TUTORIAL

Configure Windows Server 2025 Secure Boot and BitLocker Drive Encryption

Learn how to enable Secure Boot and BitLocker on Windows Server 2025 with a practical workflow covering prerequisites, implementation, validation, and operational checks.

Operating Systems - August 03, 2026

Why this matters on a server build

A server that boots without firmware trust checks and stores data on an unencrypted volume is exposed to a class of risks that are difficult to detect after the fact: offline tampering, boot-chain manipulation, and data exposure if the machine or disk is removed. Secure Boot and BitLocker solve different problems, and they work best together. Secure Boot helps verify that the boot path has not been altered before Windows loads. BitLocker protects the OS volume and, when configured correctly, can keep the data unreadable if the disk is moved to another system or the server is powered off.

After reading this tutorial, you should be able to decide whether the approach fits your server, prepare the firmware and recovery workflow, enable Secure Boot and BitLocker in a controlled way, validate that protection is active, and confirm what to verify before production use.

What you are building

The finished state is a Windows Server 2025 system that:

- Boots with UEFI Secure Boot enabled and confirmed active.
- Protects the operating system volume with BitLocker.
- Uses a recoverable key management process, not an ad hoc password or undocumented file.
- Can be validated remotely or locally with clear evidence that encryption is on and the protector state is known.

This tutorial assumes you are configuring a standard server installation, not redesigning storage from scratch. If you are also hardening the wider OS build, it is worth pairing this work with Windows Server 2025 Hardening: Secure Baseline Configuration Guide and, where applicable, Windows Server 2025 Hardening: Disable Legacy Protocols and Services.

Prerequisites and stop-here checks

Before you change anything, confirm the platform can support the security model you want.

Goal

Avoid enabling encryption on a system that cannot support it cleanly, or on a host that has not been prepared for recovery.

Action

Verify the following:

- The machine boots in UEFI mode, not legacy BIOS mode.
- The firmware exposes Secure Boot controls.
- TPM is present and enabled, preferably TPM 2.0.
- You have console access or out-of-band management in case a boot issue occurs.
- You know how recovery keys will be captured, stored, and retrieved.
- You have an approved maintenance window if this is a production host.

Stop-here-if warnings

Stop here if any of the following are true:

- The server is still configured for legacy BIOS boot.
- You cannot verify TPM status.
- You do not have a secure place to store recovery keys.
- You cannot tolerate a reboot or potential pre-boot prompt during validation.

BitLocker can be deployed with different protector combinations, but operationally you should not proceed until you know what will unlock the machine after a firmware change, hardware replacement, or recovery event.

Expected output

You should end this step with a checklist that confirms firmware mode, TPM availability, management access, and key custody.

Validation

From Windows, run:

Confirm-SecureBootUEFI should return True on a Secure Boot-enabled system. Get-Tpm should show the TPM as present and ready, or at minimum reveal why it is not usable yet.

Common failure

The most common failure is discovering that the server booted in legacy mode or that Secure Boot is disabled in firmware. In that case, do not force the issue from Windows. Reconfigure firmware/boot mode first, because BitLocker deployment decisions depend on the platform boot path.

Prepare the firmware and boot path

Goal

Establish a trusted boot path before encrypting the volume.

Action

Reboot into firmware setup and verify these settings:

- Boot mode is UEFI.
- Secure Boot is enabled.
- TPM is enabled and, if needed, activated or cleared according to your device and policy requirements.
- The boot order points to the correct OS disk.

If the system was previously installed in legacy mode, you may need to convert the disk and reinstall or migrate carefully, depending on your operational standard. Do not assume Secure Boot can be enabled later without boot-mode alignment.

Expected output

The server should boot cleanly into Windows with UEFI Secure Boot available and ready to validate.

Validation

After booting Windows, confirm Secure Boot again:

You can also inspect system information to confirm the firmware path:

In System Information, look for BIOS mode set to UEFI and Secure Boot state set to On.

Common failure

A frequent issue is a firmware configuration that advertises Secure Boot but still boots the operating system through a legacy or incompatible path. Another common issue is an out-of-date firmware database or missing firmware update that prevents a clean enablement. If firmware changes are required, apply them before moving on.

Validate TPM and decide on the protector model

Goal

Choose a BitLocker configuration that matches the server?s operational role and unlock requirements.

Action

Confirm that the TPM is available:

For most server builds, a TPM-backed protector is the simplest and most supportable option. Depending on your recovery and access model, you may also add a recovery key protector so that the volume can be recovered if TPM measurements change after hardware or firmware work.

If your environment has stricter operational controls, decide now whether recovery keys will be escrowed in a directory service, saved to a managed vault, or stored through another approved process. The exact storage method is an organizational decision, but the rule is the same: do not encrypt a server unless recovery is already designed.

Expected output

You should have a protector plan that states how the drive will unlock normally and how it will be recovered if the normal path fails.

Validation

Use TPM status to confirm readiness and, if needed, verify that the system is not reporting a disabled or unavailable TPM.

Common failure

The common failure here is mixing design and implementation. For example, enabling BitLocker first and trying to sort out recovery later creates avoidable risk. Another failure is assuming that a TPM automatically means recovery is solved; it does not.

Enable BitLocker on the OS volume

Goal

Turn on encryption for the operating system drive without breaking bootability.

Action

From an elevated PowerShell session, enable BitLocker on the OS volume. A common pattern is to protect the drive with the TPM and add a recovery key protector.

Example:

Depending on your policy, you may also want to specify encryption method or additional protectors. The key operational point is not the exact syntax, but that you deliberately define the unlock and recovery path instead of relying on defaults you have not reviewed.

If your environment requires pre-provisioning or staged rollout, consider whether encryption should be turned on during imaging, during first boot, or during a controlled post-build window. The right timing depends on how much data is already present and how much downtime you can tolerate.

Expected output

BitLocker should begin protecting the OS volume, and Windows should report an active protector state.

Validation

Check status with:

Look for the volume encryption status, percentage encrypted, and the list of key protectors. You want to see that the OS volume is protected and that at least one valid recovery method exists.

Common failure

Typical failures include a missing TPM, a firmware mismatch after Secure Boot changes, or a recovery protector not being created. If the command fails, check the exact error before retrying; repeated changes without understanding the failure can complicate recovery.

Confirm Secure Boot is actually active

Goal

Verify that the machine is not only configured for Secure Boot, but also running with it enabled.

Action

Run the Secure Boot check from Windows:

You can also verify in System Information or through your hardware management interface if you have one.

Expected output

The result should confirm Secure Boot is enabled and active on the current boot path.

Validation

Use at least two evidence points if this is a production server:

- PowerShell confirmation.
- Firmware or system information confirmation.

This is especially useful after firmware updates, boot-order changes, or motherboard replacements.

Common failure

The common failure is assuming the setting persisted after a reboot or after a firmware reset. Secure Boot can be disabled unintentionally when firmware defaults are restored, so validation should be part of change control, not a one-time setup check.

Verify BitLocker protection and recovery readiness

Goal

Ensure the OS volume is encrypted, the protector set is known, and recovery is possible.

Action

Run:

Review these items:

- Encryption percentage is complete or progressing as expected.
- Protection status indicates protection is on.
- Key protectors include the intended TPM and recovery components.
- Recovery information has been stored in the approved location.

If your policy requires it, back up the recovery key before closing the change window. The important point is to confirm that the recovery artifact exists and is accessible to the right operators.

Expected output

You should have evidence that the drive is encrypted and that recovery is possible without depending on a single person or a single local copy.

Validation

A practical validation pattern is to document the protector IDs and the recovery storage location, then test retrieval through the approved process without exposing the actual key in logs or chat.

Common failure

The most common failure is treating ?BitLocker enabled? as sufficient proof. A server is not operationally ready until you have verified how recovery works and who can execute it.

Operational follow-up after deployment

Goal

Keep the configuration supportable after firmware updates, hardware changes, or incident recovery.

Action

Add these checks to your operational routine:

- Confirm Secure Boot after firmware changes.
- Confirm BitLocker status after hardware maintenance.
- Store recovery key access in a controlled, auditable process.
- Review whether any maintenance workflow could trigger recovery mode.
- Document the baseline so future operators know what ?healthy? looks like.

For a broader hardening program, pair this work with your baseline control checks so that encryption and boot trust are not managed in isolation. If you are reducing the attack surface further, a companion hardening plan such as Windows Server 2025 Security Baseline Hardening for Zero Trust can help align the build with a stricter trust model.

Expected output

The server remains bootable, the protection status stays on, and you have a repeatable process for validating both controls after change events.

Validation

Use a simple post-change checklist:

- Secure Boot still reports enabled.
- BitLocker still reports protection on.
- Recovery key access still works through the approved path.
- No unexpected boot prompts appeared during validation.

Common failure

The common operational failure is failing to revalidate after legitimate maintenance. Firmware updates, storage controller changes, and motherboard replacements can alter measurements or boot behavior, so the server should be checked again whenever those events occur.

A practical deployment rule

If you only remember one rule from this tutorial, use this: do not enable BitLocker until you have already proven the platform boots with Secure Boot, the TPM is ready, and recovery is documented. That sequence keeps the server recoverable while still giving you the protections you expect from a trusted boot chain and disk encryption.

When the setup is complete, your evidence should be straightforward: firmware is set to UEFI with Secure Boot on, the OS volume reports BitLocker protection, and recovery can be executed without guesswork.

Use this guidance together with RHEL vulnerability scanning and harden SSH access on RHEL to connect the workflow with related operational context already available on the site.