ODYSSEY LINUX · DOCUMENTAÇÃO DE REFERÊNCIA

Odyssey Control Center — Guia Prático

Cada painel, seção por seção. O que ele faz, o que muda, e no que clicar — escrito para quem nunca usou antes.

Launch Edition

Don’t trust me. Verify me.

— nobody

Odyssey Linux — Odyssey Control Center: Guia Prático
Launch Edition. Cobre o Control Center tal como é distribuído na ISO da Launch Edition — dez painéis completos, de Boot a About, mais um capítulo provisório para o módulo do driver NVIDIA (visível apenas em hardware NVIDIA, aguardando capturas de tela de uma máquina que o possua).

Este guia é o complemento prático do Security and Hardening Guide, que cobre os painéis Firewall e AppArmor com total profundidade técnica — aqui você encontra o guia visual passo a passo e uma referência cruzada para o resto.

Cada tela, botão e comportamento descrito aqui foi lido do código- fonte real do Control Center e de capturas de tela reais, não de memória.

Os rótulos dos botões, os nomes dos painéis e abas, os caminhos dos arquivos e os comandos permanecem em inglês porque são a interface real do programa: procurá-los traduzidos no Control Center não os encontraria.

SUMÁRIO

CHAPTER 1

O que é o Control Center

O Odyssey Control Center são dez pequenos programas independentes atrás de um único lançador — um para cada área do sistema: Boot, Kernel, Memory & Swap, Firewall, AppArmor, Privacy, Containers, Gaming Mode, Telemetry, About, e (em hardware NVIDIA) o módulo do driver NVIDIA.

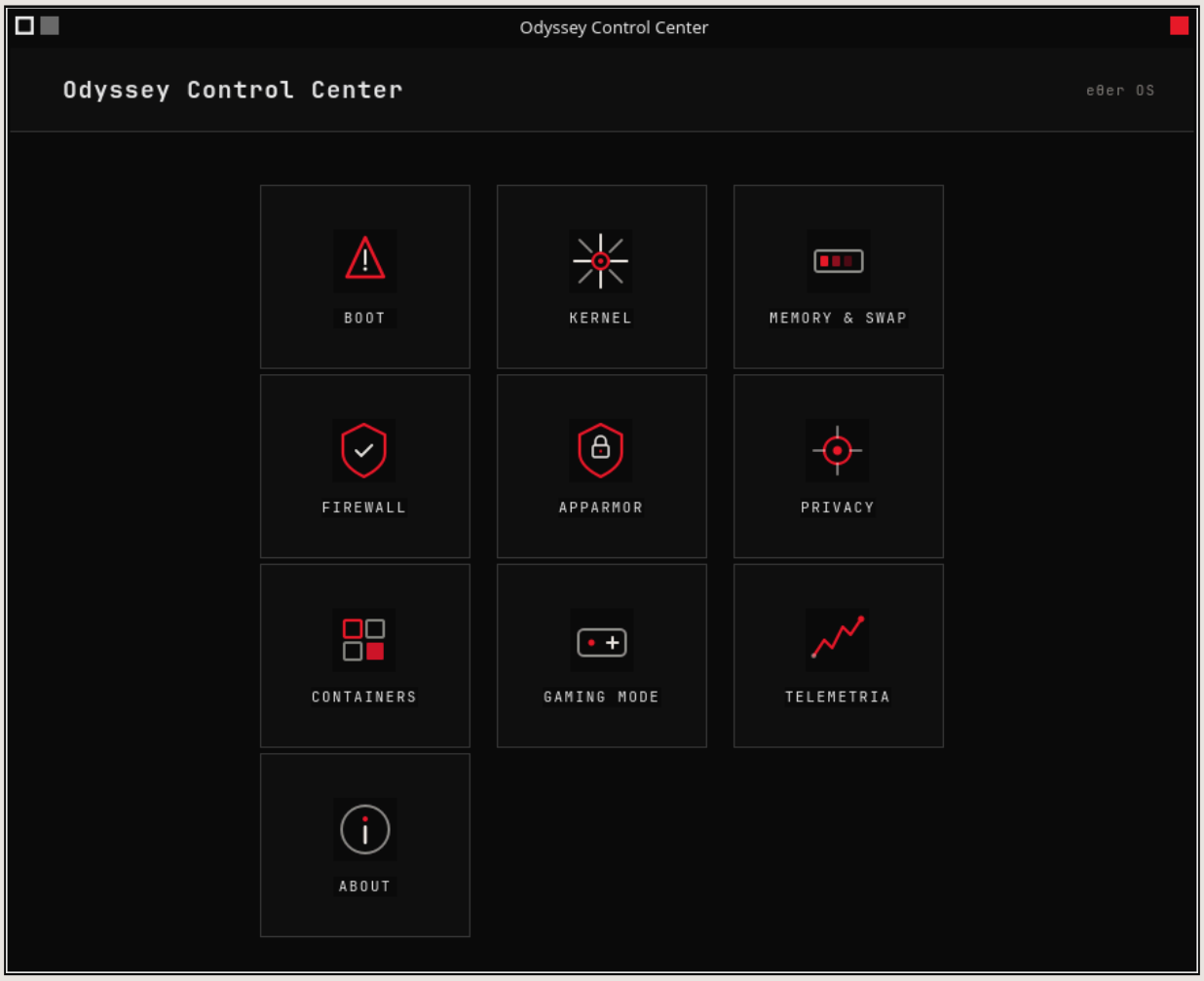

FIG · A tela inicial do Control Center — um bloco para cada painel.

Cada painel é uma camada gráfica fina sobre o mecanismo real do Linux por baixo: a configuração do GRUB, a própria interface sysctl do kernel, `nftables`, `podman`. Um painel nunca inventa seu próprio estado privado — ele lê os arquivos e comandos reais, mostra a você, e escreve exatamente o que você pediu. Nada do que ele faz está oculto, e tudo o que ele faz também pode ser feito manualmente, de um terminal, com os comandos exatos que este guia mostra ao lado de cada botão.

Este é um guia prático, não uma referência de cada opção — ele responde a "o que eu clico, e o que acontece se eu fizer isso?" Cada capítulo tem a mesma estrutura em duas partes: primeiro um **guia visual**, captura após captura, botão após botão, escrito para quem nunca abriu o Control Center antes — depois uma seção **ADVANCED**, sinalizada por um banner preto, que entra no detalhe: caminhos exatos dos arquivos, os comandos reais que cada botão executa por baixo, e o que fazer no terminal

se você preferir pular o painel completamente. Leia só o guia visual, ou pule direto para Advanced se você já conhece Linux — ambos são completos por si só.

Para os dois painéis com implicações reais de segurança — Firewall e AppArmor — a referência aprofundada é o **Security and Hardening Guide**; aqui você encontra o guia visual, o essencial, e uma referência cruzada.

1.1 Padrões que você verá em todo painel

Algumas convenções se repetem nos dez painéis — aprendê-las uma vez vale para todos.

A SOLICITAÇÃO DE SENHA, UMA ÚNICA VEZ POR PAINEL

A maioria dos painéis precisa ler ou escrever arquivos que pertencem ao root — a configuração do GRUB, sysctl, o conjunto de regras do firewall. Na primeira vez que um painel precisa disso, você recebe uma única caixa de diálogo de autenticação. Os painéis são construídos para agrupar suas leituras: abrir Boot, por exemplo, lê a configuração do GRUB **e** o estado do firmware EFI em uma única solicitação, então você digita sua senha uma vez, não duas. Reabrir um painel na mesma sessão não pede de novo; as ações seguintes (aplicar uma mudança, recarregar) mostram seu próprio progresso no log do painel.

O RASTREADOR DE ETAPAS

Os painéis que guiam você pela instalação de algo antes que se torne utilizável — os-prober em Boot, SearXNG e Tor em Privacy, o motor de contêineres em Containers — mostram um pequeno rastreador numerado no topo: uma fileira de etapas, com a atual destacada, e um único botão de ação que sempre faz **a próxima etapa**, qualquer que seja. Você nunca precisa adivinhar o que clicar primeiro.

O LOG NO RODAPÉ

Todo painel termina em um pequeno log rolável. Não é decoração — é o comando exato que foi executado, sua saída, e um veredicto colorido (verde para sucesso, vermelho para falha). Se um painel faz algo que você não entende, o log é onde está a resposta real.

TRY VS. MAKE PERSISTENT

Um par de botões recorrente, especialmente em Kernel e Memory: **TRY NOW** (ou **APPLY NOW**) muda o sistema em execução imediatamente, e reverte no próximo reinício se você não fizer mais nada. **MAKE PERSISTENT** escreve o mesmo valor em um arquivo de configuração sob `/etc/`, para que sobreviva também aos reinícios. Essa separação existe de propósito — testar um valor não custa nada, e nada é permanente até você dizer, duas vezes.

CHAPTER 2

Boot

O painel Boot tem duas abas em qualquer máquina que inicializa em modo UEFI: **GRUB** (o próprio menu de inicialização) e **EFI** (a camada de firmware abaixo dele). Em máquinas mais antigas, só BIOS, existe apenas GRUB, porque a camada EFI abaixo simplesmente não existe.

> **DUAS CAMADAS, NÃO UMA**
>
> **GRUB** é o menu de inicialização que você vê na tela — ele decide qual sistema operacional inicia depois que o próprio GRUB já está em execução. O **firmware EFI/UEFI** é uma camada abaixo do GRUB — ele decide para qual bootloader a máquina entrega o controle primeiro, antes mesmo do GRUB ter carregado. A maioria das pessoas nunca toca na aba EFI; ela existe para os casos que o GRUB não alcança, como uma segunda instalação do Windows que gerencia sua própria entrada de inicialização de forma independente.

2.1 Guia visual — encontrar outro sistema operacional e definir o padrão

A aba GRUB abre na seção **Other OS detection**. O Odyssey não escaneia outros sistemas por padrão — a maioria das instalações só tem Odyssey no disco, e escanear custa tempo a cada regeneração do menu de inicialização à toa. Ativar isso é uma sequência guiada de três etapas, mostrada como um pequeno rastreador numerado.

1. **`Installed`**

   A ferramenta de escaneamento, os-prober, ainda não está no sistema. Um único botão: **INSTALL OS-PROBER**.

2. **`Enable scan`**

   A ferramenta está instalada, mas o GRUB ainda está configurado para ignorá-la — o padrão do Odyssey. Um único botão: **ENABLE SCAN**.

3. **`Scan`**

   O GRUB agora vai usá-la. Um único botão, o mesmo que você vai usar de novo sempre que adicionar ou remover outro sistema no futuro: **SCAN & REGENERATE**.

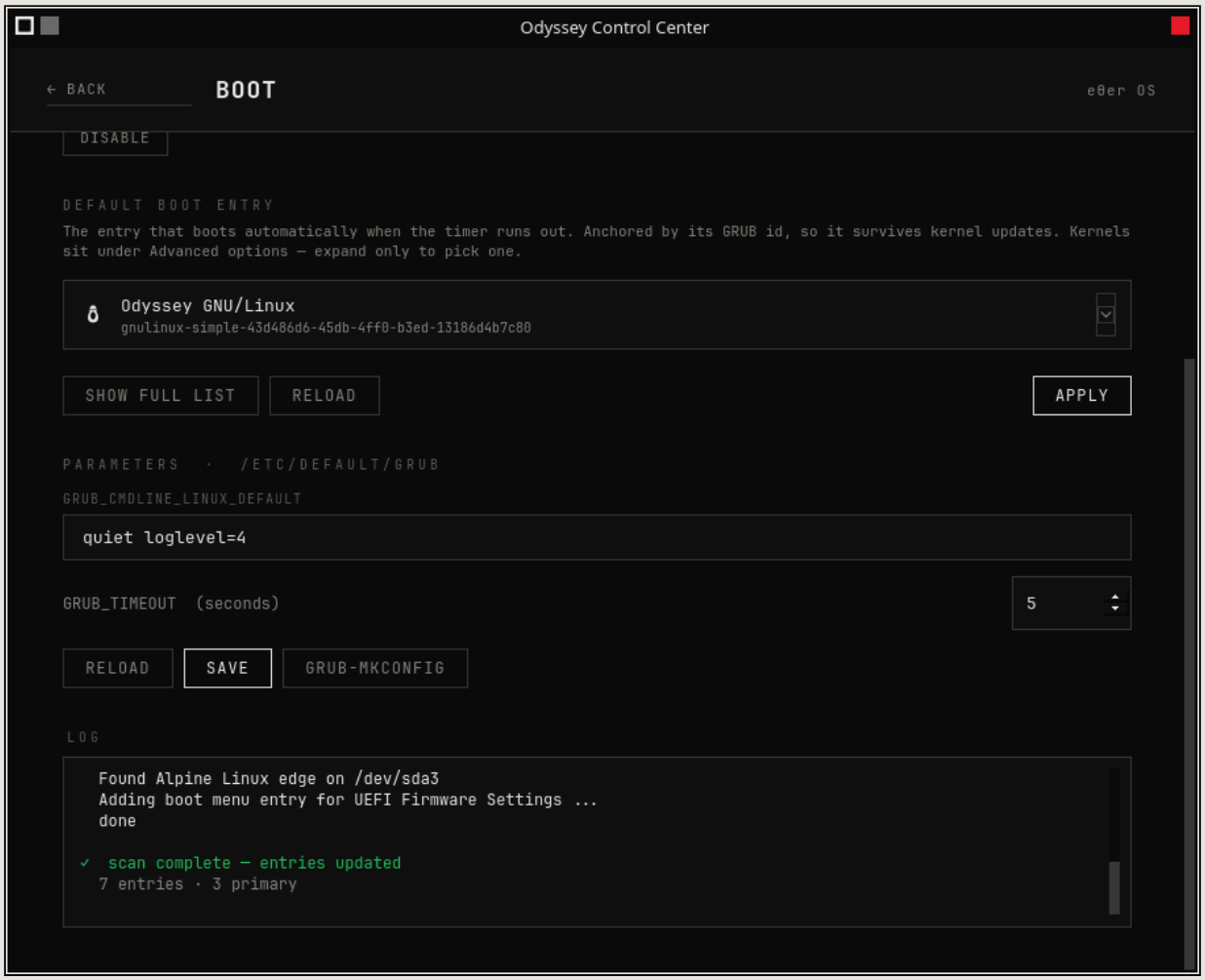

FIG · As três etapas em verde — os-prober instalado, escaneamento habilitado, pronto. O botão de ação único sempre corresponde à etapa atual.

Clique nas etapas (cada uma pede autenticação novamente só se necessário) e execute o escaneamento. O painel mostra exatamente o que encontrou antes de você sair da seção:

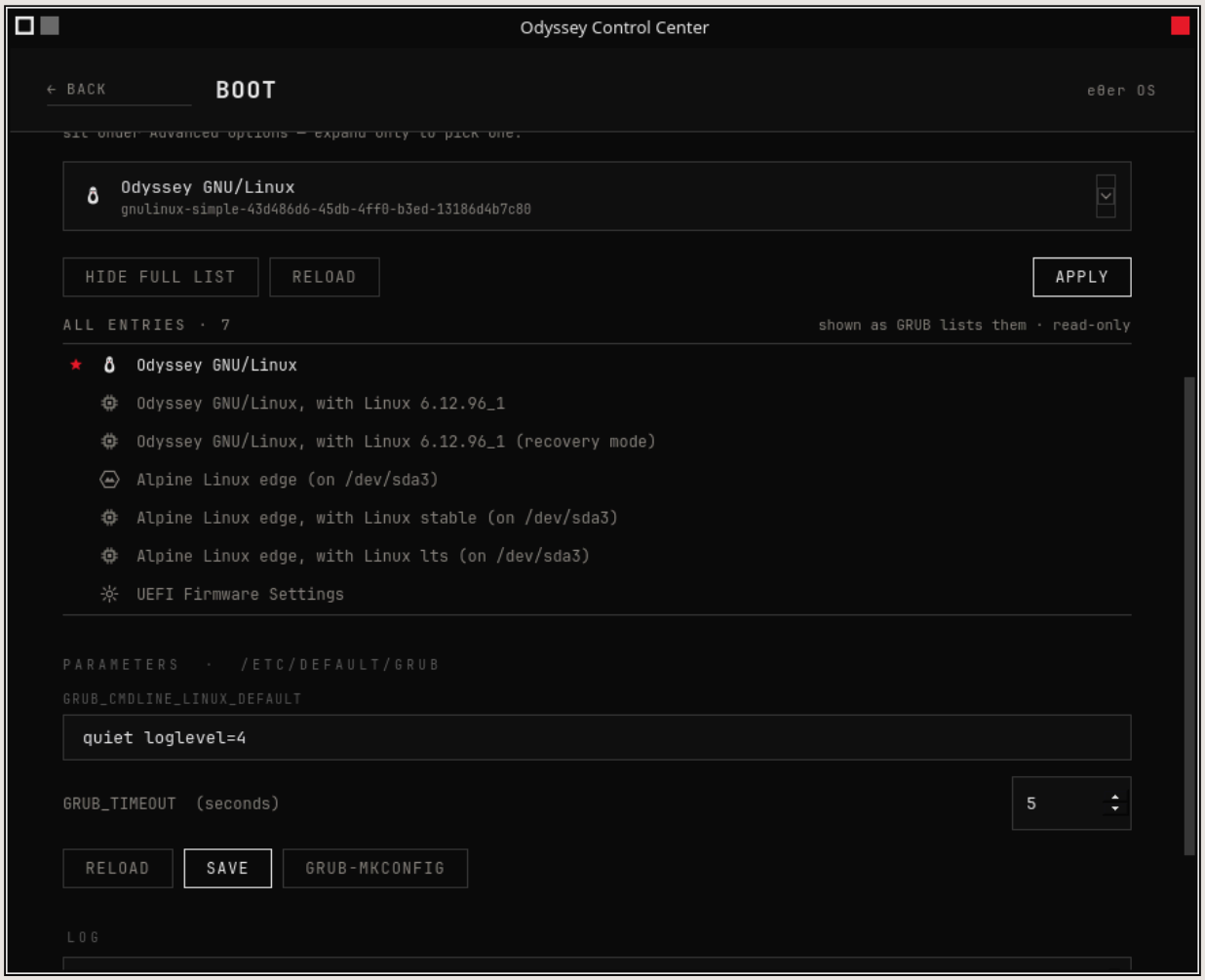

FIG · Escaneamento concluído: Alpine Linux encontrado em /dev/sda3, junto com as próprias entradas do Odyssey e a entrada UEFI Firmware Settings — 7 entradas, 3 primárias.

O log confirma isso em termos simples — **"scan complete — entries updated · 7 entries · 3 primary."** Tudo que o os-prober encontrar agora aparece automaticamente na seção abaixo, pronto para ser escolhido.

2.2 Guia visual — escolher o que inicia por padrão

Role até **Default boot entry**. Ela mostra o padrão atual como uma linha; clicar nela abre um menu suspenso com tudo que você pode escolher em vez disso.

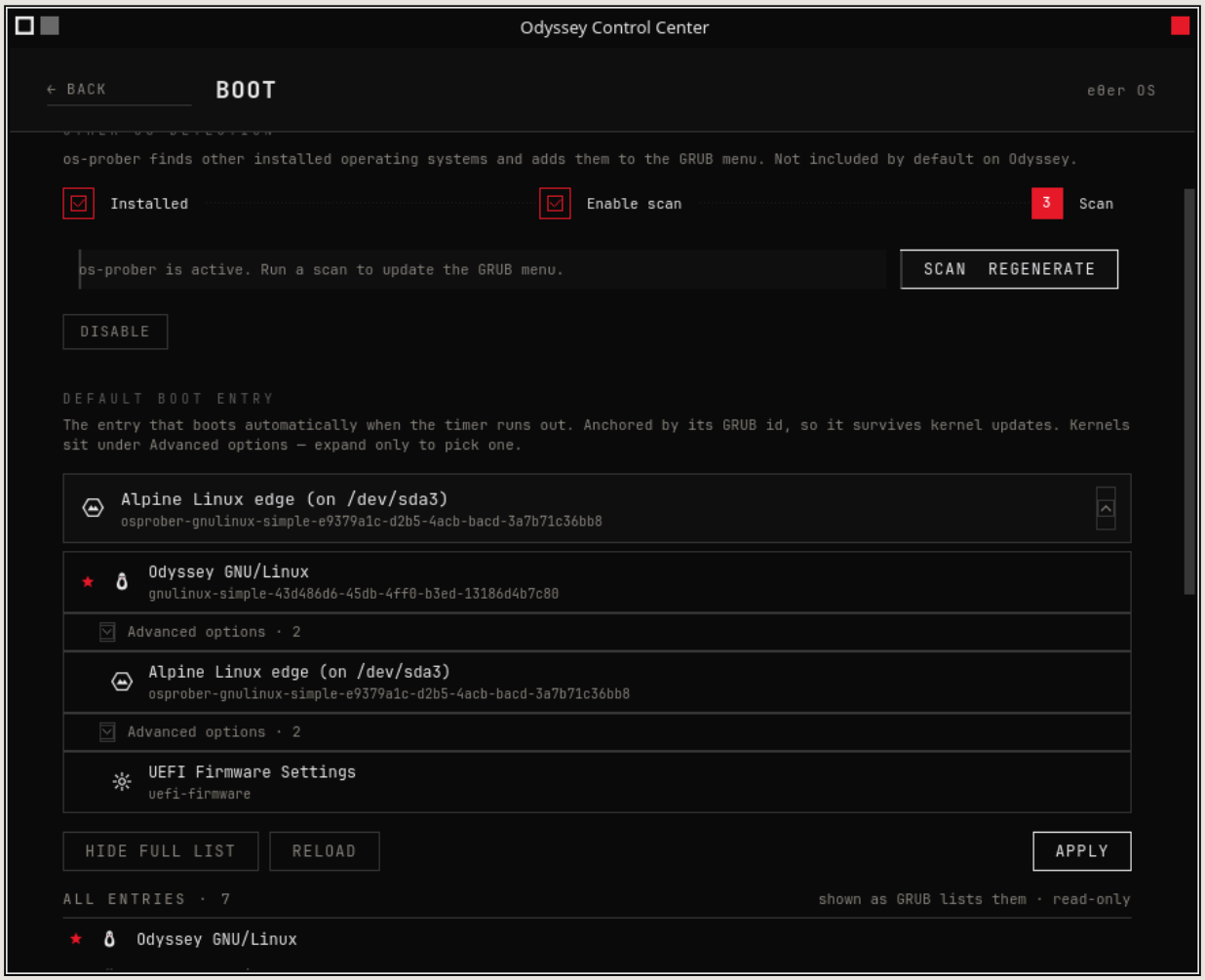

FIG · O menu suspenso aberto, com Alpine Linux selecionado como escolha pendente — ainda não aplicada.

Escolher uma entrada da lista **coloca a escolha em espera** — ela ainda não tem efeito. Dá pra ver que está só em espera porque a caixa ainda mostra a nova seleção, mas nada foi escrito. O botão que confirma é o **APPLY**, à direita:

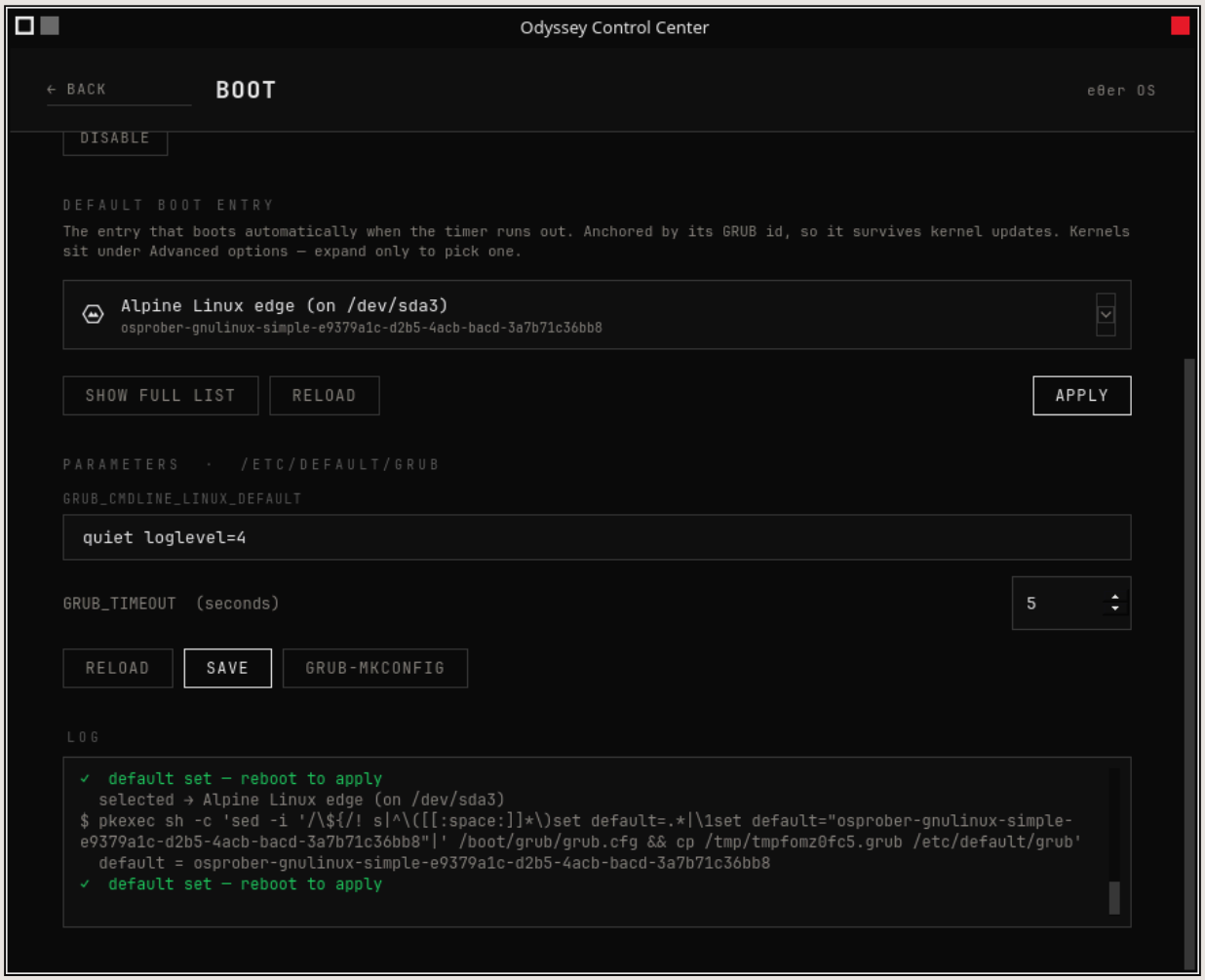

FIG · Depois de APPLY: o log confirma “default set — reboot to apply”, e o seletor agora mostra Alpine Linux como o padrão ativo.

WHAT	Boot → aba GRUB → clique na entrada padrão atual → escolha uma diferente no menu suspenso → clique em **APPLY**.
WHY	Você usa dual-boot e quer que um sistema diferente inicie automaticamente, ou uma atualização do kernel deixou um kernel antigo selecionado e você quer voltar ao mais recente.
RESULT	A entrada escolhida inicia automaticamente a partir do próximo reinício. A mudança fica ancorada ao id estável dessa entrada, então sobrevive a futuras atualizações do kernel.
RISK	Nenhum para seus dados — no pior caso você escolhe a entrada errada e volta a trocá-la de novo.

SHOW FULL LIST mostra cada entrada exatamente como o próprio GRUB as lista — somente leitura, agrupadas de forma que as variantes por kernel individual fiquem sob uma linha expansível “Advanced options” em vez de lotar a lista principal. **RELOAD** recarrega a configuração do GRUB se você acha que ela mudou fora do painel.

2.3 Guia visual — a aba EFI

Clique na aba **EFI** no topo. Esse é o nível de firmware — o que o próprio menu de inicialização da sua placa-mãe mostra antes mesmo de qualquer sistema operacional ter começado.

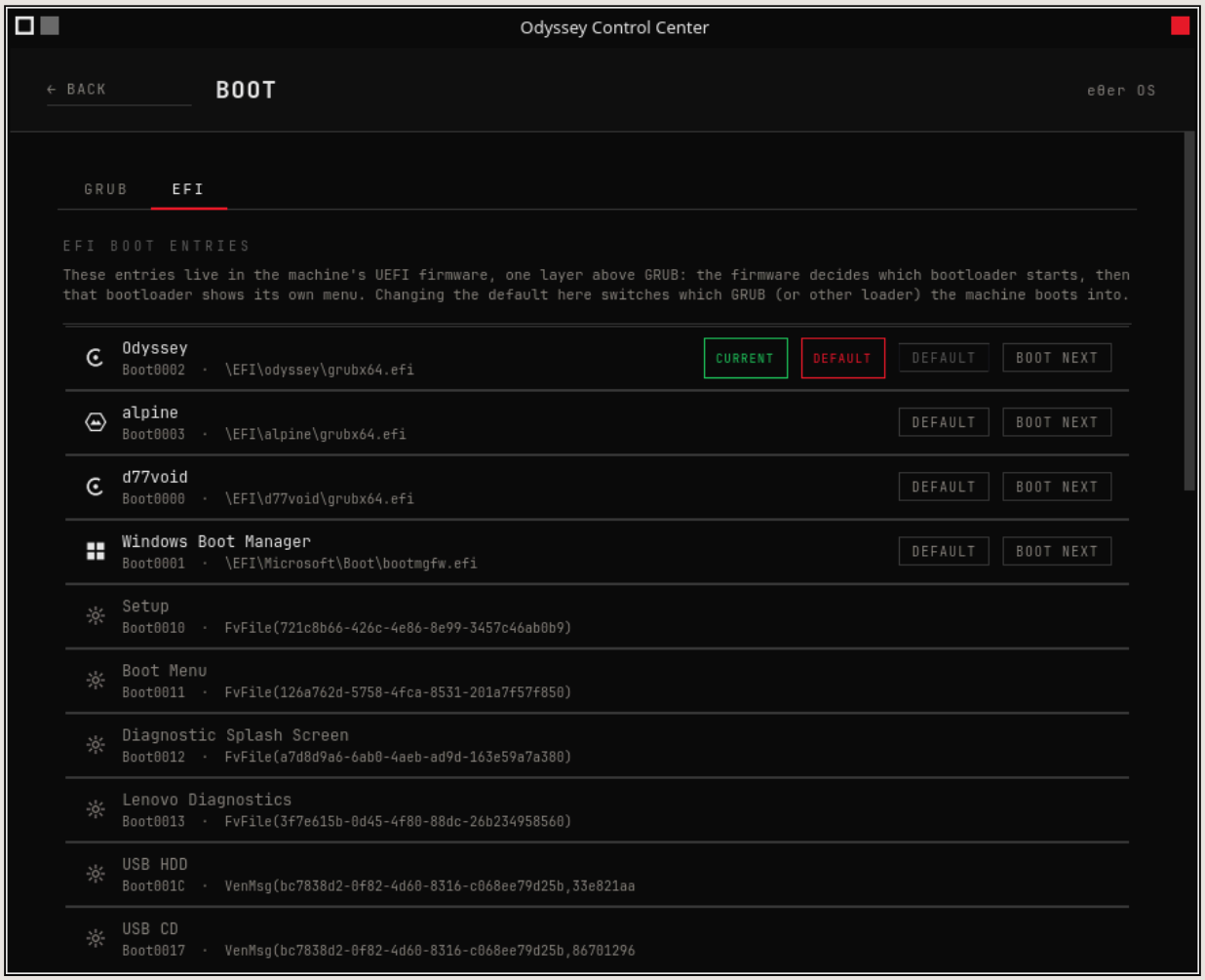

FIG · A aba EFI: Odyssey marcado como CURRENT e DEFAULT em verde/vermelho, alpine, d77void e Windows Boot Manager cada um com seus próprios botões DEFAULT / BOOT NEXT, mais as entradas utilitárias do firmware abaixo.

Cada linha é uma entrada de firmware — seu nome, seu id de GRUB (`Boot0002` e assim por diante), e o arquivo `.efi` exato para o qual aponta. Dois botões por linha:

BOTÃO	O QUE FAZ
DEFAULT	Torna essa entrada o padrão permanente do firmware — ela inicia em todo reinício a partir de agora.
BOOT NEXT	Uma exceção pontual — essa entrada inicia só no **próximo** reinício, depois a ordem normal retoma automaticamente. Um banner aparece no topo enquanto uma exceção está pendente, com um botão CANCEL.

WHAT	Boot → aba EFI → encontre a entrada que você quer (ex. “Odyssey” ou “Windows Boot Manager”) → clique em DEFAULT, ou BOOT NEXT para uma exceção pontual.
WHY	Se o próprio menu de inicialização da sua placa-mãe está iniciando o bootloader errado por completo — não é uma configuração do GRUB, é do firmware — esse é o único lugar que corrige isso.
RESULT	DEFAULT: a entrada vai para o topo da ordem de inicialização do firmware de forma permanente. BOOT NEXT: inicia uma vez, depois reverte.
RISK	Baixo, e sempre recuperável a partir dessa mesma aba ou da própria tela de configuração UEFI da sua placa-mãe (geralmente Del ou F2 ao ligar) se algo iniciar de forma inesperada.

2.4 Parâmetros do kernel

A última seção da aba GRUB é a linha de comando do kernel em texto puro — editável como um campo de texto normal, sem exigir sintaxe do GRUB — e o tempo limite do menu em segundos. **SAVE** os escreve em `/etc/default/grub`; **GRUB-MKCONFIG** regenera o menu de inicialização real a partir desse arquivo, a etapa que torna um parâmetro salvo efetivo na próxima inicialização. O painel sempre reaplica sua escolha padrão atual quando regenera, então isso nunca reseta qual sistema inicia por padrão.

```
▸ ADVANCED

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

2.5 Advanced

O QUE REALMENTE ACONTECE

A configuração GRUB do Odyssey é **plana** — sem blocos `submenu`. Cada kernel é um `menuentry` de nível superior, o que significa que um id estável (não uma posição numérica) sempre pode selecioná-lo corretamente, mesmo entre atualizações de kernel. O agrupamento "Advanced options" que você vê no seletor é pura conveniência visual construída combinando o UUID compartilhado entre a entrada simples de um sistema operacional e suas variantes por kernel — o próprio arquivo do GRUB não tem esse agrupamento.

Definir um padrão escreve em dois lugares na mesma ação: o `GRUB_DEFAULT` persistente em `/etc/default/grub`, e — porque o `grub.cfg` pode não ser regenerado imediatamente — uma edição direta e cirúrgica via `sed` da linha ativa `set default="..."` no próprio `/boot/grub/grub.cfg`, pulando o ramo one-shot `${next_entry}` que o `grub-reboot` usa. É por isso que um padrão tem efeito na próxima inicialização sem forçar primeiro um `grub-mkconfig` completo.

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

OS-PROBER MANUALMENTE

```
$ sudo xbps-install -y os-prober
# habilita o escaneamento — edita GRUB_DISABLE_OS_PROBER em /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 MANUALMENTE — EFIBOOTMGR

A aba EFI é uma interface para o `efibootmgr`. Tudo o que ela faz corresponde a um comando:

```
# lista todas as entradas de firmware, na ordem de inicialização atual
$ sudo efibootmgr -v
# reordena: coloca a entrada 0002 primeiro (= DEFAULT)
$ sudo efibootmgr -o 0002,0000,0001,0003
# inicialização única na entrada 0003 (= BOOT NEXT)
$ sudo efibootmgr -n 0003
# cancela um BootNext pendente
$ sudo efibootmgr -N
```

CARREGADORES ÓRFÃOS

Ocasionalmente um instalador escreve seu próprio bootloader na EFI System Partition sem registrar uma entrada de firmware para ele — o carregador existe no disco, mas o firmware não sabe que está lá. O painel do Odyssey escaneia a ESP diretamente e os lista separadamente como **"ON DISK, NOT IN FIRMWARE,"** cada um com um REGISTER de um clique. Manualmente, o equivalente é `efibootmgr -c -d /dev/sdX -p N -L "Nome" -l '\EFI\nome\bootx64.efi'` — note que entradas recém- criadas são colocadas **primeiro** na ordem de inicialização pelo próprio firmware, então reaplique DEFAULT na sua entrada habitual depois, se você não quiser que ela assuma.

ARQUIVOS QUE ESTE PAINEL TOCA

ARQUIVO	FUNÇÃO
`/etc/default/grub`	Linha de comando do kernel, tempo limite, GRUB_DEFAULT, GRUB_DISABLE_OS_PROBER
`/boot/grub/grub.cfg`	O menu de inicialização gerado em si — nunca edite manualmente; é sobrescrito pelo grub-mkconfig
Variáveis EFI (NVRAM)	Entradas de inicialização do firmware — não é um arquivo de forma alguma; lido e escrito inteiramente através do efibootmgr

CHAPTER 3

Kernel

Quatro abas — **SYSCTL**, **CPU GOVERNOR**, **HUGEPAGES**, **MODULES** — mais uma fileira de pre-definições de um clique acima das quatro, que se aplica a todas de uma vez.

3.1 Guia visual — as predefinições

No topo: a versão do seu kernel em execução, o escalonador ativo, e quatro botões de predefinição — **DESKTOP**, **GAMING**, **SERVER**, **BALANCED**.

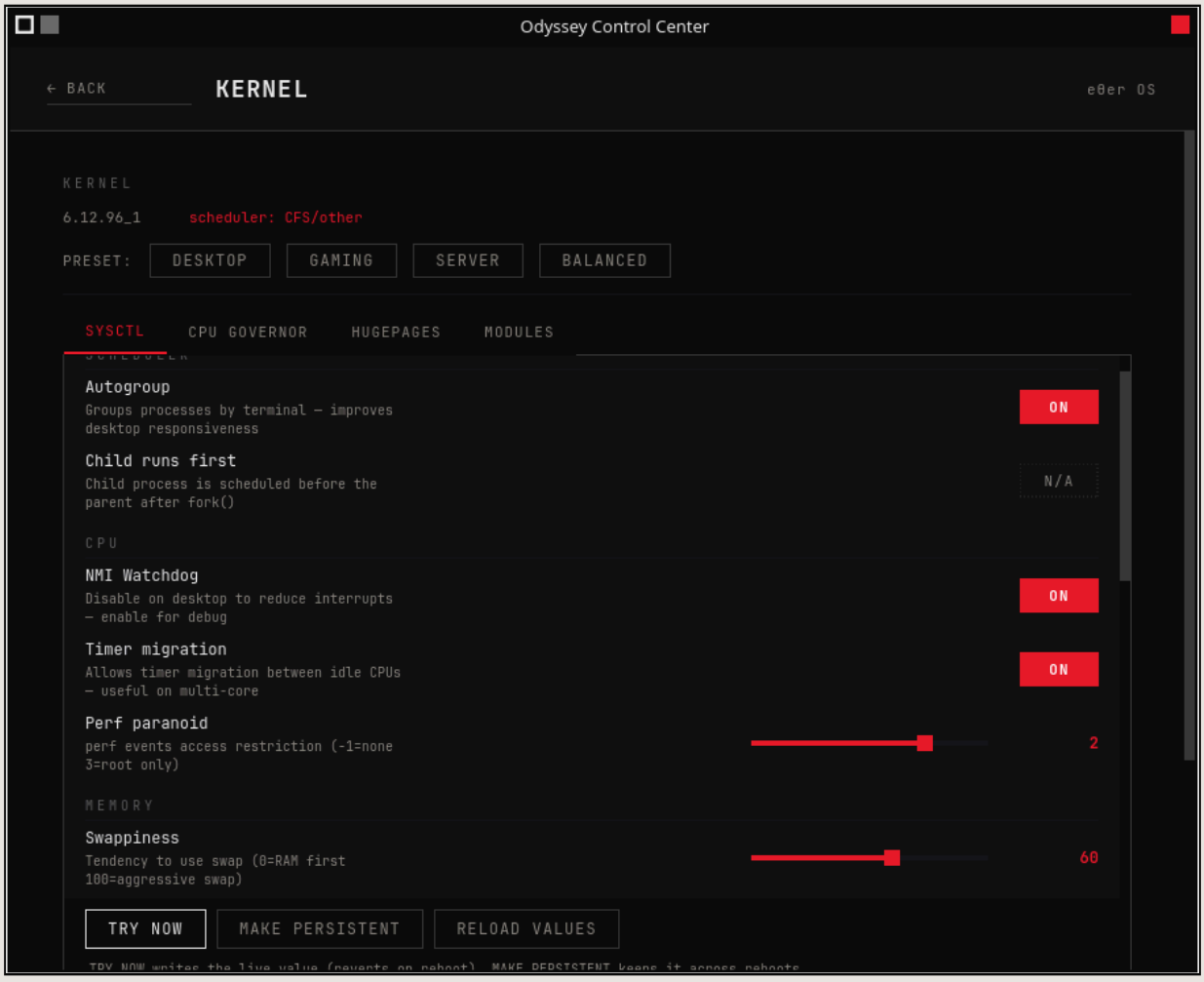

FIG · A aba SYSCTL recém-aberta: kernel 6.12.96_1, escalonador CFS/other, as quatro predefinições, e os grupos Scheduler/CPU/Memory abaixo.

PREDEFINIÇÃO	QUANDO VALE A PENA USAR
DESKTOP	Uma máquina do dia a dia — navegação, trabalho de escritório, multitarefa leve. Swappiness mais baixa, pressão moderada sobre o cache, migração de timer ativa para um uso energético suave em repouso. O padrão sensato para a maioria das pessoas.
GAMING	Swappiness empurrada quase a zero (mantenha tudo na RAM), migração de timer desativada (os núcleos ficam dedicados em vez de migrar trabalho), restrições de eventos perf relaxadas para ferramentas de profiling como MangoHud. Use quando os tempos de frame importarem mais que o consumo em repouso.
SERVER	Swappiness e tolerância à pressão de cache mais altas, NMI watchdog ativo, restrições mais rígidas sobre ponteiros do kernel/logs. Use em uma máquina sem monitor que prioriza estabilidade e segurança sobre responsividade interativa.
BALANCED	Um meio-termo entre Desktop e Gaming — para uma máquina que às vezes é uma estação de trabalho, às vezes uma sessão de jogo, e você não quer ficar alternando predefinições.

WHAT	Kernel → clique em DESKTOP, GAMING, SERVER ou BALANCED.
WHY	Ajustar manualmente uma dúzia de valores sysctl é pedir muito de quem só quer "boas configurações para jogar" — uma predefinição é exatamente isso, já decidido.
RESULT	Cada valor na aba SYSCTL pula para os números da predefinição, prontos e visíveis — você ainda precisa de TRY NOW ou MAKE PERSISTENT (abaixo) para realmente aplicá-los.
RISK	Nenhum — escolher uma predefinição só preenche o formulário.

3.2 Guia visual — SYSCTL, grupo por grupo

Role para baixo e cada parâmetro está agrupado sob um cabeçalho, cada um com uma descrição em linguagem simples diretamente no painel — você nunca precisa saber o que significa `vm.compaction_proactiveness` sem antes ler ali.

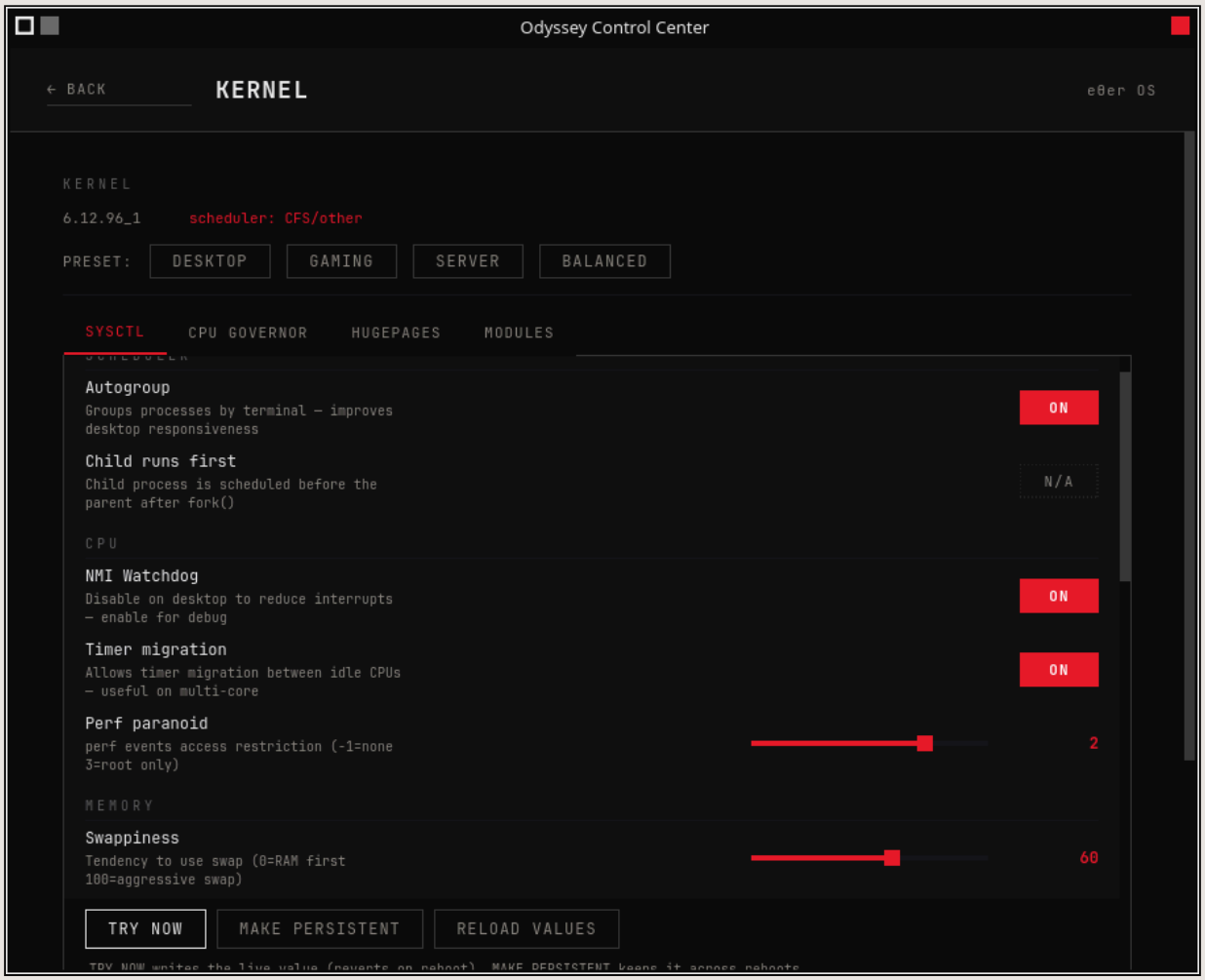

FIG · Grupos Scheduler e CPU: Autogroup, Child runs first, NMI Watchdog, Timer migration, Perf paranoid — cada um com seu próprio interruptor ou controle deslizante e uma linha de explicação.

GRUPO	O QUE CONTÉM
Scheduler	Como os processos são agrupados e priorizados
CPU	Watchdog, migração de timer, restrições de monitoramento de desempenho
Memory	Swappiness, pressão de cache, limiares de escrita de páginas sujas
Network	TCP Fast Open, tamanho da fila de backlog, limites dos buffers de socket
Security	Se ponteiros do kernel e mensagens de log ficam ocultos de usuários sem privilégios

Dois botões no final aplicam suas edições — e são deliberadamente diferentes:

WHAT	TRY NOW
WHY	Testar um valor com risco zero de que sobreviva a um reinício ruim.
RESULT	Escrito ao vivo via `sysctl` — ativo imediatamente, só para esta sessão.
RISK	Nenhum — reverte automaticamente no reinício.

WHAT	MAKE PERSISTENT
WHY	Você testou um valor (ou uma predefinição inteira) e quer que sobreviva aos reinícios.
RESULT	Escrito em `/etc/sysctl.d/99-odyssey.conf` **e** aplicado ao vivo no mesmo clique.
RISK	Baixo — é um arquivo de texto; apague a linha ou reaplique para desfazer.

RELOAD VALUES relê o que está atualmente ativo, descartando qualquer coisa digitada mas não aplicada.

3.3 Guia visual — CPU governor

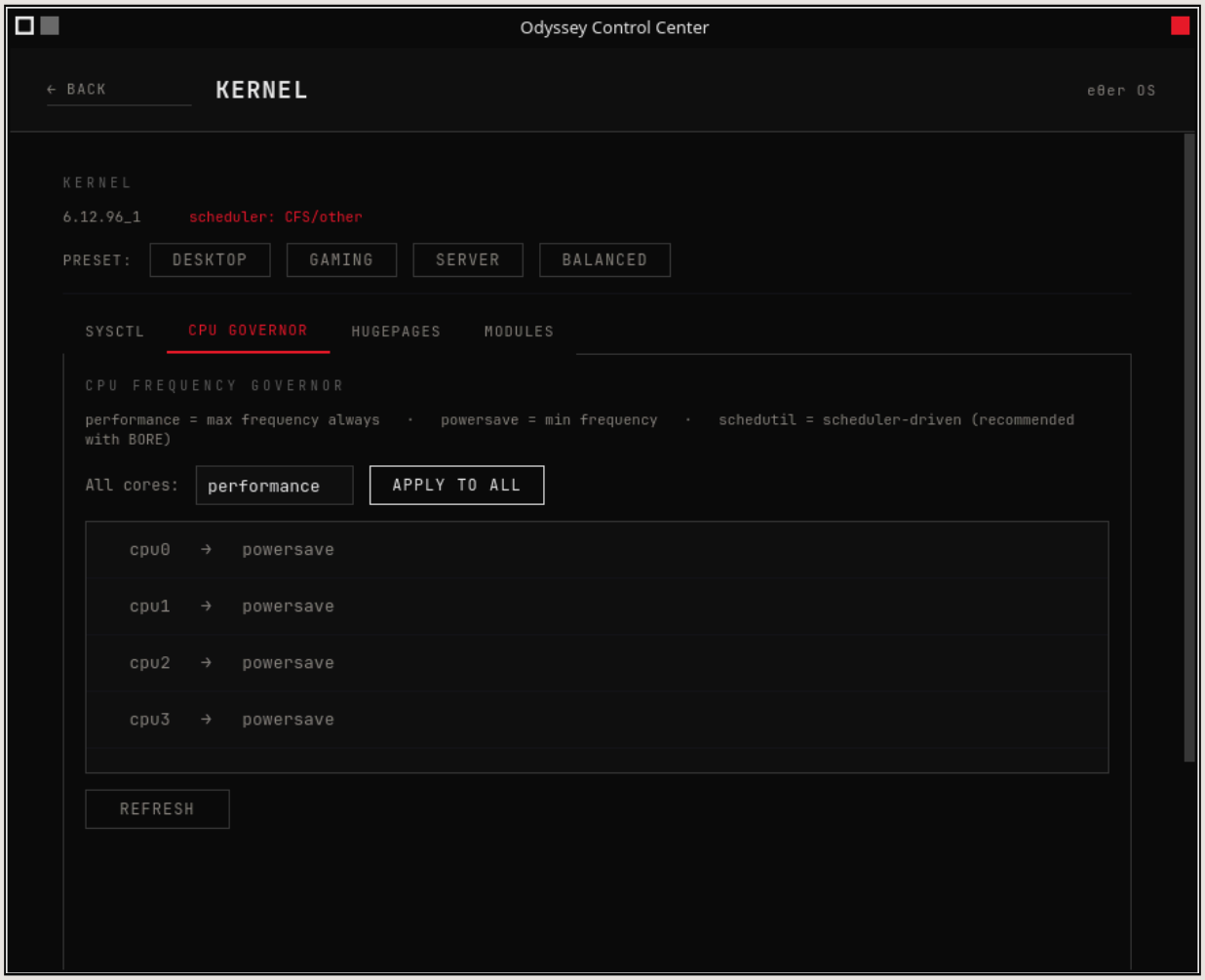

FIG · A aba CPU GOVERNOR: “All cores” definido como performance com APPLY TO ALL, e uma lista por núcleo abaixo ainda mostrando tudo em powersave — uma discrepância esperando ser corrigida.

O governor decide com que agressividade sua CPU muda a frequência do clock. Um menu suspenso aplica um governor a todos os núcleos de uma vez; a lista abaixo mostra o governor **ao vivo** por núcleo individual — é assim que você detecta um núcleo preso na configuração errada, exatamente como na captura acima onde “All cores” diz performance mas cada núcleo abaixo ainda mostra powersave, o que significa que APPLY TO ALL ainda não foi clicado.

GOVERNOR	COMPORTAMENTO
performance	Frequência máxima, sempre — mais responsivo, mais consumo
powersave	Frequência mínima, sempre — mais silencioso, menos responsivo
schedutil	Frequência guiada pela própria visão de carga do escalonador — recomendado com o escalonador BORE

Só os governors que sua CPU e kernel específicos realmente expõem aparecem no menu suspenso.

3.4 Guia visual — hugepages

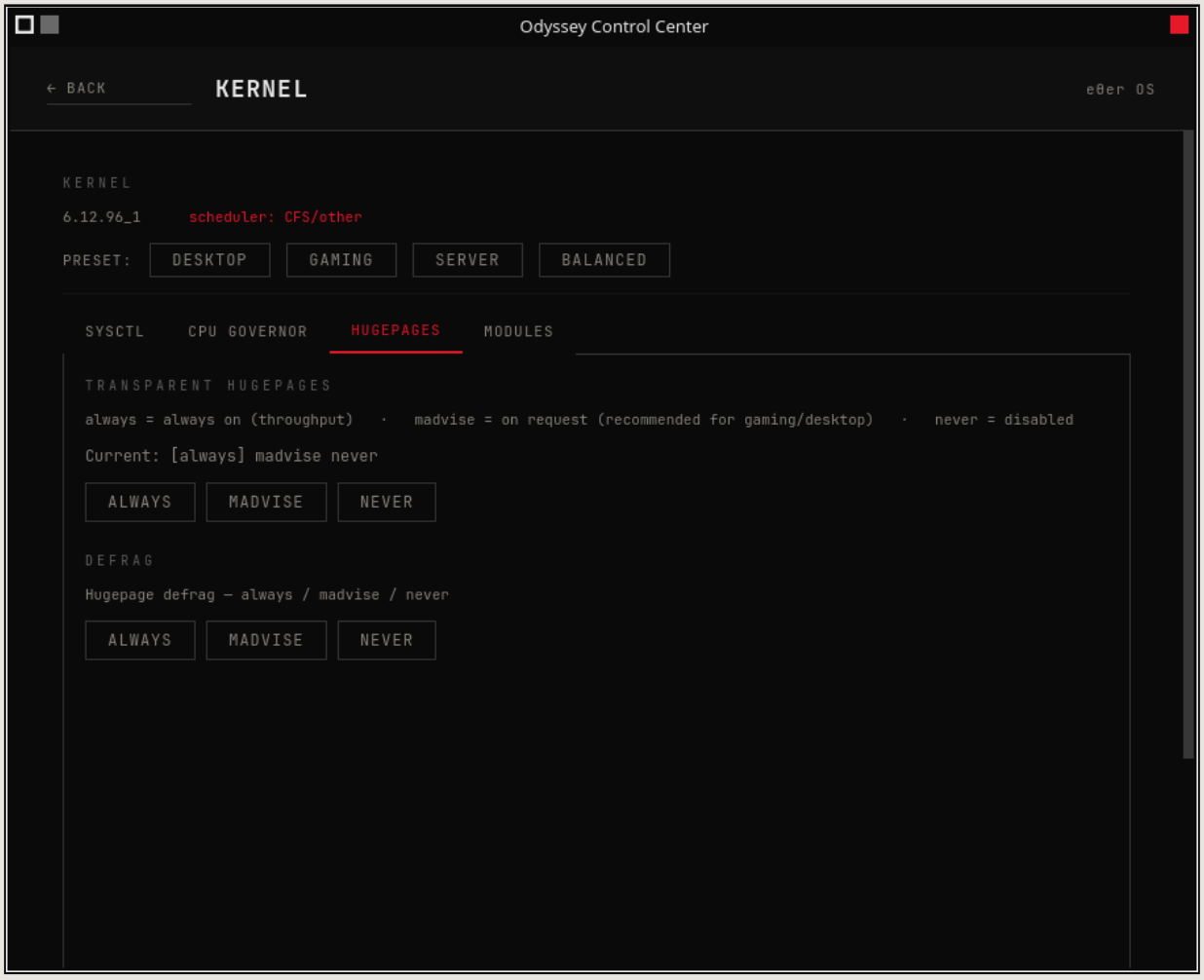

FIG · A aba HUGEPAGES: Transparent Hugepages mostrando “Current: [always] madvise never” com ALWAYS destacado, e a mesma fileira de três botões de novo para Defrag abaixo.

Duas configurações independentes, cada uma uma fileira de três botões aplicados no instante do clique — o valor ao vivo atual é sempre mostrado entre colchetes acima dos botões, como `[always] madvise never` na captura.

CONFIGURAÇÃO	O QUE CONTROLA
Transparent hugepages	ALWAYS = sempre ativo em todo lugar (throughput máximo); MADVISE = só quando um app pede explicitamente (recomendado para gaming/desktop); NEVER = completamente desativado
Defrag	Com que agressividade o kernel compacta a memória para **formar** hugepages sob demanda — mesmas três opções, independente da configuração acima

3.5 Guia visual — modules

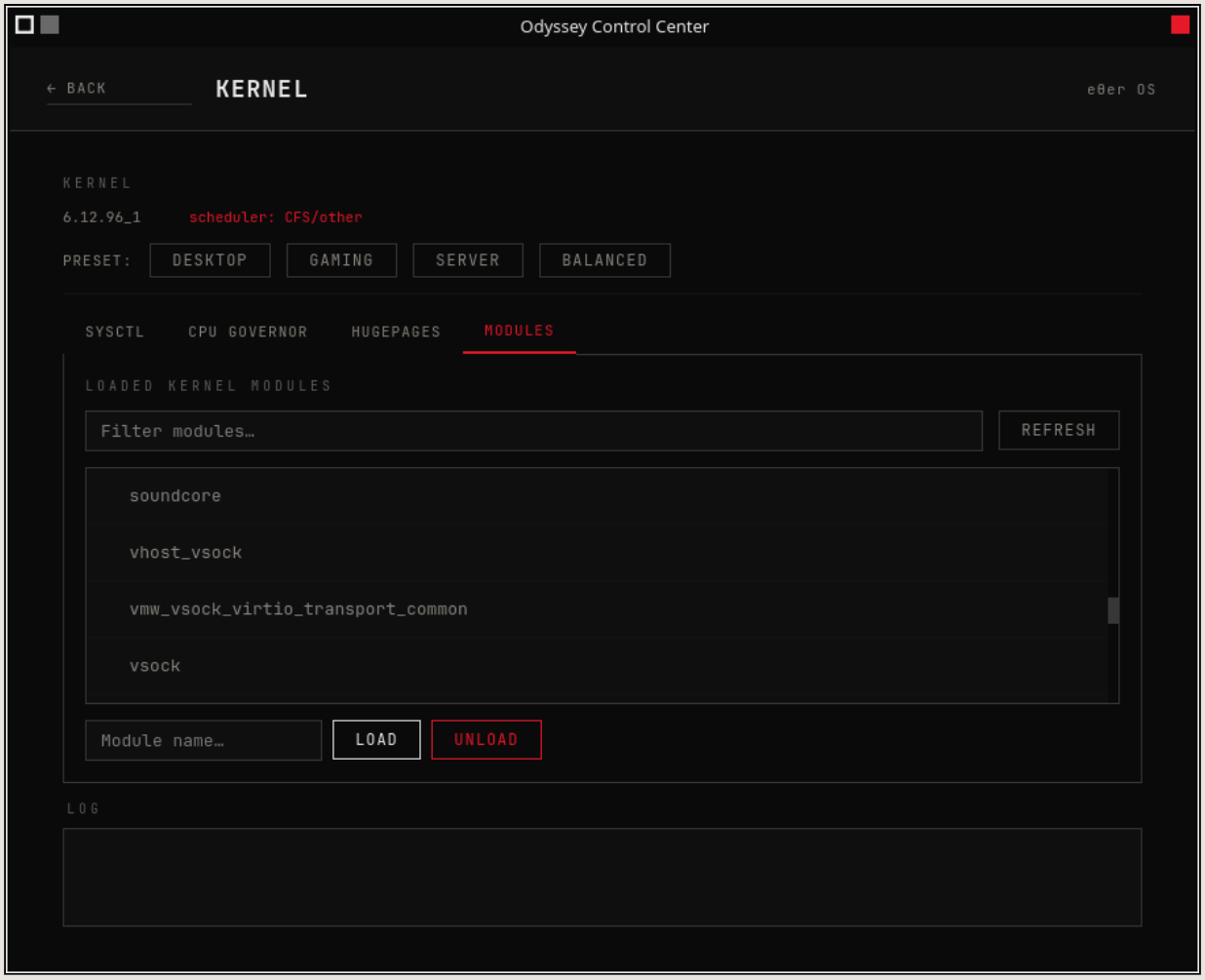

FIG · A aba MODULES: um campo de filtro, uma lista ao vivo (soundcore, vhost_vsock, vsock…), e uma fileira LOAD/UNLOAD no final para um módulo digitado pelo nome.

Uma lista ao vivo e filtrável de cada módulo do kernel atualmente carregado — digite no filtro para restringi-la, REFRESH para relê-la. Diferente das outras três abas, esta permite agir diretamente: digite um nome de módulo e clique em **LOAD** ou **UNLOAD**.

WHAT	Kernel → aba MODULES → digite um nome de módulo → LOAD ou UNLOAD.
WHY	Ocasionalmente você precisa de um driver ou módulo de função específico ativo (ou removido) sem reiniciar — testar hardware, liberar um dispositivo que outro módulo está retendo.
RESULT	O módulo é inserido ou removido do kernel em execução imediatamente.
RISK	Moderado especificamente para UNLOAD — remover um módulo do qual outra coisa depende pode desestabilizar o que o estava usando (um driver de vídeo, um sistema de arquivos). LOAD é geralmente seguro; o kernel se recusa de imediato a carregar um módulo incompatível em vez de aplicá-lo pela metade.

▸ ADVANCED

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

3.6 Advanced

AS CHAVES SYSCTL EXATAS

CHAVE	RÓTULO NO PAINEL
`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

VALORES DAS PREDEFINIÇÕES, 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
# as tabelas completas das predefinições são lidas diretamente de panel_kernel.py —
# abra o código-fonte do Control Center se quiser cada campo
```

MANUALMENTE, SEM O PAINEL

```
# ao vivo, temporário
$ sudo sysctl vm.swappiness=10
# persistente — o mesmo arquivo que o painel escreve
$ echo “vm.swappiness=10” | sudo tee -a /etc/sysctl.d/99-odyssey.conf
$ sudo sysctl -system
# CPU governor, por núcleo
$ 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
# módulos
$ sudo modprobe MODULE_NAME
$ sudo modprobe -r MODULE_NAME
```

POR QUE TRY NOW E MAKE PERSISTENT SÃO AÇÕES SEPARADAS

Escritas de `sysctl` são por natureza sempre só ao vivo — o kernel não tem persistência embutida. MAKE PERSISTENT é o Odyssey escolhendo um arquivo canônico único, `/etc/sysctl.d/99-odyssey.conf`, e fazendo a escrita ao vivo e a escrita em arquivo juntas para que as duas nunca se desalinhem. Editar esse arquivo diretamente e rodar `sysctl --system` depois é funcionalmente idêntico a clicar o botão.

CHAPTER 4

Memory & Swap

Uma única tela, três seções: um resumo somente leitura, swappiness, e ZRAM.

4.1 Guia visual

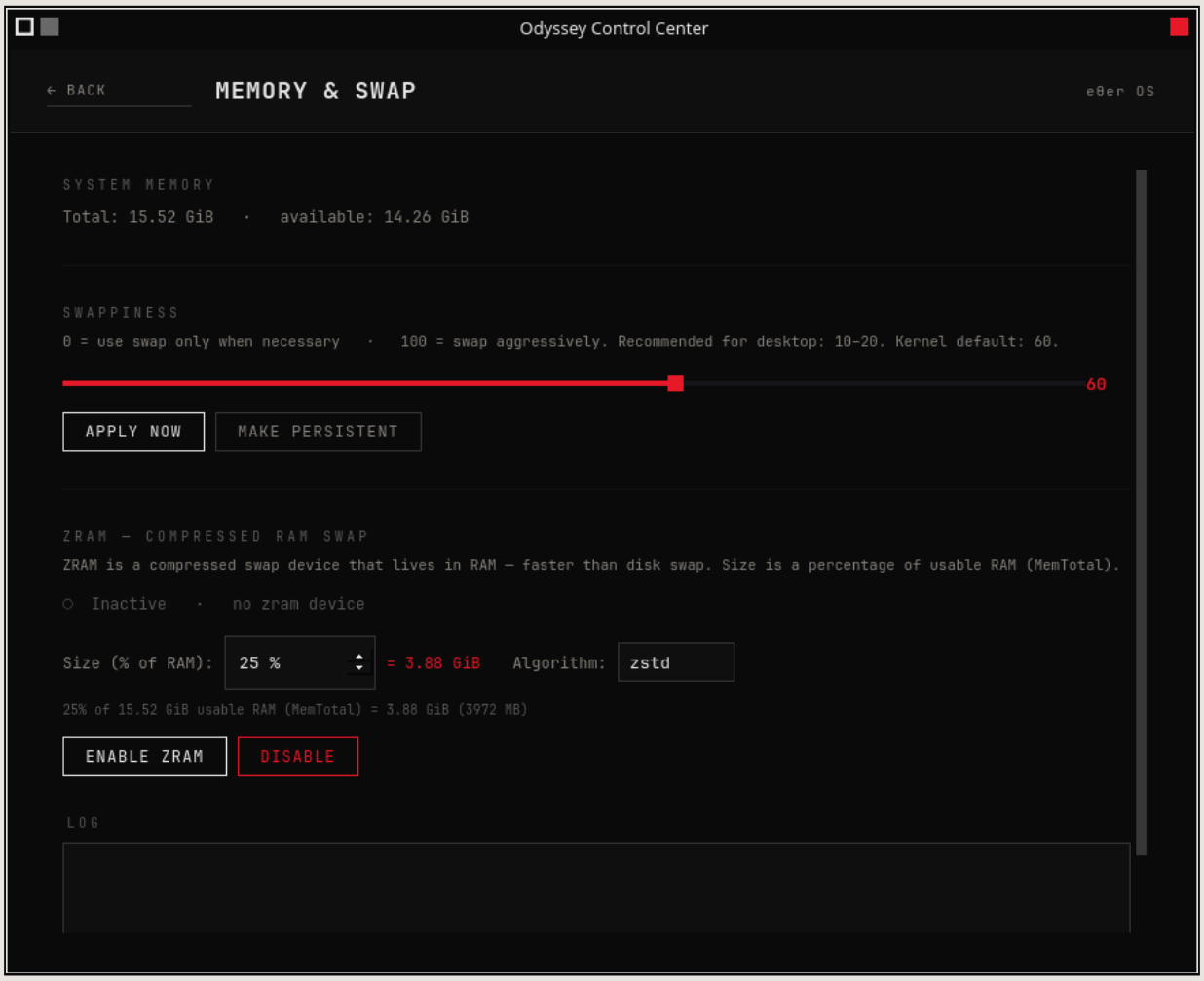

FIG · O painel completo: System memory (15,52 GiB no total, 14,26 GiB disponíveis), o controle deslizante Swappiness em 60 com APPLY NOW / MAKE PERSISTENT, e ZRAM abaixo mostrando "Inactive · no zram device" com tamanho, algoritmo e ENABLE/DISABLE.

SYSTEM MEMORY

Só um resumo no topo — RAM total e o que está disponível no momento. Nenhum controle; é o contexto para as duas seções abaixo.

SWAPPINESS

Um único controle deslizante de 0 a 100. 0 significa “use swap só quando realmente não houver outra escolha”; 100 significa “use swap agressivamente, bem antes da RAM ficar cheia.” O padrão do kernel é 60 (o que a captura mostra); a própria sugestão do painel indica 10–20 para um desktop — o mesmo número que a predefinição DESKTOP do painel Kernel define.

WHAT	Arraste o controle deslizante → APPLY NOW para testar, MAKE PERSISTENT quando estiver satisfeito.
WHY	Um valor mais baixo mantém o que você usa ativamente na RAM rápida em vez de movê-lo para o swap mais lento — geralmente o que você quer com memória suficiente instalada.
RESULT	APPLY NOW muda ao vivo, imediatamente, só para esta sessão. MAKE PERSISTENT o escreve em um arquivo de configuração para que sobreviva também ao reinício.
RISK	Muito baixo. Um valor extremamente baixo em uma máquina realmente escassa de RAM pode torná-la mais propensa a ficar sem memória abruptamente sob carga pesada em vez de fazer swap gradualmente — se isso acontecer, aumente-o de novo.

ZRAM — SWAP COMPRIMIDO EM RAM

ZRAM é um dispositivo de swap que vive **dentro** da própria RAM, comprimido — notavelmente mais rápido que o swap em disco porque nenhum disco está envolvido. O tamanho é uma porcentagem da sua RAM utilizável total, não um número fixo, então a configuração se adapta com sensatez a máquinas diferentes; uma leitura ao vivo ao lado do campo de tamanho converte a porcentagem para GiB para você (na captura: 25% = 3,88 GiB em um sistema de 15,52 GiB).

WHAT	Defina a porcentagem de tamanho e escolha um algoritmo (zstd, lz4, lzo, ou deflate) → ENABLE ZRAM.
WHY	O swap comprimido em RAM absorve a pressão de memória com um impacto no desempenho muito menor que o swap em disco — útil em qualquer máquina, especialmente uma com RAM limitada.
RESULT	Um dispositivo `zram0` é criado e ativado no tamanho e algoritmo escolhidos; a linha de status muda de “Inactive” para ativo.
RISK	Baixo. zstd (o padrão) equilibra bem a taxa de compressão e o custo de CPU para a maioria das máquinas. DISABLE o desativa de forma limpa a qualquer momento.

▸ ADVANCED

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

4.2 Advanced

O QUE “DEPENDE DO HARDWARE” REALMENTE SIGNIFICA AQUI

Ambas as configurações interagem com seu hardware específico, não só com “mais RAM = melhor”:

- **SSD NVMe rápido**: uma swappiness na faixa de 10–20 custa muito pouco mesmo que faça swap — o disco é rápido o suficiente para que seja quase imperceptível. ZRAM ainda ganha em latência e desgaste do SSD, mas a diferença é menor que com um disco mecânico.

- **HDD mecânico ou eMMC como swap**: mantenha a swappiness baixa (0–10) — o swap em disco em mídias lentas causa a travada que se chama “thrashing.” ZRAM é um ganho muito maior justamente aqui.
- **Margem de CPU**: ZRAM troca I/O de disco por ciclos de CPU gastos comprimindo e descomprimindo. Em uma máquina limitada pela CPU (hardware mais antigo, ou um já no limite durante o gaming), `lz4` comprime mais rápido que `zstd` com uma taxa pior — uma troca razoável se o tempo de CPU for o recurso mais apertado. Em qualquer coisa moderna, a melhor taxa do `zstd` vale seu custo um pouco maior.
- **Margem de RAM**: uma porcentagem grande de ZRAM (como 50%) em uma máquina com só 8 GiB no total deixa menos RAM real para os aplicativos, já que o próprio ZRAM consome memória real para armazenar as páginas comprimidas. Com 16 GiB ou mais, 25% é um padrão confortável com folga.

MANUALMENTE

```
# swappiness, ao vivo
$ sudo sysctl vm.swappiness=15
# swappiness, persistente — o mesmo arquivo que o painel escreve
$ echo “vm.swappiness=15” | sudo tee /etc/sysctl.d/99-odyssey.conf
# zram, configuração manual
$ 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
# verifica que está ativo
$ swapon -show
```

POR QUE O PAINEL REINICIA O ZRAM0 EM VEZ DE REDIMENSIONÁ-LO AO VIVO

O tamanho de um dispositivo ZRAM não pode ser alterado enquanto está ativo e em uso como swap — o kernel exige que seja primeiro desativado e reiniciado. O botão ENABLE ZRAM do painel executa essa sequência automaticamente (swapoff, reset, reconfiguração, swapon), motivo pelo qual mudar o tamanho ou o algoritmo em um dispositivo já ativo ainda aparece como uma única ação limpa em vez de um erro.

CHAPTER 5

Firewall

O painel Firewall tem seu próprio capítulo dedicado, em plena profundidade, no **Security and Hardening Guide** — o conjunto de regras explicado linha por linha, as 21 predefinições com seus riscos individuais, o editor de configuração bruta, e os equivalentes de linha de comando. Este capítulo é a versão guiada visualmente.

5.1 Guia visual

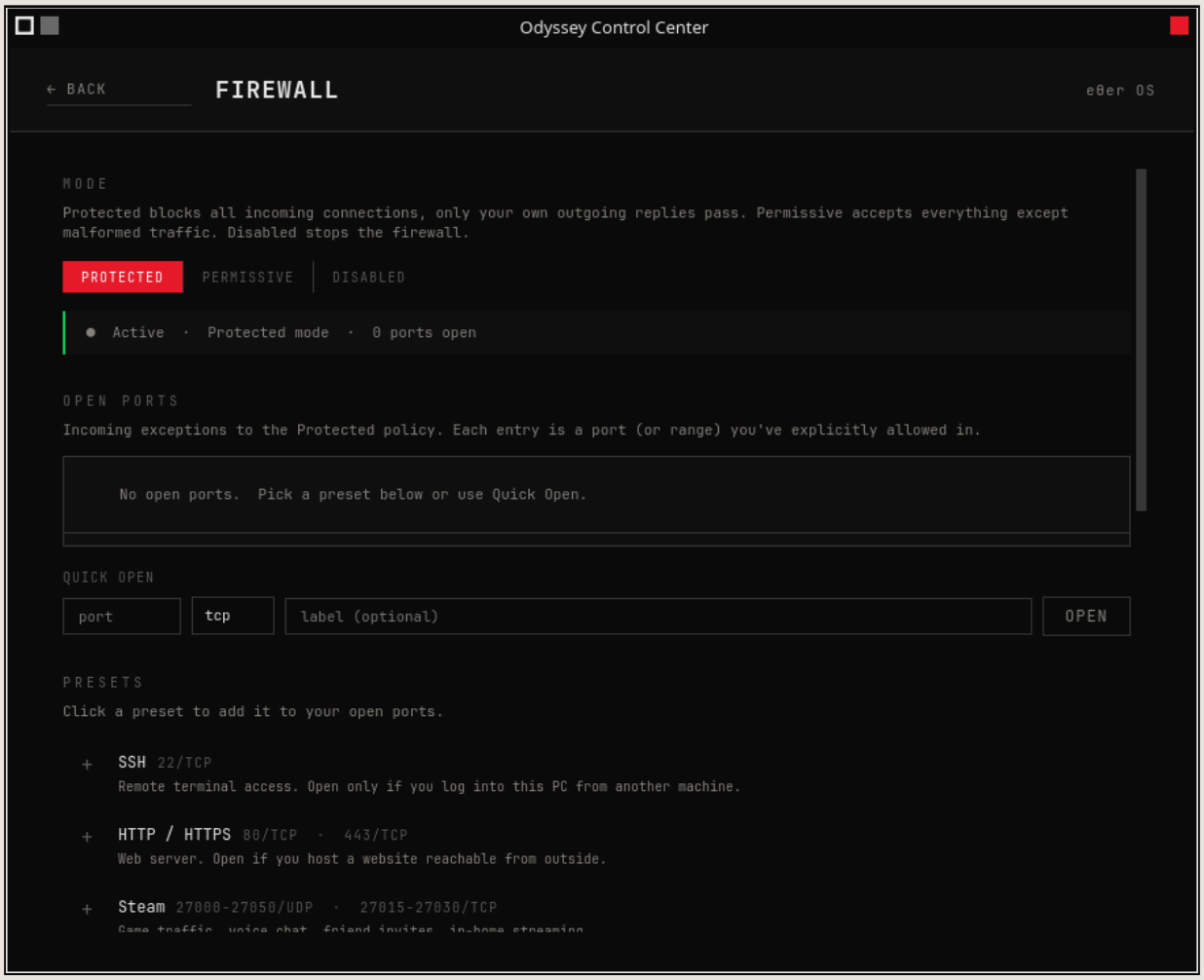

FIG · Modo PROTECTED ativo, “0 ports open,” os campos Quick Open, e os blocos de predefinição começando com SSH, HTTP/HTTPS e Steam.

MODE

Três botões no topo, e o painel explica cada um em uma única linha logo acima deles:

MODO	O QUE FAZ
PROTECTED	O padrão. Bloqueia todas as conexões de entrada; só suas próprias respostas de saída passam.
PERMISSIVE	Aceita tudo exceto tráfego malformado — uma configuração de diagnóstico para "é o firewall que está quebrando isso?", não um lugar para ficar.
DISABLED	Para o firewall completamente.

A linha de status abaixo confirma o que está realmente ativo — na captura, **"Active · Protected mode · 0 ports open."**

OPEN PORTS E PREDEFINIÇÕES

OPEN PORTS lista suas exceções explícitas — vazio por padrão, como mostrado. **QUICK OPEN** pega porta, protocolo e rótulo opcional diretamente, para qualquer coisa que não esteja na lista de predefinições. **PRESETS** abaixo oferece oito serviços comuns a um clique cada, cada um com uma linha explicando quando você realmente precisaria dele (SSH: "acesso remoto ao terminal, abra só se você acessar este PC de outra máquina"), mais treze outros sob um "MORE PRESETS" expansível.

WHAT	Firewall → clique em um bloco de predefinição, ou preencha QUICK OPEN → SAVE & APPLY.
WHY	Você está rodando algo — um servidor de jogo, um servidor de mídia, um servidor de desenvolvimento — que outra máquina precisa alcançar.
RESULT	Essa porta abre; tudo mais permanece fechado pela política PROTECTED.
RISK	Depende inteiramente da predefinição — consulte as notas de risco por predefinição no Security and Hardening Guide antes de abrir qualquer coisa além de SSH ou HTTP/HTTPS em uma máquina alcançável pela internet.

Nada tem efeito até você apertar **SAVE & APPLY** — as mudanças ficam primeiro em espera, para que você possa revisar exatamente o que está prestes a abrir antes de confirmar.

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

5.2 Advanced

A referência técnica completa — o conjunto de regras padrão do `nftables.conf` explicado regra por regra, cada uma das 21 predefinições com sua exposição específica e mitigação, receitas para limitar o acesso só à LAN, os equivalentes de linha de comando `nft`, e a resolução de uma sessão SSH trancada por fora — vive no **Capítulo 2 do Security and Hardening Guide** (`/usr/share/doc/odyssey/` no seu sistema instalado). Este manual não o duplica deliberadamente; considere aquele capítulo a continuação desta seção.

```
# o comando que vale a pena conhecer agora mesmo:
$ sudo nft list ruleset
# mostra exatamente o que está carregado no kernel, independente do painel
```

CHAPTER 6

AppArmor

Cinco abas: **STATUS**, **PROFILES**, **COMMUNITY**, **VIOLATIONS**, **EDITOR**. Mesma referência cruzada de Firewall: a referência aprofundada — o que é realmente o controle de acesso obrigatório, o fluxo completo de confinamento, um exemplo prático — vive no Security and Hardening Guide. Este é o guia visual.

6.1 Guia visual — STATUS

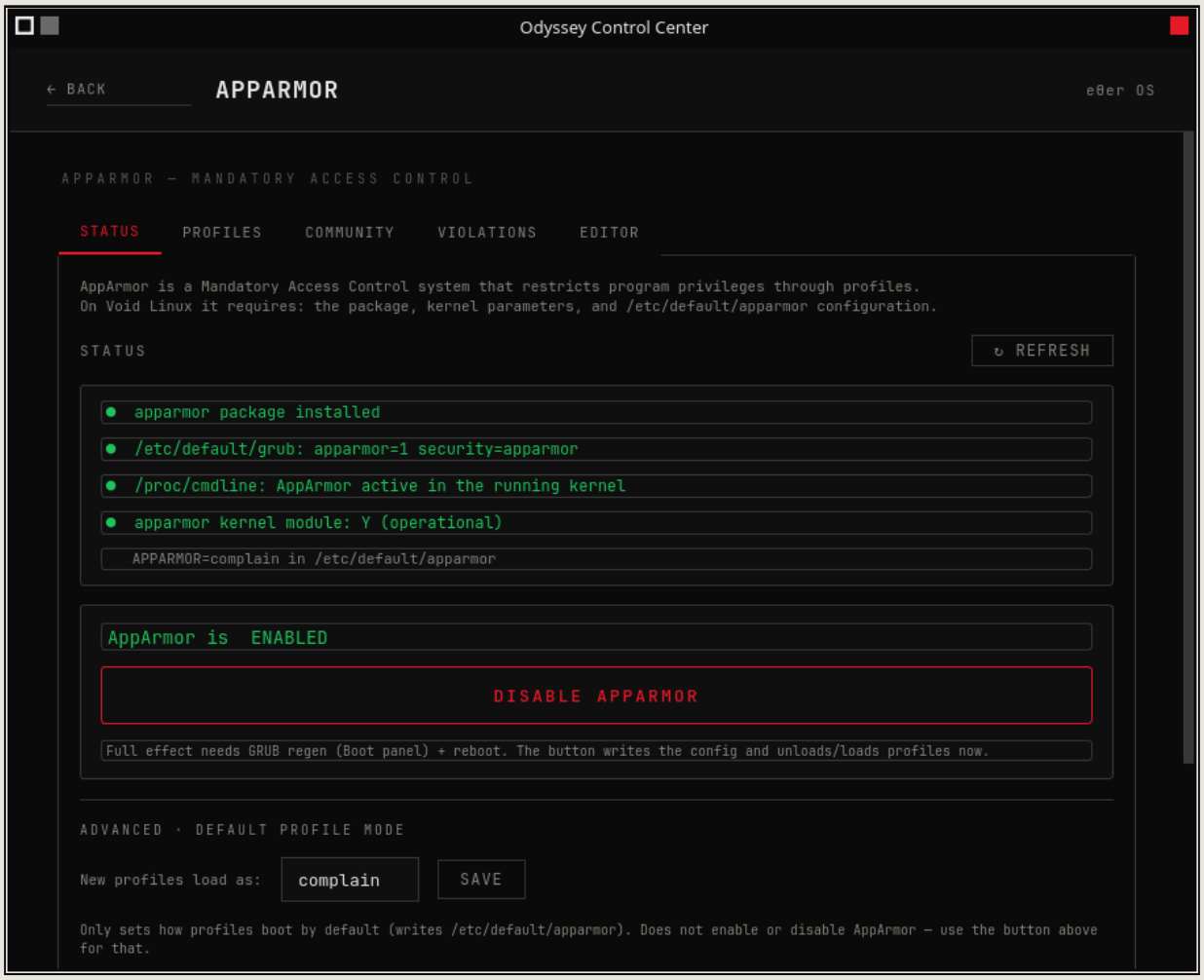

FIG · Aba STATUS: quatro marcas verdes (pacote instalado, parâmetros do GRUB, kernel em execução, módulo do kernel), "AppArmor is ENABLED" com um botão vermelho DISABLE APPARMOR, e o modo de perfil padrão definido como complain.

Quatro verificações confirmam que o framework realmente funciona, de cima a baixo: o pacote `apparmor` está instalado, os parâmetros do kernel no GRUB estão definidos, o kernel **em execução** realmente iniciou com eles (essa é a que pode ficar atrasada se você não reiniciou depois de habilitá-los),

e o próprio módulo do kernel está operacional. Abaixo: um botão grande de habilitar/desabilitar, e **ADVANCED · DEFAULT PROFILE MODE** — se os perfis recém-carregados iniciam em complain ou enforce.

WHAT	AppArmor → STATUS → confirme que as quatro verificações estão verdes.
WHY	Se alguma verificação estiver vermelha ou amarela, mais nada neste painel pode funcionar corretamente — é sempre a primeira coisa a olhar quando algo com AppArmor parece estar errado.
RESULT	Certeza de que o framework está genuinamente ativo, não só instalado.
RISK	Nenhum — verificação somente leitura.

6.2 Guia visual — PROFILES

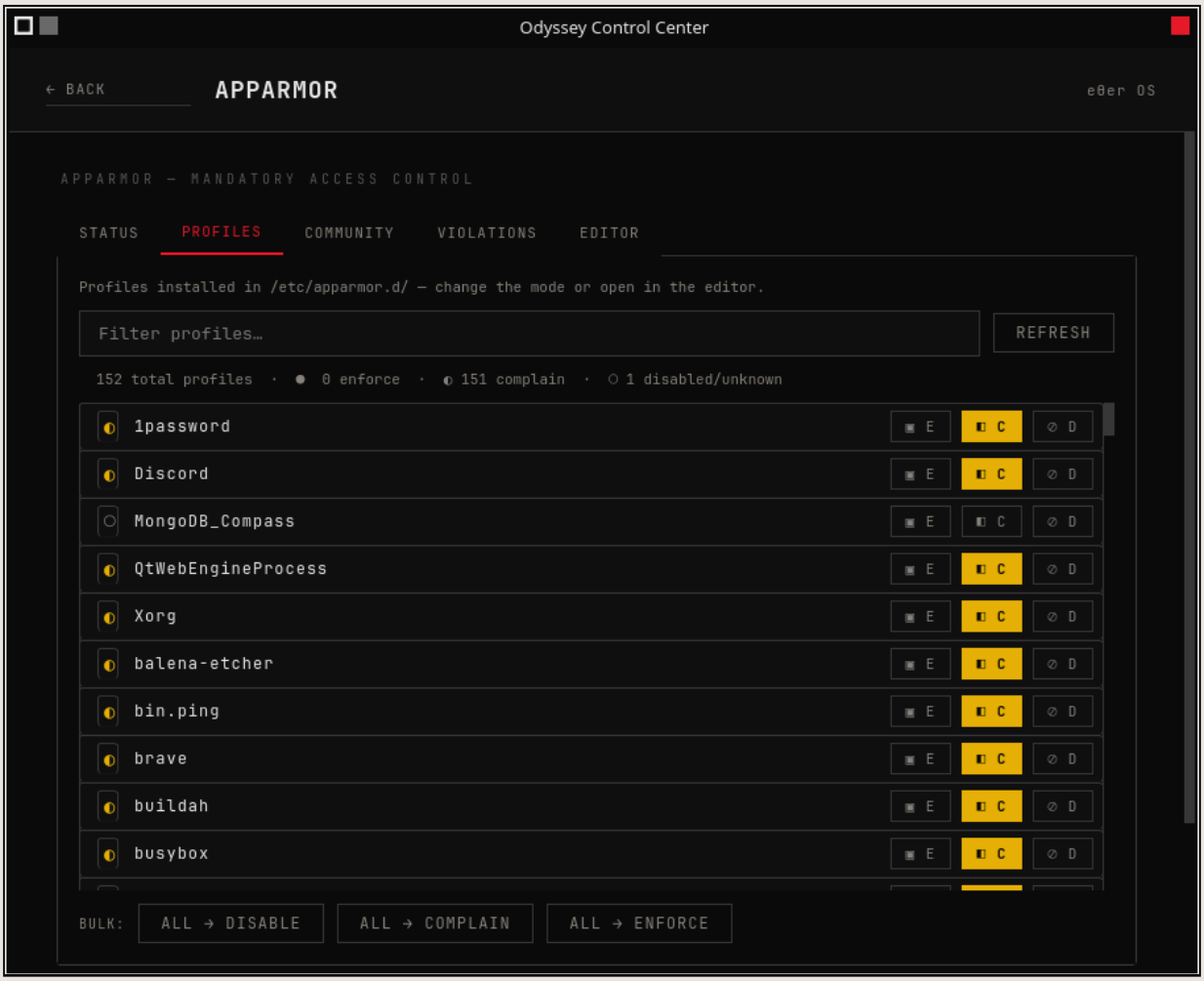

FIG · Aba PROFILES: 152 perfis no total, 0 enforce, 151 complain, 1 disabled/unknown — uma lista filtrável com 1password, Discord, MongoDB_Compass e mais, cada um com botões E / C / D, e no final ALL→DISABLE / ALL→COMPLAIN / ALL→ENFORCE.

Cada perfil carregado, uma linha cada, com um campo de filtro e uma contagem no topo (na captura: 152 perfis, todos menos um em modo complain — o padrão do Odyssey). Três botões por linha:

BOTÃO	EFEITO
E	Muda esse perfil para enforce — violações são bloqueadas, não só registradas
C	Muda para complain — registra violações, não bloqueia nada
D	Desativa o perfil completamente — o programa roda sem confinamento

Os três botões **BULK** no final aplicam a mesma mudança a cada perfil da lista de uma vez — útil logo depois de instalar um lote de COMMUNITY, mas trate ALL→ENFORCE com cautela especial; veja o bloco de ação na próxima seção.

6.3 Guia visual — COMMUNITY

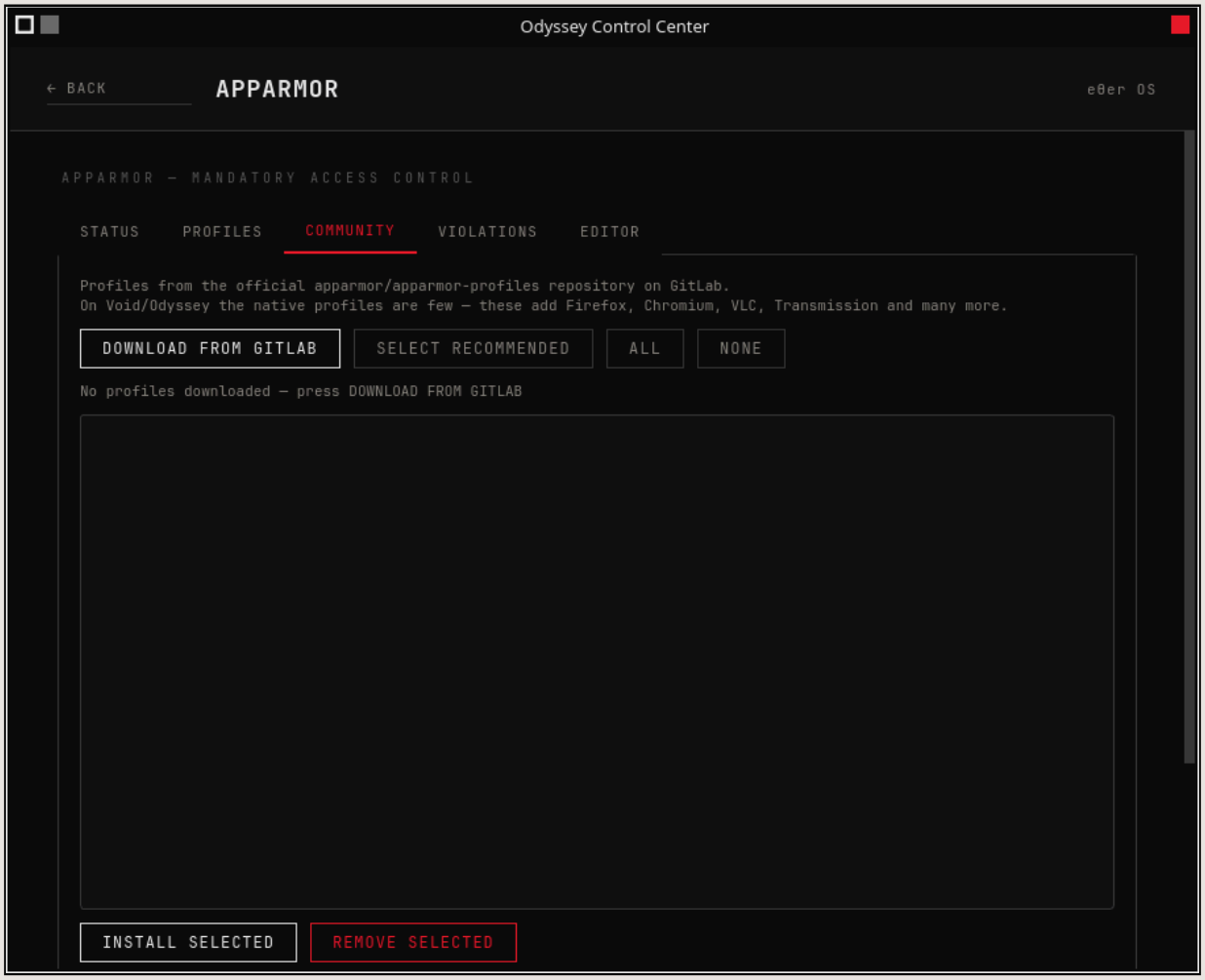

FIG · Aba COMMUNITY: "No profiles downloaded — press DOWNLOAD FROM GITLAB," com os botões DOWNLOAD FROM GITLAB, SELECT RECOMMENDED, ALL e NONE, e INSTALL SELECTED / REMOVE SELECTED no final.

O AppArmor de fábrica confina principalmente componentes do sistema — os aplicativos de desktop com os quais você realmente se preocupa (navegadores, apps de chat, players de mídia) precisam de perfis da comunidade. Esta aba os obtém diretamente do repositório oficial `apparmor/apparmor-profiles` no GitLab.

1 **DOWNLOAD FROM GITLAB**

Obtém a lista atual de perfis disponíveis — nada é instalado ainda, isso só popula a lista.

2 SELECT RECOMMENDED (ou ALL / NONE)

Marca um conjunto padrão sensato para desktop — navegadores da família Firefox, Chromium, VLC, Transmission e similares.

3 INSTALL SELECTED

Instala os perfis marcados — eles carregam em modo complain, como tudo mais, prontos para o ciclo observar-depois-aplicar abaixo.

WHAT	AppArmor → COMMUNITY → DOWNLOAD FROM GITLAB → SELECT RECOMMENDED (ou escolha individualmente) → INSTALL SELECTED.
WHY	Os aplicativos mais expostos a entradas não confiáveis — um navegador que renderiza uma página maliciosa, um player de mídia que abre um arquivo manipulado — são exatamente os que um perfil protege melhor.
RESULT	Novos perfis aparecem em PROFILES, em modo complain, registrando o que teriam bloqueado sem ainda bloquear nada.
RISK	Nenhum na instalação — nada é aplicado até você decidir em PROFILES.

6.4 Guia visual — VIOLATIONS

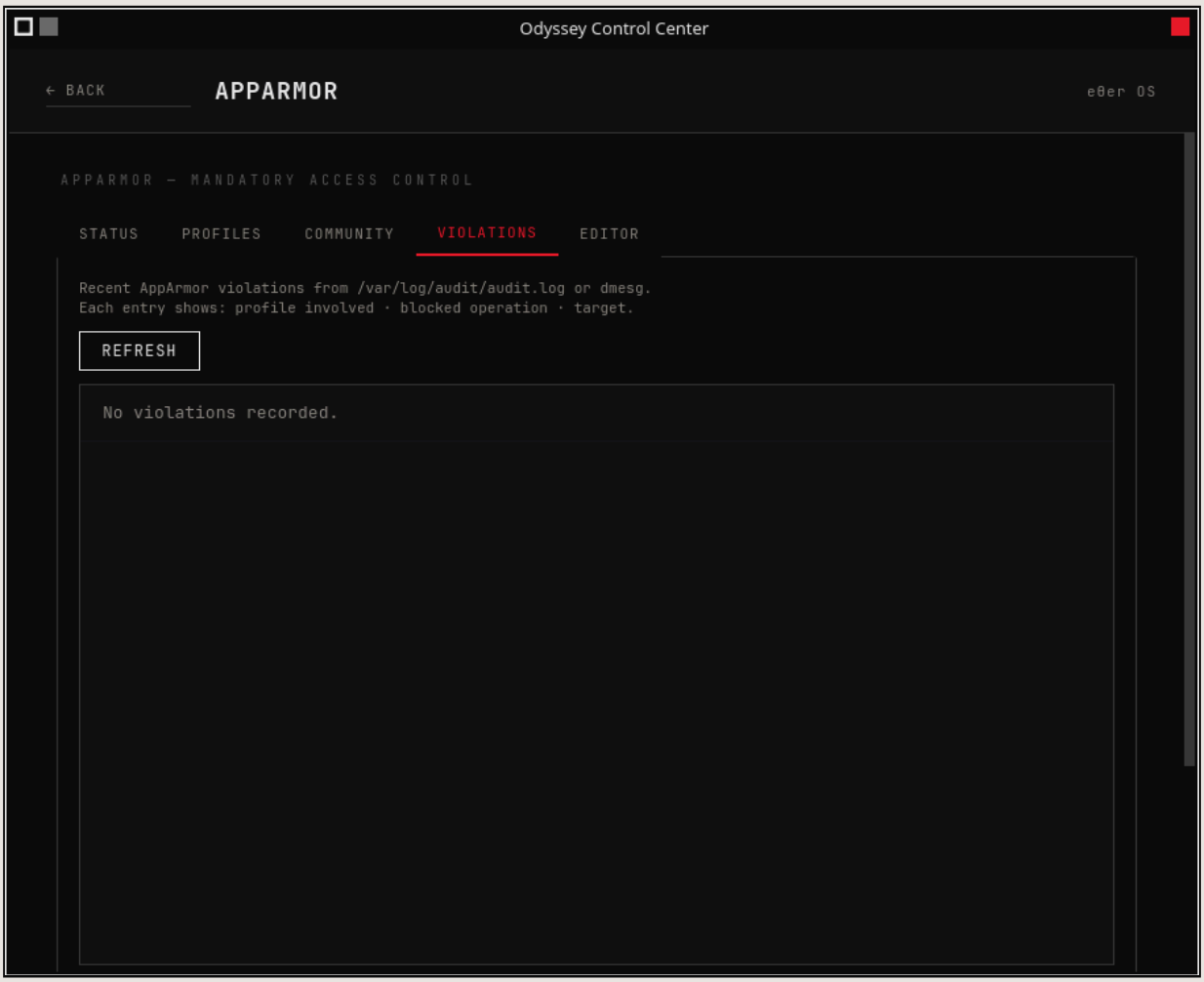

FIG · Aba VIOLATIONS: “No violations recorded,” com um botão REFRESH e uma nota de que as entradas vêm de /var/log/audit/audit.log ou dmesg.

Esta aba lê o log de auditoria do AppArmor do sistema e mostra o que um perfil **teria** bloqueado (em modo complain) ou **bloqueou** (em modo enforce) — cada entrada indica o perfil, a operação bloqueada, e o arquivo ou recurso de destino.

> **UMA LISTA VAZIA É A LEITURA ESPERADA EM UM SISTEMA SAUDÁVEL**
>
> A captura mostra exatamente o que você tipicamente verá: nenhuma violação. Isso não é um bug nem um sinal de que o painel não está funcionando — significa que todo programa perfilado permaneceu dentro dos limites permitidos desde a última verificação do log. As violações aparecem aqui no momento em que algo perfilado tenta fazer algo que seu perfil não permite; vá usar o programa em questão, volte, e faça REFRESH.

Uma vez que entradas apareçam, dar um clique duplo em uma delas é o caminho rápido para EDITOR com aquele perfil exato já aberto — veja abaixo.

6.5 Guia visual — EDITOR

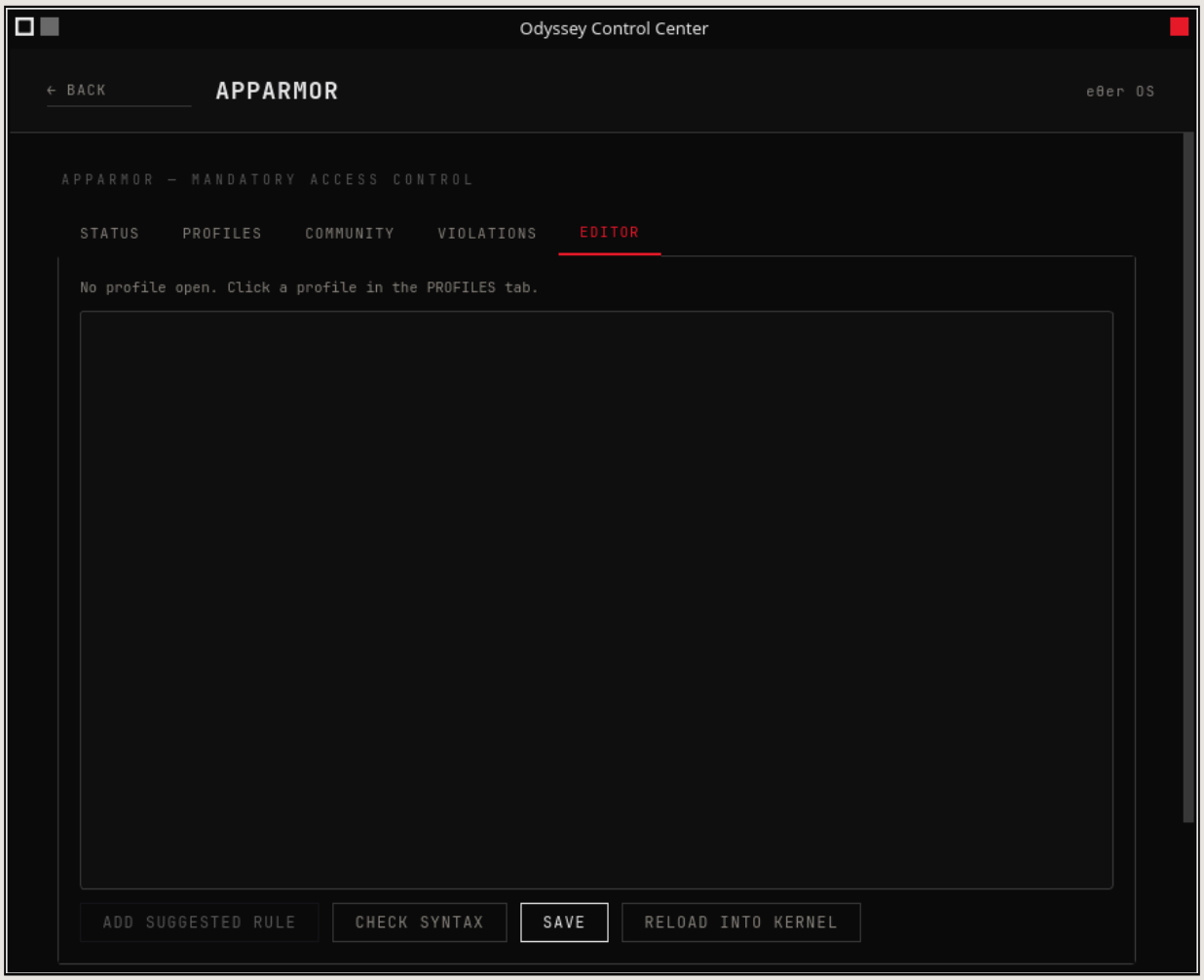

FIG · Aba EDITOR: “No profile open. Click a profile in the PROFILES tab,” uma área de texto vazia, e quatro botões — ADD SUGGESTED RULE, CHECK SYNTAX, SAVE, RELOAD INTO KERNEL — todos exceto SAVE desativados.

O editor abre deliberadamente **vazio**, como mostrado — ele só edita um perfil por vez, e você escolhe qual a partir de PROFILES (clique em uma linha) ou de VIOLATIONS (clique duplo em uma entrada). Isso evita o erro fácil de editar o arquivo errado por acidente.

BOTÃO	O QUE FAZ
ADD SUGGESTED RULE	Ativo só se você chegou de uma entrada VIOLATIONS — insere a linha de regra exata que teria permitido a ação registrada
CHECK SYNTAX	Valida o perfil sem carregá-lo — pega um erro de digitação antes que ele importe
SAVE	Escreve suas edições no arquivo do perfil no disco
RELOAD INTO KERNEL	Torna uma edição salva efetiva imediatamente, sem esperar pela próxima inicialização ou reinício do serviço

WHAT	PROFILES → clique no nome de um perfil → EDITOR o abre → faça uma edição → CHECK SYNTAX → SAVE → RELOAD INTO KERNEL.
WHY	Esse é o ciclo completo de confinamento em uma linha: observe em modo complain (VIOLATIONS), corrija o que realmente falta (EDITOR), só então mude esse perfil para enforce (PROFILES).
RESULT	O perfil se comporta exatamente como editado, ao vivo, sem reiniciar.
RISK	Editar um perfil incorretamente pode fazer algo legítimo ser bloqueado depois sob enforce — CHECK SYNTAX pega regras malformadas antes de SAVE, mas não erros lógicos. Se um perfil que você editou começar a causar problemas sob enforce, volte-o para complain a partir de PROFILES e verifique VIOLATIONS de novo.

▸ ADVANCED

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

6.6 Advanced

EQUIVALENTES DE LINHA DE COMANDO

```
$ sudo aa-status
# as mesmas contagens do cabeçalho da aba PROFILES, do 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
# equivalente a RELOAD INTO KERNEL depois de uma edição manual
$ sudo apparmor_parser -r /etc/apparmor.d/PROFILE_FILE
```

LENDO O LOG DE AUDITORIA BRUTO

```
$ sudo grep apparmor /var/log/audit/audit.log | tail -20
# ou, se nenhum daemon de auditoria dedicado estiver rodando:
$ sudo dmesg | grep -i apparmor
```

Campos úteis a conhecer ao ler uma linha manualmente: `apparmor="DENIED"` (enforce) ou `apparmor="ALLOWED"` (complain, só registrado), `operation=` (o que foi tentado), `profile=` (qual perfil decidiu — corresponde ao que EDITOR abre), `name=` (o caminho ou o recurso), `comm=` (o processo).

POR QUE ALL→ENFORCE EM MASSA É ARRISCADO

Elevar um perfil individualmente depois que seu log VIOLATIONS ficou em silêncio por um tempo é o padrão seguro — você sabe, perfil por perfil, que o uso normal não o dispara. ALL→ENFORCE pula esse período de observação para todos os perfis de uma vez, incluindo os que você talvez tenha acabado de instalar de COMMUNITY e nunca usado de fato. Um perfil aplicado antes de ter sido vivido pode bloquear uma função legítima no exato momento em que você a usa pela primeira vez. Prefira o ciclo um a um; reserve as ações em massa para ALL→COMPLAIN (sempre seguro — só afrouxa) ou ALL→DISABLE ao fazer diagnóstico.

ARQUIVOS QUE ESTE PAINEL TOCA

CAMINHO	FUNÇÃO
`/etc/apparmor.d/`	Um arquivo de perfil para cada programa confinado — o que PROFILES lista e EDITOR abre
`/etc/default/apparmor`	Modo padrão na inicialização (`APPARMOR=complain`) e o estado enable/disable
`/etc/default/grub`	Os parâmetros do kernel `apparmor=1 security=apparmor` que STATUS verifica

CHAPTER 7

Privacy

Duas ferramentas independentes, cada uma com seu próprio fluxo em três etapas install → enable → running, e uma ponte entre as duas assim que ambas estiverem em execução.

7.1 Guia visual — SearXNG, passo a passo

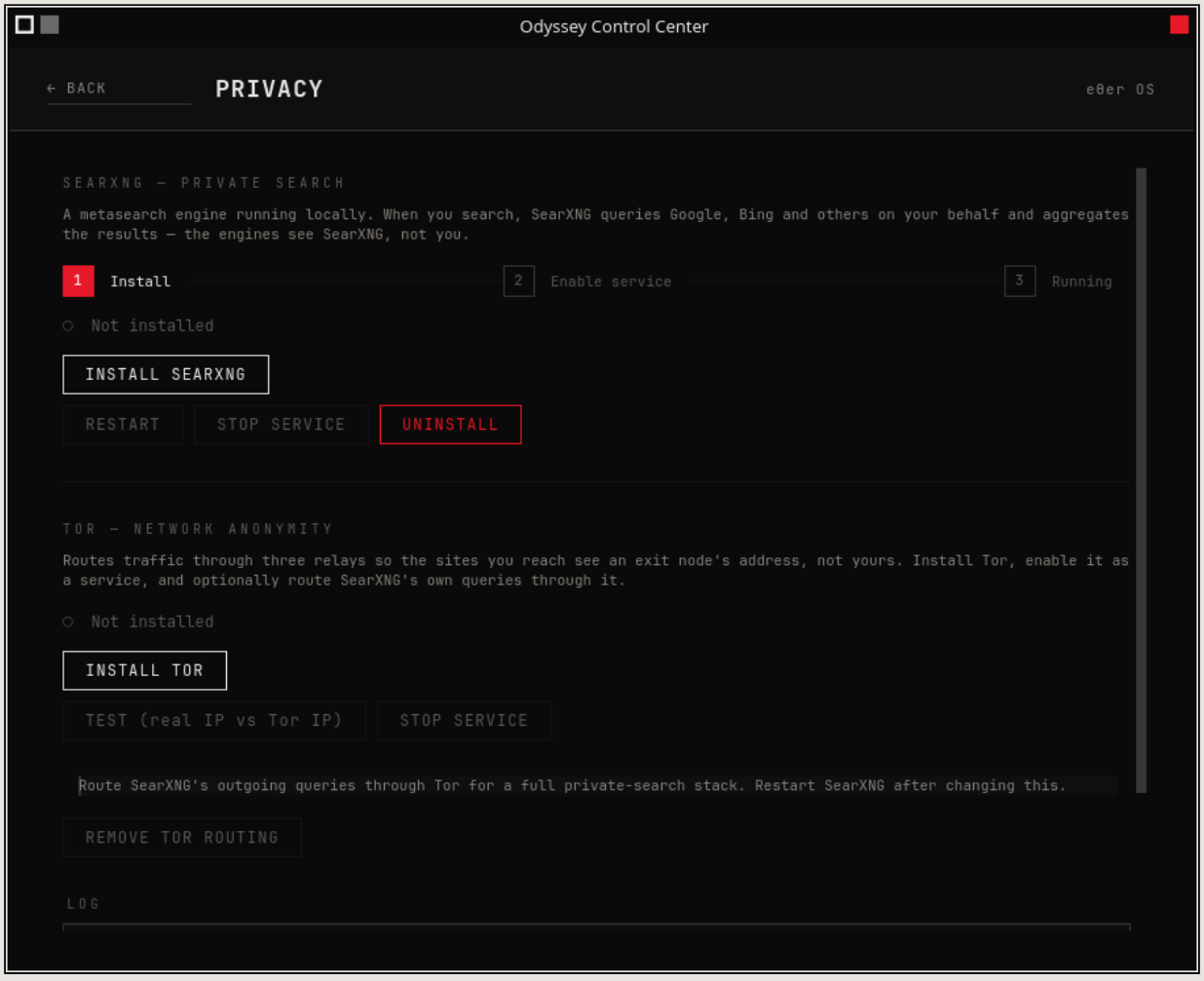

FIG · O painel em repouso: o rastreador do SearXNG na etapa 1 “Install,” status “Not installed,” um único botão INSTALL SEARXNG; o rastreador do Tor também mostra “Not installed” com INSTALL TOR abaixo.

SearXNG é um mecanismo de metabusca que roda localmente na sua própria máquina — quando você pesquisa, é o SearXNG que consulta Google, Bing e os outros em seu nome, então esses mecanismos veem o SearXNG fazendo a requisição, nunca você diretamente.

1 **INSTALL SEARXNG**

Clique. O log mostra a instalação real acontecendo — clonagem do código-fonte, criação de um usuário do sistema, configuração de um ambiente virtual Python.

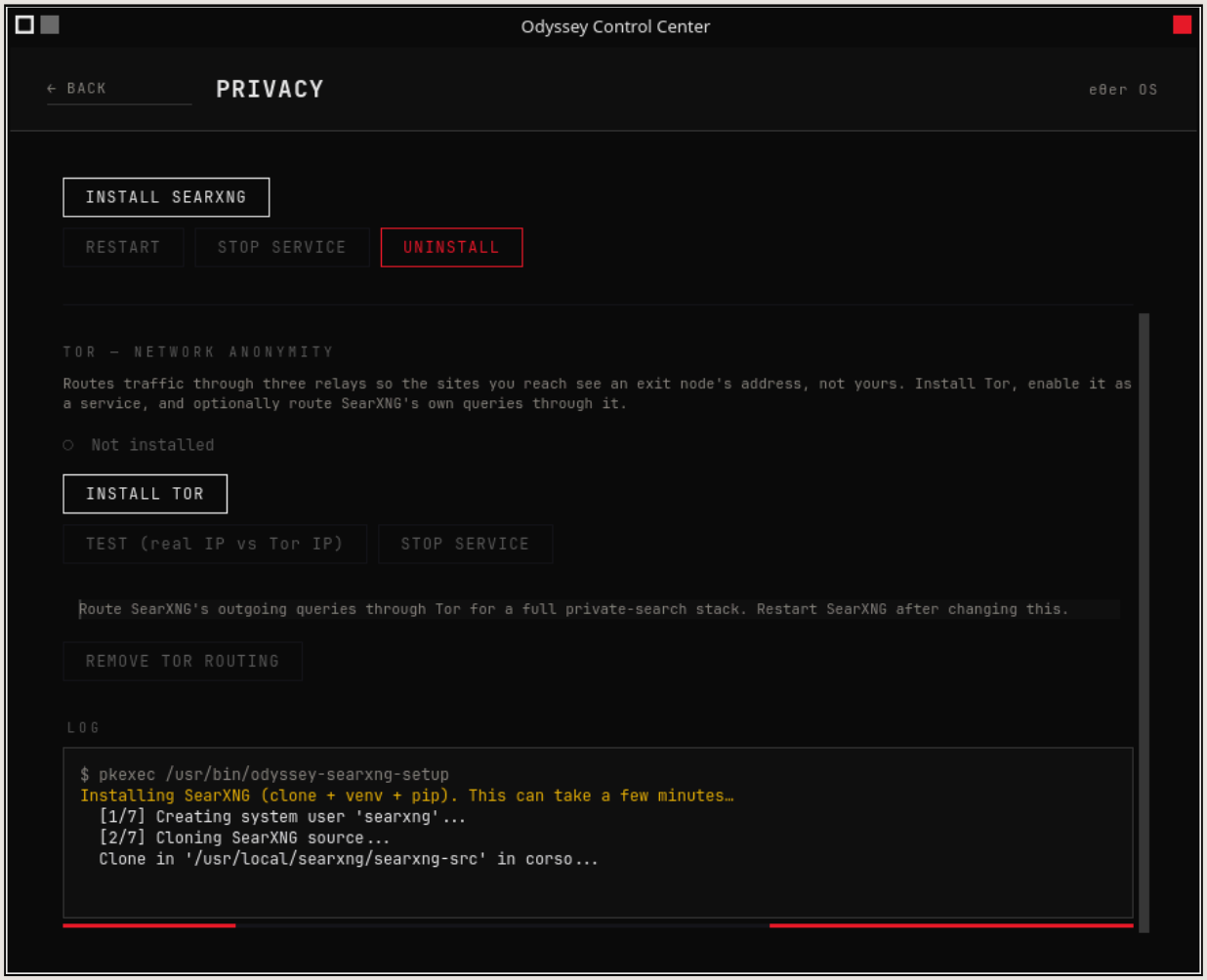

FIG · No meio da instalação: o log transmitindo saída real — “Installing SearXNG (clone + venv + pip). This can take a few minutes…” com as etapas 1/7 e 2/7 visíveis.

2 ENABLE SERVICE

Uma vez instalado, o rastreador passa para a etapa 2 e a linha de status fica amarela: “Installed — service not enabled.” Um único botão agora: ENABLE SERVICE.

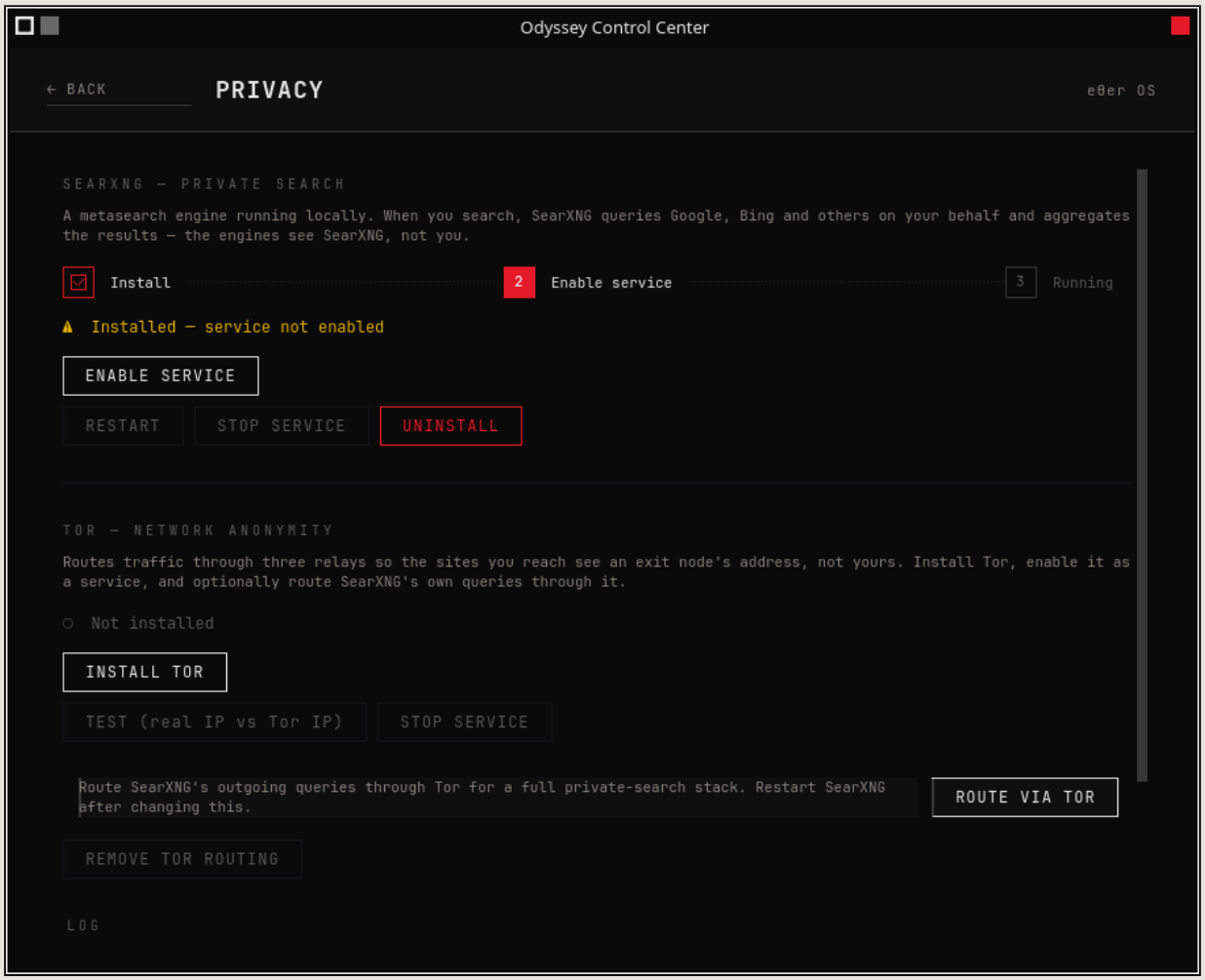

FIG · Etapa 2 ativa: "Installed — service not enabled," ENABLE SERVICE como único botão ativo, RESTART/STOP SERVICE/UNINSTALL ainda desativados abaixo.

3 Running

Depois de habilitar, o rastreador chega à etapa 3 e toda a fileira de controle se ativa. Uma vez em execução, **OPEN IN BROWSER** leva você direto para lá — o SearXNG escuta em `http://localhost:8888` por padrão.

WHAT	Privacy → INSTALL SEARXNG → ENABLE SERVICE → OPEN IN BROWSER.
WHY	Todo mecanismo de busca comercial constrói um perfil a partir das suas consultas ao longo do tempo; uma instância de metabusca local significa que só você jamais vê a consulta não agregada.
RESULT	Um mecanismo de busca privado rodando na sua própria máquina, acessível no seu navegador, consultando os grandes mecanismos em seu nome sem expor você diretamente a eles.
RISK	Baixo — é um serviço web local como qualquer outro. RESTART se ele se comportar mal, STOP SERVICE para pausá-lo sem removê-lo, UNINSTALL para removê-lo completamente.

7.2 Guia visual — Tor, passo a passo

Mesma forma em três etapas, rastreador independente, logo abaixo da seção do SearXNG.

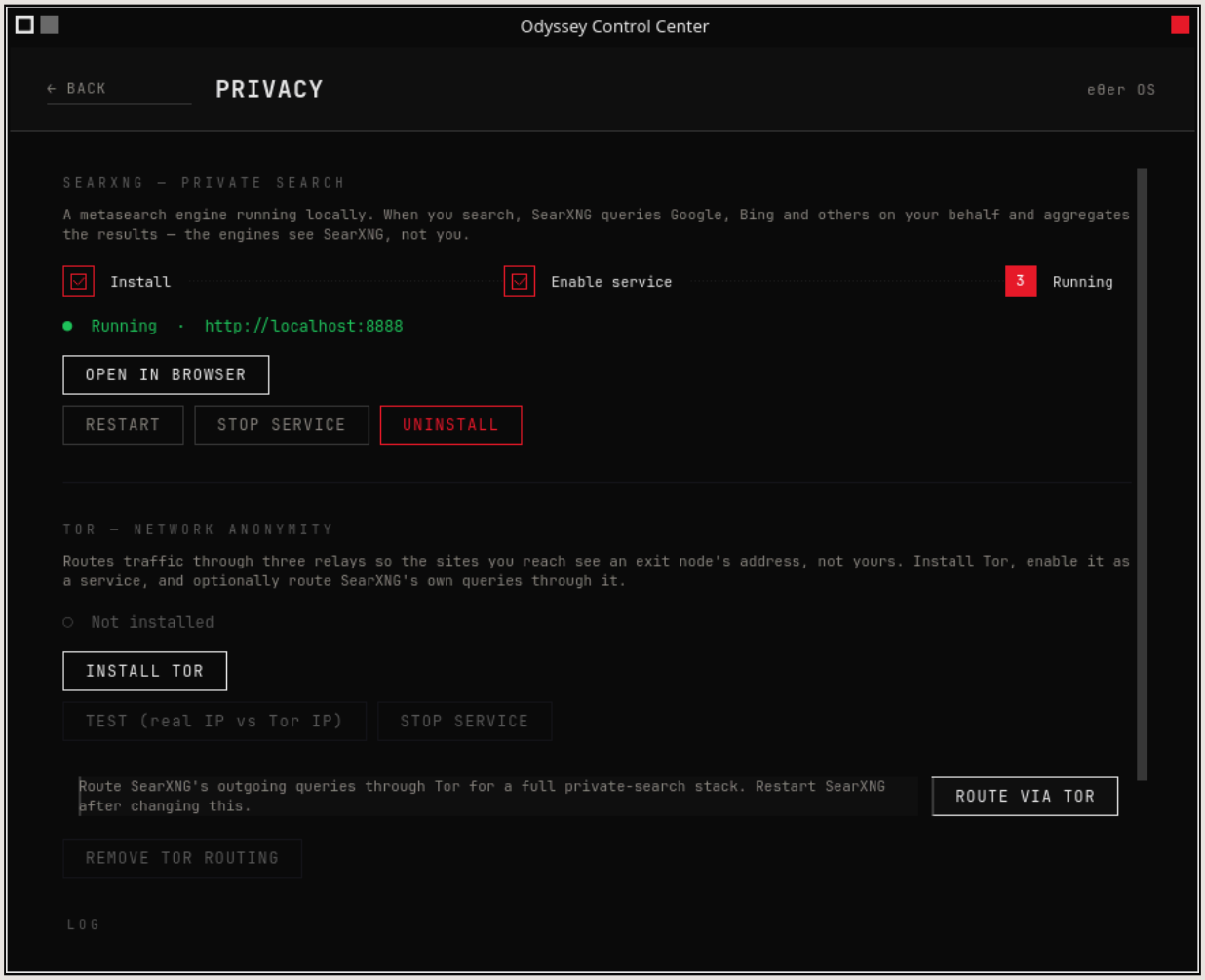

FIG · SearXNG totalmente em execução acima; a seção Tor abaixo ainda em “Not installed” com INSTALL TOR como único botão ativo.

O Tor roteia seu tráfego através de três retransmissores antes de alcançar seu destino, então o site que você visita vê o endereço do retransmissor de saída, nunca o seu.

1. **INSTALL TOR**

   O log confirma a instalação do pacote e mostra o serviço vinculado no runit.

2. **ENABLE SERVICE**

   O status muda para “Installed — service not enabled” — mesmo padrão do SearXNG.

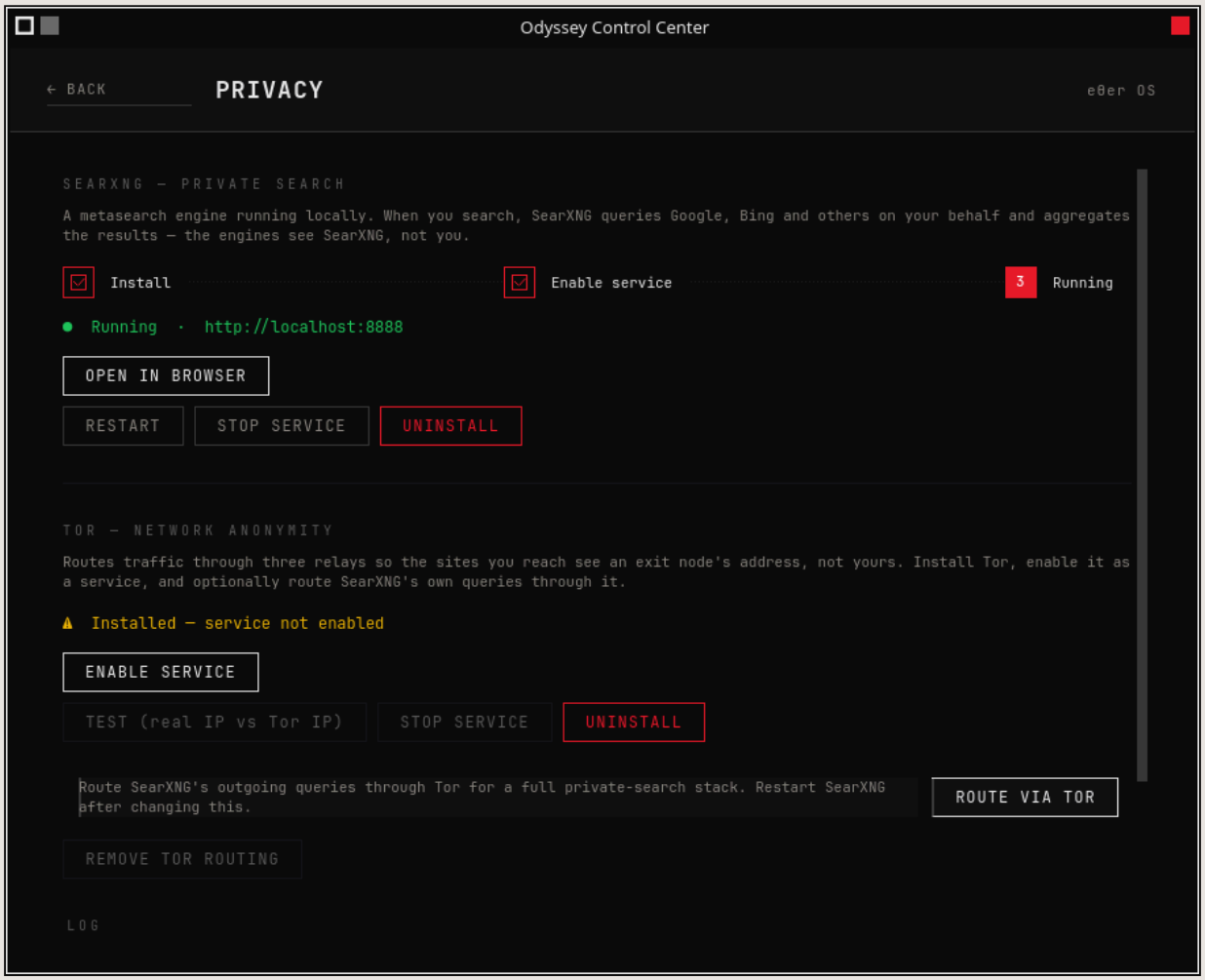

FIG · Tor na etapa 2: “Installed — service not enabled,” ENABLE SERVICE ativo, TEST e STOP SERVICE ainda desativados.

3 Running, e verifique

Uma vez habilitado, um botão dedicado **TEST (real IP vs Tor IP)** aparece — use-o. Ele compara seu endereço real com o que o Tor está apresentando, lado a lado, para que você tenha uma prova real de que funcionou em vez de uma luz de status na qual você só tem que acreditar.

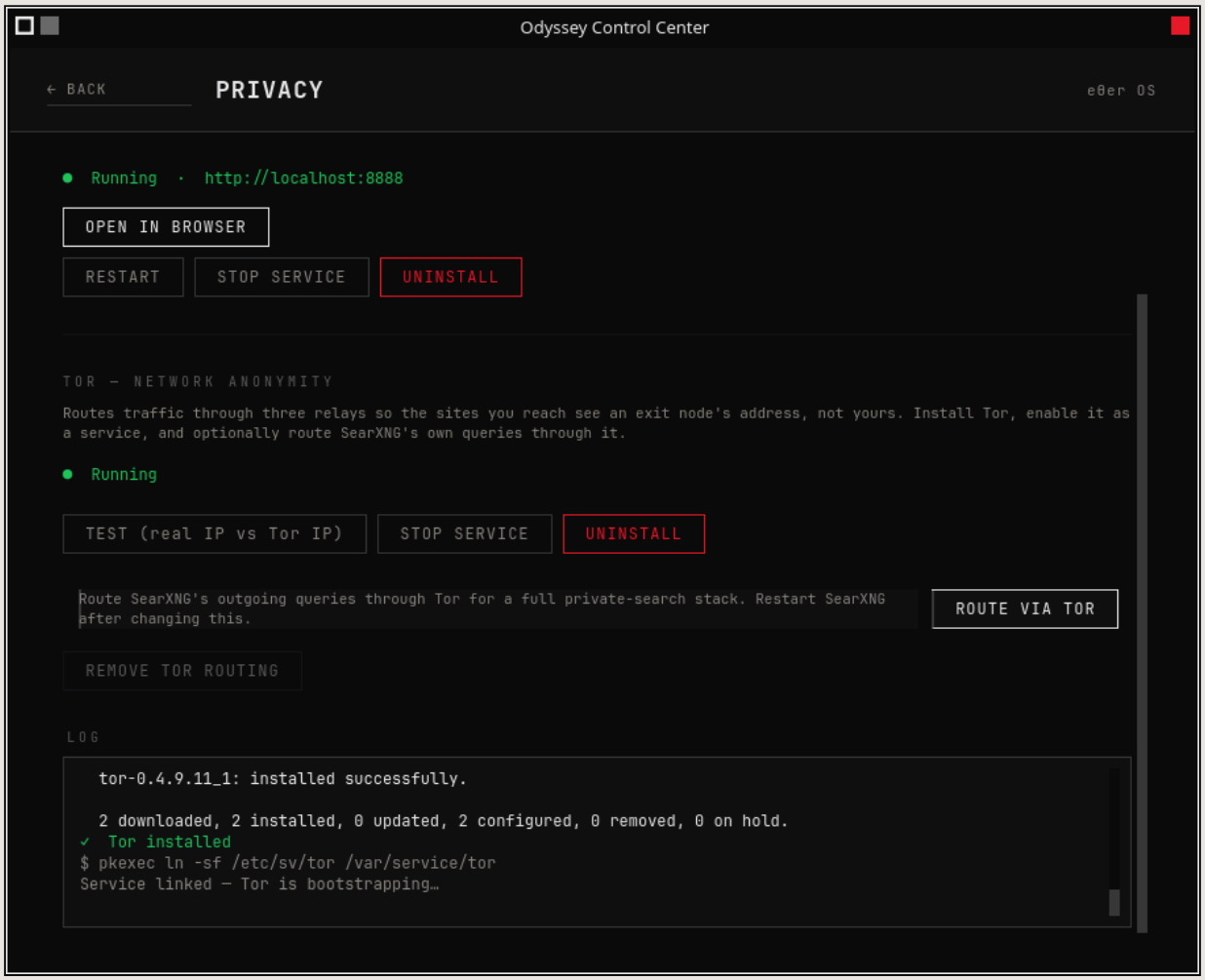

FIG · Ambas as seções verdes e em execução: SearXNG em localhost:8888 com OPEN IN BROWSER, Tor em execução com TEST / STOP SERVICE / UNINSTALL todos ativos.

WHAT	Privacy → INSTALL TOR → ENABLE SERVICE → TEST (real IP vs Tor IP) para confirmar.
WHY	Rotear a navegação sensível pelo Tor esconde seu endereço dos sites que você visita — TEST te dá uma prova real, não só uma afirmação.
RESULT	Um serviço Tor em execução pelo qual outros aplicativos podem se rotear; TEST mostra dois endereços genuinamente diferentes se estiver funcionando.
RISK	O tráfego pelo Tor é notavelmente mais lento — esse é o preço do anonimato. STOP SERVICE o pausa; UNINSTALL o remove.

7.3 Guia visual — combinando os dois

Uma vez que ambos estejam em execução, um aviso em linha aparece oferecendo rotear também as próprias consultas do SearXNG pelo Tor — não só suas requisições ao SearXNG, mas as requisições do SearXNG ao Google e ao Bing em seu nome.

WHAT	Privacy → o aviso “Route SearXNG’s outgoing queries through Tor” → ROUTE VIA TOR → RESTART na seção SearXNG para que tenha efeito.
WHY	Sem isso, o próprio SearXNG continua consultando os grandes mecanismos de busca diretamente — privado do seu lado, visível do lado deles. Rotear o próprio tráfego do SearXNG pelo Tor fecha essa lacuna para uma pilha de busca totalmente privada.
RESULT	As consultas de saída do SearXNG, não só suas consultas a ele, passam pelo Tor.
RISK	Os resultados carregam notavelmente mais devagar, pelo mesmo motivo de qualquer tráfego roteado pelo Tor. REMOVE TOR ROUTING reverte isso, novamente seguido de um reinício.

```
▸ ADVANCED

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

7.4 Advanced

O QUE INSTALL SEARXNG REALMENTE EXECUTA

```
$ pkexec /usr/bin/odyssey-searxng-setup
# cria um usuário de sistema dedicado ‘searxng’,
# clona o código-fonte em /usr/local/searxng/searxng-src,
# constrói um venv Python e instala as dependências com pip
```

SERVIÇOS E PORTAS, MANUALMENTE

```
# o serviço runit do SearXNG
$ sudo sv status searxng
$ sudo sv restart searxng
# o serviço runit do Tor — a etapa de habilitação do painel é literalmente isso
$ sudo ln -sf /etc/sv/tor /var/service/tor
$ sudo sv status tor
# verifica que o Tor realmente está roteando
$ curl -socks5 localhost:9050 https://check.torproject.org/api/ip
```

ROTEANDO O SEARXNG PELO TOR MANUALMENTE

O proxy SOCKS do Tor escuta em `127.0.0.1:9050` por padrão. Rotear o SearXNG através dele significa apontar seu cliente HTTP de saída para esse proxy — a configuração vive no próprio arquivo `settings.yml` do SearXNG, na configuração de requisições de saída, em vez de em uma variável de proxy em nível de sistema. O botão ROUTE VIA TOR do painel edita esse arquivo diretamente e reinicia o serviço; fazer isso manualmente significa localizar e editar o mesmo bloco `outgoing:` em `/usr/local/searxng/searxng-src/searx/settings.yml`.

POR QUE O SEARXNG PRECISA DE UM REINÍCIO DEPOIS DE MUDANÇAS DE ROTEAMENTO

O SearXNG lê sua configuração de proxy só uma vez, na inicialização — não é uma configuração recarregável a quente. Isso vale tanto se você mudar via painel quanto manualmente, motivo pelo qual os dois caminhos terminam na mesma etapa RESTART.

ARQUIVOS QUE ESTE PAINEL TOCA

CAMINHO	FUNÇÃO
`/usr/local/searxng/searxng-src/`	A própria instalação do SearXNG — código-fonte, venv, configuração
`/etc/sv/searxng`, `/etc/sv/tor`	Definições dos serviços runit
`/var/service/`	Onde habilitar um serviço o vincula na árvore de supervisão em execução

CHAPTER 8

Containers

Gestão de contêineres do início ao fim: configuração do motor, uma lista de contêineres em execução, um catálogo curado para começar, comandos brutos para tudo que não está nele, e uso de disco — tudo em um único painel rolável.

8.1 Guia visual — o motor

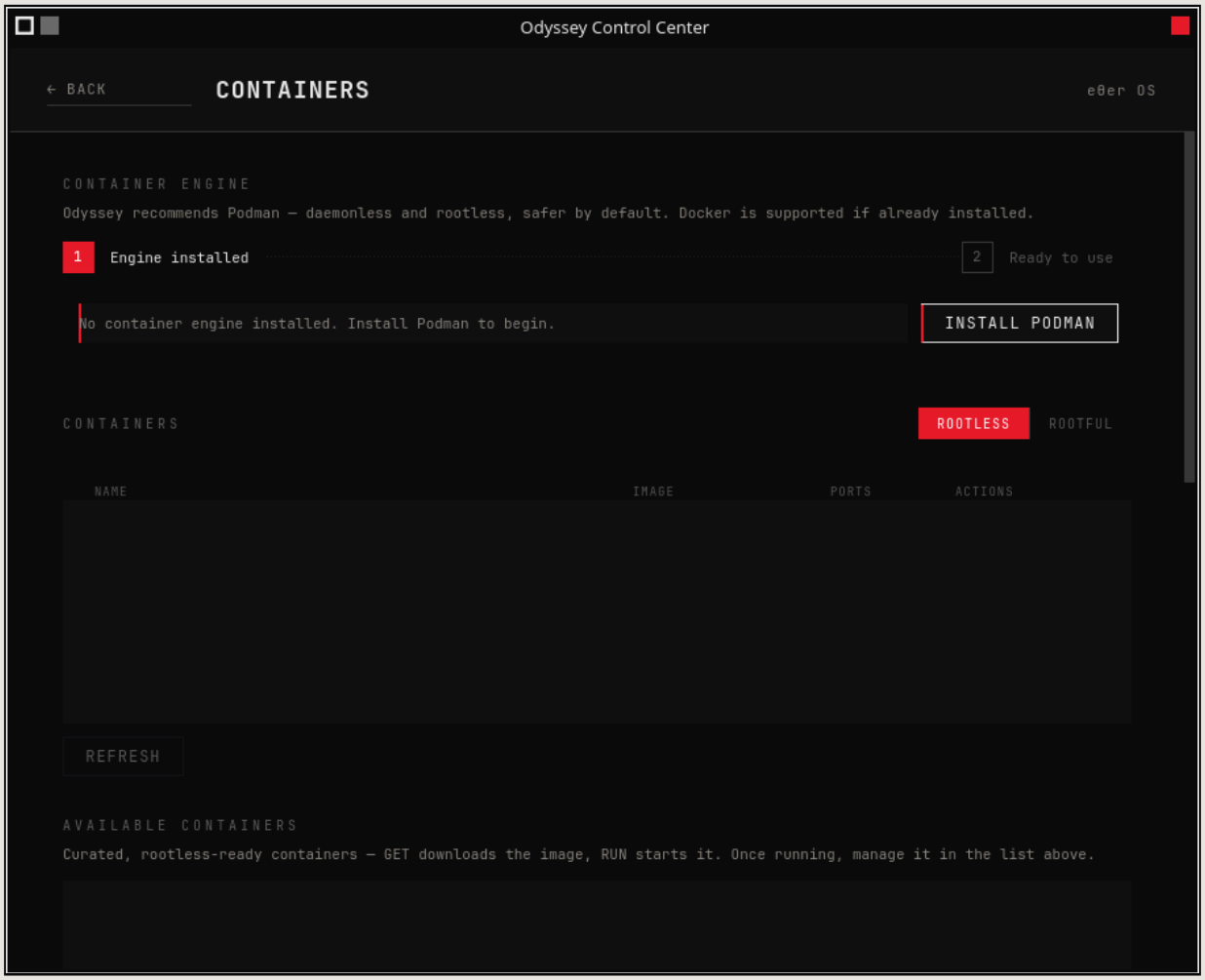

FIG · Etapa 1 de 2: “No container engine installed. Install Podman to begin,” com INSTALL PODMAN como única ação ativa. As seções Containers list e Available containers abaixo estão visíveis mas em estado vazio.

O Odyssey recomenda o **Podman** — sem daemon e rootless, o que significa que os contêineres rodam sem um serviço em segundo plano permanente e, por padrão, sem privilégios de root. O **Docker** também é suportado se já for o que você usa.

1 **INSTALL PODMAN**

O log mostra a instalação real do pacote.

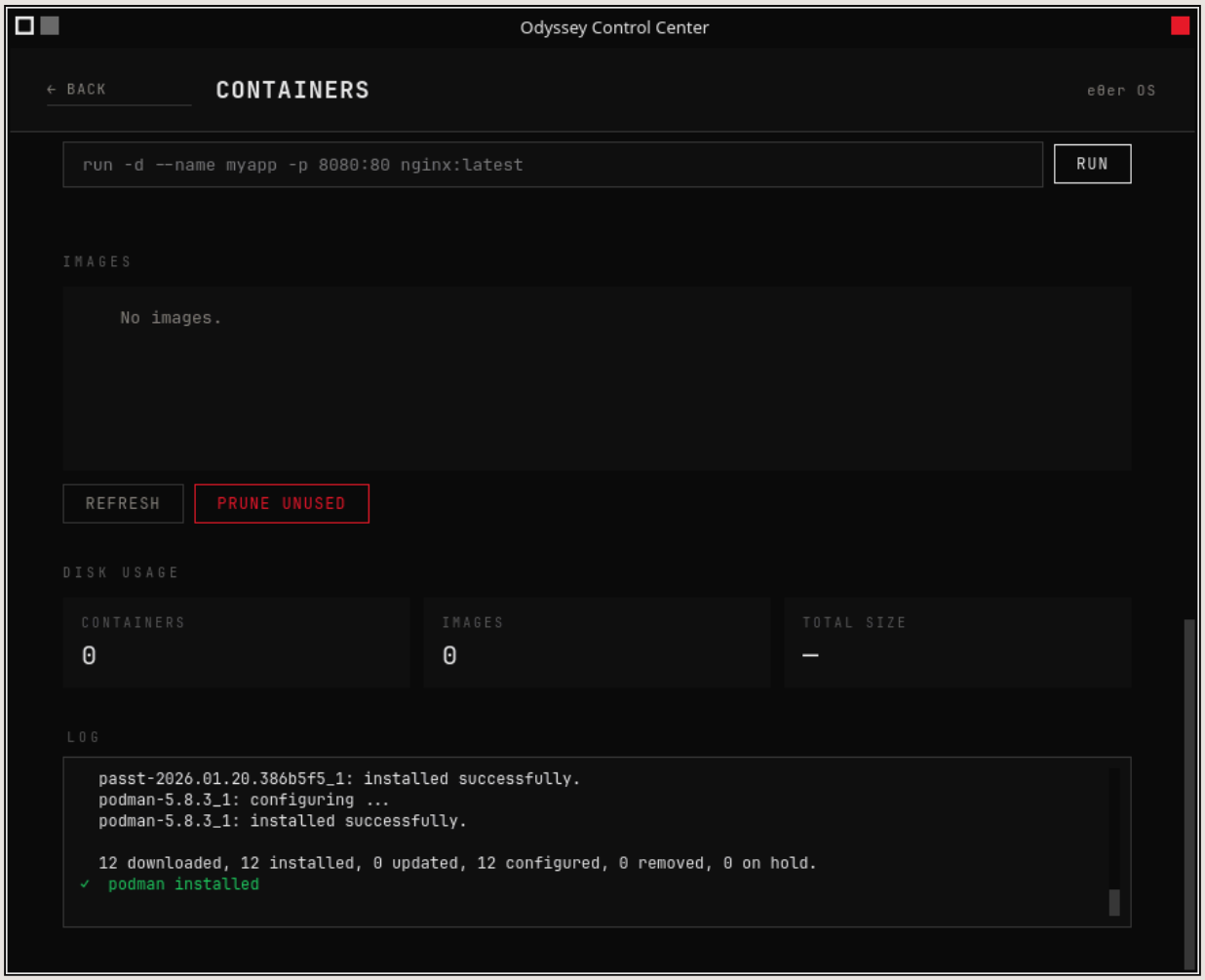

FIG · Depois da instalação: “Engine installed” marcado em vermelho, etapa 2 “Ready to use,” a linha informativa dizendo “podman version 5.8.3 · Podman installed · Docker not installed,” e um botão INSTALL DOCKER TOO para quem também quer o Docker em paralelo.

2 Ready to use

O motor está ativo. **INSTALL DOCKER TOO** continua disponível se você quiser os dois motores lado a lado — eles coexistem sem conflito.

8.2 Guia visual — rootless vs. rootful, e o estado vazio

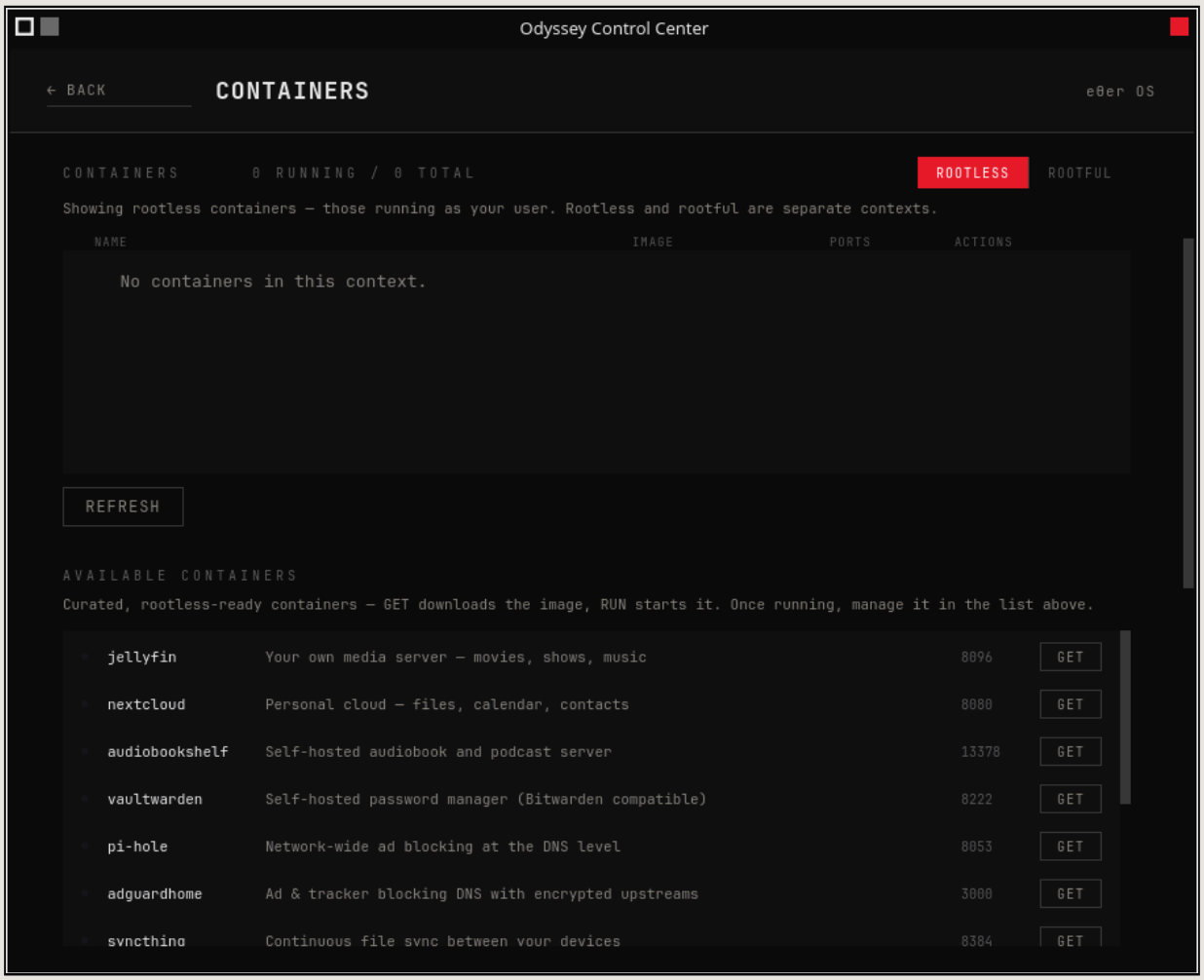

FIG · ROOTLESS selecionado (destacado em vermelho) ao lado de ROOTFUL, "0 RUNNING / 0 TOTAL," e "No containers in this context." abaixo dos cabeçalhos de coluna NAME / IMAGE / PORTS / ACTIONS.

O interruptor **ROOTLESS / ROOTFUL** no topo muda qual contexto você está visualizando e gerenciando — os dois são namespaces de contêineres completamente separados, não um filtro sobre uma lista combinada. O Odyssey mostra ROOTLESS por padrão, e esse padrão é deliberado:

> **QUAL DOS DOIS VOCÊ DEVERIA USAR**
>
> Rootless é a escolha mais segura e para onde o painel te orienta: um contêiner que escapa do confinamento em modo rootless de qualquer forma não consegue fazer nada que sua própria conta de usuário já não pudesse. Rootful existe para os casos específicos que precisam dele — certas portas de rede de número baixo, algum acesso a hardware — e deveria ser a exceção, não o hábito.

8.3 Guia visual — o catálogo curado, do início ao fim

Role até **Available containers** — uma lista já pronta para que o painel nunca seja uma caixa vazia esperando que você já saiba os nomes das imagens de cor. Cada linha mostra o contêiner, uma descrição em uma linha, e a porta que ele usa.

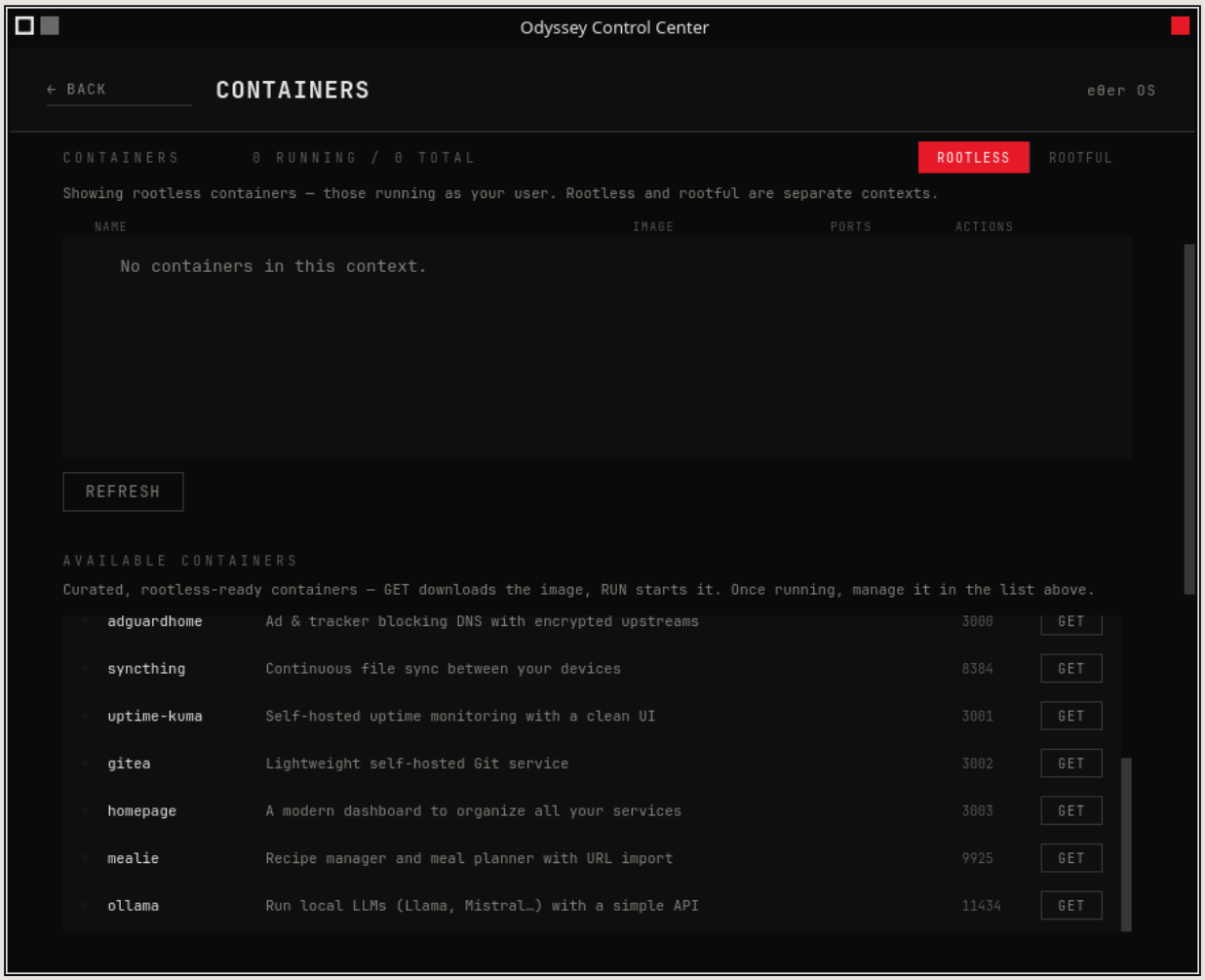

FIG · A lista do catálogo: adguardhome (porta 3000), syncthing (8384), uptime-kuma (3001), gitea (3002), homepage (3003) — com um botão GET em cada linha.

Esse número de porta é aquele em que você vai alcançar o serviço uma vez em execução — `homepage` com `3003`, por exemplo, significa `http://localhost:3003` uma vez ativo.

1 **GET**

Clique em GET em `homepage`. Isso só baixa a imagem do contêiner — nada está em execução ainda.

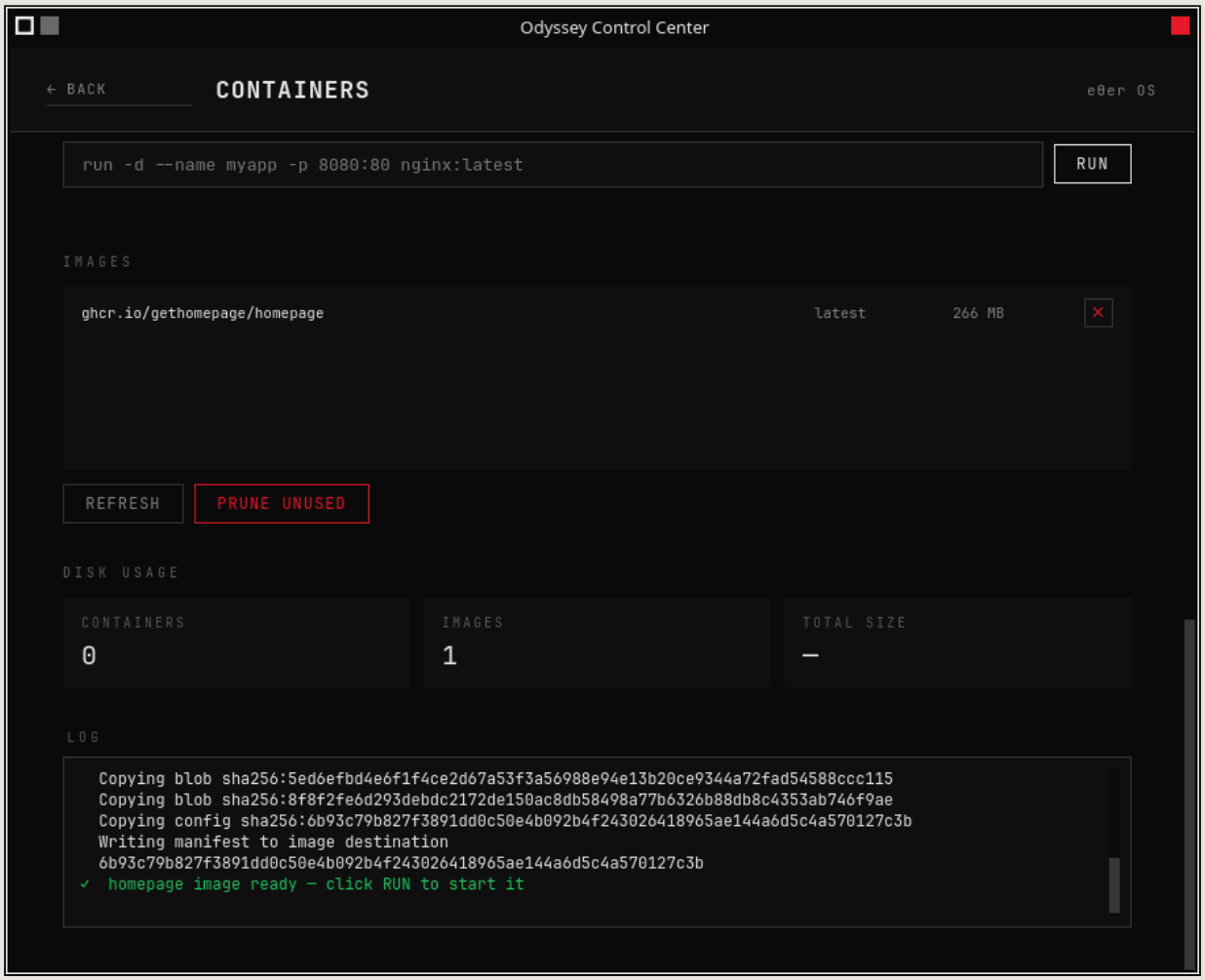

FIG · O log transmitindo um download real de imagem: “Copying blob sha256:5ed6…”, “Writing manifest to image destination,” terminando em verde — “homepage image ready — click RUN to start it.”

2 RUN

A linha do botão GET agora mostra RUN em seu lugar — clique nele para realmente iniciar o contêiner.

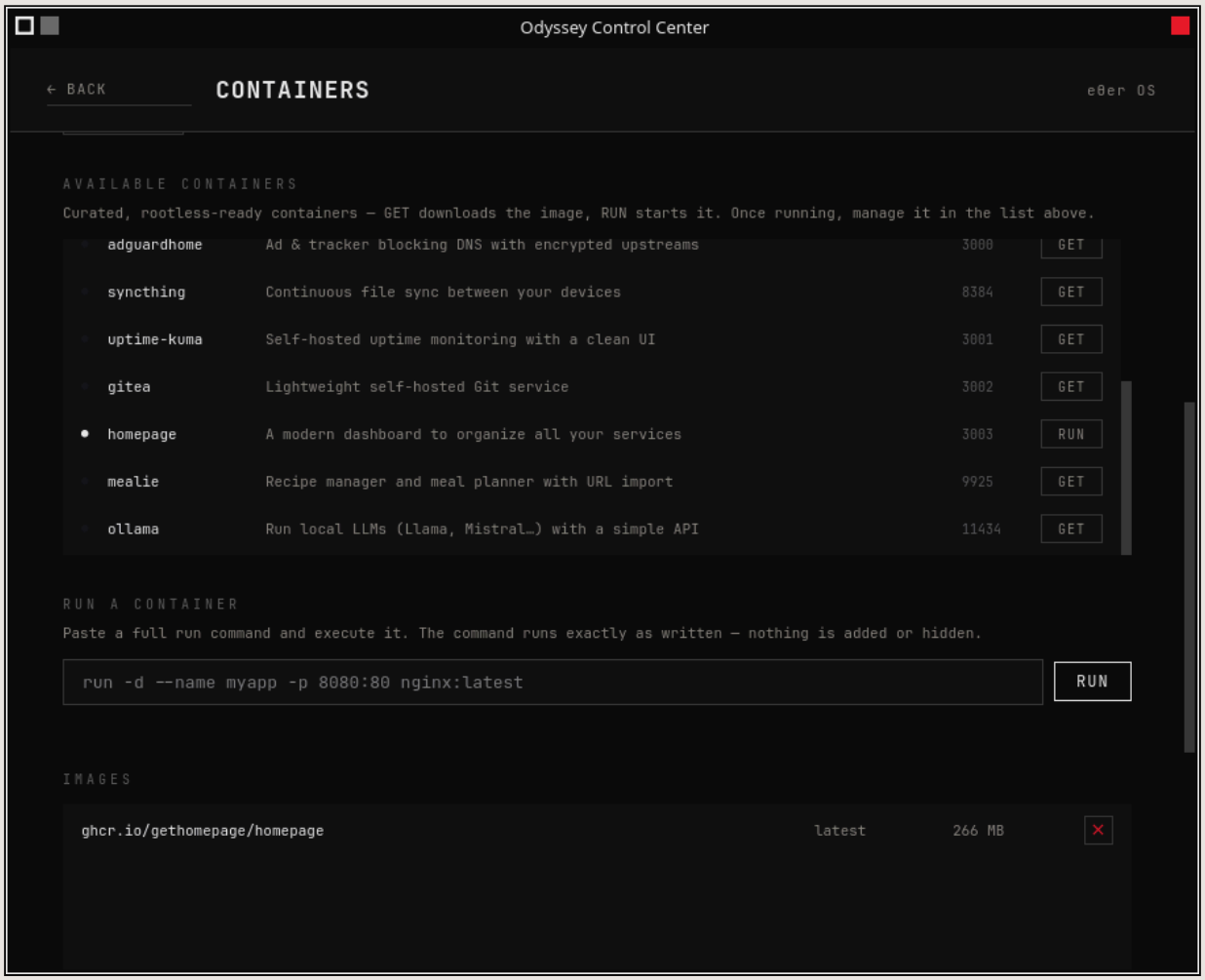

FIG · homepage agora listado na seção CONTAINERS acima, ponto verde, caminho da imagem ghcr.io/gethomepage/homepage, portas "0.0.0.0:3003→…", com sua própria linha de ícones de ação; ainda visível abaixo em Available Containers com um novo botão RUN para iniciar outra instância.

Uma vez em execução, `homepage` aparece em dois lugares ao mesmo tempo: como entrada ativa na lista **Containers** acima (com sua porta e ícones de ação), e ainda no catálogo abaixo caso você queira iniciar uma segunda instância depois.

WHAT	Containers → Available containers → GET em algo do catálogo → RUN uma vez baixado.
WHY	A divisão em duas etapas (baixar, depois iniciar) significa que você pode obter uma imagem com antecedência sem se comprometer a rodá-la imediatamente — útil em uma conexão lenta ou quando você está só explorando o que está disponível.
RESULT	Um contêiner em execução, visível na lista principal com seu mapeamento de porta, gerenciável a partir dali.
RISK	Baixo — os serviços auto-hospedados do catálogo curado são escolhidos para serem rootless-friendly e de baixo risco por padrão. A exposição é a mesma de rodar qualquer serviço web: qualquer porta em que ele escute é alcançável conforme suas configurações de Firewall.

8.4 Guia visual — a lista de contêineres e suas ações

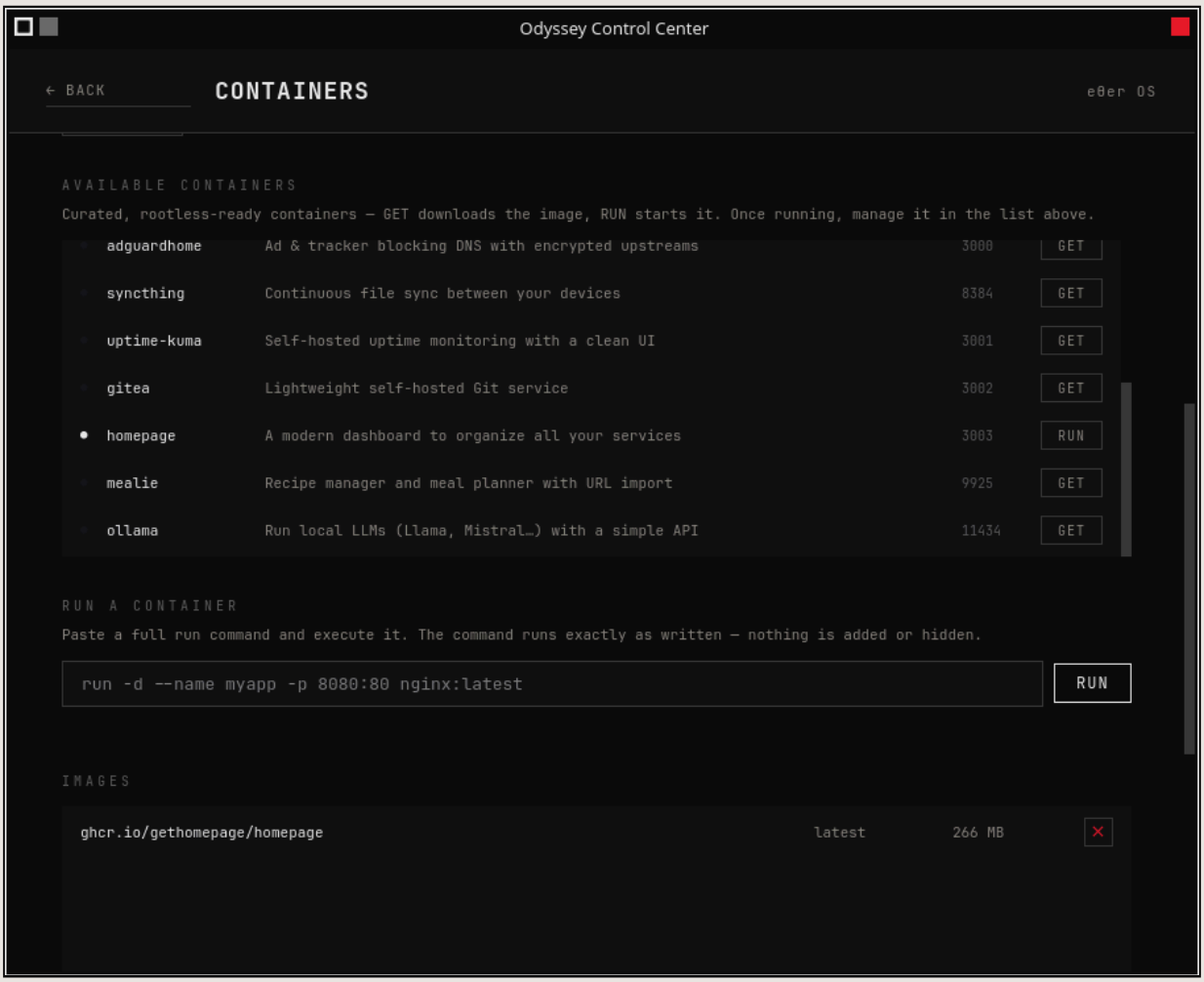

FIG · A linha do homepage na lista Containers, com seus cinco ícones de ação visíveis à direita — os explicados abaixo.

Cada linha de um contêiner em execução termina em um conjunto de ícones — passe o mouse sobre qualquer um para ver seu rótulo, mas aqui está o que cada um faz:

ÍCONE	AÇÃO
abrir (seta)	Abre o serviço diretamente no seu navegador — o caminho rápido uma vez que algo está em execução
parar / iniciar (quadrado / triângulo)	Para o contêiner sem removê-lo, ou o inicia novamente
reiniciar (seta circular)	Para e inicia em uma única ação — a solução para "está rodando mas não responde"
logs (ícone de linhas)	Abre a própria saída de log do contêiner — o primeiro lugar a olhar se algo não está funcionando
remover (×)	Remove o contêiner completamente — a imagem em si permanece baixada, listada separadamente em Images

8.5 Guia visual — executar um contêiner, e abri-lo

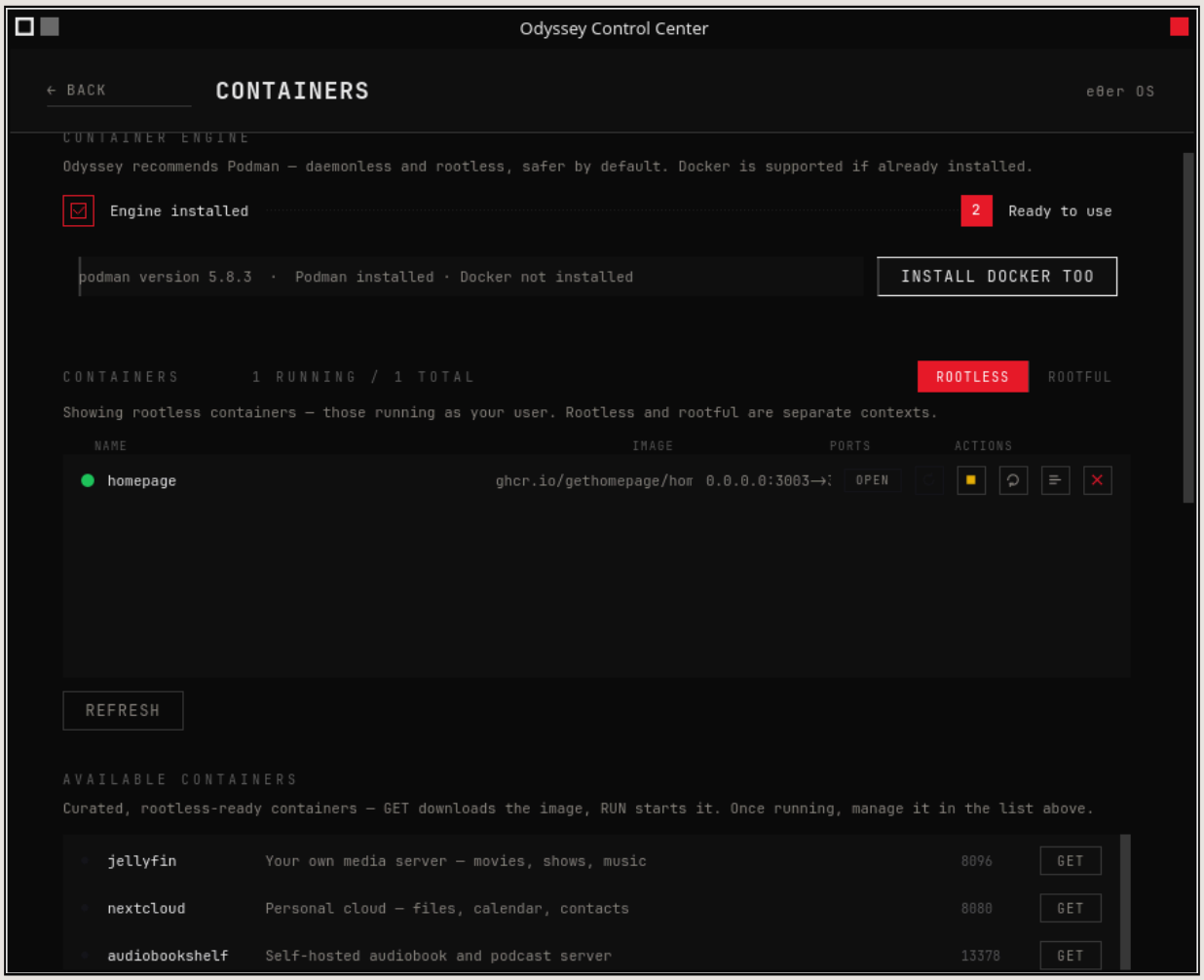

FIG · O campo “Run a container” com um comando de exemplo como texto de espaço reservado, o botão RUN, e a seção Images abaixo listando ghcr.io/gethomepage/homepage · latest · 266 MB com um botão de remoção.

Para qualquer coisa que não esteja no catálogo, **Run a container** recebe um comando `podman run` (ou `docker run`) completo e o executa exatamente como está escrito — nada é adicionado, nada é escondido.

WHAT	Containers → “Run a container” → cole um comando run completo (ex. `run -d --name myapp -p 8080:80 nginx:latest`) → RUN.
WHY	O catálogo curado cobre os casos comuns; essa é a válvula de escape para uma imagem específica, flags específicas, portas específicas que o catálogo não oferece.
RESULT	O contêiner inicia exatamente como o comando especifica, e se junta à lista principal Containers uma vez em execução.
RISK	O que quer que o comando em si faça — isso é um repasse fiel, não uma sandbox. Trate com o mesmo cuidado com que você digitaria o mesmo comando em um terminal.

Abaixo, **Images** lista cada imagem baixada independentemente de haver um contêiner rodando dela no momento, com **REFRESH** e **PRUNE UNUSED** para recuperar espaço de imagens que ninguém mais referencia.

ABRINDO UM CONTÊINER EM EXECUÇÃO

Clicar em OPEN na linha de um contêiner em execução — a mesma ação de abrir da linha de ícones acima — lança seu navegador padrão diretamente para seu endereço e porta, ex. `http://localhost:3003` para `homepage`. Não precisa lembrar você mesmo do número da porta uma vez que ele está em execução.

8.6 Uso de disco

Bem no final do painel, três totais em tempo real — contêineres, imagens, tamanho total — para que você sempre saiba quanto os contêineres estão custando em disco sem abrir um terminal.

```
▸ ADVANCED

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

8.7 Advanced

EQUIVALENTES DE LINHA DE COMANDO

```
# o que o painel chama de "GET"
$ podman pull ghcr.io/gethomepage/homepage
# o que o painel chama de "RUN" em uma entrada do catálogo
$ podman run -d -name homepage -p 3003:3000 ghcr.io/gethomepage/homepage
# lista os contêineres em execução — contexto rootless
$ podman ps
# o mesmo, rootful
$ sudo podman ps
# logs, parar, reiniciar, remover — a linha de ícones, manualmente
$ podman logs -f homepage
$ podman stop homepage
$ podman restart homepage
$ podman rm homepage
# imagens e uso de disco
$ podman images
$ podman system df
$ podman image prune
```

ROOTLESS VS. ROOTFUL, POR BAIXO

O Podman rootless executa os contêineres como o seu próprio usuário, dentro de um namespace de usuário — o "root" do contêiner é mapeado para uma faixa de UID sem privilégios no host, então mesmo um processo de contêiner totalmente comprometido continua vinculado às permissões da sua própria conta. O Podman rootful (`sudo podman ...`) funciona da maneira tradicional, com o root do contêiner mapeado para o root real do host. O interruptor do painel na verdade só alterna entre executar `podman` simples ou `sudo podman` — diretórios de estado realmente separados, listas de contêineres separadas, nada compartilhado entre os dois contextos por padrão.

```
# onde cada contexto guarda seu próprio estado
$ podman info -format '{{.Store.GraphRoot}}'
# rootless:  /.local/share/containers/storage
# rootful: /var/lib/containers/storage
```

POR QUE ALGUMAS ENTRADAS DO CATÁLOGO PRECISAM DE PORTAS ESPECÍFICAS ABERTAS

O mapeamento de porta de um contêiner (`3003:3000` no exemplo do homepage) só torna o serviço alcançável **a partir desta máquina** por padrão — o Podman o vincula a `localhost` a menos que seja dito o contrário. Alcançá-lo de outro dispositivo na sua rede requer adicionalmente que essa porta esteja aberta no painel Firewall (Capítulo 4) — os dois sistemas são camadas independentes, e um contêiner em execução com uma porta mapeada não é automaticamente alcançável pela internet ou pela LAN só porque o Podman está escutando nela.

CHAPTER 9

Gaming Mode

Um interruptor principal no topo, uma fileira de status em tempo real, e quatro abas: **PACKAGES**, **GAMEMODE**, **MANGOHUD**, **PROTON**.

9.1 Guia visual — o interruptor principal

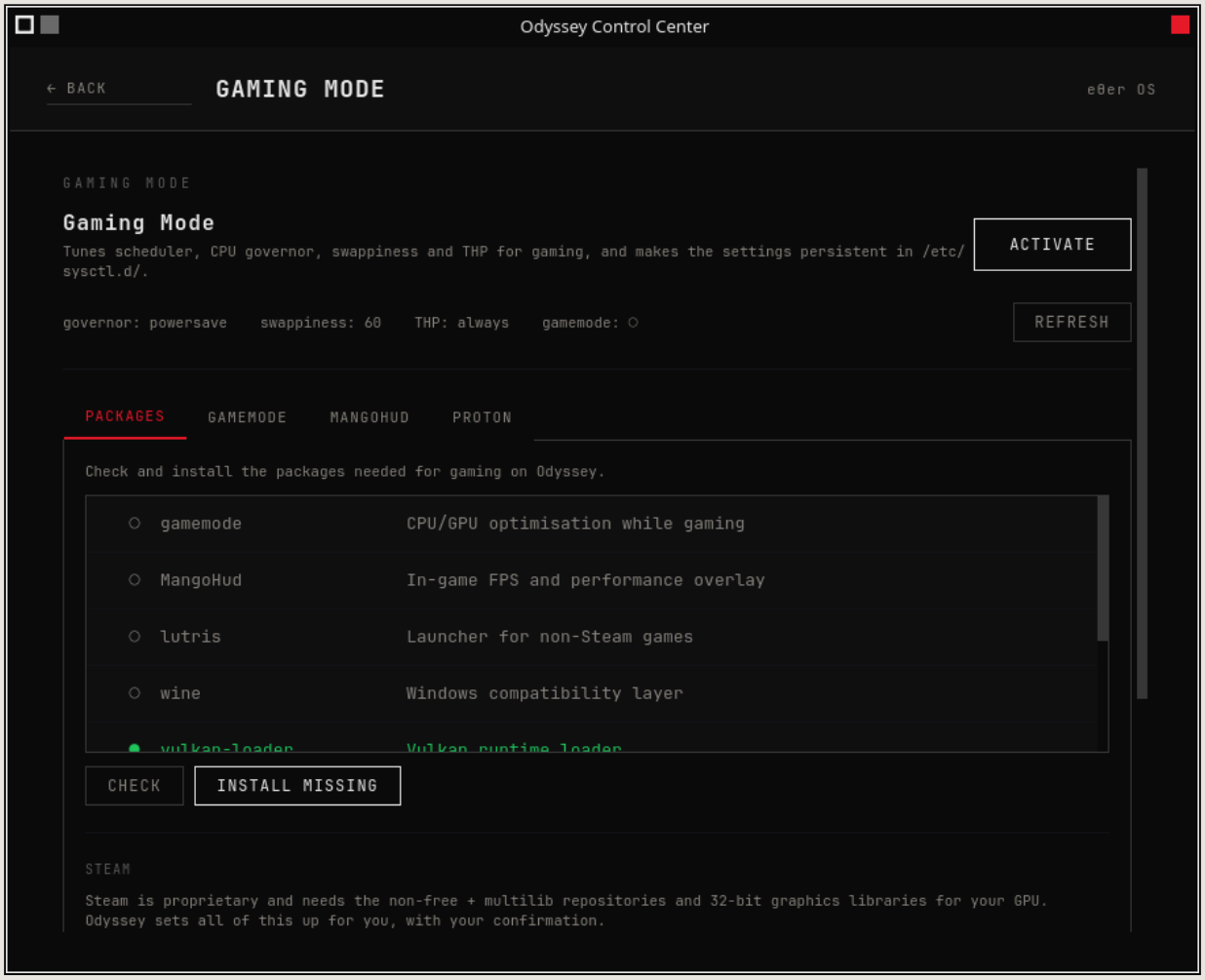

FIG · A seção superior do painel: “Gaming Mode — Tunes scheduler, CPU governor, swappiness and THP for gaming, and makes the settings persistent in /etc/sysctl.d/,” um botão ACTIVATE, um botão REFRESH, e a fileira de status ao vivo: governor powersave, swappiness 60, THP always, gamemode off.

A fileira de status mostra os mesmos valores ao vivo que os painéis Kernel e Memory mostram — governor, swappiness, estado do THP, e se o serviço `gamemode` está em execução — para que você possa confirmar de relance se a ativação realmente funcionou, sem visitar separadamente nenhum dos dois painéis.

WHAT	Gaming Mode → ACTIVATE.
WHY	Visitar manualmente Kernel e Memory para definir à mão um governor, uma swappiness e um valor de THP adequados para gaming é exatamente o tipo de tarefa em várias etapas que um modo dedicado deveria eliminar.
RESULT	Scheduler, governor, swappiness e THP passam todos para valores adequados para gaming, aplicados imediatamente e tornados persistentes — a fileira de status se atualiza para refletir a mudança.
RISK	Baixo, e reversível — os mesmos valores que você definiria de outra forma à mão em Kernel/Memory, só agrupados em um clique.

9.2 Guia visual — Packages e Steam

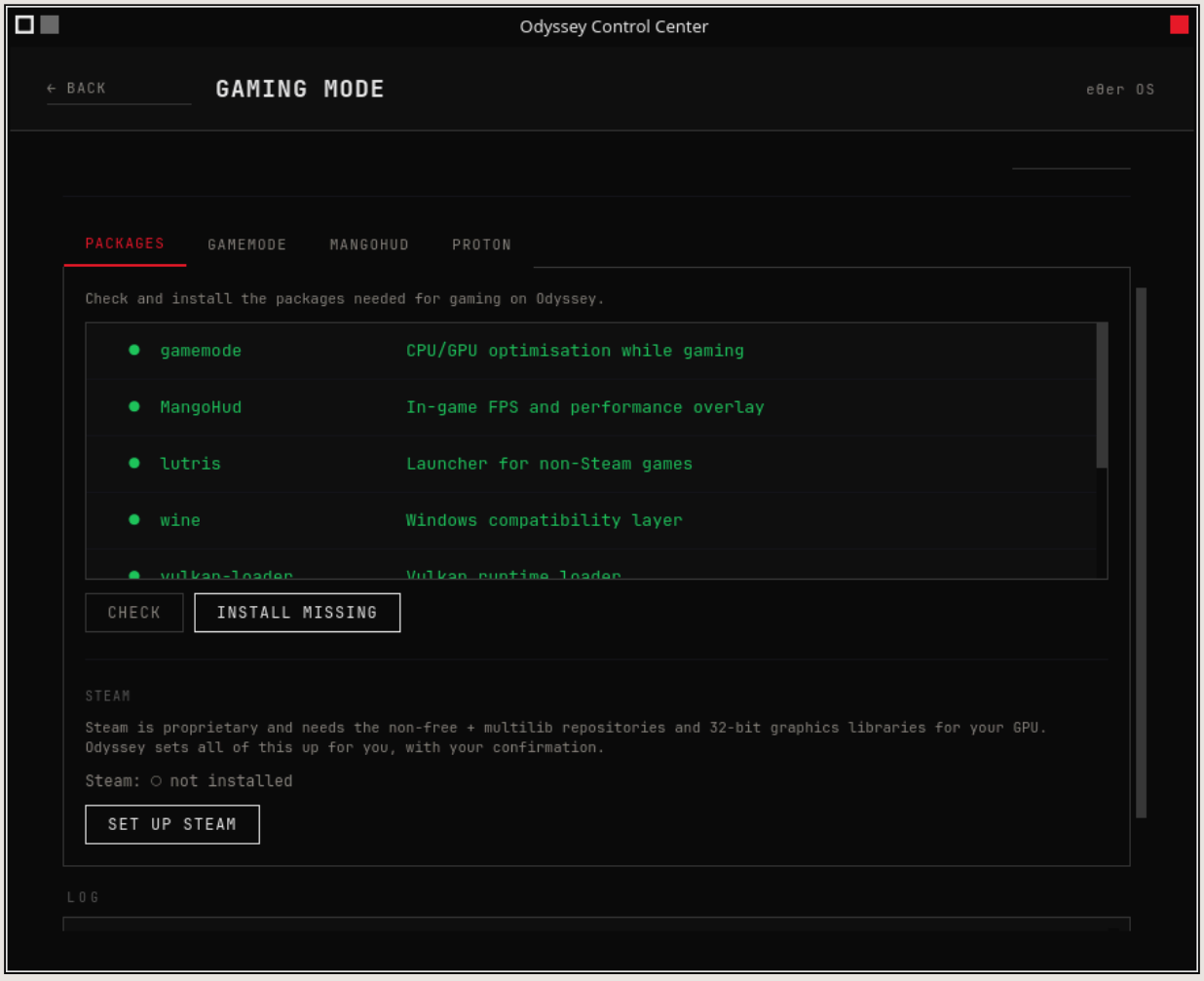

FIG · A aba PACKAGES: gamemode, MangoHud, lutris, wine, vulkan-loader todos com pontos verdes preenchidos (instalados); CHECK e INSTALL MISSING acima; a seção Steam abaixo com "Steam: ● installed" e um botão REINSTALL STEAM.

Uma lista de verificação dos pacotes geralmente necessários para gaming — CHECK vê o que falta, INSTALL MISSING preenche as lacunas em uma única ação. Uma vez que tudo mostra um ponto verde preenchido, como na captura, você está totalmente configurado.

O Steam tem seu próprio fluxo dedicado abaixo da lista de verificação, porque precisa de mais que uma simples instalação de pacote.

1 **SET UP STEAM**

Clique — isso ainda não instala nada, primeiro abre uma caixa de diálogo de confirmação informativa.

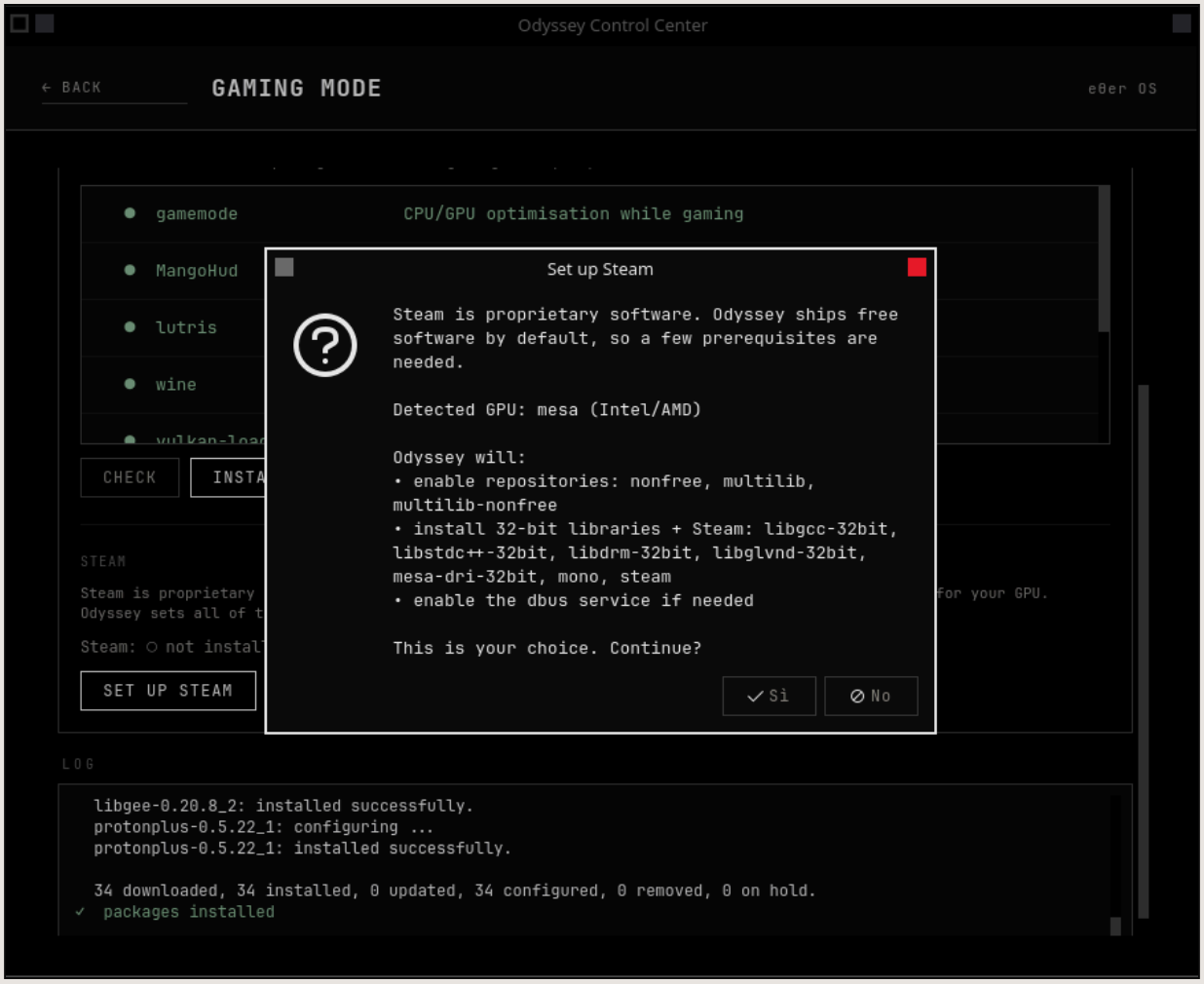

FIG · A caixa de diálogo de confirmação: “Steam is proprietary software. Odyssey ships free software by default, so a few prerequisites are needed. Detected GPU: mesa (Intel/AMD).” Uma lista com marcadores do que será exatamente habilitado e instalado, terminando em “This is your choice. Continue?” com os botões Sí / No.

A caixa de diálogo diz exatamente o que está prestes a acontecer antes de você concordar: quais repositórios são habilitados (`nonfree`, `multilib`, `multilib-nonfree`), quais bibliotecas de 32 bits são instaladas de acordo com sua GPU **detectada**, e que o serviço dbus será habilitado se ainda não estiver. Nada fica escondido atrás da confirmação.

2 **Sí**

Confirma. O log transmite a instalação real — downloads de pacotes, configuração, finalização.

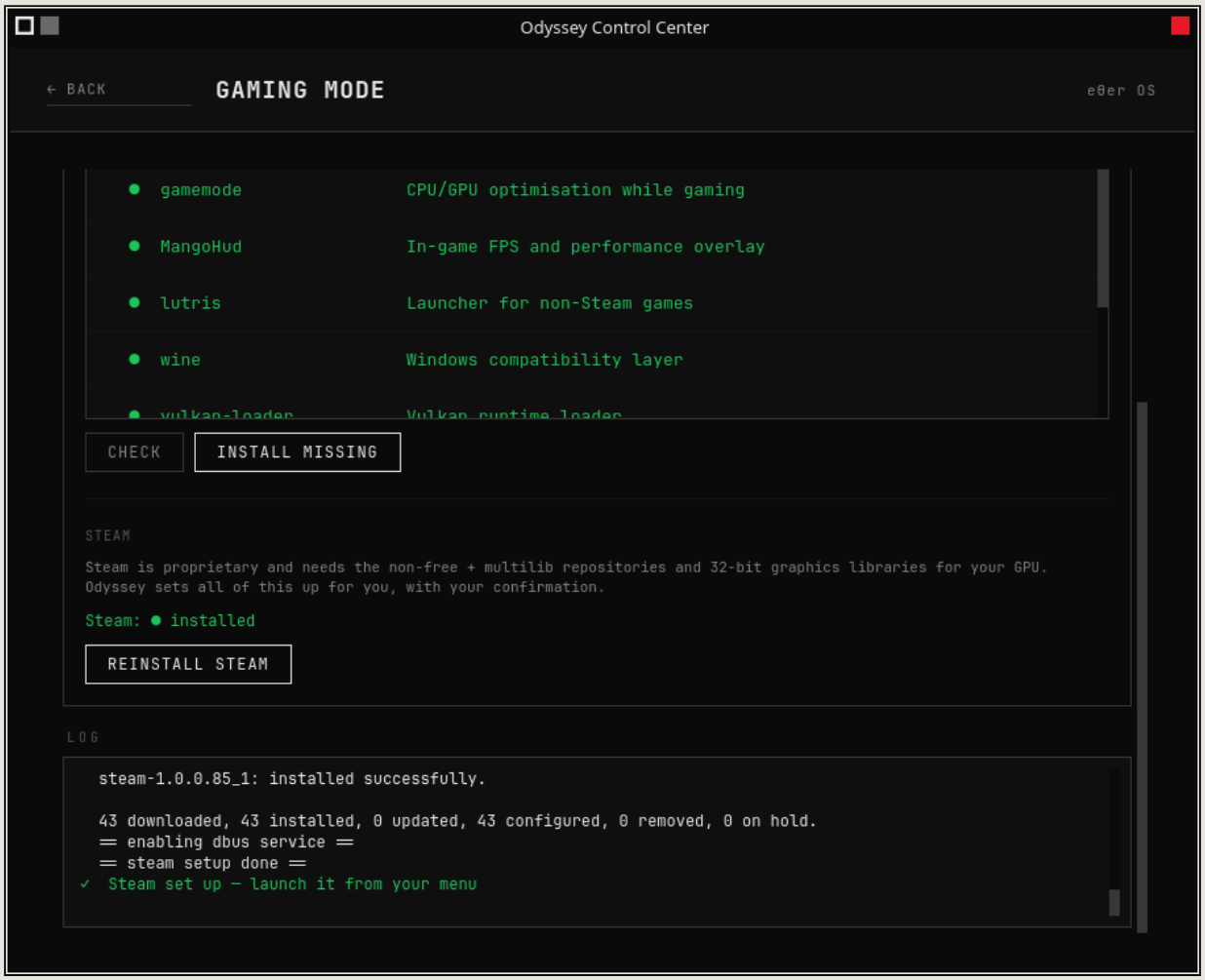

FIG · A aba PACKAGES depois da configuração: todos os pacotes em verde, Steam mostrando "● installed" com REINSTALL STEAM, e o log terminando em "Steam set up — launch it from your menu."

WHAT	Gaming Mode → aba PACKAGES → SET UP STEAM → revise a caixa de diálogo → Sí.
WHY	O Steam é proprietário e precisa dos repositórios non-free/multilib mais bibliotecas de 32 bits específicas para a GPU que o Void não habilita por padrão — fazer isso corretamente à mão é um obstáculo comum para quem está começando com um sistema baseado em Void.
RESULT	Os repositórios necessários, o próprio Steam, e as bibliotecas gráficas de 32 bits correspondentes à sua GPU específica, todos instalados — inicie o Steam pelo seu menu de aplicativos depois.
RISK	Baixo — isso só habilita repositórios e adiciona pacotes; nada existente é removido ou reconfigurado. A caixa de diálogo mostra a lista completa do que vai acontecer antes de você confirmar.

9.3 GameMode e MangoHud

GameMode (sua própria aba) é o daemon de desempenho por jogo da Feral Interactive — um jogo lançado através dele solicita um impulso temporário de desempenho (governor de CPU, prioridade de I/O) só para seu próprio processo, devolvido automaticamente quando o jogo fecha. **MangoHud** (sua própria aba) é a sobreposição na tela que mostra FPS, tempo de frame, carga de CPU/GPU e

temperaturas durante o jogo. Ambas as abas oferecem controles de instalação/verificação e, para o MangoHud, configuração rápida da aparência da sobreposição.

9.4 Proton

Gerenciamento da camada de compatibilidade para rodar jogos do Windows através do Steam — verificando o que está instalado e disponível, e instalando versões adicionais do Proton, incluindo builds da comunidade como o Proton-GE, quando um jogo específico precisa de uma que a Valve não distribui por padrão.

```
▸ ADVANCED

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

9.5 Advanced

O QUE ACTIVATE ESCREVE

Os mesmos arquivos subjacentes das ações MAKE PERSISTENT dos painéis Kernel e Memory — Gaming Mode é um pacote agrupado, não um mecanismo separado:

```
# valores sysctl tornados persistentes
$ cat /etc/sysctl.d/99-odyssey.conf
# CPU governor — não é sysctl, é baseado em sysfs, não é tornado persistente
# por esse mecanismo; reaplique depois de iniciar se você depender dele
$ cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

O QUE SET UP STEAM REALMENTE EXECUTA

A própria caixa de diálogo de confirmação é quase uma divulgação técnica completa, mas concretamente, em um sistema 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
# sistemas NVIDIA baixam adicionalmente nvidia-libs-32bit em vez de mesa-dri-32bit —
# o painel detecta o fabricante da sua GPU e ajusta a lista de pacotes de acordo
```

GAMEMODE E MANGOHUD MANUALMENTE

```
# lança qualquer jogo com GameMode ativo
$ gamemoderun ./game_binary
# ou, para um jogo Steam, opções de lançamento:
$ gamemoderun %command%
# sobreposição MangoHud, opções de lançamento:
$ mangohud %command%
# o próprio arquivo de configuração do MangoHud, para ajustar o que a sobreposição mostra
$ $EDITOR  /.config/MangoHud/MangoHud.conf
```

INSTALANDO UMA VERSÃO ESPECÍFICA DO PROTON-GE

A aba PROTON envolve o `protonplus` ou um gerenciador de versões equivalente — o equivalente manual é baixar uma release de `GloriousEggroll/proton-ge-custom` no GitHub e extraí-la no diretório de ferramentas de compatibilidade do Steam:

```
$ mkdir -p  /.steam/root/compatibilitytools.d
$ tar -xf GE-Proton-VERSION.tar.gz -C  /.steam/root/compatibilitytools.d
# depois selecione por jogo no Steam: Properties → Compatibility
```

CHAPTER 10

Telemetry

Apesar do nome, este painel não é uma configuração que você ajusta — é um monitor em tempo real. O Odyssey não coleta nem envia nada sobre você, então não há nada para desativar aqui. O que este painel faz em vez disso é mostrar a você, em tempo real, toda conexão de saída que o seu **próprio** sistema está fazendo — então "com quem minha máquina está falando agora" é algo a que você pode responder olhando, não confiando em uma afirmação. Ele se chama **Reverse Telemetry** justamente por isso: vira a observação do avesso, apontando-a para o próprio sistema, em seu nome.

10.1 Guia visual

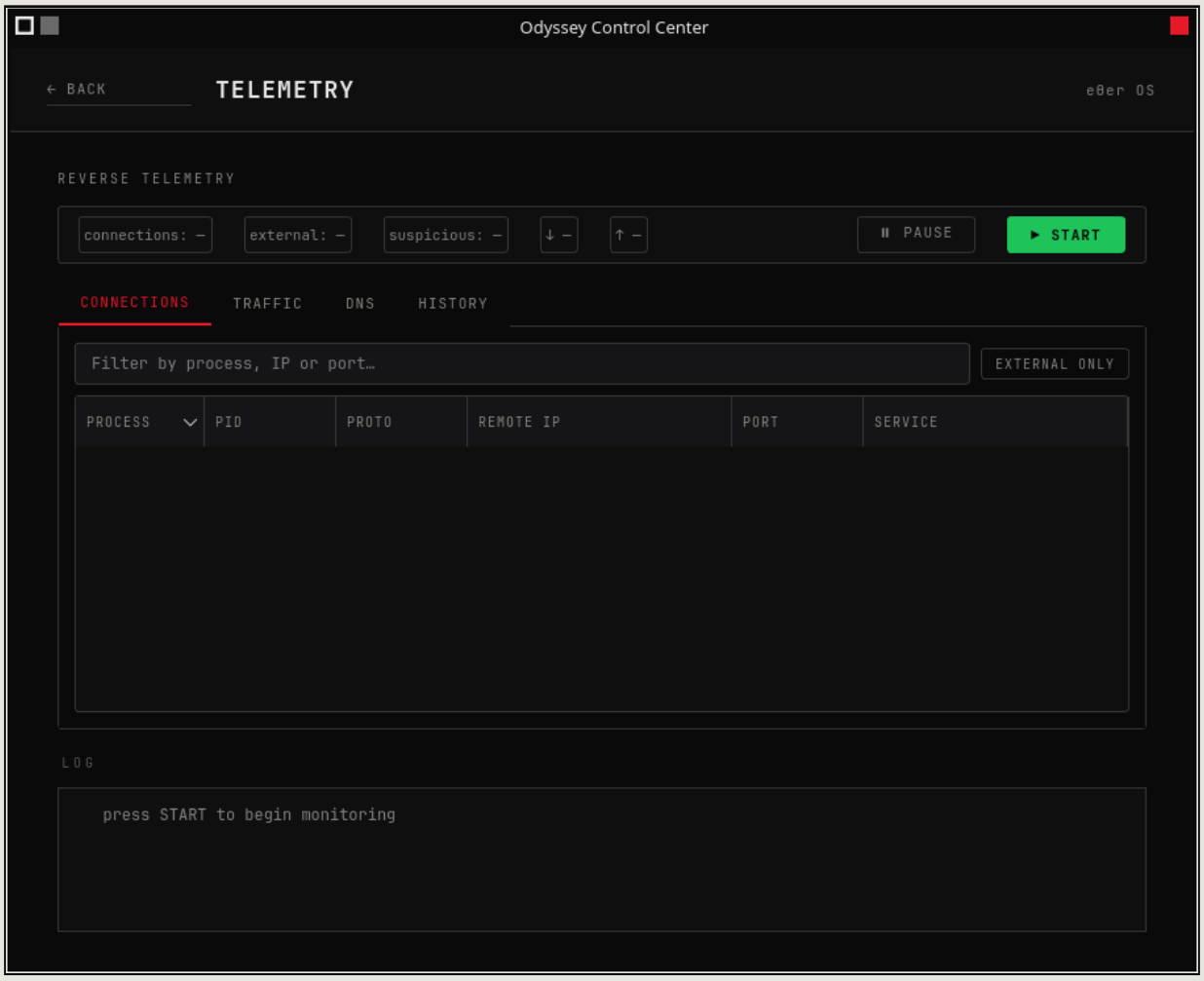

FIG · Antes de iniciar: connections/external/suspicious mostrando todos "–", um botão verde START, e o log dizendo "press START to begin monitoring." As quatro abas — CONNECTIONS, TRAFFIC, DNS, HISTORY — permanecem vazias acima de uma tabela não populada.

1 START

Inicia o monitoramento em tempo real — a barra de status e a aba ativa se populam imediatamente.

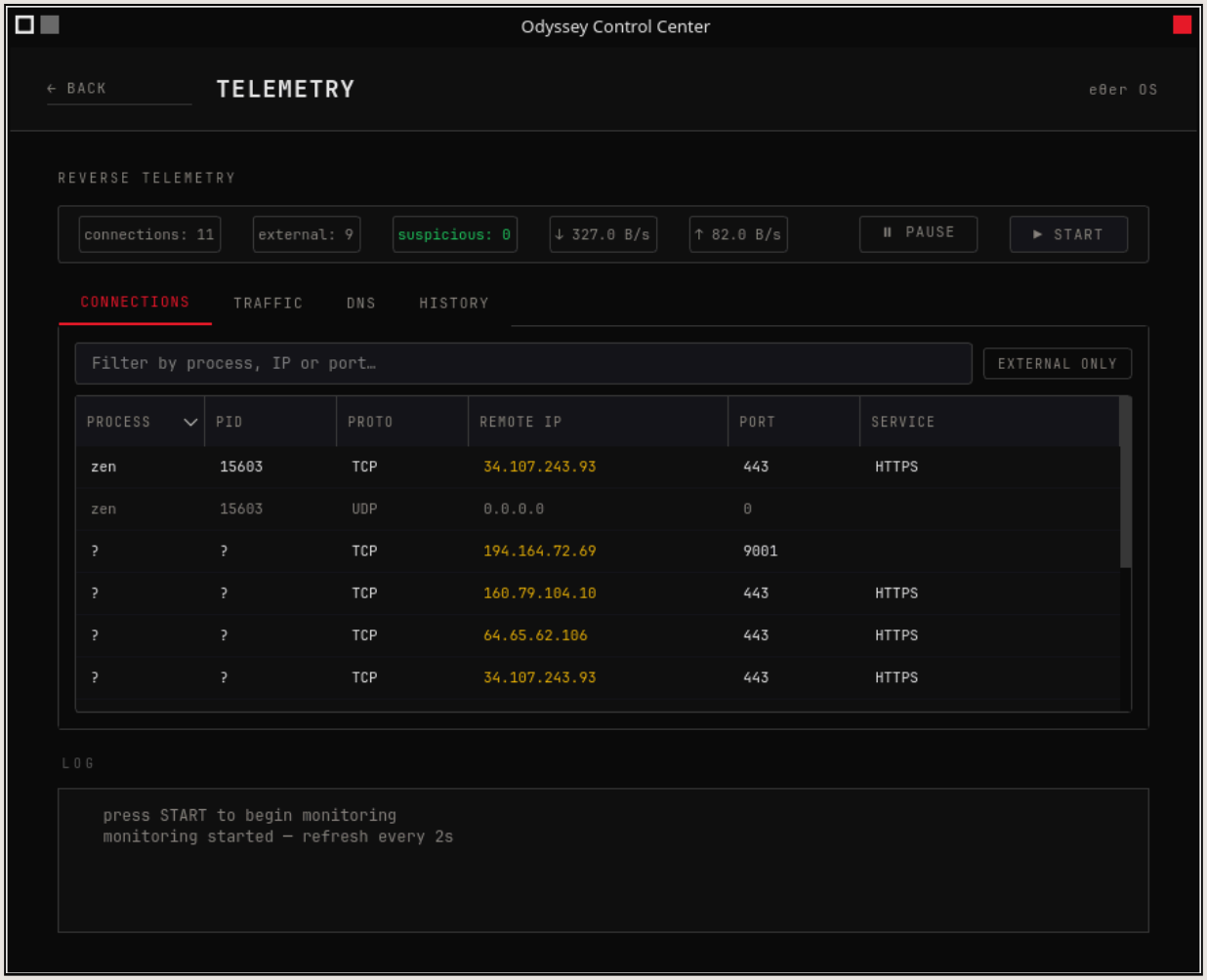

FIG · Depois de iniciar: “connections: 11 · external: 9 · suspicious: 0 · ↓327.0 B/s · ↑82.0 B/s,” PAUSE agora disponível, e a tabela CONNECTIONS cheia de linhas reais: processo “zen” em TCP para 34.107.243.93:443 (HTTPS), várias linhas marcadas “?” para processo/PID com conexões para outros endereços.

A barra de status no topo fornece totais em tempo real — conexões, externas (vs. locais), suspeitas, velocidades atuais de download/upload — e **PAUSE** congela a visualização sem perder o que foi capturado, **START** a retoma.

AS QUATRO ABAS

ABA	O QUE MOSTRA
CONNECTIONS	Uma linha por conexão ativa: processo, PID, protocolo, endereço remoto, porta, e o serviço conhecido para essa porta (443 → HTTPS, 22 → SSH). Ordenável por qualquer coluna.
TRAFFIC	Taxa de transferência por interface — velocidade de download/upload agora, mais totais acumulados desde o início do monitoramento.
DNS	Um fluxo em tempo real das resoluções de nomes de domínio à medida que acontecem — muitas vezes mais claro que um simples IP, já que um nome de host diz **o quê**, não só **onde**.
HISTORY	Tudo que a sessão capturou, como um log rolável — útil para pegar algo breve que já não é uma conexão ativa quando você olha.

Linhas onde o processo mostra `?` (visíveis na captura) significam que a conexão foi capturada mas o processo de origem não pôde ser resolvido a partir do PID naquele instante — comum em conexões muito curtas. Um campo de **filtro** acima da tabela restringe por processo, IP ou porta, e **EXTERNAL ONLY** esconde o tráfego puramente local.

WHAT	Telemetry → START → observe CONNECTIONS, ou ordene por PROCESS ou REMOTE IP para investigar algo específico.
WHY	É a resposta direta e concreta a "há algo nesta máquina alcançando a rede que não deveria" — lida diretamente das próprias tabelas de conexão do kernel, não da afirmação de um fornecedor sobre seu próprio software.
RESULT	Uma imagem ao vivo e precisa de toda conexão de saída, atualizada continuamente enquanto estiver em execução e não pausada.
RISK	Nenhum — inteiramente somente leitura.

Conexões de processos sem motivo comum para alcançar a rede por conta própria — um shell nu, um interpretador de scripts rodando de forma independente — são marcadas automaticamente sob "suspicious" na barra de status, para que uma anomalia se destaque em vez de ficar enterrada no tráfego normal do seu navegador e dos serviços do sistema.

▸ ADVANCED

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

10.2 Advanced

DE ONDE VÊM OS DADOS

O painel lê as mesmas tabelas de conexão em nível de kernel que qualquer ferramenta padrão leria, atualizadas em um intervalo curto enquanto o monitoramento está ativo — nenhum driver ou módulo de kernel especial próprio. Os equivalentes de linha de comando:

```
# todo socket aberto, com o processo dono — mesmos campos de CONNECTIONS
$ sudo ss -tupn
# taxa de transferência por interface — mesmos números de TRAFFIC
$ ip -s link
# uma visualização ao vivo pontual, se você a quiser fora do painel
$ sudo nethogs
```

RESOLUÇÕES DNS, MANUALMENTE

```
# captura ao vivo de consultas DNS de saída (precisa de uma ferramenta de captura de pacotes)
$ sudo tcpdump -i any -n port 53
```

POR QUE ALGUMAS LINHAS MOSTRAM PROCESS COMO "?"

Associar uma conexão ao vivo ao processo que a possui requer ler `/proc/PID/fd` para cada processo candidato no momento da captura — se a conexão abre e fecha mais rápido que o intervalo de

atualização do painel, ou o processo dono já terminou, o painel não tem a janela para fazer essa associação e mostra `?` em vez de adivinhar. É uma limitação de tempo, não um sinal de que algo está sendo escondido.

CHAPTER 11

About

Uma única ficha diagnóstica exaustiva da máquina, atualizada toda vez que você abre a aba, para sempre refletir o que mudou em outros painéis sem precisar reabrir o Control Center.

11.1 Guia visual

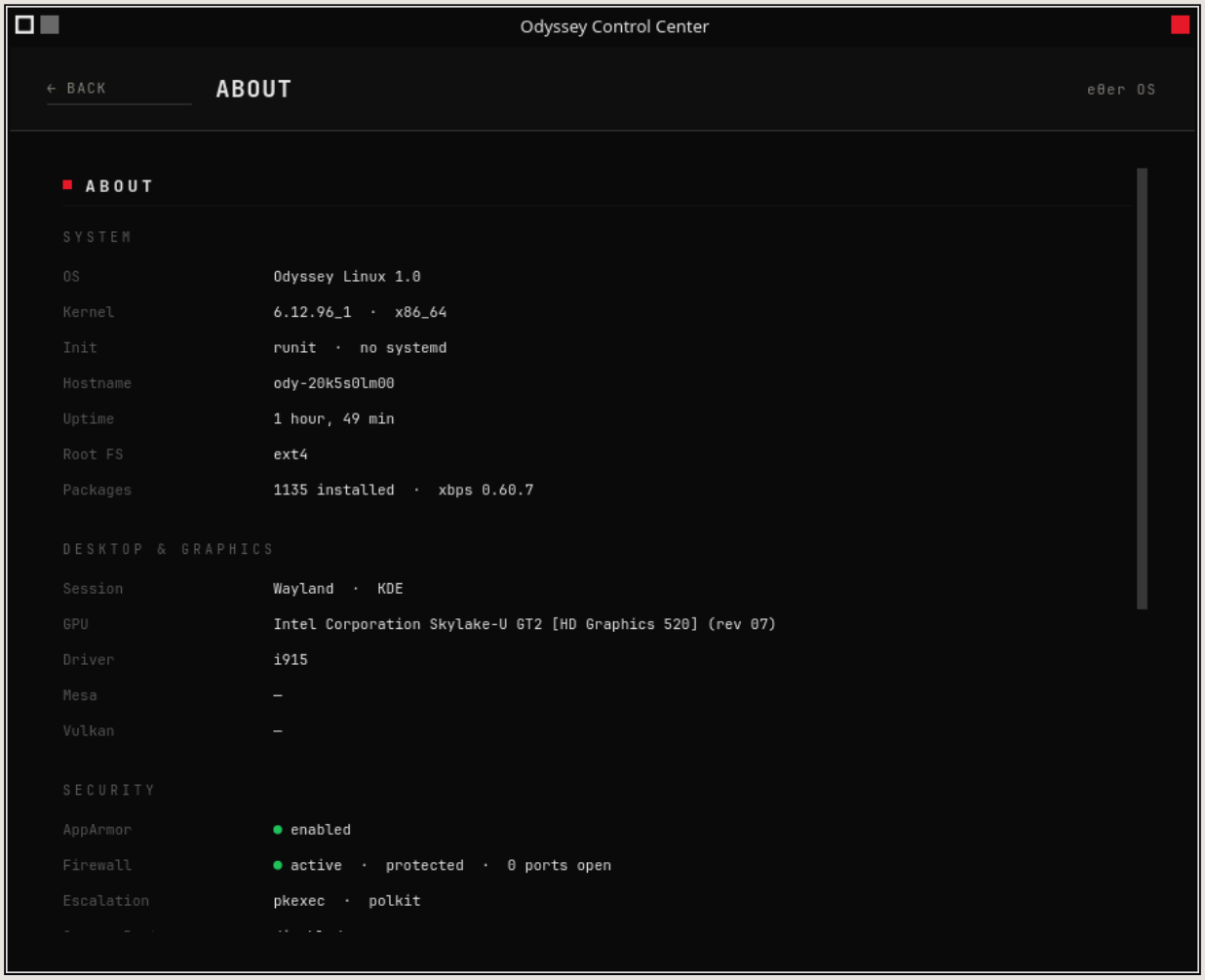

FIG · O painel rolado até o topo: seção SYSTEM mostrando OS "Odyssey Linux 1.0," Kernel 6.12.96_1 · x86_64, Init runit · no systemd, Hostname, Uptime, Root FS ext4, Packages "1135 installed · xbps 0.60.7"; DESKTOP & GRAPHICS abaixo com Session, GPU, Driver, Mesa, Vulkan; SECURITY começando a mostrar AppArmor "● enabled" e Firewall "● active · protected · 0 ports open."

Três seções, em ordem: **SYSTEM** (kernel, init, hostname, uptime, sistema de arquivos, contagem de pacotes), **DESKTOP & GRAPHICS** (tipo de sessão, GPU, driver, suporte a Mesa/Vulkan), e mais abaixo **SECURITY** (status do AppArmor e Firewall com indicadores coloridos, não só texto — verde para habilitado/ativo) e **HARDWARE** (placa-mãe, áudio).

WHAT	About → role pelas três seções para verificar qualquer valor — versão do kernel, driver em uso, se AppArmor e o firewall estão realmente ativos.
WHY	É a forma mais rápida de confirmar o status de todo o sistema em um só lugar, sem abrir cinco painéis diferentes.
RESULT	Um instantâneo completo e atual — atualizado a cada visita à aba, então nunca mostra informação desatualizada de uma sessão anterior.
RISK	Nenhum — inteiramente somente leitura.

11.2 Copy report

Um único botão no topo do painel, **COPY REPORT**, coloca toda a ficha na área de transferência como texto puro — cada seção, cada valor — pronta para ser colada diretamente em um post do fórum ou em um relatório no rastreador de bugs.

WHAT	About → COPY REPORT → cole no seu relatório de bug ou post do fórum.
WHY	Relatórios de bug são úteis só na medida dos detalhes do sistema anexados a eles, e digitar à mão a versão do kernel, o driver, o status do AppArmor e uma dúzia de outros campos é exatamente o tipo de atrito que faz muita gente pular essa etapa.
RESULT	Um relatório de sistema completo e corretamente formatado na área de transferência, pronto para colar — as mesmas informações que um mantenedor teria que pedir a você campo por campo de outra forma.
RISK	Nenhum para o seu sistema. O relatório é informação diagnóstica (versões, estados, modelo de hardware) — não inclui arquivos pessoais nem credenciais — mas como em qualquer despejo diagnóstico, releia antes de publicar em algum lugar público se não tiver certeza do que ele contém.

▸ ADVANCED

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

11.3 Advanced

DE ONDE VEM CADA CAMPO

Nada neste painel é inventado nem cacheado desde o momento da instalação — cada valor é consultado ao vivo, toda vez que a aba é aberta:

```
$ 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
```

Seção Security especificamente:

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

POR QUE VALE A PENA VERIFICAR ABOUT PRIMEIRO

Como ele consulta o sistema de novo a cada visita, About é a forma mais rápida de confirmar que uma mudança feita em outro painel realmente teve efeito — aplique um modo de firewall em Firewall, depois verifique a seção Security de About em vez de reabrir Firewall de novo. Ele é construído para ser a verificação cruzada “isso realmente funcionou?” para o resto do Control Center.

CHAPTER 12

NVIDIA driver

PENDENTE DE INTEGRAÇÃO

Este capítulo é um espaço reservado. O módulo NVIDIA só aparece no Control Center em máquinas com hardware NVIDIA detectado, e as capturas de tela para ele só podem ser obtidas em uma máquina desse tipo — serão fornecidas separadamente e este capítulo será completado em uma futura revisão deste guia. Ele é mantido deliberadamente por último, depois de About, justamente para que adicioná-lo depois não desloque a numeração de páginas nem as referências de seção de nenhum outro capítulo.

O que já se sabe e vai ancorar o capítulo terminado: o painel traz um selo **BETA** visível e uma nota inicial de que o próprio driver é proprietário — empacotado pelo Void a partir do instalador oficial da NVIDIA com uma soma de verificação fixada, mas não auditável de forma independente, a mesma honestidade que o resto do Odyssey aplica a si mesmo. Ele detecta sua GPU específica comparando-a com um banco de dados embutido e escolhe automaticamente o branch de driver correto (current/Turing-e-mais-recente, ou um dos três branches legados para placas mais antigas, recorrendo ao driver open source `nouveau` quando nada proprietário se aplica). A instalação roda como um pipeline rastreado em sete etapas com um indicador de progresso visível, e o painel imprime seu próprio comando de recuperação com antecedência — `odyssey-nvidia --revert` a partir de um console de texto — antes mesmo de você clicar em instalar. Em notebooks híbridos, uma seção PRIME render-offload aparece automaticamente, permitindo que um jogo ou programa específico rode na GPU NVIDIA enquanto tudo mais permanece na GPU integrada para a autonomia da bateria.