Sandboxing .NET Assemblies for fun, profit and, of course Security!

Niels Tanis

Who am I?

- Niels Tanis
- Sr. Principal Security Researcher @ Veracode
  - Background .NET Development, Pentesting/ethical hacking, and software security consultancy
  - Research on static analysis for .NET apps
  - Having loads of fun with Rust!
  - Microsoft MVP Developer Technologies

@nielstanis@infosec.exchange

Agenda

- Introduction
- The security risks of third party libraries
- Sandboxing techniques
- Let’s create a sandbox!
- Conclusion
- QA

{Saturday} dot ned

@nielstanis@infosec.exchange

Third Party Libraries

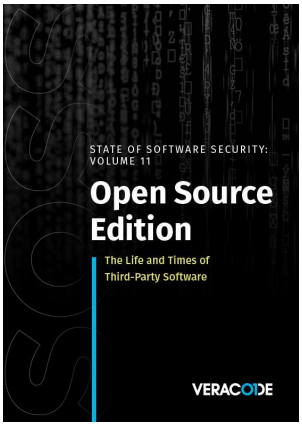
- Big chunk (80%+) of our apps consists of 3rd party libraries
- Efficient in time, why reinvent the wheel?
- How actively is it maintained?
- What do they do for security?

{Saturday} dot ned

@nielstanis@infosec.exchange

State Of Software Security v11 2021

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

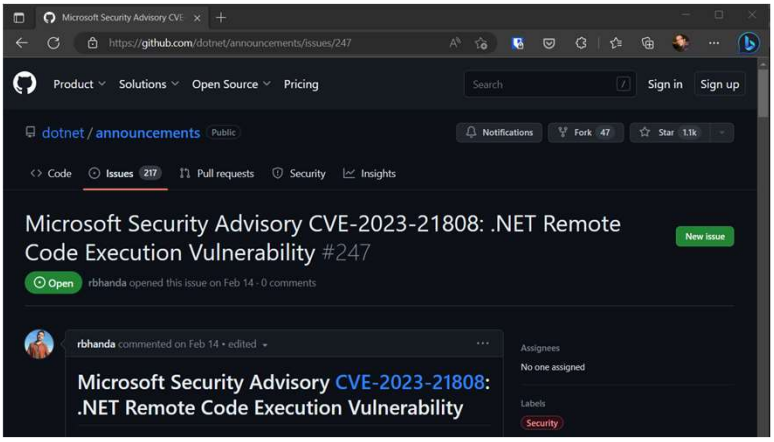


{Saturday} dot ned

@nielstanis@infosec.exchange

https://info.veracode.com/fy22-state-of-software-security-v11-open-source-edition.html

Vulnerabilities in libraries



{Saturday} dotned

@nielstanis@infosec.exchange

https://github.com/dotnet/announcements/issues/247

Vulnerabilities in libraries

CYBERSECURITY & INFRASTRUCTURE SECURITY AGENCY

Alerts and Tips Resources Industrial Control Systems

National Cyber Awareness System > Current Activity > Malware Discovered in Popular NPM Package, ua-parser-js

Malware Discovered in Popular NPM Package, ua-parser-js

Original release date: October 22, 2021

Print Tweet Send Share

Versions of a popular NPM package named `ua-parser-js` was found to contain malicious code. `ua-parser-js` is used in apps and websites to discover the type of device or browser a person is using from User-Agent data. A computer or device with the affected software installed or running could allow a remote attacker to obtain sensitive information or take control of the system.

CISA urges users and administers using compromised ua-parser-js versions 0.7.29, 0.8.0, and 1.0.0 to update to the respective patched versions: 0.7.30, 0.8.1, 1.0.1

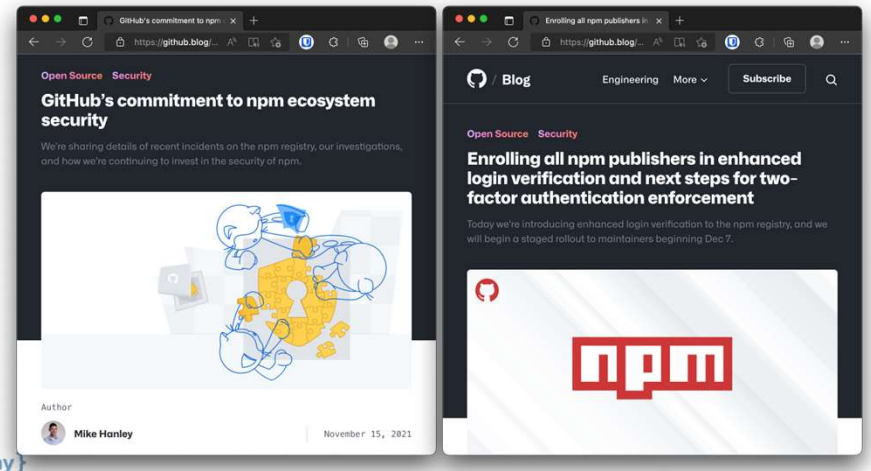
For more information, see Embedded malware in ua-parser-js.

{Saturday} dot ned

@nielstanis@infosec.exchange

https://us-cert.cisa.gov/ncas/current-activity/2021/10/22/malware-discovered-popular-npm-package-ua-parser-js
https://portswigger.net/daily-swig/popular-npm-package-ua-parser-js-poisoned-with-cryptomining-password-stealing-malware

Vulnerabilities in libraries



{Saturday} dot ned

@nielstanis@infosec.exchange

https://github.blog/2021-11-15-githubs-commitment-to-npm-ecosystem-security/
https://github.blog/2021-12-07-enrolling-npm-publishers-enhanced-login-verification-two-factor-authentication-enforcement/

Vulnerabilities in libraries

Threat Research | July 7, 2021

Third-party code comes with some baggage

Recognizing risks introduced by statically linked third-party libraries

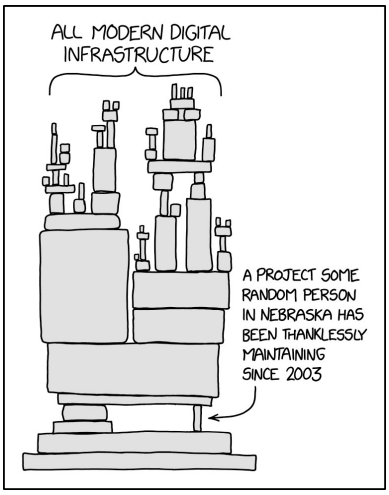
BLOG AUTHOR
Karlo Zanki, Reverse Engineer at ReversingLabs. READ MORE...

{Saturday} dot ned

@nielstanis@infosec.exchange

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

XKDC - Dependency



https://xkcd.com/2347/

@nielstanis@infosec.exchange

Sandboxing .NET Assemblies

- Is there a way we can do a better job?
- A way for us to reduce the security risks?
- Keep in mind it's not a matter of how it's more when!

{Saturday} dot ned

@nielstanis@infosec.exchange

Sandboxing .NET Assemblies

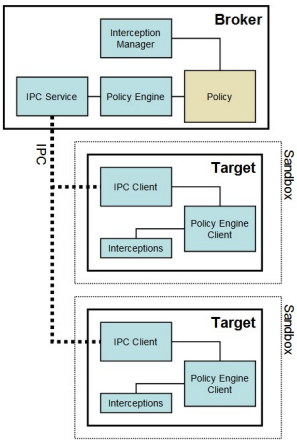
- We want to use the library without modification
- Can we maybe create a controlled (restricted) sandbox?
- A sandbox with limited capabilities?

{Saturday} dot ned

@nielstanis@infosec.exchange

Browser Sandbox

- Chromium Sandbox
- No direct system access
- Each OS related call is done via IPC
- FireFox Sandbox
  - Containers & Site Isolation
  - RLBox



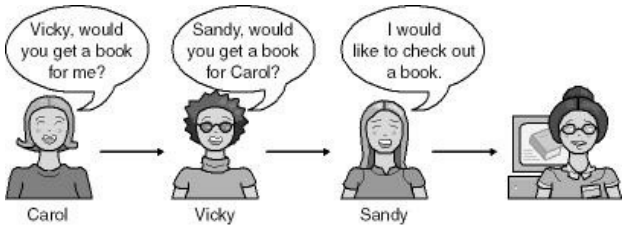
{Saturday} dot ned

@nielstanis@infosec.exchange

https://chromium.googlesource.com/chromium/src/+/refs/heads/main/docs/design/sandbox.md
https://hacks.mozilla.org/2021/05/introducing-firefox-new-site-isolation-security-architecture/

Code Access Security

- Evidence based model
- Code from different origins have different sets of rights
- Stack-walks that protect against luring attacks



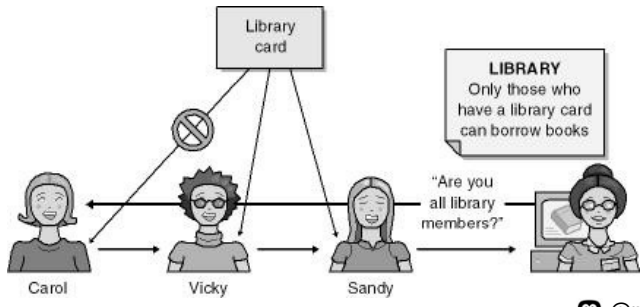
{Saturday} ned

@nielstanis@infosec.exchange

Figure 18-1; Writing Secure Code 2nd Edition

Code Access Security

- Evidence library card
- Policy → Librarian only allows members



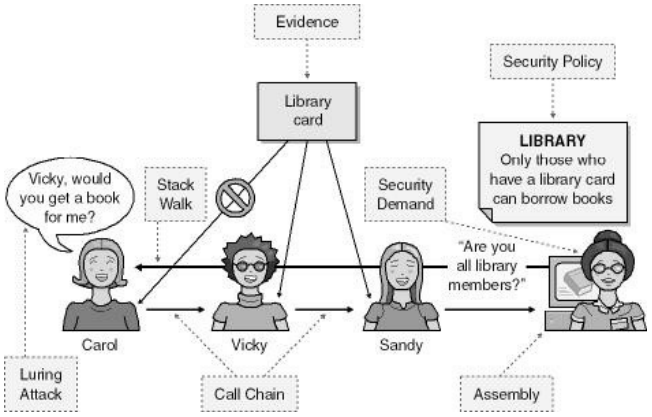
{Saturday} dot ned

@nielstanis@infosec.exchange

Figure 18-2; Writing Secure Code 2nd Edition

Code Access Security

- Stack walk



{Saturday} dot ned

@nielstanis@infosec.exchange

Figure 18-2; Writing Secure Code 2nd Edition

Code Access Security

- Most practical example, ASP.NET Medium Trust
- CAS is deprecated since .NET Framework 4
- Flipping a mutex in user memory to disable
- Too complex in administering and use?
- Too early?

{Saturday} dot ned

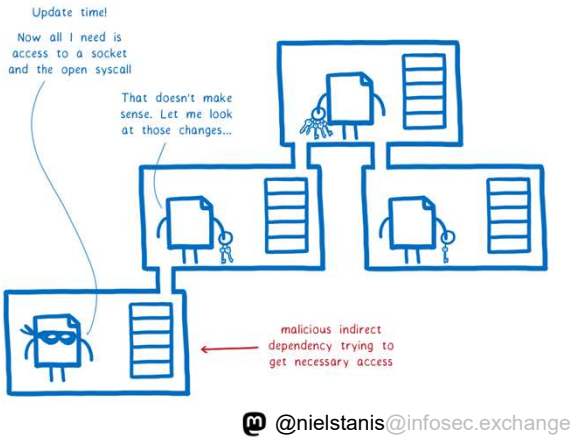
@nielstanis@infosec.exchange

https://docs.microsoft.com/en-us/previous-versions/dotnet/framework/code-access-security/code-access-security-basics
https://docs.microsoft.com/en-us/previous-versions/dotnet/framework/code-access-security/code-access-security-policy-compatibility-and-migration

WebAssembly Nanoprocess

- Lineair memory model
- Control-Flow integrity
- WASM module isolation

- Declaritive permissions
- Interface types
- WASI for BCL calls



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

Demo time!

{Saturday} dot ned

@nielstanis@infosec.exchange

Image: C# logo https://docs.microsoft.com/en-us/dotnet/csharp/

DocumentProcessor Package

- Use package as is!
  - Disclaimer: always comply with library license!
  - Not allowed to reverse engineer/decompile
- We do want to change behaviour:
  - Opening documents directly from URL – SSRF
  - Writing files to any arbitrary directory – Path Traversal
- There are *several* ways to *fix* this!

{Saturday} ned

@nielstanis@infosec.exchange

AssemblyLoadContext

- Only single AppDomain in .NET Core.
- AssemblyLoadContext replaces the isolation mechanisms provided by multiple AppDomain instances in .NET Framework.
- Conceptually, a load context creates a scope for loading, resolving, and potentially unloading a set of assemblies.

{Saturday} ned

@nielstanis@infosec.exchange

https://docs.microsoft.com/en-us/dotnet/api/system.runtime.loader.assemblyloadcontext?view=net-5.0
https://docs.microsoft.com/en-us/dotnet/core/dependency-loading/understanding-assemblyloadcontext

AssemblyLoadContext

- It allows multiple versions of the same assembly to be loaded within a single process.
- It does not provide any security features. All code has full permissions of the process.
- But it does allow us to control what gets loaded!

{Saturday} dot ned

@nielstanis@infosec.exchange

https://docs.microsoft.com/en-us/dotnet/api/system.runtime.loader.assemblyloadcontext?view=net-5.0
https://docs.microsoft.com/en-us/dotnet/core/dependency-loading/understanding-assemblyloadcontext

AssemblyLoadContext

- Interface project used as shared contract
- Remove DocumentProcessor package from ConsoleApp
  - Add reference to interface project
- Create Library that implements interface
  - Reference interface project and DocumentProcessor Package
  - Self-contained deployment to folder that has all to be loaded by our sandboxed loadcontext

{Saturday} ned

@nielstanis@infosec.exchange

https://docs.microsoft.com/en-us/dotnet/api/system.runtime.loader.assemblyloadcontext?view=net-5.0
https://docs.microsoft.com/en-us/dotnet/core/dependency-loading/understanding-assemblyloadcontext

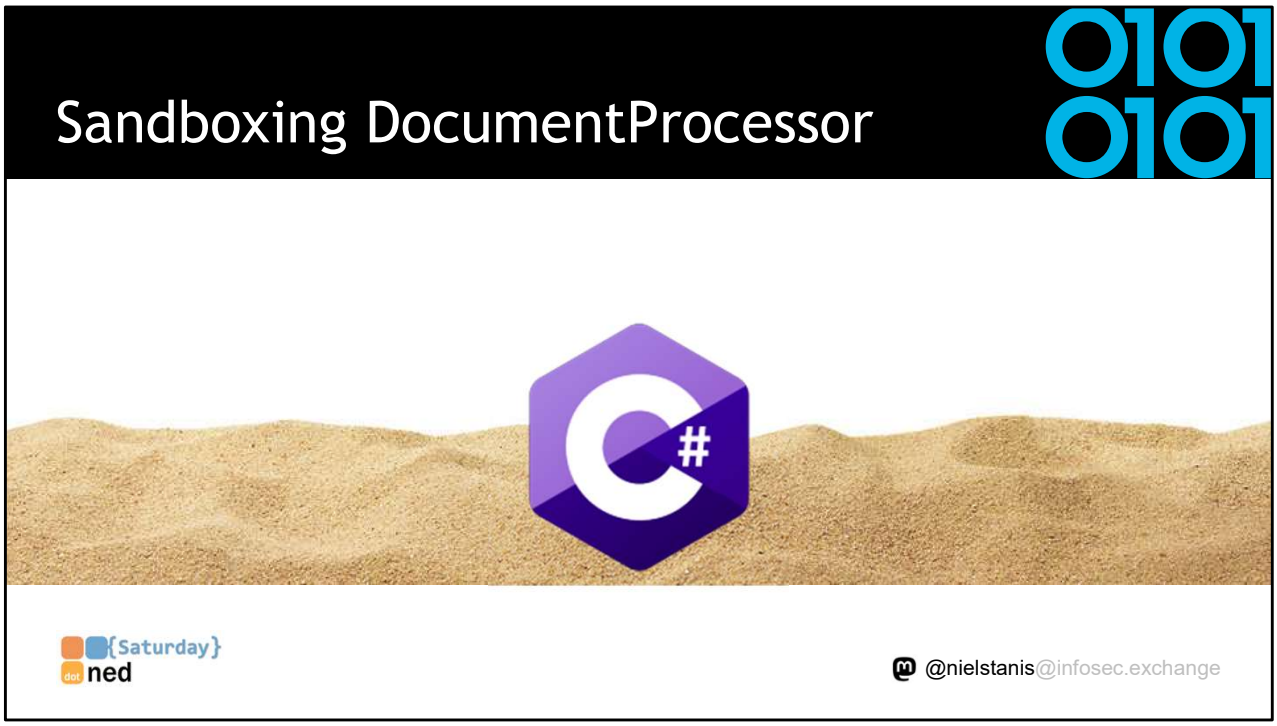
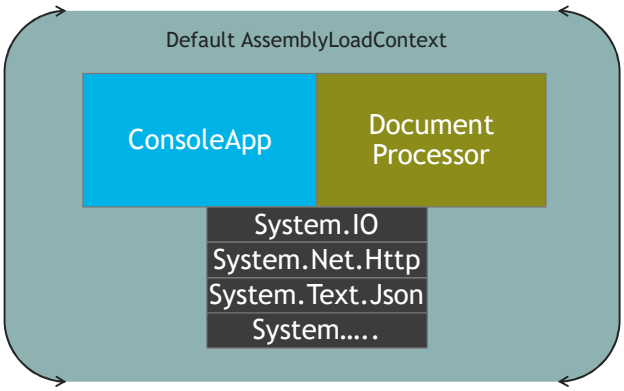


Image: C# logo https://docs.microsoft.com/en-us/dotnet/csharp/

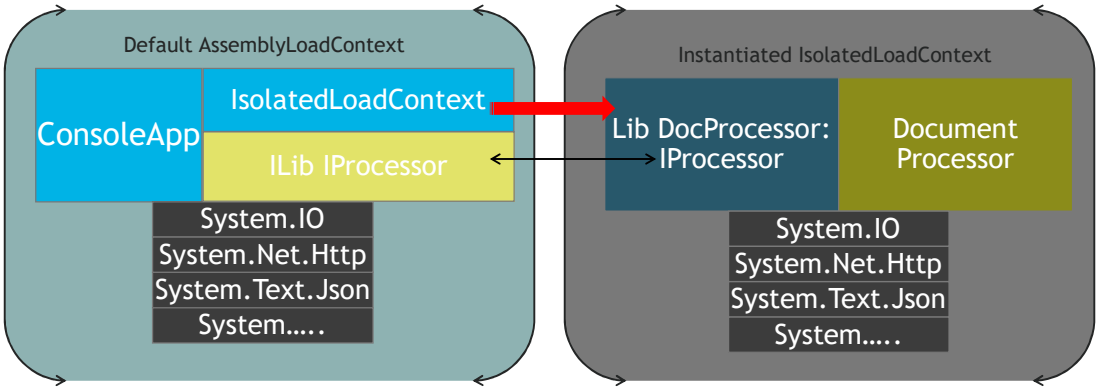
ConsoleApp Start



{Saturday} dot ned

@nielstanis@infosec.exchange

ConsoleApp & Sandboxed Library



{Saturday} dotned

@nielstanis@infosec.exchange

Removing Types?

- Self contained set of assemblies, could we maybe remove certain types?
- What about trimming that got introduced with .NET 5?
- Maybe we need something more rigorous?

{Saturday} dot ned

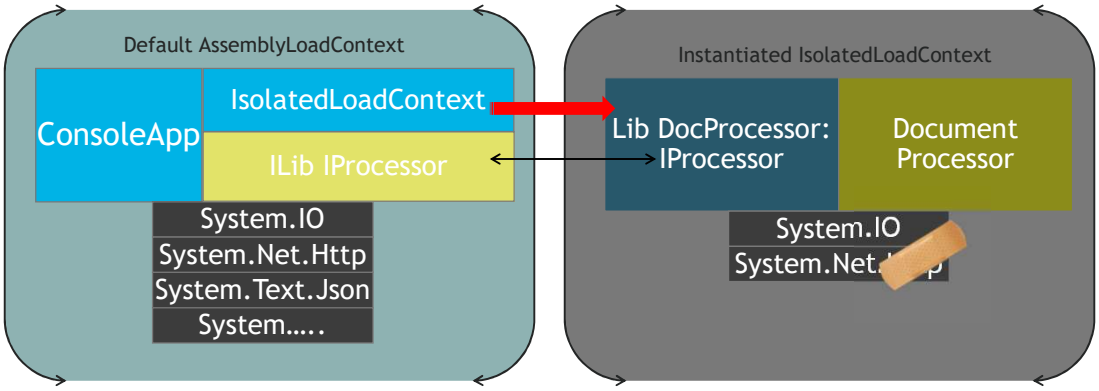
@nielstanis@infosec.exchange

Patching with Harmony2

- A library for patching, replacing and decorating .NET and Mono methods during runtime.
  - Patch at runtime (pre- and postfix)
  - Transpile at compile time (rewrite IL)
- Harmony v2
  - Lib.Harmony on NuGet
  - https://github.com/pardeike/Harmony

Image: C# logo https://docs.microsoft.com/en-us/dotnet/csharp/

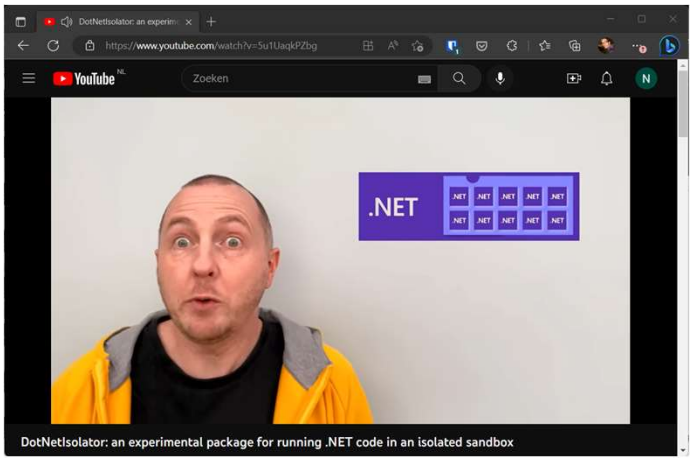
ConsoleApp & Sandboxed Library



{Saturday} dot ned

@nielstanis@infosec.exchange

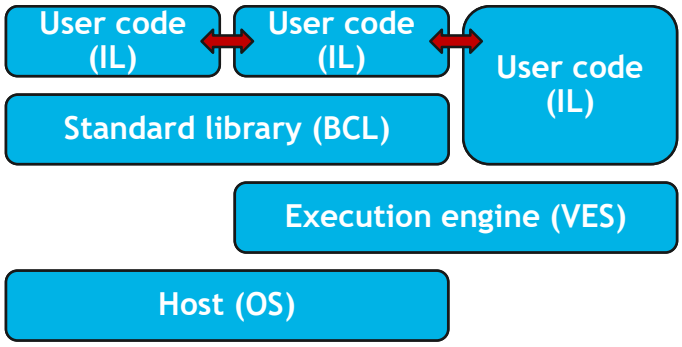
DotNetIsolator



{Saturday} dotned

@nielstanis@infosec.exchange

Running .NET on WebAssembly



{Saturday} dot ned

@nielstanis@infosec.exchange

Diagram:
https://github.com/itowlson/wasmday22/blob/main/slides/Wasm%20Interfaces%20and%20.NET.pptx

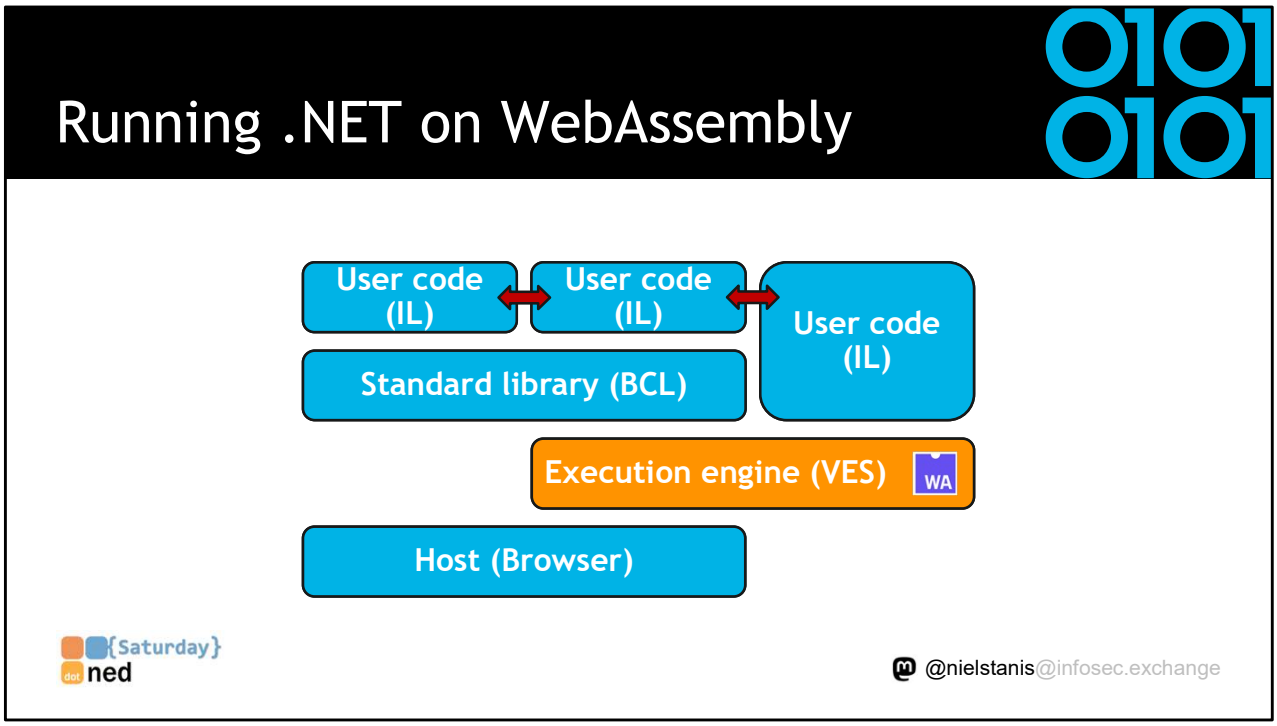
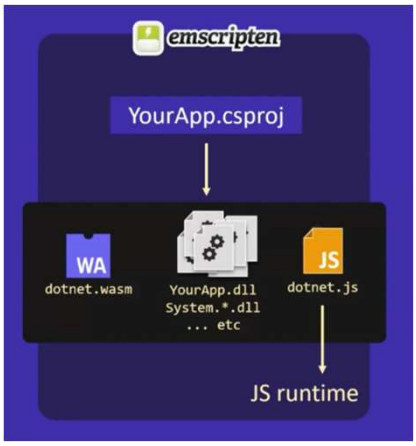


Diagram:
https://github.com/itowlson/wasmday22/blob/main/slides/Wasm%20Interfaces%20and%20.NET.pptx

Blazor WebAssembly



{Saturday} dotned

@nielstanis@infosec.exchange

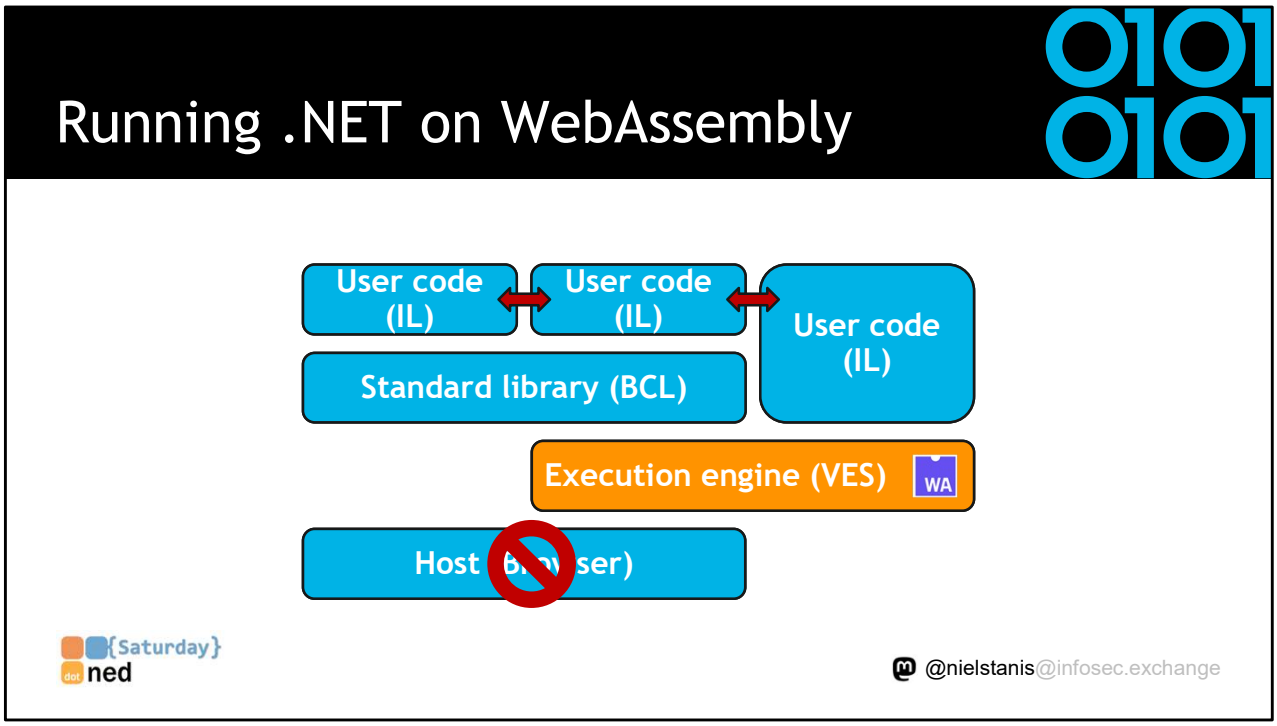
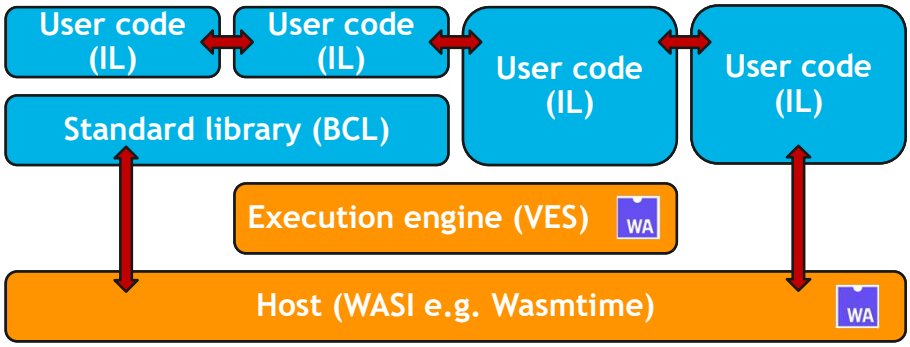


Diagram: https://github.com/itowlson/wasmday22/blob/main/slides/Wasm%20Interfaces%20and%20.NET.pptx

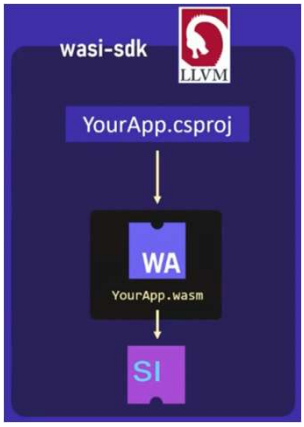
WebAssembly System Interface WASI



{Saturday} ned

@nielstanis@infosec.exchange

Demo DotNetIsolator

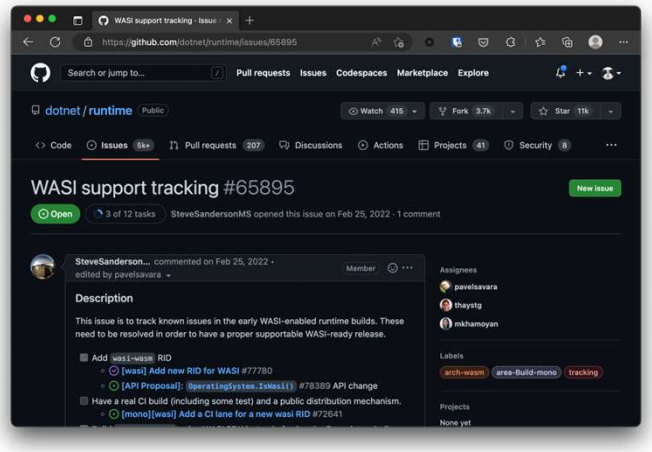


{Saturday} ned

@nielstanis@infosec.exchange

https://github.com/SteveSandersonMS/dotnet-wasi-sdk

Experimental WASI SDK for .NET



{Saturday} dotned

@nielstanis@infosec.exchange

https://github.com/dotnet/runtime/issues/65895
https://github.com/SteveSandersonMS/dotnet-wasi-sdk

Conclusion

- Update libraries; security problems get fixed
- Integrate security into your development lifecycle
- Know what libraries are used, where and what's inside and most important what you'd expect from it.

{Saturday} dot ned

@nielstanis@infosec.exchange

Conclusion

- Futures of this Sandbox Concept
  - Easier developer integration (e.g. source generator)
  - Package + good guidance on how this can be used in different application contexts like ASP.NET Core.
  - Basic patches/policy that can be applied on libraries
- Using WebAssembly to run, extend, and secure your .NET Application talk (NDC Security 2023)

{Saturday} dot ned

@nielstanis@infosec.exchange

VERACODE

Thanks! Questions?

https://github.com/nielstanis/dotnetsaturday2023
ntanis at veracode.com
@nielstanis@infosec.exchange on Mastodon