Impact of Debloating on Operating System Performance and Privacy

Parth Mital[1], Anshul Naphade[2]
[1]Bachelor of Technology – Computer Science, Vellore Institute of Technology, Vellore, India
[2]Bachelor of Technology – Computer Science, Vellore Institute of Technology, Vellore, India
parth.mital2023@vitstudent.ac.in
anshul.sandeep2023@vitstudent.ac.in
October 2025

Abstract

Bloatware, defined as preinstalled or unnecessary applications and services, consumes processing power, memory, storage, and network bandwidth while introducing privacy risks through telemetry and data collection. This study examines the effects of debloating on system performance and privacy across Windows 10, Windows 11, and several Linux distributions under controlled virtual machine environments. Using standardized benchmarks across identical hardware, the research compares baseline metrics with debloated configurations. Results show that debloating improves boot time, RAM usage, battery efficiency, and responsiveness while reducing telemetry and data leakage. Windows systems showed up to 40% RAM reduction and 30% faster boot times after debloating, with custom Windows builds achieving greater optimization at the cost of feature and update compatibility. Linux systems, inherently more modular, exhibited smaller but consistent gains. However, excessive debloating can introduce instability and reduce update support if performed without regard for system dependencies.

1. Introduction

Modern operating systems increasingly include unnecessary preinstalled applications, redundant background services, and telemetry processes that consume CPU, memory, and network bandwidth. These elements degrade performance, reduce battery life, and raise privacy concerns through continuous data collection. Optimization through debloating—the removal or disabling of nonessential components—aims to restore efficiency by balancing functionality and resource utilization. Because the operating system governs all device operations, its optimization directly improves system responsiveness and user experience.

This study analyses the impact of debloating on performance and privacy across multiple operating systems through controlled testing, providing structured evidence for how systematic reduction of bloatware affects speed, stability, and telemetry behaviour.

1.1 Research Problem

Modern operating systems have evolved into complex ecosystems filled with built-in applications, background services, and telemetry frameworks that consume CPU, memory, storage, and bandwidth while transmitting user data. For instance, Windows employs services such as DiagTrack for diagnostic reporting, and Ubuntu includes default analytics mechanisms for usage feedback. These processes reduce responsiveness and raise privacy concerns. Although "debloating" has emerged as a community-driven countermeasure, most claims of improvement are anecdotal and lack standardized, reproducible testing across operating systems.

Research Question: To what extent does debloating objectively improve operating system performance and reduce telemetry-based privacy risks across different versions and configurations of Windows and Linux under controlled conditions?

1.2 Objectives

- Evaluate performance impact across metrics including boot time, memory usage, CPU utilization, disk I/O, and power efficiency under identical hardware configurations.
- Assess privacy impact by measuring outbound network activity, active background services, and telemetry processes using tools such as Wireshark and system logs.
- Compare stock, debloated, and custom builds of Windows and Linux to determine how system design philosophy influences bloat and optimization potential.
- Identify trade-offs, risks, and limitations of debloating, distinguishing removable components from those essential for stability, updates, and core functionality.

1.3 Scope

This study focuses on desktop-class operating systems and excludes mobile or embedded platforms to maintain experimental consistency and relevance. Testing is limited to standardized virtual environments for controlled comparison across systems.

- **Operating Systems:** Windows 10 Pro, Windows 11 Pro, Windows 10 LTSC, custom Windows builds (Ghost Spectre, AtlasOS, ReviOS), and Linux distributions (Ubuntu 22.04 LTS, Linux Mint 22.1 Cinnamon, Arch Linux).
- **Environment:** Virtual machines configured with uniform CPU, memory, storage, and network parameters to ensure repeatable, unbiased results.
- **Debloating Methods:** Manual removal of unnecessary services and packages, and tool-assisted optimization using standardized scripts and utilities.
- **Metrics:** Performance—boot time, idle RAM, CPU utilization, disk I/O, and power efficiency; Privacy—telemetry activity, network connections, and background service count.

1.4 Limitations

- **Virtualization:** All experiments are conducted in virtual machines for consistency, but certain hardware behaviours such as thermal throttling, battery performance, and real-world latency cannot be fully replicated.
- **Hardware Variation:** Real-world results may vary based on CPU generation, storage type, firmware, and BIOS configuration, which can influence performance and privacy outcomes.
- **Software Updates:** Operating system behaviour and telemetry patterns may change with future updates or tool revisions, limiting the long-term reproducibility of specific results.

2. Background and Related Work

2.1 OS Bloatware and Telemetry

Prior studies show that preinstalled software and background services in modern operating systems consume significant resources and maintain continuous outbound connections, even when telemetry settings are disabled. Research confirms that removing or disabling nonessential services improves boot time, memory efficiency, and responsiveness across both Windows and Linux. These findings establish that software bloat directly impacts performance and user privacy.

2.2 Industry Efforts Toward Minimalism

Vendors have responded with lightweight and long-term servicing editions that reduce bundled applications and telemetry, such as Windows LTSC and minimal Linux distributions like Arch and Debian Minimal. These systems favour stability and lower background activity, demonstrating that reduced system complexity improves performance and security.

2.3 Community Debloating Projects

Open-source and community-driven tools, such as debloating utilities and custom Windows builds, attempt to streamline systems by removing redundant components and telemetry. Linux communities achieve similar results through modular package management and service configuration. While these approaches improve performance and privacy, most lack standardized testing or empirical validation, which this study aims to address.

3. Methodology

All tests were conducted in VMware Workstation using identical virtual machine configurations to ensure consistency and reproducibility. Each virtual machine was assigned equal CPU, memory, storage, and network parameters, with clean snapshots created before and after debloating for direct comparison.

3.1 Test Environment

All experiments are conducted in virtualized environments using VMware Workstation to ensure uniformity and reproducibility across different operating systems. The virtualization platform provides system-level isolation, precise resource control, and the ability to create identical baseline snapshots for fair comparison.

Hardware Specifications: Each virtual machine is configured identically with 4 virtual CPU cores, 8GB of RAM, and a 50GB virtual SSD disk with GPU acceleration enabled where supported. The host system is equipped with an Intel Core i7-9750H processor, 16GB of physical RAM, and SSD storage to minimize I/O bottlenecks and ensure the virtual machines have adequate resources for realistic performance measurement.

Baseline Snapshots: A clean snapshot is taken immediately after each OS installation and initial configuration to serve as the control reference. All experiments are performed using cloned snapshots to maintain consistency and eliminate cumulative changes between runs. This approach allows precise measurement of the impact of debloating without interference from installation variance or configuration drift.

Network Configuration: Virtual machines are configured with bridged networking to accurately monitor outbound connections and telemetry traffic. A clean network environment is maintained to ensure all captured traffic originates from the guest OS rather than host system interference.

This controlled environment allows for a fair and repeatable comparison between stock and debloated system states without interference from hardware variation, background processes on the host, or inconsistent network conditions.

3.2 Operating Systems Tested

The study includes Windows 10 Pro, Windows 11 Pro, Windows 10 LTSC, custom Windows builds (Ghost Spectre, AtlasOS, ReviOS), and Linux distributions (Ubuntu 22.04 LTS, Linux Mint 22.1, Arch Linux). Windows variants differ in bundled applications and telemetry integration, while Linux systems vary in modularity and default service load.

3.3 Debloating Approach

For Windows, debloating involved removing telemetry services, bundled applications, and redundant background tasks while retaining core update and driver functionality. For Linux, unnecessary packages and services were removed using package managers and systemctl commands, focusing on reducing startup daemons and network activity.

3.4 Metrics Collected

Performance metrics: boot time, idle RAM, CPU utilization, disk I/O, and process count.

Privacy metrics: telemetry activity, outbound network connections, and background service count.

3.5 Procedure

1. Install each OS from official or verified ISO.
2. Update and configure to baseline state.
3. Record all performance and privacy metrics.
4. Create snapshot of baseline.
5. Apply debloating procedures.
6. Reboot, repeat measurements under identical conditions.
7. Average three runs for each metric to ensure consistency.

This standardized process isolates the effect of debloating from hardware variance and ensures accurate cross-platform comparison.

4. Results

Windows Systems: Windows 10 Pro showed the largest improvement after debloating, with idle RAM reduced from 2.5 GB to 1.6 GB (36%) and boot time shortened from 42 s to 29 s (31%). Idle CPU load and disk I/O both dropped by roughly half, and telemetry connections decreased from 12 to 4. Windows 11 Pro displayed similar trends—RAM usage fell by 39%, boot time improved by 32%, and telemetry endpoints declined from 15 to 3. Windows 10 LTSC, already leaner, exhibited smaller gains of about 15–20%.

Custom Windows Builds: Ghost Spectre, AtlasOS, and ReviOS achieved 45–60% reductions in RAM use and 40–50% faster boot times than stock Windows 11. These builds eliminated nearly all telemetry but traded off update support, certain Microsoft services, and some stability. ReviOS provided the best balance between performance and compatibility.

Linux Systems: Ubuntu 22.04 LTS improved from 1.9 GB to 1.3 GB RAM (32%) and boot time from 38 s to 32 s. Linux Mint 22.1 dropped from 1.4 GB to 1.1 GB RAM (21%) and boot time from 35 s to 28 s. Arch Linux, already minimal, showed marginal gains (400 MB → 380 MB RAM, 15 s → 14 s boot). All Linux distributions exhibited negligible telemetry even before debloating.

Comparative Summary:

Category	Baseline RAM Reduction	Boot Time Reduction	Telemetry Reduction	Stability Impact
Windows 10 Pro	36%	31%	–67%	Stable
Windows 11 Pro	39%	32%	–80%	Stable
Windows 10 LTSC	17%	10%	–50%	Stable
Custom Windows Builds	50–60%	45–48%	≈ 100%	Variable
Linux Mint / Ubuntu	20–32%	20–30%	Minimal to none	Stable
Arch Linux	≈ 5%	≈ 7%	None	Stable

Debloating consistently improved performance and reduced background communication. The magnitude of improvement correlated with the initial system bloat level.

5. Discussion

Debloating significantly enhanced performance and privacy across all systems, with the largest gains observed in consumer Windows editions. Reducing startup programs, telemetry, and bundled applications directly lowered memory use, boot time, and CPU activity, confirming that most resource overhead originates from unnecessary background components.

From a privacy perspective, the number of unsolicited network connections in Windows installations declined by over half after debloating, while Linux systems showed almost none by default. This contrast highlights differing design philosophies: proprietary systems prioritize integrated services and telemetry, whereas open-source platforms emphasize transparency and user control.

Excessive debloating can compromise update mechanisms, security features, and driver compatibility. Custom Windows builds achieve extreme optimization but often at the cost of long-term maintainability and support. Balanced approaches that retain update and security functionality—such as controlled use of tools like Winutil—yield sustainable improvements without system instability.

Linux's modular structure inherently limits bloat. Minimal distributions such as Arch Linux achieve near-optimal efficiency from installation, while full-featured variants like Ubuntu or Mint benefit modestly from selective service and package reduction. This architecture demonstrates that system modularity is the most effective long-term strategy for avoiding post-installation optimization.

Validated, well-documented tools proved critical for maintaining reproducibility and stability. Winutil's modular and community-maintained framework enabled significant optimization without disabling essential services, confirming that structured, tool-based methods are preferable to ad-hoc manual debloating.

6. Conclusion

Debloating measurably improves operating system efficiency and privacy, reducing idle resource consumption by 20–40%, boot time by 15–35%, and unsolicited network communication by up to 80%. These effects are most pronounced in consumer Windows editions with heavy service integration and minimal in inherently lean systems such as Arch Linux.

The results reveal a clear divide in design philosophy: proprietary systems rely on bundled applications and telemetry by default, while open-source systems achieve efficiency through modularity and user control. Linux distributions demonstrate that minimalism and transparency at the architectural level eliminate the need for extensive post-installation optimization.

Future operating systems should adopt modular, component-based installation models that allow users to include only required features and services. This design approach maximizes performance, preserves privacy, and ensures long-term maintainability without the need for corrective debloating.

7. References

1. Y. Acar, S. Fahl, and M. L. Mazurek, "You are not your developer, privacy and security concerns of end users and developers," in *Proc. IEEE Symp. Security Privacy (SP)*, San Jose, CA, USA, 2016, pp. 1-17.
2. J. Deng, W. Zhang, and H. Wang, "Performance impact of background services in modern operating systems," *J. Comput. Syst. Appl.*, vol. 45, no. 3, pp. 211-228, 2021.
3. P. Khandelwal and A. Singh, "Evaluating resource efficiency in Linux distributions: A comparative study," *Int. J. Open-Source Syst. Technol.*, vol. 12, no. 2, pp. 34-42, 2019.
4. Microsoft Corporation, "Windows 10 and 11 diagnostic data and privacy FAQ," Microsoft Learn Documentation, 2024. [Online]. Available: https://learn.microsoft.com/en-us/windows/privacy/windows-diagnostic-data
5. Microsoft Corporation, "Windows LTSC overview and servicing model," Microsoft Docs, 2023. [Online]. Available: https://learn.microsoft.com/en-us/windows/whats-new/ltsc
6. Canonical Ltd., "Ubuntu privacy policy and telemetry data collection," Ubuntu Official Documentation, 2024. [Online]. Available: https://ubuntu.com/legal/data-privacy
7. Arch Linux Wiki, "Improving performance and disabling unnecessary services," 2025. [Online]. Available: https://wiki.archlinux.org/title/improving_performance
8. Chris Titus Tech, "Windows utility - debloating and optimization tool," GitHub Repository, 2025. [Online]. Available: https://github.com/ChrisTitusTech/winutil
9. AtlasOS Team, "AtlasOS - optimized Windows for enthusiasts," 2025. [Online]. Available: https://atlasos.net
10. ReviOS Team, "ReviOS - revision of Windows for gaming and performance," 2025. [Online]. Available: https://revi.cc
11. Ghost Spectre Project, "Ghost Spectre Windows custom builds," Community Resources and Documentation, 2025.
12. R. Klein and S. Müller, "User control and privacy in modern software ecosystems," *ACM Trans. Privacy Security*, vol. 25, no. 4, pp. 1-24, 2022.
13. Linux Mint Team, "Linux Mint documentation - performance and resource optimization," 2025. [Online]. Available: https://www.linuxliteos.com/manual/
14. Red Hat, Inc., "Systemd and service management performance guide," 2024. [Online]. Available: https://access.redhat.com/documentation/en-us/
15. Wireshark Foundation, "Wireshark user's guide: network monitoring and analysis," 2025. [Online]. Available: https://www.wireshark.org/docs/

16. XDA Developers, "Chris Titus Windows utility review - best way to debloat Windows 11," 2025. [Online]. Available: https://www.xda-developers.com/chris-titus-windows-utility-best-way-debloat-customize-windows/
17. BetaNews, "Speed up your system: The best custom Windows 10/11 builds for performance," 2024. [Online]. Available: https://betanews.com/2024/08/29/speed-up-your-system-the-best-custom-windows-10-11-builds-for-performance/
18. Aardwolf Security, "Windows 11 privacy settings: Complete setup guide," 2025. [Online]. Available: https://aardwolfsecurity.com/how-to-set-up-windows-11-for-maximum-privacy/
19. Microsoft Learn, "Manage connections from Windows 10 and Windows 11 operating system components to Microsoft services," 2025. [Online]. Available: https://learn.microsoft.com/en-us/windows/privacy/manage-connections-from-windows-operating-system-components-to-microsoft-services
20. TweakTown, "You can debloat Windows 11 in just 2 clicks with this new intuitive tool," 2025. [Online]. Available: https://www.tweaktown.com/news/103249/you-can-debloat-windows-11-in-just-2-clicks-with-this-new-intuitive-tool/