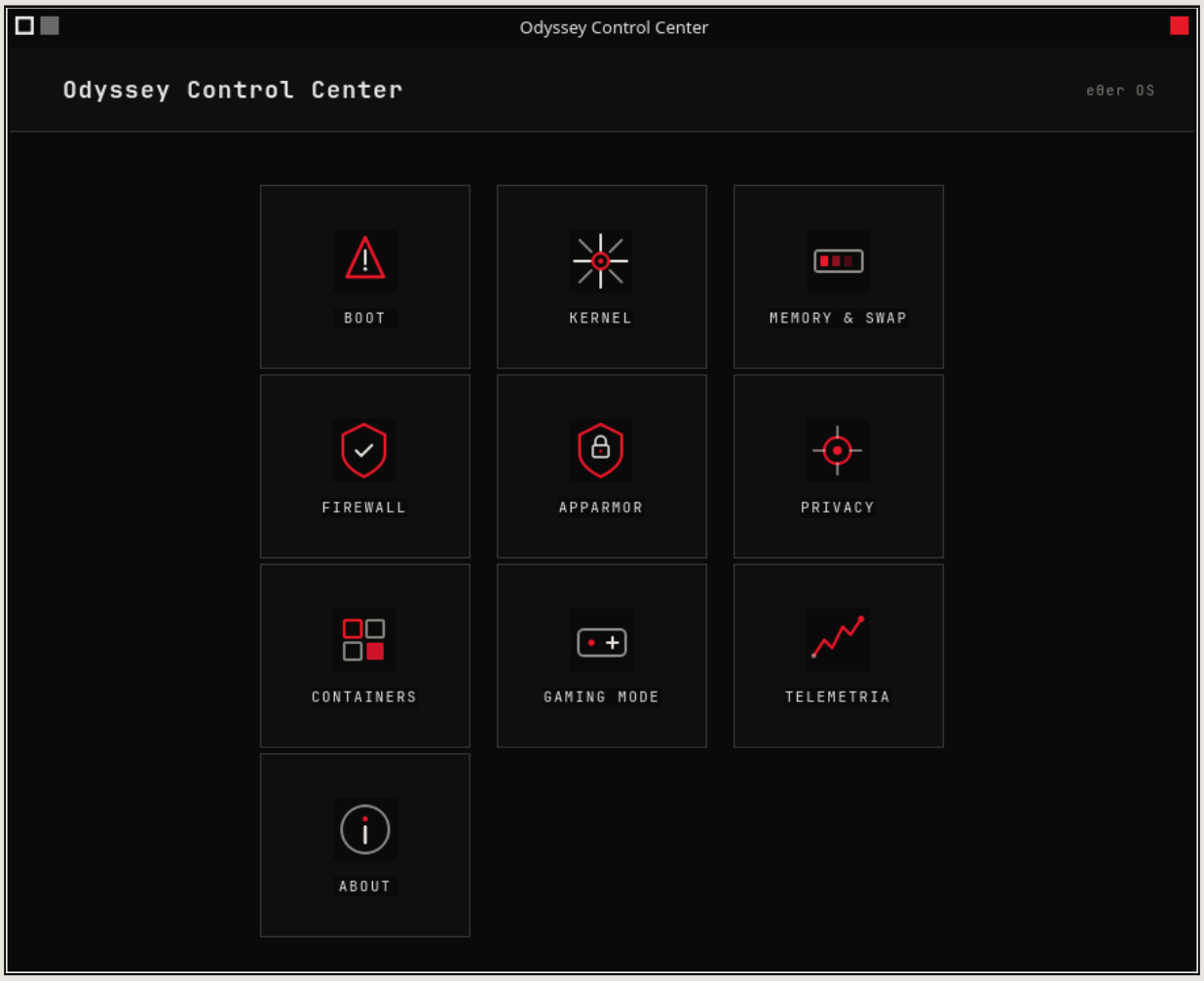

FIG · L’écran d’accueil du Control Center — une tuile par panneau.

Chaque panneau est une fine couche graphique par-dessus le véritable mécanisme Linux sous-jacent : la configuration de GRUB, l’interface sysctl du noyau lui-même, `nftables`, `podman`. Un panneau n’invente jamais son propre état privé — il lit les fichiers et commandes réels, te les montre, et écrit exactement ce que tu as demandé. Rien de ce qu’il fait n’est caché, et tout ce qu’il fait peut aussi se faire à la main, depuis un terminal, avec les commandes exactes que ce guide montre à côté de chaque bouton.

Ceci est un guide pratique, pas une référence de chaque option — il répond à « qu’est-ce que je clique, et qu’est-ce qui se passe si je le fais ». Chaque chapitre a la même structure en deux parties : d’abord un **guide visuel**, capture après capture, bouton après bouton, écrit pour qui n’a jamais ouvert le Control Center — puis une section **ADVANCED**, signalée par une bannière noire, qui entre dans le détail : chemins exacts des fichiers, les commandes réelles que chaque bouton exécute en dessous, et quoi

faire depuis un terminal si tu préfères sauter complètement le panneau. Lis seulement le guide visuel, ou saute directement à Advanced si tu connais déjà Linux — les deux sont complets à eux seuls.

Pour les deux panneaux avec de vrais enjeux de sécurité — Firewall et AppArmor — la référence approfondie est le **Security and Hardening Guide** ; ici tu trouveras le guide visuel, l'essentiel, et un renvoi.

1.1 Motifs que tu retrouveras dans chaque panneau

Certaines conventions se répètent dans les dix panneaux — les apprendre une fois vaut pour tous.

LA DEMANDE DE MOT DE PASSE, UNE SEULE FOIS PAR PANNEAU

La plupart des panneaux doivent lire ou écrire des fichiers qui appartiennent à root — la configuration de GRUB, sysctl, l'ensemble de règles du pare-feu. La première fois qu'un panneau en a besoin, tu reçois une seule boîte de dialogue d'authentification. Les panneaux sont construits pour regrouper leurs lectures : ouvrir Boot, par exemple, lit la configuration de GRUB **et** l'état du firmware EFI en une seule requête, donc tu tapes ton mot de passe une fois, pas deux. Rouvrir un panneau dans la même session ne le redemande pas ; les actions suivantes (appliquer un changement, recharger) montrent leur propre progression dans le journal du panneau lui-même.

LE SUIVI DES ÉTAPES

Les panneaux qui te guident à travers l'installation de quelque chose avant que ce soit utilisable — os-prober dans Boot, SearXNG et Tor dans Privacy, le moteur de conteneurs dans Containers — affichent un petit suivi numéroté en haut : une rangée d'étapes, celle en cours mise en évidence, et un seul bouton d'action qui fait toujours **l'étape suivante**, quelle qu'elle soit. Tu n'as jamais à deviner sur quoi cliquer en premier.

LE JOURNAL EN BAS

Chaque panneau se termine par un petit journal défilant. Ce n'est pas de la décoration — c'est la commande exacte qui a été exécutée, sa sortie, et un verdict coloré (vert pour le succès, rouge pour l'échec). Si un panneau fait quelque chose que tu ne comprends pas, le journal est l'endroit où se trouve la vraie réponse.

TRY VS. MAKE PERSISTENT

Une paire de boutons récurrente, surtout dans Kernel et Memory : **TRY NOW** (ou **APPLY NOW**) change le système en cours d'exécution immédiatement, et revient en arrière au prochain redémarrage si tu ne fais rien d'autre. **MAKE PERSISTENT** écrit la même valeur dans un fichier de configuration sous `/etc/`, pour qu'elle survive aussi aux redémarrages. Cette séparation existe exprès — essayer une valeur ne coûte rien, et rien n'est permanent tant que tu ne l'as pas dit, deux fois.

CHAPTER 2

Boot

Le panneau Boot a deux onglets sur toute machine qui démarre en mode UEFI : **GRUB** (le menu de démarrage lui-même) et **EFI** (le niveau firmware en dessous). Sur les machines plus anciennes, BIOS uniquement, seul GRUB existe, parce que la couche EFI en dessous n'est tout simplement pas là.

> **DEUX NIVEAUX, PAS UN**
>
> **GRUB** est le menu de démarrage que tu vois à l'écran — il décide quel système d'exploitation démarre une fois que GRUB lui-même est déjà en cours d'exécution. Le **firmware EFI/UEFI** est un niveau en dessous de GRUB — il décide à quel chargeur de démarrage la machine confie le contrôle en premier, avant même que GRUB ne se soit chargé. La plupart des gens ne touchent jamais l'onglet EFI ; il existe pour les cas que GRUB ne peut pas atteindre, comme une seconde installation Windows qui gère sa propre entrée de démarrage de manière indépendante.

2.1 Guide visuel — trouver un autre système d'exploitation et définir le défaut

L'onglet GRUB s'ouvre sur la section **Other OS detection**. Odyssey ne scanne pas les autres systèmes par défaut — la plupart des installations n'ont qu'Odyssey sur le disque, et scanner coûte du temps à chaque régénération du menu de démarrage pour rien. L'activer est une séquence guidée de trois étapes, affichée comme un petit suivi numéroté.

1. **Installed**

   L'outil de scan, os-prober, n'est pas encore sur le système. Un seul bouton : **INSTALL OS-PROBER**.

2. **Enable scan**

   L'outil est installé, mais GRUB est encore configuré pour l'ignorer — le défaut d'Odyssey. Un seul bouton : **ENABLE SCAN**.

3. **Scan**

   GRUB va maintenant l'utiliser. Un seul bouton, celui que tu réutiliseras chaque fois que tu ajoutes ou retires un autre système à l'avenir : **SCAN & REGENERATE**.

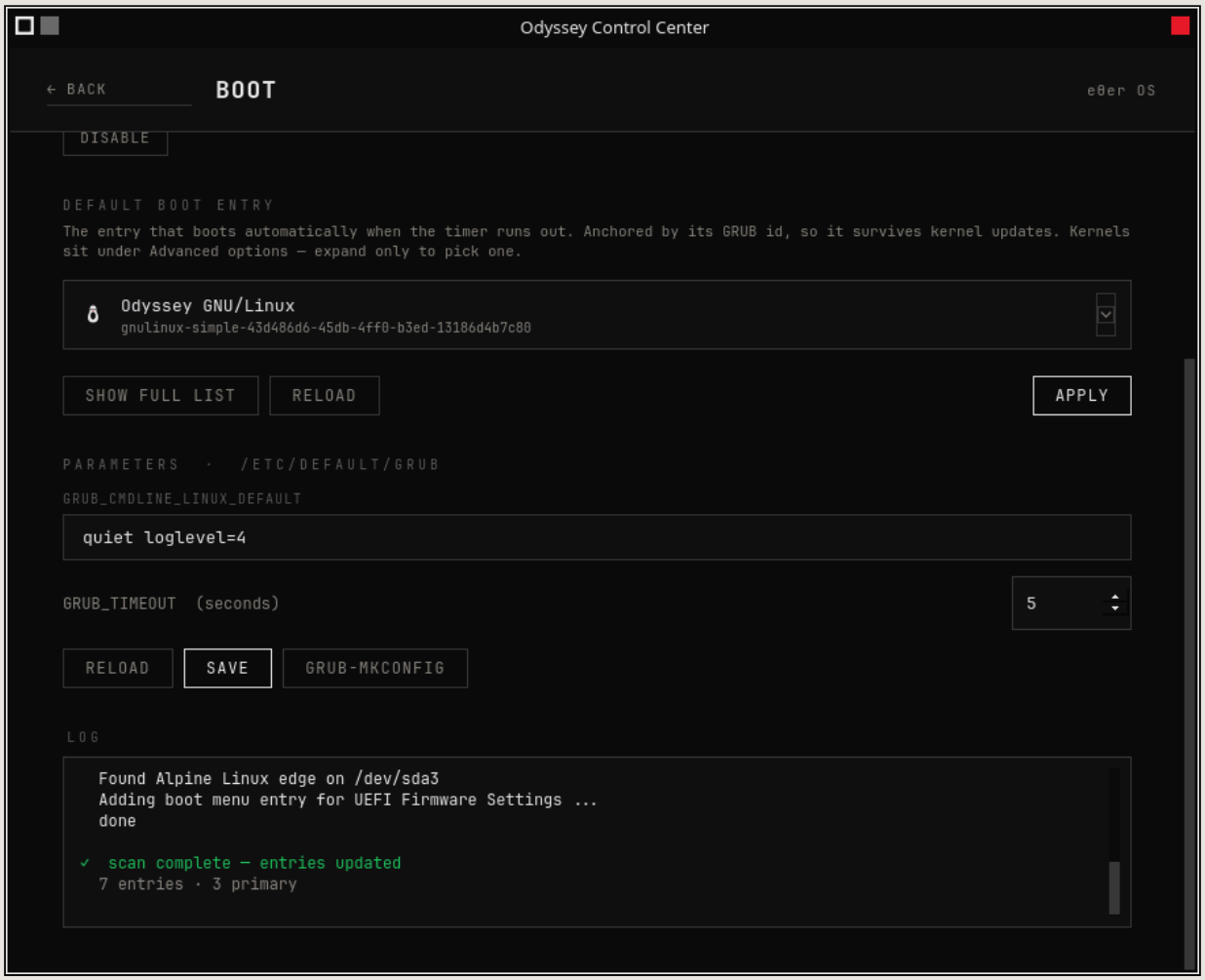

FIG · Les trois étapes en vert — os-prober installé, scan activé, prêt. Le bouton d'action unique correspond toujours à l'étape en cours.

Clique sur les étapes (chacune redemande l'authentification seulement si nécessaire) et lance le scan. Le panneau te montre exactement ce qu'il a trouvé avant que tu ne quittes la section :

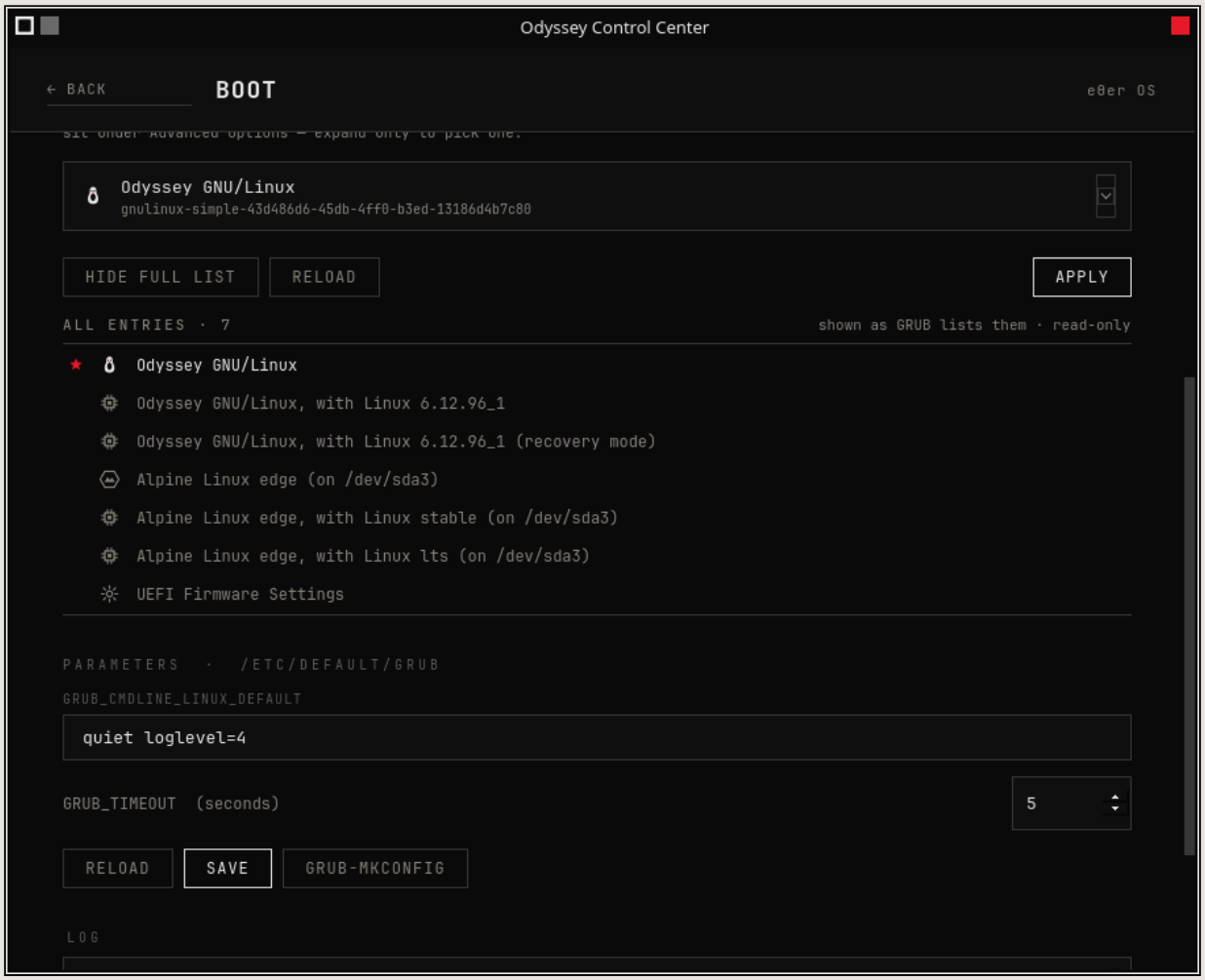

FIG · Scan terminé : Alpine Linux trouvé sur /dev/sda3, aux côtés des propres entrées d'Odyssey et de l'entrée UEFI Firmware Settings — 7 entrées, 3 primaires.

Le journal le confirme en termes simples — **« scan complete — entries updated · 7 entries · 3 primary »**. Tout ce qu'os-prober trouve désormais apparaît automatiquement dans la section ci-dessous, prêt à être choisi.

2.2 Guide visuel — choisir ce qui démarre par défaut

Fais défiler jusqu'à **Default boot entry**. Elle affiche le défaut actuel sous forme de ligne ; cliquer dessus ouvre un menu déroulant avec tout ce que tu peux choisir à la place.

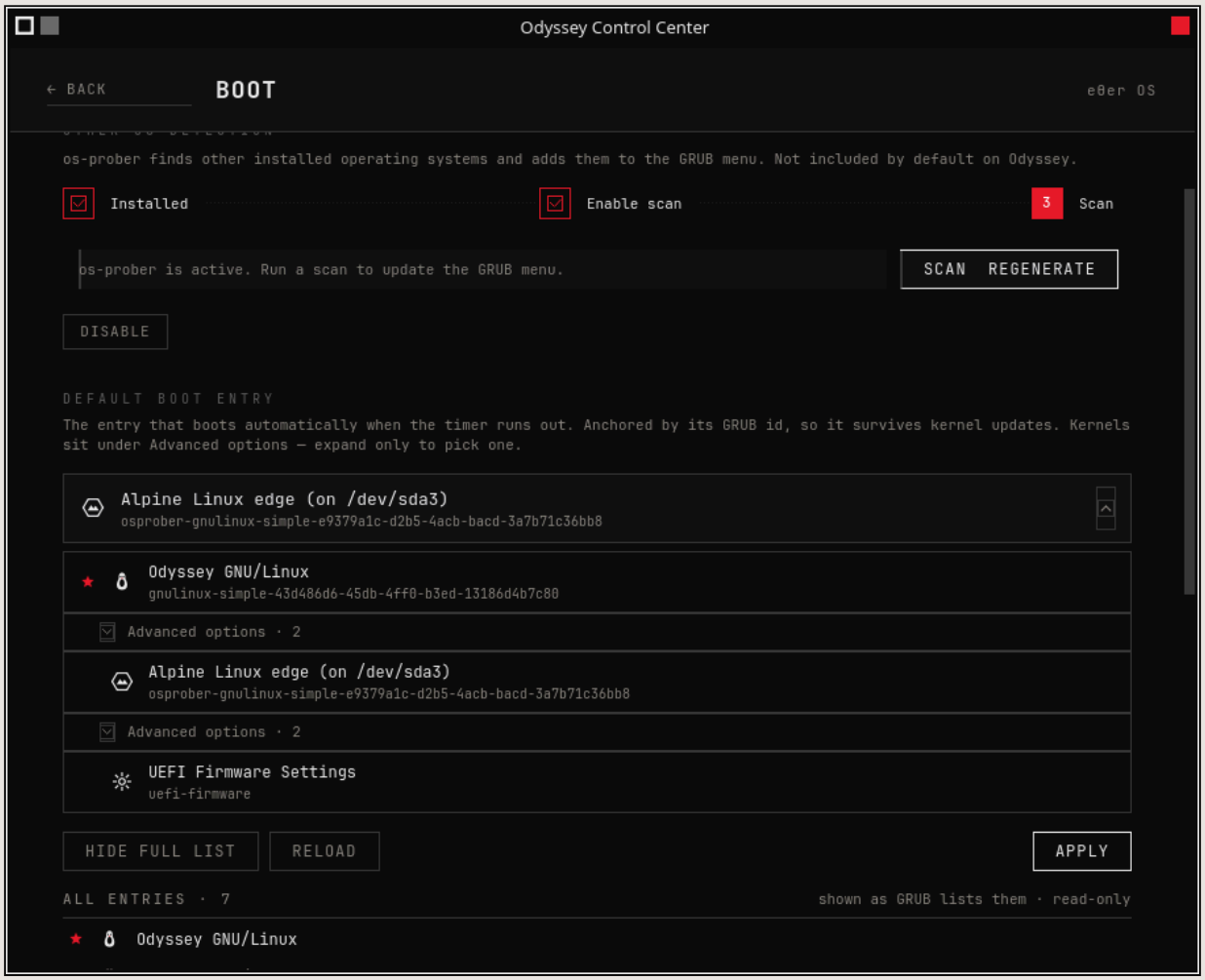

FIG · Le menu déroulant ouvert, avec Alpine Linux sélectionné comme choix en attente — pas encore appliqué.

Choisir une entrée dans la liste **met en attente** le choix — il n'a pas encore d'effet. On voit qu'il est seulement en attente parce que la case affiche toujours la nouvelle sélection mais rien n'a été écrit. Le bouton qui le confirme est **APPLY**, à droite :

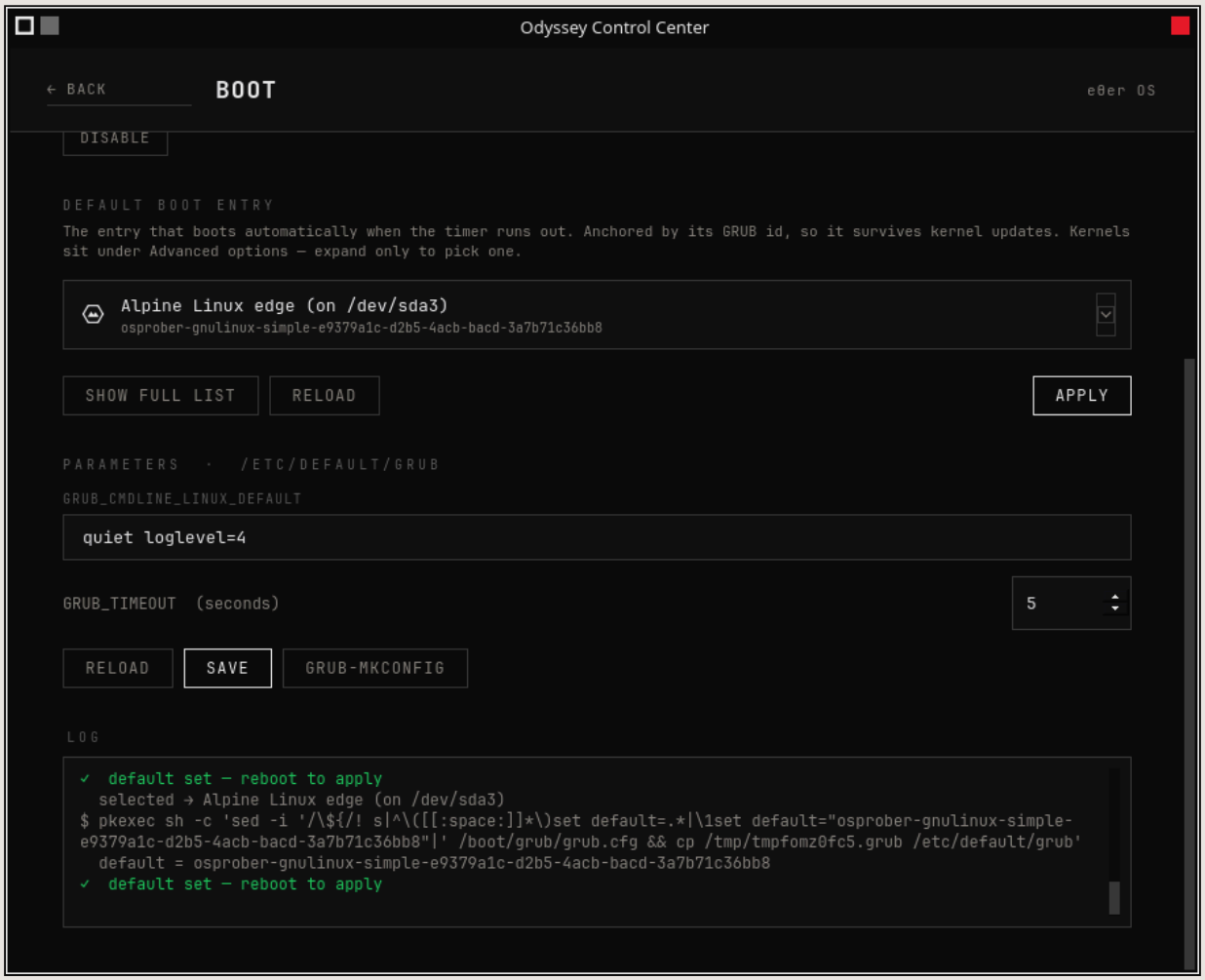

FIG · Après APPLY : le journal confirme « default set — reboot to apply », et le sélecteur affiche désormais Alpine Linux comme défaut actif.

WHAT	Boot → onglet GRUB → clique sur l'entrée par défaut actuelle → choisis-en une différente dans le menu déroulant → clique sur **APPLY**.
WHY	Tu es en double amorçage et tu veux qu'un système différent démarre automatiquement, ou une mise à jour du noyau a laissé un ancien noyau sélectionné et tu veux revenir au plus récent.
RESULT	L'entrée choisie démarre automatiquement à partir du prochain redémarrage. Le changement est ancré à l'id stable de cette entrée, donc il survit aux futures mises à jour du noyau.
RISK	Aucun pour tes données — dans le pire des cas tu choisis la mauvaise entrée et tu reviens la changer à nouveau.

SHOW FULL LIST affiche chaque entrée exactement comme GRUB lui-même les liste — en lecture seule, groupées de façon à ce que les variantes par noyau individuel se trouvent sous une ligne extensible « Advanced options » plutôt que d'encombrer la liste principale. **RELOAD** recharge la configuration de GRUB si tu penses qu'elle a changé en dehors du panneau.

2.3 Guide visuel — l'onglet EFI

Clique sur l'onglet **EFI** en haut. C'est le niveau firmware — ce que le menu de démarrage de ta propre carte mère affiche avant même qu'un système d'exploitation ait commencé.

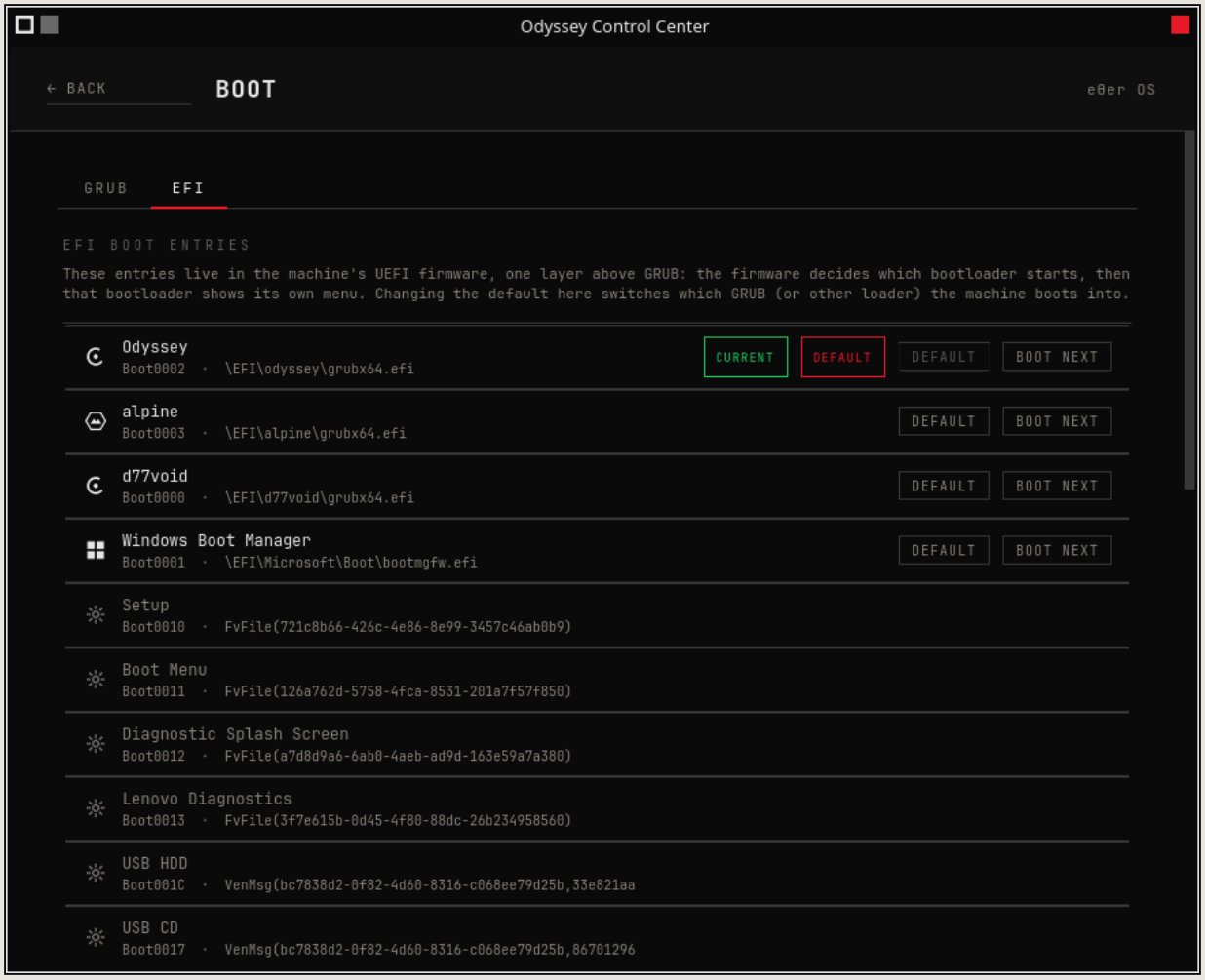

FIG · L'onglet EFI : Odyssey marqué CURRENT et DEFAULT en vert/rouge, alpine, d77void et Windows Boot Manager chacun avec leurs propres boutons DEFAULT / BOOT NEXT, plus les entrées utilitaires du firmware en dessous.

Chaque ligne est une entrée de firmware — son nom, son id GRUB (`Boot0002` et ainsi de suite), et le fichier `.efi` exact vers lequel elle pointe. Deux boutons par ligne :

BOUTON	CE QU'IL FAIT
DEFAULT	Fait de cette entrée le défaut permanent du firmware — elle démarre à chaque redémarrage à partir de maintenant.
BOOT NEXT	Une exception ponctuelle — cette entrée démarre seulement au **prochain** redémarrage, puis l'ordre normal reprend automatiquement. Une bannière apparaît en haut tant qu'une exception est en attente, avec un bouton CANCEL.

WHAT	Boot → onglet EFI → trouve l'entrée que tu veux (p. ex. « Odyssey » ou « Windows Boot Manager ») → clique sur DEFAULT, ou BOOT NEXT pour une exception ponctuelle.
WHY	Si le menu de démarrage de ta carte mère lance carrément le mauvais chargeur de démarrage — pas un réglage de GRUB, un réglage du firmware — c'est le seul endroit qui corrige ça.
RESULT	DEFAULT : l'entrée passe en tête de l'ordre de démarrage du firmware de façon permanente. BOOT NEXT : elle démarre une fois, puis revient en arrière.
RISK	Faible, et toujours récupérable depuis ce même onglet ou depuis l'écran de configuration UEFI de ta propre carte mère (généralement Suppr ou F2 à l'allumage) si quelque chose démarre de façon inattendue.

2.4 Paramètres du noyau

La dernière section de l'onglet GRUB est la ligne de commande du noyau en clair — modifiable comme un champ de texte normal, aucune syntaxe GRUB requise — et le délai du menu en secondes. **SAVE** les écrit dans `/etc/default/grub` ; **GRUB-MKCONFIG** régénère le vrai menu de démarrage à partir de ce fichier, l'étape qui rend un paramètre enregistré effectif au prochain démarrage. Le panneau réapplique toujours ton choix de défaut actuel quand il régénère, donc cela ne réinitialise jamais quel système démarre par défaut.

```
▸ ADVANCED

Past this point: config file paths, raw commands, and what the panel is doing underneath.
```

2.5 Advanced

CE QUI SE PASSE RÉELLEMENT

La configuration GRUB d'Odyssey est **plate** — aucun bloc `submenu`. Chaque noyau est un `menuentry` de premier niveau, ce qui signifie qu'un id stable (pas une position numérique) peut toujours le sélectionner correctement, même entre deux mises à jour du noyau. Le regroupement « Advanced options » que tu vois dans le sélecteur est une pure commodité visuelle construite en associant l'UUID partagé entre l'entrée simple d'un système et ses variantes par noyau — le fichier de GRUB lui-même n'a pas ce regroupement.

Définir un défaut écrit à deux endroits dans la même action : le `GRUB_DEFAULT` persistant dans `/etc/default/grub`, et — parce que `grub.cfg` peut ne pas être régénéré immédiatement — une modification directe et chirurgicale via `sed` de la ligne active `set default="..."` dans `/boot/grub/grub.cfg` lui-même, en sautant la branche one-shot `${next_entry}` qu'utilise `grub-reboot`. C'est pourquoi un défaut prend effet au prochain démarrage sans forcer d'abord un `grub-mkconfig` complet.

```
$ cat /etc/default/grub | grep GRUB_DEFAULT
$ grep 'set default=' /boot/grub/grub.cfg
```

OS-PROBER À LA MAIN

```
$ sudo xbps-install -y os-prober
# active le scan — modifie GRUB_DISABLE_OS_PROBER dans /etc/default/grub
$ sudo sed -i 's/GRUB_DISABLE_OS_PROBER=true/GRUB_DISABLE_OS_PROBER=false/' /etc/default/
grub
$ sudo os-prober
$ sudo grub-mkconfig -o /boot/grub/grub.cfg
```

EFI À LA MAIN — EFIBOOTMGR

L'onglet EFI est une interface pour `efibootmgr`. Tout ce qu'il fait correspond à une commande :

```
# liste toutes les entrées de firmware, dans l'ordre de démarrage actuel
$ sudo efibootmgr -v
# réordonne : place l'entrée 0002 en premier (= DEFAULT)
$ sudo efibootmgr -o 0002,0000,0001,0003
# démarrage ponctuel sur l'entrée 0003 (= BOOT NEXT)
$ sudo efibootmgr -n 0003
# annule un BootNext en attente
$ sudo efibootmgr -N
```

CHARGEURS ORPHELINS

Il arrive qu'un installateur écrive son propre chargeur de démarrage sur l'EFI System Partition sans enregistrer d'entrée de firmware pour lui — le chargeur existe sur le disque mais le firmware ne sait pas qu'il est là. Le panneau d'Odyssey scanne directement l'ESP et les liste séparément comme **« ON DISK, NOT IN FIRMWARE »**, chacun avec un REGISTER en un clic. À la main, l'équivalent est `efibootmgr -c -d /dev/sdX -p N -L "Nom" -l '\EFI\nom\bootx64.efi'` — notez que les entrées nouvellement créées sont placées **en premier** dans l'ordre de démarrage par le firmware lui-même, donc réapplique DEFAULT sur ton entrée habituelle après, si tu ne veux pas qu'elle prenne le dessus.

FICHIERS TOUCHÉS PAR CE PANNEAU

FICHIER	RÔLE
`/etc/default/grub`	Ligne de commande du noyau, délai, GRUB_DEFAULT, GRUB_DISABLE_OS_PROBER
`/boot/grub/grub.cfg`	Le menu de démarrage généré lui-même — ne le modifie jamais à la main ; il est écrasé par grub-mkconfig
Variables EFI (NVRAM)	Entrées de démarrage du firmware — ce n'est pas du tout un fichier ; lu et écrit entièrement via efibootmgr

CHAPTER 3

Kernel

Quatre onglets — **SYSCTL**, **CPU GOVERNOR**, **HUGEPAGES**, **MODULES** — plus une rangée de préréglages en un clic au-dessus des quatre, qui s'applique aux quatre à la fois.

3.1 Guide visuel — les préréglages

En haut : la version de ton noyau en cours d'exécution, l'ordonnanceur actif, et quatre boutons de préréglage — **DESKTOP**, **GAMING**, **SERVER**, **BALANCED**.

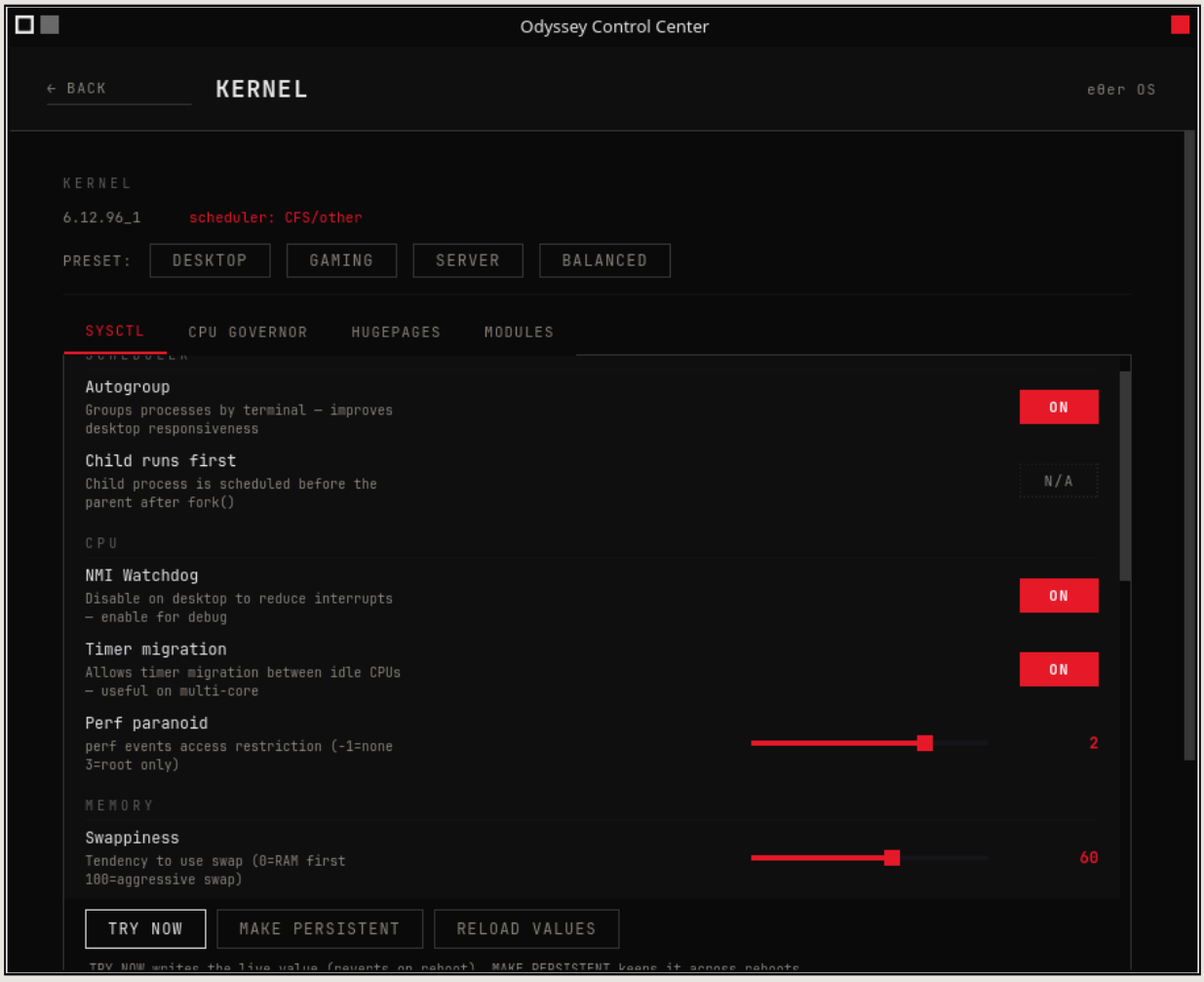

FIG · L'onglet SYSCTL fraîchement ouvert : noyau 6.12.96_1, ordonnanceur CFS/other, les quatre préréglages, et les groupes Scheduler/CPU/Memory en dessous.

PRÉRÉGLAGE	QUAND ÇA VAUT LE COUP
DESKTOP	Une machine du quotidien — navigation, bureautique, multitâche léger. Swappiness plus basse, pression modérée sur le cache, timer migration active pour une consommation énergétique douce au repos. Le défaut sensé pour la plupart des gens.
GAMING	Swappiness poussée presque à zéro (garde tout en RAM), timer migration désactivée (les cœurs restent dédiés au lieu de faire migrer le travail), restrictions sur les événements perf assouplies pour des outils de profilage comme MangoHud. À utiliser quand les temps de frame comptent plus que la consommation au repos.
SERVER	Swappiness et tolérance à la pression du cache plus élevées, NMI watchdog actif, restrictions plus strictes sur les pointeurs du noyau/journaux. À utiliser sur une machine sans tête qui privilégie la stabilité et la sécurité plutôt que la réactivité interactive.
BALANCED	Un compromis entre Desktop et Gaming — pour une machine qui est parfois une station de travail, parfois une session de jeu, et tu ne veux pas basculer entre les préréglages en permanence.

WHAT	Kernel → clique sur DESKTOP, GAMING, SERVER ou BALANCED.
WHY	Ajuster à la main une douzaine de valeurs sysctl, c'est beaucoup demander à quelqu'un qui veut juste « de bons réglages pour jouer » — un préréglage est exactement ça, déjà décidé.
RESULT	Chaque valeur dans l'onglet SYSCTL saute aux chiffres du préréglage, prête et visible — il te faut quand même TRY NOW ou MAKE PERSISTENT (ci-dessous) pour vraiment les appliquer.
RISK	Aucun — choisir un préréglage ne fait que remplir le formulaire.

3.2 Guide visuel — SYSCTL, groupe par groupe

Fais défiler vers le bas et chaque paramètre est regroupé sous un en-tête, chacun avec une description en langage simple directement dans le panneau — tu n'as jamais besoin de savoir ce que signifie `vm.compaction_proactiveness` sans d'abord le lire là.

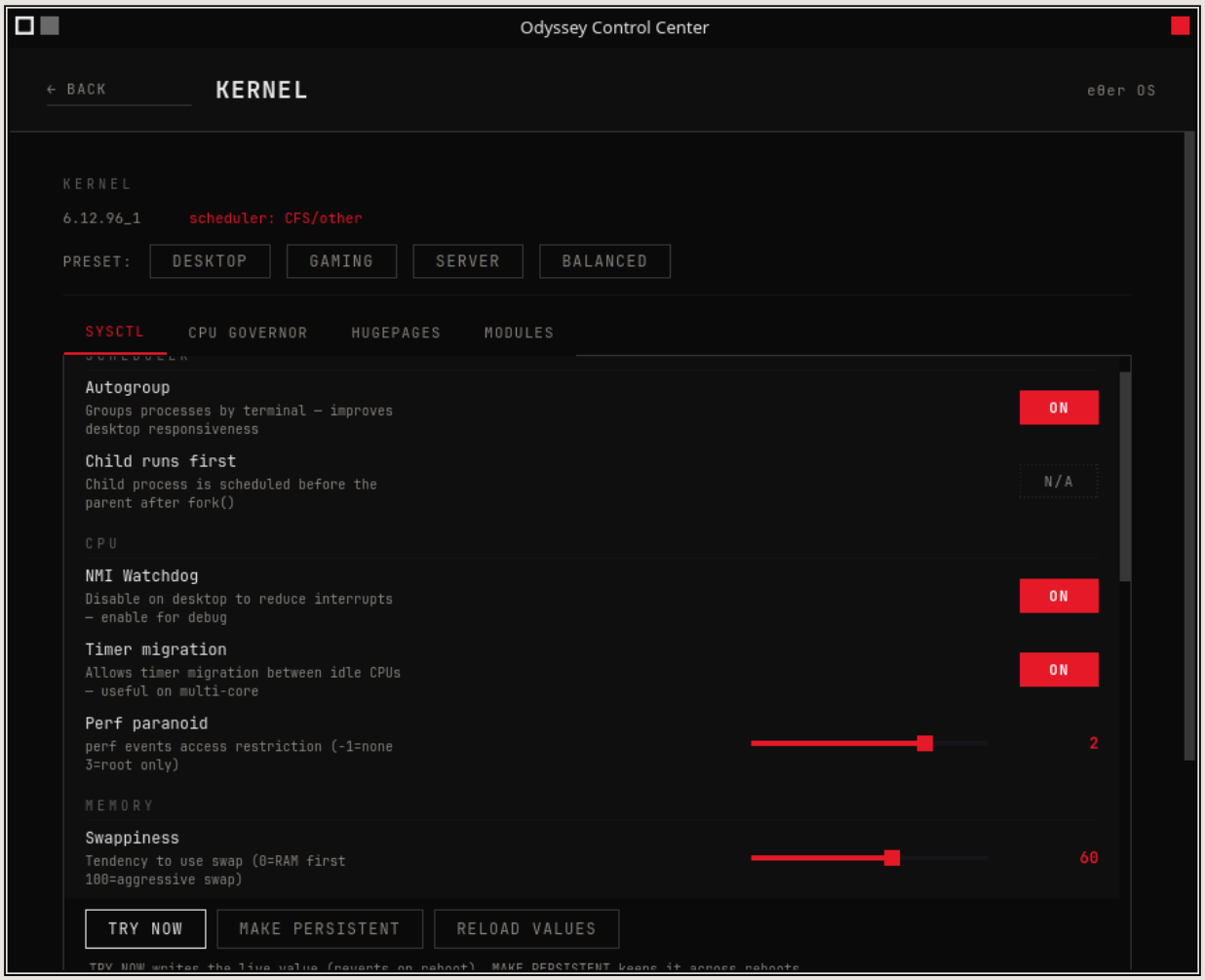

FIG · Groupes Scheduler et CPU : Autogroup, Child runs first, NMI Watchdog, Timer migration, Perf paranoid — chacun avec son propre interrupteur ou curseur et une ligne d'explication.

GROUPE	CE QU'IL CONTIENT
Scheduler	Comment les processus sont regroupés et priorisés
CPU	Watchdog, timer migration, restrictions sur le monitoring des performances
Memory	Swappiness, pression du cache, seuils d'écriture des pages sales
Network	TCP Fast Open, taille de la file de backlog, limites des tampons de socket
Security	Si les pointeurs du noyau et les messages de journal sont cachés aux utilisateurs sans privilèges

Deux boutons en bas appliquent tes modifications — et ils sont délibérément différents :

WHAT	TRY NOW
WHY	Tester une valeur avec zéro risque qu'elle survive à un mauvais redémarrage.
RESULT	Écrite en direct via `sysctl` — active immédiatement, seulement pour cette session.
RISK	Aucun — revient en arrière automatiquement au redémarrage.

WHAT	MAKE PERSISTENT
WHY	Tu as testé une valeur (ou un préréglage entier) et tu veux qu'elle survive aux redémarrages.
RESULT	Écrite dans `/etc/sysctl.d/99-odyssey.conf` **et** appliquée en direct dans le même clic.
RISK	Faible — c'est un fichier texte ; supprime la ligne ou réapplique pour annuler.

RELOAD VALUES relit ce qui est actuellement actif, en écartant tout ce qui a été saisi mais pas appliqué.

3.3 Guide visuel — CPU governor

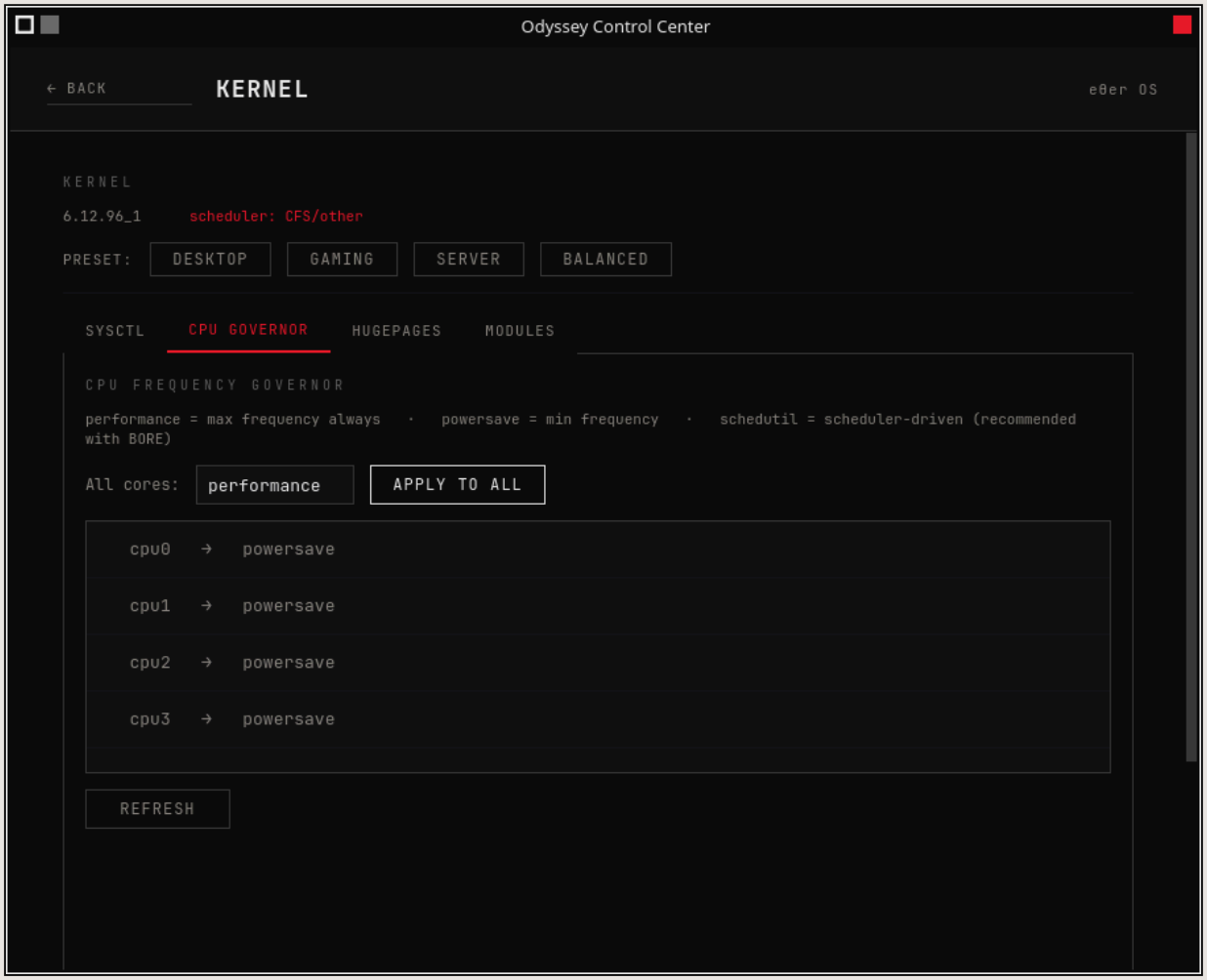

FIG · L'onglet CPU GOVERNOR : « All cores » réglé sur performance avec APPLY TO ALL, et une liste par cœur en dessous montrant encore tout en powersave — un décalage en attente d'être corrigé.

Le governor décide avec quelle agressivité ton CPU change de fréquence d'horloge. Un menu déroulant applique un governor à tous les cœurs à la fois ; la liste en dessous montre le governor **en direct** par cœur individuel — c'est ainsi que tu repères un cœur coincé sur le mauvais réglage, exactement comme dans la capture ci-dessus où « All cores » dit performance mais chaque cœur en dessous affiche encore powersave, ce qui signifie qu'APPLY TO ALL n'a pas encore été cliqué.

GOVERNOR	COMPORTEMENT
performance	Fréquence maximale, toujours — le plus réactif, le plus gourmand

GOVERNOR	COMPORTEMENT
powersave	Fréquence minimale, toujours — le plus silencieux, le moins réactif
schedutil	Fréquence pilotée par la propre vision de charge de l'ordonnanceur — recommandé avec l'ordonnanceur BORE

Seuls les governors que ton CPU et ton noyau spécifiques exposent réellement apparaissent dans le menu déroulant.

3.4 Guide visuel — hugepages

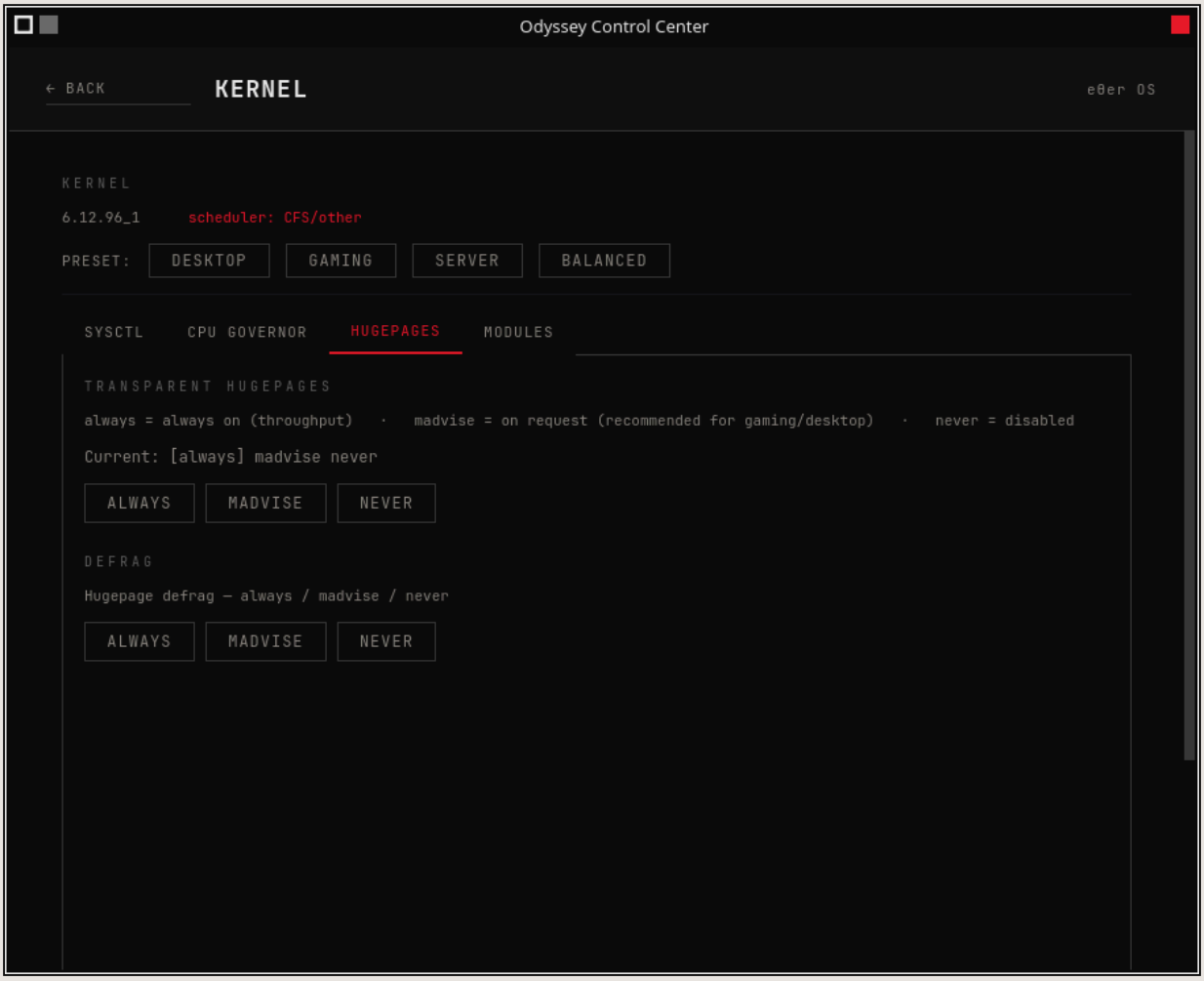

FIG · L'onglet HUGEPAGES : Transparent Hugepages affichant « Current: [always] madvise never » avec ALWAYS mis en évidence, et la même rangée de trois boutons à nouveau pour Defrag en dessous.

Deux réglages indépendants, chacun une rangée de trois boutons appliqués à l'instant du clic — la valeur en direct actuelle est toujours affichée entre crochets au-dessus des boutons, comme `[always] madvise never` dans la capture.

RÉGLAGE	CE QU'IL CONTRÔLE
Transparent hugepages	ALWAYS = toujours actif partout (débit maximal) ; MADVISE = seulement quand une appli le demande explicitement (recommandé pour le gaming/bureau) ; NEVER = complètement désactivé

RÉGLAGE	CE QU'IL CONTRÔLE
Defrag	Avec quelle agressivité le noyau compacte la mémoire pour **former** des hugepages à la demande — mêmes trois options, indépendant du réglage ci-dessus

3.5 Guide visuel — modules

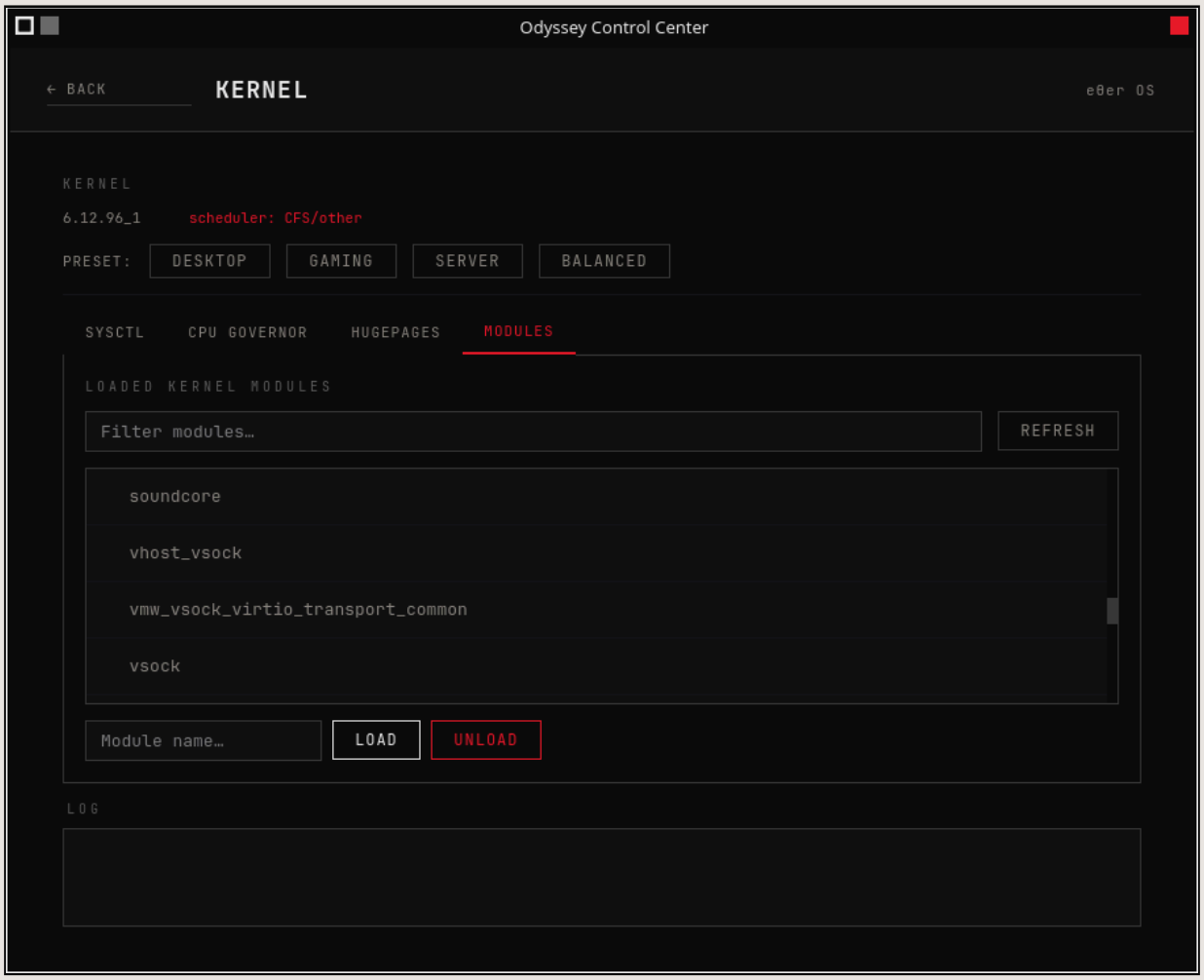

FIG · L'onglet MODULES : un champ de filtre, une liste en direct (soundcore, vhost_vsock, vsock…), et une rangée LOAD/UNLOAD en bas pour un module saisi par son nom.

Une liste en direct et filtrable de chaque module du noyau actuellement chargé — tape dans le filtre pour la restreindre, REFRESH pour la relire. Contrairement aux trois autres onglets, celui-ci te permet d'agir directement : tape un nom de module et clique sur **LOAD** ou **UNLOAD**.

WHAT	Kernel → onglet MODULES → tape un nom de module → LOAD ou UNLOAD.
WHY	Il arrive d'avoir besoin d'un pilote ou module de fonction spécifique actif (ou retiré) sans redémarrer — tester du matériel, libérer un périphérique qu'un autre module retient.
RESULT	Le module est inséré dans ou retiré du noyau en cours d'exécution immédiatement.
RISK	Modéré spécifiquement pour UNLOAD — retirer un module dont dépend autre chose peut déstabiliser ce qui l'utilisait (un pilote d'affichage, un système de fichiers). LOAD est en général sûr ; le noyau refuse d'emblée de charger un module incompatible plutôt que de l'appliquer à moitié.

▸ **ADVANCED**

Past this point: config file paths, raw commands, and what the panel is doing underneath.

3.6 Advanced

LES CLÉS SYSCTL EXACTES

CLÉ	ÉTIQUETTE DANS LE PANNEAU
`kernel.sched_autogroup_enabled`	Autogroup
`kernel.sched_child_runs_first`	Child runs first
`kernel.nmi_watchdog`	NMI Watchdog
`kernel.timer_migration`	Timer migration
`kernel.perf_event_paranoid`	Perf paranoid
`vm.swappiness`	Swappiness
`vm.vfs_cache_pressure`	VFS cache pressure
`vm.dirty_ratio` / `vm.dirty_background_ratio`	Dirty ratio % / Dirty background ratio %
`vm.compaction_proactiveness`	Compaction proactiveness
`vm.watermark_boost_factor`	Watermark boost
`net.ipv4.tcp_fastopen`	TCP Fast Open
`net.core.netdev_max_backlog`	Netdev max backlog
`net.core.rmem_max`	Socket rmem max
`kernel.kptr_restrict` / `kernel.dmesg_restrict`	kptr restrict / dmesg restrict

VALEURS DES PRÉRÉGLAGES, DESKTOP VS GAMING

```
# DESKTOP
$ sysctl vm.swappiness=10 vm.vfs_cache_pressure=50 kernel.timer_migration=1
# GAMING
$ sysctl vm.swappiness=1 vm.vfs_cache_pressure=50 kernel.timer_migration=0
# les tableaux complets des préréglages se lisent directement dans panel_kernel.py —
# ouvre le code source du Control Center si tu veux chaque champ
```

À LA MAIN, SANS LE PANNEAU

```
# en direct, temporaire
$ sudo sysctl vm.swappiness=10
# persistant — le même fichier qu'écrit le panneau
$ echo "vm.swappiness=10" | sudo tee -a /etc/sysctl.d/99-odyssey.conf
$ sudo sysctl -system
# CPU governor, par cœur
$ echo schedutil | sudo tee /sys/devices/system/cpu/cpu*/cpufreq/scaling_governor
# hugepages
$ echo madvise | sudo tee /sys/kernel/mm/transparent_hugepage/enabled
$ echo madvise | sudo tee /sys/kernel/mm/transparent_hugepage/defrag
# modules
$ sudo modprobe MODULE_NAME
$ sudo modprobe -r MODULE_NAME
```

POURQUOI TRY NOW ET MAKE PERSISTENT SONT DES ACTIONS SÉPARÉES

Les écritures `sysctl` sont par nature toujours en direct uniquement — le noyau n'a pas de persistance intégrée. MAKE PERSISTENT, c'est Odyssey qui choisit un fichier canonique unique, `/etc/sysctl.d/99-odyssey.conf`, et fait à la fois l'écriture en direct et l'écriture sur fichier ensemble pour que les deux ne divergent jamais. Modifier ce fichier directement et exécuter `sysctl --system` après est fonctionnellement identique à cliquer sur le bouton.

CHAPTER 4

Memory & Swap

Un seul écran, trois sections : un résumé en lecture seule, la swappiness, et ZRAM.

4.1 Guide visuel

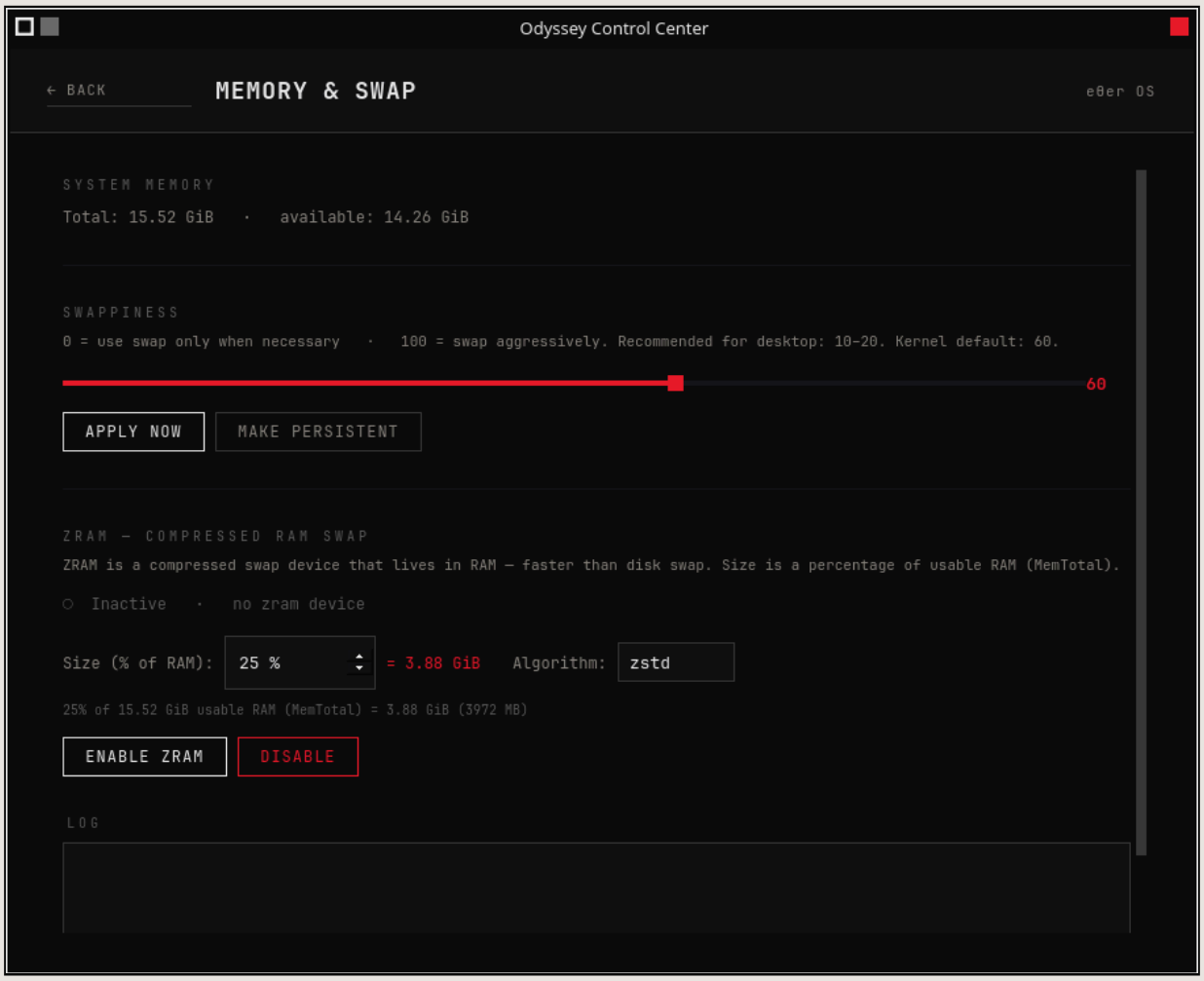

FIG · Le panneau complet : System memory (15,52 Gio au total, 14,26 Gio disponibles), le curseur Swappiness à 60 avec APPLY NOW / MAKE PERSISTENT, et ZRAM en dessous affichant « Inactive · no zram device » avec taille, algorithme et ENABLE/DISABLE.

SYSTEM MEMORY

Juste un résumé en haut — RAM totale et ce qui est actuellement disponible. Aucun contrôle ; c'est le contexte pour les deux sections ci-dessous.

SWAPPINESS

Un seul curseur de 0 à 100. 0 signifie « utilise le swap seulement quand il n'y a vraiment pas d'autre choix » ; 100 signifie « utilise le swap de façon agressive, bien avant que la RAM soit pleine ». Le défaut du noyau est 60 (ce que montre la capture) ; la suggestion du panneau lui-même indique 10–20 pour un bureau — le même chiffre que fixe le préréglage DESKTOP du panneau Kernel.

WHAT	Fais glisser le curseur → APPLY NOW pour tester, MAKE PERSISTENT une fois satisfait.
WHY	Une valeur plus basse garde ce que tu utilises activement dans la RAM rapide plutôt que de le déplacer vers le swap plus lent — généralement ce que tu veux avec suffisamment de mémoire installée.
RESULT	APPLY NOW la change en direct, immédiatement, seulement pour cette session. MAKE PERSISTENT l'écrit dans un fichier de configuration pour qu'elle survive aussi au redémarrage.
RISK	Très faible. Une valeur extrêmement basse sur une machine vraiment à court de RAM peut la rendre plus encline à manquer de mémoire brutalement sous forte charge plutôt que de faire du swap progressivement — si ça arrive, remonte-la.

ZRAM — SWAP COMPRESSÉ EN RAM

ZRAM est un périphérique de swap qui vit **à l'intérieur** de la RAM elle-même, compressé — nettement plus rapide que le swap sur disque car aucun disque n'est impliqué. La taille est un pourcentage de ta RAM utilisable totale, pas un nombre fixe, donc le réglage s'adapte intelligemment à des machines différentes ; une lecture en direct à côté du champ de taille convertit le pourcentage en Gio pour toi (dans la capture : 25 % = 3,88 Gio sur un système de 15,52 Gio).

WHAT	Fixe le pourcentage de taille et choisis un algorithme (zstd, lz4, lzo, ou deflate) → ENABLE ZRAM.
WHY	Le swap compressé en RAM absorbe la pression mémoire avec un impact sur les performances bien moindre que le swap sur disque — utile sur toute machine, spécialement une avec une RAM limitée.
RESULT	Un périphérique `zram0` est créé et activé à la taille et l'algorithme choisis ; la ligne de statut passe de « Inactive » à actif.
RISK	Faible. zstd (le défaut) équilibre bien le taux de compression et le coût CPU pour la plupart des machines. DISABLE le désactive proprement à tout moment.

▸ ADVANCED

Past this point: config file paths, raw commands, and what the panel is doing underneath.

4.2 Advanced

CE QUE « DÉPEND DU MATÉRIEL » SIGNIFIE VRAIMENT ICI

Les deux réglages interagissent avec ton matériel spécifique, pas seulement avec « plus de RAM = mieux » :

- **SSD NVMe rapide** : une swappiness dans la plage 10–20 coûte très peu même si elle fait du swap — le disque est assez rapide pour que ce soit à peine perceptible. ZRAM reste gagnant sur la latence et l'usure du SSD, mais l'écart est plus petit qu'avec un disque mécanique.
- **HDD mécanique ou eMMC comme swap** : garde la swappiness basse (0–10) — le swap sur disque sur des supports lents cause le ralentissement qu'on appelle « thrashing ». ZRAM est un gain bien plus important précisément ici.
- **Marge CPU** : ZRAM échange les I/O disque contre des cycles CPU dépensés à compresser et décompresser. Sur une machine limitée par le CPU (matériel plus ancien, ou déjà à la limite pendant le jeu), `lz4` compresse plus vite que `zstd` avec un moins bon taux — un compromis raisonnable si le temps CPU est la ressource la plus contrainte. Sur tout matériel moderne, le meilleur taux de `zstd` vaut son coût légèrement plus élevé.
- **Marge RAM** : un pourcentage de ZRAM important (comme 50 %) sur une machine avec seulement 8 Gio au total laisse moins de RAM réelle pour les applications, puisque ZRAM lui-même consomme de la mémoire réelle pour stocker les pages compressées. À 16 Gio ou plus, 25 % est un défaut confortable avec de la marge.

À LA MAIN

```
# swappiness, en direct
$ sudo sysctl vm.swappiness=15
# swappiness, persistant — le même fichier qu'écrit le panneau
$ echo "vm.swappiness=15" | sudo tee /etc/sysctl.d/99-odyssey.conf
# zram, configuration manuelle
$ sudo modprobe zram
$ echo zstd | sudo tee /sys/block/zram0/comp_algorithm
$ echo 4G | sudo tee /sys/block/zram0/disksize
$ sudo mkswap /dev/zram0
$ sudo swapon /dev/zram0 -p 100
# vérifie qu'il est actif
$ swapon -show
```

POURQUOI LE PANNEAU RÉINITIALISE ZRAM0 PLUTÔT QUE DE LE REDIMENSIONNER EN DIRECT

La taille d'un périphérique ZRAM ne peut pas être changée pendant qu'il est actif et utilisé comme swap — le noyau exige qu'il soit d'abord désactivé et réinitialisé. Le bouton ENABLE ZRAM du panneau exécute cette séquence automatiquement (swapoff, reset, reconfiguration, swapon), c'est pourquoi changer la taille ou l'algorithme sur un périphérique déjà actif se présente quand même comme une seule action propre plutôt que comme une erreur.

CHAPTER 5

Firewall

Le panneau Firewall a son propre chapitre dédié, en pleine profondeur, dans le **Security and Hardening Guide** — l'ensemble de règles expliqué ligne par ligne, les 21 préréglages avec leurs risques individuels, l'éditeur de configuration brute, et les équivalents en ligne de commande. Ce chapitre est la version guidée visuellement.

5.1 Guide visuel

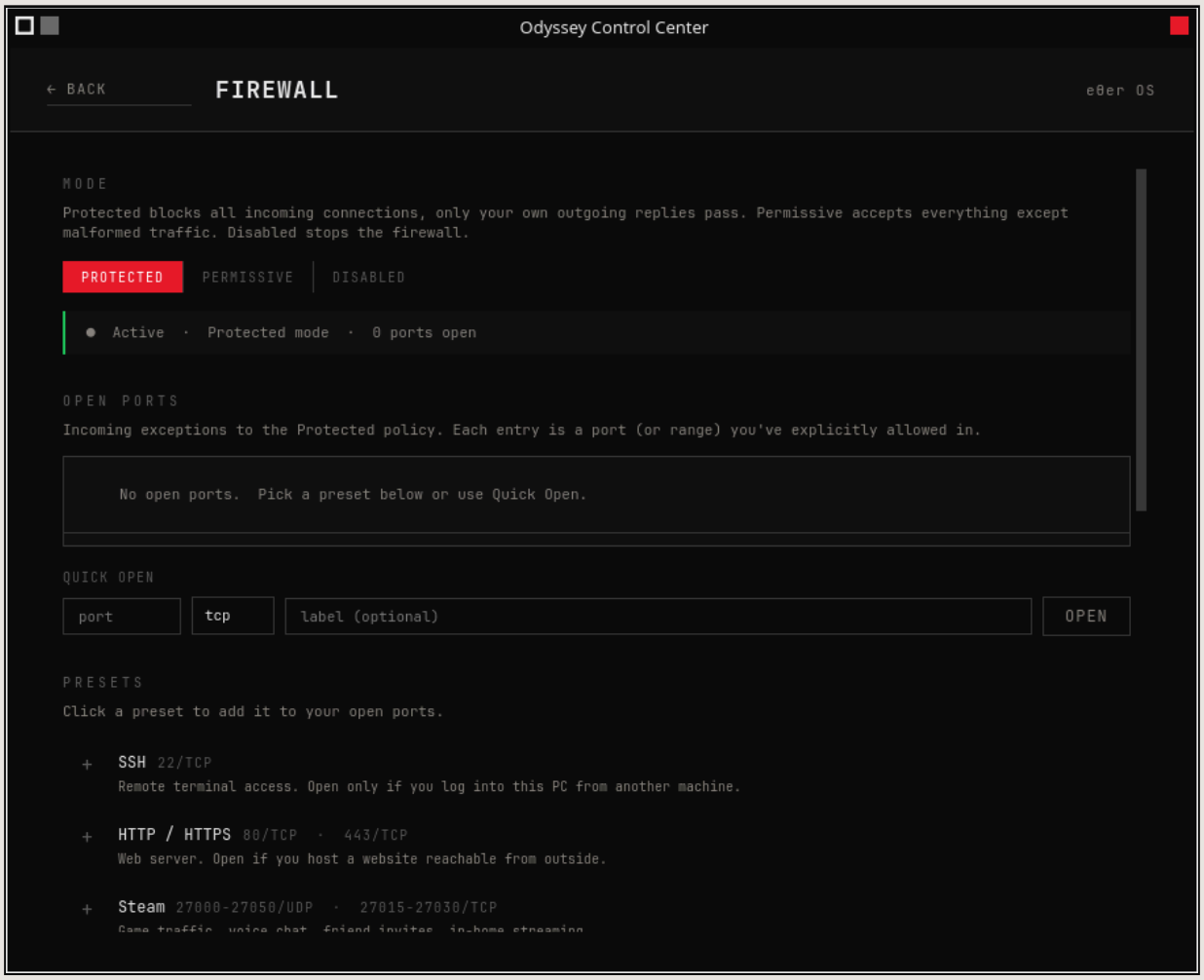

FIG · Mode PROTECTED actif, « 0 ports open », les champs Quick Open, et les cases de préréglage commençant par SSH, HTTP/HTTPS et Steam.

MODE

Trois boutons en haut, et le panneau explique chacun en une seule ligne juste au-dessus :

MODE	CE QU'IL FAIT
PROTECTED	Le défaut. Bloque toutes les connexions entrantes ; seules tes propres réponses sortantes passent.
PERMISSIVE	Accepte tout sauf le trafic malformé — un réglage de diagnostic pour « est-ce le pare-feu qui casse ceci ? », pas un endroit où rester.
DISABLED	Arrête complètement le pare-feu.

La ligne de statut en dessous confirme ce qui est réellement actif — dans la capture, **« Active · Protected mode · 0 ports open »**.

OPEN PORTS ET PRÉRÉGLAGES

OPEN PORTS liste tes exceptions explicites — vide par défaut, comme montré. **QUICK OPEN** prend port, protocole et étiquette optionnelle directement, pour tout ce qui n'est pas dans la liste des préréglages. **PRESETS** en dessous offre huit services courants en un clic chacun, chacun avec une ligne expliquant quand tu en aurais vraiment besoin (SSH : « accès terminal à distance, ouvre-le seulement si tu te connectes à ce PC depuis une autre machine »), plus treize autres sous un « MORE PRESETS » extensible.

WHAT	Firewall → clique sur une case de préréglage, ou remplis QUICK OPEN → SAVE & APPLY.
WHY	Tu fais tourner quelque chose — un serveur de jeu, un serveur de médias, un serveur de développement — qu'une autre machine doit atteindre.
RESULT	Ce port s'ouvre ; tout le reste reste fermé par la politique PROTECTED.
RISK	Dépend entièrement du préréglage — consulte les notes de risque par préréglage dans le Security and Hardening Guide avant d'ouvrir quoi que ce soit au-delà de SSH ou HTTP/HTTPS sur une machine accessible depuis internet.

Rien n'a d'effet tant que tu n'appuies pas sur **SAVE & APPLY** — les changements sont d'abord mis en attente, pour que tu puisses vérifier exactement ce que tu es sur le point d'ouvrir avant de confirmer.

```
▸ ADVANCED

Past this point: config file paths, raw commands, and what the panel is doing underneath.
```

5.2 Advanced

La référence technique complète — l'ensemble de règles par défaut de `nftables.conf` expliqué règle par règle, chacun des 21 préréglages avec son exposition spécifique et son atténuation, des recettes pour limiter l'accès au seul LAN, les équivalents en ligne de commande `nft`, et la résolution d'une session SSH bloquée à l'extérieur — vit dans le **Chapitre 2 du Security and Hardening Guide** (`/usr/share/doc/odyssey/` sur ton système installé). Ce manuel ne le duplique pas délibérément ; considère ce chapitre comme la suite de cette section.

```
# la commande qui vaut la peine de connaître tout de suite :
$ sudo nft list ruleset
# montre exactement ce qui est chargé dans le noyau, indépendamment du panneau
```

CHAPTER 6

AppArmor

Cinq onglets : **STATUS**, **PROFILES**, **COMMUNITY**, **VIOLATIONS**, **EDITOR**. Même renvoi que Firewall : la référence approfondie — ce qu'est vraiment le contrôle d'accès obligatoire, le flux complet de confinement, un exemple pratique — vit dans le Security and Hardening Guide. Voici le guide visuel.

6.1 Guide visuel — STATUS

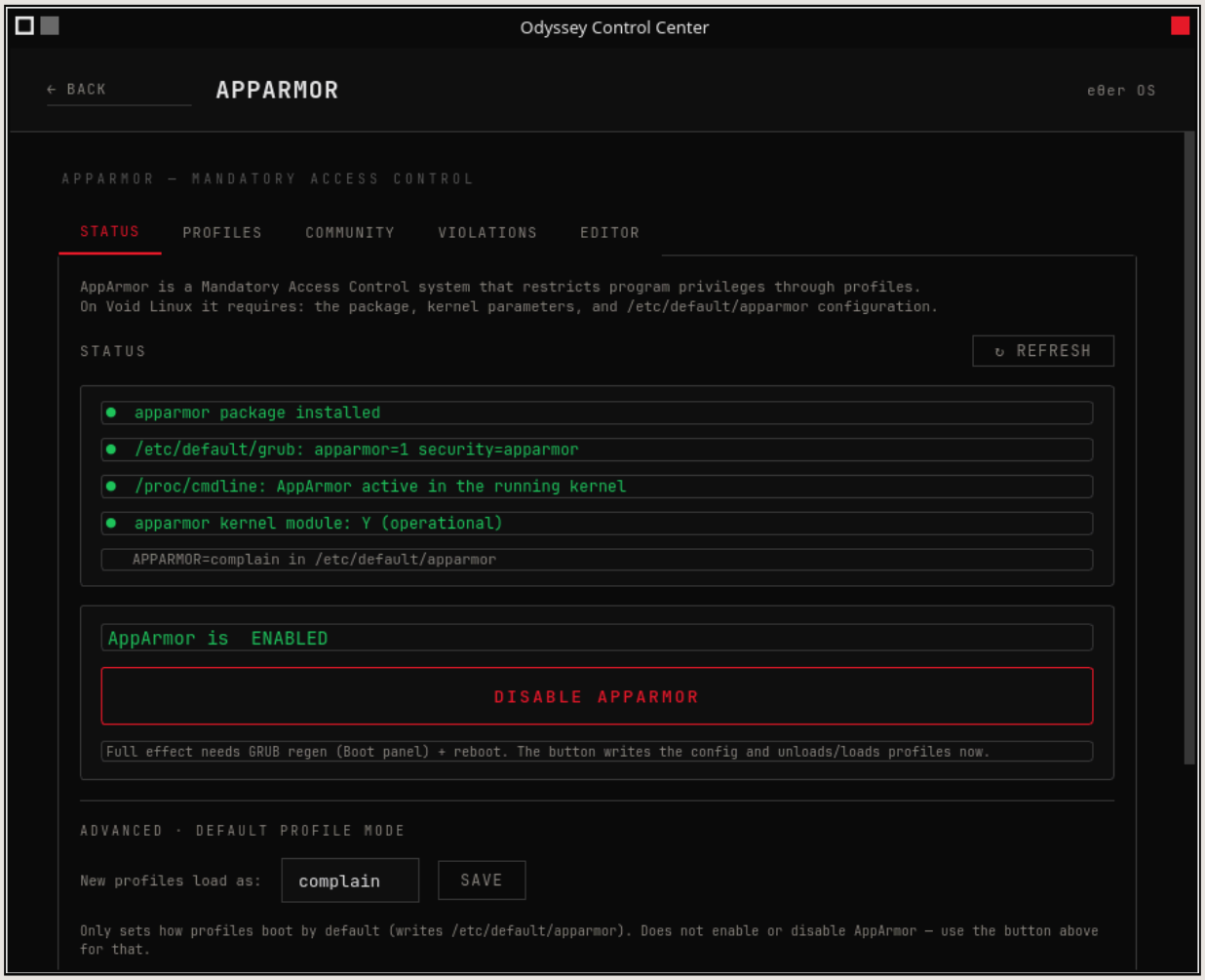

FIG · Onglet STATUS : quatre coches vertes (paquet installé, paramètres GRUB, noyau en cours d'exécution, module noyau), « AppArmor is ENABLED » avec un bouton rouge DISABLE APPARMOR, et le mode de profil par défaut réglé sur complain.

Quatre vérifications confirment que le framework fonctionne vraiment, de haut en bas : le paquet `apparmor` est installé, les paramètres noyau de GRUB sont définis, le noyau **en cours d'exécution** a vraiment démarré avec eux (c'est celle qui peut être en retard si tu n'as pas redémarré après les avoir activés), et le module noyau lui-même est opérationnel. En dessous : un gros bouton d'activation/

désactivation, et **ADVANCED · DEFAULT PROFILE MODE** — si les profils fraîchement chargés démarrent en complain ou enforce.

WHAT	AppArmor → STATUS → confirme que les quatre vérifications sont vertes.
WHY	Si une vérification est rouge ou orange, rien d'autre dans ce panneau ne peut fonctionner correctement — c'est toujours la première chose à regarder quand quelque chose avec AppArmor semble aller de travers.
RESULT	Certitude que le framework est réellement actif, pas juste installé.
RISK	Aucun — vérification en lecture seule.

6.2 Guide visuel — PROFILES

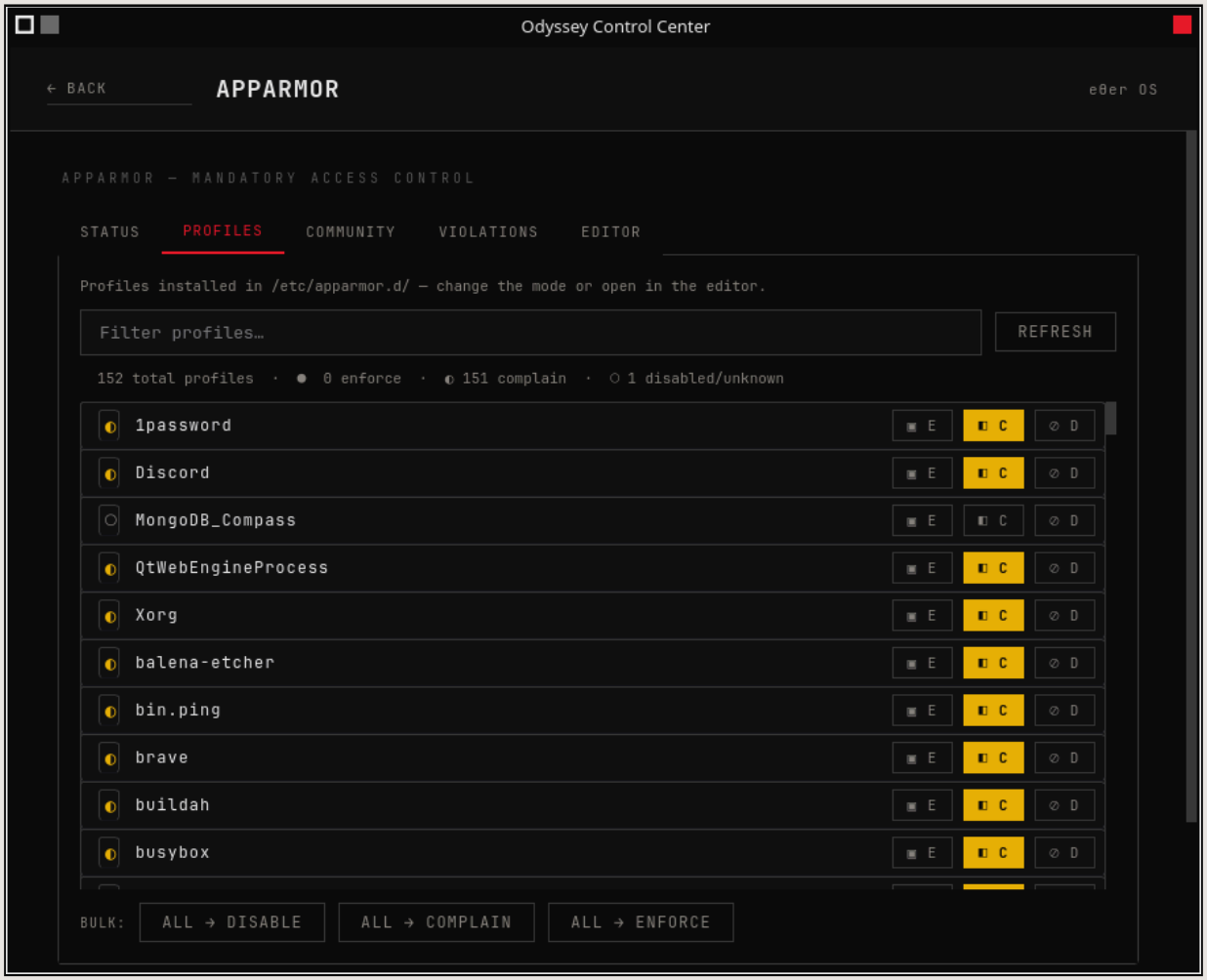

FIG · Onglet PROFILES : 152 profils au total, 0 enforce, 151 complain, 1 disabled/unknown — une liste filtrable avec 1password, Discord, MongoDB_Compass et d'autres, chacun avec des boutons E / C / D, et en bas ALL→DISABLE / ALL→COMPLAIN / ALL→ENFORCE.

Chaque profil chargé, une ligne chacun, avec un champ de filtre et un décompte en haut (dans la capture : 152 profils, tous sauf un en mode complain — le défaut d'Odyssey). Trois boutons par ligne :

BOUTON	EFFET
E	Passe ce profil en enforce — les violations sont bloquées, pas seulement enregistrées

BOUTON	EFFET
C	Passe en complain — enregistre les violations, ne bloque rien
D	Désactive complètement le profil — le programme tourne sans confinement

Les trois boutons **BULK** en bas appliquent le même changement à chaque profil de la liste en une fois — utile juste après avoir installé un lot depuis COMMUNITY, mais traite ALL→ENFORCE avec une prudence particulière ; voir le bloc d'action dans la section suivante.

6.3 Guide visuel — COMMUNITY

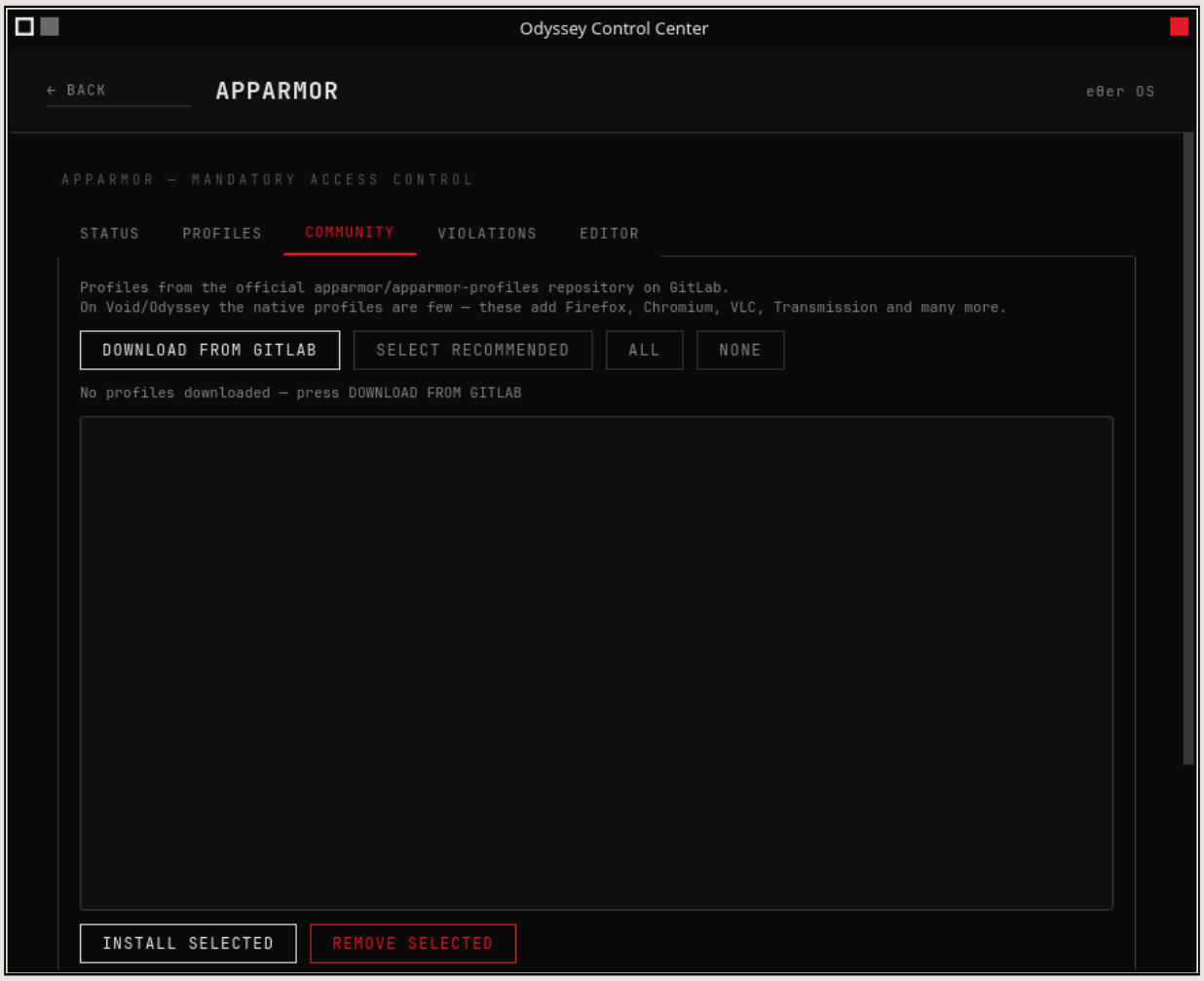

FIG · Onglet COMMUNITY : « No profiles downloaded — press DOWNLOAD FROM GITLAB », avec les boutons DOWNLOAD FROM GITLAB, SELECT RECOMMENDED, ALL et NONE, et INSTALL SELECTED / REMOVE SELECTED en bas.

AppArmor de base confine surtout des composants système — les applications de bureau qui t'inquiètent vraiment (navigateurs, apps de chat, lecteurs multimédias) ont besoin de profils de la communauté. Cet onglet les récupère directement depuis le dépôt officiel `apparmor/apparmor-profiles` sur GitLab.

1 DOWNLOAD FROM GITLAB

Récupère la liste actuelle des profils disponibles — rien n'est encore installé, cela ne fait que peupler la liste.

2 SELECT RECOMMENDED (ou ALL / NONE)

Coche un ensemble par défaut sensé pour un bureau — navigateurs de la famille Firefox, Chromium, VLC, Transmission et similaires.

3 INSTALL SELECTED

Installe les profils cochés — ils se chargent en mode complain, comme tout le reste, prêts pour le cycle observer-puis-appliquer ci-dessous.

WHAT	AppArmor → COMMUNITY → DOWNLOAD FROM GITLAB → SELECT RECOMMENDED (ou choisis individuellement) → INSTALL SELECTED.
WHY	Les applications les plus exposées aux entrées non fiables — un navigateur qui affiche une page malveillante, un lecteur multimédia qui ouvre un fichier trafiqué — sont exactement celles qu'un profil protège le mieux.
RESULT	De nouveaux profils apparaissent dans PROFILES, en mode complain, enregistrant ce qu'ils auraient bloqué sans encore rien bloquer.
RISK	Aucun à l'installation — rien n'est appliqué tant que tu ne le décides pas dans PROFILES.

6.4 Guide visuel — VIOLATIONS

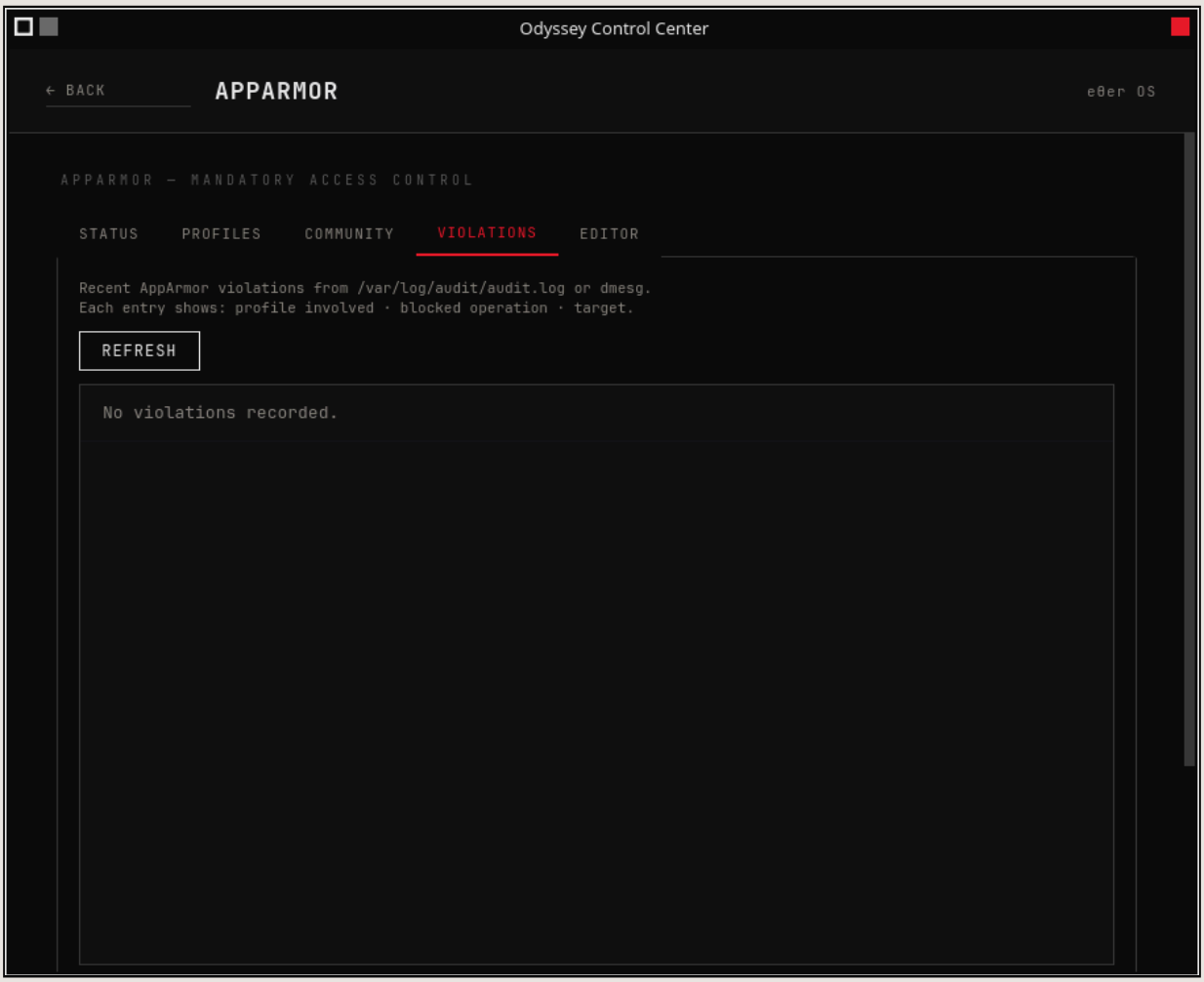

FIG · Onglet VIOLATIONS : « No violations recorded », avec un bouton REFRESH et une note indiquant que les entrées proviennent de /var/log/audit/audit.log ou dmesg.

Cet onglet lit le journal d'audit d'AppArmor du système et montre ce qu'un profil **aurait** bloqué (en mode complain) ou **a** bloqué (en mode enforce) — chaque entrée indique le profil, l'opération bloquée, et le fichier ou la ressource cible.

UNE LISTE VIDE EST LA LECTURE ATTENDUE SUR UN SYSTÈME SAIN

La capture montre exactement ce que tu verras typiquement : aucune violation. Ce n'est pas un bug ni un signe que le panneau ne fonctionne pas — cela signifie que chaque programme profilé est resté dans les limites autorisées depuis la dernière vérification du journal. Les violations apparaissent ici au moment où quelque chose de profilé essaie de faire ce que son profil n'autorise pas ; va utiliser le programme en question, reviens, et fais REFRESH.

Une fois que des entrées apparaissent, un double-clic sur l'une d'elles est le chemin rapide vers EDITOR avec ce profil exact déjà ouvert — voir ci-dessous.

6.5 Guide visuel — EDITOR

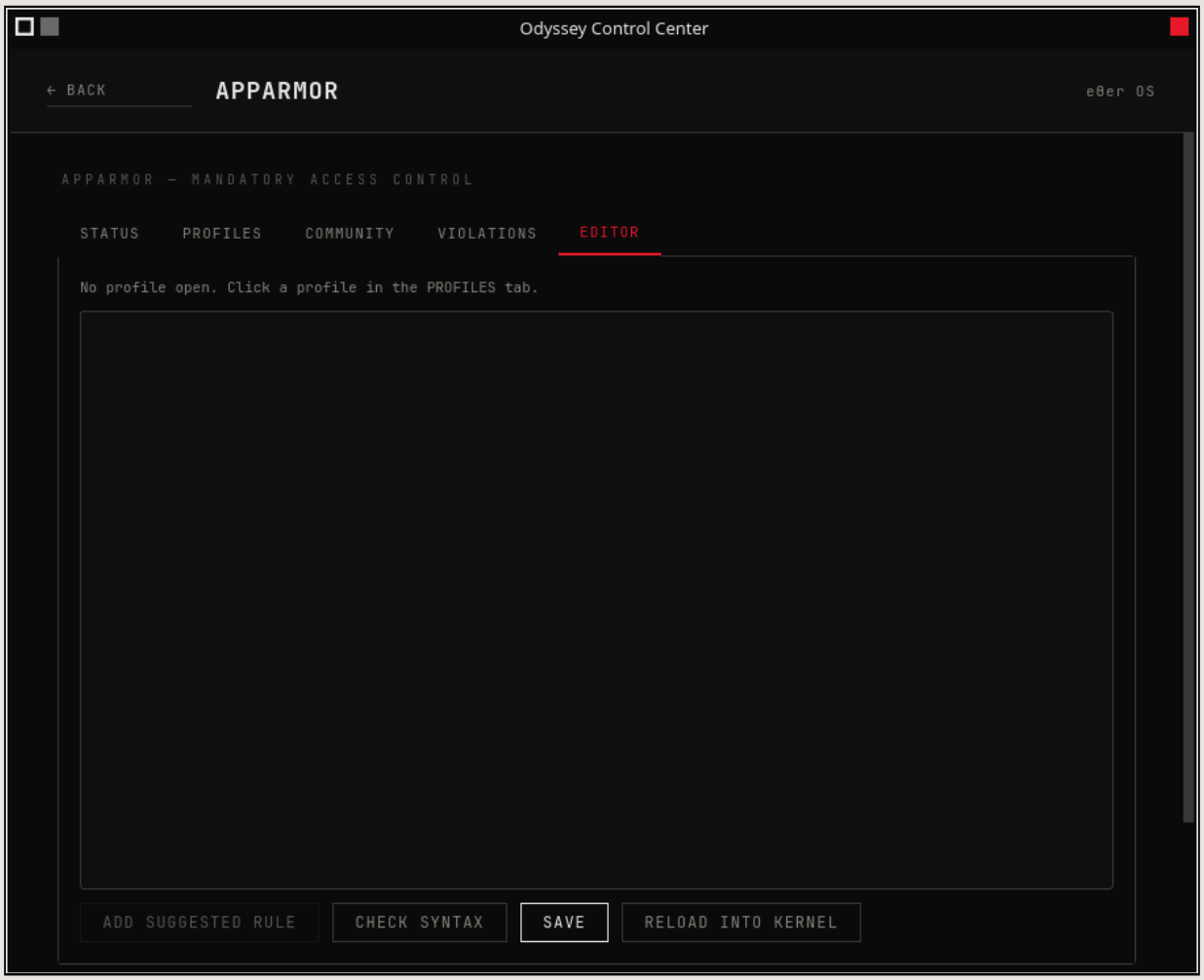

FIG · Onglet EDITOR : « No profile open. Click a profile in the PROFILES tab », une zone de texte vide, et quatre boutons — ADD SUGGESTED RULE, CHECK SYNTAX, SAVE, RELOAD INTO KERNEL — tous sauf SAVE désactivés.

L'éditeur s'ouvre délibérément **vide**, comme montré — il ne modifie qu'un seul profil à la fois, et tu choisis lequel depuis PROFILES (clique sur une ligne) ou depuis VIOLATIONS (double-clic sur une entrée). Cela évite l'erreur facile de modifier le mauvais fichier par accident.

BOUTON	CE QU'IL FAIT
ADD SUGGESTED RULE	Actif seulement si tu arrives depuis une entrée VIOLATIONS — insère la ligne de règle exacte qui aurait permis l'action enregistrée
CHECK SYNTAX	Valide le profil sans le charger — attrape une faute de frappe avant qu'elle compte
SAVE	Écrit tes modifications dans le fichier du profil sur disque
RELOAD INTO KERNEL	Rend une modification enregistrée effective immédiatement, sans attendre le prochain démarrage ou redémarrage du service

WHAT	PROFILES → clique sur le nom d'un profil → EDITOR l'ouvre → fais une modification → CHECK SYNTAX → SAVE → RELOAD INTO KERNEL.
WHY	C'est le cycle complet de confinement en une ligne : observe en mode complain (VIOLATIONS), corrige ce qui manque vraiment (EDITOR), seulement ensuite passe ce profil en enforce (PROFILES).
RESULT	Le profil se comporte exactement comme modifié, en direct, sans redémarrage.
RISK	Modifier un profil incorrectement peut faire bloquer plus tard quelque chose de légitime sous enforce — CHECK SYNTAX attrape les règles malformées avant SAVE, mais pas les erreurs logiques. Si un profil que tu as modifié commence à poser problème sous enforce, remets-le en complain depuis PROFILES et revérifie VIOLATIONS.

▸ **ADVANCED**

Past this point: config file paths, raw commands, and what the panel is doing underneath.

6.6 Advanced

ÉQUIVALENTS EN LIGNE DE COMMANDE

```
$ sudo aa-status
# les mêmes décomptes que l'en-tête de l'onglet PROFILES, depuis le terminal
$ sudo aa-enforce /etc/apparmor.d/PROFILE_FILE
$ sudo aa-complain /etc/apparmor.d/PROFILE_FILE
$ sudo aa-disable /etc/apparmor.d/PROFILE_FILE
# équivalent à RELOAD INTO KERNEL après une modification manuelle
$ sudo apparmor_parser -r /etc/apparmor.d/PROFILE_FILE
```

LIRE LE JOURNAL D'AUDIT BRUT

```
$ sudo grep apparmor /var/log/audit/audit.log | tail -20
# ou, si aucun daemon d'audit dédié n'est en cours d'exécution :
$ sudo dmesg | grep -i apparmor
```

Champs utiles à connaître en lisant une ligne à la main : `apparmor="DENIED"` (enforce) ou `apparmor="ALLOWED"` (complain, enregistré seulement), `operation=` (ce qui a été tenté), `profile=` (quel profil a décidé — correspond à ce qu'ouvre EDITOR), `name=` (le chemin ou la ressource), `comm=` (le processus).

POURQUOI ALL→ENFORCE EN MASSE EST RISQUÉ

Faire passer un profil en enforce individuellement après que son journal VIOLATIONS est resté silencieux pendant un moment est le schéma sûr — tu sais, profil par profil, que l'usage normal ne le déclenche pas. ALL→ENFORCE saute cette période d'observation pour tous les profils à la fois, y compris ceux que tu viens peut-être d'installer depuis COMMUNITY et jamais vraiment utilisés. Un profil appliqué avant d'avoir été vécu peut bloquer une fonction légitime au moment même où tu l'utilises pour la première fois. Préfère le cycle un par un ; réserve les actions en masse pour ALL→COMPLAIN (toujours sûr — ne fait qu'assouplir) ou ALL→DISABLE pendant le dépannage.

FICHIERS TOUCHÉS PAR CE PANNEAU

CHEMIN	RÔLE
`/etc/apparmor.d/`	Un fichier de profil par programme confiné — ce que liste PROFILES et ouvre EDITOR
`/etc/default/apparmor`	Mode par défaut au démarrage (`APPARMOR=complain`) et l'état enable/disable
`/etc/default/grub`	Les paramètres noyau `apparmor=1 security=apparmor` que STATUS vérifie

CHAPTER 7

Privacy

Deux outils indépendants, chacun avec son propre flux en trois étapes install → enable → running, et un pont entre les deux une fois que les deux sont en cours d'exécution.

7.1 Guide visuel — SearXNG, étape par étape

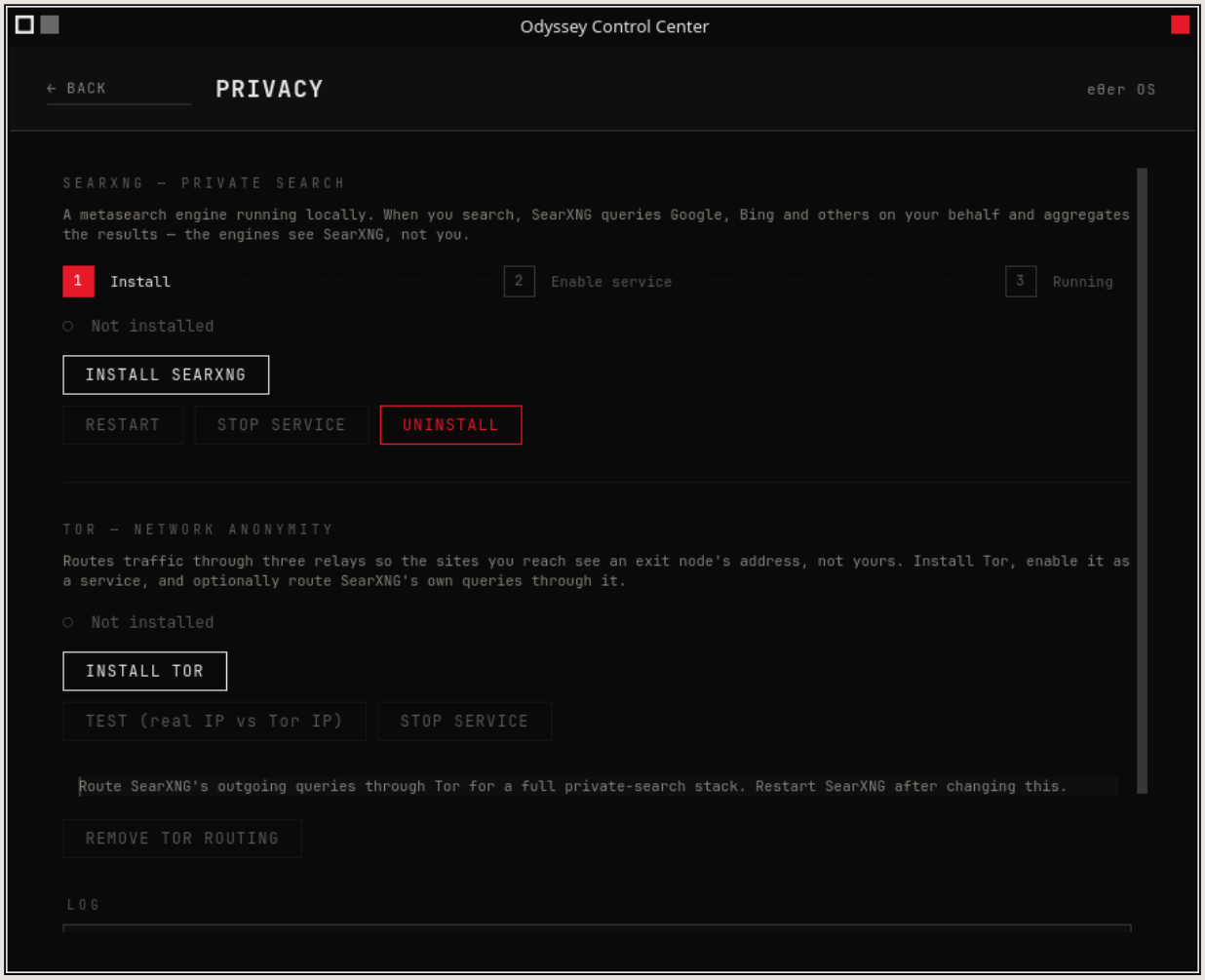

FIG · Le panneau au repos : le suivi de SearXNG à l'étape 1 « Install », statut « Not installed », un seul bouton INSTALL SEARXNG ; le suivi de Tor affiche aussi « Not installed » avec INSTALL TOR en dessous.

SearXNG est un métamoteur de recherche qui tourne localement sur ta propre machine — quand tu cherches, c'est SearXNG qui interroge Google, Bing et les autres en ton nom, donc ces moteurs voient SearXNG faire la requête, jamais toi directement.

1 **INSTALL SEARXNG**

Clique. Le journal montre l'installation réelle en cours — clonage du code source, création d'un utilisateur système, configuration d'un environnement virtuel Python.

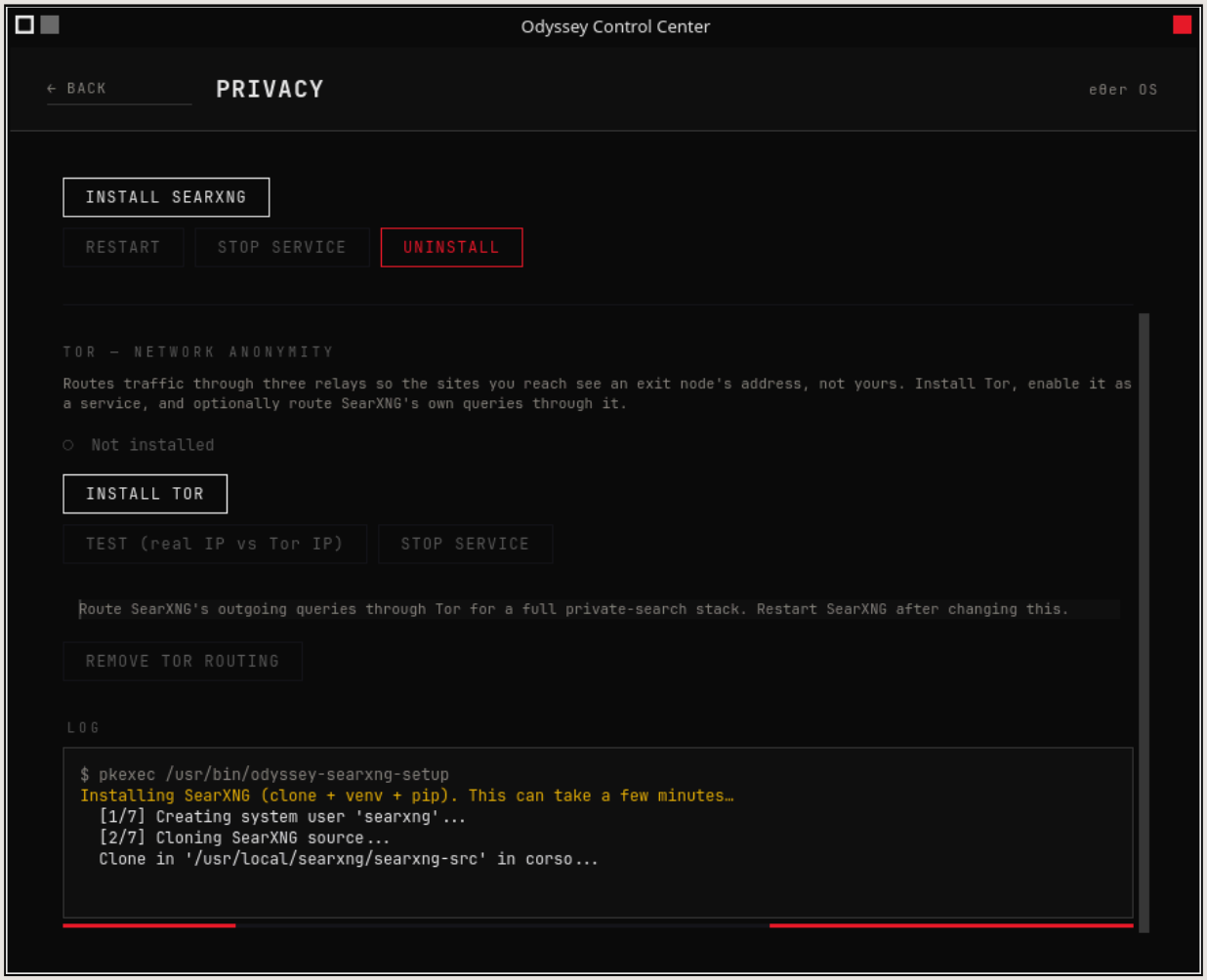

FIG · À mi-installation : le journal transmettant une sortie réelle — « Installing SearXNG (clone + venv + pip). This can take a few minutes… » avec les étapes 1/7 et 2/7 visibles.

2 ENABLE SERVICE

Une fois installé, le suivi passe à l'étape 2 et la ligne de statut devient jaune : « Installed — service not enabled ». Un seul bouton maintenant : ENABLE SERVICE.

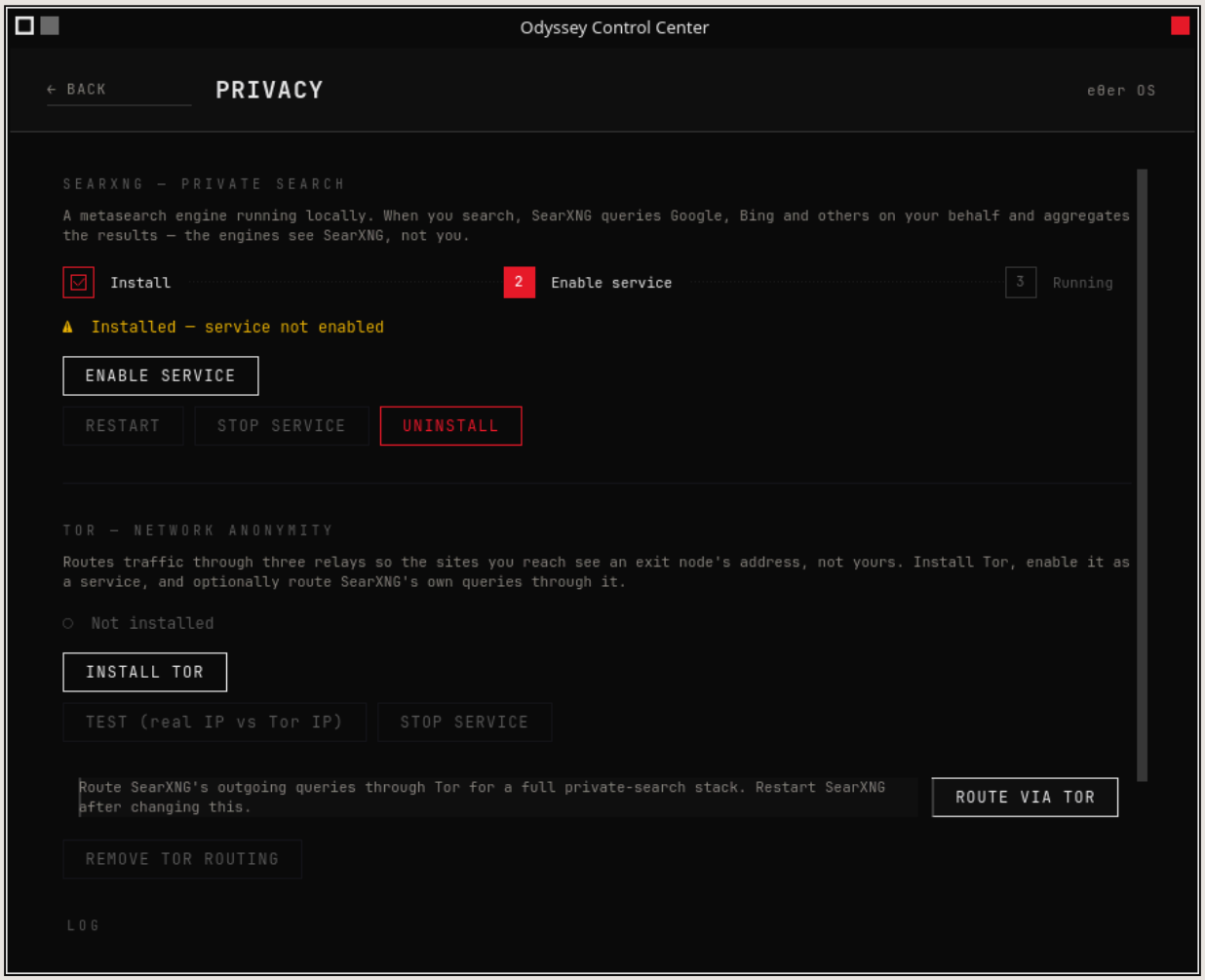

FIG · Étape 2 active : « Installed — service not enabled », ENABLE SERVICE comme seul bouton actif, RESTART/STOP SERVICE/UNINSTALL encore désactivés en dessous.

3 Running

Après l'activation, le suivi atteint l'étape 3 et toute la rangée de contrôle s'active. Une fois en cours d'exécution, **OPEN IN BROWSER** t'y amène directement — SearXNG écoute sur `http://localhost:8888` par défaut.

WHAT	Privacy → INSTALL SEARXNG → ENABLE SERVICE → OPEN IN BROWSER.
WHY	Chaque moteur de recherche commercial construit un profil à partir de tes requêtes au fil du temps ; une instance de métarecherche locale signifie que toi seul vois jamais la requête non agrégée.
RESULT	Un moteur de recherche privé tournant sur ta propre machine, accessible dans ton navigateur, interrogeant les grands moteurs en ton nom sans t'exposer directement à eux.
RISK	Faible — c'est un service web local comme un autre. RESTART s'il se comporte mal, STOP SERVICE pour le mettre en pause sans le supprimer, UNINSTALL pour le supprimer complètement.

7.2 Guide visuel — Tor, étape par étape

Même forme en trois étapes, suivi indépendant, juste en dessous de la section SearXNG.

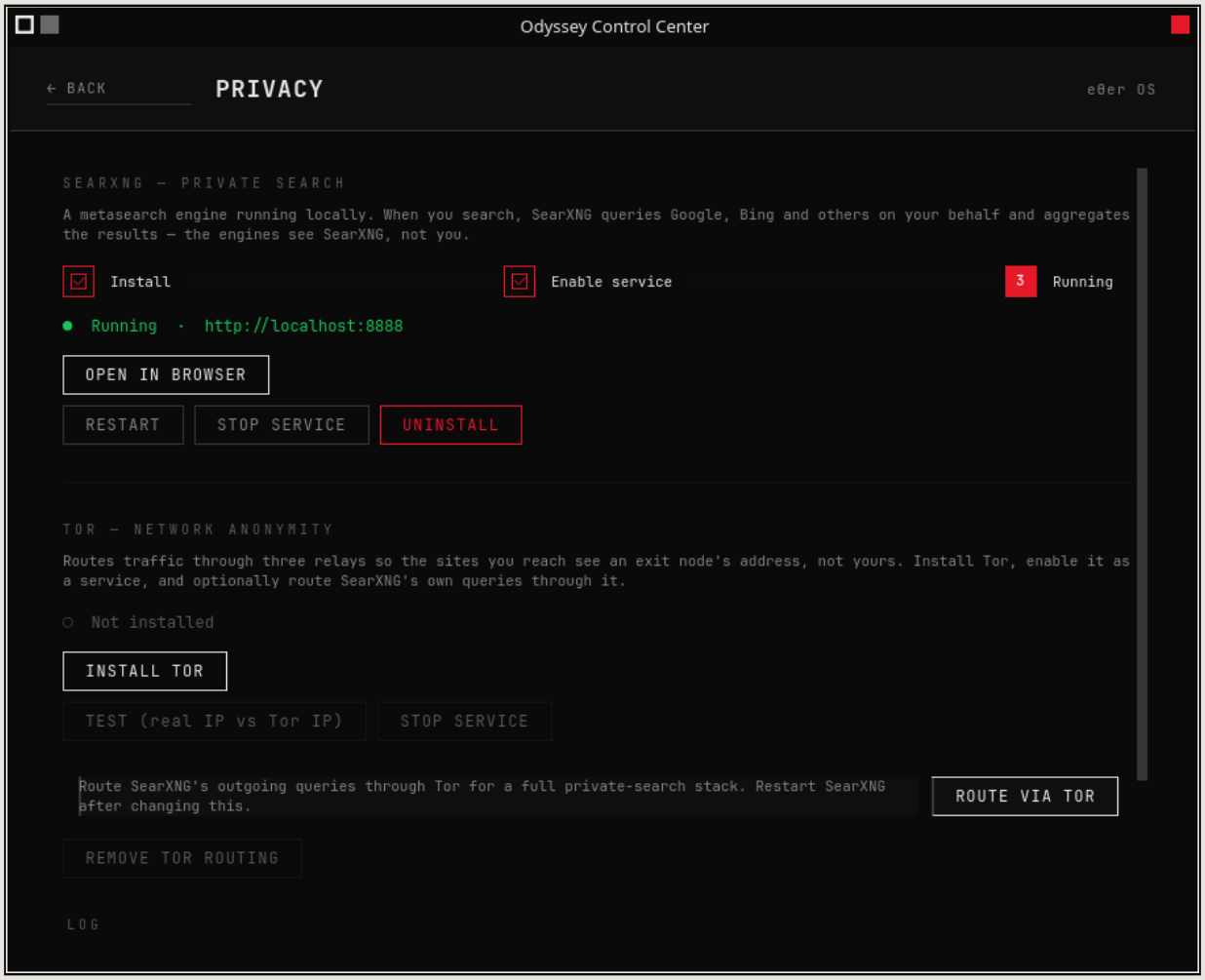

FIG · SearXNG entièrement en cours d'exécution en haut ; la section Tor en dessous encore sur « Not installed » avec INSTALL TOR comme seul bouton actif.

Tor fait passer ton trafic par trois relais avant qu'il n'atteigne sa destination, donc le site que tu visites voit l'adresse du relais de sortie, jamais la tienne.

1. **INSTALL TOR**

   Le journal confirme l'installation du paquet et montre le service lié dans runit.

2. **ENABLE SERVICE**

   Le statut passe à « Installed — service not enabled » — même schéma que SearXNG.

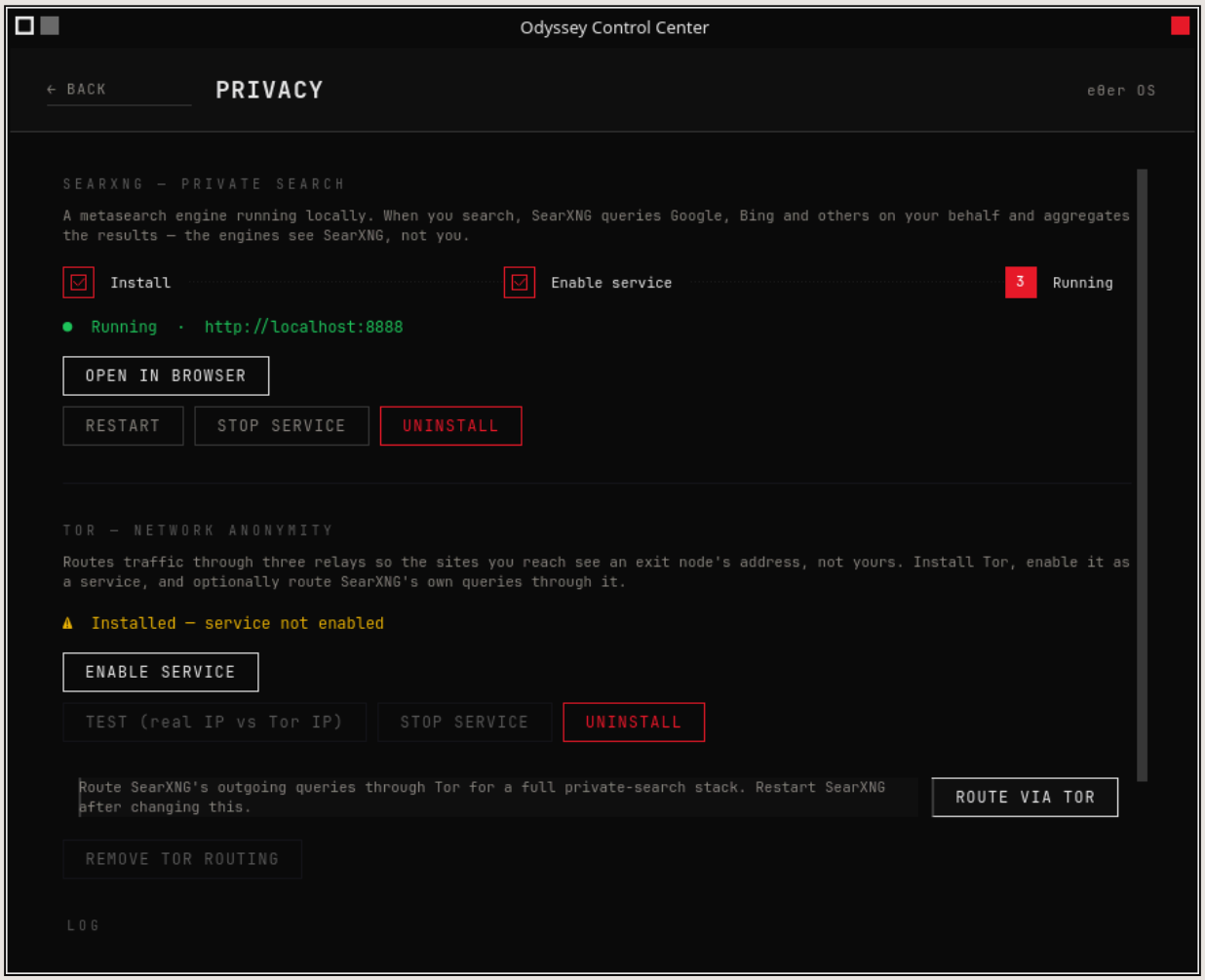

FIG · Tor à l'étape 2 : « Installed — service not enabled », ENABLE SERVICE actif, TEST et STOP SERVICE encore désactivés.

3 Running, et vérifie-le

Une fois activé, un bouton dédié **TEST (real IP vs Tor IP)** apparaît — utilise-le. Il compare ton adresse réelle à celle que Tor présente, côte à côte, pour que tu aies une vraie preuve que ça fonctionne plutôt qu'un simple voyant de statut auquel il faut juste croire.

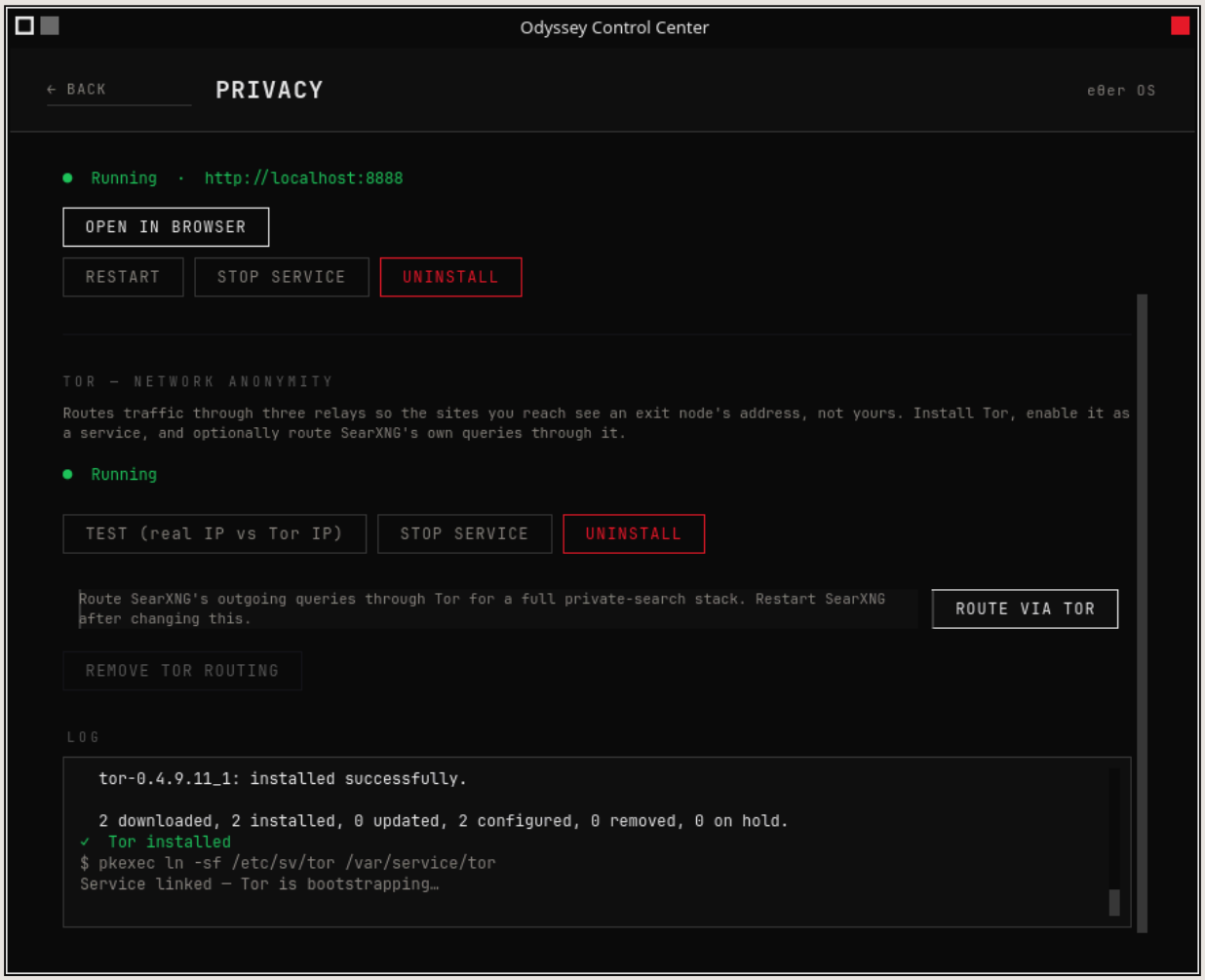

FIG · Les deux sections en vert et en cours d'exécution : SearXNG sur localhost:8888 avec OPEN IN BROWSER, Tor en cours d'exécution avec TEST / STOP SERVICE / UNINSTALL tous actifs.

WHAT	Privacy → INSTALL TOR → ENABLE SERVICE → TEST (real IP vs Tor IP) pour confirmer.
WHY	Faire passer la navigation sensible par Tor cache ton adresse aux sites que tu visites — TEST te donne une preuve, pas juste une affirmation.
RESULT	Un service Tor en cours d'exécution à travers lequel d'autres applications peuvent s'acheminer ; TEST montre deux adresses réellement différentes si ça fonctionne.
RISK	Le trafic à travers Tor est nettement plus lent — c'est le prix de l'anonymat. STOP SERVICE le met en pause ; UNINSTALL le supprime.

7.3 Guide visuel — les combiner

Une fois que les deux sont en cours d'exécution, un avis en ligne apparaît proposant d'acheminer aussi les propres requêtes de SearXNG via Tor — pas seulement tes requêtes vers SearXNG, mais les requêtes de SearXNG vers Google et Bing en ton nom.

WHAT	Privacy → l'avis « Route SearXNG's outgoing queries through Tor » → ROUTE VIA TOR → RESTART sur la section SearXNG pour que ça prenne effet.
WHY	Sans cela, SearXNG lui-même continue d'interroger les grands moteurs de recherche directement — privé de ton côté, visible du leur. Acheminer le propre trafic de SearXNG via Tor comble cet écart pour une pile de recherche entièrement privée.
RESULT	Les requêtes sortantes de SearXNG, pas seulement tes requêtes vers lui, passent par Tor.
RISK	Les résultats se chargent nettement plus lentement, pour la même raison que tout trafic acheminé via Tor. REMOVE TOR ROUTING l'annule, encore suivi d'un redémarrage.

▸ ADVANCED

Past this point: config file paths, raw commands, and what the panel is doing underneath.

7.4 Advanced

CE QU'EXÉCUTE RÉELLEMENT INSTALL SEARXNG

```
$ pkexec /usr/bin/odyssey-searxng-setup
# crée un utilisateur système dédié 'searxng',
# clone le code source dans /usr/local/searxng/searxng-src,
# construit un venv Python et installe les dépendances avec pip
```

SERVICES ET PORTS, À LA MAIN

```
# le service runit de SearXNG
$ sudo sv status searxng
$ sudo sv restart searxng
# le service runit de Tor — l'étape d'activation du panneau est littéralement ceci
$ sudo ln -sf /etc/sv/tor /var/service/tor
$ sudo sv status tor
# vérifie que Tor achemine bien le trafic
$ curl -socks5 localhost:9050 https://check.torproject.org/api/ip
```

ACHEMINER SEARXNG VIA TOR À LA MAIN

Le proxy SOCKS de Tor écoute sur `127.0.0.1:9050` par défaut. Acheminer SearXNG à travers lui signifie pointer son client HTTP sortant vers ce proxy — le réglage vit dans le propre fichier `settings.yml` de SearXNG, sous la configuration des requêtes sortantes, plutôt que dans une variable de proxy à l'échelle du système. Le bouton ROUTE VIA TOR du panneau modifie directement ce fichier et redémarre le service ; le faire à la main signifie localiser et modifier le même bloc `outgoing:` dans `/usr/local/searxng/searxng-src/searx/settings.yml`.

POURQUOI SEARXNG A BESOIN D'UN REDÉMARRAGE APRÈS DES CHANGEMENTS D'ACHEMINEMENT

SearXNG lit sa configuration de proxy une seule fois, au démarrage — ce n'est pas un réglage rechargeable à chaud. C'est vrai que tu le changes via le panneau ou à la main, c'est pourquoi les deux chemins se terminent par la même étape RESTART.

FICHIERS TOUCHÉS PAR CE PANNEAU

CHEMIN	RÔLE
`/usr/local/searxng/searxng-src/`	L'installation même de SearXNG — code source, venv, configuration
`/etc/sv/searxng`, `/etc/sv/tor`	Définitions des services runit
`/var/service/`	Où l'activation d'un service le lie dans l'arbre de supervision en cours d'exécution

CHAPTER 8

Containers

Gestion des conteneurs de bout en bout : configuration du moteur, une liste des conteneurs en cours d'exécution, un catalogue sélectionné pour démarrer, des commandes brutes pour tout ce qui n'y figure pas, et l'usage disque — tout dans un seul panneau défilant.

8.1 Guide visuel — le moteur

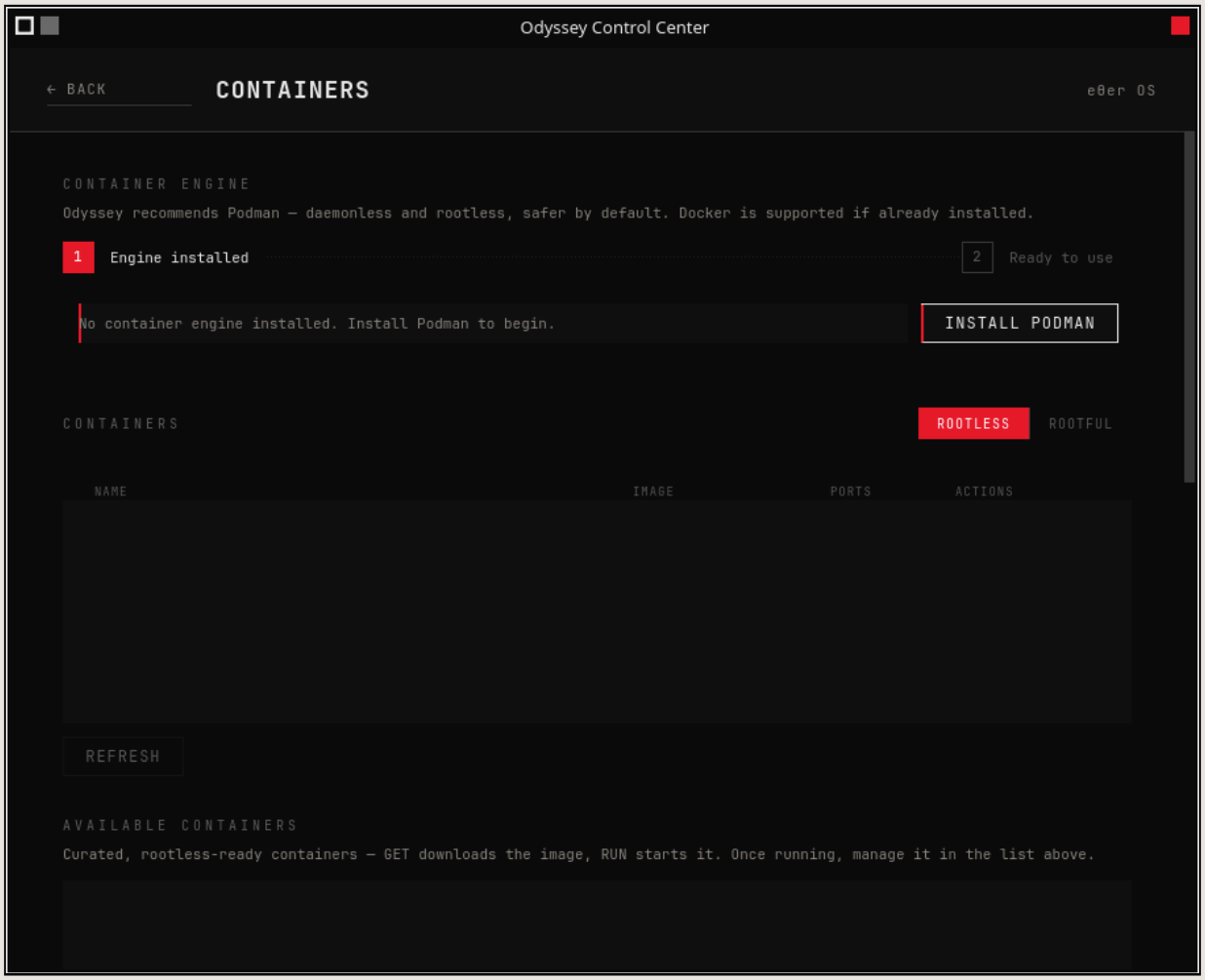

FIG · Étape 1 sur 2 : « No container engine installed. Install Podman to begin », avec INSTALL PODMAN comme seule action active. Les sections Containers list et Available containers en dessous sont visibles mais en état vide.

Odyssey recommande **Podman** — sans démon et rootless, ce qui signifie que les conteneurs tournent sans service en arrière-plan permanent et, par défaut, sans privilèges root. **Docker** est aussi pris en charge si c'est déjà ce que tu utilises.

1 INSTALL PODMAN

Le journal montre l’installation réelle du paquet.

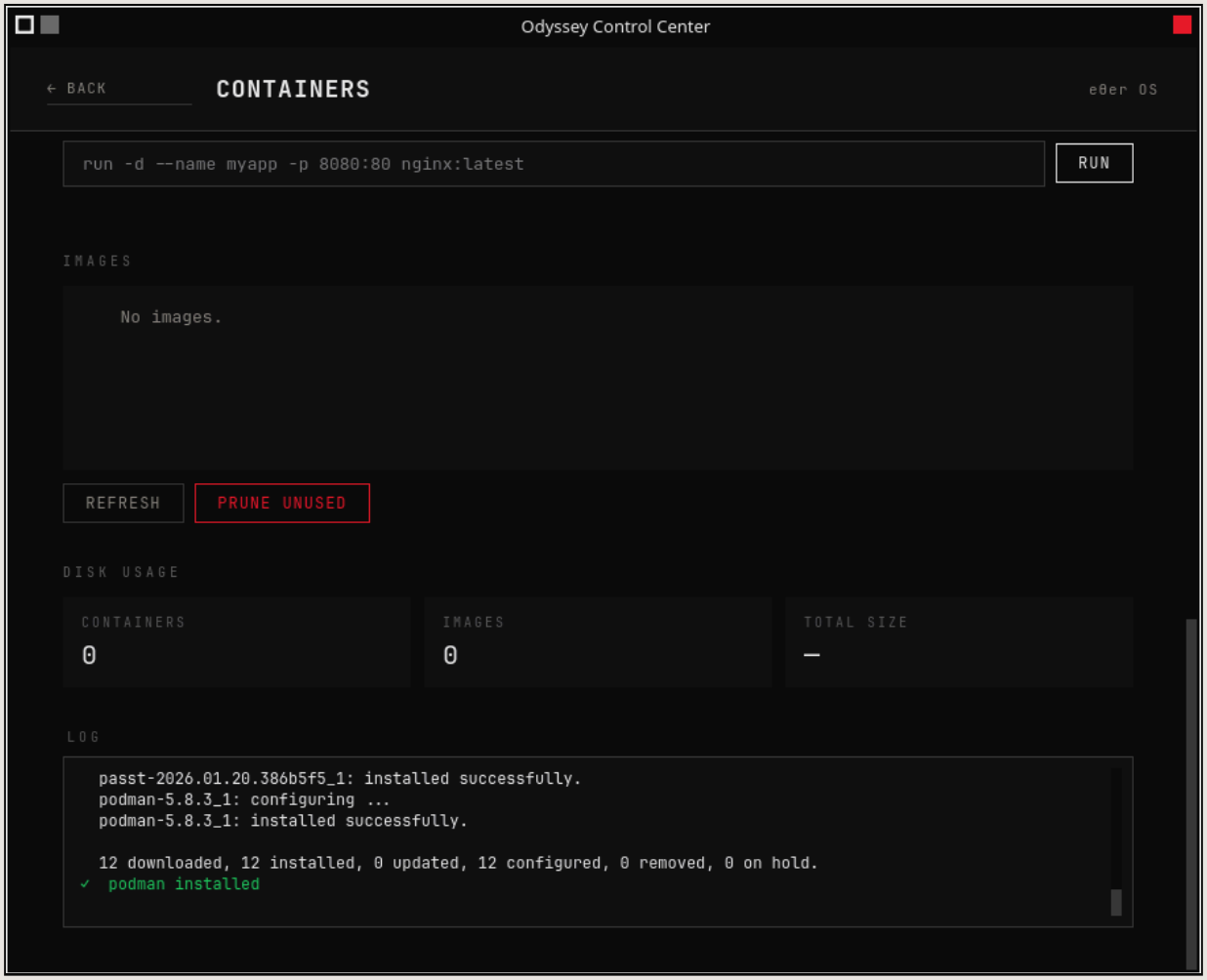

FIG · Après l’installation : « Engine installed » coché en rouge, étape 2 « Ready to use », la ligne d’information indiquant « podman version 5.8.3 · Podman installed · Docker not installed », et un bouton INSTALL DOCKER TOO pour qui veut aussi Docker en parallèle.

2 Ready to use

Le moteur est actif. **INSTALL DOCKER TOO** reste disponible si tu veux les deux moteurs côte à côte — ils coexistent sans conflit.

8.2 Guide visuel — rootless vs. rootful, et l'état vide

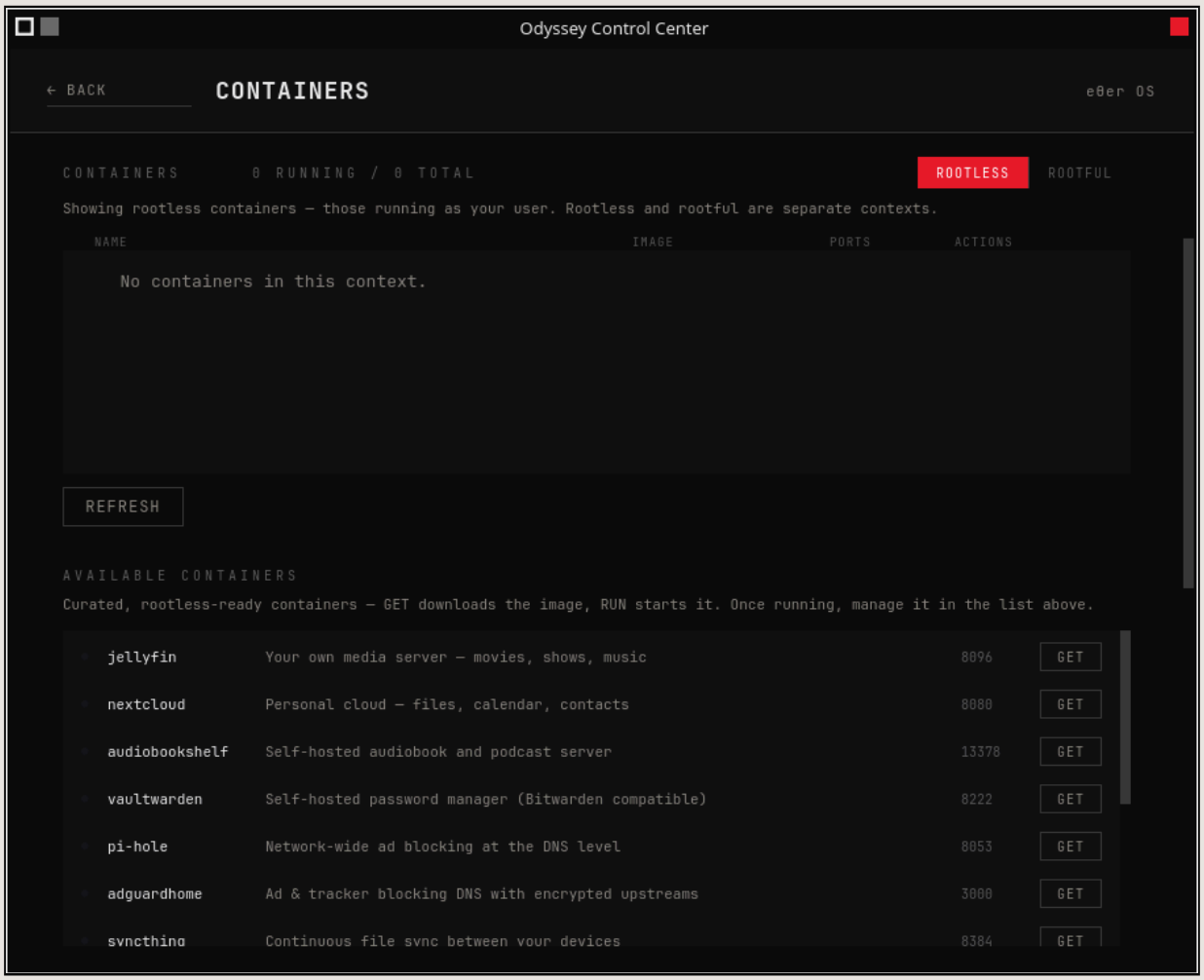

FIG · ROOTLESS sélectionné (mis en évidence en rouge) à côté de ROOTFUL, « 0 RUNNING / 0 TOTAL », et « No containers in this context. » sous les en-têtes de colonne NAME / IMAGE / PORTS / ACTIONS.

L'interrupteur **ROOTLESS / ROOTFUL** en haut change quel contexte tu visualises et gères — les deux sont des espaces de noms de conteneurs complètement séparés, pas un filtre sur une liste combinée. Odyssey affiche ROOTLESS par défaut, et ce choix est délibéré :

> **LEQUEL DES DEUX DEVRAIS-TU UTILISER**
>
> Rootless est le choix le plus sûr et celui vers lequel le panneau t'oriente : un conteneur qui échappe au confinement en mode rootless ne peut de toute façon rien faire que ton propre compte utilisateur ne pourrait déjà faire. Rootful existe pour les cas spécifiques qui en ont besoin — certains ports réseau à numéro bas, certains accès matériels — et devrait être l'exception, pas l'habitude.

8.3 Guide visuel — le catalogue sélectionné, du début à la fin

Fais défiler jusqu'à **Available containers** — une liste déjà prête pour que le panneau ne soit jamais une boîte vide attendant que tu connaisses déjà les noms d'images par cœur. Chaque ligne montre le conteneur, une description en une ligne, et le port qu'il utilise.

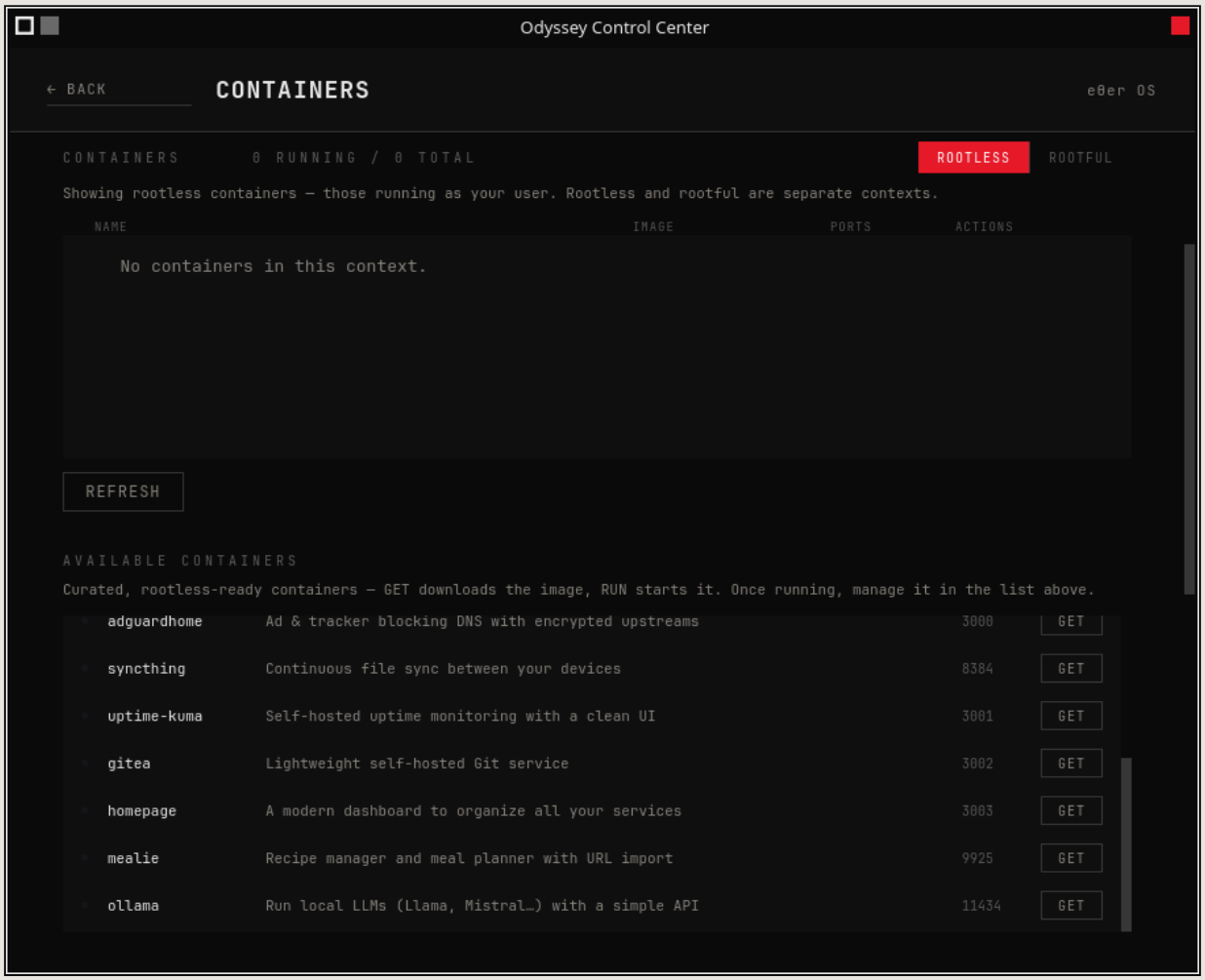

FIG · La liste du catalogue : adguardhome (port 3000), syncthing (8384), uptime-kuma (3001), gitea (3002), homepage (3003) — avec un bouton GET sur chaque ligne.

Ce numéro de port est celui sur lequel tu atteindras le service une fois en cours d'exécution — `homepage` avec `3003`, par exemple, signifie `http://localhost:3003` une fois actif.

1 **GET**

Clique sur GET pour `homepage`. Cela télécharge seulement l'image du conteneur — rien n'est encore en cours d'exécution.

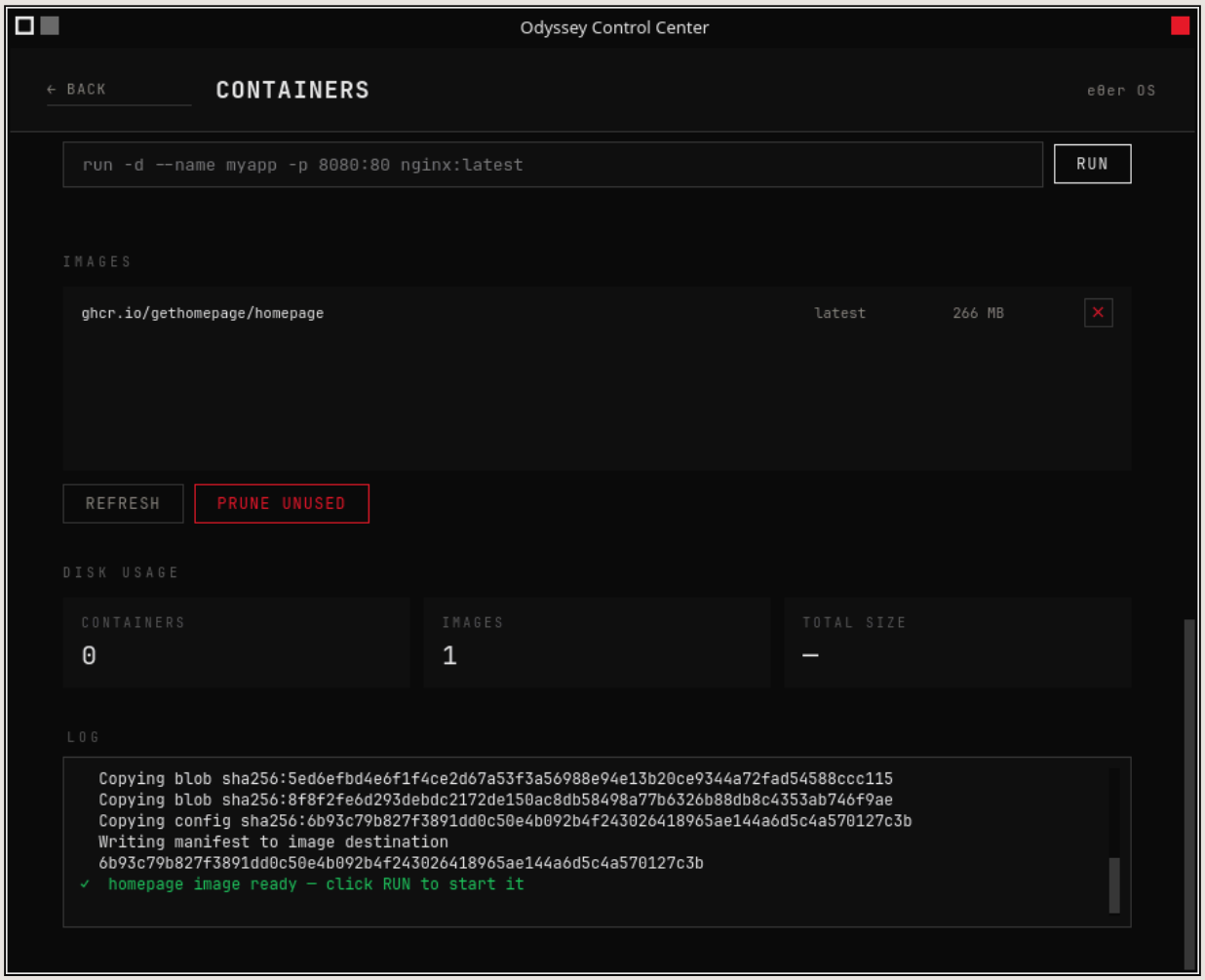

FIG · Le journal transmettant un vrai téléchargement d'image : « Copying blob sha256:5ed6… », « Writing manifest to image destination », se terminant en vert — « homepage image ready — click RUN to start it. »

2 RUN

La ligne du bouton GET affiche maintenant RUN à sa place — clique dessus pour vraiment démarrer le conteneur.

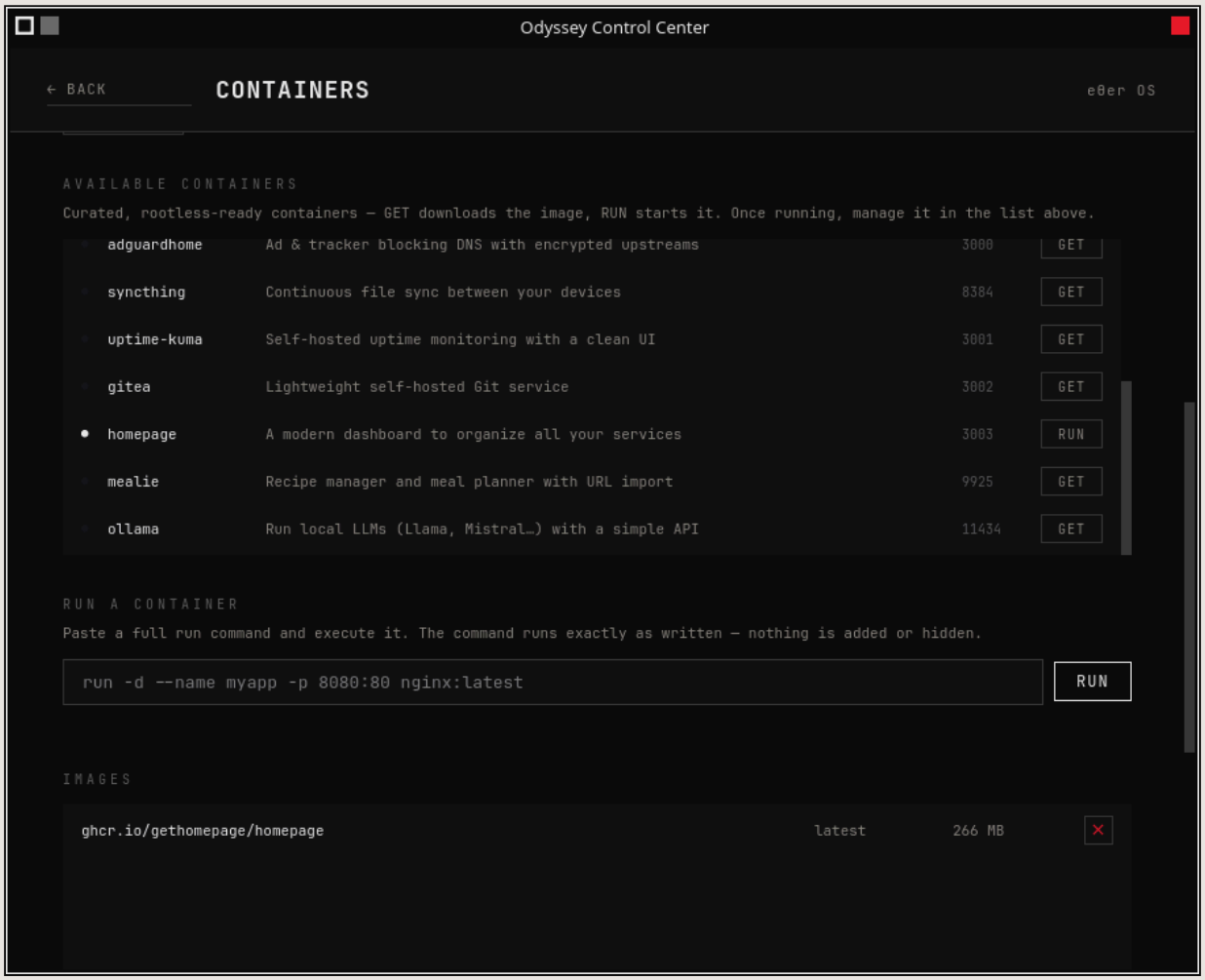

FIG · homepage maintenant listé dans la section CONTAINERS ci-dessus, point vert, chemin d'image ghcr.io/gethomepage/homepage, ports « 0.0.0.0:3003→… », avec sa propre ligne d'icônes d'action ; encore visible en dessous dans Available Containers avec un nouveau bouton RUN pour démarrer une autre instance.

Une fois en cours d'exécution, homepage apparaît à deux endroits en même temps : comme entrée active dans la liste **Containers** en haut (avec son port et ses icônes d'action), et encore dans le catalogue en dessous au cas où tu voudrais démarrer une seconde instance plus tard.

WHAT	Containers → Available containers → GET sur quelque chose du catalogue → RUN une fois téléchargé.
WHY	La division en deux étapes (télécharger, puis démarrer) signifie que tu peux récupérer une image à l'avance sans t'engager à l'exécuter immédiatement — utile sur une connexion lente ou quand tu explores simplement ce qui est disponible.
RESULT	Un conteneur en cours d'exécution, visible dans la liste principale avec son mappage de port, gérable à partir de là.
RISK	Faible — les services auto-hébergés du catalogue sélectionné sont choisis pour être rootless-friendly et à faible risque par défaut. L'exposition est la même que l'exécution de n'importe quel service web : tout port sur lequel il écoute est accessible selon tes réglages Firewall.

8.4 Guide visuel — la liste des conteneurs et ses actions

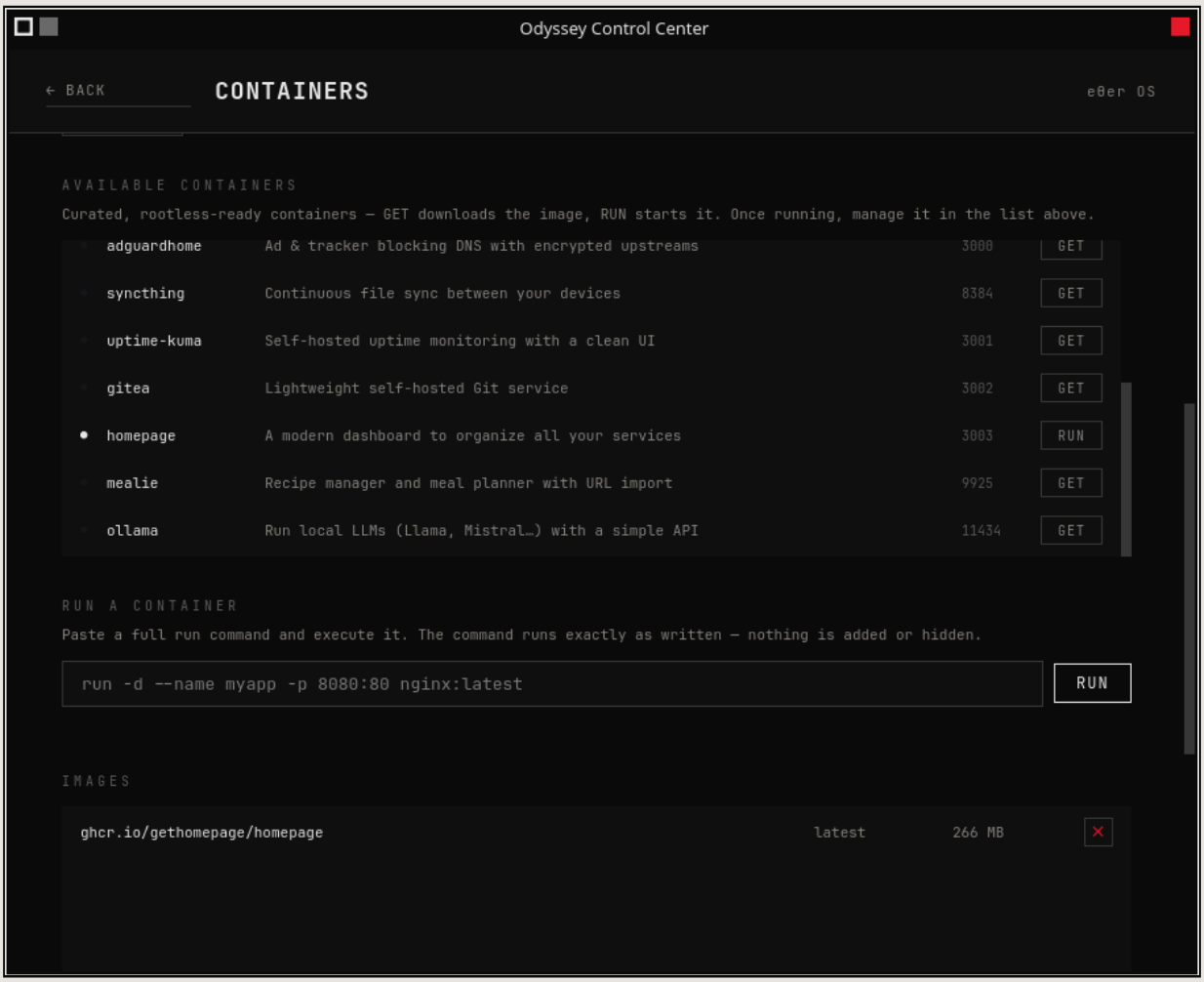

FIG · La ligne de homepage dans la liste Containers, avec ses cinq icônes d'action visibles à droite — celles expliquées ci-dessous.

Chaque ligne d'un conteneur en cours d'exécution se termine par un ensemble d'icônes — survole n'importe laquelle pour voir son étiquette, mais voici ce que fait chacune :

ICÔNE	ACTION
ouvrir (flèche)	Ouvre le service directement dans ton navigateur — le chemin rapide une fois que quelque chose tourne
arrêter / démarrer (carré / triangle)	Arrête le conteneur sans le supprimer, ou le redémarre
redémarrer (flèche circulaire)	L'arrête et le démarre en une seule action — la solution pour « ça tourne mais ça ne répond pas »
journaux (icône de lignes)	Ouvre la propre sortie de journal du conteneur — le premier endroit où regarder si quelque chose ne fonctionne pas
supprimer (×)	Supprime complètement le conteneur — l'image elle-même reste téléchargée, listée séparément sous Images

8.5 Guide visuel — exécuter un conteneur, et l'ouvrir

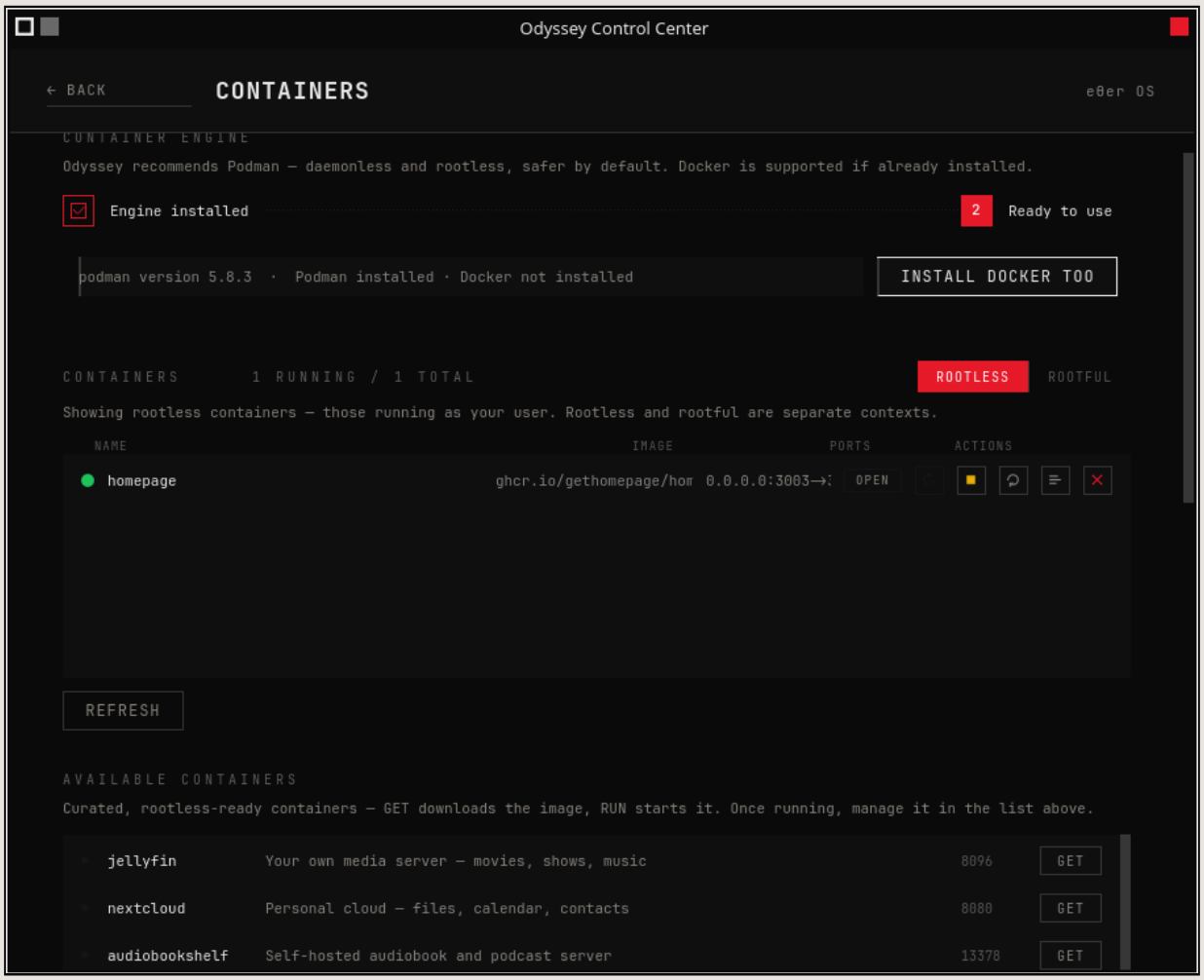

FIG · Le champ « Run a container » avec un exemple de commande en texte d'aide, le bouton RUN, et la section Images en dessous listant ghcr.io/gethomepage/homepage · latest · 266 MB avec un bouton de suppression.

Pour tout ce qui n'est pas dans le catalogue, **Run a container** prend une commande `podman run` (ou `docker run`) complète et l'exécute exactement telle qu'écrite — rien n'est ajouté, rien n'est caché.

WHAT	Containers → « Run a container » → colle une commande run complète (p. ex. `run -d --name myapp -p 8080:80 nginx:latest`) → RUN.
WHY	Le catalogue sélectionné couvre les cas courants ; ceci est l'échappatoire pour une image spécifique, des options spécifiques, des ports spécifiques que le catalogue n'offre pas.
RESULT	Le conteneur démarre exactement comme la commande le spécifie, et rejoint la liste principale Containers une fois en cours d'exécution.
RISK	Ce que fait la commande elle-même — c'est un relais fidèle, pas un bac à sable. Traite-la avec le même soin qu'en tapant la même commande dans un terminal.

En dessous, **Images** liste chaque image téléchargée indépendamment du fait qu'un conteneur en cours d'exécution en découle actuellement, avec **REFRESH** et **PRUNE UNUSED** pour récupérer de l'espace des images que plus personne ne référence.

OUVRIR UN CONTENEUR EN COURS D'EXÉCUTION

Cliquer sur OPEN sur la ligne d'un conteneur en cours d'exécution — la même action d'ouverture depuis la ligne d'icônes ci-dessus — lance ton navigateur par défaut directement vers son adresse et son port, p. ex. `http://localhost:3003` pour `homepage`. Pas besoin de te souvenir toi-même du numéro de port une fois qu'il est en cours d'exécution.

8.6 Usage disque

En bas du panneau, trois totaux en temps réel — conteneurs, images, taille totale — pour toujours savoir ce que te coûtent les conteneurs sur le disque sans ouvrir un terminal.

```
▸ ADVANCED

Past this point: config file paths, raw commands, and what the panel is doing underneath.
```

8.7 Advanced

ÉQUIVALENTS EN LIGNE DE COMMANDE

```
# ce que le panneau appelle « GET »
$ podman pull ghcr.io/gethomepage/homepage
# ce que le panneau appelle « RUN » sur une entrée du catalogue
$ podman run -d -name homepage -p 3003:3000 ghcr.io/gethomepage/homepage
# liste les conteneurs en cours d'exécution — contexte rootless
$ podman ps
# la même chose, rootful
$ sudo podman ps
# journaux, arrêt, redémarrage, suppression — la ligne d'icônes, à la main
$ podman logs -f homepage
$ podman stop homepage
$ podman restart homepage
$ podman rm homepage
# images et usage disque
$ podman images
$ podman system df
$ podman image prune
```

ROOTLESS VS. ROOTFUL, EN DESSOUS

Podman rootless exécute les conteneurs comme ton propre utilisateur, à l'intérieur d'un espace de noms utilisateur — le « root » du conteneur est mappé sur une plage d'UID sans privilèges sur l'hôte, donc même un processus de conteneur entièrement compromis reste lié aux permissions de ton propre compte. Podman rootful (`sudo podman ...`) fonctionne de la manière traditionnelle, avec le root du conteneur mappé sur le vrai root de l'hôte. L'interrupteur du panneau ne fait en réalité que basculer entre exécuter `podman` simple ou `sudo podman` — des répertoires d'état réellement séparés, des listes de conteneurs séparées, rien de partagé entre les deux contextes par défaut.

```
# où chaque contexte garde son propre état
$ podman info -format '{{.Store.GraphRoot}}'
# rootless :  /.local/share/containers/storage
# rootful : /var/lib/containers/storage
```

POURQUOI CERTAINES ENTRÉES DU CATALOGUE ONT BESOIN DE PORTS SPÉCIFIQUES OUVERTS

Le mappage de port d'un conteneur (`3003:3000` dans l'exemple homepage) ne rend le service accessible **que depuis cette machine** par défaut — Podman le lie à `localhost` sauf indication contraire. L'atteindre depuis un autre appareil sur ton réseau nécessite en plus que ce port soit ouvert dans le panneau Firewall (Chapitre 4) — les deux systèmes sont des couches indépendantes, et un conteneur en cours d'exécution avec un port mappé n'est pas automatiquement accessible depuis internet ou le LAN juste parce que Podman écoute dessus.

CHAPTER 9

Gaming Mode

Un interrupteur principal en haut, une rangée de statut en temps réel, et quatre onglets : **PACKAGES**, **GAMEMODE**, **MANGOHUD**, **PROTON**.

9.1 Guide visuel — l'interrupteur principal

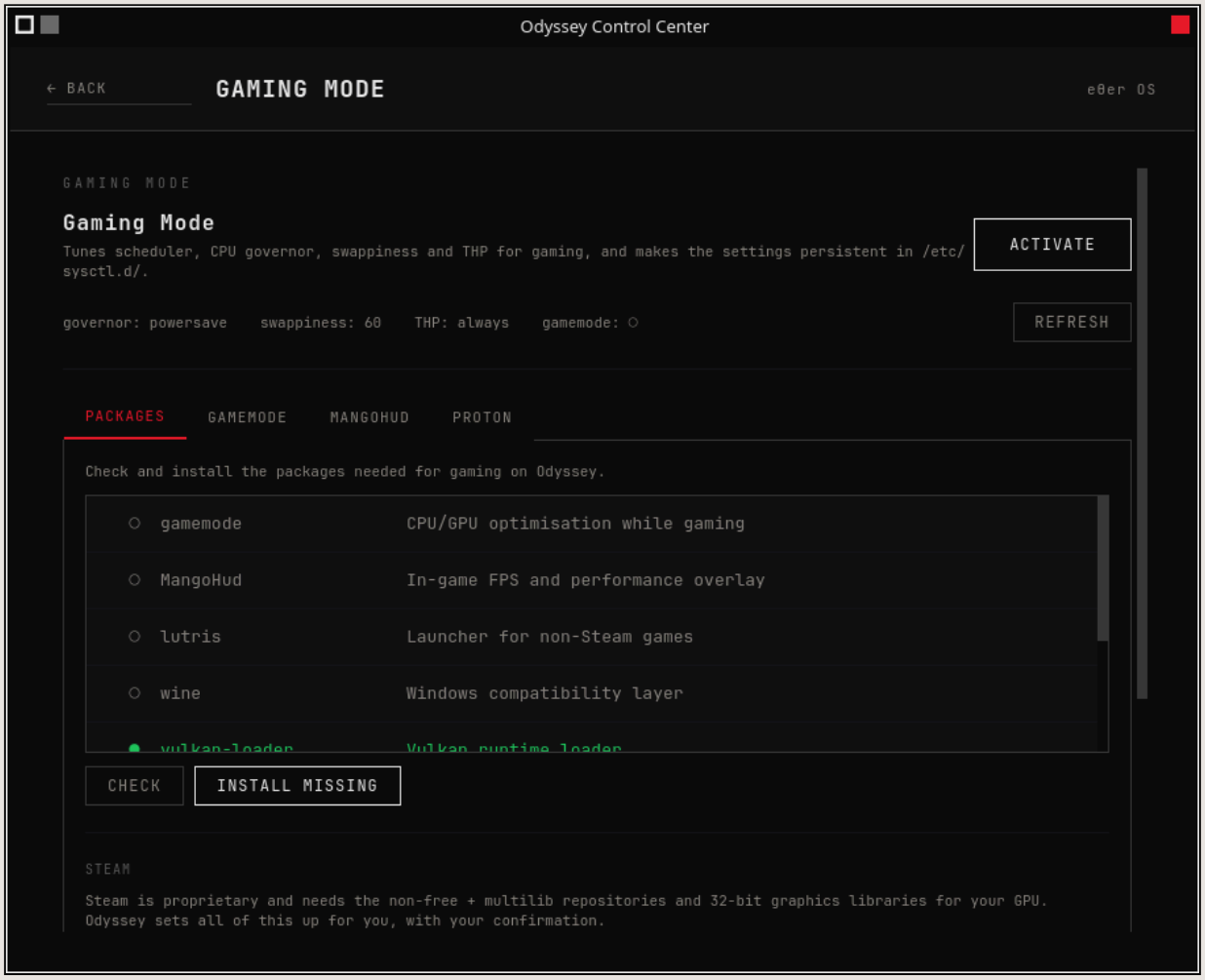

FIG · La section supérieure du panneau : « Gaming Mode — Tunes scheduler, CPU governor, swappiness and THP for gaming, and makes the settings persistent in /etc/sysctl.d/ », un bouton ACTIVATE, un bouton REFRESH, et la rangée de statut en direct : governor powersave, swappiness 60, THP always, gamemode off.

La rangée de statut affiche les mêmes valeurs en direct que montrent les panneaux Kernel et Memory — governor, swappiness, état THP, et si le service `gamemode` est en cours d'exécution — pour que tu puisses confirmer d'un coup d'œil si l'activation a vraiment fonctionné, sans visiter séparément l'un ou l'autre panneau.

WHAT	Gaming Mode → ACTIVATE.
WHY	Visiter manuellement Kernel et Memory pour fixer à la main un governor, une swappiness et une valeur THP adaptés au gaming, c'est exactement le genre de tâche à plusieurs étapes qu'un mode dédié devrait éliminer.
RESULT	Scheduler, governor, swappiness et THP passent tous à des valeurs adaptées au gaming, appliquées immédiatement et rendues persistantes — la rangée de statut se met à jour pour refléter le changement.
RISK	Faible, et réversible — les mêmes valeurs que tu fixerais autrement à la main dans Kernel/ Memory, juste regroupées en un clic.

9.2 Guide visuel — Packages et Steam

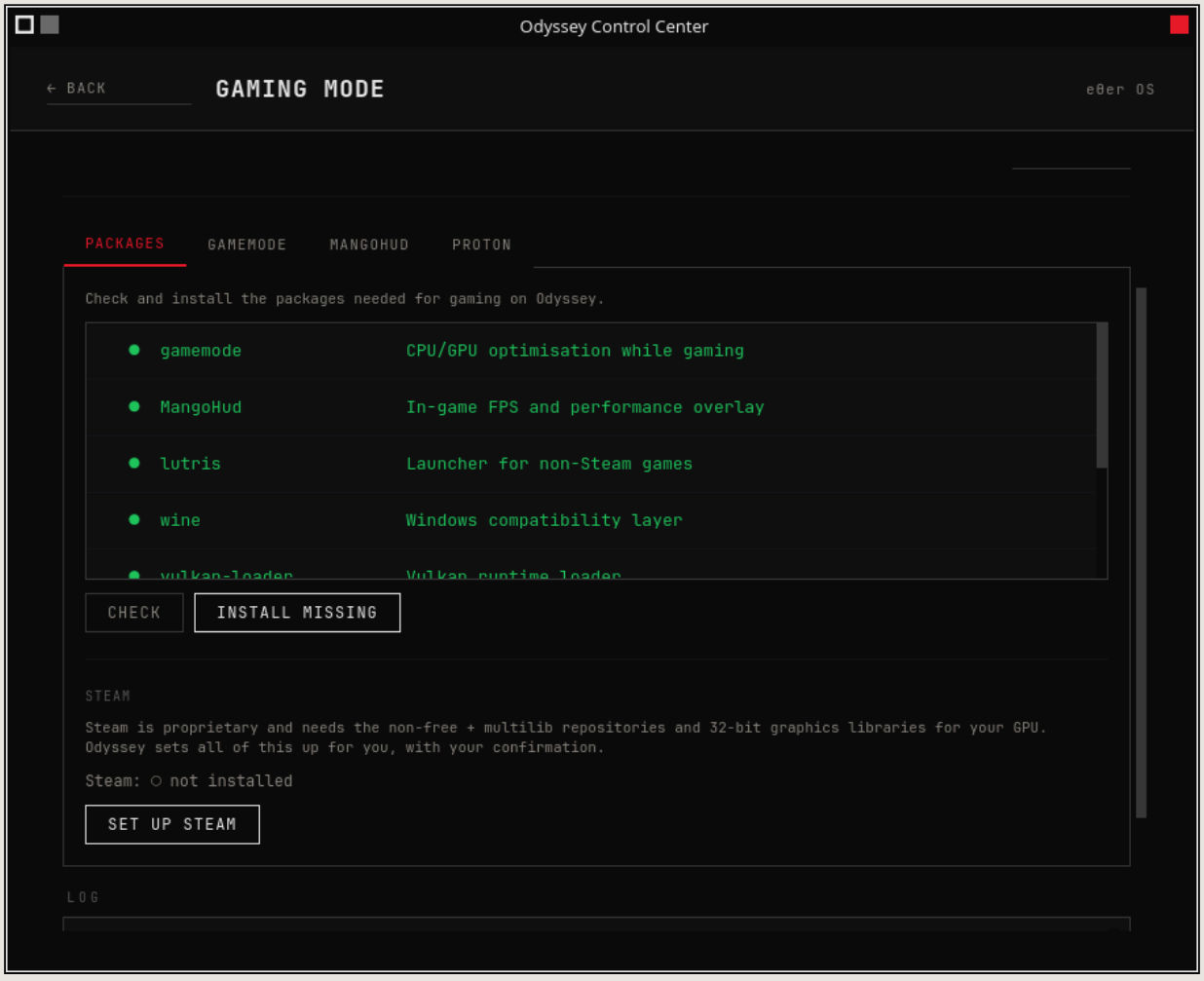

FIG · L'onglet PACKAGES : gamemode, MangoHud, lutris, wine, vulkan-loader tous avec des points verts pleins (installés) ; CHECK et INSTALL MISSING en haut ; la section Steam en dessous avec « Steam: ● installed » et un bouton REINSTALL STEAM.

Une liste de vérification des paquets généralement nécessaires pour le gaming — CHECK voit ce qui manque, INSTALL MISSING comble les lacunes en une seule action. Une fois que tout affiche un point vert plein, comme dans la capture, tu es entièrement configuré.

Steam a son propre flux dédié sous la liste de vérification, parce qu'il a besoin de plus qu'une simple installation de paquet.

1 **SET UP STEAM**

Clique dessus — cela n'installe encore rien, ça ouvre d'abord une boîte de dialogue de confirmation informative.

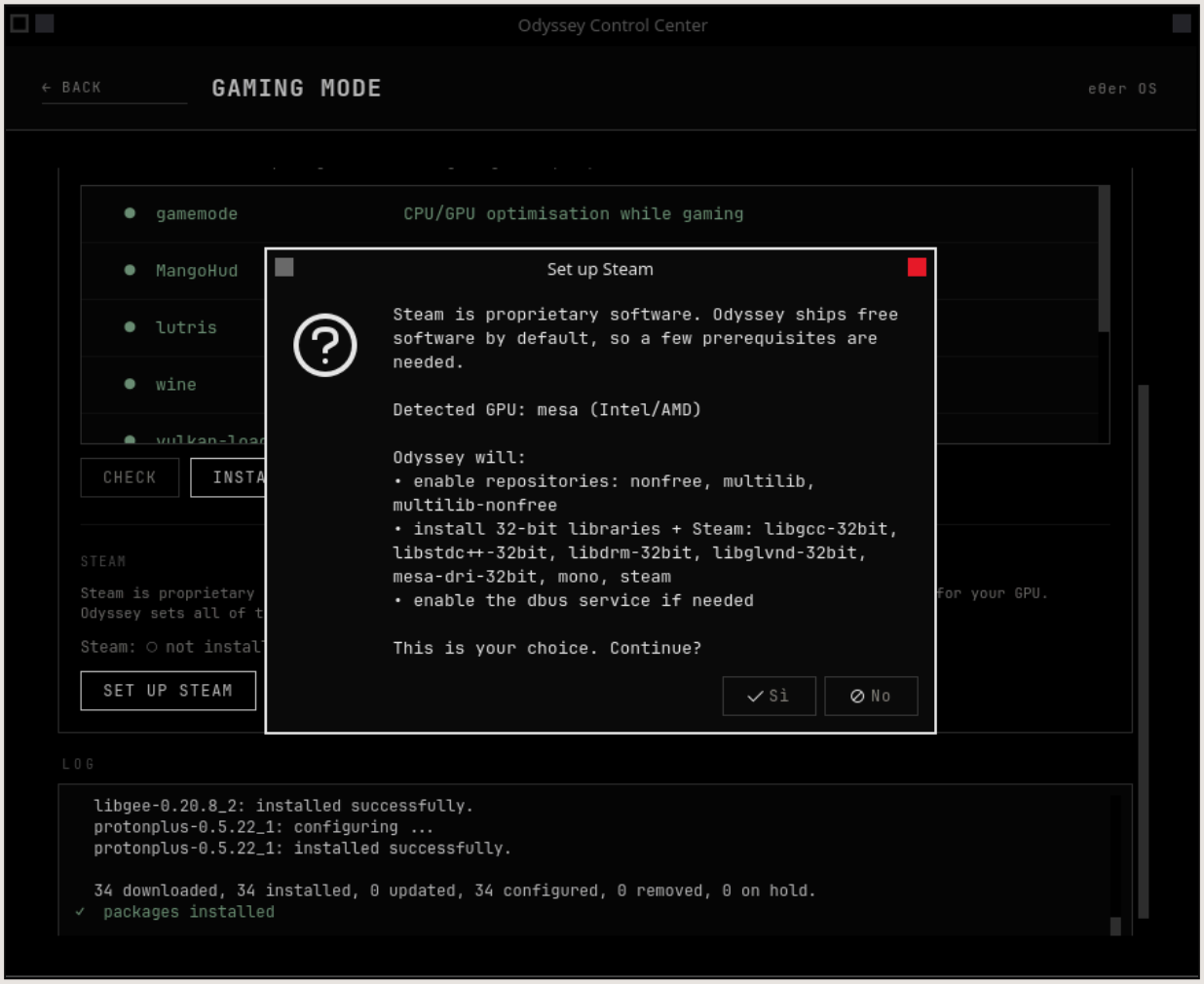

FIG · La boîte de dialogue de confirmation : « Steam is proprietary software. Odyssey ships free software by default, so a few prerequisites are needed. Detected GPU: mesa (Intel/AMD). » Une liste à puces de ce qui sera exactement activé et installé, se terminant par « This is your choice. Continue? » avec les boutons Sí / No.

La boîte de dialogue te dit exactement ce qui est sur le point de se passer avant que tu acceptes : quels dépôts sont activés (`nonfree`, `multilib`, `multilib-nonfree`), quelles bibliothèques 32 bits sont installées selon ton GPU **détecté**, et que le service dbus sera activé s'il ne l'est pas déjà. Rien n'est caché derrière la confirmation.

2 **Sí**

Le confirme. Le journal transmet l'installation réelle — téléchargements de paquets, configuration, finalisation.

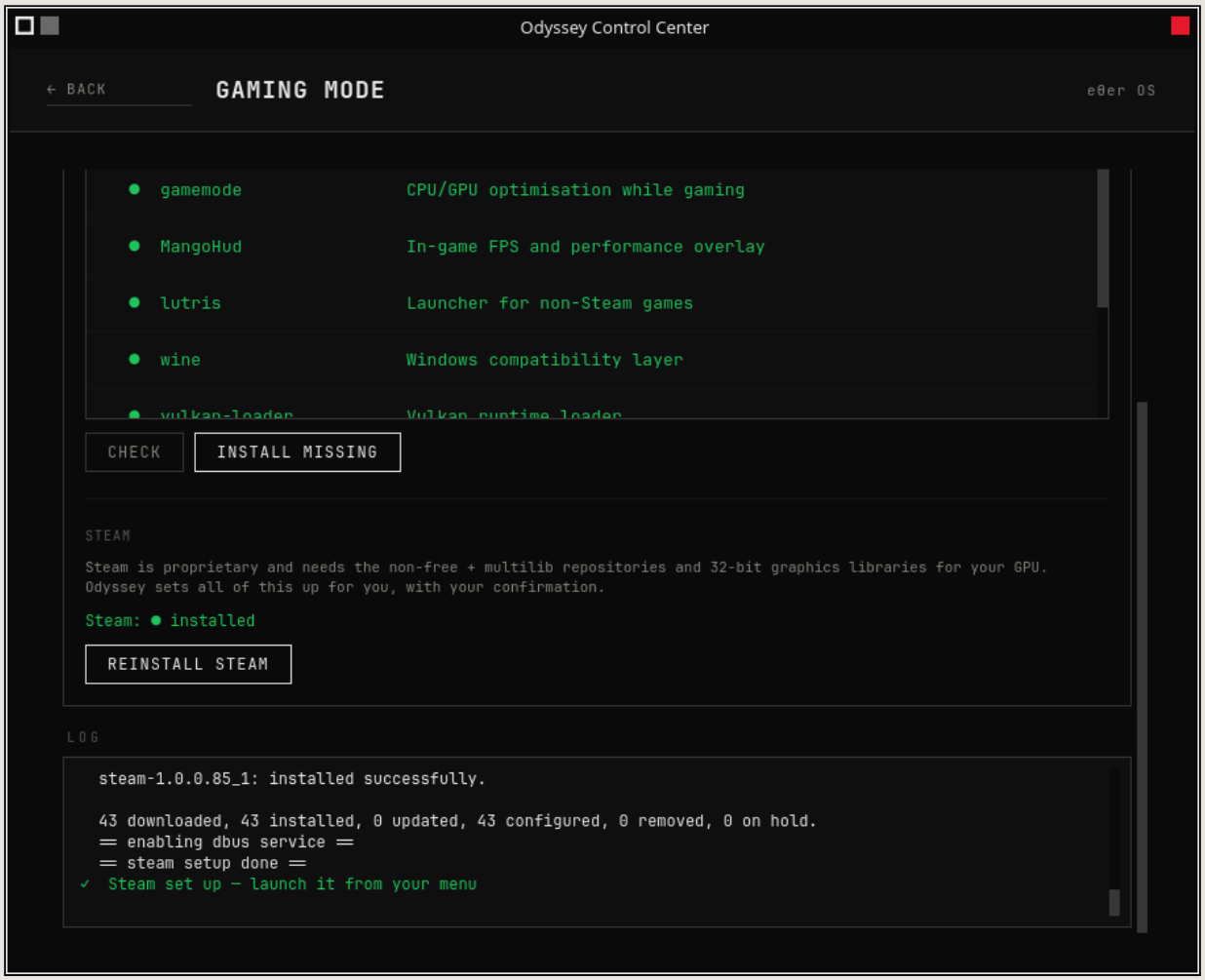

FIG · L'onglet PACKAGES après la configuration : tous les paquets en vert, Steam affichant « ● installed » avec REINSTALL STEAM, et le journal se terminant par « Steam set up — launch it from your menu. »

WHAT	Gaming Mode → onglet PACKAGES → SET UP STEAM → vérifie la boîte de dialogue → Sí.
WHY	Steam est propriétaire et a besoin des dépôts non-free/multilib plus de bibliothèques 32 bits spécifiques au GPU que Void n'active pas par défaut — bien faire cela à la main est un obstacle courant pour qui débute avec un système basé sur Void.
RESULT	Les dépôts nécessaires, Steam lui-même, et les bibliothèques graphiques 32 bits correspondant à ton GPU spécifique, tous installés — lance Steam depuis ton menu d'applications ensuite.
RISK	Faible — cela ne fait qu'activer des dépôts et ajouter des paquets ; rien d'existant n'est supprimé ni reconfiguré. La boîte de dialogue montre la liste complète de ce qui va se passer avant que tu confirmes.

9.3 GameMode et MangoHud

GameMode (son propre onglet) est le démon de performance par jeu de Feral Interactive — un jeu lancé à travers lui demande un boost temporaire de performance (governor CPU, priorité I/O) juste pour son propre processus, rendu automatiquement quand le jeu se ferme. **MangoHud** (son propre onglet) est la surimpression à l'écran montrant FPS, temps de frame, charge CPU/GPU et températures

pendant le jeu. Les deux onglets offrent des contrôles d'installation/vérification et, pour MangoHud, une configuration rapide de l'apparence de la surimpression.

9.4 Proton

Gestion de la couche de compatibilité pour exécuter des jeux Windows à travers Steam — vérifiant ce qui est installé et disponible, et installant des versions supplémentaires de Proton, y compris des builds communautaires comme Proton-GE, quand un jeu spécifique en nécessite une que Valve ne distribue pas par défaut.

```
▸ ADVANCED

Past this point: config file paths, raw commands, and what the panel is doing underneath.
```

9.5 Advanced

CE QU'ÉCRIT ACTIVATE

Les mêmes fichiers sous-jacents que les actions MAKE PERSISTENT des panneaux Kernel et Memory — Gaming Mode est un paquet groupé, pas un mécanisme séparé :

```
# valeurs sysctl rendues persistantes
$ cat /etc/sysctl.d/99-odyssey.conf
# CPU governor — ce n'est pas sysctl, c'est basé sur sysfs, non rendu persistant
# par ce mécanisme ; réapplique après le démarrage si tu en dépends
$ cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

CE QU'EXÉCUTE RÉELLEMENT SET UP STEAM

La boîte de dialogue de confirmation elle-même est presque une divulgation technique complète, mais concrètement, sur un système Mesa (Intel/AMD) :

```
$ sudo xbps-install -y void-repo-nonfree void-repo-multilib void-repo-multilib-nonfree
$ sudo xbps-install -Sy steam libgcc-32bit libstdc++-32bit libdrm-32bit libglvnd-32bit
mesa-dri-32bit mono
# les systèmes NVIDIA téléchargent en plus nvidia-libs-32bit au lieu de mesa-dri-32bit —
# le panneau détecte le fabricant de ton GPU et ajuste la liste de paquets en conséquence
```

GAMEMODE ET MANGOHUD À LA MAIN

```
# lance n'importe quel jeu avec GameMode actif
$ gamemoderun ./game_binary
# ou, pour un jeu Steam, options de lancement :
$ gamemoderun %command%
# surimpression MangoHud, options de lancement :
$ mangohud %command%
# le propre fichier de configuration de MangoHud, pour ajuster ce que montre la surim-
pression
$ $EDITOR  /.config/MangoHud/MangoHud.conf
```

INSTALLER UNE VERSION SPÉCIFIQUE DE PROTON-GE

L'onglet PROTON enveloppe `protonplus` ou un gestionnaire de versions équivalent — l'équivalent manuel consiste à télécharger une release depuis `GloriousEggroll/proton-ge-custom` sur GitHub et à l'extraire dans le répertoire des outils de compatibilité de Steam :

```
$ mkdir -p  /.steam/root/compatibilitytools.d
$ tar -xf GE-Proton-VERSION.tar.gz -C  /.steam/root/compatibilitytools.d
# puis sélectionne-la par jeu dans Steam : Properties → Compatibility
```

CHAPTER 10

Telemetry

Malgré son nom, ce panneau n'est pas un réglage à configurer — c'est un moniteur en temps réel. Odyssey ne collecte ni n'envoie rien te concernant, donc il n'y a rien à désactiver ici. Ce que ce panneau fait à la place, c'est te montrer, en temps réel, chaque connexion sortante que ton **propre** système établit — donc « à qui parle ma machine en ce moment » est quelque chose auquel tu peux répondre en regardant, pas en te fiant à une affirmation. Il s'appelle lui-même **Reverse Telemetry** précisément pour ça : il retourne l'observation, la pointant vers le système lui-même, en ton nom.

10.1 Guide visuel

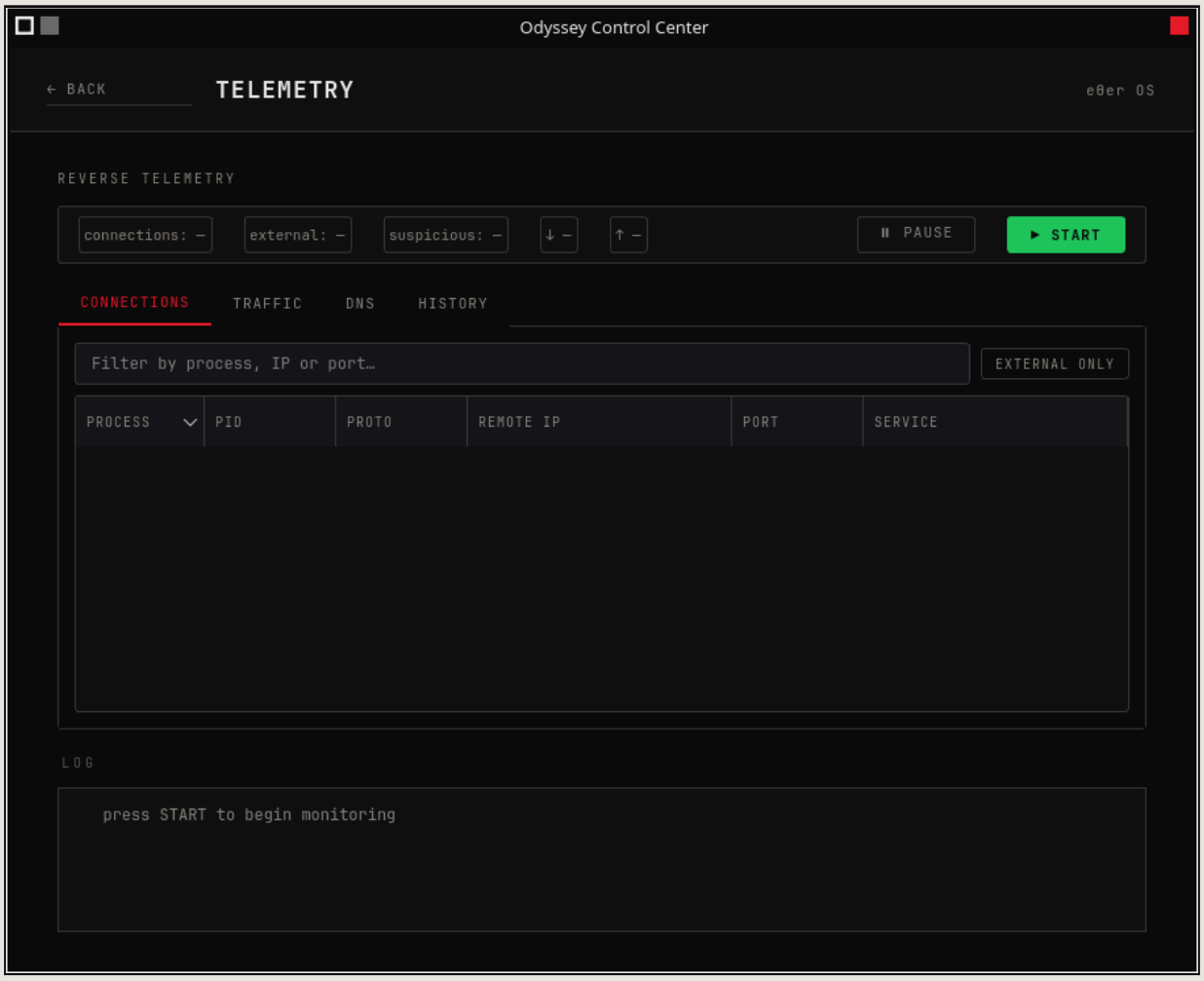

FIG · Avant de démarrer : connections/external/suspicious affichant tous « – », un bouton vert START, et le journal disant « press START to begin monitoring ». Les quatre onglets — CONNECTIONS, TRAFFIC, DNS, HISTORY — restent vides au-dessus d'un tableau non peuplé.

1 START

Démarre la surveillance en temps réel — la barre de statut et l'onglet actif se peuplent immédiatement.

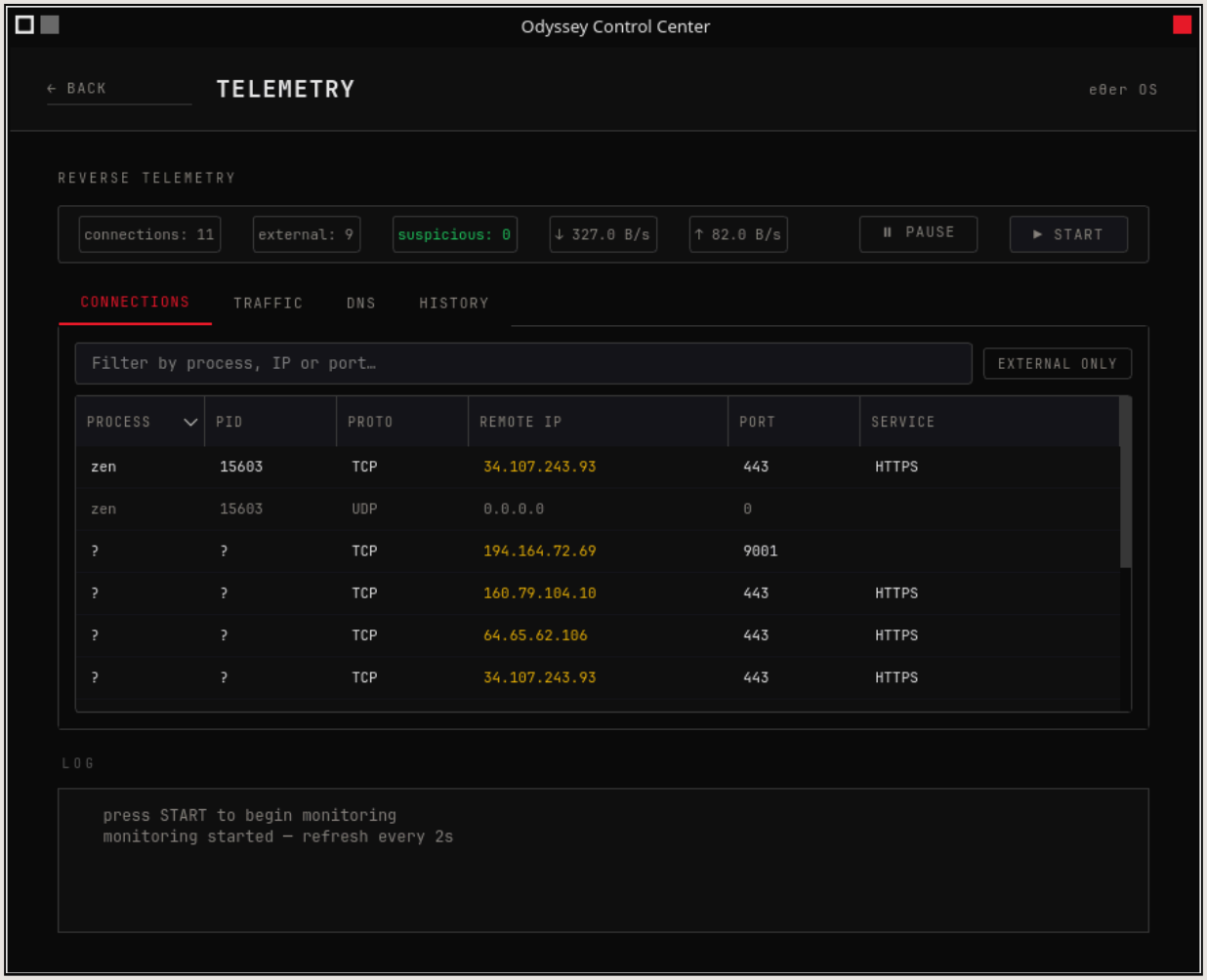

FIG · Après le démarrage : « connections: 11 · external: 9 · suspicious: 0 · ↓327.0 B/s · ↑82.0 B/s », PAUSE maintenant disponible, et le tableau CONNECTIONS rempli de vraies lignes : processus « zen » en TCP vers 34.107.243.93:443 (HTTPS), plusieurs lignes marquées « ? » pour processus/PID avec des connexions vers d'autres adresses.

La barre de statut en haut fournit des totaux en temps réel — connexions, externes (vs. locales), suspectes, vitesses actuelles de téléchargement/envoi — et **PAUSE** fige la vue sans perdre ce qui a été capturé, **START** la reprend.

LES QUATRE ONGLETS

ONGLET	CE QU'IL MONTRE
CONNECTIONS	Une ligne par connexion active : processus, PID, protocole, adresse distante, port, et le service connu pour ce port (443 → HTTPS, 22 → SSH). Triable par n'importe quelle colonne.
TRAFFIC	Débit par interface — vitesse de téléchargement/envoi en ce moment, plus les totaux cumulés depuis le début de la surveillance.
DNS	Un flux en temps réel des résolutions de noms de domaine au fur et à mesure — souvent plus clair qu'une simple IP, puisqu'un nom d'hôte te dit **quoi**, pas seulement **où**.

ONGLET	CE QU'IL MONTRE
HISTORY	Tout ce que la session a capturé, sous forme de journal défilant — utile pour repérer quelque chose de bref qui n'est plus une connexion active au moment où tu regardes.

Les lignes où le processus affiche `?` (visibles dans la capture) signifient que la connexion a été capturée mais que le processus d'origine n'a pas pu être résolu à partir du PID à cet instant — courant pour des connexions très brèves. Un champ de **filtre** au-dessus du tableau restreint par processus, IP ou port, et **EXTERNAL ONLY** cache le trafic purement local.

WHAT	Telemetry → START → observe CONNECTIONS, ou trie par PROCESS ou REMOTE IP pour enquêter sur quelque chose de spécifique.
WHY	C'est la réponse directe et concrète à « y a-t-il quelque chose sur cette machine qui atteint le réseau et qui ne devrait pas » — lue directement dans les propres tables de connexion du noyau, pas dans l'affirmation d'un fournisseur sur son propre logiciel.
RESULT	Une image en direct et précise de chaque connexion sortante, mise à jour en continu tant que c'est en cours d'exécution et non en pause.
RISK	Aucun — entièrement en lecture seule.

Les connexions de processus sans raison ordinaire d'atteindre le réseau par eux-mêmes — un shell nu, un interpréteur de scripts en exécution autonome — sont marquées automatiquement sous « suspicious » dans la barre de statut, pour qu'une anomalie ressorte au lieu d'être enterrée dans le trafic normal de ton navigateur et des services système.

```
▸ ADVANCED

Past this point: config file paths, raw commands, and what the panel is doing underneath.
```

10.2 Advanced

D'OÙ VIENNENT LES DONNÉES

Le panneau lit les mêmes tables de connexion au niveau du noyau que lirait n'importe quel outil standard, actualisées à un intervalle court pendant que la surveillance est active — aucun pilote ou module noyau spécial propre. Les équivalents en ligne de commande :

```
# chaque socket ouvert, avec le processus propriétaire — mêmes champs que CONNECTIONS
$ sudo ss -tupn
# débit par interface — mêmes chiffres que TRAFFIC
$ ip -s link
# une vue en direct ponctuelle, si tu la veux en dehors du panneau
$ sudo nethogs
```

RÉSOLUTIONS DNS, À LA MAIN

```
# capture en direct des requêtes DNS sortantes (nécessite un outil de capture de paquets)
$ sudo tcpdump -i any -n port 53
```

POURQUOI CERTAINES LIGNES AFFICHENT PROCESS COMME « ? »

Associer une connexion en direct à son processus propriétaire nécessite de lire `/proc/PID/fd` pour chaque processus candidat au moment de la capture — si la connexion s'ouvre et se ferme plus vite que l'intervalle de rafraîchissement du panneau, ou que le processus propriétaire a déjà terminé, le panneau n'a pas la fenêtre pour faire cette association et affiche `?` plutôt que de deviner. C'est une limitation de timing, pas un signe que quelque chose est caché.

CHAPTER 11

About

Une unique fiche diagnostique exhaustive de la machine, actualisée chaque fois que tu ouvres l'onglet, pour toujours refléter ce qui a changé dans les autres panneaux sans avoir besoin de rouvrir le Control Center.

11.1 Guide visuel

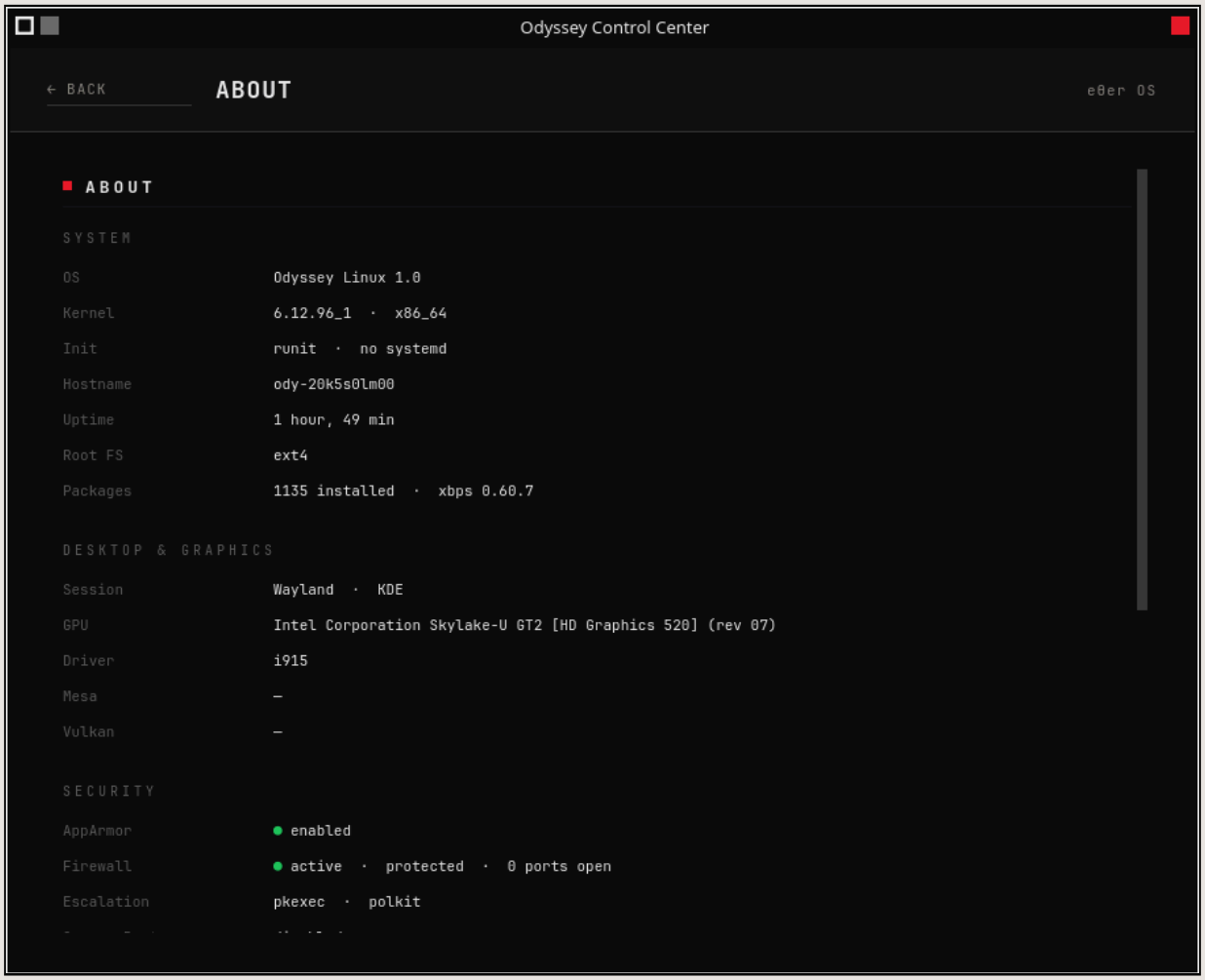

FIG · Le panneau défilé jusqu'en haut : section SYSTEM montrant OS « Odyssey Linux 1.0 », Kernel 6.12.96_1 · x86_64, Init runit · no systemd, Hostname, Uptime, Root FS ext4, Packages « 1135 installed · xbps 0.60.7 » ; DESKTOP & GRAPHICS en dessous avec Session, GPU, Driver, Mesa, Vulkan ; SECURITY commençant à montrer AppArmor « ● enabled » et Firewall « ● active · protected · 0 ports open ».

Trois sections, dans l'ordre : **SYSTEM** (noyau, init, nom d'hôte, uptime, système de fichiers, décompte de paquets), **DESKTOP & GRAPHICS** (type de session, GPU, pilote, support Mesa/Vulkan), et plus bas

SECURITY (statut d'AppArmor et Firewall avec des indicateurs colorés, pas juste du texte — vert pour activé/actif) et **HARDWARE** (carte mère, audio).

WHAT	About → fais défiler les trois sections pour vérifier n'importe quelle valeur — version du noyau, pilote utilisé, si AppArmor et le pare-feu sont vraiment actifs.
WHY	C'est le moyen le plus rapide de confirmer l'état de tout le système en un seul endroit, sans ouvrir cinq panneaux différents.
RESULT	Un instantané complet et actuel — actualisé à chaque visite de l'onglet, donc il ne montre jamais d'informations obsolètes d'une session précédente.
RISK	Aucun — entièrement en lecture seule.

11.2 Copy report

Un seul bouton en haut du panneau, **COPY REPORT**, met toute la fiche dans le presse-papiers en texte brut — chaque section, chaque valeur — prête à être collée directement dans un message de forum ou un rapport sur le suivi des bugs.

WHAT	About → COPY REPORT → colle dans ton rapport de bug ou message de forum.
WHY	Les rapports de bugs ne sont utiles qu'autant que les détails système qui les accompagnent, et taper à la main la version du noyau, le pilote, l'état d'AppArmor et une douzaine d'autres champs est exactement le genre de friction qui fait sauter cette étape à beaucoup de gens.
RESULT	Un rapport système complet et correctement formaté dans le presse-papiers, prêt à coller — les mêmes informations qu'un mainteneur devrait sinon te demander champ par champ.
RISK	Aucun pour ton système. Le rapport est de l'information diagnostique (versions, états, modèle matériel) — il n'inclut pas de fichiers personnels ni d'identifiants — mais comme pour tout export diagnostique, relis-le avant de le publier quelque part en public si tu n'es pas sûr de ce qu'il contient.

▸ ADVANCED

Past this point: config file paths, raw commands, and what the panel is doing underneath.

11.3 Advanced

D'OÙ VIENT CHAQUE CHAMP

Rien dans ce panneau n'est inventé ni mis en cache depuis le moment de l'installation — chaque valeur est interrogée en direct, à chaque ouverture de l'onglet :

```
$ uname -r # Kernel
$ cat /proc/sys/kernel/osrelease
$ hostnamectl 2>/dev/null || hostname
$ uptime -p
$ xbps-query -l | wc -l # Packages installed
$ xbps-query -v # xbps version
$ lspci -k | grep -A3 VGA # GPU + Driver
$ glxinfo | grep “OpenGL version” # Mesa
$ vulkaninfo –summary # Vulkan
```

Section Security en particulier :

```
$ sudo aa-status –enabled && echo enabled
$ sudo nft list ruleset | head -1 # firewall active/mode
```

POURQUOI ÇA VAUT LE COUP DE VÉRIFIER ABOUT EN PREMIER

Comme il réinterroge le système à chaque visite, About est le moyen le plus rapide de confirmer qu’un changement fait dans un autre panneau a vraiment pris effet — applique un mode de pare-feu dans Firewall, puis vérifie la section Security d’About plutôt que de rouvrir Firewall lui-même. Il est construit pour être la vérification croisée « est-ce que ça a vraiment marché ? » pour le reste du Control Center.

CHAPTER 12

NVIDIA driver

EN ATTENTE D'INTÉGRATION

Ce chapitre est un espace réservé. Le module NVIDIA n'apparaît dans le Control Center que sur les machines équipées de matériel NVIDIA détecté, et les captures d'écran pour lui ne peuvent être obtenues que sur une machine de ce type — elles seront fournies séparément et ce chapitre sera complété dans une future révision de ce guide. Il est délibérément gardé en dernier, après About, précisément pour que l'ajouter plus tard ne déplace la numérotation des pages ni les références de section d'aucun autre chapitre.

Ce que l'on sait déjà et qui ancrera le chapitre terminé : le panneau porte un badge **BETA** visible et une note initiale indiquant que le pilote lui-même est propriétaire — empaqueté par Void à partir de l'installateur officiel de NVIDIA avec une somme de contrôle fixée, mais non auditable de façon indépendante, la même honnêteté que le reste d'Odyssey s'applique à lui-même. Il détecte ton GPU spécifique en le comparant à une base de données intégrée et choisit automatiquement la bonne branche de pilote (current/Turing-et-plus-récent, ou l'une des trois branches héritées pour les cartes plus anciennes, se rabattant sur le pilote open source `nouveau` quand rien de propriétaire ne s'applique). L'installation tourne comme un pipeline suivi en sept étapes avec un indicateur de progression visible, et le panneau imprime sa propre commande de récupération à l'avance — `odyssey-nvidia --revert` depuis une console texte — avant même que tu cliques sur installer. Sur les ordinateurs portables hybrides, une section PRIME render-offload apparaît automatiquement, permettant à un jeu ou programme spécifique de tourner sur le GPU NVIDIA pendant que tout le reste reste sur le GPU intégré pour l'autonomie de la batterie.