Comprehensive Arch Linux Installation and Optimization Guide for High-End Systems (2025)

Report Date: 2025-06-25

Introduction

This report provides a comprehensive, in-depth guide for the installation, optimization, and hardening of Arch Linux on a high-performance desktop system configured with 2025-era hardware. The target specification for this guide includes an AMD Ryzen 9 9950X processor, an ASRock X670E Taichi motherboard, an NVIDIA RTX 5090 graphics card, 64GB of DDR5 RAM, and a Norwegian keyboard layout. The objective is to deliver a detailed, step-by-step methodology that leverages the latest installation practices, hardware-specific performance tuning, robust security measures, and a complete development environment setup. The instructions are designed to be implemented directly or adapted into automated scripts, providing a definitive resource for power users and system administrators seeking to maximize the potential of their state-of-the-art hardware on the flexible and powerful Arch Linux platform. This document synthesizes expert knowledge and best practices to ensure a stable, secure, and highly optimized computing environment.

Pre-Installation and BIOS Configuration

Before commencing the Arch Linux installation, it is imperative to correctly configure the system's UEFI (BIOS) to ensure hardware compatibility, stability, and security. For the ASRock X670E Taichi motherboard, the first and most critical step is to update the BIOS to the latest available version. As of mid-2025, recent firmware releases contain crucial AMD AGESA patches that improve compatibility and performance for the Zen 5 architecture of the Ryzen 9 9950X CPU, as well as enhanced support for high-speed DDR5 memory and PCIe 5.0 devices. The update can be performed safely using the BIOS Flashback feature, which allows updating the firmware without a CPU or RAM installed, minimizing risk. Once the latest BIOS is installed, you should enter the setup utility to configure several key settings. Ensure the system is set to boot in pure UEFI mode by disabling the Compatibility Support Module (CSM). This is a prerequisite for modern features like Secure Boot and is the standard for contemporary operating systems. For users who may dual-boot with Windows 11 or wish to leverage certain security features, the firmware TPM (fTPM) must be enabled; this setting is typically found under the "Security" or "Trusted Computing" sections and may be labeled as "AMD CPU fTPM". To achieve optimal memory performance, navigate to the overclocking or memory settings and enable the appropriate EXPO profile for your 64GB DDR5 RAM kit, which will configure the correct speed, timings, and voltage, moving beyond the conservative JEDEC defaults. For advanced users planning to use virtualization with GPU passthrough (VFIO), it is essential to enable IOMMU support in the advanced chipset settings. Be aware that some BIOS versions for the X670E Taichi have exhibited issues with IOMMU groupings, potentially complicating the passthrough of certain NVMe drives or USB controllers. If such issues arise, consulting community forums for the most stable BIOS version for VFIO is recommended.

System Installation Using `archinstall`

With the BIOS correctly configured, the next phase is the installation of Arch Linux itself. The modern approach utilizes the `archinstall` guided installer, which is included on the official Arch Linux installation medium. As of 2025, `archinstall` has matured significantly, with recent versions like 3.0.8 introducing enhanced stability and features. To begin, boot from the latest Arch Linux ISO. It is a best practice to ensure the installer itself is up to date before proceeding; this can be done by connecting to the internet and running the command `pacman -Sy archin-`

`stall archlinux-keyring`. Once updated, launch the installer by typing `archinstall` at the command prompt. The guided menu will walk you through the necessary configuration steps. For the keyboard layout, select the Norwegian (`no`) keymap when prompted. This setting will be written to `/etc/vconsole.conf`, ensuring the correct layout is active in the virtual console. When you reach the disk configuration section, `archinstall` offers both automated and manual partitioning. While manual partitioning using tools like `cfdisk` provides maximum control, the automated options are robust. A highly recommended configuration for modern systems is to use the Btrfs filesystem with a subvolume layout. Recent versions of `archinstall` facilitate the post-installation setup of snapshot tools like Snapper, providing powerful system rollback capabilities, which is invaluable for a rolling-release distribution. During the installation, you will be prompted to select a profile, such as “desktop” or “minimal”. You can also specify additional packages to be installed. It is advisable to add essential tools like `git` and `base-devel` at this stage. For network configuration, NetworkManager is a versatile choice. If you configured a wireless network on the live medium, `archinstall` can copy this configuration to the new system. Finally, you will configure the root password and create a user account. Leaving the root password blank will disable the root account, enforcing the use of `sudo` for administrative tasks, which is a sound security practice. After the script completes the installation, it will prompt you to reboot into your new Arch Linux system.

Hardware-Specific Configuration and Optimization

After the initial installation, the system must be tuned to extract maximum performance from its high-end components. This involves installing the correct drivers and applying specific optimizations for the CPU, chipset, and GPU.

AMD Ryzen 9 9950X and X670 Chipset Optimization

To fully leverage the power of the AMD Ryzen 9 9950X CPU and the X670E chipset, several software-level optimizations are necessary. The foundation of this is running a recent Linux kernel; version 6.11 or newer is recommended to ensure robust support for the latest hardware features, including PCIe 5.0 and USB4. Arch Linux, being a rolling-release distribution, will typically provide a very current kernel. For performance tuning, one of the most impactful strategies is to use compiler flags tailored to the Zen 5 architecture. When compiling software from source or using repositories that provide optimized builds like CachyOS, specifying flags such as `-march=znver5 -O3` instructs the compiler to generate code that takes full advantage of the CPU’s instruction set, including AVX-512. This can yield significant performance gains in computationally intensive workloads. Beyond compilation, power management is crucial for both performance and efficiency. Ensure the `amd_pstate` driver is active, as it provides modern and granular control over CPU frequency scaling and power states, which is superior to the older ACPI-based drivers. You can monitor CPU performance and power states using tools like `turbostat` and `powertop`. Some users have reported that Linux’s default core parking behavior can sometimes impact latency-sensitive applications; this can be investigated and potentially mitigated with specific kernel parameters if performance issues are observed in certain tasks.

NVIDIA RTX 5090 Driver Installation

Properly installing the driver for the NVIDIA RTX 5090 is critical for graphics performance, from desktop compositing to gaming and GPU-accelerated computing. The recommended and most stable method for Arch Linux is to install the driver packages from the official repositories using `pacman`, as this ensures seamless integration with system updates and kernel upgrades. Avoid using the proprietary `.run` installer from NVIDIA’s website, as it can lead to system instability and complications during kernel updates. The RTX 5090, based on the Blackwell architecture, requires a recent driver branch, such as version 575 or newer. The installation process begins by installing the primary driver packages with the command `sudo pacman -S nvidia nvidia-utils nvidia-settings`. For 32-bit application support, which is necessary for many games and applications like Steam, you must also install the corresponding library from the multilib repository: `sudo pacman -S lib32-nvidia-utils`. A crucial step for modern Linux desktops, especially those using the Wayland display protocol, is to enable DRM kernel mode setting (KMS). This is achieved by adding the kernel parameter `nvidia_drm.modeset=1`. This parameter should be added to your bootloader configuration. To ensure the NVIDIA modules are loaded early in the boot process, which can

prevent display manager issues, you should add `nvidia`, `nvidia_modeset`, `nvidia_uvm`, and `nvidia_drm` to the `MODULES` array in `/etc/mkinitcpio.conf` and then regenerate the initramfs with `sudo mkinitcpio -P`. To automate this regeneration process after every driver update, a pacman hook can be created, ensuring the initramfs is always synchronized with the installed driver version.

MediaTek MT7927 Wireless Card: A Critical Incompatibility

A significant challenge with the specified hardware configuration is the MediaTek MT7927 wireless card. Based on extensive research of kernel and firmware development repositories as of mid-2025, this Wi-Fi 7 and Bluetooth 5.4 chipset is **not supported** by the mainline Linux kernel. The `mt76` driver family, which provides support for many other MediaTek wireless chips, has not yet incorporated drivers for the MT7927. Furthermore, the necessary firmware blobs are absent from the `linux-firmware` repository. This means that out of the box, neither the Wi-Fi nor the Bluetooth functionality of this card will work on Arch Linux. This is a critical hardware incompatibility that must be addressed for a functional system. The most practical and reliable solution is to physically replace the M.2 wireless card. A highly recommended alternative is an Intel AX210 card, which offers robust Wi-Fi 6E and Bluetooth 5.2 support with excellent, mature drivers that are included in the Linux kernel. Other supported MediaTek or Qualcomm cards are also viable options. Until official support for the MT7927 materializes in the kernel, which is not guaranteed on any specific timeline, replacing the hardware is the only effective path forward to achieve wireless connectivity on this system.

Security Hardening

Building a secure system is as important as optimizing its performance. Arch Linux provides a minimal base that can be hardened through a multi-layered approach, encompassing boot integrity, access control, and network security.

Implementing Secure Boot

Secure Boot is a UEFI feature that ensures only cryptographically signed and trusted software is loaded during the boot process, preventing bootkit and rootkit attacks. While historically complex to set up on Arch Linux, modern tools have streamlined the process. The `sbctl` utility offers a user-friendly command-line interface for managing Secure Boot keys and signing boot components. The process begins by entering your motherboard's UEFI setup and putting Secure Boot into "Setup Mode," which typically involves clearing the existing platform keys. Once booted back into Arch Linux, you install `sbctl` and use it to create your own set of keys with `sudo sbctl create-keys`. You then enroll these keys into the firmware. It is highly recommended to enroll Microsoft's keys alongside your own by using the `-m` flag (`sudo sbctl enroll-keys -m`), which maintains compatibility for dual-booting with Windows and ensures that firmware for peripherals like the GPU can be properly authenticated. After enrolling the keys, you must sign all necessary boot files, including your bootloader (e.g., systemd-boot or GRUB) and the Linux kernel image (`vmlinuz-linux`). The `sbctl sign -s /path/to/file` command is used for this. To maintain security across updates, `sbctl` provides a pacman hook that automatically signs newly installed kernels and bootloaders. Once all components are signed, you can reboot, re-enter the UEFI setup, and enable Secure Boot.

General System Hardening Practices

Beyond boot security, a comprehensive hardening strategy involves securing user access, the network, and the kernel itself. Start by enforcing strong password policies using `pam_pwquality`. By editing `/etc/pam.d/passwd`, you can mandate minimum password length, complexity requirements, and prevent the use of dictionary words. For remote access, SSH should be hardened significantly. Disable root login (`PermitRootLogin no`), disable password-based authentication in favor of SSH key pairs, and consider changing the default port. Tools like `fail2ban` can be deployed to automatically block IP addresses that exhibit malicious behavior, such as repeated failed login attempts. Network security should be managed with a firewall. `ufw` (Uncomplicated Firewall) is a simple yet effective choice for defining a default-deny policy and only allowing traffic on specific, necessary ports. At the kernel level, Arch Linux benefits from upstream hardening features. You can further enhance this by installing and configuring a

Mandatory Access Control (MAC) system like AppArmor, which confines applications to a specific set of permissions, limiting the damage an exploited program can cause. Regularly auditing your system's security posture with tools like `lynis` can help identify potential misconfigurations and vulnerabilities. Finally, maintaining system security is an ongoing process that requires diligence in applying updates promptly with `sudo pacman -Syu` to patch known vulnerabilities as they are discovered.

Development Environment Setup

Arch Linux is an exceptional platform for software development due to its access to the latest tools and libraries. Setting up a productive environment begins with installing the `base-devel` package group, which contains essential build tools like `make` and `gcc`, and `git` for version control. To manage packages from the Arch User Repository (AUR), an AUR helper is indispensable; `yay` is a popular and powerful choice. It can be installed by cloning its git repository and building it with `makepkg`. With the basics in place, you can install language-specific toolchains. For Python, it is best practice to use `pyenv` to manage multiple Python versions, allowing you to switch between interpreters on a per-project basis. Similarly, for Node.js, `nvm` (Node Version Manager) provides the same flexibility. For compiled languages, you can install the latest toolchains directly from the repositories, such as `go` for Golang and `rustup` for Rust. For containerized development, Docker is the industry standard. Install it with `sudo pacman -S docker docker-compose`, enable the Docker service with `sudo systemctl enable --now docker`, and add your user to the `docker` group to manage containers without `sudo`. Your choice of text editor or IDE is personal, but both Visual Studio Code (`visual-studio-code-bin` from the AUR) and Neovim (`neovim`) are excellent, highly extensible options. To enhance the command-line experience, consider replacing the default bash shell with Zsh, augmented with a framework like Oh My Zsh for powerful plugins and themes. Tools like `tmux` for terminal multiplexing, `fzf` for fuzzy finding, and `starship` for a rich, informative shell prompt can dramatically improve workflow efficiency.

Post-Installation Maintenance and Package Management

Maintaining an Arch Linux system is straightforward but requires regular attention. The primary tool for this is `pacman`. The most frequent command you will use is `sudo pacman -Syu`, which synchronizes the local package database with the remote repositories and upgrades all installed packages. It is crucial to run this command regularly to keep the system secure and stable. Occasionally, package updates may introduce conflicts, often because two packages attempt to provide the same file. `pacman` will report these conflicts and halt the transaction. You can often resolve these by first removing the conflicting package or, if you are certain it is safe, using the `--overwrite` flag to force `pacman` to overwrite the conflicting file. For example, `sudo pacman -Syu --overwrite '/path/to/conflicting/file'`. Over time, the `pacman` cache can grow large. You can clean it safely with `sudo pacman -Sc`, which removes old package versions that are no longer installed. To keep the system lean, it is also good practice to periodically remove orphaned packages—dependencies that were installed for another package but are no longer required. This can be done with the command `sudo pacman -Rns $(pacman -Qtdq)`. For system-wide configuration, such as the Norwegian keyboard layout in the graphical environment, this is typically handled by your chosen desktop environment's settings panel or by using `localectl set-x11-keymap no` for a system-wide Xorg configuration.

Conclusion

This report has detailed a comprehensive methodology for installing, optimizing, and securing Arch Linux on a state-of-the-art hardware platform featuring the AMD Ryzen 9 9950X and NVIDIA RTX 5090. By following the prescribed steps, from initial BIOS configuration and `archinstall` execution to hardware-specific driver installation and performance tuning, users can establish a highly performant and stable system. The guide emphasized the importance of security through the implementation of Secure Boot and general system hardening practices, creating a resilient

foundation for any workload. Furthermore, the instructions for setting up a modern development environment provide a clear path to a productive workspace. A critical finding of this analysis is the current lack of Linux support for the MediaTek MT7927 wireless card, a significant incompatibility that necessitates a hardware replacement for wireless functionality. By addressing this issue and diligently applying the optimization and maintenance strategies outlined, the user can fully harness the formidable capabilities of their high-end system, resulting in a powerful, secure, and customized Arch Linux experience tailored for the demands of 2025 and beyond.

References

archinstall - ArchWiki (https://wiki.archlinux.org/title/Install_Arch_Linux_with_menu-guided_archinstall)
archinstall 3.0.7 Release Notes (https://github.com/archlinux/archinstall/releases/tag/3.0.7)
archinstall 3.0.8 Release Details (httpshttps://linuxiac.com/archinstall-3-0-8-breaking-changes-bug-fixes-and-new-features/)
Community Guides and GitHub Discussions (https://github.com/archlinux/archinstall)
Official Arch Linux Installation Guide (https://wiki.archlinux.org/index.php/Installation_guide)
ArchWiki NVIDIA Driver Page (https://wiki.archlinux.org/title/NVIDIA)
GitHub NVIDIA open GPU kernel modules (https://github.com/NVIDIA/open-gpu-kernel-modules)
NVIDIA Developer Forums (https://forums.developer.nvidia.com/t/rtx-50-series-blackwell-gpu-drivers-on-linux/335669)
Manjaro Linux Forum - MediaTek Wi-Fi 7 MT7927 (https://forum.manjaro.org/t/mediatek-wi-fi-7-mt7927/154995)
Ask Ubuntu - No driver available for MediaTek MT7927 (https://askubuntu.com/questions/1535319/no-driver-available-for-mediatek-mt7927-wifi-or-bluetooth-on-ubuntu-24-10-with-k)
Linux Wireless Documentation (https://wireless.docs.kernel.org/en/latest/en/users/drivers/mediatek.html)
OpenWrt Support for MT7927 (https://github.com/openwrt/mt76/issues/927)
MX Linux Forum - Support for MT7927 (https://forum.mxlinux.org/viewtopic.php?t=82139)
Arch Linux Forum - Support Status (https://bbs.archlinux.org/viewtopic.php?id=303402)
Fedora Discussion - Bluetooth Support (https://discussion.fedoraproject.org/t/mt7927-bluetooth-support-in-kernel/147702)
Git Kernel Repository (https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/)
Phoronix Benchmarking and Performance Analysis (https://fudzilla.net/news/59573-amd-s-latest-ryzen-9-9950x-goes-like-the-clappers-on-linux)
Support for AMD X670E and Ryzen 9950X (https://forums.opensuse.org/t/does-linux-support-chipset-x870e-and-cpu-ryzen-9950x/178253/)
Motherboard Compatibility and Chipsets (https://www.techreviewer.com/tech-specs/chipsets-for-amd-9950x/)
Core Parking and Linux Performance (https://forum.level1techs.com/t/9950x-core-parking-linux-vs-windows/215056)
Arch Wiki Secure Boot (https://wiki.archlinux.org/title/Unified_Extensible_Firmware_Interface/Secure_Boot)
Arch Install Guide 2025 - PART 1 (https://gist.github.com/mark-akturatech/577e78ecfe33a5dd6386e46a4cfb833a)
Setup Guide - Binary is better (https://binary-manu.github.io/binary-is-better/linux/archlinux-secure-boot)
How to - Enable/Set-up secure boot on Arch-based distributions (https://ewilcox.blogspot.com/2025/01/how-to-enableset-up-secure-boot-on-arch.html)
Secure Boot with shim and sbctl (https://github.com/Foxboron/sbctl)
ArchWiki Security (https://wiki.archlinux.org/title/Security)
Develop and Solve: Arch Linux Hardening (https://developnsolve.com/a-beginners-guide-to-arch-linux-hardening)
Linux Kernel Runtime Guard (https://vez.mrsk.me/linux-hardening)
Optimizing Linux Security in 2025 (https://linuxsecurity.com/features/effective-strategies-to-optimize-linux-security-in-2025)
Netwrix Linux Hardening Guide (https://www.netwrix.com/linux_hardening_security_best_practices.html)
Maalavs Blog: Linux Security and Hardening (https://blogs.maalavs.com/linux/linux-security-and-hardening-guide/)
ArchWiki - Linux console/Keyboard configuration (https://wiki.archlinux.org/title/Linux_console/Keyboard_configuration)

KBD Layout Info (https://kbdlayout.info/KBDNO/)
Xmodmap configuration (https://wiki.archlinux.org/index.php/Xmodmap)
Reddit - Persistent keyboard layout (https://www.reddit.com/r/linuxquestions/comments/7rkt53/arch_linux_how_to_set_keyboard_layout_permanently/)
Arch Linux Wiki: Pacman (https://wiki.archlinux.org/title/Pacman?pubDate=20250618)
Arch Linux Forums: Conflicting Packages and Dependencies (https://bbs.archlinux.org/viewtopic.php?id=245967)
It's FOSS Community: Conflicting Files in Pacman (https://itsfoss.community/t/conflicting-files-in-pacman-arch-linux/10706)
Setting Up the Perfect Developer Environment on Linux for 2025 (https://najad.dev/blog/setting-up-the-perfect-developer-environment-on-linux-for-2025/)
How to Setup an Arch Linux Environment for Coding - HackerNoon (https://hackernoon.com/how-to-setup-an-arch-linux-environment-for-coding)
Building a Lean Development Environment on Arch Linux (https://chrisworth.dev/posts/arch-linux-development-environment-guide/)
Level1Techs Forum - Ryzen 7950X / X670E Taichi VFIO Success (https://forum.level1techs.com/t/ryzen-7950x-x670e-taichi-vfio-success-with-notes/189567)
ASRock BIOS Download and Setup Guide (https://download.asrock.com/Manual/Software/AMD%20X670/Software_BIOS%20Setup%20Guide_English.pdf)
ASRock Official Website (https://www.asrock.com/mb/AMD/X670E%20Taichi/index.asp)