GUIDE

AWS SDK for .NET V3: End-of-Support Announcement & V4 Migration Guide

> 🚨 **CRITICAL** — Security fix. Upgrade immediately.
>
> **Version:** V3 | **Released:** 2026-03-01 | **Upgrade from:** unknown

Release at a Glance

The AWS SDK for .NET V3 is nearing its end-of-support. This isn't just a version bump; it's a critical call to action for every developer leveraging AWS services with .NET.

- **End-of-Support**: AWS SDK for .NET V3 enters maintenance mode on March 1, 2026, and reaches **End-of-Support on June 1, 2026**.
- **Security Risk**: After June 1, 2026, V3 will receive no further updates, including critical security patches, leaving applications vulnerable.
- **Mandatory Upgrade**: All V3 users must migrate to AWS SDK for .NET V4 to maintain security, receive ongoing support, and benefit from significant improvements.
- **Key Benefits of V4**: Enjoy substantial performance gains (e.g., `AWSSDK.Core` assembly size reduced from 2MB to ~900KB), modernized API contracts, and enhanced usability.

End-of-Life Notice for AWS SDK for .NET V3

The AWS SDK for .NET V3, a cornerstone for many .NET applications interacting with AWS, is officially moving into its end-of-life phase. This announcement marks a significant shift, signaling the need for all development teams to plan and execute a migration to the successor, AWS SDK for .NET V4.

This isn't merely a recommendation; it's a critical directive to ensure the continued security, reliability, and performance of your applications. Ignoring this transition will expose your systems to increasing risks and technical debt.

Support Timeline and Key Dates

Understanding the support lifecycle is crucial for planning your migration strategy. The transition for AWS SDK for .NET V3 follows a clear, two-phase timeline:

- **March 1, 2026: Maintenance Mode Begins**
  - AWS SDK for .NET V3 officially enters maintenance mode.
  - During this period (March 1, 2026 - June 1, 2026), AWS will limit releases for V3 to address **critical bug fixes and security patches only**. No new features or general bug fixes will be introduced.
  - Previously published releases will remain available via public package feeds, but no new updates beyond critical issues will be pushed.
- **June 1, 2026: End-of-Support (EOS)**
  - AWS SDK for .NET V3 reaches its official End-of-Support date.
  - **No further updates or releases will be provided for V3**, including security patches, bug fixes, or compatibility updates.
  - Applications still relying on V3 after this date will be exposed to potential security vulnerabilities and compatibility issues with future AWS service updates.

Implications for Developers

The end-of-support for AWS SDK for .NET V3 carries significant implications for any developer or organization currently using it.

Security Vulnerabilities

🧠 `Important:` After June 1, 2026, AWS SDK for .NET V3 will no longer receive security updates. This means any newly discovered vulnerabilities will remain unpatched, leaving your applications and the data they handle exposed to potential exploits. This is a critical risk that cannot be overstated, especially for production systems handling sensitive information.

Lack of Bug Fixes and Compatibility

Beyond security, V3 will cease to receive general bug fixes. As AWS services evolve, V3 might also encounter compatibility issues with new features or

changes in AWS APIs, leading to unexpected runtime errors or degraded functionality.

Stagnation and Technical Debt

Staying on V3 means foregoing the advancements and improvements in V4. Your codebase will accumulate technical debt, making future updates or migrations even more challenging and costly. Integrating with new AWS services or features might require custom workarounds or become impossible.

⚠️ `What can go wrong:` Continuing to use an unsupported SDK version is a common source of production outages and security breaches. It's a ticking time bomb for your application's stability and integrity.

Why Upgrade to AWS SDK for .NET V4? (Performance, API Improvements)

Migrating to AWS SDK for .NET V4 is not just about avoiding risks; it's about embracing a superior development experience and unlocking significant benefits for your applications.

Performance Enhancements

V4 introduces substantial performance improvements, making your applications faster and more efficient.

- **Reduced Assembly Size**: A key highlight is the reduction in the `AWSSDK.Core` assembly size. In V3, it was approximately 2MB. In V4, this has been significantly trimmed down to **around 900KB**. This smaller footprint contributes to faster startup times, reduced memory consumption, and more efficient deployment packages, especially beneficial for serverless functions (e.g., AWS Lambda).
- **General Optimizations**: Beyond size, V4 incorporates general performance enhancements and optimizations across the SDK, leading to better overall responsiveness and resource utilization.

Modernized and Simplified API Contracts

V4 brings a fresh perspective to the SDK's design, focusing on consistency and developer experience.

- **API Consistency**: V4 introduces modernized and simplified API contracts, enhancing consistency with other AWS SDKs. This means a more unified

developer experience across different languages and platforms, reducing the learning curve for developers working with multiple SDKs.

- **Improved Usability and Bug Fixes**: The new version incorporates improved usability patterns and addresses various bug fixes, leading to a more robust and intuitive development experience. This includes cleaner interfaces and more predictable behavior.

⚡ `Real-world insight:` For applications deployed on AWS Lambda, a smaller SDK footprint directly translates to faster cold starts and lower memory usage, leading to cost savings and improved user experience.

Key Breaking Changes and Migration Guidance

The migration from AWS SDK for .NET V3 to V4 is a major version upgrade, which inherently involves breaking changes. These changes are necessary to deliver the performance improvements and modernized API contracts.

Modernized API Contracts and Models

- **API Signatures**: Many API method signatures and object models have been updated to be more consistent and idiomatic for .NET, and to align better with other AWS SDKs. This will likely require modifications to your existing V3 code that interacts with AWS services.
- **Type Changes**: Expect changes in data types, enum values, and class structures. These changes aim to improve clarity and correctness but necessitate code adjustments.

📌 `Key Idea:` While the migration requires effort, it's an investment in a more stable, performant, and future-proof codebase.

Migration Resources

AWS provides comprehensive resources to guide you through this transition:

- **Official Migration Guide**: This is your primary resource for understanding all breaking changes and the necessary adjustments. It details specific API changes, recommended refactoring patterns, and best practices for a smooth migration.
    - AWS SDK for .NET V4 Migration Guide

🔥 `Optimization / Pro tip:` Start your migration by updating a non-critical component or a test project first. This allows you to identify common patterns of breaking changes in your codebase and refine your migration strategy before

tackling larger, more complex parts of your application. Automated refactoring tools or custom scripts can significantly aid in this process.

Strong Recommendation and Call to Action

Given the imminent end-of-support for AWS SDK for .NET V3 and the critical security implications of remaining on an unsupported version, we issue a **strong recommendation for all developers to initiate their migration to AWS SDK for .NET V4 immediately.**

This is not a task to defer. The maintenance window is short, and the end-of-support date will arrive quickly. Proactive migration ensures your applications remain secure, performant, and compatible with the evolving AWS ecosystem.

How to Upgrade

Upgrading your project to AWS SDK for .NET V4 primarily involves updating your NuGet package references.

1. **Backup Your Project**: Always start by creating a backup or committing your current changes to version control.
2. **Update NuGet Packages**:
    - Open your project in Visual Studio.
    - Right-click on your project or solution in the Solution Explorer and select "Manage NuGet Packages...".
    - Go to the "Updates" tab and look for `AWSSDK.*` packages.
    - Select the packages you wish to upgrade and choose the latest V4 version.
    - Alternatively, use the NuGet Package Manager Console: `powershell Update-Package AWSSDK.Core -Version 4.0.0.0 -ProjectName YourProjectName # Repeat for other AWSSDK.* packages you are using, e.g., AWSSDK.S3, AWSSDK.DynamoDBv2`
    - For .NET CLI users: `bash dotnet add package AWSSDK.Core --version 4.0.0 # Repeat for other AWSSDK.* packages`
3. **Address Breaking Changes**: After updating, your project will likely have compilation errors due to the breaking changes. Refer to the official migration guide to resolve these. This will involve updating API calls, object models, and potentially configuration.

4. **Thorough Testing**: After resolving all compilation errors, perform comprehensive testing of your application to ensure all AWS interactions function as expected.

🧠 Check Your Understanding

- What is the critical date after which AWS SDK for .NET V3 will no longer receive any updates, including security patches?
- Name two significant benefits of upgrading to AWS SDK for .NET V4, beyond just security.
- Why is it important to consult the official migration guide during your upgrade process?

⚡ Mini Task

Identify one `AWSSDK.*` package currently used in your project's `csproj` file or `packages.config`. Imagine you're upgrading it to V4. Write down the `dotnet add package` command you would use.

🚀 Scenario

Your team manages a critical e-commerce backend built on .NET using AWS SDK V3. The end-of-support date is approaching rapidly. Your manager is hesitant to allocate resources for the migration due to ongoing feature development. How would you articulate the urgency and risks to convince your manager to prioritize the V4 migration? Focus on the tangible impacts of not upgrading.

📌 TL;DR

- AWS SDK for .NET V3 enters maintenance mode March 1, 2026, with End-of-Support on June 1, 2026.
- Post-June 1, 2026, V3 will receive no security updates, posing critical risks.
- V4 offers significant performance gains (e.g., `AWSSDK.Core` from 2MB to ~900KB) and modernized APIs.

🧠 Core Flow

1. **Acknowledge EOL**: Recognize the AWS SDK for .NET V3 end-of-support announcement and its critical implications.
2. **Plan Migration**: Allocate resources and time for a phased migration to AWS SDK for .NET V4.
3. **Execute Upgrade**: Update NuGet packages to V4 and systematically address breaking changes using the official migration guide.
4. **Test and Deploy**: Thoroughly test the upgraded application and deploy the V4-compatible version to production.

🚀 Key Takeaway

Proactive migration to AWS SDK for .NET V4 is imperative for maintaining application security, stability, and leveraging modern performance improvements, transforming a compliance necessity into a strategic upgrade.