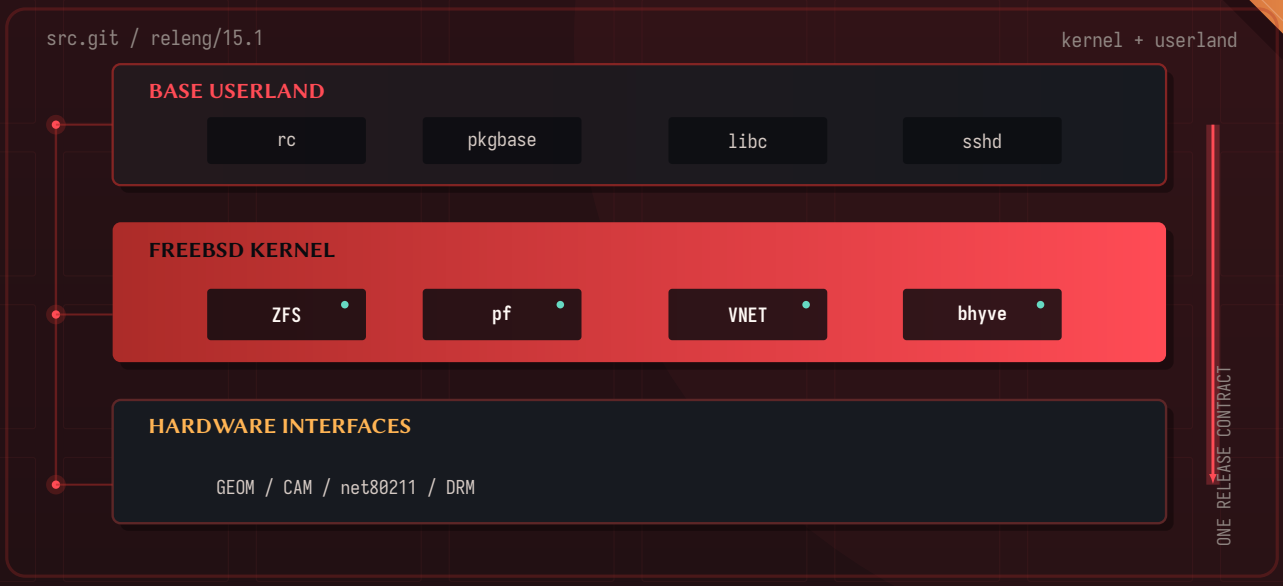
Covers FreeBSD 15



FreeBSD

for Linux Professionals

The Complete Guide to FreeBSD 15.1

for SRE and DevOps Practitioners

Wolfgang Kerschbaumer

SYSINIT PRESS

FreeBSD for Linux Professionals

Preface and First Chapter

Wolfgang Kerschbaumer

Contents

Preface **1**

Complete edition outline **3**

Front . . . 3
Part1 Orientation . . . 3
Part2 First Boot . . . 3
Part3 Storage . . . 3
Part4 Networking . . . 3
Part5 Compartments . . . 4
Part6 Production . . . 4
Part7 Under The Hood . . . 4
Part8 Ecosystem . . . 4
Part9 Appendices . . . 4

1 One System, Whole: The FreeBSD Model **5**

1.1 One artifact, not an assembly . . . 5
1.2 Kernel and userland, provably in step . . . 7
1.3 Batteries included, quiet by default . . . 8
1.4 What the 15.x line changed . . . 8
1.5 pkgbase: the base system becomes packages . . . 9
1.6 Release engineering and the four-year clock . . . 11
1.7 Where FreeBSD is the wrong tool . . . 13
1.8 How this book works . . . 13
1.9 Takeaways . . . 14

Preface

This book is about operating FreeBSD as a production system. It starts from one premise: FreeBSD is not a Linux distribution with different commands. Its kernel, base userland, service framework, network stack, release engineering, and documentation form one operating-system project. That changes how you install it, patch it, debug it, automate it, and decide whether it belongs in a fleet.

The version baseline is FreeBSD 15.1-RELEASE, with transcripts verified on the lab at patch levels p1 and p2. The main text documents that release as it exists, including pkgbase, the 15.x network defaults, the selectable scheduler framework, the current jail and bhyve interfaces, and the base-system software shipped with it. Migration notes identify older practice when it remains operationally relevant. Features that exist only in 16-CURRENT never appear here as 15.1 behavior.

This is version 1.0, prepared on November 5, 2026. The manuscript was rechecked against the security advisories and errata notices published through 15.1-RELEASE-p6 (October 21, 2026), and the lab host runs that patch level.

Who this book is for. You operate Linux or another Unix at depth, and you know systemd, namespaces, cgroups, nftables, LVM, containers, CI pipelines, and incident response. You do not need an introduction to virtual memory or DNS. You do need an exact account of where a jail differs from a container, why a loader tunable is not a sysctl, how rc ordering works without unit files, and which parts of a pkgbase upgrade remain manual.

What makes this book different. Four rules shape the material:

1. *Source-grounded.* Claims are checked against the FreeBSD 15.1 source tree and the current ports tree. The Handbook and release documentation provide context, but executable code, defaults, build files, and manual pages in the source tree settle technical behavior.
2. *Operational.* Chapters include runnable examples, failure modes, rollback paths, and the state you should inspect before changing a production host. Commands are not substitutes for understanding the subsystem.
3. *Verified where labeled.* Captured output marked as verified came from the FreeBSD 15.1 lab described in *The Lab: FreeBSD 15.1 on Proxmox with OpenTofu*. Draft output is marked for verification rather than invented.
4. *Honest about the Linux comparison.* FreeBSD sometimes has a cleaner whole-system contract. Linux often has broader hardware support, a larger orchestration ecosystem, or a more mature implementation of a given workload. The useful question is which system makes your constraints cheaper.

How the book is organized. Part I replaces Linux-shaped assumptions with a FreeBSD model and builds the reference lab. Parts II through IV move from installation and first boot into storage and networking. Part V covers jails and bhyve. Part VI treats updates, interoperability, high availability, security, observability, and performance as production disciplines. Part VII descends into the kernel, ports, source builds, and the developer toolchain. Part VIII closes with Linux ABI compatibility, hardware,

the project itself, and an adoption playbook. Use the appendices during incidents and migrations rather than reading them in sequence.

Conventions. An unprivileged shell uses a `$` prompt; a root shell uses `#`. Examples assume the base `sh(1)` unless they explicitly name another shell. Paths under `/usr/local` belong to installed packages and local administration; paths under `/etc`, `/boot`, and `/usr/src` usually belong to the base system. Version, warning, and Linux-comparison callouts identify details that deserve a context switch. Every chapter ends with takeaways, and non-trivial examples also ship as files in the accompanying examples archive.

Reporting an error. Send error reports to errata@sysinit-press.com. Include the book version, the chapter and section title, the command or instruction, the result you expected, and the result you observed. Add the FreeBSD patch level and the relevant package versions. Remove keys, passwords, and private data from logs and transcripts before you send them.

Read FreeBSD as a coherent artifact. Start from the sources, make state visible before you change it, and test the rollback path. Keep the Linux analogy only while it remains accurate.

Wolfgang Kerschbaumer, November 2026. sysinit.at

Complete edition outline

Front

- Front matter: Colophon
- Front matter: Preface
- Front matter: Version History

Part1 Orientation

- Chapter 1: One System, Whole: The FreeBSD Model
- Chapter 2: The Translation Table: FreeBSD for Linux Eyes
- Chapter 3: The Lab: FreeBSD 15.1 on Proxmox with OpenTofu

Part2 First Boot

- Chapter 4: Installing FreeBSD: From bsdinstall to Unattended
- Chapter 5: The Boot Chain: Firmware to /sbin/init
- Chapter 6: Init, rc, and Service Management
- Chapter 7: Software Management with pkg
- Chapter 8: Users, Authentication, and Login Classes
- Chapter 9: Batteries Included: The Base-System Daemons

Part3 Storage

- Chapter 10: GEOM and UFS: The Traditional Storage Stack
- Chapter 11: ZFS Fundamentals
- Chapter 12: ZFS in Operation
- Chapter 13: Backup and Disaster Recovery

Part4 Networking

- Chapter 14: The Network Stack: Links, Routes, and TCP
- Chapter 15: Plugging In: DHCP, IPv6, and Name Resolution
- Chapter 16: Firewalls: pf, ipfw, and Traffic Shaping

Part5 Compartments

- Chapter 17: Jails: The Model
- Chapter 18: Jails in Production
- Chapter 19: bhyve: Virtual Machines

Part6 Production

- Chapter 20: Staying Current: pkgbase, freebsd-update, and Security Advisories
- Chapter 21: Cross-OS Interop: SMB, NFS with Linux, and Foreign Disks
- Chapter 22: High Availability: CARP, pfsync, and HAST
- Chapter 23: Security Architecture and Hardening
- Chapter 24: Observability: DTrace, Tracing, and Crash Dumps
- Chapter 25: Performance Tuning

Part7 Under The Hood

- Chapter 26: The Kernel: GENERIC, Modules, and sysctl
- Chapter 27: Building Your Own: Ports and poudriere
- Chapter 28: Building FreeBSD from Source
- Chapter 29: The Developer's FreeBSD: Toolchain and Contributing

Part8 Ecosystem

- Chapter 30: The Linuxulator: Running Linux Binaries
- Chapter 31: FreeBSD on Your Hardware: Wi-Fi, Graphics, and the Desktop
- Chapter 32: The Project: Governance, Releases, and the Adoption Playbook

Part9 Appendices

- Appendix A: Linux-to-FreeBSD Command and Concept Reference
- Appendix B: Troubleshooting Playbook
- Appendix C: Configuration File Reference
- Appendix D: Removed, Renamed, and Deprecated in 15.x
- Appendix E: Glossary
- Appendix F: Sources and Further Reading
- Appendix G: Subject Index

1 One System, Whole: The FreeBSD Model

FreeBSD is not a Linux distribution, and treating it like one is the most common mistake experienced Linux engineers make with it. It is one operating system whose kernel, userland, and network daemons version and patch together. The rest of the chapter works through that model, the changes the 15.x line brought, the support lifecycle, and the workloads where FreeBSD is a poor fit. Every fact comes from the pinned `releng/15.1` source tree, and every transcript targets one reproducible lab VM.

1.1 One artifact, not an assembly

A Linux operating system is a federation. The kernel comes from kernel.org, carrying distro patches; glibc, coreutils, util-linux, systemd, and OpenSSH each have their own upstream, their own version number, and their own release rhythm. What turns that federation into "Debian 13" or "RHEL 10" is the distributor: it freezes a set of versions, patches them into agreement, and takes on the QA of the combination. Version questions about a Linux box therefore have compound answers: kernel 6.12, glibc 2.41, and systemd 257. The distributor's test matrix is the only contract binding them.

FreeBSD has no federation to manage. The kernel, the C library, the userland from `ls` to `dtrace`, the base network daemons, and the init framework are developed in a single Git repository (`src.git`), reviewed by one project, and released and patched as one artifact. The in-tree manual page development(7) describes the model: all changes land on the `main` branch (CURRENT) first and are merged back to `stable/N` branches, from which frozen `releng/N.M` release branches are cut. There is no "which distribution" question; FreeBSD answers it itself.

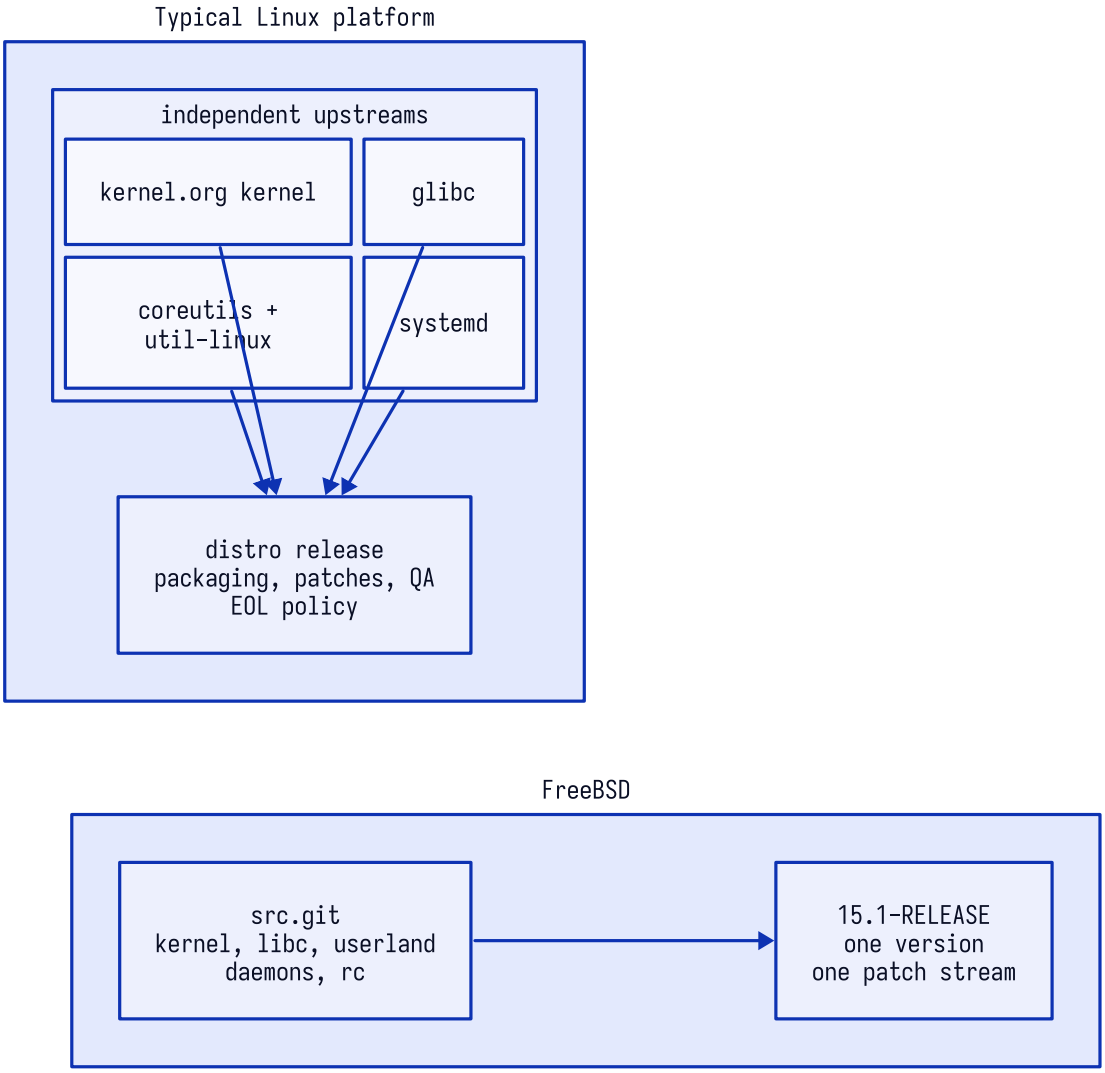


Figure 1.1: A Linux platform is an assembly; FreeBSD is one versioned artifact

That single artifact has concrete consequences:

- **No internal compatibility matrix.** Userland 15.1 runs on kernel 15.1 because both are the same commit. Cross-version tolerance exists only where deliberately engineered (the `COMPAT_FREEBSD14` kernel option carries 14.x binaries), not as an emergent property of distro QA.
- **One changelog.** `UPDATING` and `RELNOTES` at the top of the source tree record every incompatible change and notable feature for the branch. There is no aggregation of a dozen upstream NEWS files.
- **Coherent API changes.** When 15.0 added the native inotify(2) family, the syscalls, the libc wrappers, and the Linux-emulation wiring shipped in the same release. The equivalent change on Linux crosses kernel, glibc, and distro boundaries on three different clocks.
- **One security patch stream for the OS.** A FreeBSD Security Advisory patches kernel and userland together and bumps one patch level (`15.1-RELEASE-p1`), whichever side the bug was on.
- **A sharp third-party boundary.** Everything that is not base lives under `/usr/local`, installed by pkg from a separate repository on a separate release cadence. Base configuration lives in `/etc`; third-party configuration in `/usr/local/etc`. The filesystem itself tells you who owns a file.

- **Manual pages live next to the code.** ifconfig(8)'s source and its man page sit in the same directory of the same repository and change in the same commit, which is why this book cites in-tree man pages as primary sources.

  Linux: FreeBSD's base system is roughly what you would get if kernel.org, glibc, coreutils, systemd, and OpenSSH shared one repository, one release manager, and one security team. The ports collection is the part that behaves like a distro package archive, see *Software Management with pkg*.

The systematic Linux-to-FreeBSD mapping (init, packaging, storage, firewalls, tracing, and filesystem layout) lives in *The Translation Table: FreeBSD for Linux Eyes*. Later chapters point at it instead of repeating it.

1.2 Kernel and userland, provably in step

One command checks that kernel and userland version together, and it replaces the three-source audit the same question needs on Linux. The freebsd-version(1) tool reports three things: the version of the kernel installed on disk (`-k`), the version of the kernel that is running (`-r`), and the version of the installed userland (`-u`).

On the lab VM this book uses (a stock 15.1 cloud image, patched to p1):

```
$ freebsd-version -kru
15.1-RELEASE-p1
15.1-RELEASE-p1
15.1-RELEASE-p1
```

All three agree, and they match what the kernel says about itself. The same box reports `15.1-RELEASE-p1 releng/15.1-n283582-0f691888dc56 GENERIC amd64` in `uname -a`: release, branch, build metadata, kernel configuration, and architecture in one string. `uname -K` and `uname -U` print the same information as numeric `__FreeBSD_version` values, for the kernel and userland respectively. Scripts can compare those numbers directly.

The three values diverge in two ordinary situations:

- `-k` differs from `-r` when a new kernel has been installed but the machine has not rebooted yet. This is the standard state mid-upgrade.
- `-u` differs from `-r` after a userland-only security patch. `uname -r` reflects only the running kernel, so a monitoring check that scrapes `uname` will under-report the patch level of a system that took a userland-only advisory. Fleet audits belong on `freebsd-version -kru`, not `uname`.

  Linux: the closest habit is correlating `uname -r`, `/etc/os-release`, and `dpkg -l linux-image-\* libc6` (or `rpm -q kernel glibc`), three sources with no contract between them. FreeBSD generates `/etc/os-release` too, at boot, from `freebsd-version -u`. The file is a symlink to `/var/run/os-release`, written by the `os-release` rc script, so tooling that parses os-release works unchanged.

`examples/ch01-why-freebsd/version-coherence.sh` runs the sequence and ends with the probe that tells you which update plane owns the base system (the pkgbase section explains that distinction):

```
$ pkg which /usr/bin/uname
/usr/bin/uname was installed by package FreeBSD-runtime-15.1p2
```

On a pkgbase install, base files are registered with pkg and the query names the owning package. On a traditional distribution-set install, pkg knows nothing about `/usr/bin/uname` and freebsd-update(8) owns the file. Base packages carry the patch level in pkg's spelling, so `15.1p2` is `15.1-RELEASE-p2`. This capture postdates the baseline transcript above, which the lab VM produced before it moved from p1 to p2.

1.3 Batteries included, quiet by default

Because the daemons ship in the same tree as the kernel, a fresh FreeBSD install carries a complete network-service stack with no packages installed: OpenSSH 10.0p2 (with FreeBSD-specific compiled-in defaults, password authentication and root login both off), ntpd 4.2.8p18, a caching DNSSEC-validating resolver (`local-unbound`, Unbound 1.25.1), a Capsicum-sandboxed syslogd, the dma mail agent for local and smarthost delivery, an in-kernel NFS server up to NFSv4.2, an iSCSI and NVMe-over-Fabrics target daemon (ctld), CARP failover, and certctl for the TLS trust store. All of it is versioned with the OS and patched through the same advisory stream.

Two aspects of this posture differ from common Linux defaults. First, almost none of it runs on a stock 15.1 install: the only network-aware daemon enabled by default is syslogd, and it starts with `-s`, which discards remote messages. Everything else (sshd, ntpd, NFS, the resolver) is one `foo_enable="YES"` line away, but off until you ask. Second, the defaults are opinionated where Linux distros are permissive: sshd compiles in `PasswordAuthentication no` and `PermitRootLogin no`, and the NFS server, when enabled, speaks NFSv3 only and rejects unprivileged source ports unless you configure it otherwise.

The full tour (each daemon, its rc.conf knobs, and its Linux counterpart) is *Batteries Included: The Base-System Daemons*; service management itself is *Init, rc, and Service Management*. On FreeBSD, "the OS" includes the service layer that on Linux arrives as distro-selected packages (openssh-server, chrony, rsyslog, nfs-utils) with distro-chosen defaults.

1.4 What the 15.x line changed

This book targets FreeBSD 15.1-RELEASE exactly. Several 15.x changes invalidate assumptions formed on earlier releases. The chapter that owns each change covers it in full.

FreeBSD 15.0 (released December 2, 2025):

- **pkgbase.** The base system became installable and upgradable through pkg(8). The next section and *Staying Current: pkgbase, freebsd-update, and Security Advisories* cover it.
- **armv6 retirement.** armv6 is gone. i386 remains buildable and 15.1 publishes VM images, though the platform is unsupported; 32-bit powerpc kernel configurations also remain. Running 32-bit binaries on 64-bit kernels (`COMPAT_FREEBSD32`, `/usr/lib32`) is still supported.
- **MIT Kerberos.** MIT krb5 replaced Heimdal as the base Kerberos (version 1.22.2 as of 15.1), the same implementation Linux distros ship, so `krb5.conf` knowledge now transfers directly.
- **Native inotify(2)**, API-compatible with Linux via libc wrappers.
- **Kernel TLS on by default** (`kern.ipc.tls.enable=1`), integrated with sendfile and base OpenSSL.

- **Hardened network defaults** that break Linux-formed and 14.x-formed habits alike: TIME_WAIT is no longer skipped for local connections, `connect()` to 0.0.0.0 is refused, IP addresses on bridge member interfaces are deprecated, and the ipfw kernel/userland ABI changed. The operational details are in *The Network Stack: Links, Routes, and TCP* and *Firewalls: pf, ipfw, and Traffic Shaping*.
- **Current crypto userland**: OpenSSH 10.0p2 and OpenSSL 3.5 LTS in base.

FreeBSD 15.1 (released June 16, 2026):

- **Boot-time scheduler selection**: the `kern.sched.name` loader tunable picks between ULE (default) and 4BSD, both compiled into amd64 GENERIC; `kern.sched.available` lists the choices.
- **Scripted installs default to pkgbase**, and disc1 media carry an offline base-package repository. See *Installing FreeBSD: From bsdinstall to Unattended*.
- **root's shell is now sh(1)** in release images, ending the csh era for the superuser.
- **pf gained NAT64** (`af-to`), ipfw gained masked table lookups, and RFC 7217 stable IPv6 addresses arrived as an ifconfig flag.
- **Fresh removal notices** for 16.0: the lpr suite, bsdlabel and fdisk, and several legacy drivers.

  Version: if you are coming from 14.x, treat every "it worked on 14" assumption as suspect until checked. Appendix D, *Removed, Renamed, and Deprecated in 15.x*, is the consolidated catalog of what is gone, renamed, or on notice.

1.5 pkgbase: the base system becomes packages

The single-artifact story historically had an awkward corner: the OS itself was delivered as tar archives ("distribution sets") and patched by a dedicated tool (freebsd-update), while everything under `/usr/local` was pkg's business. FreeBSD 15 begins closing that gap. With pkgbase, the base system is built as several hundred fine-grained packages: `FreeBSD-runtime`, `FreeBSD-ssh`, `FreeBSD-zfs`, and so on. Every one has an origin starting in `base/` and comes from a repository named `FreeBSD-base`, managed by the same pkg(8) as third-party software. The freebsd-base(7) manual describes the layout. Packages split into subpackages such as `-lib`, `-dev`, `-dbg`, `-lib32`, `-dev-lib32`, and optionally `-man`. `FreeBSD-set-*` meta packages pull in workloads such as `minimal`, `minimal-jail`, `base`, `devel`, `optional`, `optional-jail`, `lib32`, `base-jail`, `kernels`, `src`, and `tests`.

Where 15.1 stands, precisely:

- pkgbase is a **technology preview**, but a heavily used one: all official VM and cloud images are pkgbase-managed, `bsdinstall script` performs pkgbase installs by default in 15.1, and the interactive installer offers "Packages (Tech Preview)" next to "Distribution Sets".
- Distribution sets and freebsd-update remain fully supported for the life of stable/15 and are **planned for removal in FreeBSD 16**.
- The two planes are mutually exclusive per install, and `pkg which /usr/bin/uname` is the discriminator. On pkgbase systems, `make installworld` is blocked to protect the package database, and upgrading a 14.x pkgbase experiment to 15 is unsupported. Reinstall instead.
- The repository definition ships in `/etc/pkg/FreeBSD.conf`, part of the `FreeBSD-pkg-bootstrap` package, and is **disabled by default on every install**, pkgbase ones included:

```
FreeBSD-base: {
  url: "pkg+https://pkg.FreeBSD.org/${ABI}/base_release_${VERSION_MINOR}",
  mirror_type: "srv",
  signature_type: "fingerprints",
  fingerprints: "/usr/share/keys/pkgbase-${VERSION_MAJOR}",
  enabled: no
}
```

Do not edit that file. It belongs to base, and upgrades replace it. Overrides go in a drop-in:

```
# mkdir -p /usr/local/etc/pkg/repos
# echo 'FreeBSD-base: { enabled: yes }' > /usr/local/etc/pkg/repos/FreeBSD-base.conf
```

pkg.conf(5) makes the file name in that directory arbitrary as long as it ends in `.conf`. The official 15.1 cloud images ship this override as `/usr/local/etc/pkg/repos/FreeBSD.conf`, which is why `FreeBSD-base` reads as enabled on the lab VM while the file in `/etc` still says otherwise.

The lab VM, built from the official 15.1 cloud image, shows what pkgbase looks like in practice. The baseline survey counts 553 installed packages, nearly all of them `FreeBSD-*` base packages, and the image preinstalls thirteen set meta packages:

```
$ pkg query '%n-%v' | grep '^FreeBSD-set-'
FreeBSD-set-base-15.1
FreeBSD-set-base-dbg-15.1
FreeBSD-set-devel-15.1
FreeBSD-set-devel-dbg-15.1
FreeBSD-set-kernels-15.1
FreeBSD-set-kernels-dbg-15.1
FreeBSD-set-lib32-15.1
FreeBSD-set-lib32-dbg-15.1
FreeBSD-set-minimal-15.1
FreeBSD-set-minimal-dbg-15.1
FreeBSD-set-optional-15.1
FreeBSD-set-optional-dbg-15.1
FreeBSD-set-tests-15.1
```

`examples/ch01-why-freebsd/base-inventory.sh` reports the full inventory: configured repositories, base packages by `base/` origin, installed sets, the patch-level split, and the third-party remainder. The script's central one-liner is useful on its own, because it draws the base/ports boundary on any pkgbase machine:

```
$ pkg query '%o %n-%v' | awk '$1 ~ /^base\//' | wc -l
```

Fine-grained packaging also changes what a security advisory costs. When the lab VM took the advisory batch after p1, pkg replaced only the base packages the fix touched; the rest kept their plain `15.1` version. `freebsd-version` still reports one system-wide patch level, because `/bin/freebsd-version` ships in `FreeBSD-runtime`, one of the packages that moved:

```
$ pkg query '%o-%v' | grep -c '^base/.*p2$'
48
$ pkg query '%o-%v' | grep -c '^base/'
499
$ freebsd-version -u
15.1-RELEASE-p2
```

freebsd-update(8) has no equivalent: it patches the files an advisory names and moves the whole install to the new `-pN`. Under pkgbase the package database names the 48 packages the advisory replaced.

Since 15.1, cloud images also self-update their base packages on first boot (the lab image ships a `firstboot_pkg_upgrade` hook pointed at the `FreeBSD-base` repository), which is how a freshly provisioned VM comes up already at the current patch level. Neither update plane touches the boot loader on the EFI system partition. Copying a new `/boot/loader.efi` there is a documented manual step covered in *Staying Current: pkgbase, freebsd-update, and Security Advisories*.

1.6 Release engineering and the four-year clock

An adopting team needs the lifecycle facts more than the governance story (which lives in *The Project: Governance, Releases, and the Adoption Playbook*). Development happens on `main` (today 16.0-CURRENT). Changes soak there for a minimum of three days by convention before they are merged ("MFC") to `stable/15`. Releases are branched off as `releng/15.1` and frozen forever, after which only the Security Officer lands changes on them.

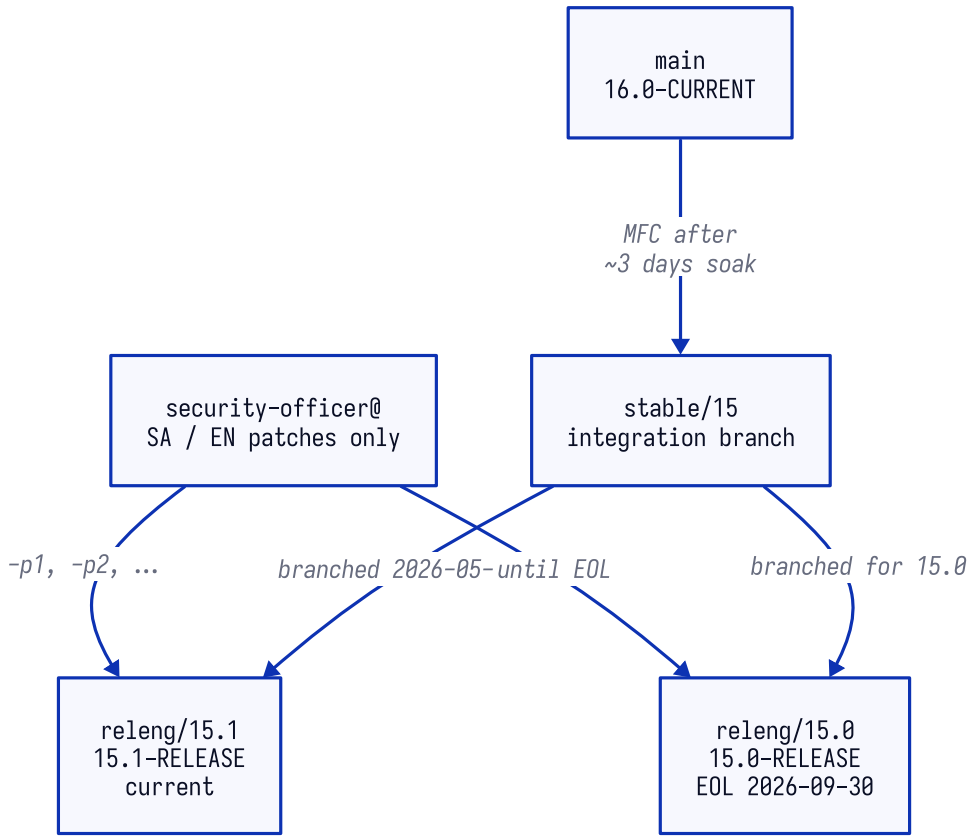


Figure 1.2: src branches: all work lands in main; releases freeze on releng

> **Warning:** FreeBSD "-STABLE" is not stable in the Debian or kernel.org sense. `stable/15` is a rolling development branch with a stable ABI. The Handbook itself says not to run it on production systems without your own testing. Production runs RELEASE plus patch levels.

Major releases are cut from `main` roughly every two years; point releases from the stable branch roughly every six months (15.2 is scheduled for December 2026, 16.0 for December 2027). Support is branch-based, and since FreeBSD 15 the terms are explicit. Each stable branch is supported for four years from its dot-zero release, and each point release is supported only until three months after the next one ships.

Branch or release	Released	Support ends
15.1-RELEASE	2026-06-16	2027-03-31
15.0-RELEASE	2025-12-02	2026-09-30
stable/15 series	2025-12-02	2029-12-31
stable/14 series	2023-11-20	2028-11-30

There is no ten-year RHEL-style horizon and no extended support for old minors: staying on 15.0 past September 2026 means no more security patches even though 15.x itself lives until the end of 2029. The operational contract is "take the point release every six months, plan a major upgrade every two to four years." Patches between point releases arrive as Security Advisories (SAs) and Errata Notices (ENs), each of which bumps the `-pN` patch level; the pinned 15.1-p1 rolls up the advisory batch of June 30, 2026.

Base and ports also have separate security streams. SAs cover only the base system, while third-party package vulnerabilities flow through the VuXML database that `pkg audit` consults. *Staying Current: pkgbase, freebsd-update, and Security Advisories* turns that split into an operating procedure.

Package repositories are decoupled from OS releases as well. Releases point at *quarterly* ports branches (cut each January, April, July, October, then fed security and bug fixes only), so there is no "FreeBSD 15.1 package set" the way there is a Debian 13 package set. On architecture support, amd64 and aarch64 are the Tier 1 platforms in 15.x with full release, advisory, and package guarantees; armv7, powerpc64, powerpc64le, and riscv64 ship as Tier 2.

1.7 Where FreeBSD is the wrong tool

FreeBSD earns its keep where a coherent base system, ZFS, jails, and the network stack are the workload: storage servers, firewalls and routers, multi-tenant service hosts, long-lived appliance fleets. Before committing a team to it, consider where it is the wrong choice:

- **Kubernetes nodes.** There is no kubelet for FreeBSD and the container ecosystem assumes Linux kernels. FreeBSD 15 publishes OCI base images and jails solve real isolation problems, see *Jails: The Model*, but if your platform is Kubernetes, FreeBSD serves beside it, not inside it.
- **GPU compute and ML stacks.** CUDA and ROCm target Linux; FreeBSD has graphics drivers, not a compute ecosystem.
- **Linux-only vendor software.** Commercial agents (EDR, backup, observability) ship Linux builds. The Linuxulator runs many unmodified Linux binaries, as described in *The Linuxulator: Running Linux Binaries*, but a vendor that says "Linux only" is also defining the support contract.
- **Bleeding-edge and consumer hardware.** Driver enablement lags Linux. Wi-Fi illustrates the gap: recent Intel parts work through LinuxKPI-ported drivers, but 802.11ac is the practical ceiling in 15.1 and firmware ships from ports, not base; see *FreeBSD on Your Hardware: Wi-Fi, Graphics, and the Desktop*.
- **SMB clients.** The in-kernel SMB client speaks SMB1 only; anything modern needs userspace help from ports, see *Cross-OS Interop: SMB, NFS with Linux, and Foreign Disks*.
- **Live kernel patching.** There is no kpatch/livepatch equivalent; a kernel advisory means a reboot. ZFS boot environments make that reboot fast and reversible, see *ZFS in Operation*, but they do not eliminate it.
- **eBPF-based tooling.** Your bpftrace scripts and eBPF-based observability agents do not transfer. DTrace is the native answer and it is excellent, see *Observability: DTrace, Tracing, and Crash Dumps*, but it is a different toolchain your team must learn.

None of these are secrets in the FreeBSD community; they are the standing trade-offs. The adoption playbook in *The Project: Governance, Releases, and the Adoption Playbook* builds on this list.

1.8 How this book works

Every chapter that follows uses three rules so that you can judge the evidence behind each claim.

Version discipline. The book targets FreeBSD 15.1-RELEASE at patch level p1. Every flag, sysctl, default, and file path is checked against a source tree pinned at `releng/15.1` commit `0f691888dc56`. That commit is 15.1-RELEASE-p1, the build the lab machine was provisioned from. Facts come from that tree

and its in-tree manual pages, not from web copies or memory. This source grounding is why citations point at places like `sbin/ifconfig/ifconfig.8` rather than URLs. Features that exist only in 16-CURRENT do not appear here, and behavior that changed within 15.x gets a `> **Version:**` callout. The lab machine has since taken the next advisory batch, so transcripts captured after that point read `15.1-RELEASE-p2`; the source pin stays at p1.

A reproducible lab. Transcripts run on `freebsd-lab-01`, a Proxmox VM built from the official 15.1 cloud image with OpenTofu: 16 cores, 16 GB RAM, a 64 GB ZFS root, one uplink NIC and one isolated lab NIC. *The Lab: FreeBSD 15.1 on Proxmox with OpenTofu* builds it from scratch, and every chapter after it opens with something you can run there. The VM carries a pristine boot environment as a reset point:

```
$ bectl list
BE       Active Mountpoint Space Created
default  NR     /          3.47G 1970-01-01 01:00
pristine -      -          312K  2026-07-16 13:38
```

Verified transcripts. Command output shown in a `console` block was captured on that VM. Prompts are `$` for an unprivileged user and `#` for root. Non-trivial samples also exist as runnable files under `examples/`, like the two this chapter references. Where a claim is stated without a transcript, it traces to the pinned tree.

Next, *The Translation Table: FreeBSD for Linux Eyes* compresses your existing Linux knowledge into the FreeBSD namespace, one row per subsystem, each pointing at the chapter that owns the details.

1.9 Takeaways

- FreeBSD versions kernel, libc, userland, and base daemons as one artifact from one repository; `freebsd-version -kru` must show agreement, and `uname -r` alone under-reports userland patches.
- Base and third-party software are separate planes: base under `/`, patched via SAs to a `-pN` level; ports under `/usr/local`, patched via quarterly repos and audited through VuXML.
- 15.0's headline is pkgbase, base-as-packages, a technology preview that already owns all cloud images and (since 15.1) scripted installs; distribution sets and freebsd-update are scheduled to disappear in FreeBSD 16.
- `pkg which /usr/bin/uname` tells you which update plane owns a given system.
- Support is branch-based and short by Linux enterprise standards: four years per major series, and each point release dies three months after the next one; 15.1 is supported to 2027-03-31, 15.x to 2029-12-31.
- `stable/15` is a development branch despite its name; production runs RELEASE at the latest patch level.
- A stock install is quiet: syslogd is the only network-aware daemon enabled by default, and it ignores remote input.
- FreeBSD is the wrong tool for Kubernetes nodes, GPU compute, Linux-only vendor agents, and bleeding-edge Wi-Fi; it is at its best where ZFS, jails, pf, and a coherent base system are the point.