C# and .NET: which APIs and defaults turn into vulnerabilities?

Move every app off .NET 8 and 9 to .NET 10.0.14 or later now, as both lose support on 10 November 2026[1,4], and before your next release remove `BinaryFormatter` and set a fallback authorization policy. Judge everything else by whether outside data can pick a type, reach SQL, a command line, a path, a URL or raw HTML, or skip an authorization check.

Reviewed	Next review	Version
5 Nov 2026	**5 Dec 2026**	**1.1**

Written for

Developers and tech leads writing production C# on .NET and ASP.NET Core, including legacy .NET Framework code, and the security people who review it

Latest version

sheets.einblick.dev/secure-csharp/

If you only do five things

1. **Move to .NET 10 and patch every month.** .NET 8 and 9 lose support on 10 November 2026; .NET 10 is supported until 14 November 2028[1]. Run 10.0.14 or later[4] and patch within the patching sheet's deadlines.
2. **Deserialize only into types you name.** `BinaryFormatter` can't be made secure[9], and CISA listed four exploited SharePoint deserialization flaws in 2026[8]. Use System.Text.Json; keep Json.NET's `TypeNameHandling` at `None`[12].
3. **Require login by default, then check each record.** The default fallback policy is `null`, so endpoints without authorization metadata are public[15]. Set it to `RequireAuthenticatedUser()`, then check records with `IAuthorizationService.AuthorizeAsync`[14].
4. **Keep input out of SQL, command lines and paths.** Use LINQ or `FromSql`, not `FromSqlRaw`[16], and `ArgumentList` without a shell[18]. SQL injection, path traversal and command injection rank 2nd, 6th and 9th in the CWE Top 25[7].
5. **Turn on the security analyzers and package audit.** Only some rules run by default[23]: set `<AnalysisModeSecurity>All</AnalysisModeSecurity>`, and `<NuGetAuditMode>all</NuGetAuditMode>` below `net10.0`[26].

Review a C# pull request

1. **Where does outside data enter?** Find the controllers, minimal API handlers, Razor Pages, Blazor components, SignalR hubs and queue consumers it touches. Each should bind a request type, not an EF Core entity.
2. **Who may call it, on which record?** Ask why for every `[AllowAnonymous]`, `AllowAnonymous()`, `DisableAntiforgery()` and `[IgnoreAntiforgeryToken]`. Neither `[Authorize]` nor the fallback policy checks the record[14].
3. **Does input reach a type name, SQL, a process or a path?** Search for `TypeNameHandling`, `Type.GetType`, `FromSqlRaw`, `ExecuteSqlRaw`, `CommandText`, `Process.Start`, `Path.Combine` and `ExtractToFile`.
4. **Does it reach HTML or the network?** Check `Html.Raw`, `MarkupString`, `HttpClient` calls to URLs from input and every `ServerCertificateCustomValidationCallback`.
5. **Does it switch off the runtime's checks?** `unsafe` blocks, `Unsafe` and `MemoryMarshal` calls, `[DllImport]` and `[LibraryImport]` need a second reviewer and a test.

C# and .NET traps and what to use instead

- **`BinaryFormatter` and its relatives.** `SoapFormatter`, `NetDataContractSerializer`, `LosFormatter` and `ObjectStateFormatter` also let the payload pick the type[9]. .NET 9 and later throw on `BinaryFormatter`, but the `System.Runtime.Serialization.Formatters` package restores it, risks included. Read old payloads with the NRBF reader, which creates no types[10].
- **Json.NET `TypeNameHandling` other than `None`.** The JSON then names the type to create, and Newtonsoft says to vet it with a `SerializationBinder`[11]. Keep `None`, never give a serializer `Type.GetType(name)` with a name from data[12], and in System.Text.Json declare subtypes with `[JsonDerivedType]`.
- **Known `machineKey` values in ASP.NET Framework.** Microsoft found over 3,000 public keys usable for ViewState code injection[13], and CISA lists Sitecore's default keys as exploited[8]. Generate unique keys, rotate them and encrypt the `machineKey` section[13].
- **`FromSqlRaw($"... {id}")`.** `FromSqlRaw` and `ExecuteSqlRaw` paste interpolated values into the SQL; `FromSql` and `ExecuteSql` send each as a `DbParameter`[16], and EF Core 10 warns on concatenation[17]. Map column names from a fixed list[14].
- **`Process.Start("cmd.exe", "/c ...")`.** Start a fixed full path with `UseShellExecute = false`, not the .NET Framework default[19], use `ArgumentList` on .NET, and never pass input to `cmd.exe`, `powershell.exe` or `sh -c`[18].
- **`Path.Combine(root, name)`.** A rooted `name` discards `root`, and Microsoft says to use `Path.Join` for user input[20]. That still lets `..` climb out, so require `Path.GetFullPath()` of the result to start with the root plus a separator.
- **`HttpClient.GetAsync(url)` with a URL from input.** Server-side request forgery ranks 22nd in the CWE Top 25[7]. Allowlist hosts, check the resolved address in `SocketsHttpHandler.ConnectCallback` and set `AllowAutoRedirect = false`.
- **`Html.Raw`, `HtmlString` and `MarkupString`.** They skip Razor's encoding[14,21], and cross-site scripting ranks 1st in the CWE Top 25[7]. Sanitize user HTML with a maintained sanitizer.

Tools that catch it

- **.NET analyzers.** Set `<AnalysisModeSecurity>All</AnalysisModeSecurity>`[23], then fix CA2300 to CA2330 (deserializers), CA2100 and CA3001 (SQL), CA3006 (processes) and CA5359 (certificates) first[24].
- **BannedApiAnalyzers** flags every use of the APIs listed in `BannedSymbols.txt`[25].
- **NuGet Audit** runs on every restore: add NU1903 and NU1904 to `<WarningsAsErrors>`[26]. The supply-chain sheet covers the rest.

What you can skip

- **`SecureString`.** Microsoft advises against it for new code, and it isn't encrypted outside Windows[27].
- **A `SerializationBinder` to tame `BinaryFormatter`.** It may not stop code execution, and Microsoft doesn't commit to patching bypasses[9]. Migrate.
- **In-process sandboxes.** Code Access Security and partial trust aren't supported security boundaries[14]. Use a separate process or container.

What changed in 2026

- **.NET 8 and 9 reach end of support on 10 November**[1], and .NET Framework 4.6.2 on 12 January 2027[2].
- **October's two .NET 10 releases fixed 46 CVEs,** including critical Kestrel request smuggling (CVE-2026-88092)[3,5]. Two fixes were complete only in 10.0.14[4].
- **Data Protection let attackers forge auth cookies** in the NuGet package 10.0.0 to 10.0.6, mainly outside the shared framework on Linux and macOS (CVE-2026-40372). If you ran it exposed, rotate the key ring and tokens issued meanwhile[6].

Mistakes that cost the most

- **Patching the host but not the app.** Self-contained apps stay vulnerable until recompiled and redeployed[5], and October's Blazor WebAssembly fix needed updated package references[4]. Rebuild container images too.
- **Unprotected Data Protection keys.** Default storage encrypts them only on Windows, and naming your own store turns that off[28]. Add `ProtectKeysWithAzureKeyVault`.
- **Development settings in production.** Keep the `Development` environment and unencrypted User Secrets on developer machines[14].

What this looks like for your code

New project

- Start on .NET 10 with nullable checks, all security rules and warnings as errors.
- Set the fallback policy before the first endpoint.
- Reject unknown and duplicate JSON properties, as .NET 10's `JsonSerializerOptions.Strict` does, but keep your naming policy[22].

Existing codebase

- Leave .NET 8 and 9 now; move .NET Framework 4.6.2 to 4.8 or 4.8.1 before 12 January 2027[2].
- Set `<NuGetAuditMode>all</NuGetAuditMode>` in `Directory.Build.props`.

High-assurance or regulated

- Require CodeQL's `security-extended` suite; private code needs paid Code Security[29].
- Turn on `CheckForOverflowUnderflow` where code computes sizes or money.
- Keep a Key Vault-protected key ring per app and rehearse revoking it[6].

Version history

5 Nov 2026, version 1.1 Reviewed for company readers: CodeQL on private code needs a paid GitHub Code Security license.

Sources

1. .NET and .NET Core Support Policy. Microsoft, updated 25 Oct 2026. dotnet.microsoft.com
2. Microsoft .NET Framework lifecycle. Microsoft Learn, checked 5 Nov 2026. learn.microsoft.com
3. .NET 10.0.13 release notes. Microsoft, 13 Oct 2026. github.com
4. .NET October 2026 out-of-band security updates. Microsoft, 25 Oct 2026. github.com
5. Advisory CVE-2026-88092 (Kestrel). Microsoft, 13 Oct 2026. github.com
6. Advisory CVE-2026-40372 (Data Protection). Microsoft, 21 Apr 2026. github.com
7. 2025 CWE Top 25. MITRE CWE, Dec 2025. cwe.mitre.org
8. Known Exploited Vulnerabilities Catalog. CISA, checked 5 Nov 2026. cisa.gov
9. BinaryFormatter security guide. Microsoft Learn, updated 8 Aug 2024. learn.microsoft.com
10. BinaryFormatter migration guide. Microsoft Learn, updated 8 Aug 2024. learn.microsoft.com
11. TypeNameHandling Enumeration. Json.NET docs, checked 5 Nov 2026. newtonsoft.com
12. Deserialization Cheat Sheet. OWASP, checked 5 Nov 2026. cheatsheetseries.owasp.org
13. Code injection attacks using publicly disclosed ASP.NET machine keys. Microsoft Threat Intelligence, 6 Feb 2025. microsoft.com
14. DotNet Security Cheat Sheet. OWASP, checked 5 Nov 2026. cheatsheetseries.owasp.org
15. Policy-based authorization. Microsoft Learn, updated 20 Sep 2026. learn.microsoft.com
16. SQL Queries. EF Core docs, checked 5 Nov 2026. learn.microsoft.com
17. What's New in EF Core 10. EF Core docs, 2 Oct 2025. learn.microsoft.com
18. OS Command Injection Defense Cheat Sheet. OWASP, checked 5 Nov 2026. cheatsheetseries.owasp.org
19. ProcessStartInfo.UseShellExecute. Microsoft Learn, checked 5 Nov 2026. learn.microsoft.com
20. Path.Combine. Microsoft Learn, checked 5 Nov 2026. learn.microsoft.com
21. Razor components: raw HTML. Microsoft Learn, checked 5 Nov 2026. learn.microsoft.com
22. .NET 10 libraries: what's new. Microsoft Learn, 7 Nov 2025. learn.microsoft.com
23. MSBuild properties: code analysis. Microsoft Learn, checked 5 Nov 2026. learn.microsoft.com
24. Security rules (code analysis). Microsoft Learn, checked 5 Nov 2026. learn.microsoft.com
25. BannedApiAnalyzers help. .NET on GitHub, checked 5 Nov 2026. github.com
26. Auditing package dependencies. NuGet docs, updated 5 May 2026. learn.microsoft.com
27. SecureString Class. Microsoft Learn, checked 5 Nov 2026. learn.microsoft.com
28. Data Protection key management and lifetime. Microsoft Learn, checked 5 Nov 2026. learn.microsoft.com
29. CodeQL CLI. GitHub Docs, checked 5 Nov 2026. docs.github.com