Odyssey Control Center

الدليل العملي

كلّ لوحة، قسمًا قسمًا. ماذا تفعل، وماذا تغيّر،
وما الذي تنقره — مكتوبًا لمن لم يستعملها قطّ.

Launch Edition

.Don’t trust me. Verify me

nobody —

Odyssey Linux — مركز التحكّم Odyssey Control Center: الدليل العملي
Launch Edition. يغطّي مركز التحكّم كما يُشحن على قرص Launch Edition — عشر لوحات كاملة، من Boot إلى About، مع فصل نائب لوحدة مشغّل NVIDIA (المشروطة بالعتاد، بانتظار لقطات من آلة تحويه).

Copyright © 2026 مشروع Odyssey Linux.
منشور برخصة المشاع الإبداعي: النسب بالمثل 4.0 دولي (CC BY-SA 4.0).

هذا الدليل رفيقٌ لدليل الأمان والتحصين، الذي يغطّي لوحتَي Firewall و AppArmor بعمق كامل — أمّا هذا المستند فيعالجهما بجولة مصوّرة ويحيل إلى هناك لما عداها.

كلّ شاشة وزرّ وسلوك موصوف هنا قُرئ من مصدر مركز التحكّم الفعلي ومن لقطات حقيقية، لا من الذاكرة.

ملاحظة عن اللغة: تسميات الواجهة وأسماء اللوحات والتبويبات والمسارات والأوامر ومخرجات الطرفية بقيت بالإنجليزية — فهي الواجهة الحقيقية للنظام. وما عداها مترجَم.

المحتويات

1 ما هو مركز التحكّم 1
1.1 أنماط ستراها في كلّ لوحة 2
2 الإقلاع 3
2.1 جولة — العثور على نظام آخر وضبط الافتراضي 3
2.2 جولة — اختيار ما يُقلع افتراضيًا 5
2.3 جولة — تبويب EFI 7
2.4 معاملات النواة 9
2.5 متقدّم 9
3 النواة 11
3.1 جولة — الإعدادات الجاهزة 11
3.2 جولة — SYSCTL، مجموعةً مجموعة 12
3.3 جولة — منظّم المعالج 14
3.4 جولة — الصفحات الضخمة 15
3.5 جولة — الوحدات 16
3.6 متقدّم 17
4 الذاكرة والتبديل 19
4.1 جولة 19
4.2 متقدّم 20
5 جدار الحماية 22
5.1 جولة 22
5.2 متقدّم 23
6 AppArmor 24
6.1 جولة — STATUS 24
6.2 جولة — PROFILES 25
6.3 جولة — COMMUNITY 26
6.4 جولة — VIOLATIONS 28
6.5 جولة — EDITOR 29
6.6 متقدّم 30
7 الخصوصية 32
7.1 جولة — SearXNG خطوةً خطوة 32
7.2 جولة — Tor خطوةً خطوة 34
7.3 جولة — الجمع بينهما 37
7.4 متقدّم 38
8 الحاويات 40
8.1 جولة — المحرّك 40
8.2 جولة — بلا جذر مقابل بجذر، والحالة الفارغة 42
8.3 جولة — الفهرس المنتقى، من البداية إلى النهاية 42
8.4 جولة — قائمة الحاويات وإجراءاتها 46
8.5 جولة — تشغيل حاوية وفتحها 47
8.6 استهلاك القرص 48
8.7 متقدّم 48

9 وضع الألعاب 50
9.1 جولة — المفتاح الرئيسي 50
9.2 جولة — الحزم وSteam 51
9.3 GameMode و MangoHud 53
9.4 Proton 54
9.5 متقدّم 54
10 القياس عن بُعد 56
10.1 جولة 56
10.2 متقدّم 58
11 عن النظام 60
11.1 جولة 60
11.2 نسخ التقرير 61
11.3 متقدّم 61
12 مشغّل NVIDIA 63

ما هو مركز التحكّم

مركز التحكّم Odyssey Control Center هو عشرة برامج صغيرة مستقلّة خلف مُطلِق واحد — واحدٌ لكلّ ناحية من النظام: Boot و Kernel و Memory & Swap و Firewall و AppArmor و Privacy و Containers و Gaming Mode و Telemetry و About، وعلى عتاد NVIDIA وحدة تعريف NVIDIA كذلك.

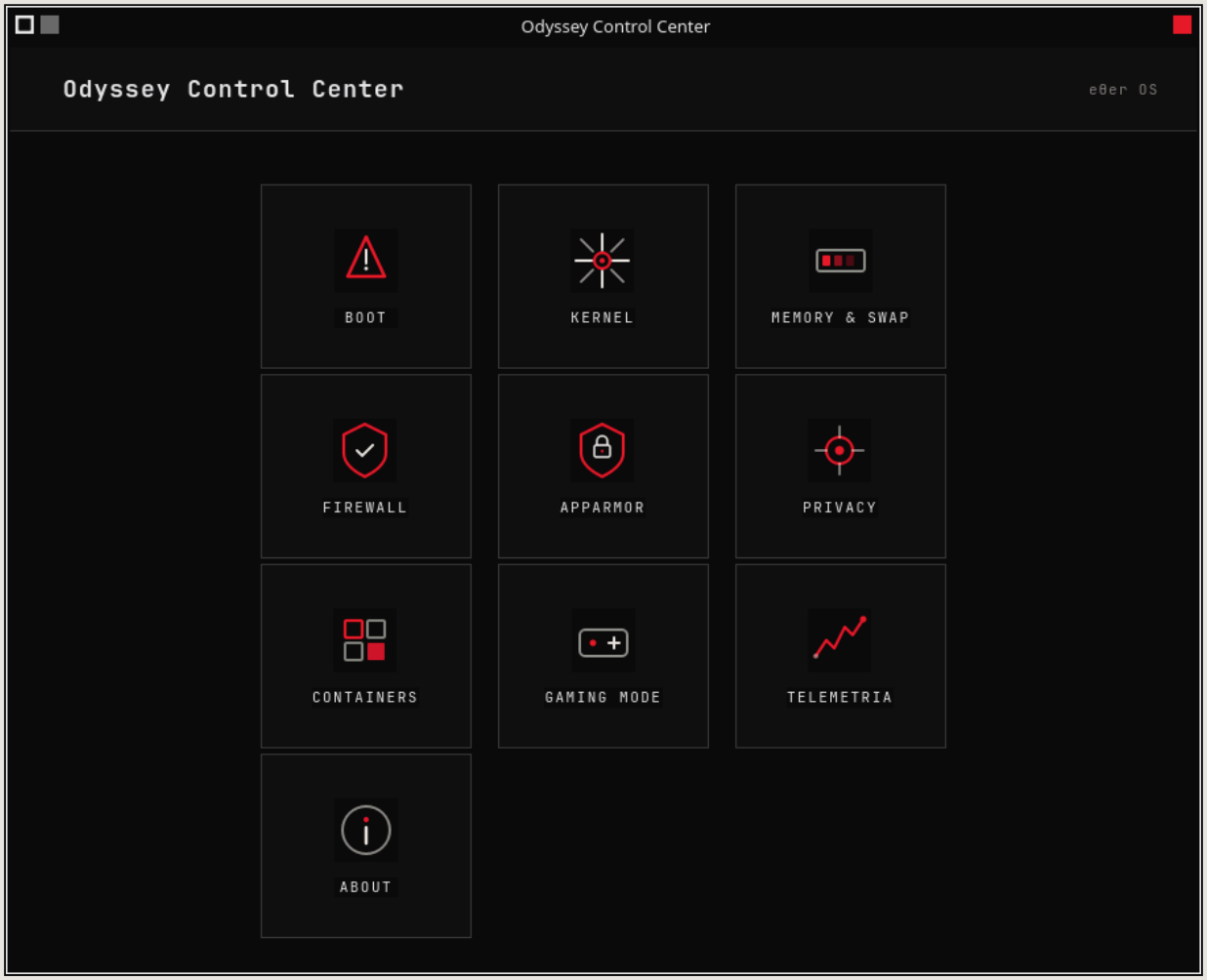

FIG · شاشة مركز التحكّم الرئيسية — بلاطة واحدة لكلّ لوحة.

كلّ لوحة طبقة رسومية رقيقة فوق آلية لينكس الحقيقية تحتها: إعداد GRUB نفسه، وواجهة sysctl الخاصّة بالنواة، و`nftables`، و`podman`. ولا تخترع لوحة قطّ حالةً خاصّة بها — بل تقرأ الملفّات والأوامر الفعلية، وتعرضها عليك، وتكتب ما طلبته أنت. لا شيء ممّا تفعله مخفيّ، وكلّ ما تفعله يمكن فعله يدويًا بدلًا منها، من طرفية، بالأوامر ذاتها التي يعرضها هذا الدليل بجوار كلّ زرّ.

هذا دليل عملي، لا مرجعٌ لكلّ خيار — فهو يجيب عن سؤال «ما الذي أنقره، وماذا يحدث إن نقرته». ولكلّ فصل الشكل ذاته المؤلَّف من جزأين: **جولة مصوّرة** أوّلًا، لقطةً لقطة وزرًّا زرًّا، مكتوبة لمن لم يفتح مركز التحكّم في حياته — ثمّ قسم **متقدّم**، تعلّمه لافتة

سوداء، يدخل في التفاصيل: المسارات الدقيقة للملفّات، والأوامر الخام التي ينفّذها كلّ زرّ تحته، وما تفعله من الطرفية إن آثرت تخطّي اللوحة كلّيًا. اقرأ الجولة وحدها، أو اقفز مباشرةً إلى المتقدّم إن كنت تعرف طريقك في لينكس — وكلا القسمين كامل بذاته.

أمّا اللوحتان ذواتا الرهان الأمني الحقيقي — Firewall و AppArmor — فمرجعهما العميق هو **دليل الأمان والتحصين**؛ وهنا تجد الجولة والأساسيات وإشارةً إلى هناك.

1.1 أنماط ستراها في كلّ لوحة

بضع أعراف تتكرّر في اللوحات العشر — تعلّمها مرّة واحدة.

طلب كلمة المرور، مرّة واحدة لكلّ لوحة

معظم اللوحات تحتاج إلى قراءة ملفّات يملكها الجذر أو الكتابة فيها — إعداد GRUB، و sysctl، ومجموعة قواعد جدار الحماية. وفي أوّل مرّة تحتاج فيها اللوحة إلى ذلك تظهر نافذة استيثاق واحدة. واللوحات مبنيّة لتجمّع قراءاتها: ففتح Boot مثلًا يقرأ إعداد GRUB **و** حالة برنامج EFI الثابت في طلب واحد، فتكتب كلمة مرورك مرّةً لا مرّتين. وإعادة فتح اللوحة في الجلسة نفسها لا تسأل ثانيةً؛ أمّا الإجراءات التالية (تطبيق تغيير، إعادة تحميل) فتُظهر تقدّمها في سجلّ اللوحة نفسه.

متتبّع الخطوات

اللوحات التي ترشدك إلى تثبيت شيء قبل أن يصير صالحًا للاستعمال — os-prober في Boot، و SearXNG و Tor في Privacy، ومحرّك الحاويات في Containers — تعرض متتبّعًا صغيرًا في الأعلى: صفٌّ من الخطوات، المرحلة الحالية مضاءة فيه، وزرّ إجراء واحد ينفّذ دائمًا **الخطوة التالية**، أيًّا كانت. فلا تحتاج أبدًا إلى تخمين ما تنقره أوّلًا.

السجلّ في الأسفل

تنتهي كلّ لوحة بسجلّ صغير قابل للتمرير. وهو ليس زينة — بل هو الأمر الذي نُفِّذ بالضبط، ومخرجاته، وحكمٌ ملوَّن (أخضر للنجاح، وأحمر للفشل). وإن فعلت لوحةٌ شيئًا لا تفهمه، فالسجلّ هو موضع الجواب الحقيقي.

TRY في مقابل MAKE PERSISTENT

زوجٌ متكرّر من الأزرار، في Kernel و Memory خصوصًا: **TRY NOW** (أو **APPLY NOW**) يغيّر النظام العامل فورًا، ويعود إلى سابق عهده عند إعادة الإقلاع التالية إن لم تفعل شيئًا آخر. أمّا **MAKE PERSISTENT** فيكتب القيمة نفسها في ملف إعداد تحت `/etc/`، فتبقى بعد إعادة الإقلاع أيضًا. وهذا الفصل مقصود — فلا تكلفة لتجربة قيمة، ولا شيء يصير دائمًا حتى تقول ذلك مرّتين.

الإقلاع

للوحة Boot تبويبان على أيّ آلة تُقلِع بنمط UEFI: **GRUB** (قائمة الإقلاع نفسها) و**EFI** (مستوى البرنامج الثابت تحتها). أمّا على الآلات القديمة ذات BIOS فقط فلا يوجد إلّا GRUB، لأنّ طبقة EFI أسفله غير موجودة أصلًا.

طبقتان لا طبقة واحدة

GRUB هو قائمة الإقلاع التي تراها على الشاشة — وهو يقرّر أيّ نظام تشغيل يبدأ بعد أن يكون GRUB نفسه قد عمل. أمّا **البرنامج الثابت EFI/UEFI** فهو مستوًى أسفل GRUB — وهو يقرّر إلى أيّ محمّل إقلاع تسلّم الآلة زمام الأمور ابتداءً، قبل أن يُحمَّل GRUB أصلًا. ومعظم الناس لا يلمسون تبويب EFI البتّة؛ فهو موجود للحالات التي لا يبلغها GRUB، مثل تثبيت ويندوز ثانٍ يدير مدخل إقلاعه بنفسه.

2.1 جولة – العثور على نظام آخر وضبط الافتراضي

يفتح تبويب GRUB على قسم **Other OS detection**. وOdyssey لا يمسح بحثًا عن أنظمة أخرى افتراضيًا — فمعظم التثبيتات لا تحوي إلّا Odyssey على القرص، والمسح يكلّف وقتًا عند كلّ إعادة توليد لقائمة الإقلاع بلا طائل. وتشغيله ثلاث خطوات موجَّهة، تظهر في متتبّع صغير مرقَّم.

1 التثبيت

أداة المسح، os-prober، ليست على النظام بعد. زرّ واحد: **INSTALL OS-PROBER**.

2 تفعيل المسح

الأداة مثبَّتة، لكنّ GRUB ما يزال مأمورًا بتجاهلها — وهو افتراضي Odyssey. زرّ واحد: **ENABLE SCAN**.

3 المسح

سيستعملها GRUB الآن. زرّ واحد، وهو نفسه الذي ستستعمله كلّما أضفت نظامًا آخر أو أزلته لاحقًا: **SCAN & REGENERATE**.

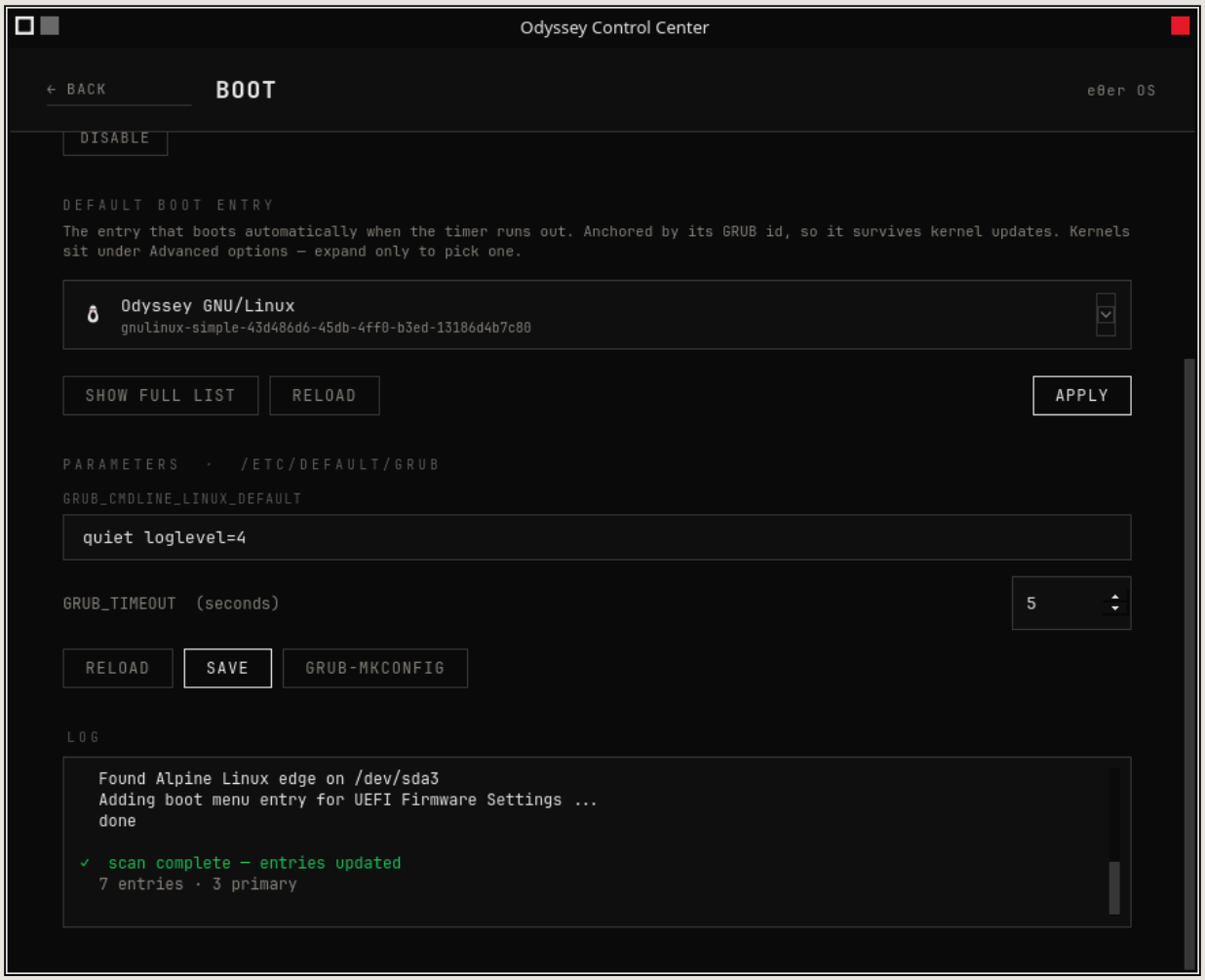

FIG · الخطوات الثلاث خضراء — os-prober مثبَّت، والمسح مفعّل، وكلّ شيء جاهز. وزرّ الإجراء الواحد يطابق دائمًا الخطوة الحالية.

اضغط الخطوات واحدةً واحدة (ولا تعيد كلّ خطوة طلب الاستيثاق إلّا عند اللزوم) ثمّ شغّل المسح. وتعرض اللوحة عليك ما وجدته بالضبط قبل أن تغادر القسم:

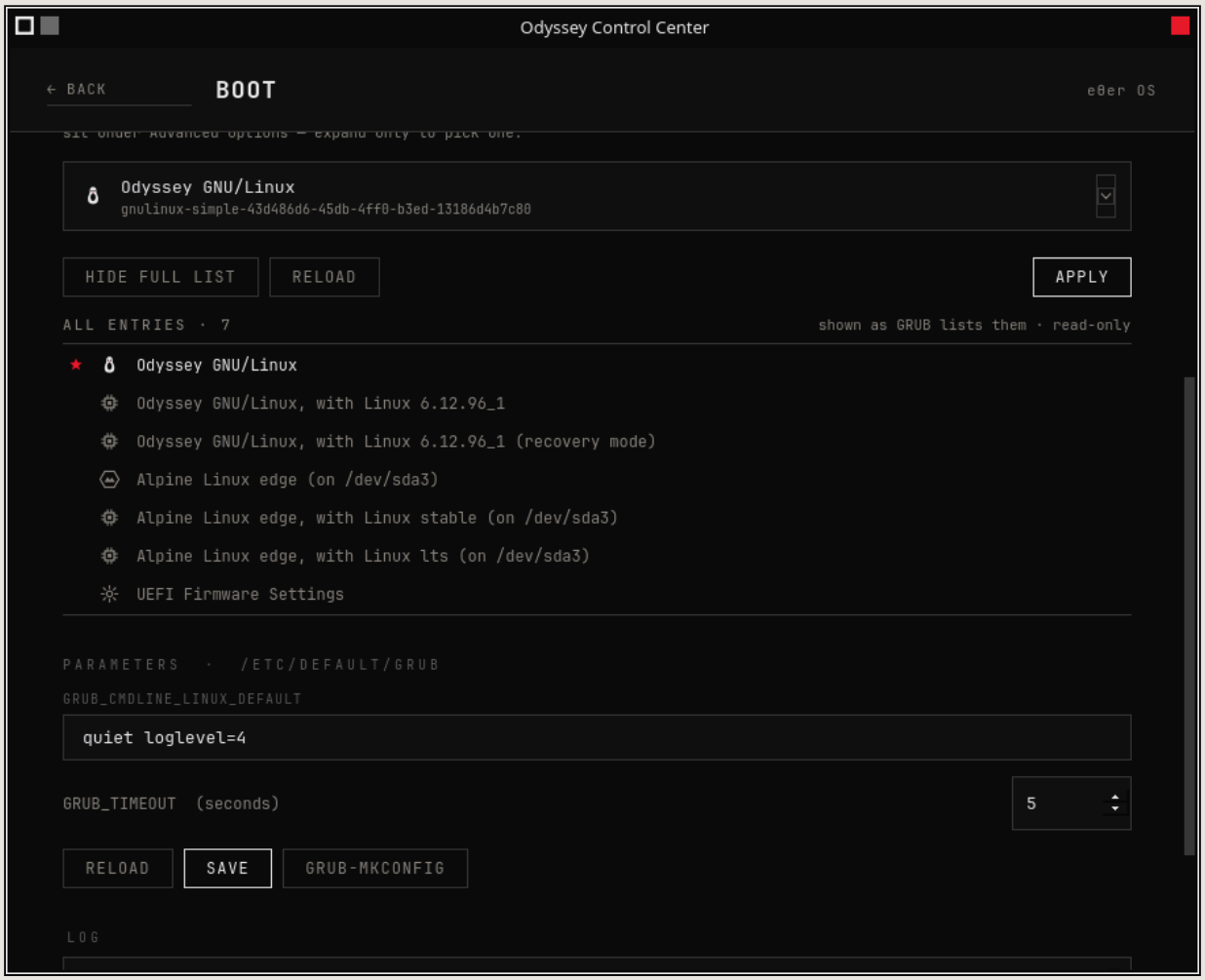

FIG · اكتمل المسح: عُثر على Alpine Linux على /dev/sda3، إلى جانب مدخلات Odyssey نفسها ومدخل إعدادات برنامج UEFI الثابت — 7 مدخلات، 3 منها أساسية.

ويؤكّد السجلّ ذلك بعبارة صريحة — **«scan complete — entries updated · 7 entries · 3 primary.»** وكلّ ما يجده os-prober يظهر تلقائيًا في القسم الذي يليه، جاهزًا للاختيار.

2.2 جولة — اختيار ما يُقلِع افتراضيًا

انزل إلى **Default boot entry**. يعرض القسم الافتراضي الحالي في سطر واحد؛ انقره لتفتح قائمة منسدلة بكلّ ما يمكنك اختياره بدلًا منه.

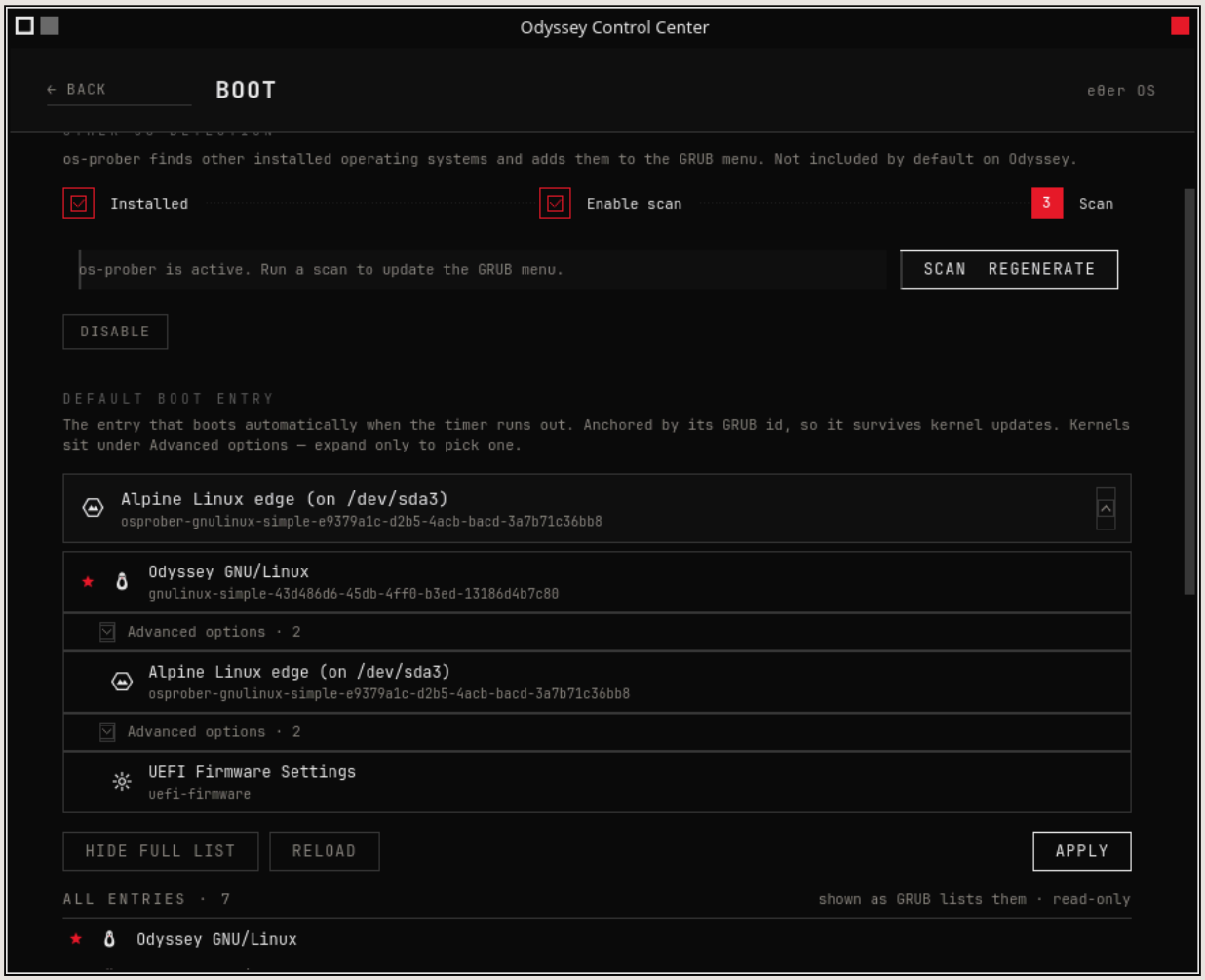

FIG · القائمة المنسدلة مفتوحة، و Alpine Linux محدّد كاختيار معلَّق — لم يُطبَّق بعد.

اختيار مدخل من القائمة **يُجهّزه** فقط — ولا يسري بعد. وتعرف أنّه معلَّق لأنّ الصندوق يعرض الاختيار الجديد بينما لم يُكتب شيء. والزرّ الذي يثبّته هو **APPLY** على اليمين:

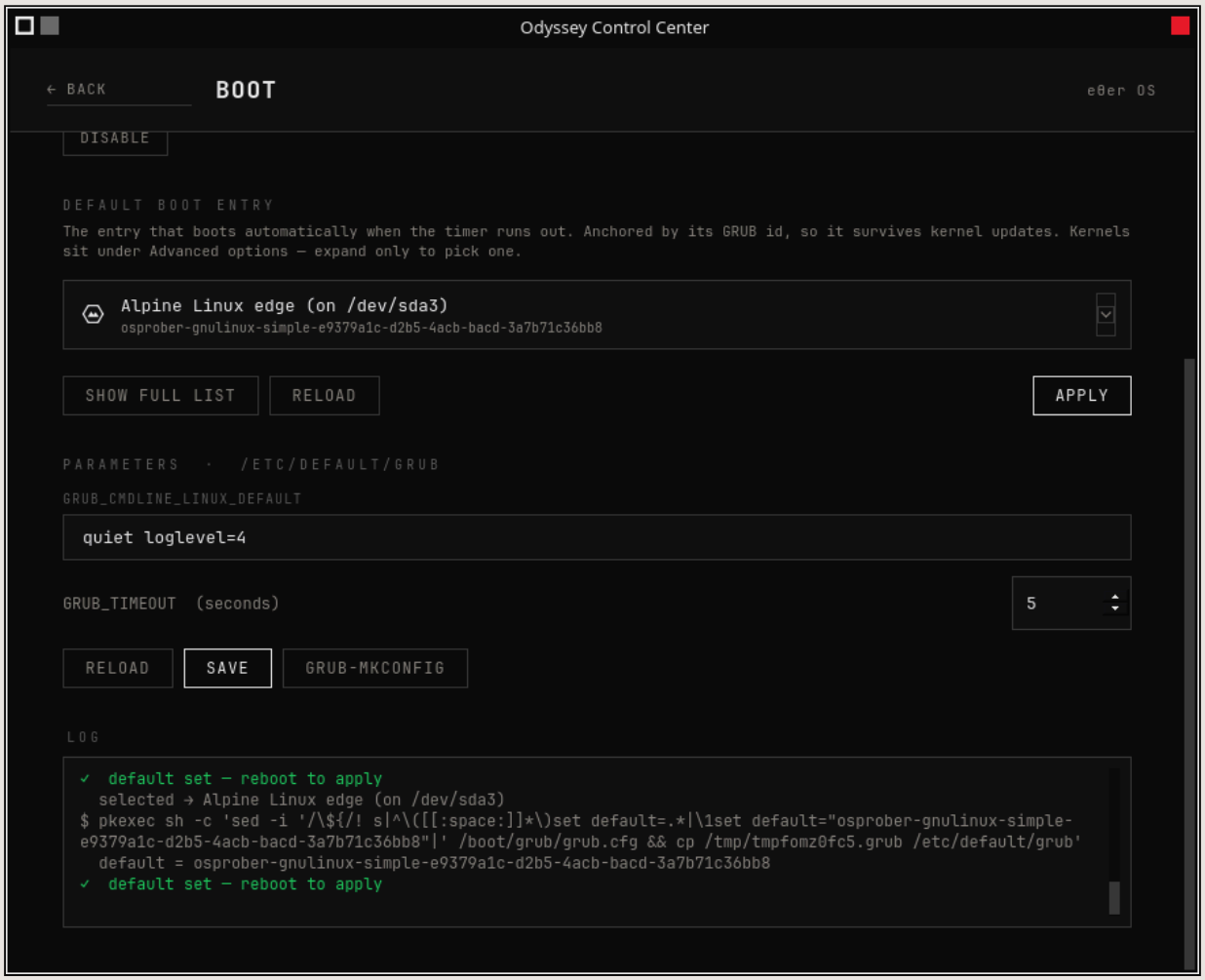

FIG · بعد APPLY: يؤكّد السجلّ «default set — reboot to apply»، ويعرض المحدَّد الآن Alpine Linux افتراضيًا فاعلًا.

WHAT	Boot ← تبويب GRUB ← انقر المدخل الافتراضي الحالي ← اختر غيره من القائمة ← انقر **APPLY**.
WHY	عندك إقلاع ثنائي وتريد أن يبدأ نظام آخر تلقائيًا، أو أنّ تحديث نواة ترك نواةً قديمة محدَّدة وتريد استعادة الأحدث.
RESULT	يُقلِع المدخل المختار تلقائيًا اعتبارًا من التشغيل التالي. والتغيير مثبَّت على المعرّف الثابت لذلك المدخل، فيصمد أمام تحديثات النواة اللاحقة.
RISK	لا خطر على بياناتك — وأسوأ ما قد يحدث أن تختار المدخل الخطأ فتعود لتغيّره من جديد.

وزرّ **SHOW FULL LIST** يعرض كلّ مدخل كما يسرده GRUB نفسه — للقراءة فقط، مجمَّعًا بحيث تجلس متغايرات كلّ نواة تحت سطر «Advanced options» قابل للطيّ بدل أن تزدحم القائمة الرئيسية. أمّا **RELOAD** فيعيد قراءة إعداد GRUB إن ظننت أنّه تغيّر خارج اللوحة.

2.3 جولة — تبويب EFI

انقر تبويب **EFI** في الأعلى. هذا مستوى البرنامج الثابت — أي ما تعرضه قائمة إقلاع لوحتك الأمّ قبل أن يبدأ أيّ نظام تشغيل.

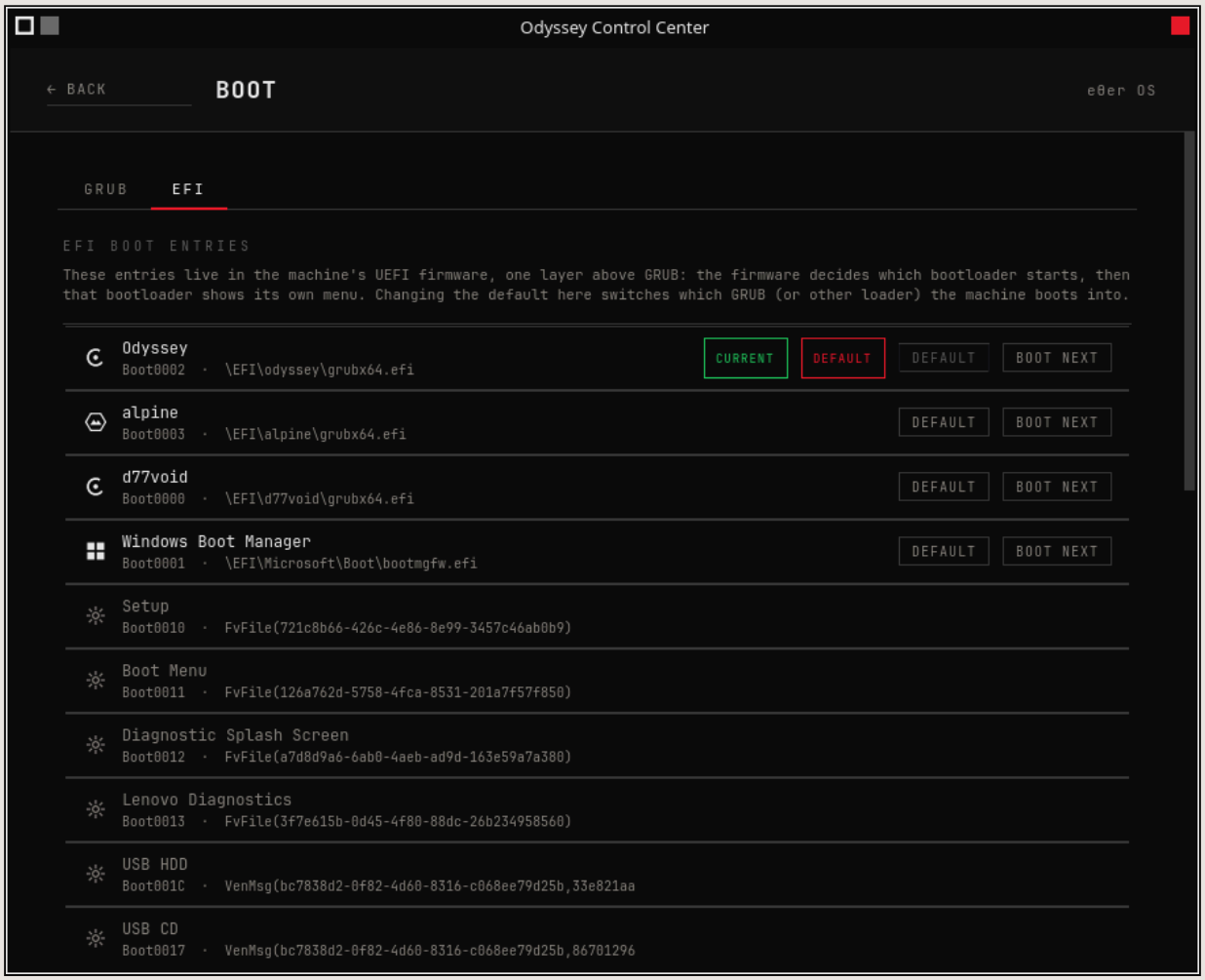

FIG · تبويب EFI: Odyssey موسوم بـCURRENT و DEFAULT بالأخضر والأحمر، ولكلّ من alpine و d77void و Windows Boot Manager زرّا DEFAULT / BOOT NEXT الخاصّان به، مع مدخلات أدوات البرنامج الثابت أدناه.

كلّ سطر مدخل واحد في البرنامج الثابت — باسمه، ومعرّفه (Boot0002 وما شابه)، وملف efi. الذي يشير إليه بالضبط. وزرّان لكلّ سطر:

الزرّ	ماذا يفعل
DEFAULT	يجعل هذا المدخل الافتراضي الدائم لدى البرنامج الثابت — فيبدأ عند كلّ إعادة إقلاع من الآن فصاعدًا.
BOOT NEXT	استثناء لمرّة واحدة — يبدأ هذا المدخل عند إعادة الإقلاع **التالية** فقط، ثمّ يعود الترتيب المعتاد تلقائيًا. وتظهر لافتة في الأعلى ما دام أحدها معلّقًا، ومعها زرّ CANCEL.

WHAT	Boot ← تبويب EFI ← ابحث عن المدخل الذي تريده (مثل «Odyssey» أو «Windows Boot Manager») ← انقر DEFAULT، أو BOOT NEXT لاستثناء لمرّة واحدة.
WHY	إن كانت قائمة إقلاع لوحتك الأمّ تُشغّل محمّل إقلاع خطأ تمامًا — وهو إعداد في البرنامج الثابت لا في GRUB — فهذا هو المكان الوحيد الذي يصلحه.
RESULT	DEFAULT: ينتقل المدخل إلى مقدّمة ترتيب إقلاع البرنامج الثابت بصورة دائمة. و BOOT NEXT: يبدأ مرّةً واحدة ثمّ يعود الوضع.
RISK	منخفض، وقابل للاسترجاع دائمًا من هذا التبويب نفسه أو من شاشة إعداد UEFI في لوحتك الأمّ (وغالبًا بمفتاح Del أو F2 عند التشغيل) إن أقلع شيء على غير المتوقّع.

2.4 معاملات النواة

آخر أقسام تبويب GRUB هو سطر أوامر النواة العادي — يُحرَّر كحقل نصّ عادي، بلا حاجة إلى صياغة GRUB — ومهلة القائمة بالثواني. وزرّ **SAVE** يكتبهما في `/etc/default/grub`؛ وزرّ **GRUB-MKCONFIG** يعيد توليد قائمة الإقلاع الفعلية من ذلك الملف، وهي الخطوة التي تجعل المعامل المحفوظ يسري عند الإقلاع التالي. وتعيد اللوحة دائمًا تطبيق اختيارك الافتراضي الحالي عند إعادة التوليد، فلا يعيد ذلك ضبط النظام الذي يُقلِع افتراضيًا.

> ◂ **متقدّم**
>
> ما بعد هذه النقطة: مسارات ملفّات الإعداد، والأوامر الخام، وما تفعله اللوحة تحت الغطاء.

2.5 متقدّم

ما الذي يجري فعليًا

إعداد GRUB في Odyssey **مسطّح** — بلا كتل `submenu`. فكلّ نواة مدخلٌ `menuentry` في المستوى الأعلى، ما يعني أنّ معرّفًا ثابتًا (لا موضعًا رقميًا) يستطيع دائمًا اختيارها بصورة صحيحة، حتى عبر تحديثات النواة. أمّا تجميع «Advanced options» الذي تراه في المحدِّد فهو تسهيل عرضٍ محض بُني بمطابقة معرّف UUID المشترك بين المدخل البسيط لنظام ما ومتغايراته لكلّ نواة — وملف GRUB نفسه لا يحوي أيّ تجميع كهذا.

وضبط الافتراضي يكتب في موضعين ضمن الإجراء نفسه: القيمة الدائمة `GRUB_DEFAULT` في `/etc/default/grub/`، و — لأنّ `grub.cfg` قد لا يُعاد توليده فورًا — تعديلٌ جراحي مباشر بـ`sed` على سطر `"=set default` `".` الحيّ داخل `/boot/grub/grub.cfg`. نفسه، متجاوزًا فرع `${next_entry}` ذا المرّة الواحدة الذي يستعمله `grub-reboot`. ولهذا يسري الافتراضي عند الإقلاع التالي دون إجبارك على `grub-mkconfig` كاملة أوّلًا.

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

OS-PROBER يدويًا

```
$ sudo xbps-install -y os-prober
# enable scanning — edits GRUB_DISABLE_OS_PROBER in /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 يدويًا — EFIBOOTMGR

تبويب EFI واجهة أمامية لـefibootmgr. وكلّ ما يفعله يقابله أمر واحد:

```
# list all firmware entries, in current boot order
$ sudo efibootmgr -v
# reorder: put entry 0002 first (= DEFAULT)
$ sudo efibootmgr -o 0002,0000,0001,0003
# one-time boot into entry 0003 (= BOOT NEXT)
$ sudo efibootmgr -n 0003
# cancel a pending BootNext
$ sudo efibootmgr -N
```

محمّلات يتيمة

يحدث أحيانًا أن يكتب مثبّتُ محمّل إقلاعه على قسم نظام EFI دون أن يسجّل له مدخلًا في البرنامج الثابت — فيوجد المحمّل على القرص بينما لا يعلم البرنامج الثابت بوجوده. ولوحة Odyssey تمسح قسم ESP مباشرةً وتسرد هذه الحالات على حدة بوصفها **«DISK, NOT IN FIRMWARE ON»**، ولكلّ منها زرّ REGISTER بنقرة واحدة. ويدويًا يكافئ ذلك `efibootmgr -c -d /dev/sdX -p N -L "Name" -l '\\EFI\\name\\bootx64.efi'` — ولاحظ أنّ البرنامج الثابت يضع المدخلات المنشأة حديثًا **أوّلًا** في ترتيب الإقلاع، فأعد تطبيق DEFAULT على مدخلك المعتاد بعدها إن لم ترد أن يستولي عليه.

الملفّات التي تمسّها هذه اللوحة

الملف	الدور
`/etc/default/grub`	سطر أوامر النواة، والمهلة، و GRUB_DEFAULT، و GRUB_DISABLE_OS_PROBER
`/boot/grub/grub.cfg`	قائمة الإقلاع المولَّدة نفسها — لا تحرّرها يدويًا أبدًا؛ فـgrub-mkconfig يكتب فوقها
متغيّرات EFI (NVRAM)	مدخلات إقلاع البرنامج الثابت — وليست ملفًّا أصلًا؛ تُقرأ وتُكتب كلّيًا عبر efibootmgr

النواة

أربعة تبويبات — **SYSCTL** و**CPU GOVERNOR** و**HUGEPAGES** و**MODULES** — إضافةً إلى صفّ من الإعدادات الجاهزة بنقرة واحدة، يجلس فوق الأربعة جميعًا ويسري عليها كلّها دفعةً واحدة.

3.1 جولة — الإعدادات الجاهزة

في أعلى اللوحة تمامًا: إصدار النواة العاملة، والمجدوِل الفاعل، وأربعة أزرار جاهزة — **DESKTOP** و**GAMING** و**SERVER** و**BALANCED**.

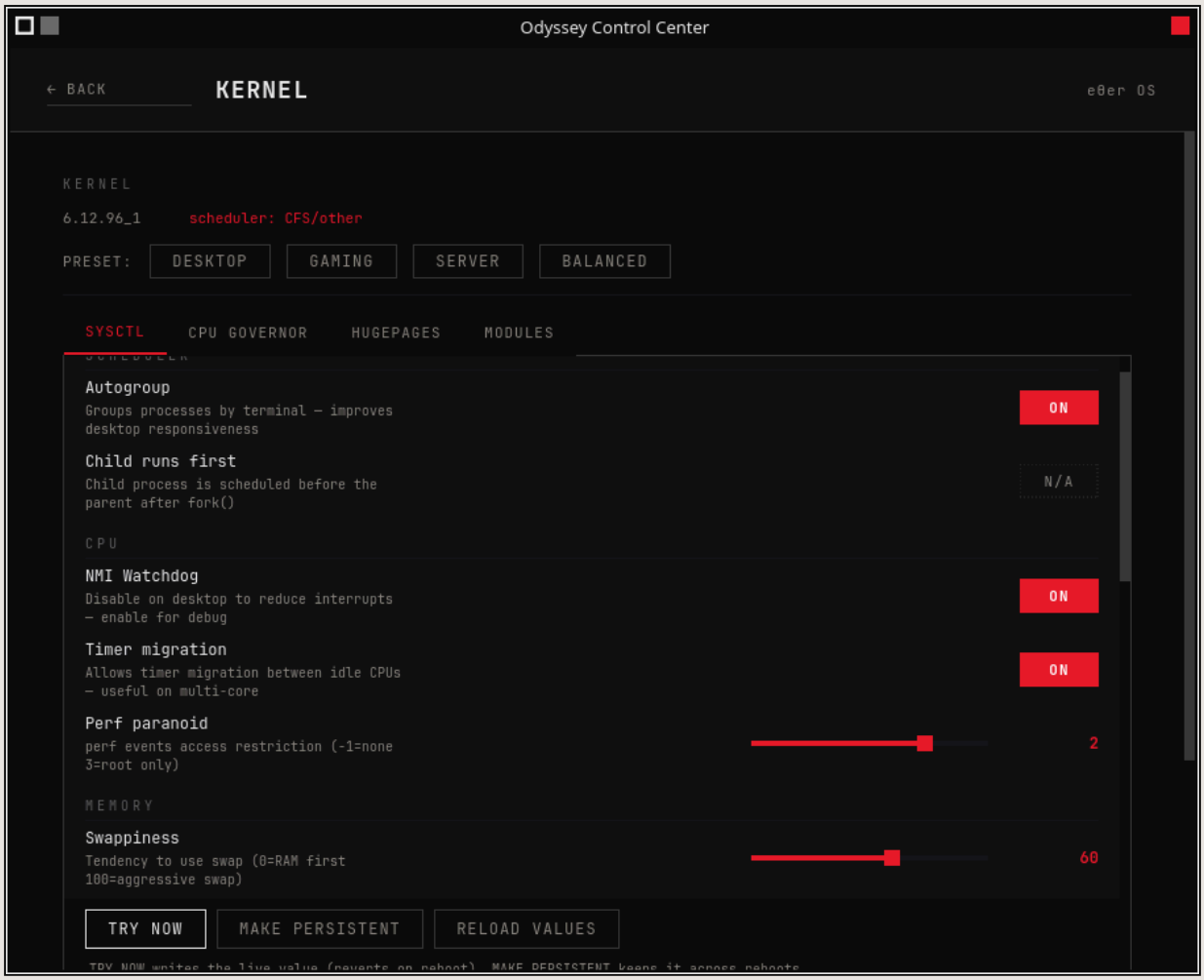

FIG · تبويب SYSCTL بعد فتحه للتوّ: النواة 6.12.96_1، والمجدوِل CFS/other، والإعدادات الجاهزة الأربعة، ومجموعات Scheduler و CPU و Memory أدناه.

الإعداد الجاهز	متى يستحقّ الاستعمال
DESKTOP	آلة يومية — تصفّح وأعمال مكتبية وتعدّد مهامّ خفيف. تبديل أقلّ إلى القرص، وضغط ذاكرة مخبّأة معتدل، وترحيل المؤقّتات مفعَّل لاستهلاك طاقة سلس عند الخمول. وهو الافتراضي المعقول لمعظم الناس.
GAMING	التبديل إلى القرص مدفوعٌ إلى ما يقارب الصفر (أبقِ كلّ شيء في الذاكرة)، وترحيل المؤقّتات مطفأ (فتبقى الأنوية مخصّصة بدل ترحيل العمل بينها)، وقيود perf-event مخفّفة لأدوات القياس مثل MangoHud. استعمله حين تكون أزمنة الإطارات أهمّ من استهلاك الطاقة في الخمول.
SERVER	تسامح أعلى مع التبديل وضغط الذاكرة المخبّأة، وNMI watchdog مفعَّل، وقيود أشدّ على مؤشّرات النواة وسجلّاتها. استعمله على آلة بلا شاشة تقدّم الاستقرار والأمان على سرعة الاستجابة التفاعلية.
BALANCED	حلّ وسط بين Desktop و Gaming — لآلة تكون أحيانًا محطّة عمل وأحيانًا جلسة ألعاب، ولا تريد تبديل الإعدادات جيئةً وذهابًا.

WHAT	Kernel ← انقر DESKTOP أو GAMING أو SERVER أو BALANCED.
WHY	ضبط عشرات قيم sysctl يدويًا مطلبٌ كبير على من يريد ببساطة «إعدادات جيّدة للألعاب» — والإعداد الجاهز هو ذلك بعينه، محسومًا سلفًا.
RESULT	تقفز كلّ قيمة في تبويب SYSCTL إلى أرقام الإعداد الجاهز، مجهَّزةً وظاهرة — ويبقى عليك TRY NOW أو MAKE PERSISTENT (أدناه) لتطبيقها فعلًا.
RISK	لا شيء — فاختيار إعداد جاهز يملأ الاستمارة لا أكثر.

3.2 جولة — SYSCTL، مجموعةً مجموعة

انزل قليلًا وستجد كلّ قيمة قابلة للضبط مجمَّعة تحت عنوان، ولكلّ واحدة وصفٌ بلغة عادية داخل اللوحة نفسها — فلن تحتاج أبدًا إلى معرفة معنى `vm.compaction_proactiveness` قبل أن تقرأه هناك أوّلًا.

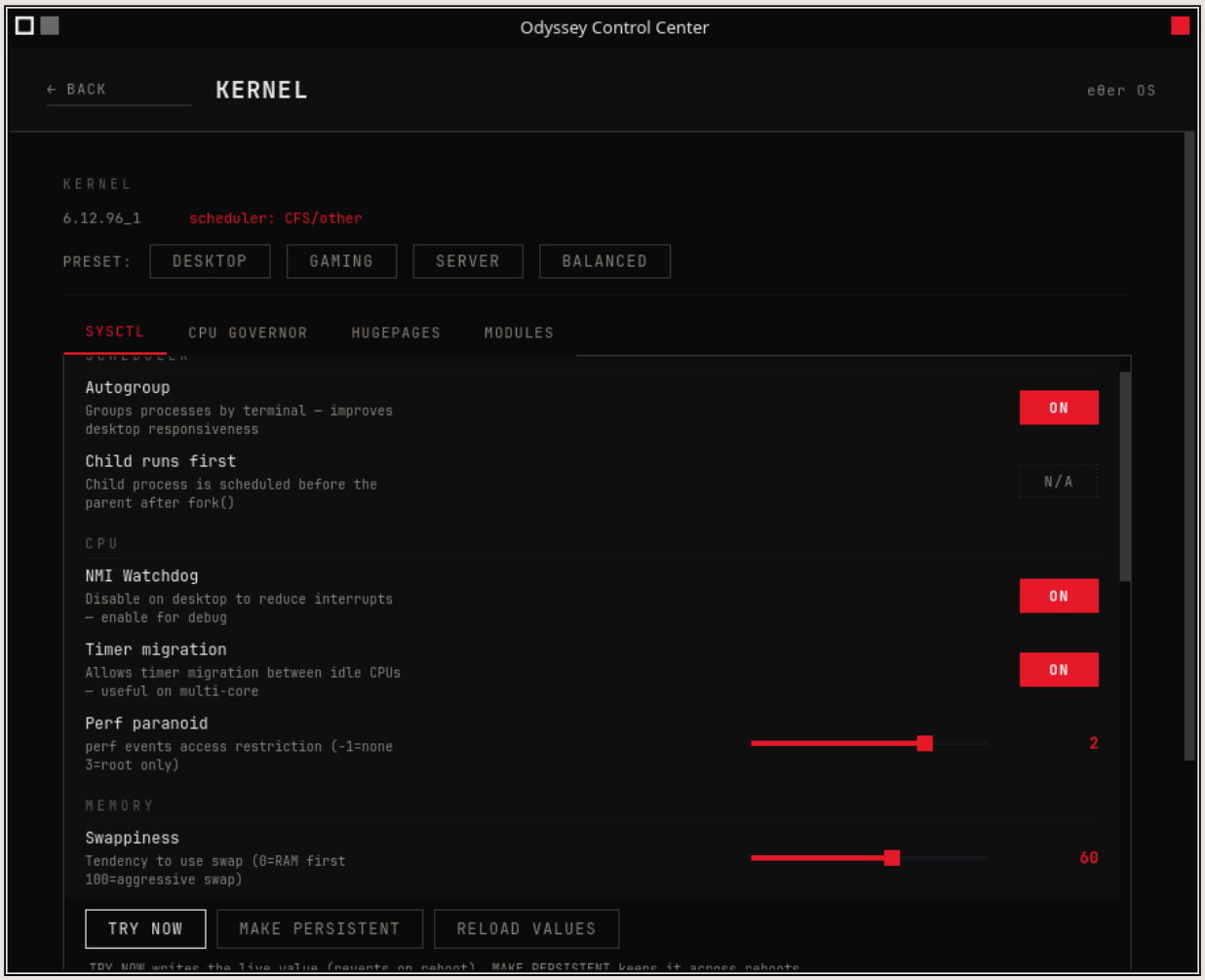

FIG · مجموعتا Scheduler و CPU: Autogroup و Child runs first و NMI Watchdog و Timer migration و Perf paranoid — لكلّ منها مفتاح تشغيل/إطفاء أو منزلق وشرحٌ في سطر واحد.

المجموعة	ما الذي فيها
Scheduler	كيف تُجمَّع العمليات وتُرتَّب أولوياتها
CPU	المراقِب، وترحيل المؤقّتات، وقيود مراقبة الأداء
Memory	التبديل إلى القرص، وضغط الذاكرة المخبّأة، وعتبات كتابة الصفحات المتّسخة
Network	TCP Fast Open، وحجم طابور الانتظار، وحدود مخازن المقابس
Security	هل تُخفى مؤشّرات النواة ورسائل سجلّها عن المستخدمين غير الممتازين

وزرّان في الأسفل يطبّقان تعديلاتك — وهما مختلفان عمدًا:

WHAT	TRY NOW
WHY	لتجرّب قيمةً بلا أيّ خطر من بقائها بعد إقلاع سيّئ.
RESULT	تُكتب حيًّا عبر `sysctl` — فاعلة فورًا، ولهذه الجلسة فقط.
RISK	لا شيء — تعود تلقائيًا عند إعادة الإقلاع.

WHAT	MAKE PERSISTENT
WHY	جرّبت قيمةً (أو إعدادًا جاهزًا كاملًا) وتريدها أن تبقى بعد إعادة الإقلاع.
RESULT	تُكتب في `/etc/sysctl.d/99-odyssey.conf` و تُطبَّق حيًّا في النقرة نفسها.
RISK	منخفض — فهو ملف نصّي؛ احذف السطر أو أعد التطبيق للتراجع.

وزرّ **RELOAD VALUES** يعيد قراءة ما هو فاعل الآن، ملغيًا كلّ ما كُتب ولم يُطبَّق.

3.3 جولة — منظّم المعالج

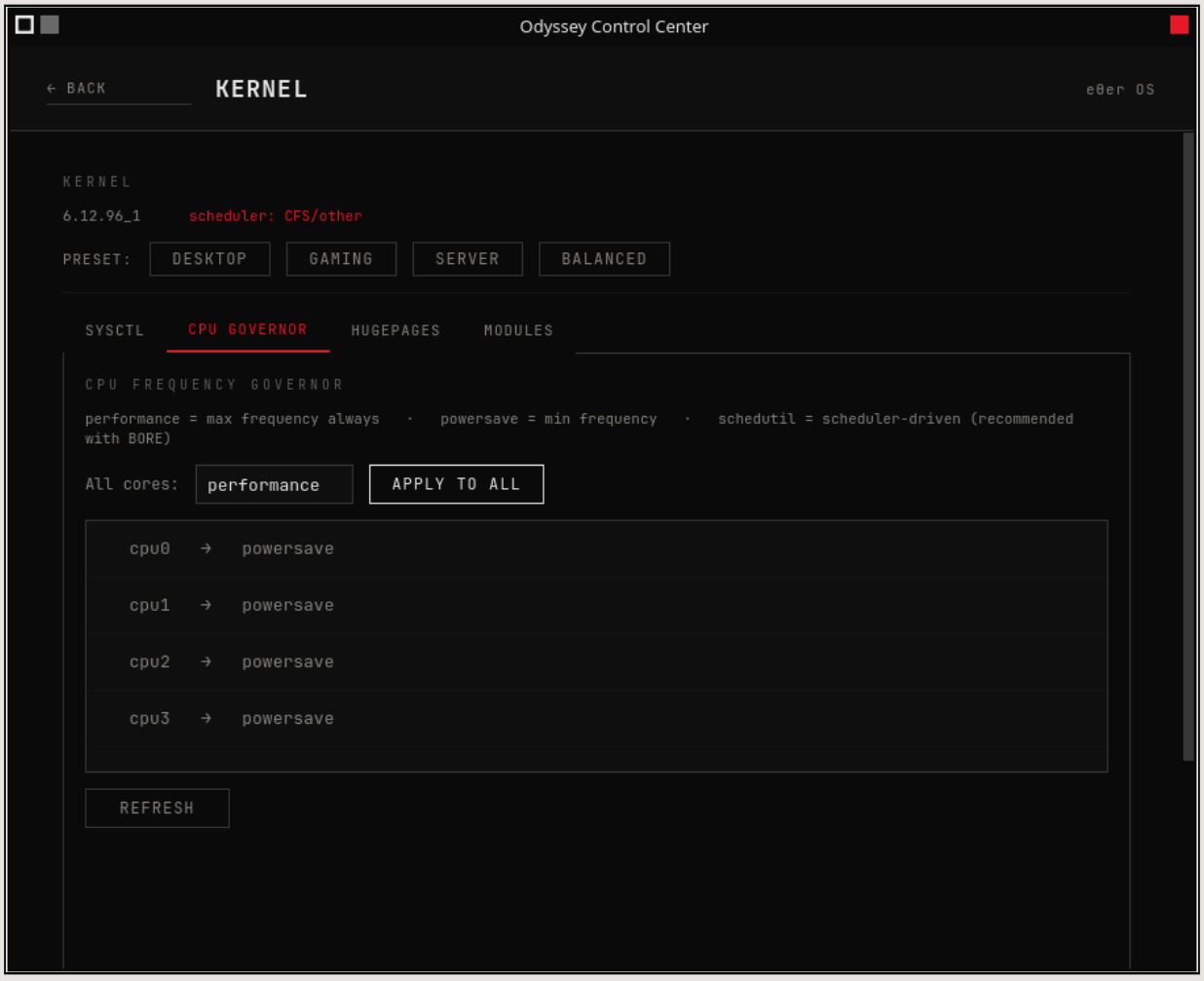

FIG · تبويب «CPU GOVERNOR: «All cores مضبوط على performance مع APPLY TO ALL، وقائمة الأنوية أدناه ما تزال كلّها تعرض powersave — تعارضٌ ينتظر الإصلاح.

المنظّم يقرّر كم يغيّر معالجك تردّده بجرأة. وقائمة منسدلة تطبّق منظّمًا واحدًا على كلّ الأنوية دفعةً واحدة؛ أمّا القائمة تحتها فتعرض المنظّم **الحيّ** لكلّ نواة على حدة — وهكذا تلتقط نواةً عالقة على إعداد خطأ، تمامًا كما في اللقطة أعلاه حيث تقول «All cores» إنّها performance بينما كلّ نواة أدناه ما تزال تقرأ powersave، أي أنّ APPLY TO ALL لم يُنقر بعد.

المنظّم	السلوك
performance	أقصى تردّد دائمًا — أسرع استجابة وأعلى استهلاك
powersave	أدنى تردّد دائمًا — أهدأ وأبطأ استجابة
schedutil	تردّد يقوده تصوّر المجدوِل نفسه للحمل — وهو الموصى به مع مجدوِل BORE

ولا يظهر في القائمة المنسدلة إلّا المنظّمات التي يوفّرها معالجك ونواتك فعلًا.

3.4 جولة — الصفحات الضخمة

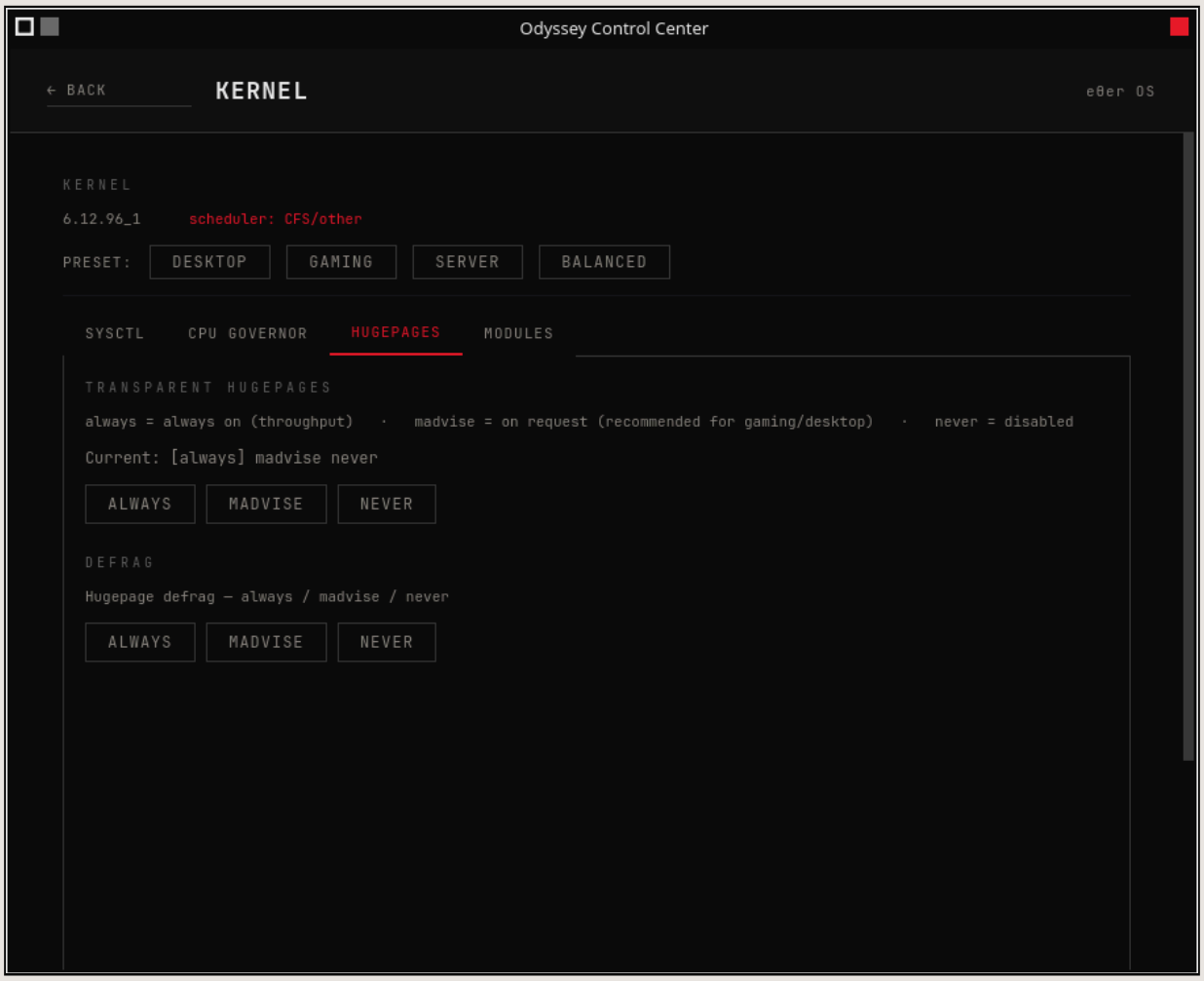

FIG · تبويب HUGEPAGES: الصفحات الضخمة الشفّافة تعرض «Current: [always] madvise never» مع إبراز ALWAYS، وصفّ الأزرار الثلاثة نفسه مكرّرًا أدناه لـDefrag.

إعدادان مستقلّان، كلّ منهما صفّ من ثلاثة أزرار تُطبَّق لحظة نقر أحدها — والقيمة الحيّة الحالية معروضة دائمًا فوق الأزرار بين معقوفتين، مثل `madvise never [always]` في اللقطة.

الإعداد	ما الذي يتحكّم به
Transparent hugepages	ALWAYS = مفعّل في كلّ مكان (أفضل إنتاجية)؛ MADVISE = فقط حين يطلب تطبيقٌ ذلك صراحةً (وهو الموصى به للألعاب وسطح المكتب)؛ NEVER = مطفأ تمامًا

الإعداد	ما الذي يتحكّم به
Defrag	كم يجتهد النواة في ضغط الذاكرة لـ**تكوين** صفحات ضخمة عند الطلب — الخيارات الثلاثة نفسها، ومستقلّة عن الإعداد أعلاه

3.5 جولة — الوحدات

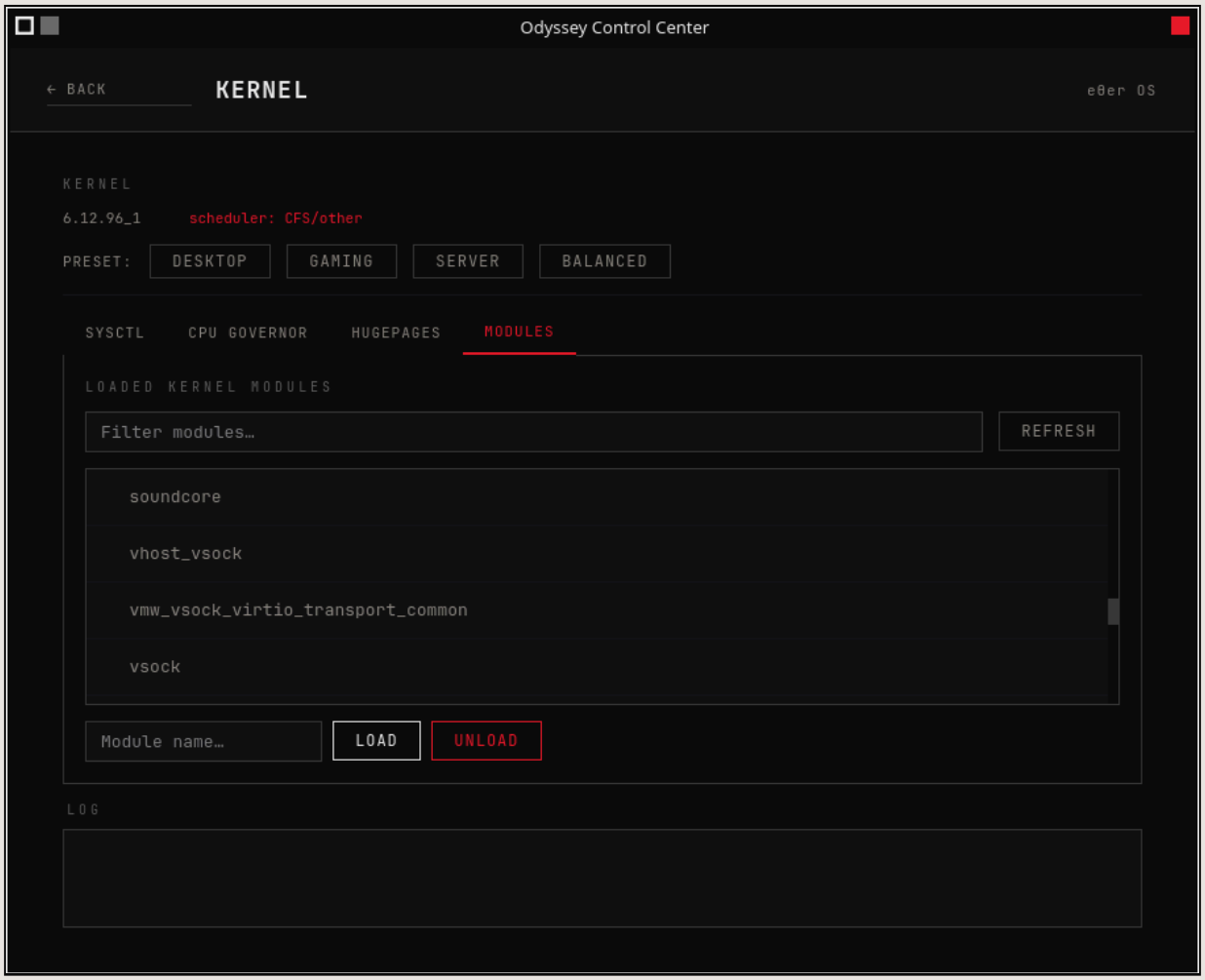

FIG • تبويب MODULES: صندوق ترشيح، وقائمة حيّة (soundcore و vhost_vsock و vsock…)، وصفّ LOAD/UNLOAD في الأسفل لوحدة تكتب اسمها.

قائمة حيّة قابلة للترشيح بكلّ وحدة نواة محمَّلة حاليًا — اكتب في المرشِّح لتضييقها، واضغط REFRESH لإعادة القراءة. وبخلاف التبويبات الثلاثة الأخرى، يتيح لك هذا التبويب التصرّف مباشرةً: اكتب اسم وحدة ثمّ انقر **LOAD** أو **UNLOAD**.

WHAT	Kernel ← تبويب MODULES ← اكتب اسم وحدة ← LOAD أو UNLOAD.
WHY	أحيانًا تحتاج إلى تفعيل مشغّل أو وحدة ميزة بعينها (أو إزالتها) دون إعادة إقلاع — لاختبار عتاد، أو لتحرير جهاز تمسكه وحدة أخرى.
RESULT	تُدرَج الوحدة في النواة العاملة أو تُزال منها فورًا.
RISK	متوسّط في حالة UNLOAD تحديدًا — فإزالة وحدة يعتمد عليها شيء آخر قد تزعزع ما كان يستعملها (مشغّل عرض، أو نظام ملفّات). أمّا LOAD فآمن عمومًا؛ إذ ترفض النواة تحميل وحدة غير متوافقة رفضًا صريحًا بدل تطبيقها نصف تطبيق.

◂ متقدّم

ما بعد هذه النقطة: مسارات ملفّات الإعداد، والأوامر الخام، وما تفعله اللوحة تحت الغطاء.

3.6 متقدّم

مفاتيح SYSCTL بالضبط

المفتاح	التسمية في اللوحة
`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

قيم الإعدادات الجاهزة: DESKTOP مقابل 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
# the full preset tables are read directly from panel_kernel.py —
# open the Control Center source if you want every field
```

يدويًا، بلا اللوحة

```
# live, temporary
$ sudo sysctl vm.swappiness=10
# persistent — same file the panel writes
$ echo “vm.swappiness=10” | sudo tee -a /etc/sysctl.d/99-odyssey.conf
$ sudo sysctl -system
# CPU governor, per core
$ 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
```

لماذا TRY NOW و MAKE PERSISTENT نداءان منفصلان

كتابات `sysctl` حيّة فقط بطبيعتها — فلا دوام مدمجًا في النواة. و MAKE PERSISTENT هو اختيار Odyssey لملف معياري واحد، `/etc/sysctl.d/99-odyssey.conf`، وتنفيذ الكتابة الحيّة والكتابة في الملف معًا كي لا يفترق الاثنان أبدًا. وتحرير ذلك الملف مباشرةً ثمّ تشغيل `sysctl --system` بعده يكافئ نقر الزرّ وظيفيًا تمامًا.

الذاكرة والتبديل

شاشة واحدة وثلاثة أقسام: ملخّص للقراءة فقط، ثمّ swappiness، ثمّ ZRAM.

4.1 جولة

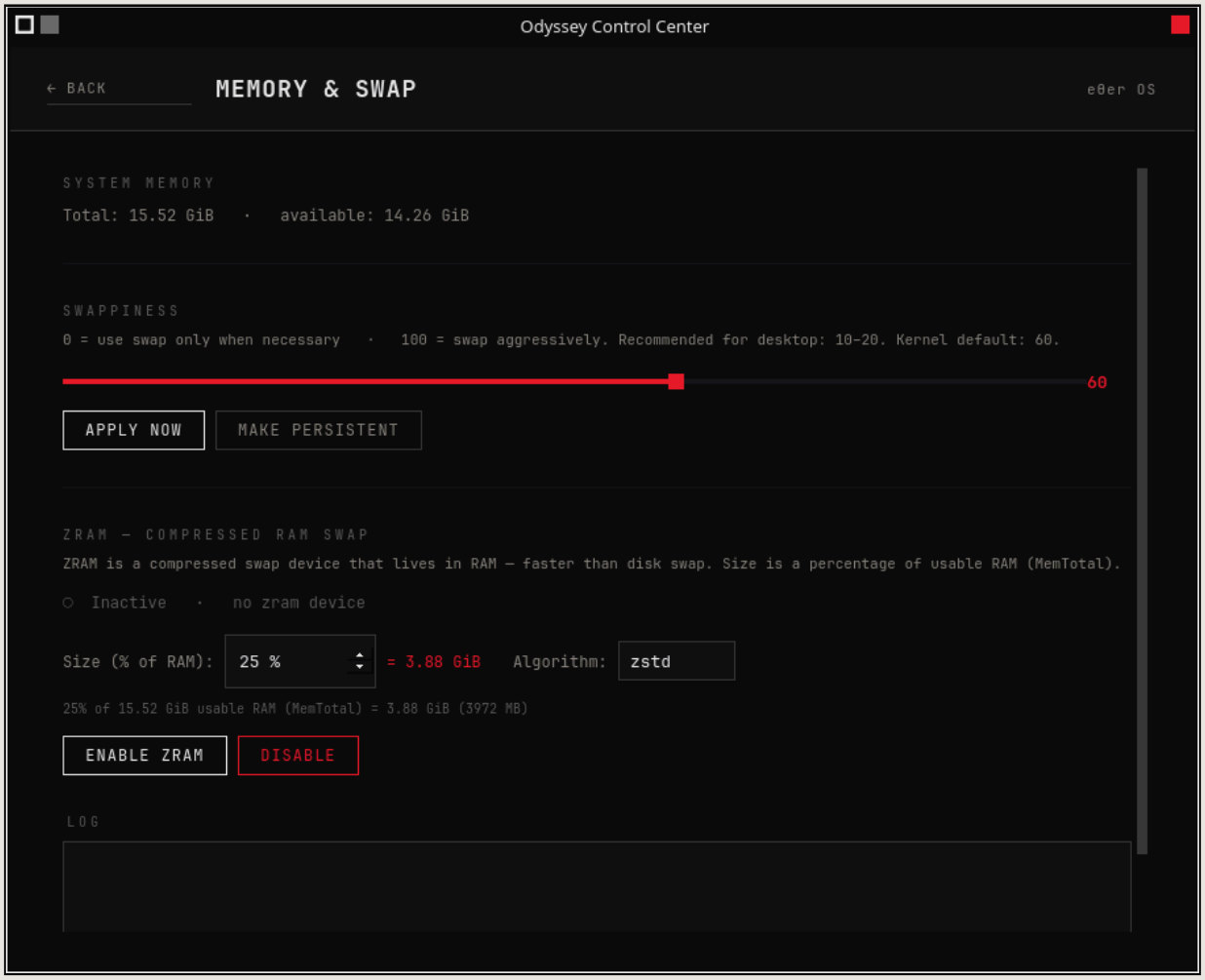

FIG · اللوحة كاملة: ذاكرة النظام (15.52 غيبي بايت إجمالًا، و14.26 غيبي بايت متاحة)، ومنزلق Swappiness عند 60 مع APPLY NOW / MAKE PERSISTENT، وZRAM أدناه يعرض «Inactive · no zram device» مع الحجم والخوارزمية وزرّي ENABLE/DISABLE.

ذاكرة النظام

مجرّد قراءة في الأعلى — إجمالي الذاكرة وما هو متاح منها الآن. لا عناصر تحكّم؛ فهي سياقٌ للقسمين التاليين.

SWAPPINESS

منزلق واحد من 0 إلى 100. فالصفر يعني «لا تبدّل إلى القرص إلّا حين لا يبقى خيار آخر حقًّا»؛ والمئة تعني «بدّل بجرأة، قبل أن تمتلئ الذاكرة بكثير». والقيمة الافتراضية للنواة هي 60 (وهي ما تعرضه اللقطة)؛ أمّا تلميح اللوحة نفسه فيقترح 10–20 لسطح المكتب — وهو الرقم ذاته الذي يضبطه الإعداد الجاهز DESKTOP في لوحة Kernel.

WHAT	اسحب المنزلق ← APPLY NOW للتجربة، ثمّ MAKE PERSISTENT حين ترضى عن القيمة.
WHY	القيمة الأدنى تُبقي ما تستعمله فعلًا في الذاكرة السريعة بدل دفعه إلى تبديل أبطأ — وهو ما تريده عادةً إن كانت الذاكرة المركّبة كافية.
RESULT	APPLY NOW يغيّرها حيًّا وفورًا، لهذه الجلسة فقط. وMAKE PERSISTENT يكتبها في ملف إعداد فتبقى بعد إعادة الإقلاع أيضًا.
RISK	منخفض جدًّا. لكنّ قيمةً شديدة الانخفاض على آلة تعاني فعلًا من قلّة الذاكرة قد تجعلها أميل إلى نفاد الذاكرة فجأةً تحت حمل ثقيل بدل التبديل بهدوء — وإن حدث ذلك فارفعها من جديد.

ZRAM — تبديل مضغوط داخل الذاكرة

ZRAM جهاز تبديل يعيش **داخل** الذاكرة نفسها، مضغوطًا — وهو أسرع بفارق معتبر من التبديل إلى قرص لأنّه لا قرص في الأمر أصلًا. والحجم نسبةٌ مئوية من ذاكرتك الصالحة للاستعمال، لا رقمًا ثابتًا، فيتدرّج الإعداد بعقل بين الآلات المختلفة؛ وقراءة حيّة بجانب حقل الحجم تحوّل النسبة إلى غيبي بايت نيابةً عنك (وفي اللقطة: 25% = 3.88 غيبي بايت على نظام سعته 15.52).

WHAT	اضبط نسبة الحجم واختر خوارزمية (zstd أو lz4 أو lzo أو deflate) ← ENABLE ZRAM.
WHY	التبديل المضغوط داخل الذاكرة يمتصّ ضغط الذاكرة بخسارة أداء أقلّ بكثير من التبديل إلى القرص — وهو مفيد على أيّ آلة، ولا سيّما ذات الذاكرة المحدودة.
RESULT	يُنشَأ جهاز `zram0` ويُفعَّل بالحجم والخوارزمية المختارين؛ ويتحوّل سطر الحالة من «Inactive» إلى فاعل.
RISK	منخفض. وzstd (الافتراضية) توازن جيّدًا بين نسبة الضغط وكلفة المعالج على معظم الآلات. وزرّ DISABLE يطفئه بنظافة في أيّ وقت.

متقدّم

ما بعد هذه النقطة: مسارات ملفّات الإعداد، والأوامر الخام، وما تفعله اللوحة تحت الغطاء.

4.2 متقدّم

ما الذي تعنيه «التبعية للعتاد» هنا فعلًا

الإعدادان يتفاعلان مع عتادك بعينه، لا مع قاعدة «ذاكرة أكثر = أفضل» وحدها:

- **قرص NVMe سريع:** قيمة swappiness في نطاق 10–20 لا تكلّفك شيئًا يُذكر حتى لو حدث تبديل — فالقرص سريع بما يجعل الأمر بالكاد ملحوظًا. ويبقى ZRAM متفوّقًا في زمن الاستجابة وفي تقليل تآكل القرص، لكنّ الفارق أصغر منه على قرص دوّار.

- **قرص دوّار أو eMMC للتبديل**: أبقِ swappiness منخفضة (0–10) — فالتبديل على وسائط بطيئة يسبّب التقطّع الذي يسمّيه الناس «thrashing». وZRAM مكسب أكبر بكثير هنا تحديدًا.
- **سعة المعالج**: يقايض ZRAM دخل القرص وخرجه بدورات معالج تُنفَق في الضغط وفكّه. وعلى آلة مقيَّدة بالمعالج (عتاد قديم، أو آلة مشغولة أصلًا أثناء اللعب) تضغط `lz4` أسرع من `zstd` بنسبة أسوأ — وهي مقايضة معقولة إن كان وقت المعالج هو المورد الأضيق. أمّا على أيّ عتاد حديث فنسبة `zstd` الأفضل تستحقّ كلفتها الأعلى قليلًا.
- **سعة الذاكرة**: نسبة ZRAM كبيرة (مثل 50%) على آلة بذاكرة 8 غيبي بايت فقط تترك ذاكرة حقيقية أقلّ للتطبيقات، لأنّ ZRAM نفسه يستهلك ذاكرة حقيقية لتخزين الصفحات المضغوطة. أمّا على 16 غيبي بايت فأكثر فنسبة 25% افتراضٌ مريح مع هامش وافر.

يدويًا

```
# swappiness, live
$ sudo sysctl vm.swappiness=15
# swappiness, persistent — same file the panel writes
$ echo "vm.swappiness=15" | sudo tee /etc/sysctl.d/99-odyssey.conf
# zram, manual setup
$ 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
# check it's active
$ swapon -show
```

> **لماذا تعيد اللوحة تهيئة ZRAM0 بدل تغيير حجمه حيًا**
>
> لا يمكن تغيير حجم جهاز ZRAM وهو فاعل ومستعمَل للتبديل — إذ تشترط النواة إيقاف التبديل عنه وإعادة تهيئته أوّلًا. وزرّ ENABLE ZRAM في اللوحة ينفّذ هذا التسلسل تلقائيًا (swapoff، ثمّ reset، ثمّ إعادة الإعداد، ثمّ swapon)، ولهذا يظهر تغيير الحجم أو الخوارزمية على جهاز فاعل أصلًا إجراءً واحدًا نظيفًا بدل رسالة خطأ.

جدار الحماية

للوحة Firewall فصلٌ كامل مخصَّص لها في **دليل الأمان والتحصين** — مجموعة القواعد مشروحةً سطرًا سطرًا، والإعدادات الجاهزة الواحد والعشرون كلّها بمخاطر كلّ منها، ومحرّر الإعداد الخام، ومكافئات سطر الأوامر. وهذا الفصل هو نسخة الجولة المصوّرة.

5.1 جولة

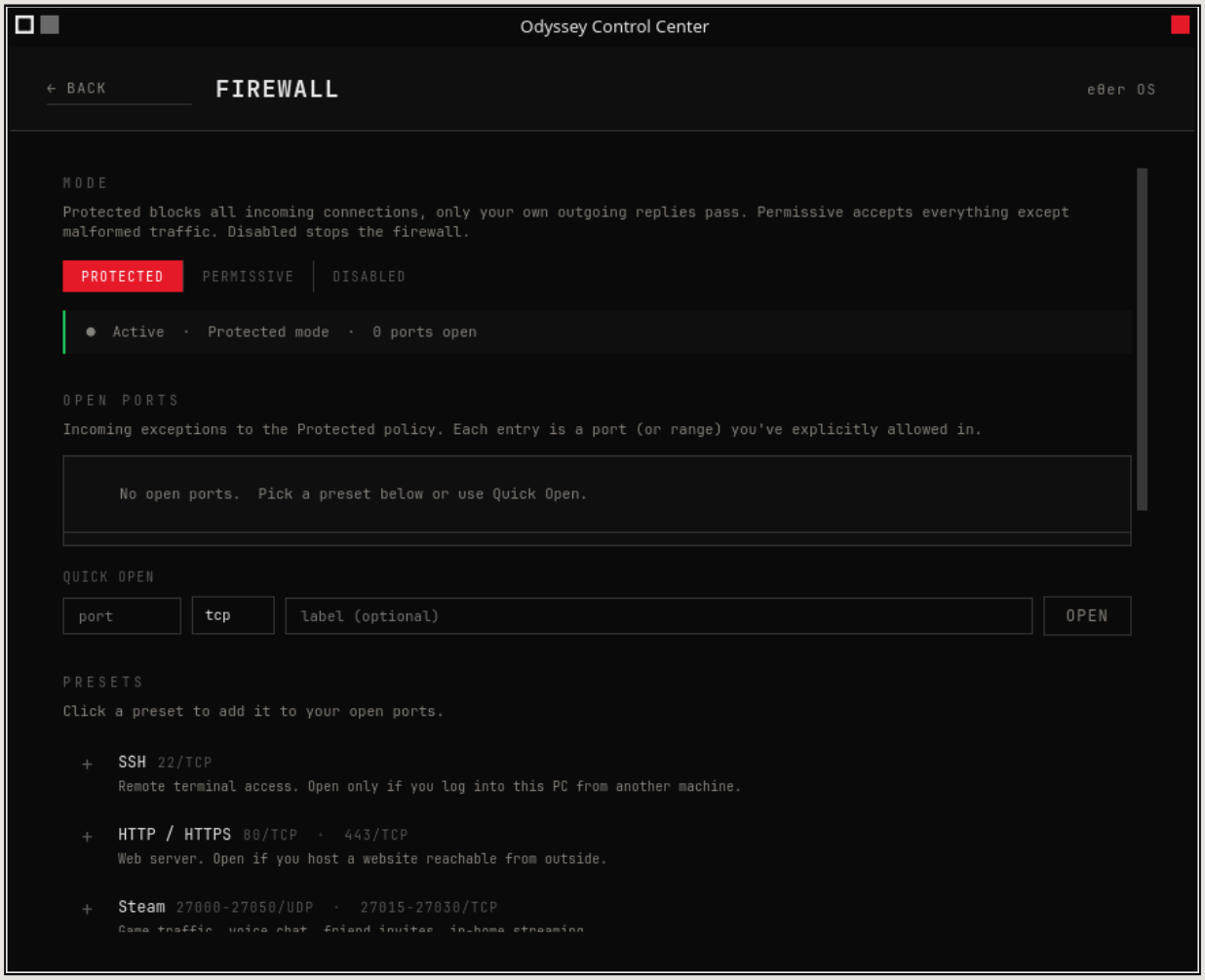

FIG · وضع PROTECTED فاعل، و«0 ports open»، وحقول Quick Open، وبلاطات الإعدادات الجاهزة تبدأ بـSSH و HTTP/HTTPS و Steam.

الوضع

ثلاثة أزرار في الأعلى، وتشرح اللوحة كلًّا منها في سطر واحد فوقها مباشرةً:

الوضع	ماذا يفعل
PROTECTED	الافتراضي. يمنع كلّ الاتصالات الواردة؛ ولا يمرّ إلّا ردود اتصالاتك الصادرة.
PERMISSIVE	يقبل كلّ شيء إلّا الحركة المشوَّهة — وهو وضع تشخيصي لسؤال «هل جدار الحماية هو ما يكسر هذا؟»، لا مكانًا للإقامة.
DISABLED	يوقف جدار الحماية كلّيًا.

ويؤكّد سطر الحالة تحته ما هو فاعل بالفعل — وفي اللقطة: **«Active · Protected mode · 0 ports open.»**

المنافذ المفتوحة والإعدادات الجاهزة

OPEN PORTS يسرد استثناءاتك الصريحة — وهو فارغ افتراضيًا كما ترى. و**QUICK OPEN** يأخذ منفذًا وبروتوكولًا ولافتةً اختيارية مباشرةً، لكلّ ما ليس في قائمة الإعدادات الجاهزة. أمّا **PRESETS** أدناه فيعطيك ثماني خدمات شائعة بنقرة واحدة لكلّ منها، ولكلّ واحدة وصفٌ في سطر يبيّن متى تريدها فعلًا (SSH: «وصول بعيد إلى الطرفية، لا تفتحه إلّا إن كنت تلج إلى هذا الجهاز من آلة أخرى»)، مع ثلاث عشرة خدمة إضافية تحت «MORE PRESETS» القابل للتوسيع.

WHAT	Firewall ← انقر بلاطة إعداد جاهز، أو املأ QUICK OPEN ← SAVE & APPLY.
WHY	أنت تشغّل شيئًا — خادوم ألعاب، أو خادوم وسائط، أو خادوم تطوير — تحتاج آلة أخرى إلى بلوغه.
RESULT	يُفتح ذلك المنفذ؛ ويبقى كلّ ما عداه مغلقًا بحكم سياسة PROTECTED.
RISK	يعتمد كلّيًا على الإعداد الذي اخترته — راجع ملاحظات المخاطر لكلّ إعداد في دليل الأمان والتحصين قبل أن تفتح أيّ شيء يتجاوز SSH أو HTTP/HTTPS على آلة يمكن بلوغها من الإنترنت.

ولا يسري شيء حتى تضغط **SAVE & APPLY** — فالتعديلات تُجهَّز أوّلًا، لتراجع بالضبط ما أنت مقدم على فتحه قبل أن تلتزم به.

◂ متقدّم

ما بعد هذه النقطة: مسارات ملفّات الإعداد، والأوامر الخام، وما تفعله اللوحة تحت الغطاء.

5.2 متقدّم

المرجع التقني الكامل — مجموعة قواعد `nftables.conf` الافتراضية مشروحةً قاعدةً قاعدة، وكلّ واحد من الإعدادات الجاهزة الواحد والعشرين بانكشافه المحدَّد وسبل تخفيفه، ووصفات حصر الخدمة بالشبكة المحلية، ومكافئات `nft` في سطر الأوامر، ومعالجة جلسة SSH محبوسة خارجًا — يعيش في **الفصل 2 من دليل الأمان والتحصين** (وهو في `/usr/share/doc/odyssey/` على نظامك المثبَّت). وهذا الكتيّب لا يكرّره عمدًا؛ فعُدّ ذلك الفصل امتدادًا لهذا القسم.

```
# the one command worth knowing right now:
$ sudo nft list ruleset
# shows exactly what's loaded in the kernel, independent of the panel
```

AppArmor

خمسة تبويبات: **STATUS** و**PROFILES** و**COMMUNITY** و**VIOLATIONS** و**EDITOR**. والإشارة هنا كما في Firewall: المرجع العميق — ما هو التحكّم الإلزامي في الوصول فعلًا، ودورة التقييد كاملةً، ومثالٌ مشروح — يعيش في دليل الأمان والتحصين. وهذا هو الجولة المصوّرة.

6.1 جولة — STATUS

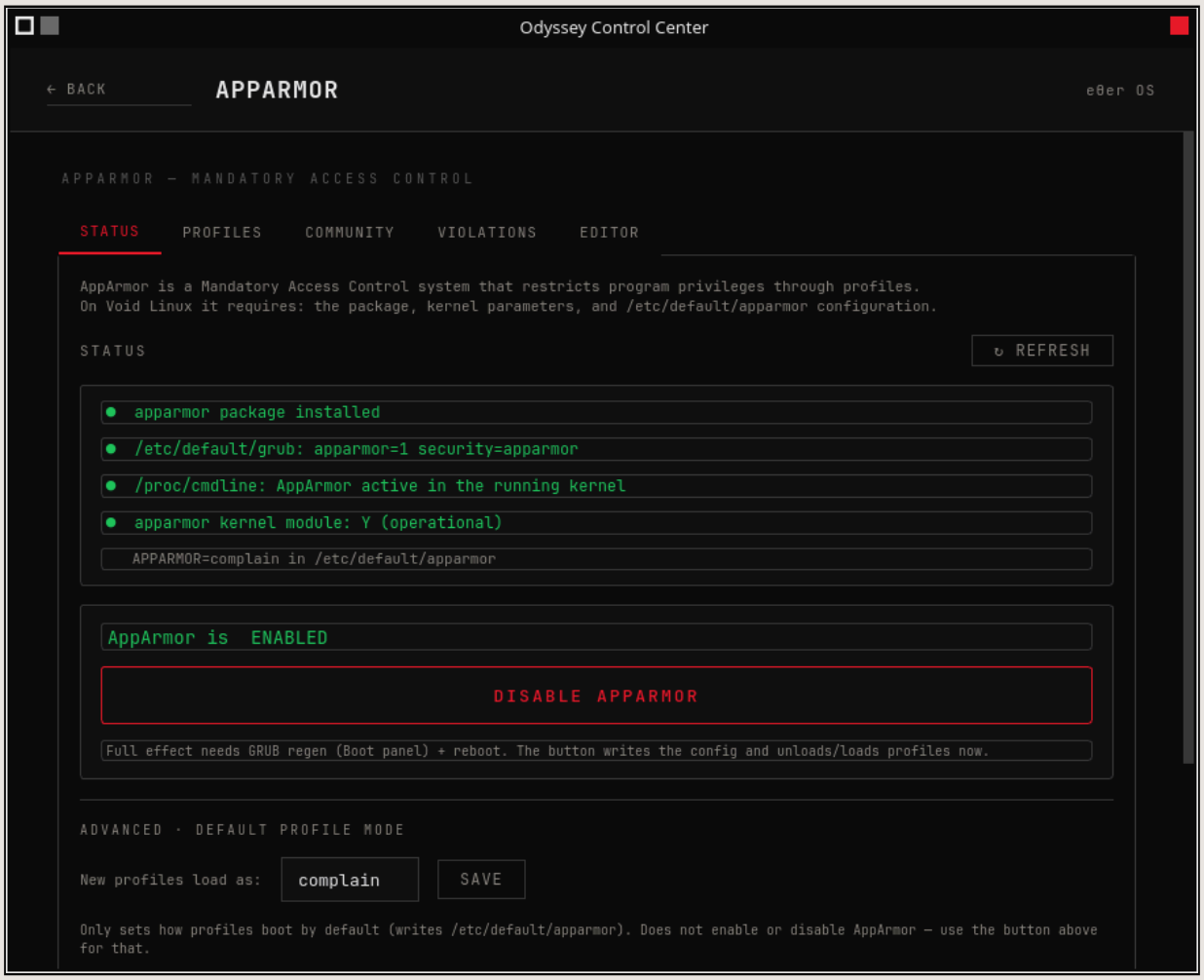

FIG · تبويب STATUS: أربعة فحوص خضراء (الحزمة مثبّتة، ومعاملات GRUB، والنواة العاملة، ووحدة النواة)، و«AppArmor is ENABLED» مع زرّ أحمر DISABLE APPARMOR، ووضع الملفات التعريفية الافتراضي مضبوط على complain.

أربعة فحوص تتحقّق من أنّ الإطار يعمل فعلًا، من أعلى إلى أسفل: حزمة apparmor مثبّتة، ومعاملات نواة GRUB مضبوطة، والنواة **العاملة** أقلعت بها حقًّا (وهذا هو الفحص الذي قد يتأخّر إن لم تعد الإقلاع بعد التفعيل)، ووحدة النواة نفسها تعمل. وتحت ذلك: زرّ

كبير للتفعيل والتعطيل، وقسم **ADVANCED · DEFAULT PROFILE MODE** — أي هل تبدأ الملفات التعريفية المحمَّلة حديثًا في complain أم في enforce.

WHAT	STATUS ← AppArmor ← تأكّد من أنّ الفحوص الأربعة خضراء.
WHY	إن كان أيّ فحص أحمر أو كهرمانيًّا فلن يعمل شيء آخر في هذه اللوحة كما ينبغي — وهذا أوّل ما يُنظر إليه دائمًا حين يبدو شيء في AppArmor غير سليم.
RESULT	يقين بأنّ الإطار فاعل حقًّا، لا مثبَّت فحسب.
RISK	لا شيء — تحقّق للقراءة فقط.

6.2 جولة — PROFILES

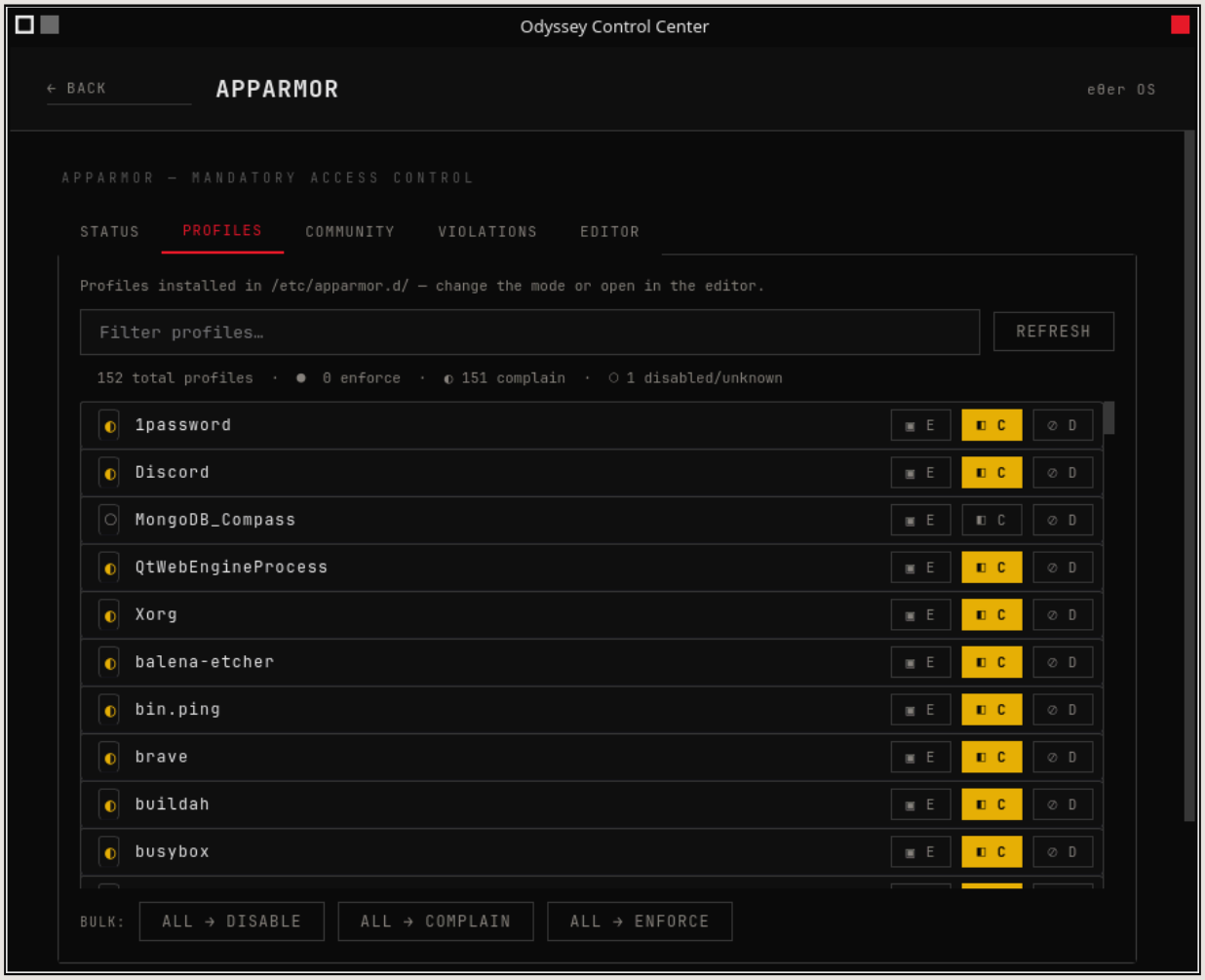

FIG · تبويب PROFILES: 152 ملفًا تعريفيًا إجمالًا، 0 في enforce، و151 في complain، وواحد معطّل/مجهول — قائمة قابلة للترشيح فيها 1password و Discord و MongoDB_Compass وغيرها، ولكلّ صفّ أزرار D / C / E، مع أزرار جماعية ALL→ENFORCE / ALL→COMPLAIN / ALL→DISABLE في الأسفل.

كلّ ملف تعريفي محمَّل في صفّ، مع صندوق ترشيح وعدّاد جارٍ في الأعلى (وفي اللقطة: 152 ملفًا، جميعها إلّا واحدًا في وضع complain — وهو افتراضي Odyssey المشحون). وثلاثة أزرار لكلّ صفّ:

الزرّ	الأثر
E	نقل هذا الملف إلى enforce — فتُمنَع المخالفات لا تُسجَّل فقط
C	النقل إلى complain — تُسجَّل المخالفات ولا يُمنَع شيء
D	تعطيل الملف كلّيًا — فيعمل البرنامج بلا تقييد

وأزرار **BULK** الثلاثة في الأسفل تطبّق التبديل نفسه على كلّ ملف في القائمة دفعةً واحدة — وهي مفيدة بُعيد تثبيت دفعة من COMMUNITY، لكن عامِل ALL→ENFORCE بحذر خاصّ؛ انظر كتلة الإجراء في القسم التالي.

6.3 جولة — COMMUNITY

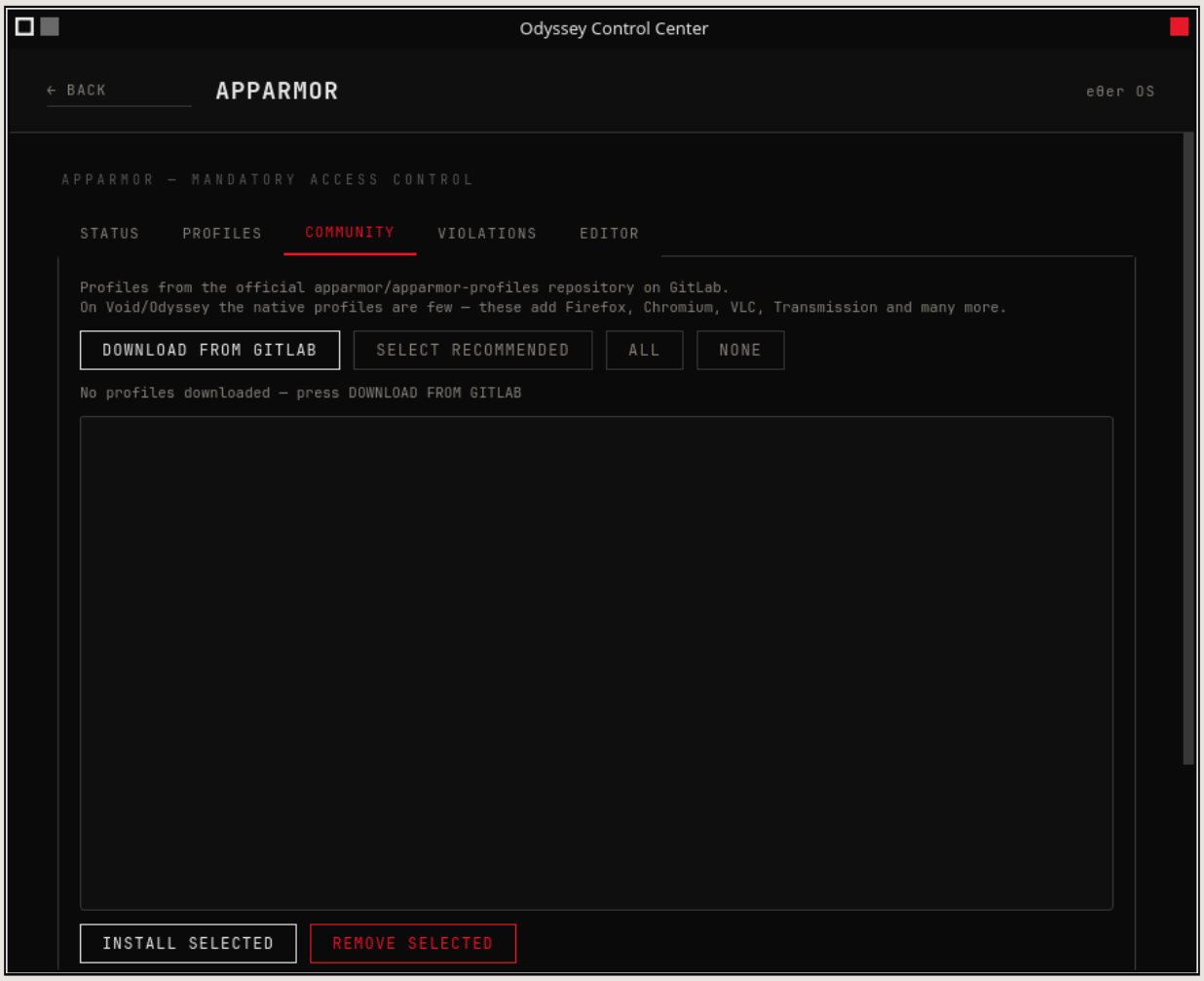

FIG · تبويب COMMUNITY: «No profiles downloaded — press DOWNLOAD FROM GITLAB»، مع أزرار DOWNLOAD FROM GITLAB و SELECT RECOMMENDED و ALL و NONE، وزرّي INSTALL SELECTED / REMOVE SELECTED في الأسفل.

AppArmor القياسي يقيّد مكوّنات النظام غالبًا — أمّا تطبيقات سطح المكتب التي تقلقك فعلًا (المتصفّحات، وتطبيقات المحادثة، ومشغّلات الوسائط) فتحتاج إلى ملفات تعريفية من المجتمع. وهذا التبويب يجلبها مباشرةً من مستودع `apparmor/apparmor-profiles` الرسمي على GitLab.

1 **DOWNLOAD FROM GITLAB**

يجلب القائمة الحالية للملفات المتاحة — ولا يُثبِّت شيء بعد، فهذه الخطوة تملأ القائمة فقط.

2 SELECT RECOMMENDED (أو ALL / NONE)

يؤشّر مجموعةً افتراضية معقولة تناسب سطح المكتب — متصفّحات عائلة Firefox، وChromium، وVLC، وTransmission وما شابهها.

3 INSTALL SELECTED

يثبّت الملفّات المؤشَّرة — وتُحمَّل في وضع complain، كسائر الملفّات، جاهزةً لدورة «راقب ثمّ افرض» أدناه.

WHAT	SELECT RECOMMENDED ← DOWNLOAD FROM GITLAB ← COMMUNITY ← AppArmor (أو اختر يدويًا) ← INSTALL SELECTED.
WHY	التطبيقات الأكثر تعرّضًا للمدخلات غير الموثوقة — متصفّح يعرض صفحة خبيثة، أو مشغّل وسائط يفتح ملفًّا مصنوعًا بعناية — هي بالضبط التي يحميها الملف التعريفي أفضل حماية.
RESULT	تظهر ملفّات تعريفية جديدة في PROFILES، في وضع complain، تسجّل ما كانت ستمنعه دون أن تمنع شيئًا بعد.
RISK	لا شيء عند التثبيت — فلا شيء يُفرَض حتى تأمر بذلك في PROFILES.

6.4 جولة — VIOLATIONS

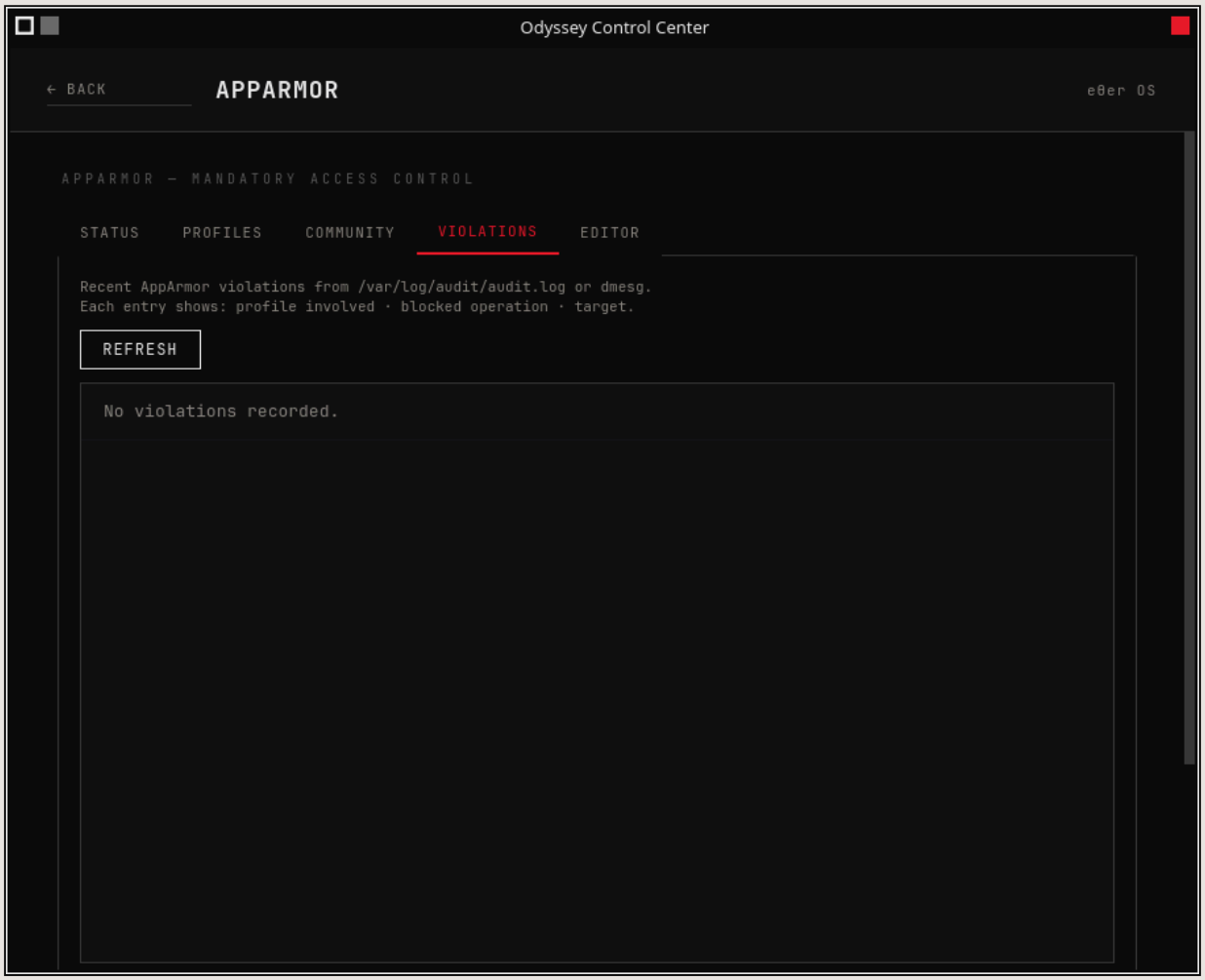

FIG · تبويب VIOLATIONS: «No violations recorded»، مع زرّ REFRESH وملاحظة بأنّ المدخلات تأتي من /var/log/audit/audit.log أو من dmesg.

يقرأ هذا التبويب سجلّ تدقيق AppArmor في النظام ويعرض ما **كان** ملفٌّ تعريفي **سيمنعه** (في وضع complain) أو ما **منعه فعلًا** (في وضع enforce) — وكلّ مدخل يسمّي الملف التعريفي والعملية الممنوعة والملف أو المورد المستهدَف.

> **القائمة الفارغة هي القراءة المتوقَّعة على نظام سليم**
>
> تعرض اللقطة بالضبط ما ستراه عادةً: لا مخالفات. وهذا ليس عطلًا ولا دليلًا على أنّ اللوحة لا تعمل — بل يعني أنّ كلّ برنامج ذي ملف تعريفي بقي داخل حدوده المسموح بها منذ آخر فحص للسجلّ. وتظهر المخالفات هنا لحظة أن يحاول برنامج مقيَّد فعل شيء لا يسمح به ملفّه؛ فاستعمل البرنامج المعنيّ، ثمّ عد واضغط REFRESH.

وحين تظهر المدخلات فعلًا، يكون النقر المزدوج على واحدها أسرع طريق إلى EDITOR وذلك الملف مفتوحٌ فيه سلفًا — انظر أدناه.

6.5 جولة — EDITOR

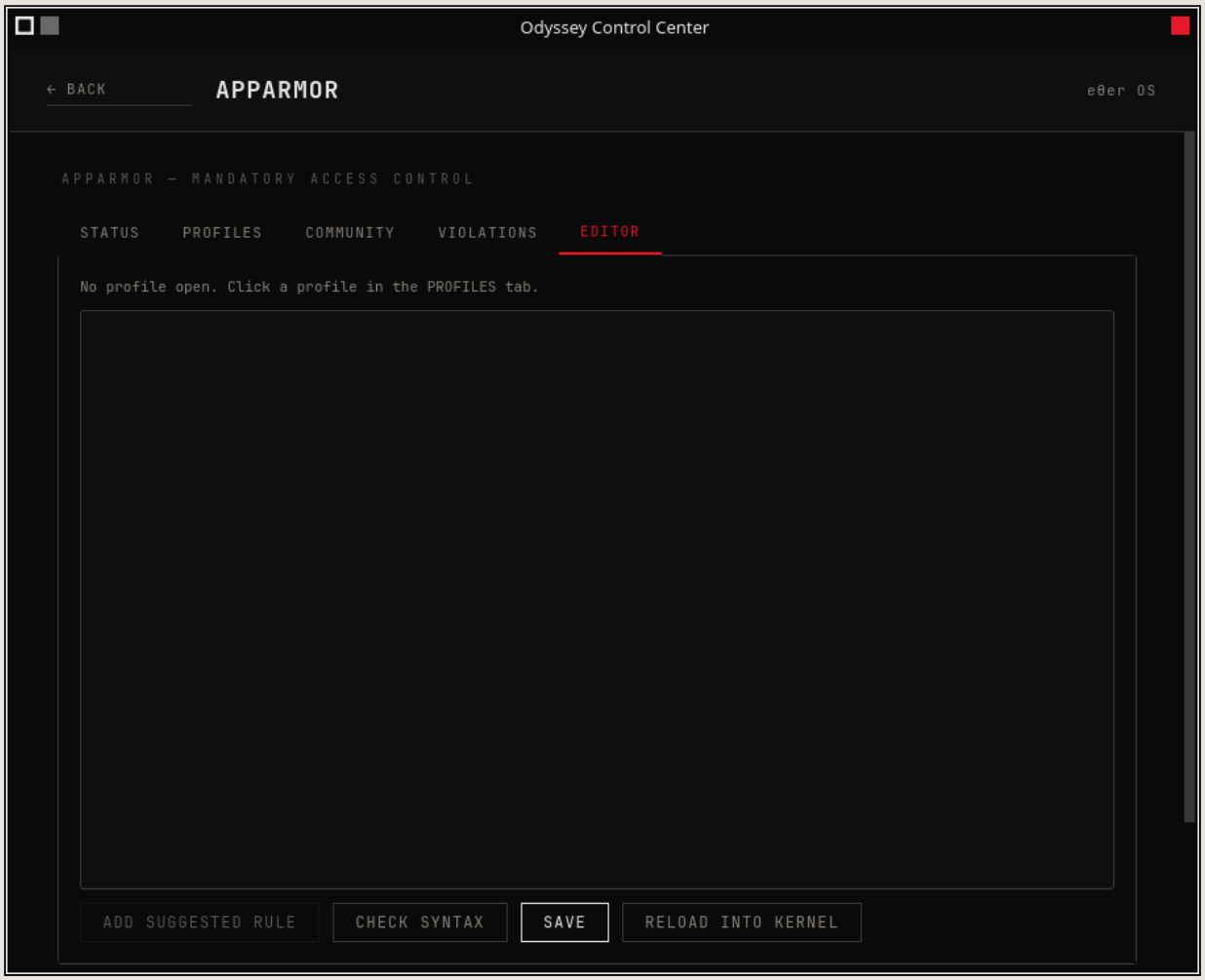

FIG · تبويب EDITOR: «No profile open. Click a profile in the PROFILES tab»، ومنطقة نصّ فارغة، وأربعة أزرار — ADD SUGGESTED RULE و CHECK SYNTAX و SAVE و RELOAD INTO KERNEL — كلّها باهتة إلّا SAVE.

يفتح المحرّر **فارغًا** عمدًا، كما ترى — فهو لا يحرّر إلّا ملفًّا تعريفيًا واحدًا في المرّة، وأنت من يختار أيّها من PROFILES (بنقر صفّ) أو من VIOLATIONS (بنقر مزدوج على مدخل). وهذا يمنع الخطأ السهل: تحرير الملف الخطأ سهوًا.

الزرّ	ماذا يفعل
ADD SUGGESTED RULE	لا يعمل إلّا إن جئت من مدخل في VIOLATIONS — ويُدرج سطر القاعدة الذي كان سيسمح بالفعل المسجَّل بالضبط
CHECK SYNTAX	يتحقّق من صحّة الملف دون تحميله — فيلتقط خطأً مطبعيًا قبل أن يهمّ
SAVE	يكتب تعديلاتك في ملف الملف التعريفي على القرص
RELOAD INTO KERNEL	يجعل التعديل المحفوظ ساريًا فورًا، بلا انتظار الإقلاع التالي أو إعادة تشغيل الخدمة

WHAT	PROFILES ← انقر اسم ملف تعريفي ← يفتحه EDITOR ← أجرِ تعديلًا ← ← SAVE ← CHECK SYNTAX .RELOAD INTO KERNEL
WHY	هذه هي دورة التقييد كاملةً في سطر واحد: راقب في وضع complain (VIOLATIONS)، وأصلح ما ينقص فعلًا (EDITOR)، وعندها فقط انقل الملف إلى enforce (PROFILES).
RESULT	يتصرّف الملف التعريفي كما حرّرته بالضبط، حيًا وبلا إعادة إقلاع.
RISK	تحرير ملف تعريفي على نحو خاطئ قد يجعل وضع enforce يمنع شيئًا مشروعًا لاحقًا — و CHECK SYNTAX يلتقط القواعد المشوَّهة قبل SAVE، لكنّه لا يلتقط أخطاء المنطق. وإن بدأ ملف حرّرته يسبّب متاعب تحت enforce، فأعده إلى complain من PROFILES وراجع VIOLATIONS من جديد.

◂ متقدّم

ما بعد هذه النقطة: مسارات ملفّات الإعداد، والأوامر الخام، وما تفعله اللوحة تحت الغطاء.

6.6 متقدّم

مكافئات سطر الأوامر

```
$ sudo aa-status
# same counts as the PROFILES tab header, from the 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
# equivalent to RELOAD INTO KERNEL after a manual edit
$ sudo apparmor_parser -r /etc/apparmor.d/PROFILE_FILE
```

قراءة سجلّ التدقيق الخام

```
$ sudo grep apparmor /var/log/audit/audit.log | tail -20
# or, if no dedicated audit daemon is running:
$ sudo dmesg | grep -i apparmor
```

والحقول التي يجدر معرفتها عند قراءة سطر يدويًا: `"apparmor="DENIED` (في enforce) أو `"apparmor="ALLOWED` (في complain، تسجيلًا فقط)، و`=operation` (ما الذي حُوِل فعله)، و`=profile` (أيّ ملف تعريفي حكم — وهو ما يطابق ما يفتحه EDITOR)، و`=name` (المسار أو المورد)، و`=comm` (العملية).

لماذا ALL→ENFORCE الجماعي محفوف بالخطر

الترقية الفردية لملف تعريفي بعد أن يبقى سجلّ VIOLATIONS الخاصّ به هادئًا مدّةً هي النمط الآمن — إذ تعرف، ملفًا ملفًا، أنّ الاستعمال الطبيعي لا يوقعه. أمّا ALL→ENFORCE فيتخطّى فترة المراقبة تلك لكلّ الملفّات دفعةً واحدة، بما فيها ملفّات ربّما ثبّتّها للتوّ من COMMUNITY ولم تشغّلها فعليًا قطّ. والملف الذي يُفرَض قبل أن تعيش معه قد يمنع ميزةً مشروعة لحظة أوّل

استعمال. ففضّل دورة الواحد تلو الآخر؛ واترك الإجراءات الجماعية لـALL→COMPLAIN (وهو آمن دائمًا لأنّه لا يفعل إلّا التخفيف) أو لـALL→DISABLE عند تتبّع عطل.

الملفّات التي تمسّها هذه اللوحة

المسار	الدور
`/etc/apparmor.d/`	ملف تعريفي واحد لكلّ برنامج مقيَّد — وهو ما يسرده PROFILES ويفتحه EDITOR
`etc/default/apparmor/`	وضع الإقلاع الافتراضي (`APPARMOR=complain`) وحالة التفعيل/التعطيل
`etc/default/grub/`	معاملا النواة `apparmor=1 security=apparmor` اللذان يفحصهما STATUS

الخصوصية

أداتان مستقلّتان، لكلّ منهما مسارها الثلاثي: التثبيت ← التفعيل ← التشغيل، ثمّ جسرٌ بينهما حين تعملان معًا.

7.1 جولة — SearXNG خطوةً خطوة

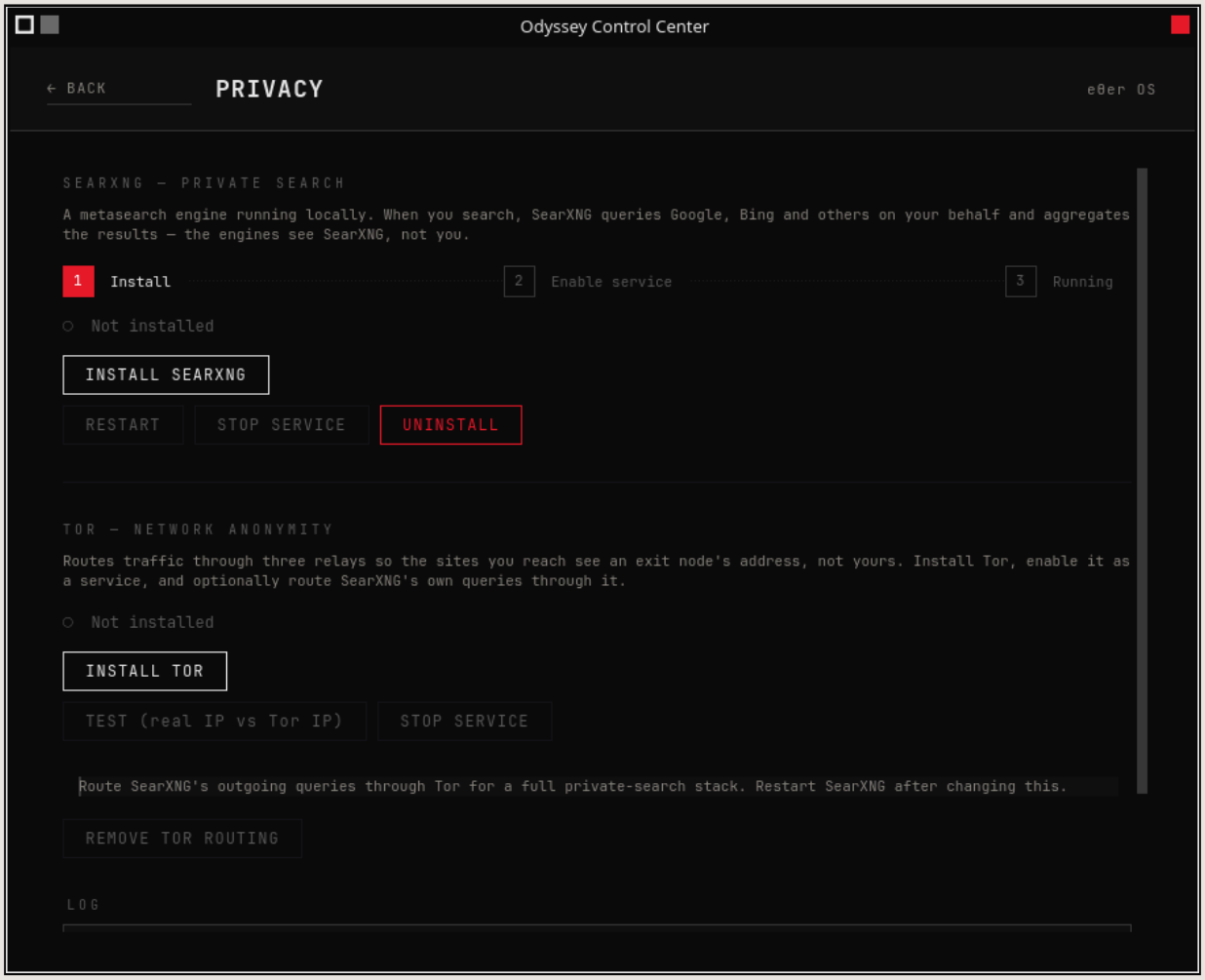

FIG · اللوحة في حالة السكون: متتبّع SearXNG على الخطوة 1 «Install»، والحالة «Not installed»، وزرّ واحد INSTALL SEARXNG؛ ومتتبّع Tor يعرض أيضًا «Not installed» مع زرّ INSTALL TOR تحته.

SearXNG محرّك بحث وسيط يعمل محليًّا على آلتك أنت — فحين تبحث، يكون SearXNG هو من يستعلم من غوغل وبينغ وسائرها نيابةً عنك، فترى تلك المحرّكات SearXNG يقدّم الطلب، لا أنت مباشرةً.

1 **INSTALL SEARXNG**

انقره. ويعرض السجلّ التثبيت الحقيقي وهو يجري — استنساخ المصدر، وإنشاء مستخدم نظام، وتهيئة بيئة بايثون افتراضية.

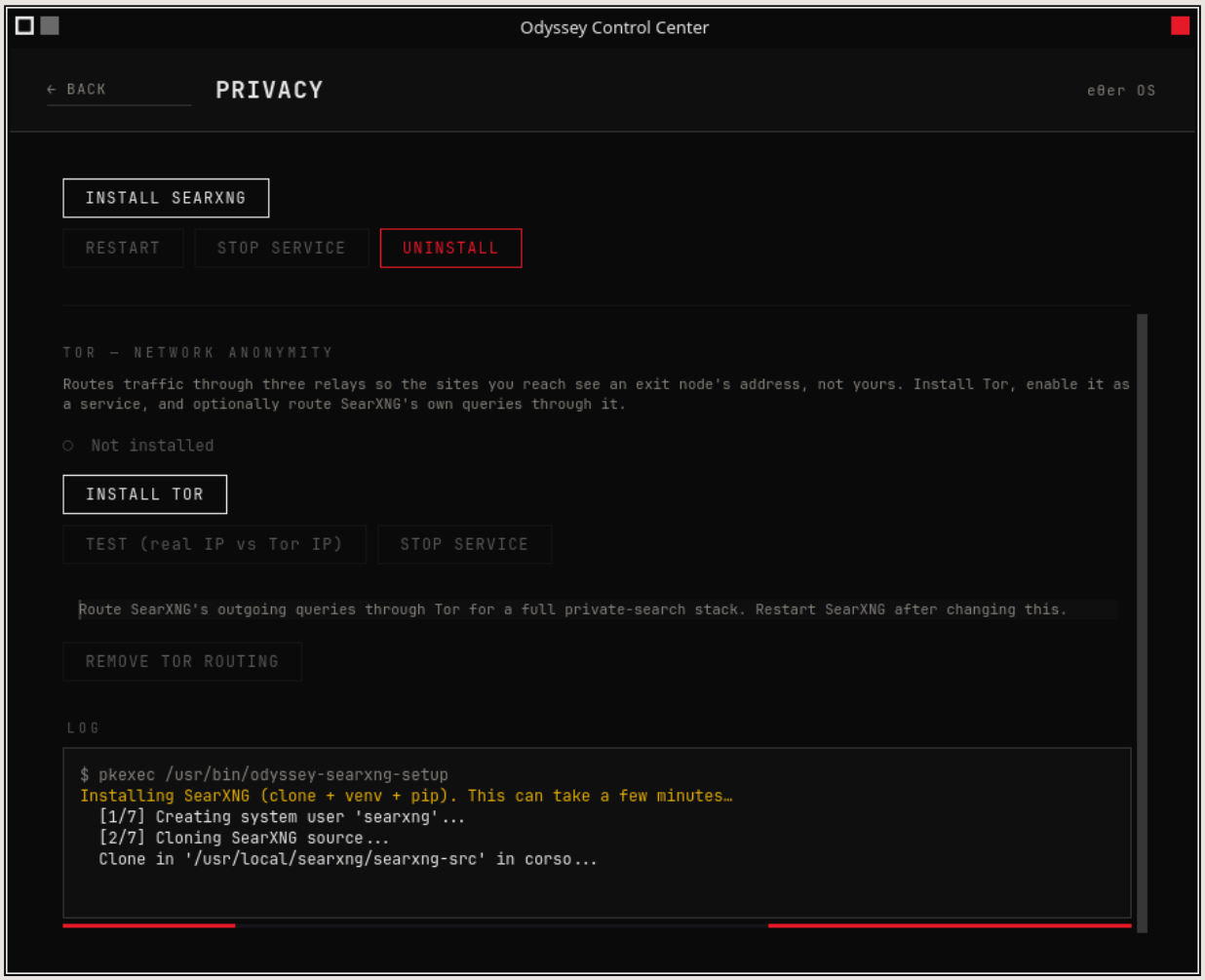

FIG · أثناء التثبيت: السجلّ يبثّ مخرجات حقيقية — «Installing SearXNG (clone + venv + pip). This can take a few minutes…» مع ظهور الخطوتين 1/7 و2/7.

2 ENABLE SERVICE

بعد التثبيت ينتقل المتتبّع إلى الخطوة 2 ويصفرّ سطر الحالة: «Installed — service not enabled». وزرّ واحد الآن: ENABLE SERVICE.

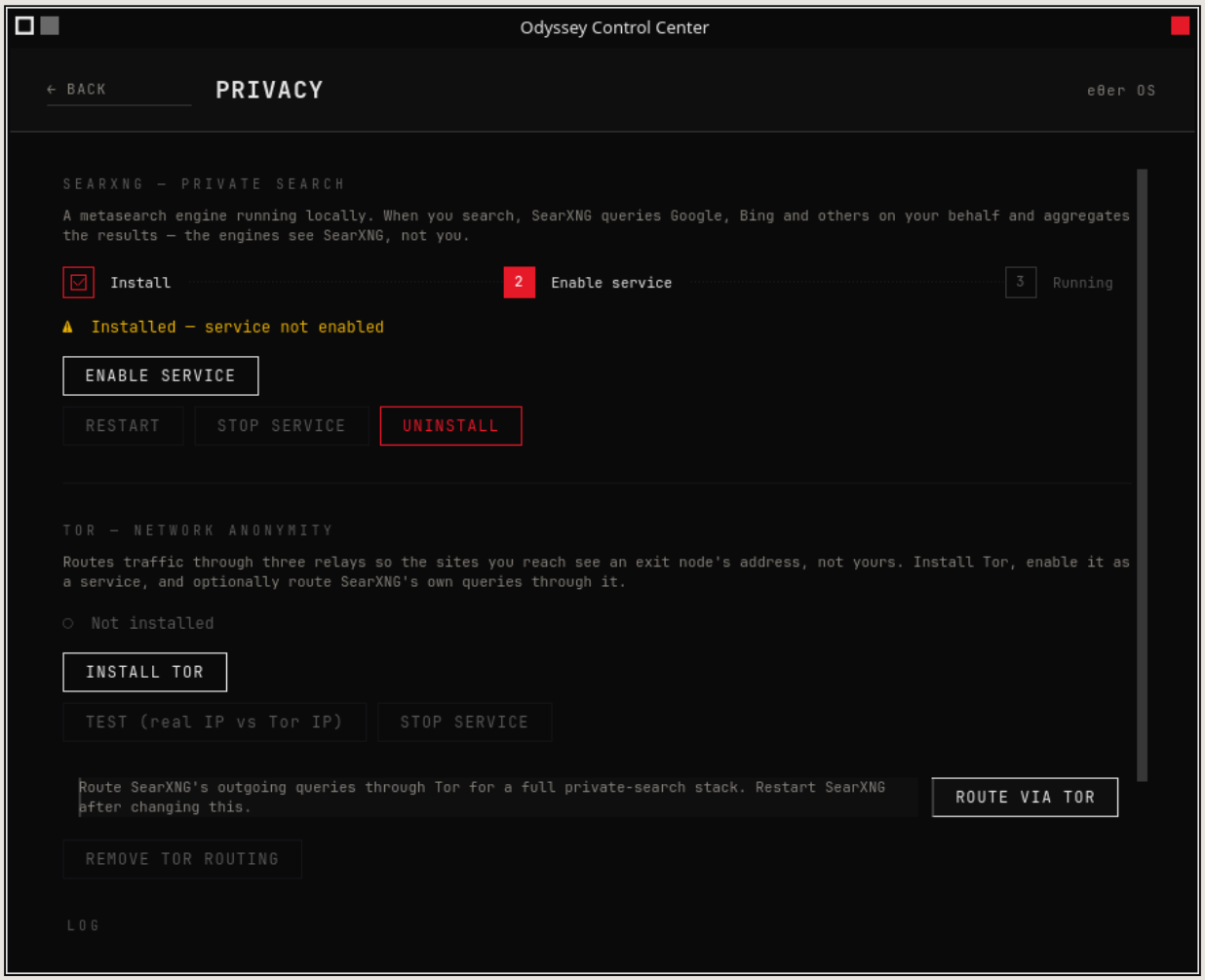

FIG · الخطوة 2 فاعلة: «Installed — service not enabled»، و ENABLE SERVICE هو الزرّ الحيّ الوحيد، بينما تبقى RESTART و STOP و SERVICE و UNINSTALL باهتة أدناه.

3 قيد التشغيل

بعد التفعيل يبلغ المتتبّع الخطوة 3 ويضيء صفّ التحكّم كاملًا.

وحين يعمل، ينقلك زرّ **OPEN IN BROWSER** إليه مباشرةً — فـSearXNG ينصت على `http://localhost:8888` افتراضيًا.

WHAT	Privacy ← INSTALL SEARXNG ← ENABLE SERVICE ← OPEN IN BROWSER.
WHY	كلّ محرّك بحث تجاري يبني ملفًّا عنك من استعلاماتك مع الوقت؛ ونسخة بحث وسيط محلية تعني أنّك وحدك من يرى الاستعلام غير المجمَّع.
RESULT	محرّك بحث خاصّ يعمل على آلتك، تبلغه من متصفّحك، ويستعلم من المحرّكات الكبرى نيابةً عنك دون أن يكشفك لها مباشرةً.
RISK	منخفض — فهو خدمة ويب محلية كأيّ خدمة أخرى. استعمل RESTART إن أساء التصرّف، و STOP SERVICE للإيقاف المؤقّت دون إزالة، و UNINSTALL لإزالته كلّيًا.

7.2 جولة Tor — خطوةً خطوة

الشكل الثلاثي نفسه، ومتتبّع مستقلّ، أسفل قسم SearXNG مباشرةً.

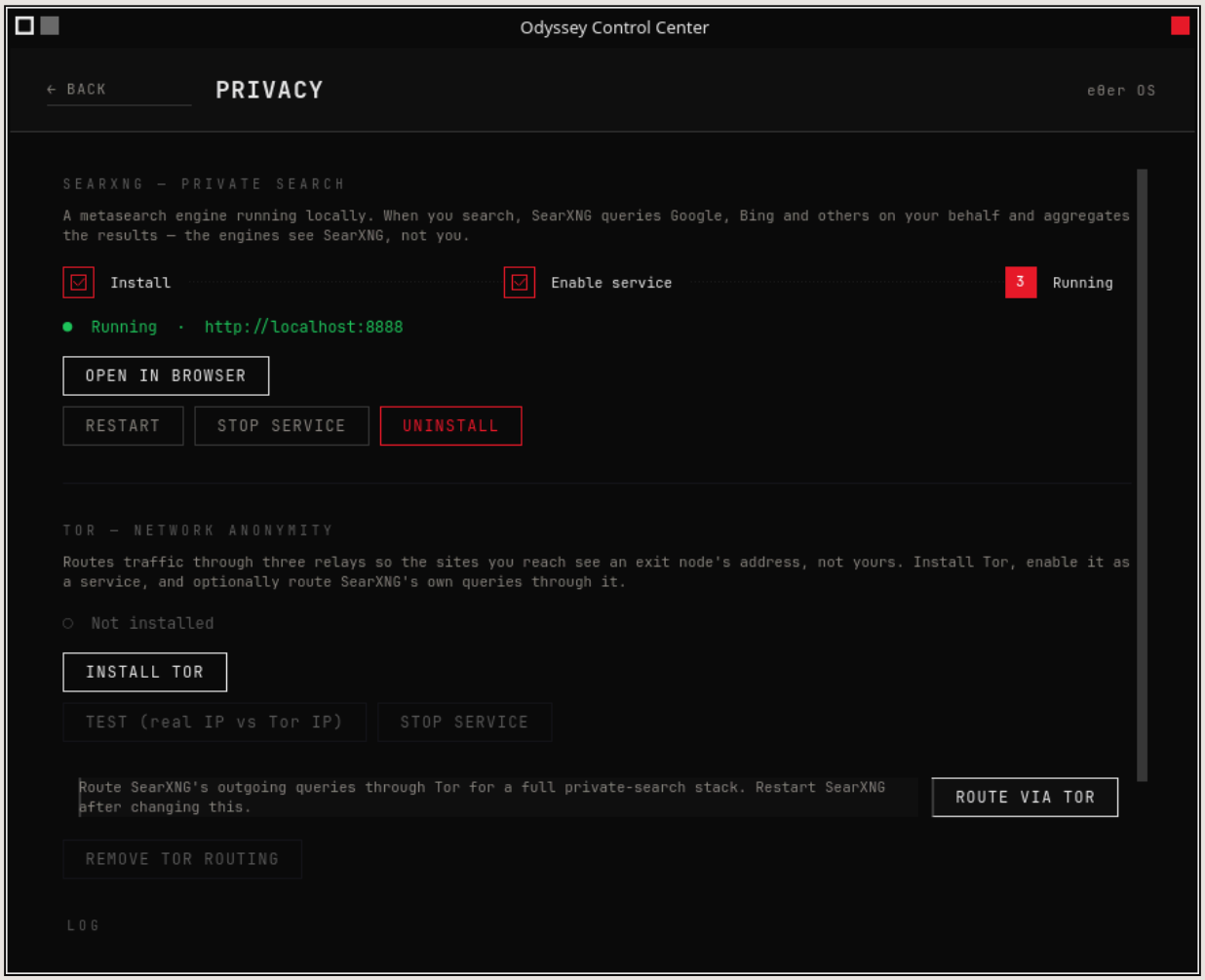

FIG · SearXNG يعمل بالكامل في الأعلى؛ وقسم Tor أدناه ما يزال عند «Not installed» و INSTALL TOR هو الزرّ الفاعل الوحيد.

يوجّه Tor حركتك عبر ثلاث مرحّلات قبل أن تبلغ وجهتها، فيرى الموقع الذي تزوره عنوان مرحّل الخروج لا عنوانك.

1 **INSTALL TOR**

يؤكّد السجلّ تثبيت الحزمة ويعرض ربط الخدمة بـrunit.

2 **ENABLE SERVICE**

تتحوّل الحالة إلى «Installed — service not enabled» — النمط ذاته كما في SearXNG.

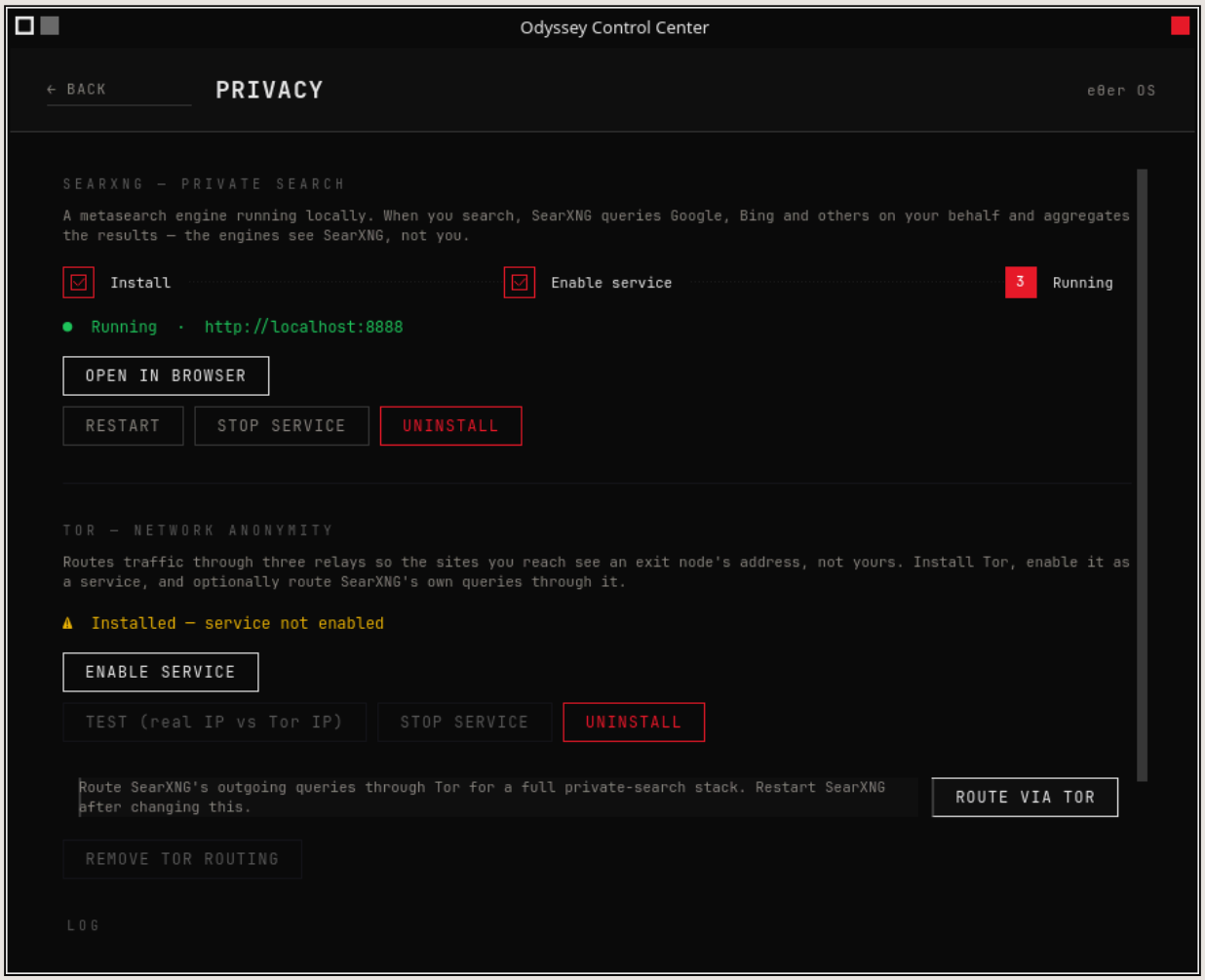

FIG · Tor عند الخطوة 2: «Installed — service not enabled»، و ENABLE SERVICE حيّ، بينما يبقى TEST و STOP SERVICE باهتين.

3 قيد التشغيل، مع التحقّق

بعد التفعيل يظهر زرّ مخصَّص **TEST (real IP vs Tor IP)** — استعمله. فهو يقارن عنوانك الحقيقي بالعنوان الذي يعرضه Tor، جنبًا إلى جنب، فتحصل على برهان فعلي على أنّه اشتغل بدل ضوء حالة عليك تصديقه.

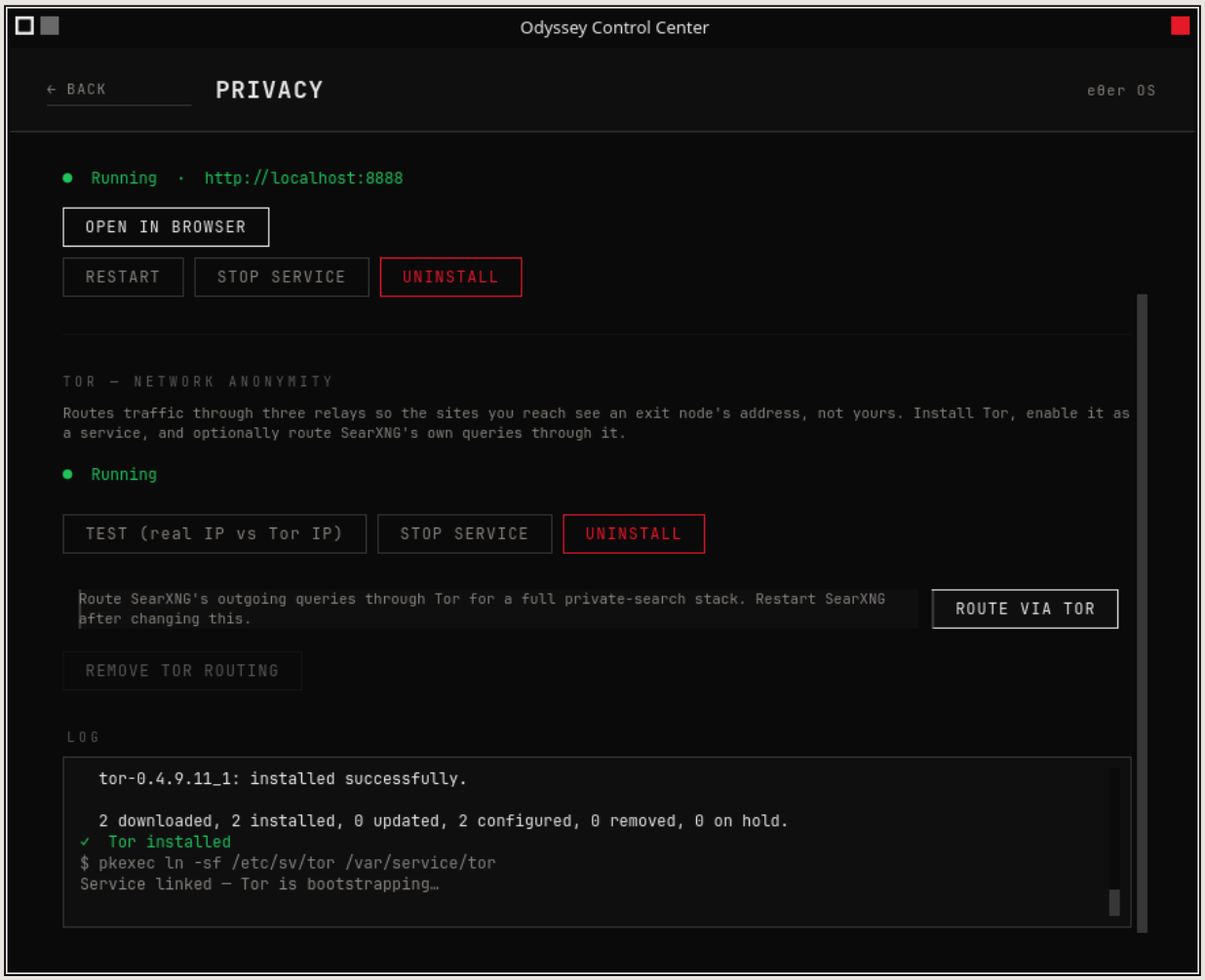

FIG · القسمان أخضران ويعملان: SearXNG على localhost:8888 مع OPEN IN BROWSER، و Tor يعمل مع TEST / STOP SERVICE / UNINSTALL جميعها فاعلة.

WHAT	ENABLE SERVICE ← INSTALL TOR ← Privacy ← ثمّ TEST (real IP vs Tor IP) للتأكيد.
WHY	توجيه التصفّح الحسّاس عبر Tor يخفي عنوانك عن المواقع التي تزورها — و TEST يعطيك برهانًا لا مجرّد ادّعاء.
RESULT	خدمة Tor عاملة تستطيع تطبيقات أخرى التوجيه عبرها؛ ويعرض TEST عنوانين مختلفين فعلًا إن كانت تعمل.
RISK	الحركة عبر Tor أبطأ بفارق ملحوظ — وتلك مقايضة إخفاء الهويّة. و STOP SERVICE يوقفه مؤقّتًا؛ و UNINSTALL يزيله.

7.3 جولة — الجمع بينهما

حين تعمل الأداتان معًا، يظهر إشعار داخلي يعرض توجيه استعلامات SearXNG **الصادرة** عبر Tor كذلك — أي لا طلباتك أنت إلى SearXNG فحسب، بل طلبات SearXNG إلى غوغل وبينغ نيابةً عنك.

WHAT	Privacy ← إشعار «ROUTE VIA TOR ← Route SearXNG's outgoing queries through Tor» ← ثمّ RESTART في قسم SearXNG ليسري المفعول.
WHY	من دون ذلك يبقى SearXNG نفسه يستعلم من المحرّكات الكبرى مباشرةً — خاصّ من ناحيتك، مكشوف من ناحيتها. وتوجيه حركة SearXNG نفسها عبر Tor يسدّ تلك الثغرة فتكتمل خصوصية منظومة البحث.
RESULT	تمرّ استعلامات SearXNG الصادرة، لا استعلاماتك إليه فقط، عبر Tor.
RISK	تصل النتائج أبطأ بوضوح، للسبب نفسه في أيّ حركة عبر Tor. وزرّ REMOVE TOR ROUTING يتراجع عن ذلك، مع إعادة تشغيل بعده أيضًا.

◂ متقدّم

ما بعد هذه النقطة: مسارات ملفّات الإعداد، والأوامر الخام، وما تفعله اللوحة تحت الغطاء.

7.4 متقدّم

ما الذي ينفّذه INSTALL SEARXNG فعليًا

```
$ pkexec /usr/bin/odyssey-searxng-setup
# creates a dedicated system user 'searxng',
# clones the source to /usr/local/searxng/searxng-src,
# builds a Python venv and installs dependencies with pip
```

الخدمات والمنافذ يدويًا

```
# SearXNG's runit service
$ sudo sv status searxng
$ sudo sv restart searxng
# Tor's runit service — the panel's own enable step is literally this
$ sudo ln -sf /etc/sv/tor /var/service/tor
$ sudo sv status tor
# verify Tor is actually routing
$ curl -socks5 localhost:9050 https://check.torproject.org/api/ip
```

توجيه SEARXNG عبر TOR يدويًا

ينصت وسيط SOCKS الخاصّ بـTor على `127.0.0.1:9050` افتراضيًا. وتوجيه SearXNG عبره يعني توجيه عميل HTTP الصادر لديه إلى ذلك الوسيط — والإعداد يعيش في ملف `settings.yml` الخاصّ بـSearXNG، ضمن إعداد الطلبات الصادرة، لا في متغيّر وسيط على مستوى النظام. وزرّ ROUTE VIA TOR في اللوحة يحرّر ذلك الملف مباشرةً ثمّ يعيد تشغيل الخدمة؛ أمّا فعل ذلك يدويًا فيعني إيجاد كتلة `outgoing:` نفسها وتحريرها في `/usr/local/searxng/searxng-src/searx/settings.yml`.

لماذا يحتاج SEARXNG إلى إعادة تشغيل بعد تغيير التوجيه

يقرأ SearXNG إعداد الوسيط مرّةً واحدة عند الإقلاع — فهو ليس إعدادًا قابلًا لإعادة التحميل حيًّا. وهذا صحيح سواء غيّرته من اللوحة أم يدويًا، ولهذا ينتهي المساران بخطوة RESTART نفسها.

الملفّات التي تمسّها هذه اللوحة

المسار	الدور
`/usr/local/searxng/searxng-src/`	تثبيت SearXNG نفسه — المصدر والبيئة الافتراضية والإعداد
`/etc/sv/tor/`، `etc/sv/searxng`	تعريفات خدمات runit
`/var/service/`	حيث يربط تفعيل الخدمة إيّاها بشجرة الإشراف العاملة

الحاويات

إدارة الحاويات من طرف إلى طرف: تهيئة المحرّك، وقائمة بالحاويات العاملة، وفهرس منتقًى للبدء منه، وأوامر خام لكلّ ما ليس فيه، واستهلاك القرص — كلّ ذلك في لوحة واحدة قابلة للتمرير.

8.1 جولة — المحرّك

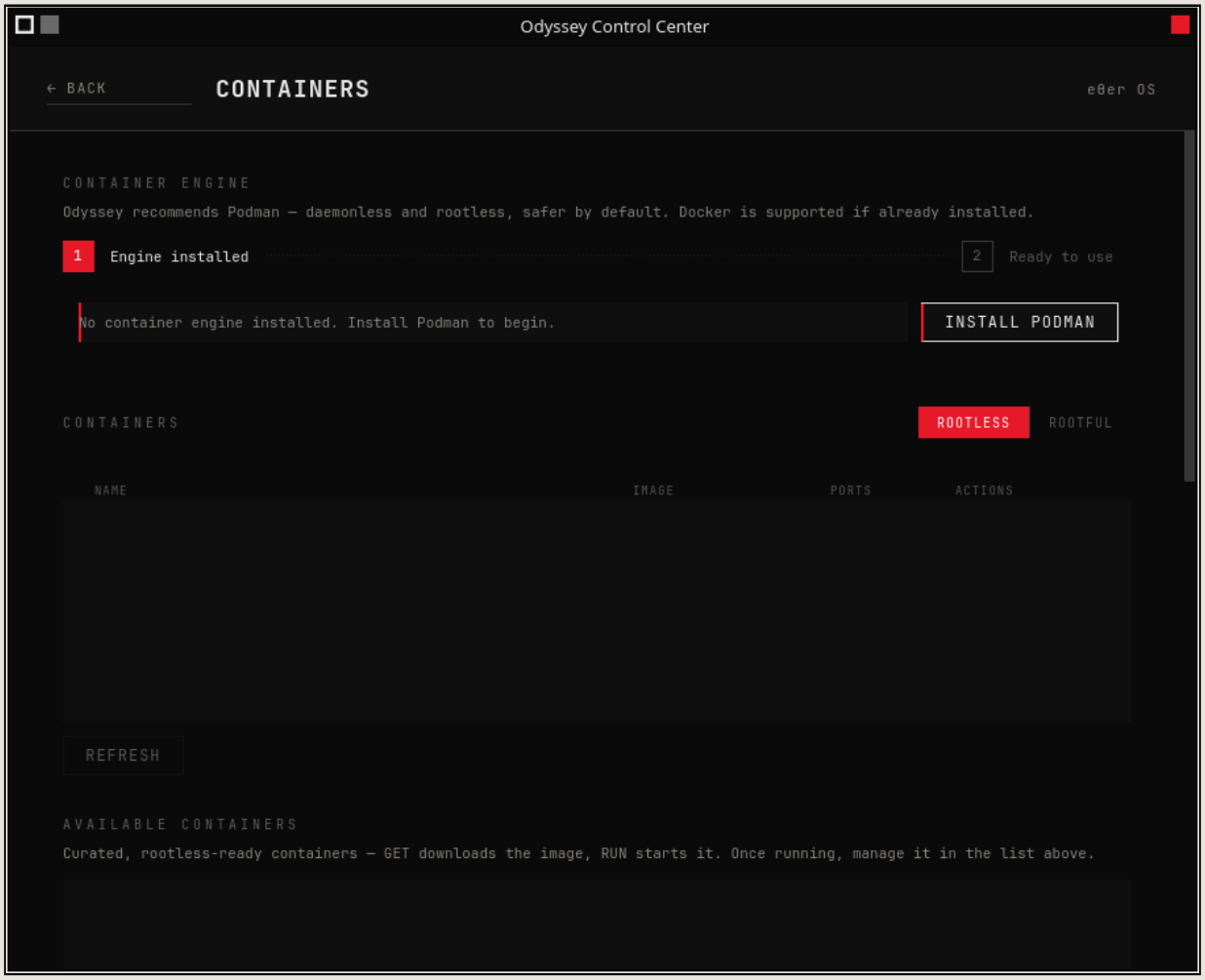

FIG · الخطوة 1 من 2: «No container engine installed. Install Podman to begin»، مع INSTALL PODMAN بوصفه الإجراء الحيّ الوحيد. وقسما Containers و Available containers أدناه ظاهران لكن في حالة فارغة.

يوصي Odyssey بـ**Podman** — فهو بلا عفريت وبلا جذر، أي أنّ الحاويات تعمل دون خدمة خلفية دائمة، وبلا صلاحيات جذر افتراضيًا. و**Docker** مدعوم أيضًا إن كان هو ما تستعمله سلفًا.

1 INSTALL PODMAN

يعرض السجلّ تثبيت الحزمة الحقيقي.

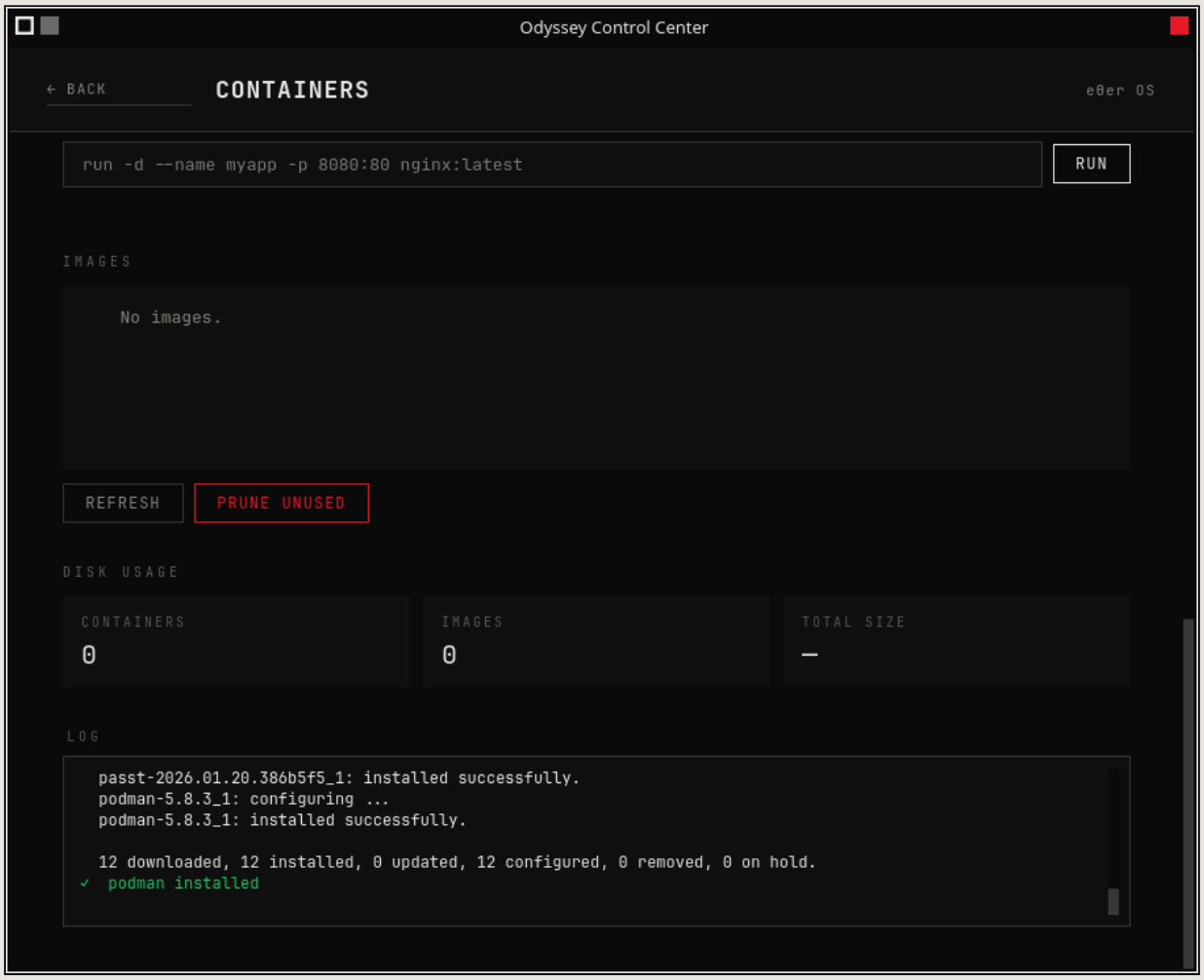

FIG · بعد التثبيت: «Engine installed» مؤشَّرة بالأحمر، والخطوة 2 «Ready to use»، وسطر المعلومات يقرأ «podman version 5.8.3 · Podman installed · Docker not installed»، مع زرّ INSTALL DOCKER TOO لمن يريد Docker إلى جانبه.

2 جاهز للاستعمال

المحرّك يعمل. ويبقى **INSTALL DOCKER TOO** متاحًا إن أردت المحرّكين جنبًا إلى جنب — فهما يتعايشان بلا تعارض.

8.2 جولة — بلا جذر مقابل بجذر، والحالة الفارغة

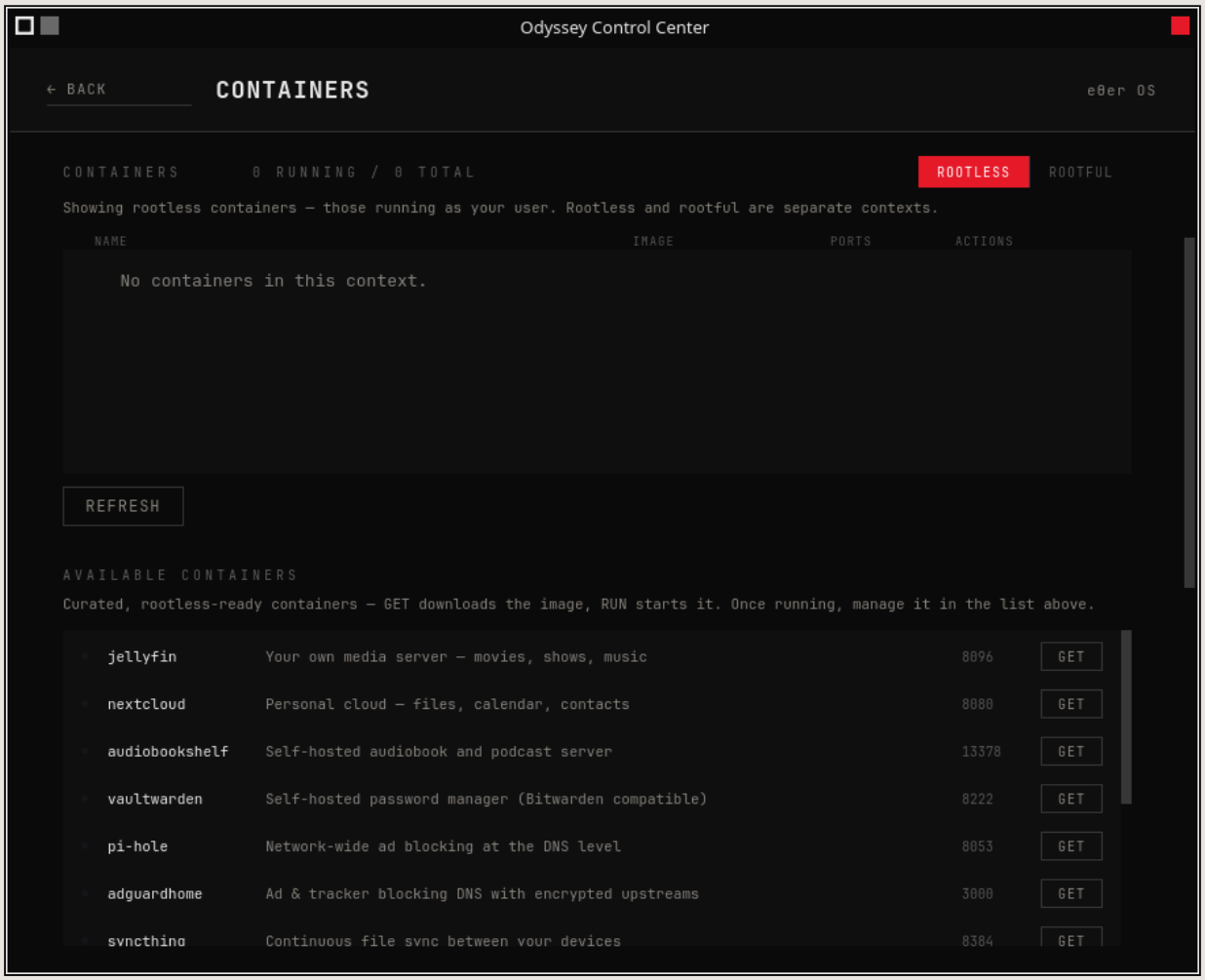

FIG · ROOTLESS محدّد (مضاء بالأحمر) بجوار ROOTFUL، و«0 TOTAL 0 RUNNING /»، و«No containers in this context.» تحت عناوين الأعمدة NAME / IMAGE / PORTS / ACTIONS.

مفتاح **ROOTLESS / ROOTFUL** في الأعلى يبدّل السياق الذي تعرضه وتديره — فالاثنان فضاءا أسماء منفصلان تمامًا للحاويات، لا مرشِّح على قائمة واحدة مدمجة. وOdyssey يعرض ROOTLESS افتراضيًا، وهذا افتراض مقصود:

> **أيّهما ينبغي أن أستعمل**
>
> بلا جذر هو الخيار الأأمن وهو ما توجّهك إليه اللوحة: فالحاوية التي تفلت من تقييدها في وضع بلا جذر تظلّ عاجزة عن فعل أيّ شيء لا يستطيع حسابك أنت فعله أصلًا. أمّا الوضع بجذر فموجود للحالات التي تحتاجه بعينها — منافذ شبكة ذات أرقام منخفضة، وبعض الوصول إلى العتاد — وينبغي أن يكون الاستثناء لا العادة.

8.3 جولة — الفهرس المنتقى، من البداية إلى النهاية

انزل إلى **Available containers** — قائمة جاهزة كي لا تكون اللوحة صندوقًا فارغًا ينتظر منك أن تحفظ أسماء الصور عن ظهر قلب. ويعرض كلّ سطر الحاوية، ووصفًا في سطر، والمنفذ الذي تستعمله.

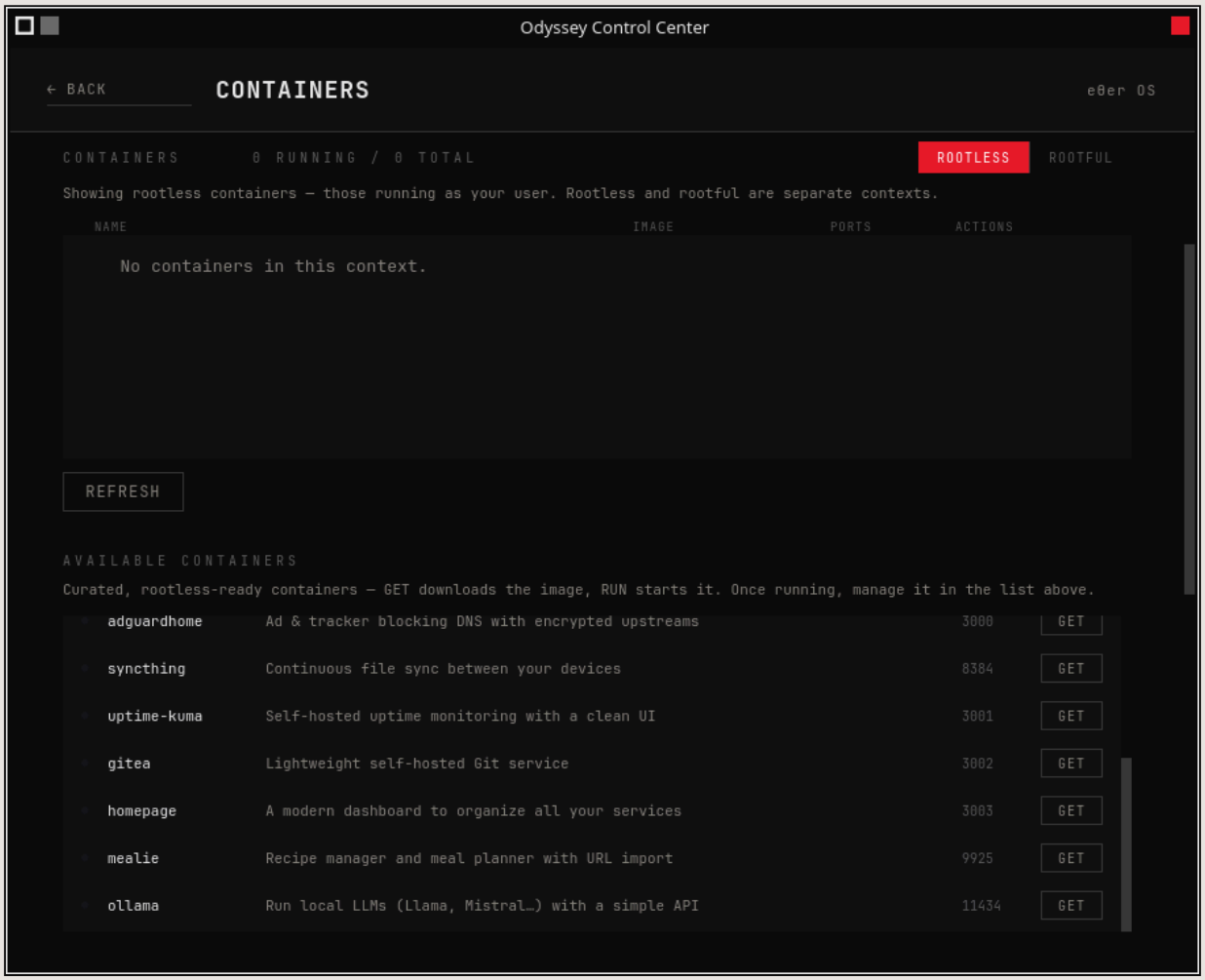

FIG · قائمة الفهرس: adguardhome (المنفذ 3000)، و syncthing (8384)، و uptime-kuma (3001)، و gitea (3002)، و homepage (3003) — مع زرّ GET في كلّ سطر.

ورقم المنفذ ذاك هو الذي ستبلغ الخدمة عليه حين تعمل — فمنفذ `3003` الخاصّ بـ`homepage` مثلًا يعني `http://localhost:3003` بعد تشغيلها.

1 **GET**

انقر GET على `homepage`. وهذا ينزّل صورة الحاوية فقط — ولا شيء يعمل بعد.

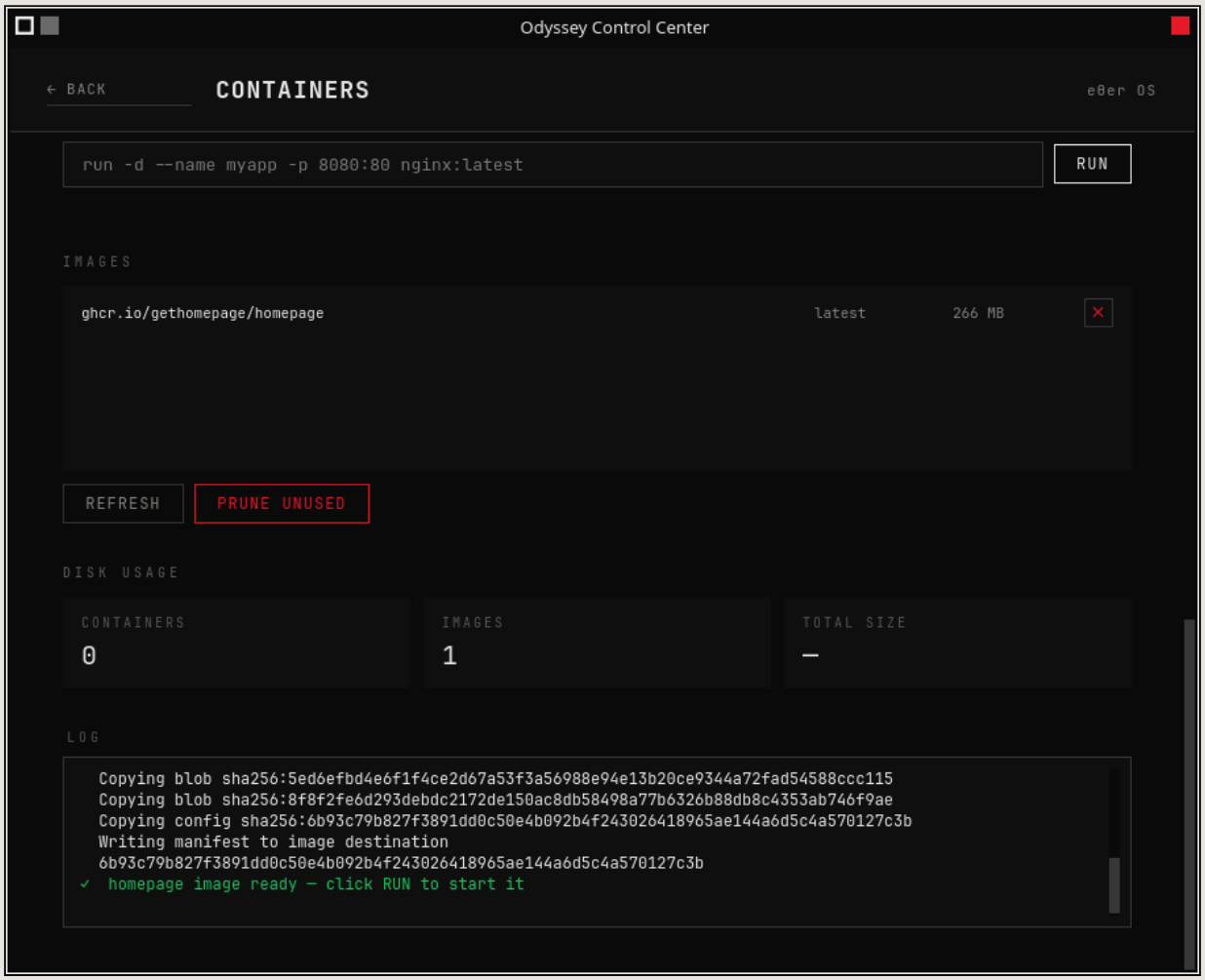

FIG · السجلّ يبثّ سحب صورة حقيقيًا: «Copying blob sha256:5ed6…»، و«Writing manifest to image destination»، وينتهي بالأخضر «homepage image ready — click RUN to start it».

2 RUN

يعرض السطر نفسه الآن RUN بدل GET — انقره لتشغيل الحاوية فعلًا.

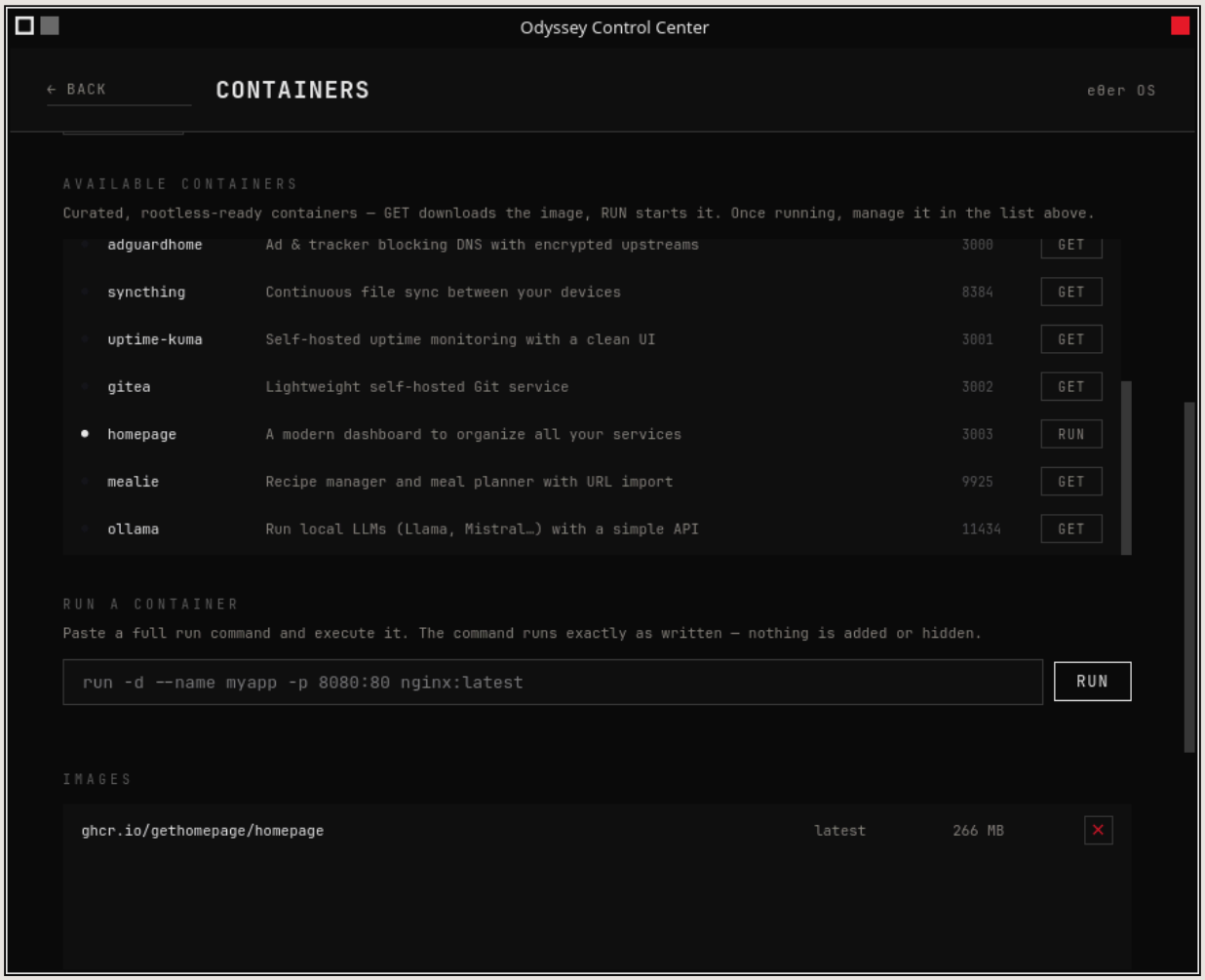

FIG · homepage صارت مدرّجة في قسم CONTAINERS أعلاه، بنقطة خضراء، ومسار الصورة ghcr.io/gethomepage/homepage، والمنافذ «0.0.0.0:3003→…»، ومعها صفّ أيقونات الإجراءات؛ وتبقى ظاهرة أدناه في Available Containers بزرّ RUN جديد لتشغيل نسخة أخرى.

وحين تعمل، تظهر homepage في موضعين معًا: مدخلًا حيًّا في قائمة **Containers** في الأعلى (بمنفذها وأيقونات إجراءاتها)، وفي الفهرس أدناه أيضًا تحسّبًا لرغبتك في تشغيل نسخة ثانية لاحقًا.

WHAT	GET ← Available containers ← Containers على شيء في الفهرس ← ثمّ RUN بعد التنزيل.
WHY	الفصل بين الخطوتين (التنزيل ثمّ التشغيل) يعني أنّك تستطيع جلب صورة مقدَّمًا دون الالتزام بتشغيلها فورًا — وهو مفيد على اتصال بطيء أو حين تتصفّح المتاح لا أكثر.
RESULT	حاوية عاملة، ظاهرة في القائمة الرئيسية مع ربط منفذها، وقابلة للإدارة من هناك فصاعدًا.
RISK	منخفض — فالخدمات المستضافة ذاتيًا في الفهرس المنتقى مختارة لتكون صديقةً للوضع بلا جذر ومنخفضة الخطر افتراضيًا. والانكشاف كانكشاف تشغيل أيّ خدمة ويب: فالمنفذ الذي تنصت عليه قابل للبلوغ بحسب إعدادات جدار الحماية عندك.

8.4 جولة — قائمة الحاويات وإجراءاتها

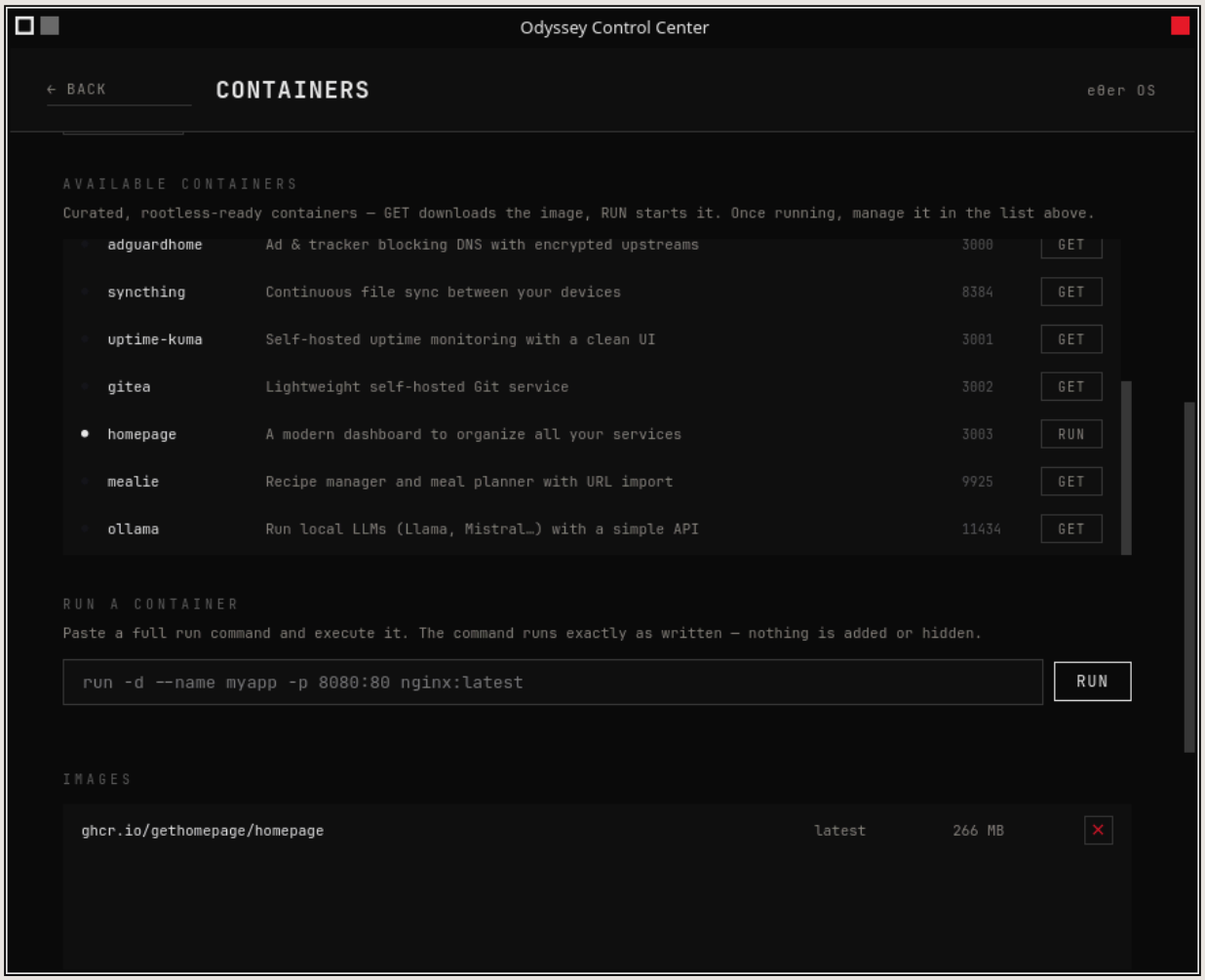

FIG · سطر homepage في قائمة Containers، وأيقونات الإجراءات الخمس ظاهرة على اليمين — وهي التي يشرحها ما يلي.

ينتهي سطر كلّ حاوية عاملة بمجموعة أيقونات — مرّر المؤشّر فوق أيّ منها لترى تسميتها، وإليك ما يفعله كلّ منها:

الأيقونة	الإجراء
فتح (سهم)	يفتح الخدمة في متصفّحك مباشرةً — وهو أسرع طريق بعد التشغيل
إيقاف / تشغيل (مربّع / مثلّث)	يوقف الحاوية دون إزالتها، أو يشغّلها من جديد
إعادة تشغيل (سهم دائري)	يوقفها ويشغّلها في إجراء واحد — وهو علاج «تعمل لكنّها لا تستجيب»
السجلّات (أيقونة الأسطر)	يفتح مخرجات سجلّ الحاوية نفسها — وهو أوّل موضع تنظر فيه إن لم يعمل شيء
إزالة (×)	يزيل الحاوية كلّيًا — وتبقى الصورة نفسها منزَّلة، مدرَجةً على حدة تحت Images

8.5 جولة — تشغيل حاوية وفتحها

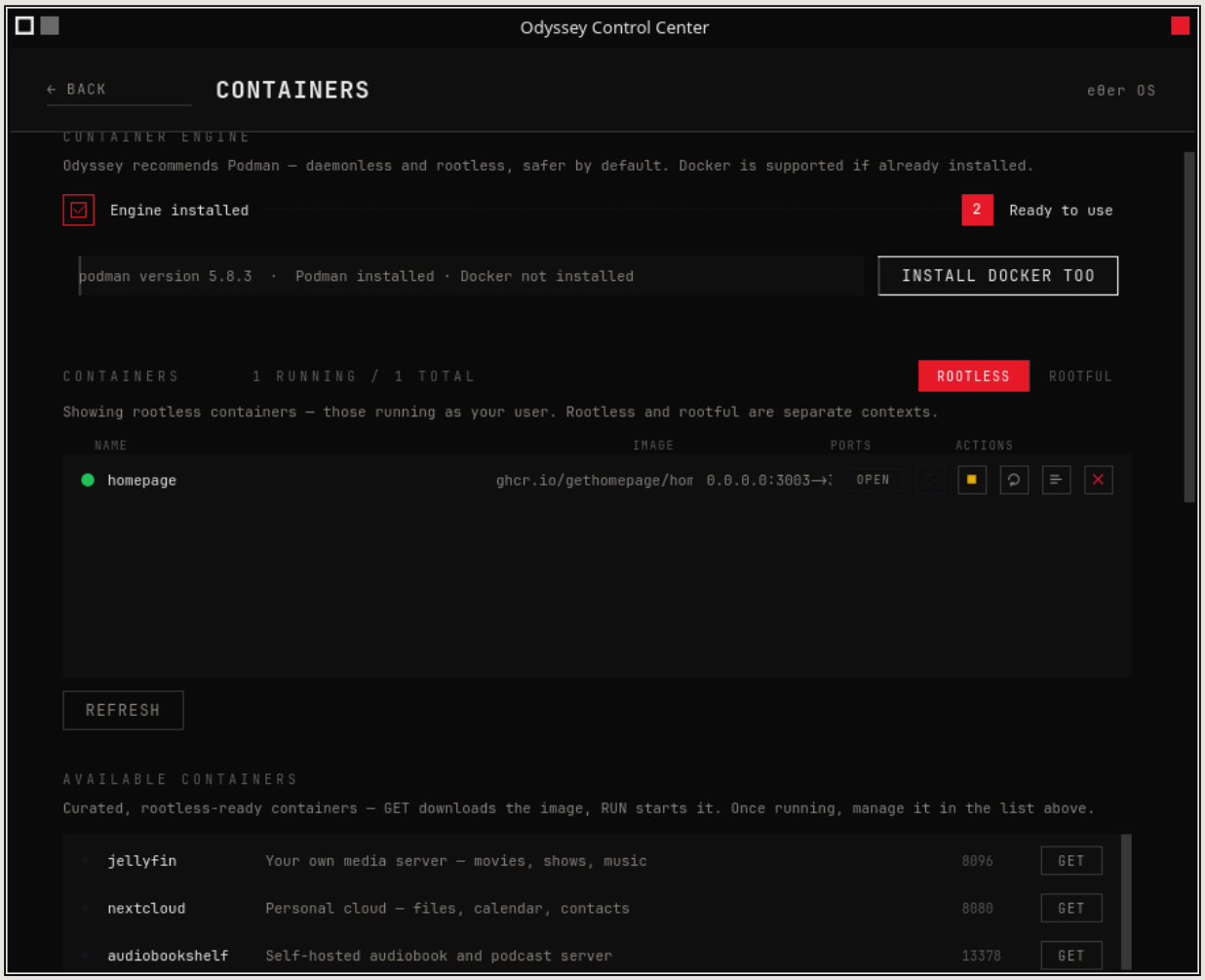

FIG · حقل «Run a container» مع مثال أمر نائب، وزرّ RUN، وقسم Images أدناه يسرد 266 · latest · ghcr.io/gethomepage/homepage ميغابايت مع زرّ إزالة.

ولكلّ ما ليس في الفهرس، يأخذ حقل **Run a container** أمر `podman run` (أو `docker run`) كاملًا وينفّذه كما كُتب بالضبط — بلا إضافة ولا إخفاء.

WHAT	الصق أمر تشغيل كاملًا (مثل `run -d --name myapp -p 8080:80`) ← «Run a container» ← «Containers» ← (`nginx:latest`) ← RUN.
WHY	الفهرس المنتقى يغطّي الحالات الشائعة؛ وهذا هو باب الخروج لصورة بعينها، وبرايات بعينها، ومنافذ بعينها لا يوفّرها الفهرس.
RESULT	تبدأ الحاوية تمامًا كما يحدّد الأمر، وتنضمّ إلى قائمة Containers الرئيسية بعد تشغيلها.
RISK	ما يفعله الأمر نفسه — فهذا تمرير أمين لا صندوق رمل. عامله بالحذر ذاته الذي تعامل به كتابة الأمر نفسه في طرفية.

وأسفل منه يسرد قسم **Images** كلّ صورة منزَّلة بصرف النظر عن وجود حاوية عاملة منها، مع زرّي **REFRESH** و**PRUNE UNUSED** لاسترجاع مساحة من الصور التي لم يعد يشير إليها شيء.

فتح حاوية عاملة

نقر OPEN في سطر حاوية عاملة — وهو إجراء الفتح نفسه من صفّ الأيقونات أعلاه — يطلق متصفّحك الافتراضي مباشرةً إلى عنوانها ومنفذها، مثل `http://localhost:3003` لـ`homepage`. فلا حاجة إلى تذكّر رقم المنفذ بنفسك بعد التشغيل.

8.6 استهلاك القرص

في أسفل اللوحة تمامًا ثلاثة مجاميع جارية — الحاويات، والصور، والحجم الكلّي — لتعرف دائمًا ما تكلّفك الحاويات على القرص دون فتح طرفية.

◂ متقدّم

ما بعد هذه النقطة: مسارات ملفّات الإعداد، والأوامر الخام، وما تفعله اللوحة تحت الغطاء.

8.7 متقدّم

مكافئات سطر الأوامر

```
"# what the panel calls "GET
$ podman pull ghcr.io/gethomepage/homepage
# what the panel calls "RUN" on a catalog entry
$ podman run -d -name homepage -p 3003:3000 ghcr.io/gethomepage/homepage
# list running containers — rootless context
$ podman ps
# same, rootful
$ sudo podman ps
# logs, stop, restart, remove — the icon row, by hand
$ podman logs -f homepage
$ podman stop homepage
$ podman restart homepage
$ podman rm homepage
# images and disk usage
$ podman images
$ podman system df
$ podman image prune
```

بلا جذر مقابل بجذر، من الداخل

Podman بلا جذر يشغّل الحاويات بحساب مستخدمك، داخل فضاء أسماء للمستخدم — فيُخطَّط «جذر» الحاوية إلى نطاق UID غير ممتاز على المضيف، حتى إنّ عملية حاوية مخترَقة بالكامل تظلّ مقيَّدة بأذونات حسابك أنت. أمّا Podman بجذر (`sudo podman`).فيعمل بالطريقة التقليدية، بحيث يُخطَّط جذر الحاوية إلى الجذر الحقيقي على المضيف. ومفتاح اللوحة إنّما يبدّل

في الحقيقة بين تشغيل `podman` المجرّد و`sudo podman` — بأدلّة حالة منفصلة فعلًا، وقوائم حاويات منفصلة، ولا شيء مشترك بين السياقين افتراضيًا.

```
# where each context keeps its state
'$ podman info -format '{{.Store.GraphRoot}}
# rootless:  /.local/share/containers/storage
# rootful: /var/lib/containers/storage
```

لماذا تحتاج بعض مدخلات الفهرس إلى فتح منافذ بعينها

ربط منفذ الحاوية (`3003:3000` في مثال homepage) لا يجعل الخدمة قابلة للبلوغ إلّا **من هذه الآلة** افتراضيًا — إذ يربطها Podman بـ`localhost` ما لم يُؤمَر بغير ذلك. أمّا بلوغها من جهاز آخر على شبكتك فيتطلّب إضافةً إلى ذلك فتح ذلك المنفذ في لوحة Firewall (الفصل 4) — فالنظامان طبقتان مستقلّتان، والحاوية العاملة بمنفذ مربوط ليست قابلةً للبلوغ من الإنترنت أو من الشبكة المحلية تلقائيًا لمجرّد أنّ Podman ينصت.

وضع الألعاب

مفتاح رئيسي واحد في الأعلى، وسطر حالة حيّ، وأربعة تبويبات: **PACKAGES** و**GAMEMODE** و**MANGOHUD** و**PROTON**.

9.1 جولة — المفتاح الرئيسي

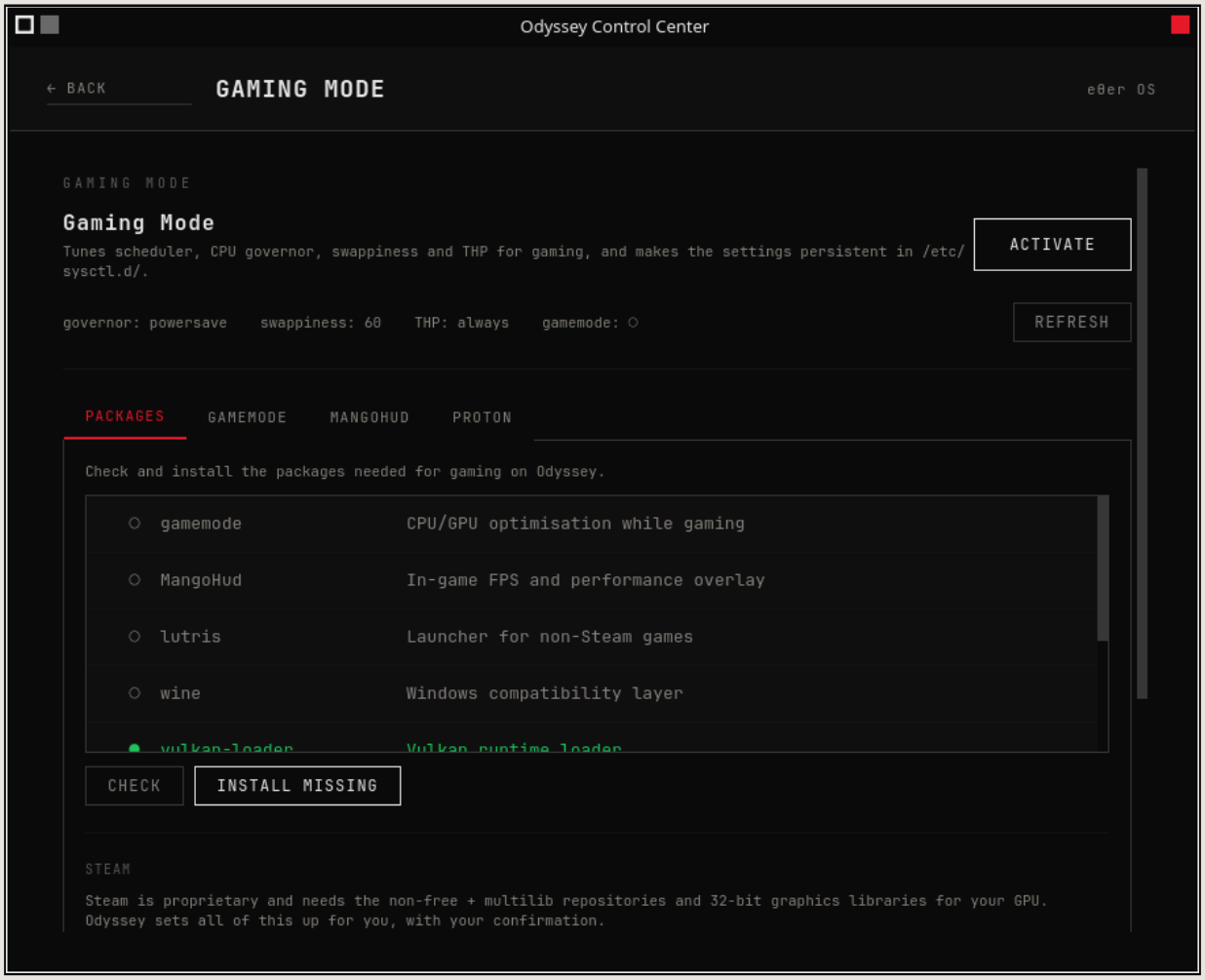

FIG · قسم اللوحة العلوي: «Gaming Mode — Tunes scheduler, CPU governor, swappiness and THP for gaming, and makes the settings persistent in /etc/sysctl.d/»، وزرّ ACTIVATE، وزرّ REFRESH، وسطر الحالة الحيّ: المنظّم powersave، وswappiness 60، وTHP always، وgamemode مطفأ.

يقرأ سطر الحالة القيم الحيّة نفسها التي تعرضها لوحتا Kernel و Memory — المنظّم، وswappiness، وحالة THP، وهل تعمل خدمة gamemode — فتتأكّد بنظرة واحدة من أنّ التفعيل قد سرى فعلًا، بلا زيارة أيّ من اللوحتين على حدة.

WHAT	Gaming Mode ← ACTIVATE.
WHY	زيارة Kernel و Memory يدويًا لضبط منظّم وswappiness وTHP بقيم مناسبة للألعاب هي بالضبط نوع العمل المتعدّد الخطوات الذي ينبغي لوضع مخصَّص أن يريحك منه.
RESULT	ينتقل المجدوِل والمنظّم وswappiness وTHP كلّها إلى قيم مناسبة للألعاب، تُطبَّق فورًا وتُجعَل دائمة — ويتحدّث سطر الحالة ليعكس التغيير.
RISK	منخفض وقابل للعكس — فهي القيم ذاتها التي كنت ستضبطها يدويًا في Kernel/Memory، مجموعةً في نقرة واحدة.

9.2 جولة — الحزم وSteam

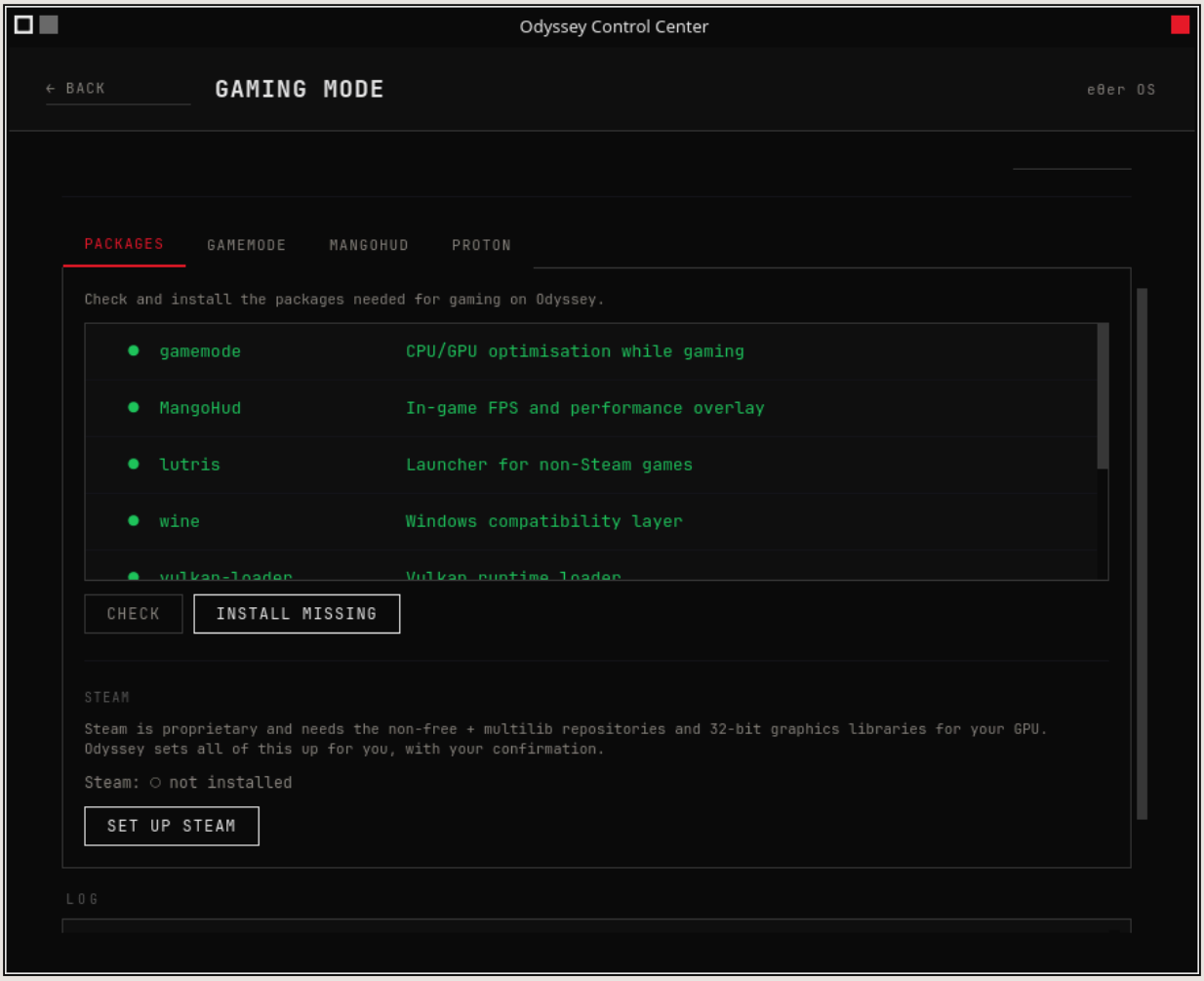

FIG · تبويب PACKAGES: gamemode و MangoHud و lutris و wine و vulkan-loader جميعها بنقاط خضراء ممتلئة (مثبّتة)؛ وزرّا CHECK و INSTALL MISSING أعلاه؛ وقسم Steam أدناه يعرض «Steam: ● installed» وزرّ REINSTALL STEAM.

قائمة تحقّق بالحزم التي يحتاجها اللعب عادةً — زرّ CHECK يرى ما ينقص، وزرّ INSTALL MISSING يسدّ الفجوات في إجراء واحد. وحين تظهر لكلّ عنصر نقطة خضراء ممتلئة، كما في اللقطة، تكون التهيئة كاملة.

ولـSteam مسارٌ خاصّ به أسفل قائمة التحقّق، لأنّه يحتاج إلى أكثر من تثبيت حزمة عادية.

1 **SET UP STEAM**

انقره — وهذا لا يثبّت شيئًا بعد، بل يفتح أوّلًا نافذة تأكيد شارحة.

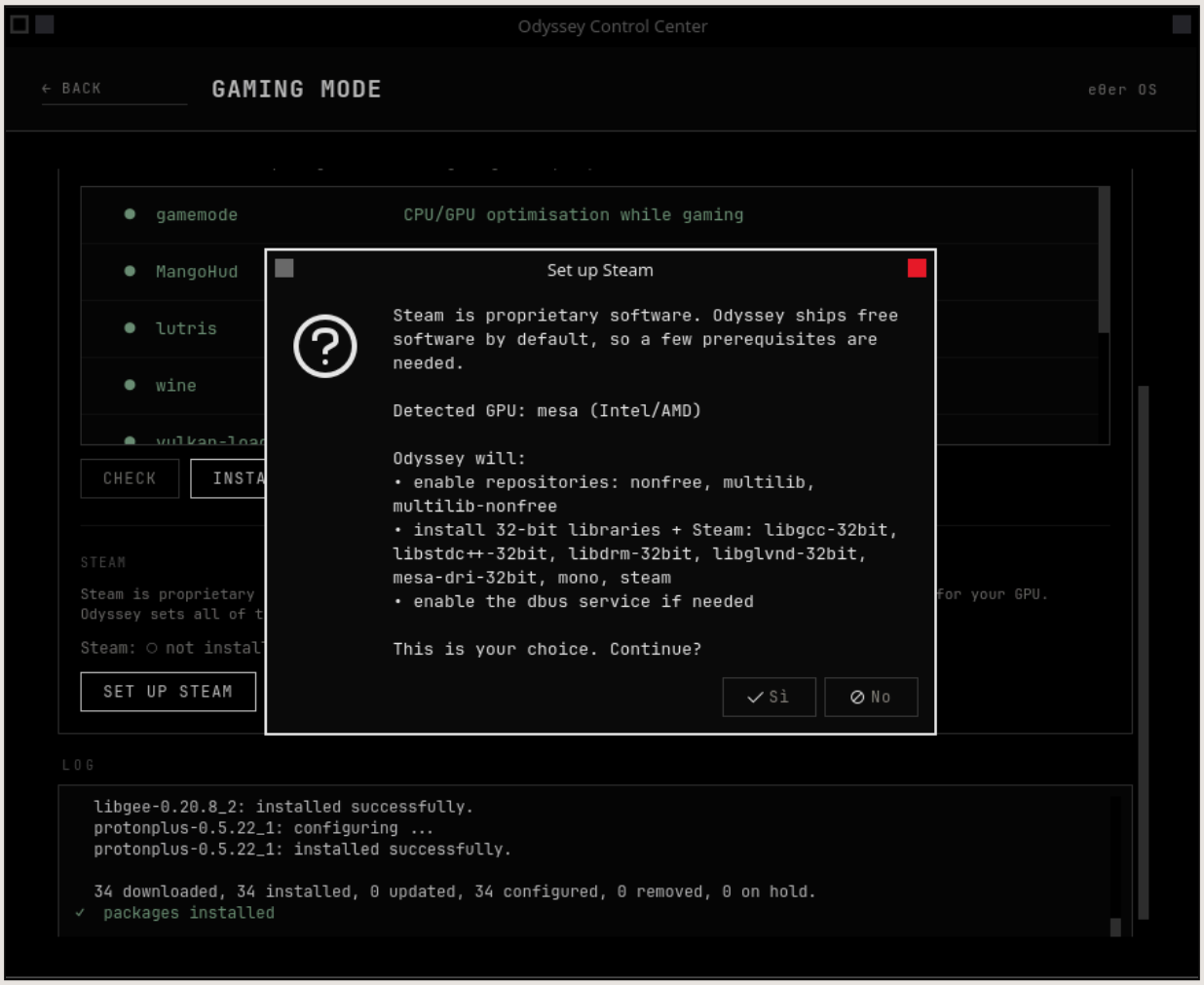

FIG · نافذة التأكيد: «Steam is proprietary software. Odyssey ships free software by default, so a few prerequisites are needed. Detected GPU: mesa (Intel/AMD).» مع قائمة نقطية بما سيُفعَّل ويُثبَّت بالضبط، تنتهي بـ«This is your choice. Continue?» وزرّي Sì / No.

تخبرك النافذة بالضبط بما سيحدث قبل أن توافق عليه: أيّ مستودعات ستُفعَّل (nonfree وmultilib وmultilib-nonfree)، وأيّ مكتبات 32-بت ستُثبَّت مطابقةً لمزوّد بطاقتك الرسومية **المكتشَف**، وأنّ خدمة dbus ستُفعَّل إن لم تكن مفعَّلة. ولا شيء مخبوء خلف التأكيد.

Sì 2

يؤكّد العملية. ويبثّ السجلّ التثبيت الحقيقي — تنزيل الحزم، والإعداد، ثمّ الاكتمال.

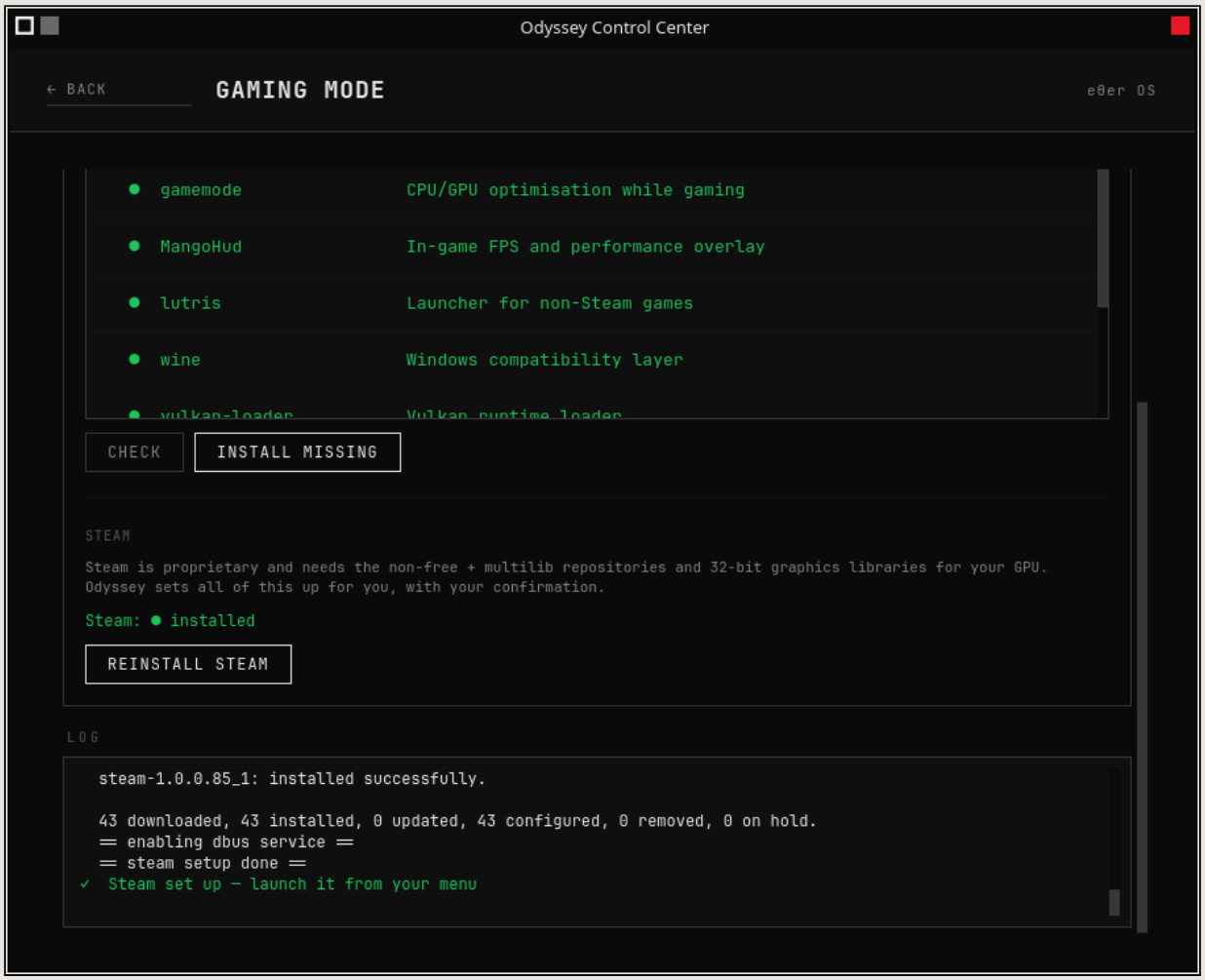

FIG · تبويب PACKAGES بعد التهيئة: كلّ الحزم خضراء، وSteam يعرض «● installed» مع REINSTALL STEAM، والسجلّ ينتهي بـ«Steam set up — launch it from your menu».

WHAT	Gaming Mode ← تبويب SET UP STEAM ← PACKAGES ← راجع النافذة ← Sì.
WHY	Steam برمجية احتكارية وتحتاج إلى مستودعات nonfree/multilib ومكتبات 32-بت مطابقة لبطاقتك الرسومية لا يفعّلها Void افتراضيًا — وضبط ذلك يدويًا عثرةٌ شائعة لأوّل مرّة على أيّ نظام قائم على Void.
RESULT	تُثبَّت المستودعات المطلوبة، وSteam نفسه، ومكتبات الرسوميات 32-بت المطابقة لبطاقتك بعينها — ثمّ أطلق Steam من قائمة تطبيقاتك بعدها.
RISK	منخفض — فهذا يفعّل مستودعات ويضيف حزمًا فقط؛ ولا يُزال شيء قائم ولا يُعاد إعداده. والنافذة تعرض قائمة كاملة بما سيحدث قبل أن تؤكّد.

9.3 GameMode و MangoHud

GameMode (وله تبويبه) هو عفريت الأداء لكلّ لعبة من Feral Interactive — فاللعبة المطلَقة عبره تطلب دفعة أداء مؤقّتة (منظّم المعالج، وأولوية الدخل والخرج) لعمليتها وحدها، وتُعاد تلقائيًا حين تُغلَق اللعبة. أمّا **MangoHud** (وله تبويبه أيضًا) فهو الطبقة المعروضة على الشاشة التي تُظهر عدد الإطارات وزمن الإطار وحمل المعالج والبطاقة ودرجات الحرارة أثناء اللعب. ويقدّم التبويبان أدوات تثبيت وفحص، ولـMangoHud إعدادًا سريعًا لمظهر الطبقة.

Proton 9.4

إدارة طبقة التوافق لتشغيل ألعاب ويندوز عبر Steam — تفحص ما هو مثبَّت وتثبّت إصدارات Proton إضافية، بما فيها بناءات المجتمع مثل Proton-GE، حين تحتاج لعبة بعينها إلى إصدار لا تشحنه Valve افتراضيًا.

> **متقدّم**
>
> ما بعد هذه النقطة: مسارات ملفّات الإعداد، والأوامر الخام، وما تفعله اللوحة تحت الغطاء.

9.5 متقدّم

ما الذي يكتبه ACTIVATE

الملفّات الكامنة نفسها التي تكتب فيها أزرار MAKE PERSISTENT في لوحتي Kernel و Memory — فوضع الألعاب حزمةٌ لا آليةٌ منفصلة:

```
# persisted sysctl values
$ cat /etc/sysctl.d/99-odyssey.conf
# CPU governor — not sysctl, sysfs-based, not persisted across reboot
# by this mechanism; re-apply after boot if you rely on it
$ cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

ما الذي ينفّذه SET UP STEAM فعليًا

نافذة التأكيد نفسها قريبة من إفصاح تقني كامل، لكن على نظام Mesa (إنتل/AMD) تحديدًا:

```
$ 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
# NVIDIA systems additionally pull nvidia-libs-32bit instead of mesa-dri-32bit —
# the panel detects your GPU vendor and adjusts the package list accordingly
```

GAMEMODE و MANGOHUD يدويًا

```
# launch any game with GameMode active
$ gamemoderun ./game_binary
# or, for a Steam game, launch options:
$ gamemoderun %command%
# MangoHud overlay, launch options:
$ mangohud %command%
# MangoHud's own config file, for tuning what the overlay shows
$ $EDITOR  /.config/MangoHud/MangoHud.conf
```

تثبيت إصدار PROTON-GE بعينه

يغلّف تبويب PROTON أداة protonplus أو مدير إصدارات مكافئًا — ويكافئه يدويًا تنزيل إصدار من GloriousEggroll/proton-ge-custom على GitHub واستخراجه في دليل أدوات التوافق الخاصّ بـSteam:

```
$ mkdir -p  /.steam/root/compatibilitytools.d
$ tar -xf GE-Proton-VERSION.tar.gz -C  /.steam/root/compatibilitytools.d
# then select it per-game in Steam: Properties → Compatibility
```

القياس عن بُعد

رغم الاسم، ليست هذه اللوحة إعدادًا تضبطه — بل مرقابٌ حيّ. فOdyssey لا يجمع عنك شيئًا ولا يرسل، فلا شيء هنا لتطفئه. وما تفعله هذه اللوحة بدلًا من ذلك أن تعرض عليك، في الزمن الحقيقي، كلّ اتصال صادر يجريه نظامك **أنت** — فيصير سؤال «بمن تتحدّث آلتي الآن؟» شيئًا تجيب عنه بالنظر، لا بتصديق ادّعاء. ولهذا بالضبط تسمّي نفسها **القياس المعكوس**: إذ تقلب المراقبة نحو النظام نفسه، لصالحك أنت.

10.1 جولة

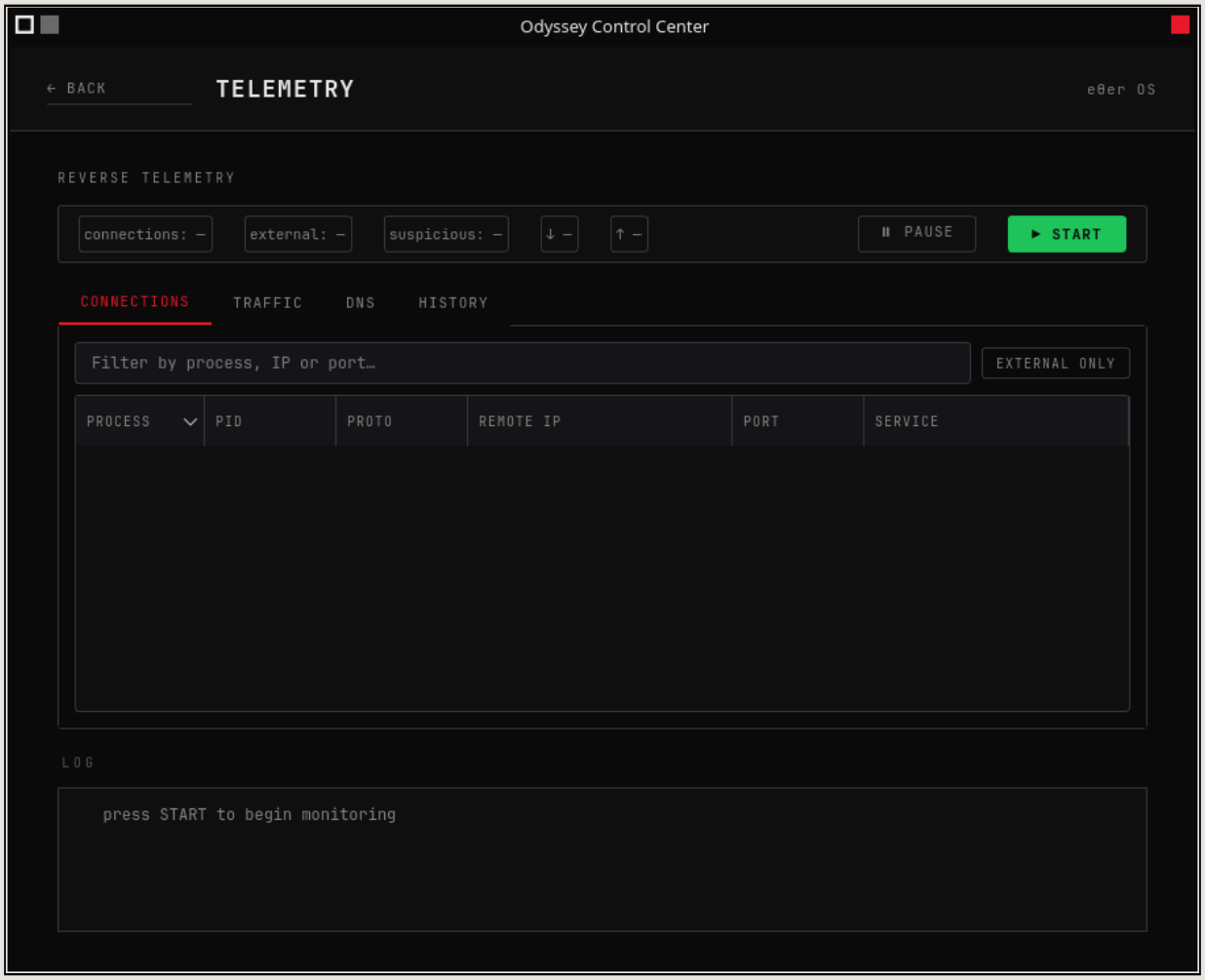

FIG · قبل البدء: connections و external و suspicious جميعها تعرض «–»، وزرّ START أخضر، والسجلّ يقرأ «press START to begin monitoring». والتبويبات الأربعة — CONNECTIONS و TRAFFIC و DNS و HISTORY — فارغة فوق جدول غير مملوء.

1 START

يبدأ المراقبة الحيّة — فيمتلئ شريط الحالة والتبويب الفاعل فورًا.

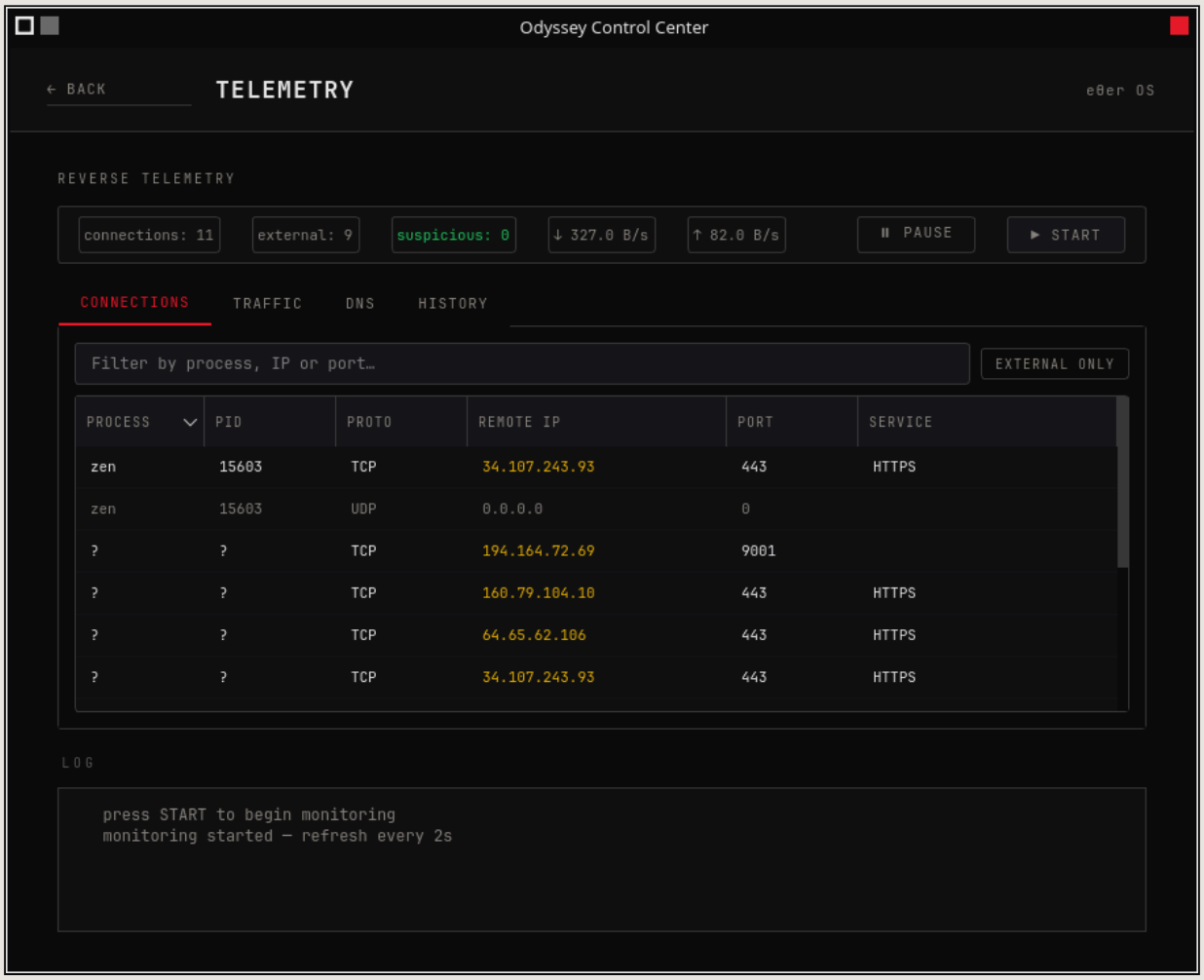

FIG • بعد البدء: «connections: 11 · external: 9 · suspicious: 0 · ↓327.0 B/s · ↑82.0 B/s»، وزرّ PAUSE متاح الآن، وجدول CONNECTIONS مملوء بصفوف حقيقية — العملية «zen» عبر TCP إلى 34.107.243.93:443 (HTTPS)، وعدّة صفوف موسومة بـ«?» في خانتَي العملية والمعرّف مع اتصالات إلى عناوين أخرى.

يعطي شريط الحالة في الأعلى مجاميع جارية — الاتصالات، والخارجية (في مقابل المحلية)، والمشبوهة، ومعدّلا التنزيل والرفع الحاليان — ويجمّد زرّ **PAUSE** العرض دون فقدان ما التُقِط، ويستأنفه **START**.

التبويبات الأربعة

التبويب	ماذا يعرض
CONNECTIONS	صفّ واحد لكلّ اتصال فاعل: العملية، ومعرّفها، والبروتوكول، والعنوان البعيد، والمنفذ، والخدمة المعروفة لذلك المنفذ (443 ← HTTPS، و22 ← SSH). وقابلٌ للفرز بأيّ عمود.
TRAFFIC	الإنتاجية لكلّ واجهة — معدّل التنزيل والرفع الآن، مع مجاميع جارية منذ بدء المراقبة.
DNS	بثّ حيّ لعمليات ترجمة أسماء النطاقات وقت حدوثها — وهو أوضح من عنوان مجرّد غالبًا، لأنّ اسم المضيف يخبرك **بماذا** لا **بأين** فقط.
HISTORY	كلّ ما التقطته الجلسة، سجلًّا قابلًا للتمرير — وهو مفيد لالتقاط شيء عابر لم يعد اتصالًا فاعلًا حين تنظر.

والصفوف التي تظهر فيها العملية بعلامة ? (وهي مرئية في اللقطة) تعني أنّ الاتصال التُقِط لكنّ العملية المُنشِئة تعذّر تحديدها من معرّفها في تلك اللحظة — وهو أمر شائع في الاتصالات القصيرة جدًّا. وفوق الجدول صندوق **ترشيح** يضيّق النتائج بحسب العملية أو العنوان أو المنفذ، وزرّ **EXTERNAL ONLY** يخفي الحركة المحلية البحتة.

WHAT	START ← Telemetry ← راقب CONNECTIONS، أو افرز بحسب PROCESS أو REMOTE IP لتقصّي شيء بعينه.
WHY	هذا هو الجواب المباشر والواقعي عن سؤال «هل على هذه الآلة شيء يبلغ الشبكة وما كان ينبغي له؟» — مقروءًا من جدول اتصالات النواة نفسها، لا من ادّعاء أيّ مورّد بشأن برمجيته.
RESULT	صورة حيّة ودقيقة لكلّ اتصال صادر، تُحدَّث باستمرار ما دامت المراقبة تعمل ولم تُوقَف مؤقّتًا.
RISK	لا شيء — العملية للقراءة فقط تمامًا.

والاتصالات الصادرة عن عمليات لا سبب عاديًّا لها لتبلغ الشبكة من تلقائها — صدَفة مجرّدة، أو مفسّر برمجة يعمل بمفرده — تُوسَم تلقائيًا تحت «suspicious» في شريط الحالة، فيبرز الشاذّ بدل أن يُدفن وسط الحركة الطبيعية لمتصفّحك وخدمات نظامك.

> ◂ **متقدّم**
>
> ما بعد هذه النقطة: مسارات ملفّات الإعداد، والأوامر الخام، وما تفعله اللوحة تحت الغطاء.

10.2 متقدّم

من أين تأتي البيانات

تقرأ اللوحة جداول الاتصالات على مستوى النواة نفسها التي تقرأها أيّ أداة قياسية، وتحدّثها بفاصل قصير ما دامت المراقبة فاعلة — بلا مشغّل خاصّ ولا وحدة نواة خاصّة بها. ومكافئات سطر الأوامر:

```
# every open socket, with the owning process — same fields as CONNECTIONS
$ sudo ss -tupn
# per-interface throughput — same numbers as TRAFFIC
$ ip -s link
# a one-shot live view, if you want it outside the panel
$ sudo nethogs
```

استعلامات DNS يدويًا

```
# live capture of outgoing DNS queries (needs a packet-capture tool)
$ sudo tcpdump -i any -n port 53
```

لماذا تعرض بعض الصفوف علامة «?» في خانة العملية

مطابقة اتصال حيّ بالعملية التي تملكه تقتضي قراءة `/proc/PID/fd` لكلّ عملية مرشَّحة في لحظة الالتقاط — فإن فُتح الاتصال وأُغلق أسرع من فاصل تحديث اللوحة، أو كانت العملية المالكة قد انتهت أصلًا، فلا نافذة أمام اللوحة لإجراء تلك المطابقة، فتعرض `?` بدل التخمين. وهذا قيدٌ زمني لا علامةَ على إخفاء شيء.

عن النظام

صفحة تشخيص واحدة شاملة للآلة، تُحدَّث في كلّ مرّة تفتح فيها التبويب، فتعكس دائمًا ما تغيّر في اللوحات الأخرى دون حاجة إلى إعادة فتح مركز التحكّم.

11.1 جولة

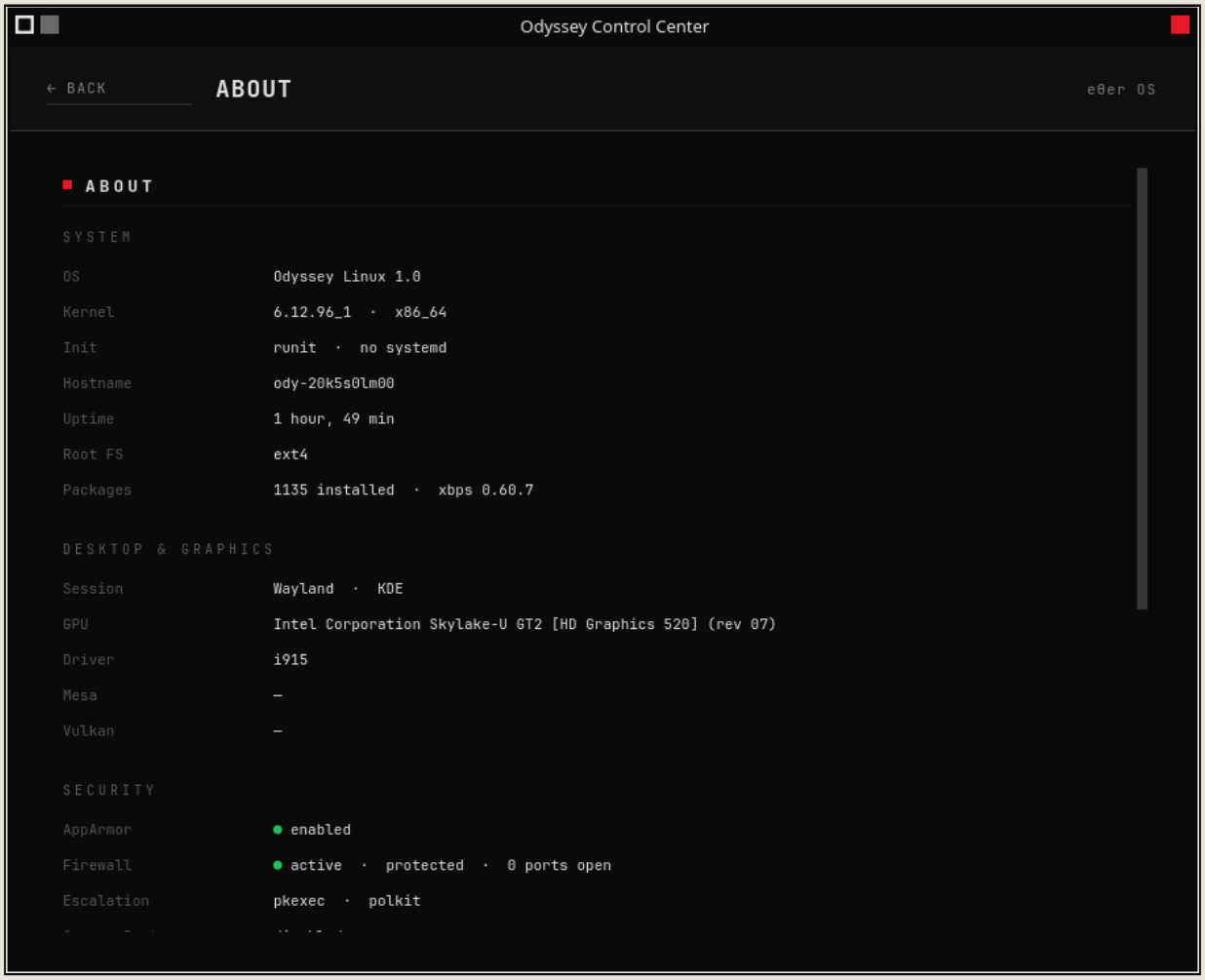

FIG · اللوحة عند أعلاها: قسم SYSTEM يعرض النظام «Odyssey Linux 1.0»، والنواة 6.12.96_1 · x86_64، والإقلاع runit · بلا systemd، واسم المضيف، ومدّة التشغيل، ونظام ملفّات الجذر ext4، والحزم «1135 installed · xbps 0.60.7»؛ وقسم DESKTOP & GRAPHICS أدناه بالجلسة والبطاقة والمشغِّل و Mesa و Vulkan؛ ثمّ يبدأ قسم SECURITY بإظهار «● enabled AppArmor» وجدار الحماية «active · protected · 0 ● ports open».

ثلاثة أقسام بالترتيب: **SYSTEM** (النواة، ونظام الإقلاع، واسم المضيف، ومدّة التشغيل، ونظام الملفّات، وعدد الحزم)، و**DESKTOP & GRAPHICS** (نوع الجلسة، والبطاقة الرسومية، والمشغّل، ودعم Mesa و Vulkan)، وأسفل منهما **SECURITY** (حالة AppArmor وجدار الحماية بمؤشّرات ملوَّنة لا بنصّ فحسب — أخضر للمفعَّل والفاعل) و**HARDWARE** (اللوحة الأمّ والصوت).

WHAT	About ← مرّر عبر الأقسام الثلاثة لتفحص أيّ قيمة — إصدار النواة، والمشغّل المستعمَل، وهل AppArmor وجدار الحماية فاعلان فعلًا.
WHY	إنّها أسرع طريقة لتأكيد حالة النظام كلّه في مكان واحد، بلا فتح خمس لوحات مختلفة.
RESULT	لقطة كاملة ومحدَّثة — تُنعَش عند كلّ زيارة للتبويب، فلا تعرض أبدًا معلومات قديمة من جلسة سابقة.
RISK	لا شيء — العملية للقراءة فقط تمامًا.

11.2 نسخ التقرير

زرّ واحد في أعلى اللوحة، **COPY REPORT**، يضع الصفحة كاملةً في حافظتك نصًّا عاديًا — كلّ قسم وكلّ قيمة — جاهزةً للّصق مباشرةً في منشور منتدى أو في مدخل متعقّب علل.

WHAT	COPY REPORT ← About ← الصقه في بلاغ العلّة أو منشور المنتدى.
WHY	بلاغات العلل لا تنفع إلّا بقدر تفاصيل النظام المرفقة بها، وكتابة إصدار النواة والمشغّل وحالة AppArmor وعشرة حقول أخرى يدويًا هي بالضبط نوع الاحتكاك الذي يجعل الناس يتجاوزونها.
RESULT	تقرير نظام كامل ومنسَّق تنسيقًا صحيحًا في حافظتك، جاهزٌ للّصق — وهي المعلومات ذاتها التي كان المشرف سيضطرّ إلى سؤالك عنها حقلًا حقلًا.
RISK	لا خطر على نظامك. والتقرير معلومات تشخيصية (إصدارات وحالات وطراز عتاد) — ولا يتضمّن ملفّات شخصية ولا بيانات اعتماد — لكن كأيّ تفريغ تشخيصي، اقرأه قبل نشره في مكان عامّ إن لم تكن متيقّنًا ممّا يحويه.

◂ متقدّم

ما بعد هذه النقطة: مسارات ملفّات الإعداد، والأوامر الخام، وما تفعله اللوحة تحت الغطاء.

11.3 متقدّم

من أين يأتي كلّ حقل

لا شيء في هذه اللوحة مخترَع ولا مخزَّن من وقت التثبيت — فكلّ قيمة تُسبَر حيًّا في كلّ مرّة يُفتح فيها التبويب:

```
$ 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
```

وقسم الأمان تحديدًا:

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

لماذا يستحقّ ABOUT أن يُفحَص أوّلًا

لأنّه يعيد سبر النظام من جديد عند كلّ زيارة، فإنّ About أسرع طريقة لتأكيد أنّ تغييرًا أجريته في لوحة أخرى قد سرى فعلًا — افرض وضع جدار حماية في Firewall، ثمّ افحص قسم Security في About بدل إعادة فتح Firewall نفسها. فهو مبنيّ ليكون تدقيق «هل نجح ذلك فعلًا» لبقيّة مركز التحكّم.

مشغّل NVIDIA

> **بانتظار الدمج**
>
> هذا الفصل نائبٌ مؤقّت. فوحدة NVIDIA لا تظهر في مركز التحكّم إلّا على الآلات التي يُكتشَف فيها عتاد NVIDIA، ولقطات الشاشة الخاصّة بها لا يمكن التقاطها إلّا على آلة كهذه — وستُقدَّم على حدة ويُستكمَل هذا الفصل في مراجعة لاحقة لهذا الدليل. وقد أُبقي أخيرًا، بعد About، تحديدًا كي لا تزيح إضافته لاحقًا أرقام صفحات أيّ فصل آخر ولا إحالاته.

وما هو معلوم سلفًا وسيكون أساس الفصل المكتمل: تحمل اللوحة شارة **BETA** ظاهرة وملاحظةً صريحة بأنّ المشغّل نفسه احتكاري — تحزّمه Void من مثبِّت NVIDIA الرسمي ببصمة مثبَّتة، لكنّه غير قابل للتدقيق باستقلال، وهي الصراحة ذاتها التي يطبّقها Odyssey على نفسه. وهي تكتشف بطاقتك بعينها بمقابلتها بقاعدة بيانات مدمجة وتختار فرع المشغّل الصحيح تلقائيًا (الحالي لبطاقات Turing وما بعدها، أو أحد ثلاثة فروع قديمة للبطاقات الأقدم، مع الرجوع إلى المشغّل الحرّ `nouveau` حين لا ينطبق أيّ مشغّل احتكاري). ويجري التثبيت في خطّ أنابيب متتبَّع من سبع خطوات مع متتبّع تقدّم ظاهر، وتطبع اللوحة أمر التعافي الخاصّ بها مقدَّمًا — `odyssey-nvidia --revert` من طرفية نصّية — قبل أن تنقر التثبيت أصلًا. وعلى الحواسيب المحمولة الهجينة يظهر قسم PRIME لتفريغ التصيير تلقائيًا، فيتيح تشغيل لعبة أو برنامج بعينه على بطاقة NVIDIA بينما يبقى كلّ ما عداه على البطاقة المدمجة حفاظًا على البطارية.