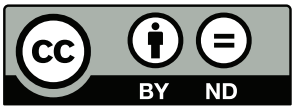
The Administrator's Guide

Migrating from HP-UX to SLES 16

Matthias G. Eckermann

15 December 2025



Contents

1 Preface

Welcome! If you open this guide, you may have spent significant time in your life working on and with HP-UX, and you are tasked to migrate an infrastructure from HP-UX to the modern Linux, SUSE Linux Enterprise Server 16.

This transition from HP-UX on Itanium architecture to SUSE Linux Enterprise Server 16 (SLES 16) on x86-64 represents one of the most significant paradigm shifts in the career of a UNIX system administrator. For over two decades, HP-UX 11i has stood as a paragon of vertical scalability, deterministic stability, and rigorous adherence to the System V (SysV) UNIX philosophy. It is an operating system designed for an era where uptime was maintained by hardware redundancy and monolithic software architectures. The administrator's interaction with HP-UX was imperative, procedural, and deeply rooted in Korn shell (ksh) scripting.

However, the release of SLES 16 in late 2025 marks the arrival of a new era in enterprise Linux, one that diverges not only from legacy UNIX but also from previous SUSE Linux generations. SLES 16 is designed for a world of horizontal scaling, immutable infrastructure, and automated compliance. It introduces radical architectural changes — such as the UsrEtc configuration model, dynamic network configuration using NetworkManager, and the mandatory enforcement of SELinux — that require a complete re-evaluation of established administrative workflows. And SLES 16 continues to deliver added value such as snapshot-rollback capabilities and the transactional update mode of its sibbling, SUSE Linux Micro 6.2.

For the System Administrator, the learning curve is steepest in three areas:

1. **Systemd:** Moving from script-based sequencing to dependency-based state management.

2. **Filesystem & Storage:** Adapting to Btrfs subvolumes, copy-on-write snapshots, and the split UsrEtc hierarchy.

3. **Security:** Shifting from the user-centric DAC model to the policy-centric SELinux MAC model.

This document serves as a technical guide for the Senior Systems Administrator bridging this gap. It does not merely list command equivalents; it explores the philosophical and architectural chasms between the two systems. It provides the "why" behind the "how," ensuring that the transitioning professional understands the underlying mechanisms of systemd resource control, the copy-on-write semantics of Btrfs, and the state-based logic of modern clustering, enabling them to architect solutions that leverage the full capability of the modern Linux world.

To achieve this, we have chosen to describe the differences and approaches in four topical groups:

1. The Architectural Divide (System Core & Philosophy)
2. Daily Administration and Command Tools
3. Advanced Systems and Security
4. Future-Proofing and Development Ecosystem

If you find errors or would otherwise like to participate in this guide, do not hesitate to send contributions via Github.

2 SUSE Linux 16.0 Basics

2.1 Official Documentation and Technical Resources

SLES 16 introduces a reorganized documentation structure. The "System Analysis and Tuning Guide" and "Storage Administration Guide" have been updated significantly to reflect the shift to Btrfs and systemd. Below are the essential resources for the transitioning administrator.

() Document Title	Target Audience & Use Case	Direct Link
Release Notes SLES 16.0	**Start Here.** Covers deprecated packages (e.g., removal of wicked, xinetd), new kernel modules, and known issues.	(https://documentation.suse.com/releasenotes/sles/html/releasenotes_sles_16.0/index.html)
Product Documentation Portal	The central hub for all SLES 16 manuals, accessible in HTML and PDF formats.	documentation.suse.com/sles/16.0
Key Technical Differences	Specifically detailed for admins moving from SLES 15, but highly relevant for UNIX admins to understand the UsrEtc and systemd shifts.	(https://documentation.suse.com/sles/16.0/html/SLE-comparison/)
Storage Administration Guide	Critical for managing Btrfs subvolumes, Snapper rollbacks, and software RAID.	(https://documentation.suse.com/sles/16.0/html/storage/index.html)
Security & Hardening Guide	Covers the implementation of SELinux (Targeted Policy) and auditing with auditd.	(https://documentation.suse.com/sles/16.0/html/security/index.html)
AutoYaST / Agama Guide	Documentation on the new Agama installer and automated deployment profiles.	(https://documentation.suse.com/sles/16.0/html/autoyast/index.html)

2.2 Availability on different Hardware Architectures

Unlike HP-UX, which was tightly coupled to PA-RISC and later Itanium (IA64), SLES 16 is a cross-platform operating system. However, with version 16, SUSE has raised the instruction set baseline for modern processors. SLES 16 is officially available and supported on the following four architectures:

2.2.1 x86-64 (AMD64 / Intel 64)

- **Status:** Primary Enterprise Platform.
- **Requirement: x86-64-v2** microarchitecture level or higher.
- **Impact:** SLES 16 will **not boot** on very old x86-64 CPUs (roughly pre-2009, e.g., first-gen Core 2 Duo or early Opterons) that lack support for SSE4.2, SSSE3, and POPCNT instructions. Administrators repurposing old hardware for sandboxes must verify CPU flags using /proc/cpuinfo.

2.2.2 IBM Power (ppc64le)

- **Status:** High-performance database and analytics platform (SAP HANA).
- **Requirement: IBM Power10** or higher.
- **Note:** While SLES 16 may technically boot on Power9 systems (running in Little Endian mode), SUSE officially supports only Power10 and newer processors for this release. This aligns with the lifecycle of high-end RISC hardware refresh cycles often seen in former HP-UX environments.

2.2.3 IBM Z and LinuxONE (s390x)

- **Status:** Mainframe Linux.
- **Requirement: IBM z14** or higher.
- **Context:** For organizations consolidating HP-UX workloads onto mainframes, SLES 16 leverages the z/Architecture's specific crypto-acceleration and I/O channeling capabilities.

2.2.4 Arm64 (AArch64)

- **Status:** Edge computing and high-density datacenter servers.
- **Requirement: Armv8.0-A** or higher.
- **Use Case:** Increasingly common in cloud environments (e.g., AWS Graviton) and energy-efficient datacenters. SLES 16 includes full support for ACPI on Arm, making it a viable target for general-purpose application migration from legacy UNIX systems.

2.3 Quick Start: Two Paths to SLES 16

For an HP-UX administrator, the fastest way to understand SLES 16 is not to read about it, but to interact with the new CLI and directory structure. SUSE provides two friction-less paths to get a shell prompt without performing a full ISO installation.

2.3.1 Path 1: Local Virtualization (The "Minimal VM" Image)

The **Minimal VM** images contain a stripped-down SLES 16 userland with systemd, zypper, and network-manager pre-configured. These are perfect to start your journey with. **Steps:**

1. **Download:** Visit download.suse.com and select "SLES 16".

2. **Select Image Format:**
   - **KVM/QEMU:** Download SLES-16.0-Minimal-VM.x86_64-kvm-and-xen-GM.qcow2.
   - **VMware ESXi/Workstation:** Download the .vmdk or .ova variant.
   - **Hyper-V:** Download the .vhdx variant.

3. **Boot:** Attach the image to a new VM.

4. **First Boot:** The image utilizes **Combustion** or **Ignition** for configuration, but if no config drive is found, it will default to an interactive wizard to set the root password and network.

2.3.2 Path 2: Public Cloud (AWS, Azure, Google Cloud)

Running SLES 16 in the cloud is the preferred method for testing the new **Agentic AI** features and high-availability stacks without hardware provisioning.
Availability:

- **AWS:** Available as an AMI in the EC2 console. Search for "SUSE Linux Enterprise Server 16".
- **Azure:** Available in the Azure Marketplace.
- **Google Cloud:** Select "SUSE Linux Enterprise Server 16" from the boot disk options when creating an instance.

Cloud-Specific Defaults to Note:
Unlike the "Minimal VM" or ISO installs, public cloud images often have specific pre-configurations:

- **Network:** NetworkManager is the mandatory backend (replacing wicked).2
- **Security:** SELinux is set to Enforcing by default on standard SLES 16 images, though SLES for SAP variants may default to Permissive to ensure database compatibility during the transition period.2
- **Updates:** Instances are pre-wired to the Public Cloud Update Infrastructure (PCUI), meaning zypper update works immediately without manual registration to SUSE Customer Center (PAYG models).

3 The Architectural Divide (System Core & Philosophy)

The divergence between HP-UX and SLES 16 begins at the very core of the operating system: how it boots, how it organizes files, and how it manages memory. These are not superficial differences but fundamental architectural disagreements on how an operating system should function.

3.1 Init System and Service Management: From Sequence to Dependency

The most immediate and jarring change for an HP-UX administrator is the replacement of the sequential System V initialization system with systemd. To understand the magnitude of this shift, one must analyze the boot process of both systems.

3.1.1 Theoretical Divergence: Deterministic Sequence vs. Goal-Oriented Parallelism

3.1.1.1 The HP-UX Boot Process

In HP-UX, the boot process is strictly linear. The Processor Dependent Code (PDC) loads the Initial System Loader (ISL), which loads hpux. The kernel spawns /sbin/init (PID 1). This process reads /etc/inittab, determines the default runlevel (usually 3 or 4), and begins executing scripts in /sbin/init.d/.
These scripts are linked to directories like /sbin/rc3.d/ with names starting with S (Start) or K (Kill) followed by a number (e.g., S100nfs.client). The sequencer executes S100 before S200. This is imperative programming: "Do X, wait for it to finish, then do Y." If S100nfs.client hangs due to a DNS timeout, the entire boot process stalls. The administrator has complete visibility but limited flexibility; optimizing boot time requires manually renumbering scripts.

3.1.1.2 The SLES 16 Systemd Architecture

SLES 16 utilizes systemd, which views the boot process not as a sequence, but as a dependency graph. The administrator defines a "Target" (a desired state, e.g., multi-user.target). Systemd calculates the most efficient path to reach that state. If Unit A depends on Unit B, but Unit C is independent, systemd will start B and C simultaneously.

This approach, known as aggressive parallelization, drastically reduces boot times but introduces non-determinism in startup order—services start as soon as their dependencies are met, not at a fixed time. Furthermore, systemd utilizes Socket Activation, a concept foreign to HP-UX's standard init (though similar to inetd), where systemd listens on a network port on behalf of a service and only starts the daemon when the first packet arrives, further optimizing resource usage.

3.1.2 Directory Structure and Runlevel Mapping

The rigid directory structure of HP-UX has been replaced by a split-location model in SLES 16 to support the distinction between vendor defaults and administrator overrides.

() Feature	HP-UX Implementation	SLES 16 Implementation	Context & Notes
Boot Orchestrator	/sbin/rc (The Sequencer)	/usr/lib/systemd/system	Systemd is the binary orchestrator; rc is a shell script.
Script/Unit Location	/sbin/init.d/	/usr/lib/systemd/system (Vendor) /etc/systemd/system/ (Admin)	**Critical:** Never edit files in /usr/lib. Copy them to /etc/systemd/system/ to override.6
Runlevel 0 (Halt)	/sbin/rc0.d	poweroff.target	
Runlevel 1 (Single)	/sbin/rc1.d	rescue.target	Used for root password recovery and filesystem repair.
Runlevel 3 (Multi)	/sbin/rc3.d	multi-user.target	The standard server state for non-GUI systems.
Runlevel 5 (X11)	/sbin/rc5.d	graphical.target	SLES 16 defaults to this if GNOME is installed.
Config Configuration	/etc/rc.config.d/*	EnvironmentFile inside Unit	HP-UX sources variables from separate files; Systemd loads them as env vars.

3.1.3 The Command-Level Transition: systemctl

The systemctl command is the omnipotent tool in SLES 16, replacing init, rc, and direct script execution.

3.1.3.1 Scenario 1: Managing a Service (e.g., SSH)

- **HP-UX:**
  - *Start:* /sbin/init.d/secsh start
  - *Stop:* /sbin/init.d/secsh stop
  - *Status:* No native status command in the init script usually; admin checks ps -ef | grep sshd.8
- **SLES 16:**
  - *Start:* systemctl start sshd.service

- *Stop:* systemctl stop sshd.service
- *Status:* systemctl status sshd.service
- *Insight:* The status command in systemd is far superior. It displays the service state (Active/Inactive), the main PID, the memory usage (via cgroups), and the last 10 lines of log output from Journald. This instant context is unavailable in HP-UX.9

3.1.3.2 Scenario 2: Enabling a Service at Boot

- **HP-UX:** The admin must edit /etc/rc.config.d/secsh and set SSHD_START=1. This variable is checked by the script at runtime.
- **SLES 16:** systemctl enable sshd.service.
  - *Mechanism:* This command reads the [Install] section of the unit file and creates a symbolic link in /etc/systemd/system/multi-user.target.wants/ pointing to the unit. It is a filesystem operation, not a variable setting.

3.1.4 Anatomy of a Systemd Unit vs. Init Script

To illustrate the shift from imperative scripting to declarative definition, we compare a theoretical custom service.

3.1.4.1 HP-UX /sbin/init.d/custom_app (Abbreviated)

```
#!/sbin/sh
# Source configuration
if [ -f /etc/rc.config.d/custom_app ] ; then
   . /etc/rc.config.d/custom_app
fi

case $1 in
'start_msg')
    echo "Starting Custom App"
    ;;
'start')
    if; then
        # Manual PID management required
        /opt/app/bin/server > /var/log/app.log 2>&1 &
        echo $! > /var/run/custom_app.pid
    fi
    ;;
'stop')
    # Fragile kill logic
    if [ -f /var/run/custom_app.pid ]; then
        kill $(cat /var/run/custom_app.pid)
    fi
    ;;
esac
```

Critique: The script must handle logging redirection, PID file creation, and variable sourcing manually. Error handling is the responsibility of the script author.

3.1.4.2 SLES 16 /etc/systemd/system/custom_app.service

```
[Unit]
Description=Custom Application Service
After=network-online.target remote-fs.target
Wants=network-online.target

Type=simple
ExecStart=/opt/app/bin/server
# Systemd handles logging (stdout/stderr goes to Journal)
StandardOutput=journal
StandardError=journal
# Automatic Restart Logic
Restart=on-failure
RestartSec=5s
# Security Hardening (Unavailable in HP-UX Init)
User=appuser
Group=appgroup
ProtectHome=true
PrivateTmp=true

[Install]
WantedBy=multi-user.target
```

Analysis: The unit file is declarative. The After= directive handles ordering. Restart=on-failure provides high availability functionality that would require a separate monitoring tool (like Monit or HP Serviceguard) on HP-UX. The security directives (ProtectHome, PrivateTmp) leverage kernel namespacing to sandbox the process, a feature HP-UX init scripts cannot easily replicate.

3.2 The UsrEtc Revolution and Filesystem Hierarchy Divergence

One of the most profound changes in SLES 16 is the adoption of the **UsrEtc** (Hermetic /usr) model. This is a significant deviation from the traditional Filesystem Hierarchy Standard (FHS) used by HP-UX and earlier Linux distributions.

3.2.1 The Philosophy: Immutable Vendors, Mutable Admins

In HP-UX, the operating system and the local configuration are commingled.

- **HP-UX Reality:** When you install a patch (PHSS_xxxxx), it might update a config file in /etc/opt/. If the administrator had modified that file, SD-UX might create a .save file or simply overwrite it, leading to "configuration drift" and breakage during patching.

- **SLES 16 Philosophy:** The /usr directory is the sole property of the OS vendor (SUSE). It is treated as immutable and can be mounted read-only. The /etc directory is the sole property of the administrator.

3.2.2 Practical Implications of UsrEtc

In SLES 16, configuration files are looked up in a specific hierarchy.

1. **Vendor Default:** /usr/etc/ (or /usr/share/defaults/etc/)

2. **Admin Override:** /etc/

Workflow Example: Configuring SSH
On HP-UX, you edit /etc/ssh/sshd_config directly. If you mess up, you must restore from backup.
On SLES 16:

1. The vendor supplies /usr/etc/ssh/sshd_config (Read-only).

2. The admin wants to change the port to 2222.

3. The admin creates /etc/ssh/sshd_config containing *only* the override, or copies the vendor file to /etc and modifies it.

4. The application reads /etc first.

Insight: This eliminates the problem of .save (HP-UX) or .rpmnew and .rpmsave (Linux) files. The package manager can update /usr/etc/ssh/sshd_config freely without touching the admin's customization in /etc. For the administrator, this means grep -r in /etc is no longer sufficient to find all configurations; one must check both /usr/etc and /etc to understand the *effective* configuration.

3.2.3 The Partitioning of /opt and /usr/local

- **HP-UX:** /opt is the primary location for almost all software (SD-UX bundles install to /opt/package). /usr is often a mix of OS links and binaries.

- **SLES 16:**
  - /usr/bin: Primary location for package binaries.

  - /opt: Reserved for large, self-contained third-party applications (Oracle DB, SAP HANA, Microsoft, proprietary agents).

  - /usr/local: Reserved for software compiled manually by the admin (./configure && make install). SLES package manager will *never* touch files in /usr/local.

3.3 Memory Management

3.3.1 Memory Overcommit vs. Strict Allocation

HP-UX is conservative. A malloc() call generally reserves backing store immediately. If swap+RAM is full, malloc fails.

SLES 16 (Linux kernel) is optimistic. It uses Overcommit. malloc almost always succeeds, returning a pointer to virtual memory. Physical memory is only allocated when the page is written to (Fault-in). Risk: If the system runs out of physical RAM, the Linux OOM Killer (Out of Memory Killer) activates. It uses a heuristic to select a process (often the one using the most memory, like a Database) and kill -9 it to save the kernel.
Mitigation for DB Servers:
On SLES 16 hosting Oracle/SAP, administrators often tune /etc/sysctl.conf:

```
vm.overcommit\_memory \= 2  \# Strict accounting (like HP-UX)
vm.overcommit\_ratio \= 50  \# Limit to Swap \+ 50% RAM
```

This forces the kernel to behave more like HP-UX, returning errors on malloc rather than killing random processes later.

Caveat: Please check the latest information on the SUSE website, and in the documentation of your ISV, whether these settings are recommended.

4 Daily Administration and Command Tools

Moving from the HP-UX sam and command-line ecosystem to SLES 16 requires mastering a new set of tools that are generally more automated but abstract away more detail.

- Package Management: SD-UX vs. Zypper and Cockpit
- Disk, LVM, and Filesystems (HP LVM/VxFS vs. Linux LVM/Btrfs/Snapper)
- Command Shells (sh/ksh vs. Bash)
- Missing
  - System Management Tools (SMH/SAM vs. Cockpit)
  - Networking Configuration (Traditional vs. NetworkManager)

4.1 Package Management: SD-UX vs. Zypper and Cockpit

For the HP-UX administrator, swinstall, swlist, and swremove (SD-UX) are muscle memory. The transition to SLES 16 requires a conceptual shift from managing static "software depots" to interacting with dynamic, dependency-aware "repositories" via the **Zypper** command-line engine and the **Cockpit** web interface.

4.1.1 The SAT Solver Revolution: Zypper vs. SD-UX

The most critical architectural difference lies in dependency resolution. HP-UX SD-UX is fundamentally a package *installer*. It verifies prerequisites but generally places the burden of retrieving dependencies on the administrator.
SLES 16's **Zypper** utilizes libzypp, which is built on a Boolean Satisfiability (SAT) solver. When an administrator requests a package, the solver computes the most efficient path to a consistent system state, automatically selecting necessary libraries, resolving version conflicts, and handling architecture constraints.

4.1.1.1 Workflow Comparison

A. Installing Software with Dependencies

- **HP-UX (SD-UX):**
  - *Action:* Install git.
  - *Command:* swinstall -s /var/spool/sw/git.depot git
  - *Outcome:* If git requires libiconv and it is not in the depot or installed, the operation fails with a prerequisite error. The admin must manually locate the libiconv depot, register it, and retry.
- **SLES 16 (Zypper):**
  - *Action:* Install git.

– *Command:* zypper install git (or zypper in git)

– *Outcome:* Zypper queries all subscribed repositories. It identifies that git requires libsha1, libpcre2, and perl-Error. It calculates the install order, presents a transaction summary (size, packages to be installed), and prompts for confirmation. Upon y, it downloads and installs all components in a single transaction.

B. Querying Package Metadata

- **HP-UX:** Finding which product owns a binary is slow because swlist parses a flat-file database.
  – *Command:* swlist -l file | grep /usr/bin/perl

- **SLES 16:** The RPM database is an indexed binary database, allowing near-instant queries.
  – *Find owner of file:* rpm -qf /usr/bin/perl

  – *Info about available package:* zypper info git (Shows version, source repo, support status, and description).

  – *Search by capability:* zypper search --provides /usr/bin/perl (Finds any package that provides this path, even if not installed).

C. Verification and Integrity

- **HP-UX:** swverify git checks the file existence and permissions against the IPD (Installed Products Database).

- **SLES 16:** rpm -V git verifies file size, MD5/SHA256 checksum, permissions, type, owner, and group against the RPM database.

4.1.2 Repository Management vs. Depots

In HP-UX, you manage "Depots" (local directories or tape devices). In SLES 16, you manage "Repositories" (remote HTTP/HTTPS/FTP servers or local directories).

() Task	HP-UX SD-UX	SLES 16 Zypper
List Sources	swlist -l depot	zypper repos (or zypper lr -d)
Add Source	swreg -l depot /path/to/depot	zypper addrepo <url> <alias>
Refresh Metadata	(Manual re-registration usually required)	zypper refresh
Clean Cache	N/A	zypper clean

Expert Tip: SLES 16 introduces the concept of **Vendor Stickiness**. If a package is installed from the SUSE-Main repo, Zypper will *not* automatically update it to a higher version found in a third-party repo (e.g., Pacman). This prevents stability regressions. To force a vendor change, the administrator must explicitly use zypper dup --from <repo> or zypper install --from <repo> <package>.4

4.1.3 Cockpit: The Modern Replacement for SAM

In SLES 16, the monolithic sam (System Administration Manager) and legacy YaST modules have been succeeded by **Cockpit** for 1:1 system management.1

Cockpit is a lightweight, web-based interface that communicates directly with system APIs (like systemd, NetworkManager, and PackageKit) via a websocket. It does not maintain its own database state, meaning changes made in the CLI (zypper) are instantly reflected in the GUI, and vice versa.

4.1.3.1 Key Capabilities for the Administrator

1. Software Management:
   Unlike the static view of sam, Cockpit's software module provides a live view of the system's compliance status. It integrates with SUSE Customer Center to track subscriptions and available patches. Administrators can trigger updates, manage repositories, and inspect installed packages directly from the browser.1

2. Systemd Integration:
   Cockpit provides a visual hierarchy of the systemd dependency graph. An administrator can view unit logs (journald), override unit files, and manage targets (runlevels) without parsing complex systemctl list-dependencies output.

3. Storage and Networking:
   Cockpit visualizes the Btrfs subvolume layouts and Snapper snapshots discussed in Section 2.2, simplifying the complexity of rollback management that would otherwise require intricate CLI commands.

Accessing Cockpit:

- **Service:** systemctl enable --now cockpit.socket
- **URL:** https://<server-ip>:9090
- **Authentication:** Uses standard PAM system credentials (root or sudo users).

By leveraging **Zypper** for scriptable, granular control and **Cockpit** for high-level visualization and management, the SLES 16 administrator gains a toolset that is significantly faster and more responsive than the legacy SD-UX/SAM combination.

4.1.4 Command Mapping: SD-UX vs. Zypper

The following table provides a direct translation of common SD-UX tasks to their SLES 16 equivalents. Note that while SD-UX operates on local or NFS-mounted depots, Zypper interacts with networked repositories, handling metadata synchronization automatically.

() Administrative Task	HP-UX (SD-UX)	SLES 16 (Zypper/RPM)	Context & Critical Differences
Install Package	swinstall -s /source package	zypper install package	Zypper automatically resolves and fetches dependencies (SAT Solver). SD-UX fails if prereqs are missing.
Remove Package	swremove package	zypper remove package	Zypper also offers to remove unused dependencies (-u).

() Administrative Task	HP-UX (SD-UX)	SLES 16 (Zypper/RPM)	Context & Critical Differences
List Installed	swlist	zypper search -i rpm -qa	rpm -qa lists all packages; zypper se -i provides a searchable, formatted list.
Verify Integrity	swverify package	rpm -V package	Checks checksums, permissions, and ownership against the local RPM database.
Find File Owner	swlist -l file	grep file	rpm -qf /path/to/file
Search Remote	*(Manual Depot Browsing)*	zypper search query	Searches all subscribed metadata (descriptions, summaries, and names).
Add Source/Repo	swreg -l depot /path	zypper addrepo <url> <alias>	SLES repos persist in /etc/zypp/repos.d/. No need to re-register after reboots.
Patch/Update	swinstall -x match_target=true	zypper update zypper patch	patch installs security fixes; update upgrades packages to newer versions.
Clean Up	swremove -d (Depot cleanup)	zypper clean	Removes cached headers and RPM files from /var/cache/zypp.

4.2 Disk, LVM, and Filesystems (HP LVM/VxFS vs. Linux LVM/Btrfs/Snapper)

4.2.1 HP LVM vs. Linux LVM2

Linux LVM was originally based on HP-UX LVM, so the concepts map 1:1, but the limitations differ.

() Feature	HP-UX LVM	SLES 16 LVM2	Critical Difference
Physical Volume	pvcreate /dev/rdsk/c2t0d0	pvcreate /dev/sdb	Linux uses block devices, not raw (r) devices for LVM.
Volume Group	vgcreate vg01 /dev/dsk/c2t0d0	vgcreate vg01 /dev/sdb	
Logical Volume	lvcreate -L 1000 -n lvol1 vg01	lvcreate -L 1G -n lvol1 vg01	Linux accepts human-readable sizes (G, M, T).
Extend	lvextend -L 2000 /dev/vg01/lvol1	lvextend -L +1G /dev/vg01/lvol1	Linux syntax +1G allows relative extension.

() Feature	HP-UX LVM	SLES 16 LVM2	Critical Difference
Max Extents	Rigid max_pe set at creation.	Virtually unlimited (64-bit).	HP-UX often requires recreating the VG if max_pe is exceeded. Linux does not.27
Device Nodes	/dev/vg01/lvol1 (Character & Block)	/dev/vg01/lvol1 (Symlink to dm device)	Linux uses Device Mapper (/dev/dm-5).

4.2.2 The Root Filesystem: Btrfs and Subvolumes

HP-UX 11i v3 defaults to VxFS (Veritas), utilizing Extent-based allocation and intent logging. SLES 16 defaults to **Btrfs** for the root partition.
Why Btrfs?

1. **Copy-on-Write (CoW):** When a file is modified, Btrfs writes the new data to a free block and then updates the metadata pointer. The old data remains until explicitly freed. This enables instant snapshots.

2. **Bitrot Protection:** Btrfs calculates checksums for data and metadata. If a block is corrupted on disk, Btrfs detects the mismatch on read (unlike VxFS which might silently return corrupted data).

4.2.2.1 Disk Layout & Subvolumes

In HP-UX, /, /usr, /var, and /tmp are usually separate Logical Volumes (LVs). In SLES 16, they are Subvolumes within a single Btrfs partition.

- subvolid=5 (Top Level)
- @ (Root filesystem, mounted at /)
- @/var, @/usr/local, @/tmp (Nested subvolumes)
- **Crucial:** Certain directories like /usr/local, /var/log and /var/lib/pgsql, /var/lib/mysql are created as subvolumes *exempt* from snapshots. This prevents the database or logs from being "rolled back" when the OS is reverted.

4.2.3 Snapper: Pre/Post Transactional Rollbacks

This is the "killer feature" for the SLES administrator.

- **The Scenario:** You run zypper update to patch glibc and the kernel. The update completes, but upon reboot, the application fails to start due to a library incompatibility.
- **HP-UX Recovery:** Requires restoring from an Ignite-UX tape or network image, which takes hours. Or, if you were prudent, switching to an alternate boot disk (split mirror), which requires significant pre-planning.

- **SLES 16 Recovery (Snapper):**
  1. Zypper automatically triggered a "Pre" snapshot before the update and a "Post" snapshot after.
  2. Admin reboots into a read-only snapshot via the GRUB2 menu to verify the old state works.
  3. Admin runs snapper rollback.
  4. The system makes the "Pre" snapshot the new default boot target.
  5. Reboot. Total downtime: Minutes. For just "undoing" configuration changes, which do not require a reboot, there is an even more lightweight method by running "snapper undochange".

For more information on snapper, please use the SUSE Linux Enterprise documentation at (LINK TO BE DONE)

4.2.4 UEFI and the /boot/efi Partition

HP-UX on Itanium was one of the first OSs to use EFI. SLES 16 on x86-64 mandates UEFI.

- **Requirement:** A dedicated partition (formatted VFAT/FAT32) mounted at /boot/efi.
- **Sizing:** While HP-UX EFI partitions were often small, SLES 16 recommendations are **512 MB to 1 GB**.
  - *Reason:* SLES stores the GRUB2 bootloader shim.efi, grubx64.efi, and potentially kernel images here (though usually kernels stay in /boot on XFS/Btrfs). Linux firmware updates (fwupd) also utilize this space. A partition smaller than 256MB will frequently fail during Service Pack upgrades due to lack of space for temporary boot images.32
- **Partition Table:** Must be GPT (GUID Partition Table). The legacy MBR (Master Boot Record) is not supported for the boot drive in default SLES 16 UEFI installations.

4.2.5 Architectural Deep Dive: HP LVM vs. Linux LVM2

While Linux LVM was originally inspired by HP-UX LVM, the implementations have diverged significantly. For the HP-UX administrator, the command syntax is familiar, but the underlying mechanisms for metadata, device nodes, and allocation policies are fundamentally different.

4.2.5.1 The "Group File" vs. Udev and Device Mapper

In HP-UX, the Volume Group (VG) requires a character device file (the "group file") to communicate with the kernel.

- **HP-UX Legacy Workflow:**
  1. mkdir /dev/vg01
  2. mknod /dev/vg01/group c 64 0x010000 (Manual creation of major/minor numbers).
  3. vgcreate /dev/vg01...

- SLES 16 Architecture:
  Linux uses udev and the Device Mapper kernel framework. There is no concept of a "group file" or manual major/minor number assignment for VGs.
  1. vgcreate vg01 /dev/sdb

  2. The kernel automatically assigns a dynamic minor number.

  3. udev automatically creates the device nodes in /dev/vg01/ and /dev/mapper/.

     - **Educational Note:** In Linux, /dev/vg01/lvol1 is merely a symbolic link to /dev/dm-X. The real block device is managed by the device mapper.

4.2.5.2 Metadata: Binary VGRA vs. Text-Based LVM2

- **HP-UX:** Uses binary structures like the **VGRA** (Volume Group Reserved Area) and **VGDA** (Volume Group Descriptor Area). If these headers are corrupted, specialized binary tools (or dd wizardry) are required to recover them.

- **SLES 16 (LVM2):** Uses a **Text-Based Metadata** format.
  - *Feature:* The metadata stored on the disk (typically in the first 1MB) is a human-readable ASCII configuration.

  - *Recovery:* SLES keeps automatic backups of this text metadata in /etc/lvm/archive/. If a PV header is wiped, an administrator can literally cat the backup file to find the UUIDs and extent maps, then restore it using vgcfgrestore.

4.2.5.3 Allocation Policies and "Strict" Mirroring

- **HP-UX:** Admins often rely on **PVG-Strict** (Physical Volume Group) mirroring to ensure mirrors sit on separate hardware controllers.

- **SLES 16:** Linux LVM does not use "PV Groups" in the same way. Instead, it uses allocation policies:
  - --alloc contiguous: Enforces sequential blocks.

  - --alloc cling: Prefers keeping data on the same PV.

  - --mirrors 1: By default, LVM attempts "strict" allocation (allocating the mirror leg on a different PV). To replicate complex PVG-strict logic, SLES admins use **Tags** on Physical Volumes or explicitly specify PVs during creation (lvcreate... /dev/sda /dev/sdb).

4.2.5.4 Bad Block Relocation (BBR)

- **HP-UX:** LVM historically handled Bad Block Relocation (lvcreate -r y).

- **SLES 16:** Linux LVM **does not** perform software-based bad block relocation. The philosophy is that modern storage arrays and drive firmware handle sector remapping transparently. If a drive presents a bad block to the OS, the filesystem (Btrfs) or the hardware RAID controller is expected to handle it, not the Volume Manager.

4.3 Command Shells (sh/ksh vs. Bash)

A significant friction point for HP-UX administrators moving to Linux is the shell environment. While HP-UX scripts and interactive sessions are deeply rooted in the **Korn Shell (ksh88)**, SLES 16 standardizes on **GNU Bash (Bourne Again SHell)**.

4.3.1 2.4.1 The Default Shell Landscape

- **HP-UX (Legacy):** The standard interactive shell is /usr/bin/ksh, which is based on the AT&T **ksh88** standard. While ksh93 (dtksh) is available on later HP-UX 11i versions, most system scripts and administrator muscle memory rely on ksh88 behaviors (e.g., set -A for arrays).
- **SLES 16 (Modern):** The default system and user shell is /bin/bash (version 5.x). Bash is largely a superset of the Bourne shell (sh) but includes features from ksh and csh.

4.3.1.1 A Note on mksh in SLES 16

SLES 16 repositories include the **MirBSD Korn Shell (mksh)**. Administrators might be tempted to install this and symlink /bin/ksh to mksh to maintain legacy scripts.

- **Warning:** mksh is a descendant of pdksh (Public Domain ksh), not the official AT&T ksh88/93 source. It is **not** fully compatible with HP-UX ksh88. It handles arrays, coprocesses, and scoping differently.
- **Recommendation:** Rewrite system automation scripts in Bash. Use mksh only for interactive familiarity, not for critical infrastructure scripts.

4.3.2 2.4.2 Comparative Cheat Sheet: ksh88 vs. Bash

The following table highlights the 20 most critical syntax differences an administrator will encounter when porting scripts or working interactively.

() Feature/Task	HP-UX (ksh88)	SLES 16 (Bash)	Migration Note
Define Array	`set -A arr val1 val2`	`arr=(val1 val2)`	**Major Breaker.** Bash does not support set -A.
Access Array	`${arr[1]}`	`${arr[1]}`	Same syntax (0-indexed).
List All Array Elements	`${arr[*]}`	`${arr[@]}` or `${arr[*]}`	Bash prefers @ for quoted expansion `${arr[@]}`.
String Replacement	*(External sed required)*	`${var/pattern/repla`	Bash has built-in string manipulation; ksh88 does not.
String Slicing	*(External cut required)*	`${var:offset:length`	e.g., `${var:0:4}` extracts first 4 chars.

() Feature/Task	HP-UX (ksh88)	SLES 16 (Bash)	Migration Note
Upper/Lower Case	`typeset -u var` / `typeset -l`	`${var^^}` / `${var,,}`	Bash uses parameter expansion for one-off case conversion.
Process Substitution	*(Requires named pipes/fifos)*	`diff <(cmd1) <(cmd2)`	Bash can feed command output as a file descriptor dynamically.
Here Strings	`echo "$var"|cmd`	`cmd <<< "$var"`	Bash shortcut to feed a string to stdin.
Regular Expressions	*(External grep required)*	`[[ $var =~ ^regex$ ]]`	Bash supports regex matching natively inside `[[ ]]`.
Prompt Customization	PS1='$PWD $'	PS1='\u@\h:\w\$'	Bash uses \ escapes (\u user, \h host, \w cwd).
Command Aliases	alias name='cmd'	alias name='cmd'	Same syntax.
Repeat Last Command	r	!!	r is a standard alias in ksh; !! is history expansion in Bash.
Search History	r string	!string or Ctrl+R	Bash Ctrl+R provides interactive reverse search.
Arithmetic	((i=i+1)) or let	((i++))	Bash supports C-style increment/decrement operators.
Sequence Generation	*(External loop required)*	{1..10}	Bash Brace Expansion generates sequences; ksh88 does not.
Co-processes	'cmd	&'	coproc cmd
Local Variables	typeset var	local var	local is the preferred Bash keyword inside functions.
Function Export	typeset -fx func	export -f func	Required to make functions visible to subshells.
Test File Exists	[[-a file]]	[[-e file]]	-a is deprecated in Bash; use -e.
Reading User Input	read var?Prompt	read -p "Prompt" var	ksh88 embeds prompt in variable; Bash uses -p flag.

4.3.2.1 Practical Migration Strategy

1. **Shebang Update:** Change #!/usr/bin/ksh to #!/bin/bash.

2. **Linting:** Install shellcheck on SLES 16 (zypper in ShellCheck). Run it against legacy scripts to instantly identify ksh88 syntax that fails in Bash. Mind that ShellCheck is not part of SLES 16.0,

but the community supported PackageHub repository.

3. **Environment:** Do not copy .profile from HP-UX to SLES. Use the SLES default .bashrc and port aliases manually to avoid environment pollution.

5 Advanced Systems Management and Security

- Security and Access Control
- Logging and Auditing
- Clustering and High Availability: Serviceguard vs. Pacemaker
- Kernel Patching: Zero Downtime with kGraft
- Missing
  - Time Sync Services (ntpd vs. Chrony)

5.1 Security and Access Control

5.1.1 DAC vs. MAC: The Conceptual Leap

HP-UX security is based on **Discretionary Access Control (DAC)**. A user (or root) has discretion over the files they own. "Trusted System" (TCB) mode in HP-UX adds auditing and shadow passwords, but it is still fundamentally DAC.
SLES 16 employs **Mandatory Access Control (MAC)** via **SELinux** (Security Enhanced Linux). In MAC, the *policy* dictates access, not the user.

- **The Difference:** In HP-UX, if the apache user has rwx permissions on /etc/passwd, it can read/write it. In SLES 16 with SELinux, even if the apache user has 777 permission on /etc/passwd, the kernel will block the access because the SELinux policy for the httpd_t domain does not grant write access to the passwd_file_t type.

5.1.2 Administrative Challenges with SELinux

SLES 16 defaults to SELinux Enforcing mode (moving away from the AppArmor default of SLES 15).

- **Context Labels:** Every file has a label (viewable with ls -Z). When an admin copies a config file from their home directory to /var/www, the file might retain the user_home_t label. Apache will be denied access.

- **Troubleshooting:**
  - *Wrong Way:* setenforce 0 (Disabling security).

  - *Right Way:* Check /var/log/audit/audit.log. Use restorecon -v /var/www/file to apply the correct label defined by the policy.

- **Boolean Flags:** SLES 16 provides "knobs" to tune policy without rewriting it.
  - Example: Allow Apache to connect to a remote database?

- Command: setsebool -P httpd_can_network_connect_db 1.
  This level of granular control does not exist in HP-UX.

5.2 Logging and Auditing

5.2.1 Syslog vs. Journald

HP-UX uses the classic text-based syslog. SLES 16 uses **Systemd Journald**.

- **Structure:** Journald stores logs in a binary format (/var/log/journal). It automatically indexes metadata: PID, UID, Systemd Unit, Executable Path.

- **Querying:**
  - *HP-UX:* grep "error" /var/adm/syslog/syslog.log

  - *SLES 16:* journalctl -p err -u apache2 --since "1 hour ago"

  - *Benefit:* The ability to filter by *Unit* (-u) is incredibly powerful, isolating logs for a specific service regardless of where it sent its output (stdout, stderr, or syslog facility).

5.2.2 Auditing: audsys vs. Linux Auditd

HP-UX audsys creates binary logs in /var/adm/audit/ that must be converted. Linux auditd does similarly but with different tooling.
Task: Find all failed login attempts in the last 24 hours.

- HP-UX:
  The admin must process the binary btmps file or audit logs.
  Bash
  # /usr/sbin/acct/fwtmp -X < /var/adm/btmps | grep "non-zero-exit"
  # Or parsing audisp output
- SLES 16:
  The ausearch tool queries the audit daemon logs directly.
  Bash
  # ausearch -m USER_LOGIN -sv no -ts recent
  Explanation: -m USER_LOGIN filters for login messages. -sv no filters for Success Value = No (failed). -ts recent implies the last 10 minutes, or use -ts today.
  Alternatively, for basic wtmp/btmp analysis:
  Bash
  # lastb --since yesterday
  This command reads /var/log/btmp (binary file of bad logins) and formats it instantly.

5.3 Clustering and High Availability: Serviceguard vs. Pacemaker

HP Serviceguard is widely considered the most mature UNIX clustering solution. SLES 16 utilizes the **Pacemaker** stack (with Corosync), which has reached feature parity but operates differently.

5.3.1 Architecture Comparisons

() Component	HP Serviceguard	SLES 16 (Pacemaker)	Transition Note
Communication Layer	TCP/UDP Heartbeat, proprietary protocol.	**Corosync** (Totem Single Ring Protocol).	Corosync handles membership and quorum.
Resource Manager	cmcld (Cluster Daemon).	**Pacemaker** (pacemakerd).	Pacemaker decides *where* things run.
Configuration	cmcluster.ascii & package.conf.	**CIB** (Cluster Information Base) XML.	Never edit XML directly; use crm or pcs shell.
Agents	Control Scripts (Shell).	**OCF Agents** (Open Cluster Framework).	OCF agents are standardized scripts with start, stop, monitor actions.36
Fencing	Safety Timer / Lock Disk.	**STONITH** (Shoot The Other Node In The Head).	Mandatory in SLES 16. Uses SBD (Storage Based Death) or IPMI.

5.3.2 Configuration Complexity and Syntax

Serviceguard:
Packages are monolithic. The control script handles IP assignment, disk mounting, and application start sequentially.
Pacemaker:
Resources are granular and linked by constraints.

- **Resource Group:** A container ensuring resources start sequentially on the same node.
- **Colocation Constraint:** "WebIP must run on the same node as WebServer."
- **Order Constraint:** "Filesystem must start before Database."

Migration Scenario:
To migrate a Serviceguard package that mounts /dev/vg01/lvol1 and starts Oracle:

1. Define a Filesystem resource (OCF).
2. Define an IPaddr2 resource (OCF).
3. Define an oracle resource (OCF).
4. Group them: pcs resource group add OracleGroup IP FS OracleDB.
   This modularity allows for more complex dependency graphs than Serviceguard's linear scripts.37

5.4 Kernel Patching: Zero Downtime with kGraft

On HP-UX, a kernel patch (PHKL_xxxx) essentially always requires a reboot to relink the kernel and load new code. This necessitates scheduled maintenance windows.

SLES 16 includes SUSE Live Patching (kGraft).

5.4.1 Technical Explanation of kGraft

kGraft allows the administrator to apply critical security patches to the running kernel *without* restarting the system or even stopping applications.

- **Mechanism:** It relies on the ftrace infrastructure in the Linux kernel. A live patch is loaded as a kernel module (.ko). This module contains the replacement code for a vulnerable function. kGraft injects a redirection (trampoline) at the beginning of the old function, diverting execution to the new function.

- **Consistency:** kGraft uses a "lazy migration" model. It tracks kernel threads. A thread is migrated to the new function world only when it exits kernel space. This ensures that no thread sees an inconsistent state (e.g., calling the old function but getting data from the new one). Once all threads have transitioned, the old function is retired.

5.4.2 Admin Workflow

1. **Check Status:** kgr status (Shows "ready").

2. **Install Patch:** zypper install kernel-livepatch-SLE15-SPx...

3. Verify: The patch applies immediately. kgr status shows "in_progress" then "ready". The uname -r version does not change, but the code in memory is patched.
   This capability fundamentally changes the patching cadence, allowing security teams to patch Shellshock/Spectre-class vulnerabilities mid-day without business interruption.38

6 Future-Proofing and Development Ecosystem

- Configuration Management: The Shift to IaC
- Dynamic Linking, Compilation, and Symbol Versioning: Itanium vs. x86-64
- Missing
  - Development Tools (Proprietary Compilers vs. GCC): Extended Development framework description
  - Virtualization/Containers (HPVM vs. KVM/Podman/Docker)
  - Monitoring & Performance (GlancePlus vs. Prometheus/Grafana)

6.1 Configuration Management: The Shift to IaC

HP-UX administration is historically imperative. Admins write intricate ksh or Perl scripts to loop through servers (remsh or ssh) to edit files using sed or awk. This leads to "Configuration Drift," where no two servers are exactly alike.
SLES 16 is designed for **Infrastructure as Code (IaC)**, specifically integrating **Ansible**.

6.1.1 Native Integration: Ansible System Roles

SLES 16 ships with **Ansible System Roles** (rhel-system-roles or suse-system-roles). These are vendor-supported playbooks.

- **The Difference:** Instead of writing a script to "Configure NTP," the admin defines the *desired state* in a YAML variable file.
  YAML
  # vars.yml
  timesync_ntp_servers:
  - hostname: pool.ntp.org
  iburst: yes
  Then runs the role:
  Bash
  ansible-playbook -i hosts timesync.yml
- **Why it matters:** The System Role handles the complexity of SLES 16 internals (editing /etc/chrony.conf, restarting chronyd, enabling the service, handling SELinux ports). It ensures **Idempotency**—running the playbook ten times results in the same state as running it once, without errors or duplicate config lines.

- **SaltStack:** While Ansible is the focus for agentless config, SUSE Manager (SUMA) relies on **Salt**. SLES 16 includes the salt-minion package, allowing the OS to be controlled by a Salt Master for event-driven automation (Reactors), a concept absent in standard HP-UX.

6.2 Dynamic Linking, Compilation, and Symbol Versioning: Itanium vs. x86-64

The transition from Itanium (IA64) to x86-64 involves a complete change in the Application Binary Interface (ABI) and memory management subsystem.

6.2.1 Shared Library Management: SHLI vs. glibc/ELF

HP-UX on Itanium uses the ELF object format, but the dynamic linking behavior retains legacy HP-UX characteristics.

- **HP-UX Dynamic Loader:** Uses dld.so. It historically respected the SHLIB_PATH environment variable for 32-bit applications and LD_LIBRARY_PATH for 64-bit applications. The loader supports "depth-first" or "breadth-first" search orders depending on compilation flags (+std, +compat).14

- **SLES 16 Dynamic Loader:** Uses ld-linux-x86-64.so.2 (part of glibc). It **only** respects LD_LIBRARY_PATH. SHLIB_PATH is ignored.

Critical Migration Action:
Administrators migration scripts must be audited. Any script exporting SHLIB_PATH must be rewritten to export LD_LIBRARY_PATH. Furthermore, relying on LD_LIBRARY_PATH for system-wide configuration is an anti-pattern in Linux. The correct method is to add the library directory to a file in /etc/ld.so.conf.d/ and run ldconfig to update the global cache /etc/ld.so.cache.

6.2.2 Compilation and Symbol Versioning

When recompiling in-house C/C++ applications from HP-UX to SLES 16:

- **Compiler:** HP-UX uses the aCC (HP ANSI C++) compiler or cc. SLES 16 uses gcc (GNU Compiler Collection).

- **Flags:** The HP-UX +z or +Z flags (for Position Independent Code - PIC) must be replaced with -fPIC in GCC. The +b flag (to embed runpaths) translates to -Wl,-rpath.18

- **Glibc:** SLES 16 uses glibc (GNU C Library). Glibc utilizes strict **Symbol Versioning**. If an application is compiled against a specific version of a library, it expects that specific version of the symbol at runtime. This prevents the "DLL Hell" sometimes seen on older UNIX systems but requires that binaries be recompiled if the underlying library major version changes significantly. HP-UX's SHLI (Shared Library) versioning was often looser, allowing simpler symlink swaps to upgrade libraries.

7 Appendix

7.1 Common Command Comparison Table

() Task Category	Description	HP-UX Command(s)	SLES 16 Command(s)
Service Control	Start/Stop/Check Service Status	`init.d/<service> start` or `/sbin/init.d/net stop`	`systemctl start <service>` or `systemctl status sshd`
Package Management	Install a Package	`swinstall -x autoreboot=true -s /tmp/depot/pkgname`	`zypper install pkgname`
Patch/System Update	Apply all system patches	`swinstall` or `patchadd`	`zypper update`
Process Monitoring	View interactive process list	`top` or `glance`	`top` or `htop`
CPU/Memory Stats	View system resources	`sar -u` or `vmstat`	`sar -u` or `vmstat` (Output differences)
Filesystem Status	Check disk space usage	`bdf` (similar to `df`)	`df -h`
LVM - Display VGs	List Volume Groups	`vgdisplay`	`vgdisplay` (similar syntax, different implementation)
Networking - IP	View/Configure IP addresses	`ifconfig <interface>`	`ip a` or `ip addr show`
Networking - Status	View network connections	`netstat -rn`	`ss -tuln` (preferred) or `netstat -rn`
Firewall Mgmt	View or modify firewall rules	`ipfstat` (if using IPFilter)	`firewall-cmd --list-all`
System Hostname	View current hostname	`hostname`	`hostnamectl` or `hostname`
System Log Review	Check the system boot log	`tail /var/adm/syslog/sys`	`journalctl -b`