



Reviewing NuGet Packages security easily using OpenSSF Scorecard

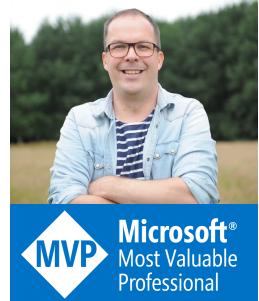
Niels Tanis
Sr. Principal Security Researcher



Who am I?

- Niels Tanis
- Sr. Principal Security Researcher
 - Background .NET Development, Pentesting/ethical hacking, and software security consultancy
 - Research on static analysis for .NET apps
 - Enjoying Rust!
- Microsoft MVP – Developer Technologies

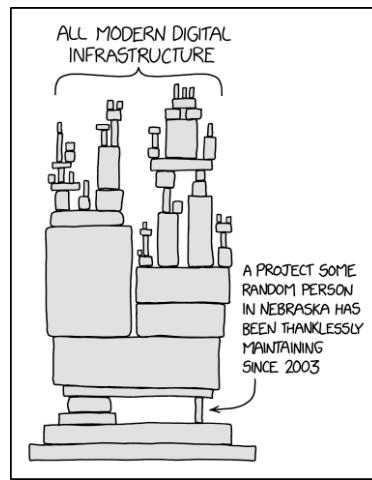
VERACODE



@niels.fennec.dev @nielstanis@infosec.exchange

Modern Application Architecture XKCD 2347

 elia group



nec.dev  @nielstanis@infosec.exchange

<https://xkcd.com/2347/>

Agenda

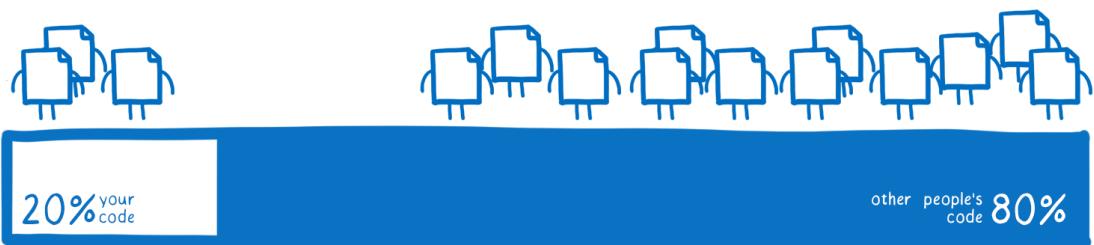
- Risks in 3rd party NuGet Packages
- OpenSFF Scorecard
- Measure, New & Improved
- Conclusion - Q&A



 elia group

 @niels.fennec.dev  @nielstanis@infosec.exchange

Average codebase composition



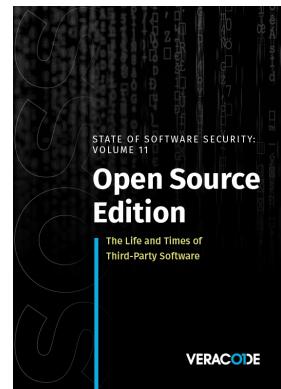
 elia group

 @niels.fennec.dev  @nielstanis@infosec.exchange

<https://hacks.mozilla.org/2019/11/announcing-the-bytecode-alliance/>

State of Software Security v11

"Despite this dynamic landscape, 79 percent of the time, developers never update third-party libraries after including them in a codebase."



@niels.fennec.dev @nielstanis@infosec.exchange

State of Log4j - 2 years later

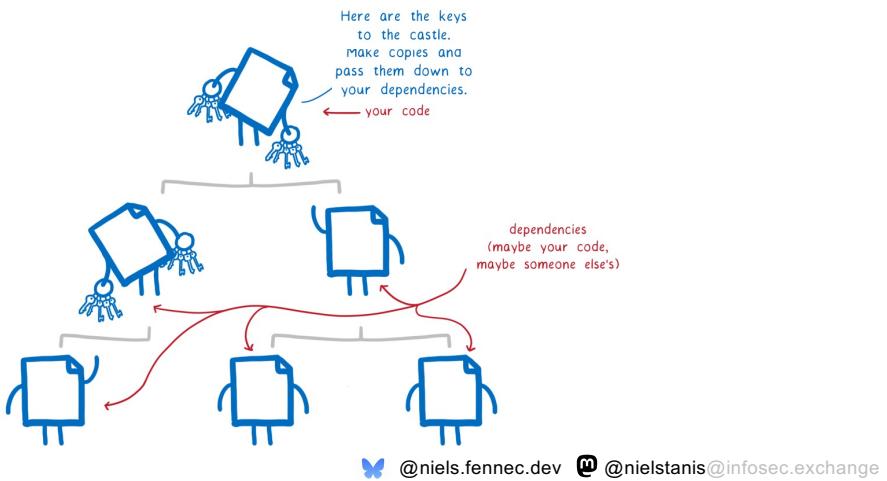
- Analysed our data August-November 2023
 - Total set of almost 39K unique applications scanned
- 2.8% run version vulnerable to Log4Shell
- 3.8% run version patched but vulnerable to other CVE
- 32% rely on a version that's end-of-life and have no support for any patches.



 @niels.fennec.dev  @nielstanis@infosec.exchange

<https://www.veracode.com/blog/research/state-log4j-vulnerabilities-how-much-did-log4shell-change>

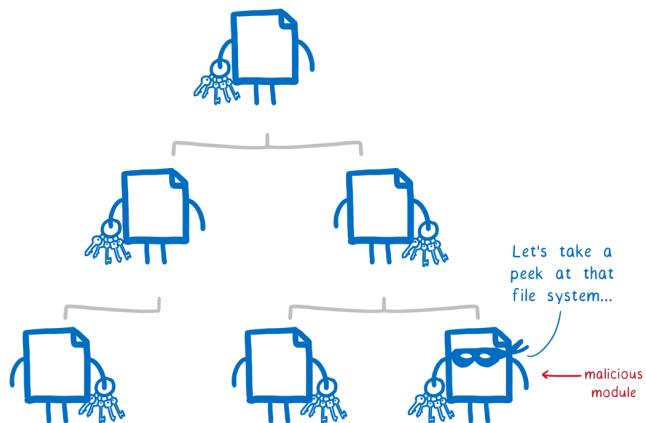
Average codebase composition



@niels.fennec.dev @nielstanis@infosec.exchange

<https://hacks.mozilla.org/2019/11/announcing-the-bytecode-alliance/>

Malicious Assembly



elia group

 @niels.fennec.dev  @nielstanis@infosec.exchange

<https://hacks.mozilla.org/2019/11/announcing-the-bytecode-alliance/>

The screenshot shows a news article titled "Hackers target .NET developers with malicious NuGet packages" by Sergiu Gatlan. The article discusses threat actors targeting .NET developers with cryptocurrency stealers delivered through the NuGet repository and impersonating multiple legitimate packages via typosquatting. It mentions three packages downloaded over 150,000 times. Researchers Natan Nehorai and Brian Moussalli spotted the campaign. The article notes that while the massive number of downloads could point to a large number of .NET developers, it could also be explained by attackers' efforts to legitimize their malicious NuGet packages. It quotes researchers saying the attack was highly successful. The threat actors used typosquatting for NuGet repository profiles. The article is dated March 20, 2023, at 03:22 PM.

elias group

@niels.fennec.dev @nielstanis@infosec.exchange

[https://www.bleepingcomputer.com/news/security/hackers-target-net-developers-with-malicious-nuget-packages//](https://www.bleepingcomputer.com/news/security/hackers-target-net-developers-with-malicious-nuget-packages/)

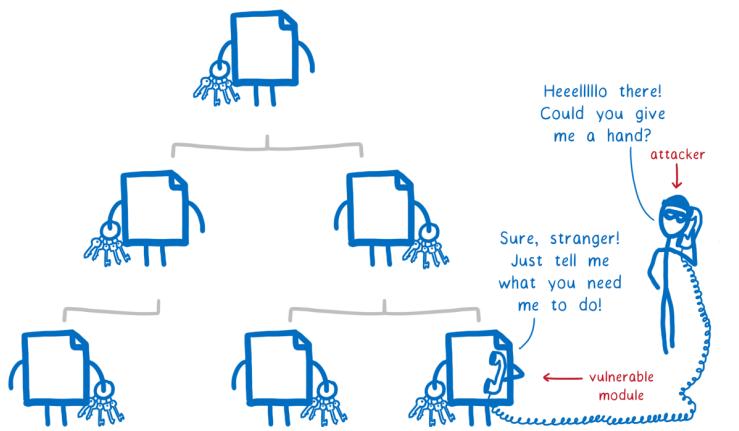
XZ Backdoor

The screenshot shows a web browser window displaying an Ars Technica article. The title of the article is "Backdoor found in widely used Linux utility targets encrypted SSH connections". The article is categorized under "SUPPLY CHAIN ATTACK". Below the title, it says "Malicious code planted in xz Utils has been circulating for more than a month." The author is listed as "DAN GOODIN - 3/29/2024, 7:50 PM". At the bottom of the page, there are social media sharing icons for elia group, Twitter (@niels.fennec.dev), and Mastodon (@nielstanis@infosec.exchange).

<https://arstechnica.com/security/2024/03/backdoor-found-in-widely-used-linux-utility-breaks-encrypted-ssh-connections/>

Vulnerable Assembly

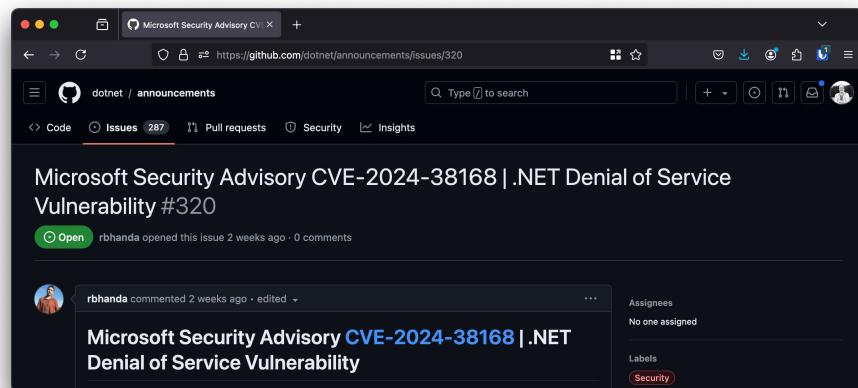
elia group



@niels.fennec.dev @nielstanis@infosec.exchange

<https://hacks.mozilla.org/2019/11/announcing-the-bytecode-alliance/>

Vulnerabilities in Libraries



🦋 @niels.fennec.dev ⚙️ @nielstanis@infosec.exchange

<https://github.com/dotnet/announcements/issues/320>

DotNet CLI

```
nelson@ghost-m2 ~/research/consoleapp $ dotnet list package
Project 'consoleapp' has the following package references
[net8.0]:
Top-Level Package      Requested    Resolved
> docgenerator          1.0.0        1.0.0

nelson@ghost-m2 ~/research/consoleapp $ dotnet list package --vulnerable

The following sources were used:
https://f.feedz.io/fennec/docgenerator/nuget/index.json
https://api.nuget.org/v3/index.json

The given project `consoleapp` has no vulnerable packages given the current sources.
nelson@ghost-m2 ~/research/consoleapp $
```



 @niels.fennec.dev  @nielstanis@infosec.exchange

DotNet CLI

```
nelson@ghost-m2:~/research/consoleapp $ dotnet list package --include-transitive
Project 'consoleapp' has the following package references
[net8.0]:
Top-level Package      Requested   Resolved
> docgenerator        1.0.0       1.0.0

Transitive Package                               Resolved
> itext7                                         7.2.2
> itext7.common                                     7.2.2
> Microsoft.CSharp                                4.0.1
> Microsoft.DotNet.PlatformAbstractions           1.1.0
> Microsoft.Extensions.DependencyInjection             5.0.0
> Microsoft.Extensions.DependencyInjection.Abstractions 5.0.0
> Microsoft.Extensions.DependencyModel            1.1.0
> Microsoft.Extensions.Logging                     5.0.0
> Microsoft.Extensions.Logging.Abstractions         5.0.0
> Microsoft.Extensions.Options                   5.0.0
> Microsoft.Extensions.Primitives                5.0.0
```



 @niels.fennec.dev  @nielstanis@infosec.exchange

DotNet CLI

```
nelson@ghost-m2 ~/research/consoleapp $ dotnet list package --vulnerable --include-transitive
The following sources were used:
https://f.feedz.io/fennec/docgenerator/nuget/index.json
https://api.nuget.org/v3/index.json

Project `consoleapp` has the following vulnerable packages
[net8.0]:
Transitive Package      Resolved    Severity    Advisory URL
> Newtonsoft.Json        9.0.1       High       https://github.com/advisories/GHSA-5crp-9r3c-p9vr

nelson@ghost-m2 ~/research/consoleapp $
```



 @niels.fennec.dev  @nielstanis@infosec.exchange

Do you know what's inside?

The screenshot shows a web browser window with the URL <https://www.reversinglabs.com/blog/third-party-code-comes-with-some-baggage>. The page is titled "ReversingLabs Blog" and features a post by "Threat Research | July 7, 2021". The main title of the post is "Third-party code comes with some baggage". Below the title, it says "Recognizing risks introduced by statically linked third-party libraries". A small bio for the author, Karlo Zanki, is present, noting he is a Reverse Engineer at ReversingLabs. The browser interface includes a navigation bar with back, forward, and search buttons, as well as a toolbar with various icons.

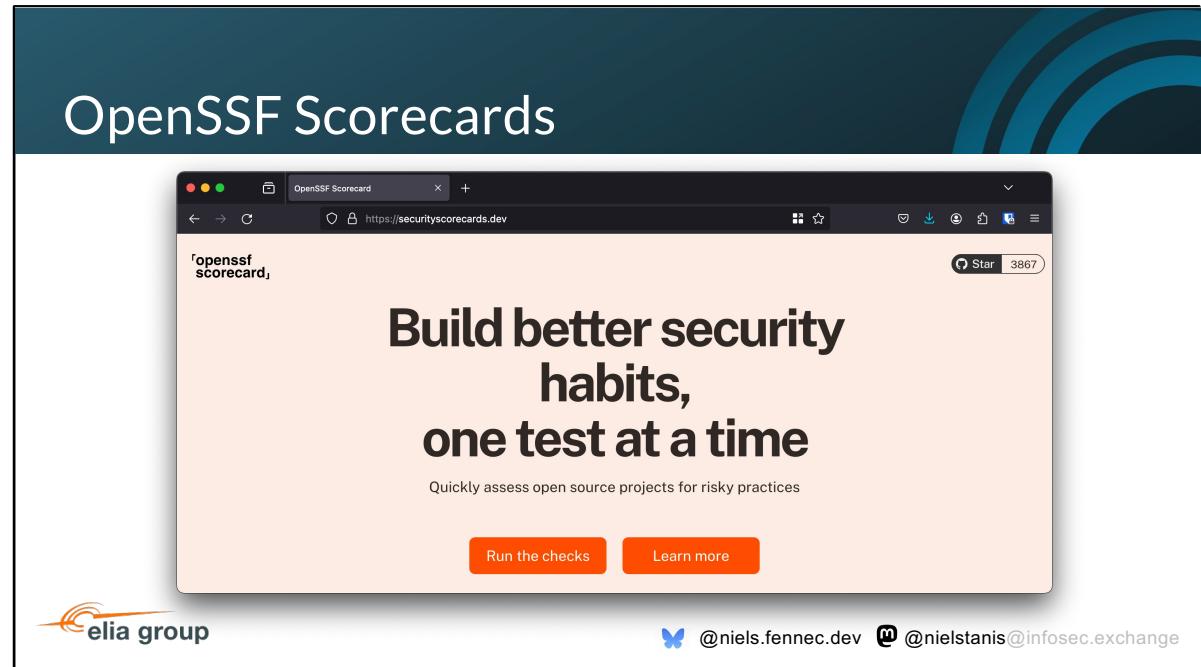
<https://www.reversinglabs.com/blog/third-party-code-comes-with-some-baggage>

Nutrition Label for Software?



@niels.fennec.dev @nielstanis@infosec.exchange

<https://securityscorecards.dev/>



<https://securityscorecards.dev/>

OpenSSF Security Scorecards

The screenshot shows a web browser window with the title 'OpenSSF Scorecard' at the top. The main content area is titled 'What is OpenSSF Scorecard?'. On the left, there's a sidebar with sections for 'Run the checks' (Using the GitHub Action, Using the CLI) and 'Learn more' (The problem, What is OpenSSF Scorecard?, How it works, The checks, Use cases, About the project name, Part of the OSS community, Get involved). The main content area describes the Scorecard as a tool for assessing security risks through automated checks, created by OSS developers to help improve critical projects. It also mentions its use for proactive assessment and improving codebases. At the bottom of the main content area are two icons: a red circle with a white dot and a grid of squares.

elia group

@niels.fennec.dev @nielstanis@infosec.exchange

<https://securityscorecards.dev/>

The screenshot shows a web browser window with the title "OpenSSF Security Scorecards". The main content area displays the "How it works" section. On the left, there's a sidebar with links for "Run the checks" (Using the GitHub Action, Using the CLI) and "Learn more" (The problem, What is OpenSSF Scorecard?, How it works, The checks, Use cases, About the project name, Part of the OSS community, Get involved). The main content area has three sections: "Scorecard checks for vulnerabilities affecting different parts of the software supply chain including source code, build, dependencies, testing, and project maintenance.", "Each automated check returns a score out of 10 and a risk level. The risk level adds a weighting to the score, and this weighting is compiled into a single, aggregate score. This score helps give a sense of the overall security posture of a project.", and "Alongside the scores, the tool provides remediation prompts to help you fix problems and strengthen your development practices." Below these sections is a horizontal bar with three segments: "CRITICAL RISK" (10), "HIGH RISK" (7.5), and "MEDIUM RISK" (5). At the bottom of the page, there are social media icons and handles: a blue butterfly icon for Twitter (@niels.fennec.dev) and a black person icon for LinkedIn (@nielstanis@infosec.exchange).

<https://securityscorecards.dev/>

OpenSSF Security Scorecards

The screenshot shows a web browser window for the OpenSSF Scorecard. The URL is <https://securityscorecards.dev/#the-checks>. The page features a large graphic in the center illustrating 'HOLISTIC SECURITY PRACTICES' as the core, surrounded by five interconnected circles: 'CODE VULNERABILITIES', 'BUILD RISK ASSESSMENT', 'MAINTENANCE', 'SOURCE RISK ASSESSMENT', and 'CONTINUOUS TESTING'. To the left of the graphic, there are sections for 'Run the checks' (GitHub Action, CLI) and 'Learn more' (problem, what it is, how it works, checks, use cases, project info, get involved). At the bottom left is the 'elia group' logo, and at the bottom right are social media links for @niels.fennec.dev and @nielstanis@infosec.exchange.

<https://securityscorecards.dev/>

Code Vulnerabilities (High)

- Does the project have unfixed vulnerabilities?
Uses the OSV service.

ID	Packages	Summary	Published	Attributes
GHSA-hrwv-x3fq-vcv7	NuGet/Umbraco.Cms	Umbraco CMS Improper Access Control vulnerability	20 Aug	Fix available Severity - 6.3 (Medium)
GHSA-77gj-chp-39yx	NuGet/Umbraco.Cms.Api.Management	Umbraco CMS vulnerable to Generation of Error Message Containing Sensitive Information	20 Aug	Fix available Severity - 5.3 (Medium)
GHSA-7qrv-8f9x-3h32	NuGet/Microsoft.AspNetCore.App.Runtime.win-arm NuGet/Microsoft.AspNetCore.App.Runtime.win-arm64 NuGet/Microsoft.AspNetCore.App.Runtime.win-x64 NuGet/Microsoft.AspNetCore.App.Runtime.win-x86	Microsoft Security Advisory CVE-2024-38168 .NET Denial of Service Vulnerability	13 Aug	Fix available Severity - 8.7 (High)



@niels.fennec.dev @nielstanis@infosec.exchange

<https://osv.dev/list?ecosystem=NuGet>

Maintenance Dependency-Update-Tool (**High**)

- Does the project use a dependency update tool?
For example Dependabot or Renovate bot?
- Out-of-date dependencies make a project vulnerable
to known flaws and prone to attacks.



 @niels.fennec.dev  @nielstanis@infosec.exchange

Maintenance Security Policy (Medium)

- Does project have published security policy?
- E.g. a file named **SECURITY.md** (case-insensitive) in a few well-known directories.
- A security policy can give users information about what constitutes a vulnerability and how to report one securely so that information about a bug is not publicly visible.



 @niels.fennec.dev  @nielstanis@infosec.exchange

Maintenance License (Low)

- Does project have license published?
- A license can give users information about how the source code may or may not be used.
- The lack of a license will impede any kind of security review or audit and creates a legal risk for potential users.



 @niels.fennec.dev  @nielstanis@infosec.exchange

Maintenance CII Best Practices (**Low**)

- OpenSSF Best Practices Badge Program
- Way for Open Source Software projects to show that they follow best practices.
- Projects can voluntarily self-certify, at no cost, by using this web application to explain how they follow each best practice.



openssf best practices passing



🔗 @niels.fennec.dev 📩 @nielstanis@infosec.exchange

<https://www.bestpractices.dev/en/criteria/0>

Continuous testing CI Tests (Low)

- Does the project run tests before pull requests are merged?
- The check works by looking for a set of CI-system names in GitHub CheckRuns and Statuses among the recent commits (~30).



 @niels.fennec.dev  @nielstanis@infosec.exchange

Continuous testing Fuzzing (Medium)

- This check tries to determine if the project uses fuzzing by checking:
 - Added to [OSS-Fuzz](#) project.
 - If [ClusterFuzzLite](#) is deployed in the repository;
- Does it make sense to do fuzzing on .NET projects?



 @niels.fennec.dev  @nielstanis@infosec.exchange

Continuous testing Static Code Analysis (Medium)

- This check tries to determine if the project uses Static Application Security Testing (SAST), also known as static code analysis. It is currently limited to repositories hosted on GitHub.
 - CodeQL
 - SonarCloud
- Definitely room for improvement!



 @niels.fennec.dev  @nielstanis@infosec.exchange

Source Risk Assessment Binary Artifacts (**High**)

- This check determines whether the project has generated executable (binary) artifacts in the source repository.
- Binary artifacts cannot be reviewed, allowing possible obsolete or maliciously subverted executables.
- There is need for **reproducible builds!**



 @niels.fennec.dev  @nielstanis@infosec.exchange

Source Risk Assesement Branch Protection (**High**)

- This check determines whether a project's default and release branches are protected with GitHub's branch protection or repository rules settings.
 - Requiring code review
 - Prevent force push, in case of public branch all is lost!



@niels.fennec.dev



@nielstanis@infosec.exchange

Source Risk Assesement Dangerous Workflow (**Critical**)

- This check determines whether the project's GitHub Action workflows has dangerous code patterns.
 - Untrusted Code Checkout with certain triggers
 - Script Injection with Untrusted Context Variables
- <https://securitylab.github.com/research/github-actions-preventing-pwn-requests/>



 @niels.fennec.dev  @nielstanis@infosec.exchange

Source Risk Assesement Code Review (**Low**)

- This check determines whether the project requires human code review before pull requests are merged.
- The check determines whether the most recent changes (over the last ~30 commits) have an approval on GitHub and merger!=committer (implicit review)



@niels.fennec.dev



@nielstanis@infosec.exchange

Source Risk Assessment Contributors (Low)

- This check tries to determine if the project has recent contributors from multiple organizations (e.g., companies).
- Relying on single contributor is a risk for sure!
- But is a large list of contributors good?



 @niels.fennec.dev  @nielstanis@infosec.exchange

Build Risk Assessment Pinned Dependencies (**High**)

- Does the project pin dependencies used during its build and release process.
- **RestorePackagesWithLockFile** in MSBuild results in **packages.lock.json** file containing versioned dependency tree with hashes
- If Workflow is present what about the Actions used?



 @niels.fennec.dev  @nielstanis@infosec.exchange

Build Risk Assessment Token Permission (**High**)

- This check determines whether the project's automated workflows tokens follow the principle of least privilege.
- This is important because attackers may use a compromised token with write access to, for example, push malicious code into the project.



 @niels.fennec.dev  @nielstanis@infosec.exchange

<https://securitylab.github.com/research/github-actions-preventing-pwn-requests/>

Build Risk Assessment Packaging (Medium)

- This check tries to determine if the project is published as a package.
- Packages give users of a project an easy way to download, install, update, and uninstall the software by a package manager.



 @niels.fennec.dev  @nielstanis@infosec.exchange

Build Risk Assessment Signed Releases (**High**)

- This check tries to determine if the project cryptographically signs release artifacts.
 - Signed release packages
 - Signed build provenance



 @niels.fennec.dev  @nielstanis@infosec.exchange

Demo OpenSSF Scorecard Fennec CLI

Running checks



[@niels.fennec.dev](https://twitter.com/nfennec) [@nielstanis@infosec.exchange](mailto:nfennec@fennec.dev)

Measure?



❖ @niels.fennec.dev Ⓜ @nielstanis@infosec.exchange

<https://www.bestpractices.dev/en/criteria/0>

OpenSSF Annual Report 2023

OpenSSF Scorecard project
has **3,776 stars** on GitHub,
and runs a **weekly automated**
assessment scan against
software security criteria
of over **1M OSS projects**



 elia group

 @niels.fennec.dev  @nielstanis@infosec.exchange

<https://openssf.org/download-the-2023-openssf-annual-report/>

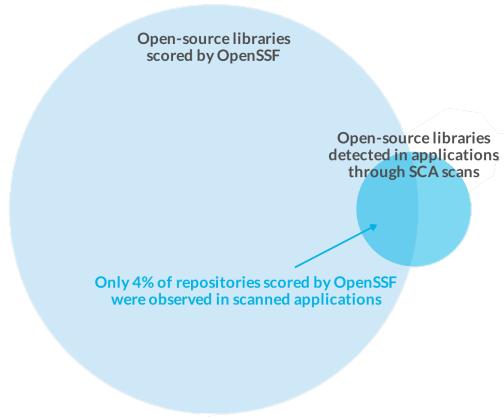
SOSS & OpenSSF Scorecard



 elia group

 @niels.fennec.dev  @nielstanis@infosec.exchange

SOSS & OpenSSF Scorecard

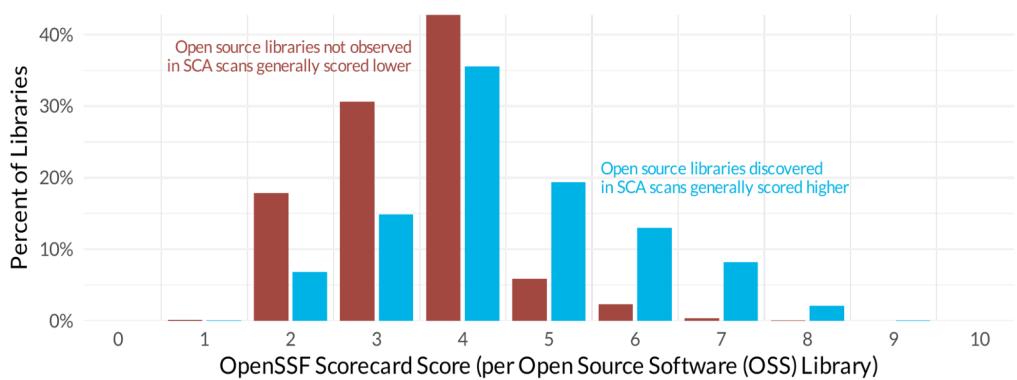


elia group

✉ @niels.fennec.dev Ⓜ @nielstanis@infosec.exchange

<https://www.rsaconference.com/Library/presentation/usa/2024/quantifying%20the%20probability%20of%20flaws%20in%20open%20source>

Correlation between SOSS

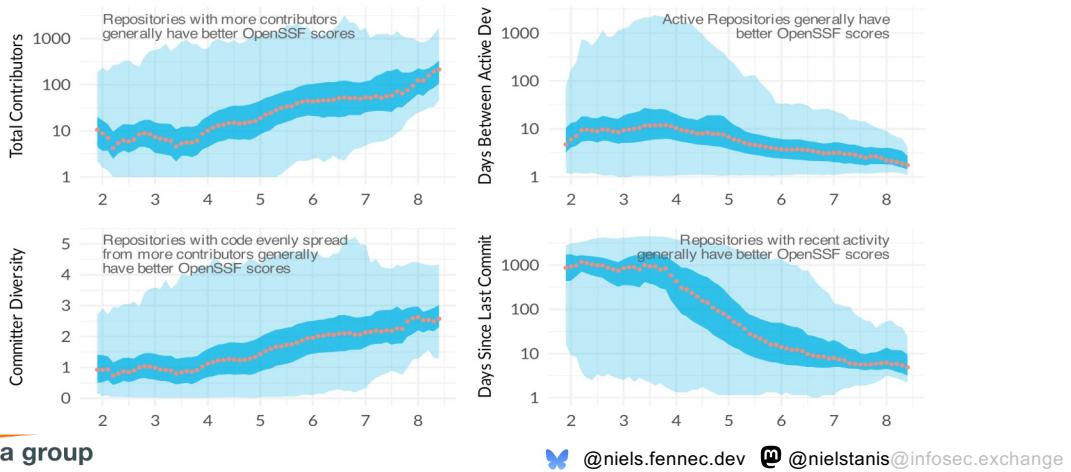


@niels.fennec.dev

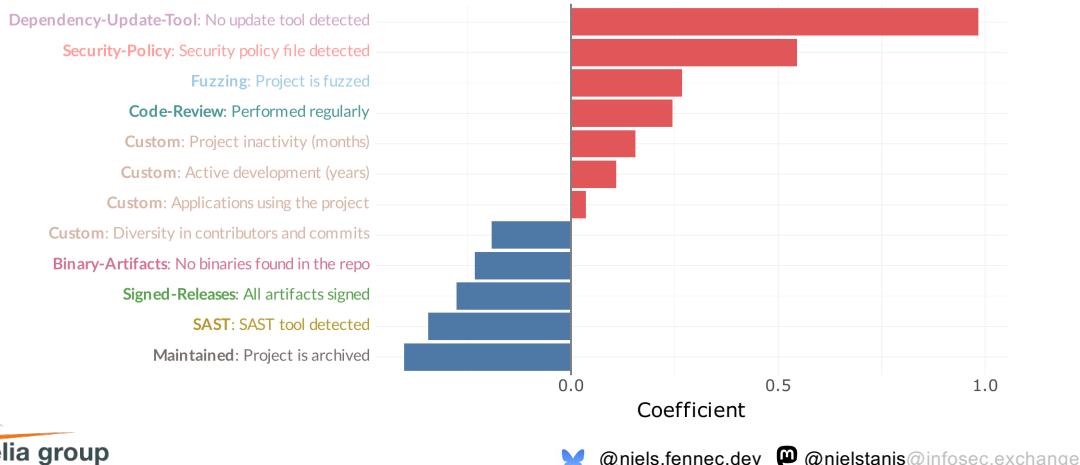


@nielstanis@infosec.exchange

Github commits vs OpenSSF



What really contributes to OSS Security?



What can we improve?



 @niels.fennec.dev  @nielstanis@infosec.exchange

<https://www.bestpractices.dev/en/criteria/0>

Fuzzing .NET



- Fuzzing, or fuzz testing
 - Automated software testing method that uses a wide range of **invalid** and unexpected data as input to find flaws
- Definitely good for finding C/C++ memory issues
- Can it be of any value with managed languages like .NET?



 @niels.fennec.dev  @nielstanis@infosec.exchange

<https://www.bestpractices.dev/en/criteria/0>

Fuzzing .NET & SharpFuzz

New & Improved!

Nemanja Mijailovic's Blog

Five years of fuzzing .NET with SharpFuzz

Jul 23, 2023

It's been almost five years since I created [SharpFuzz](#), the only .NET coverage-guided fuzzer. I already have a blog post on how it works, what it can do for you, and what bugs it found, so check it out if this is the first time you hear about SharpFuzz:

[SharpFuzz: Bringing the power of afl-fuzz to .NET platform](#)

A lot of interesting things have happened since then. SharpFuzz now works with libFuzzer, Windows, and .NET Framework. And it can finally fuzz the .NET Core base-class library! The whole fuzzing process has been dramatically simplified, too.

Not many people are aware of all these developments, so I decided to write this anniversary blog post and showcase everything SharpFuzz is currently capable of.

elia group

<https://mijailovic.net/2023/07/23/sharpfuzz-anniversary/>

[@niels.fennec.dev](#) [@nielstanis@infosec.exchange](#)

Fuzzing .NET – Jil JSON Serializer



```
public static void Main(string[] args)
{
    SharpFuzz.Fuzzer.OutOfProcess.Run(stream => {
        try
        {
            using (var reader = new System.IO.StreamReader(stream))
                JSON.DeserializeDynamic(reader);
        }
        catch (DeserializationException) { }
    });
}
```



 @niels.fennec.dev  @nielstanis@infosec.exchange

<https://github.com/google/fuzzing/blob/master/docs/structure-aware-fuzzing.md>

Static Code Analysis (SAST)

New &
Improved!

```
public byte[] CreateHash(string password)
{
    var b = Encoding.UTF8.GetBytes(password);
    return SHA1.HashData(b);
}
```



 @niels.fennec.dev  @nielstanis@infosec.exchange

<https://www.bestpractices.dev/en/criteria/0>

Static Code Analysis (SAST)

New &
Improved!

```
public class CustomerController : Controller
{
    public IActionResult GenerateCustomerReport(string customerID)
    {
        var data = Reporting.GenerateCustomerReportOverview(customerID)
        return View(data);
    }
    public static class Reporting
    {
        public static byte[] GenerateCustomerReportOverview(string ID)
        {
            return System.IO.File.ReadAllBytes("./data/{ID}.pdf");
        }
    }
}
```



 @niels.fennec.dev  @nielstanis@infosec.exchange

<https://www.bestpractices.dev/en/criteria/0>

.NET Reproducibility



- Reproducible builds → independently-verifiable path from source to binary code.
- .NET Roslyn Deterministic Inputs
- How reproducible is a simple console app?
- Fennec Diff (work-in-progress)



 @niels.fennec.dev  @nielstanis@infosec.exchange

Application Inspector

New & Improved!

Application Features

This section reports the major characteristics of the application or its primary features organized by customizable Feature Groups. Click any of the highlighted icons below (indicating at least one match) to view additional details or expand a feature group for more information. To view where in source code a specific feature was found, click the Rule name link shown on the right. A disabled icon indicates a not found status for that feature.

Feature Groups	Associated Rules
+ Select Features	Name (click to view source)
+ General Features	Authentication: Microsoft (Identity)
+ Development	Authentication: General
+ Active Content	Authentication: (OAuth)
+ Data Storage	
+ Sensitive Data	
+ Cloud Services	
+ OS Integration	
+ OS System Changes	
+ Other	

elia group

@niels.fennec.dev **@nielstanis@infosec.exchange**

<https://github.com/microsoft/ApplicationInspector>

Application Inspector



Feature	Confidence	Details
Authentication		View
Authorization		View
Cryptography		View
Object Deserialization		N/A
AV Media Parsing		N/A
Dynamic Command Execution		N/A

 [@niels.fennec.dev](#)  [@nielstanis@infosec.exchange](#)

<https://github.com/microsoft/ApplicationInspector>

The screenshot shows a blog post on the Microsoft Dev Blogs website. The title of the post is "OpenSSF Scorecard for .NET and the NuGet ecosystem". The post is dated November 4th, 2024. It features a brief introduction about the OpenSSF Scorecard tool. Below the introduction, there is a sidebar with social sharing icons for LinkedIn, GitHub, and Twitter, and a "Feedback" button. The footer of the page includes the "elia group" logo and the authors' social media handles: @niels.fennec.dev and @nielstanis@infosec.exchange.

<https://devblogs.microsoft.com/nuget/openssf-scorecard-for-net-nuget/>

Conclusion

- Scorecard helps security reviewing a NuGet Package
- Better understand what's inside, how it's build/maintained and what are the risks!
- Scorecard should not be a goal on its own!



@niels.fennec.dev



@nielstanis@infosec.exchange

Conclusion

- Room for .NET specific improvements with Fennec CLI
 - Tools (diff, insights)
 - Transitive Trust Graph
 - Contribute back to OpenSSF Scorecard



```
dotnet tool install -g fennec --prerelease
```



 @niels.fennec.dev  @nielstanis@infosec.exchange

- d

Danke, Merci, Bedankt!

- <https://github.com/niestanis/eliadevcon2024/>
- ntanis at Veracode.com
- @niestanis@infosec.exchange
- <https://www.fennec.dev>
<https://blog.fennec.dev>



 @niels.fennec.dev  @niestanis@infosec.exchange