



Using WebAssembly to run, extend, and secure your .NET application

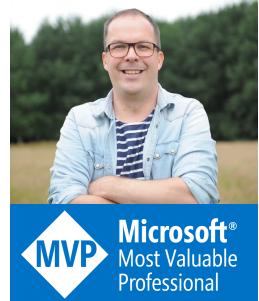
Niels Tanis
Sr. Principal Security Researcher



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

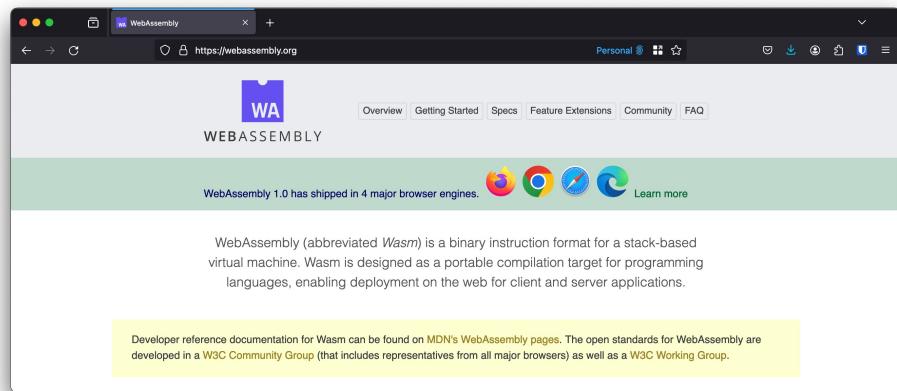


MVP Microsoft®
Most Valuable
Professional

ElevateDev'24

 @nielstanis@infosec.exchange

WebAssembly

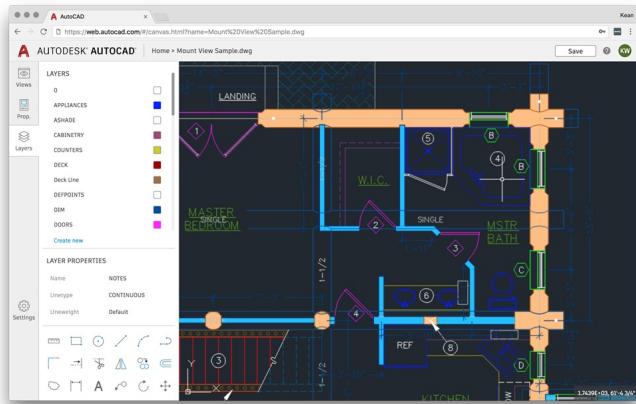


ElevateDev'24

 @nielstanis@infosec.exchange

<https://webassembly.org/>

WebAssembly - AutoCAD



ElevateDev'24

@nielstanis@infosec.exchange

WebAssembly - SDK's

The screenshot shows a Medium article page. At the top, there's a dark header with the title 'Introducing the Disney+ Application Development Kit (ADK)'. Below the header, the author's profile picture and name 'Mike Hanley' are shown, along with a 'Follow' button. The article summary reads: 'Published in disney-streaming'. The main content starts with the heading 'Introducing the Disney+ Application Development Kit (ADK)'. It includes a bio for Tom Schroeder, Sr. SWE / Technical Lead, Native Client Platform, Living Room Devices. The article has 415 views and 4 comments. There are also sections for 'Related' and 'Recent' articles.

The screenshot shows a blog post from 'amazon | science' under the 'CLOUD AND SYSTEMS' category. The title is 'How Prime Video updates its app for more than 8,000 device types'. The post discusses how switching to WebAssembly improves stability and speed. It was written by Alexandru Enă and published on January 27, 2022. The post includes a 'Share' button and a note about delivering content to millions of customers.

ElevateDev'24

@nielstanis@infosec.exchange

<https://medium.com/disney-streaming/introducing-the-disney-application-development-kit-adk-ad85ca139073>

<https://www.amazon.science/blog/how-prime-video-updates-its-app-for-more-than-8-000-device-types>

Agenda

- Introduction
- WebAssembly Design & Internals
- Running .NET on WebAssembly
- Extending .NET with WebAssembly
- Securing .NET with WebAssembly
- Conclusion
- Q&A

ElevateDev'24

 @nielstanis@infosec.exchange

WebAssembly Design

- **Be fast, efficient, and portable**

- Executed in near-native speed across different platforms

- **Be readable and debuggable**

- In low-level bytecode but also human readable

- **Keep secure**

- Run on sandboxed execution environment

- **Don't break the web**

- Ensure backwards compatibility



WEBASSEMBLY

ElevateDev'24

 @nielstanis@infosec.exchange

WebAssembly

- Binary instruction format for stack-based virtual machine similar to .NET CLR running MSIL or JVM running bytecode
- Designed as a portable compilation target



WEBASSEMBLY

ElevateDev'24

@nielstanis@infosec.exchange

<https://hacks.mozilla.org/2017/02/creating-and-working-with-webassembly-modules/>
<https://webassembly.org/>

WebAssembly

- The security model of WebAssembly:
 - Protect users from buggy or malicious modules
 - Provide developers with useful primitives and mitigations for developing safe applications



WEBASSEMBLY

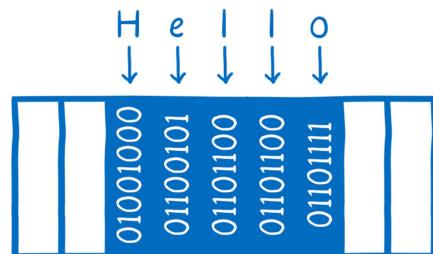
ElevateDev'24

@nielstanis@infosec.exchange

<https://hacks.mozilla.org/2017/02/creating-and-working-with-webassembly-modules/>
<https://webassembly.org/>

WebAssembly Memory

- Isolated per WASM module
- A contiguous, mutable array of uninterpreted bytes



ElevateDev'24

@nielstanis@infosec.exchange

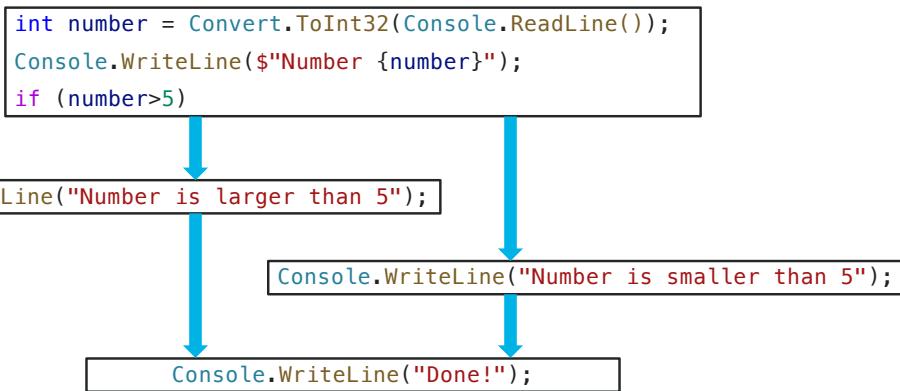
WebAssembly Control-Flow Integrity

```
int number = Convert.ToInt32(Console.ReadLine());
Console.WriteLine($"Number {number}");
if (number>5)
{
    Console.WriteLine("Number is larger than 5");
}
else
{
    Console.WriteLine("Number is smaller than 5");
}
Console.WriteLine("Done!");
```

ElevateDev'24

 @nielstanis@infosec.exchange

WebAssembly Control-Flow Integrity



ElevateDev'24

@nielstanis@infosec.exchange

FireFox RLBox

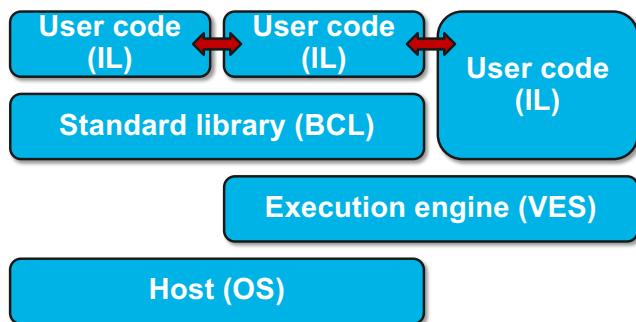
ElevateDev'24

@nielstanis@infosec.exchange

<https://rlbox.dev/>

<https://hacks.mozilla.org/2020/02/securing-firefox-with-webassembly/>

Running .NET on WebAssembly



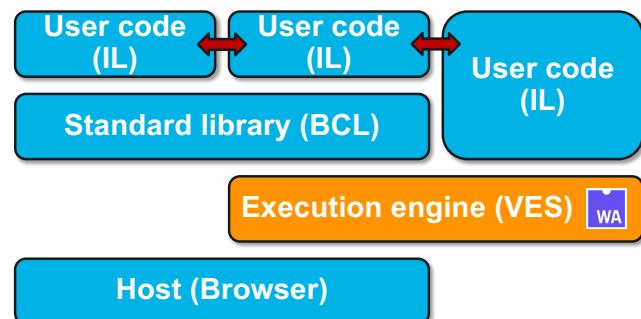
ElevateDev'24

@nielstanis@infosec.exchange

Diagram:

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

Running .NET on WebAssembly



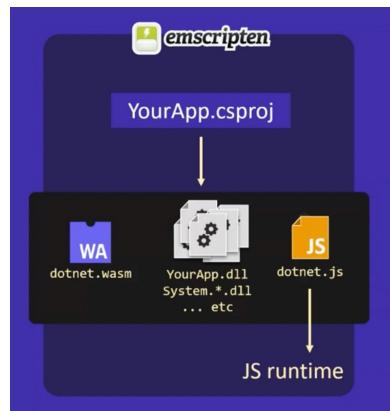
ElevateDev'24

@nielstanis@infosec.exchange

Diagram:

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

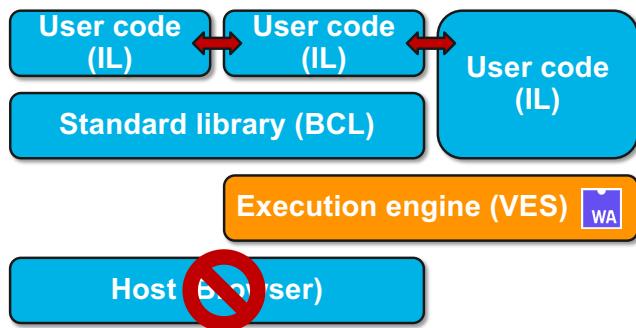
Blazor WebAssembly



ElevateDev'24

@nielstanis@infosec.exchange

Running .NET on WebAssembly



ElevateDev'24

@nielstanis@infosec.exchange

Diagram:

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

WebAssembly System Interface WASI

- Introduced in March 2019 by Bytecode Alliance
- WasmTime implementation as reference
- POSIX inspired, engine-independent, non-Web system-oriented API for WebAssembly



ElevateDev'24

@nielstanis@infosec.exchange

WebAssembly System Interface WASI

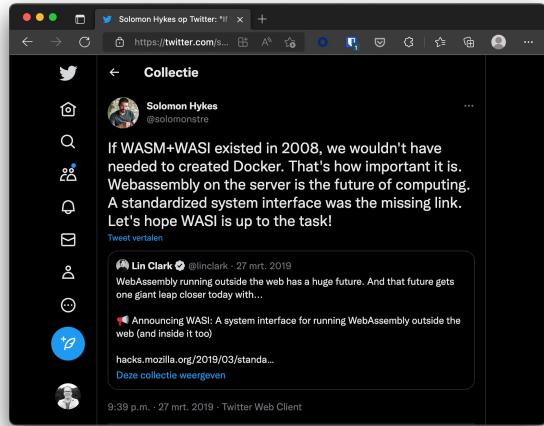
- Strong sandbox with Capability Based Security
- Preview1, supports e.g. FileSystem actions
- Future support for sockets and other system resources.
- Anyone recall .NET Standard? 😊



ElevateDev'24

@nielstanis@infosec.exchange

Docker vs WASM & WASI



ElevateDev'24

@nielstanis@infosec.exchange

Docker vs WASM & WASI

The screenshot shows a Twitter browser window with a dark theme. The tweet is from Solomon Hykes (@solomonstre) and reads:
"So will wasm replace Docker?" No, but imagine a future where Docker runs linux containers, windows containers and wasm containers side by side. Over time wasm might become the most popular container type. Docker will love them all equally, and run it all :)
Tweet vertalen

Below the tweet, there is a reply from Solomon Hykes: "If WASM+WASI existed in 2008, we wouldn't have needed to created Docker. That's how important it is. Webassembly on the server is the future of computing. A standardized system interface was the missing link. Let's hope WASI is up to the task! [twitter.com/lmclark/status...](#)"

At the bottom of the tweet card, it says "Dit bericht is nu beschikbaar in de verzamelingen van [Solomon Hykes](#)".

On the left side of the window, there is a sidebar with various icons: a Twitter icon, a home icon, a search icon, a retweet icon, a reply icon, a message icon, a profile icon, and a blue circular icon.

At the bottom left of the slide, the text "ElevateDev'24" is visible. At the bottom right, there is a small profile picture and the handle "@nielstanis@infosec.exchange".

Docker & WASM

The image contains two screenshots of a Docker+Wasm Technical Preview web page. The left screenshot shows the homepage with a title 'Introducing the Docker+Wasm Technical Preview' and author information. The right screenshot shows a diagram of the Docker architecture with a Wasm module integrated into the containerd stack.

Introducing the Docker+Wasm Technical Preview

MICHAEL IRWIN
Oct 24 2022

The Technical Preview of Docker+Wasm is now available! Wasm has been producing a lot of buzz recently, and this feature will make it easier for you to quickly build applications targeting Wasm runtimes.

Docker Engine

container^d

```
graph TD; DockerEngine[Docker Engine] --> ContainerD[containerd]; ContainerD --> ContainerdShim1[containerd-shim]; ContainerdShim1 --> Runc1[runc]; Runc1 --> ContainerProcess1[Container process]; ContainerdShim2[containerd-shim]; Runc2[runc]; ContainerProcess2[Container process]; ContainerdWasmShim[containerd-wasm -shim]; WasmEdge[wasmedge]; WasmModule[Wasm Module]; ContainerdWasmShim --> WasmEdge; WasmEdge --> WasmModule;
```

Let's look at an example!

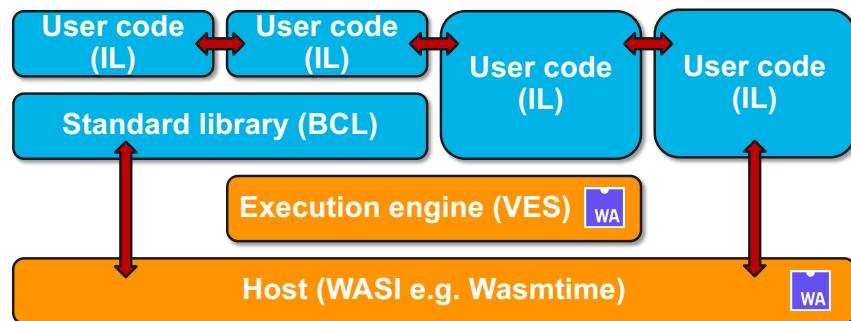
After installing the preview, we can run the following command to start an example Wasm application:

```
docker run -dp 8080:8080 --name=wasm-example --runtime=io.containerd.wasmedge.v1 --platform=wasi/wasm32 michaelirwin244/wasm-example
```

ElevateDev'24

@nielstanis@infosec.exchange

WebAssembly System Interface WASI



ElevateDev'24

👤 @nielstanis@infosec.exchange

Experimental WASI SDK for .NET

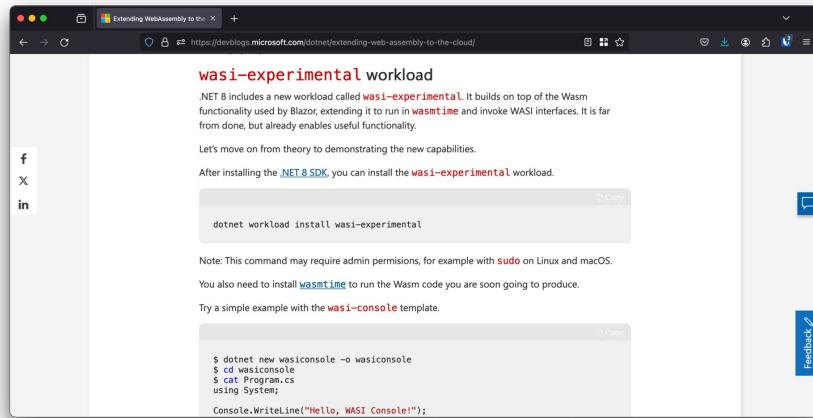


ElevateDev'24

 @nielstanis@infosec.exchange

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

.NET 8 WASI-Experimental



ElevateDev'24

 @nielstanis@infosec.exchange

<https://github.com/dotnet/runtime/issues/65895>

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

<https://devblogs.microsoft.com/dotnet/extending-web-assembly-to-the-cloud/>

Extending .NET with WASM

- WasmTime.NET NuGet package
- Can run WASM inside of any .NET application
- Extend with Rust based WASM module
- Limit capabilities
- Demo time!

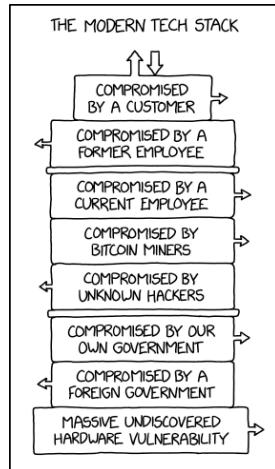
ElevateDev'24

 @nielstanis@infosec.exchange

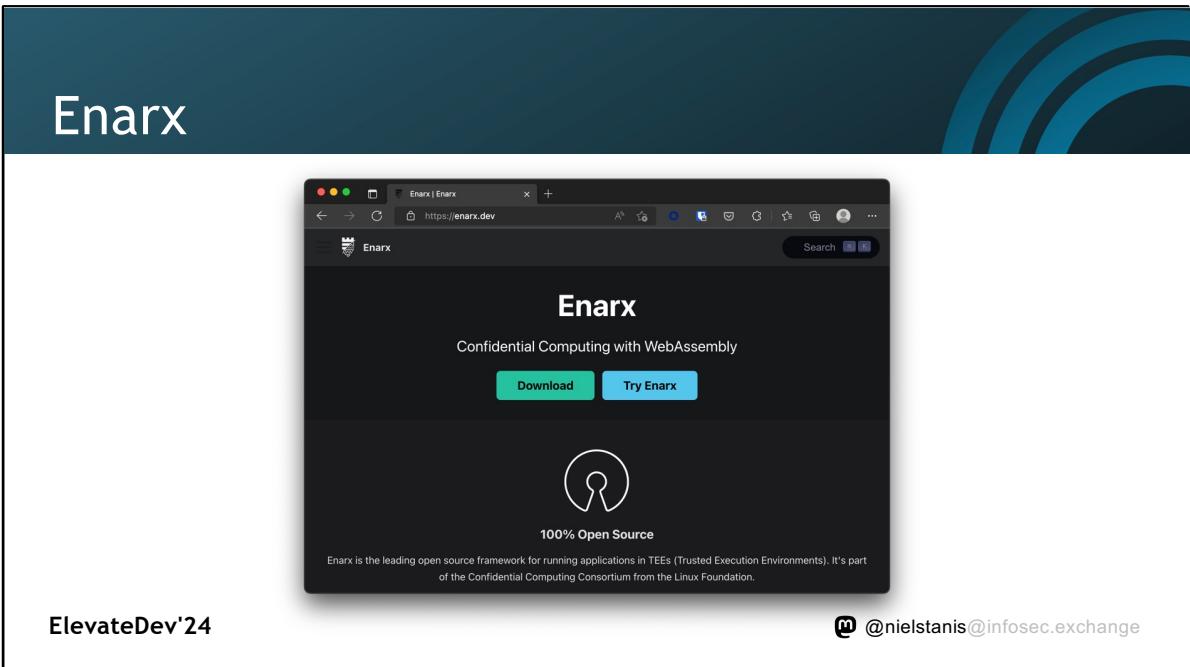
Trusted Computing - XKCD 2166

ElevateDev'24

 @nielstanis@infosec.exchange



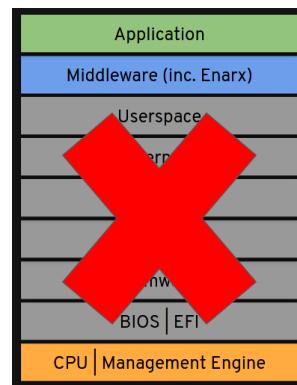
<https://xkcd.com/2166/>



<https://enarx.dev/>

Enarx Threat Model

- Don't trust the host
- Don't trust the host owner
- Don't trust the host operator
- Hardware cryptographically verified
- Software audited and cryptographically verified

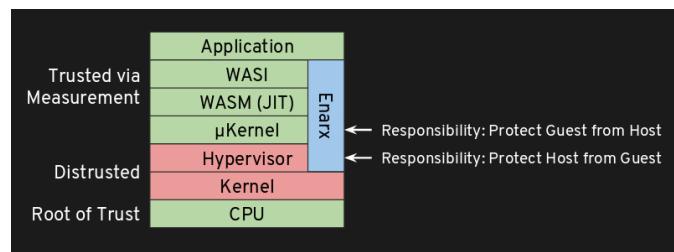


ElevateDev'24

 @nielstanis@infosec.exchange

Enarx

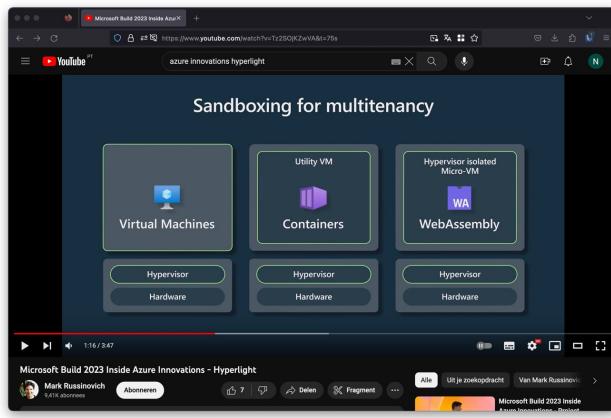
- Leverages Trusted Execution Environment (TEE) direct on processor
 - AMD's SEV, Intel's SGX and IBM's PEF
- Attestation of hardware and Enarx runtime



ElevateDev'24

@nielstanis@infosec.exchange

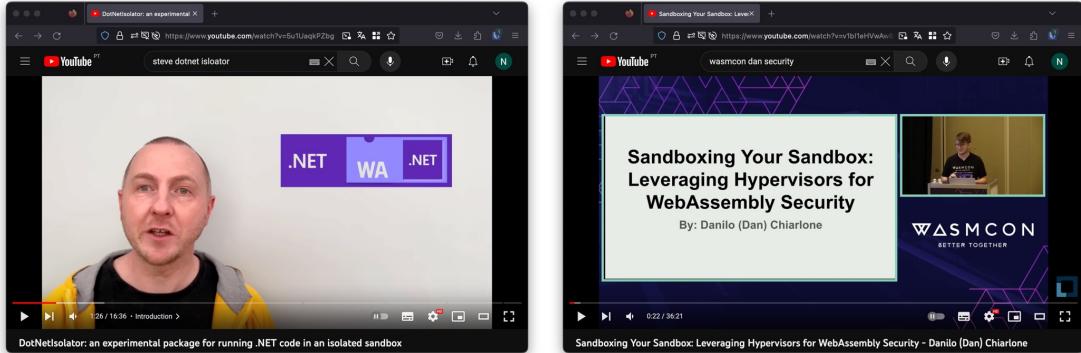
Project Hyperlight



ElevateDev'24

@nielstanis@infosec.exchange

DotNetIsolator & Project Hyperlight

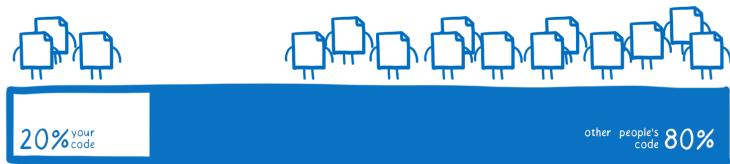


ElevateDev'24

 @nielstanis@infosec.exchange

WASM - What's next?

composition of an
average code base



ElevateDev'24

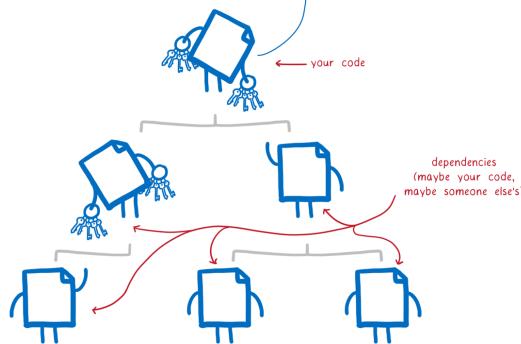
 @nielstanis@infosec.exchange

Dependencies

ElevateDev'24

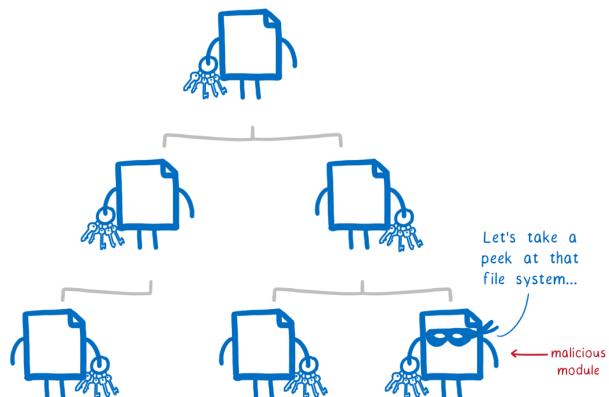
Here are the keys
to the castle.

Make copies and
pass them down to
your dependencies.



iis@infosec.exchange

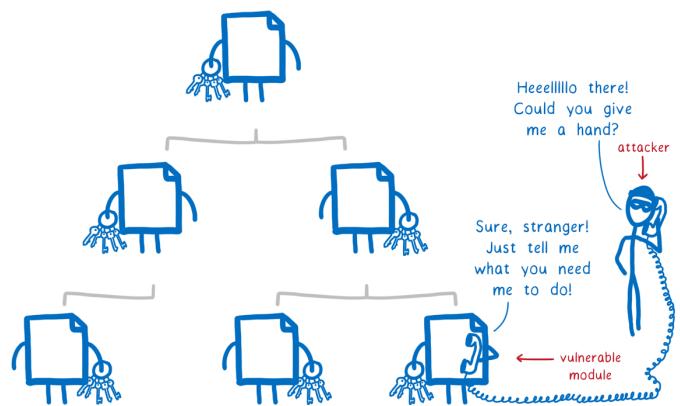
Malicious module



ElevateDev'24

@nielstanis@infosec.exchange

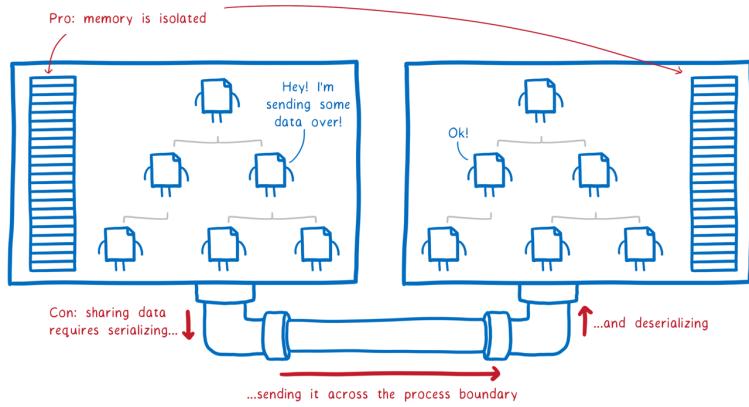
Vulnerable module



ElevateDev'24

@nielstanis@infosec.exchange

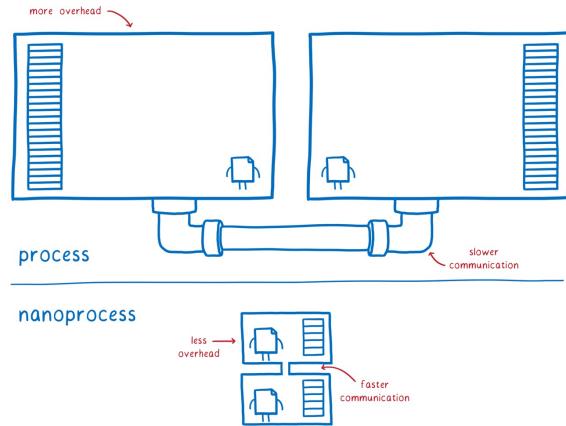
Process Isolation



ElevateDev'24

@nielstanis@infosec.exchange

WebAssembly Nano-Process

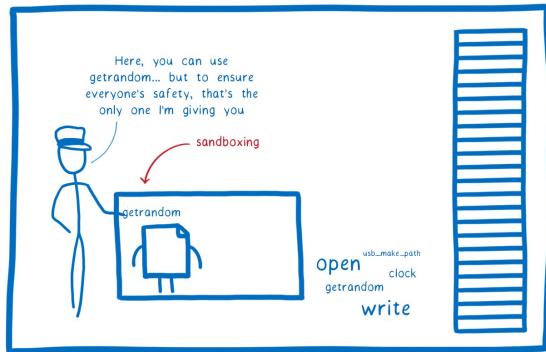


ElevateDev'24

* not drawn to scale
@nielstanis@infosec.exchange

WebAssembly Nano-Process

1. Sandboxing

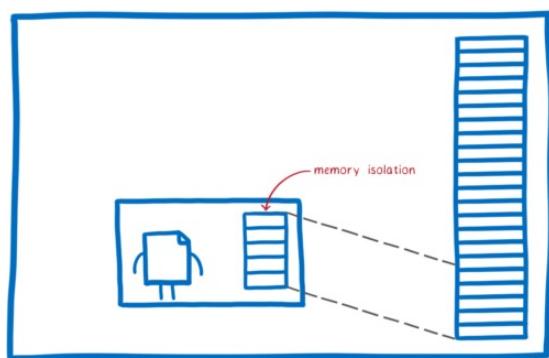


ElevateDev'24

 @nielstanis@infosec.exchange

WebAssembly Nano-Process

2. Memory model

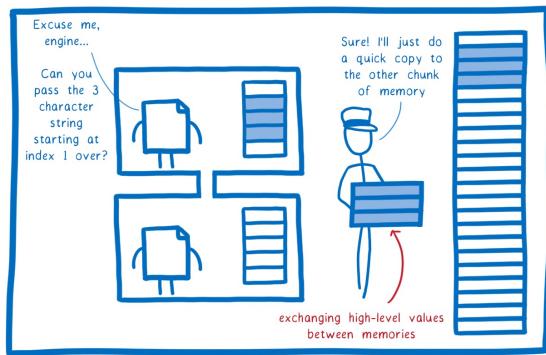


ElevateDev'24

@nielstanis@infosec.exchange

WebAssembly Nano-Process

3. Interface Types

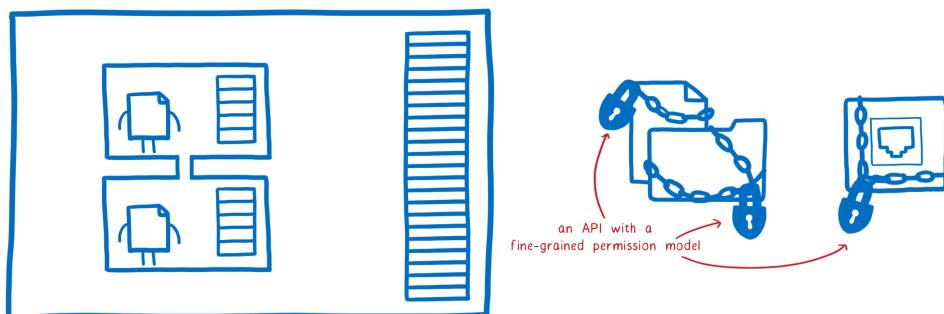


ElevateDev'24

 @nielstanis@infosec.exchange

WebAssembly Nano-Process

4. WebAssembly System Interface

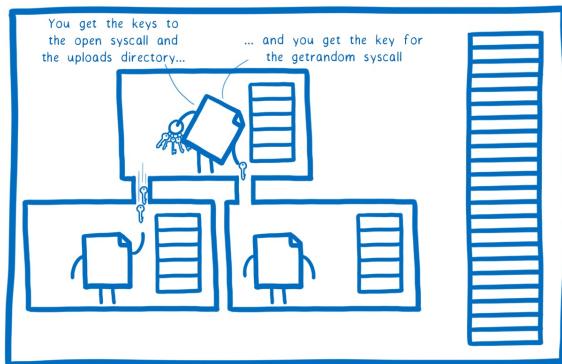


ElevateDev'24

 @nielstanis@infosec.exchange

WebAssembly Nano-Process

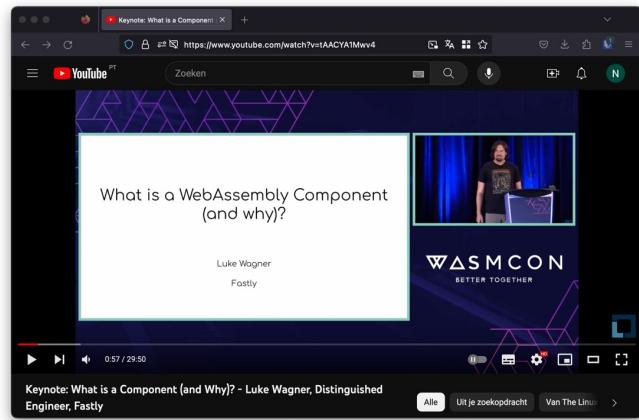
5. The missing link



@nielstanis@infosec.exchange

ElevateDev'24

WebAssembly Component Model



ElevateDev'24

 @nielstanis@infosec.exchange

<https://www.youtube.com/watch?v=tAACYA1Mwv4>

WASI Preview 2

The screenshot shows a web browser window with the title "WASI Preview 2 Launched - sunfishcode" and the URL "https://blog.sunfishcode.online/wasi-preview2/". The page content is as follows:

sunfishcode's blog
A blog by sunfishcode

WASI Preview 2 Launched

Posted on January 25, 2024

The WASI Subgroup has just voted to launch WASI Preview 2! This blog post is a brief look at the present, past, and future of WASI.

The present

The Subgroup voted to launch Preview 2!

This is a major milestone! We made it! At the same time, the journey is only just beginning. But let's talk this moment to step back and look at what this means.

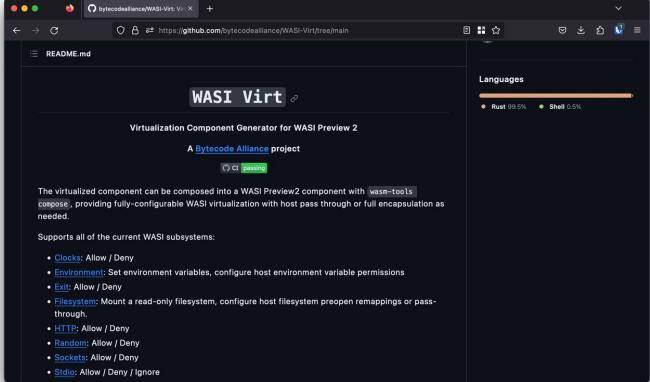
Most immediately, what this means is that the WASI Subgroup officially says that the Preview 2 APIs are stable. There is still a lot more to do, in writing more documentation, more tests, more toolchains, more implementations, and there are a lot more features that we all want to add. This vote today is a milestone along the way, rather than a destination in itself.

It also means that WASI is now officially based on the Wasm component model, which makes it cross-language and virtualizable. Figuring out what a component model even is, designing it, implementing it, and building APIs using it has been a

ElevateDev'24

 @nielstanis@infosec.exchange

WASI Virt



The screenshot shows a GitHub repository page for "bytecodealliance/WASI-Virt". The main content is the README.md file, which contains the following text:

WASI Virt

Virtualization Component Generator for WASI Preview 2

A Bytecode Alliance project

The virtualized component can be composed into a WASI Preview2 component with `wasm-tools compose`, providing fully-configurable WASI virtualization with host pass through or full encapsulation as needed.

Supports all of the current WASI subsystems:

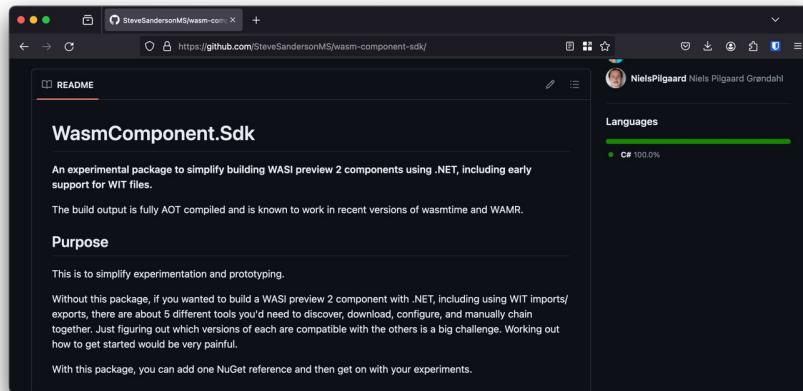
- **Clocks**: Allow / Deny
- **Environment**: Set environment variables, configure host environment variable permissions
- **Exit**: Allow / Deny
- **Filesystem**: Mount a read-only filesystem, configure host filesystem preopen remappings or pass-through.
- **HTTP**: Allow / Deny
- **Random**: Allow / Deny
- **Sockets**: Allow / Deny
- **Stdio**: Allow / Deny / Ignore

Languages: Rust 99.8%, Shell 0.5%

ElevateDev'24  @nielstanis@infosec.exchange

<https://www.youtube.com/watch?v=tAACYA1Mwv4>

WasmComponent.SDK

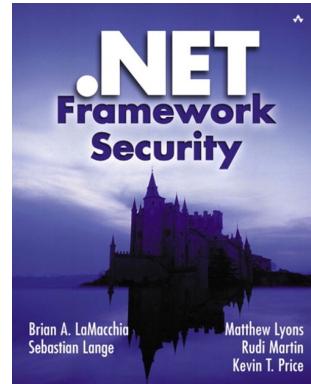
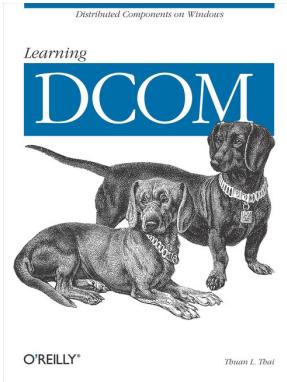


ElevateDev'24

@nielstanis@infosec.exchange

<https://github.com/SteveSandersonMS/wasm-component-sdk/>

Have we seen this before?



ElevateDev'24

@nielstanis@infosec.exchange

Runtimes and Security

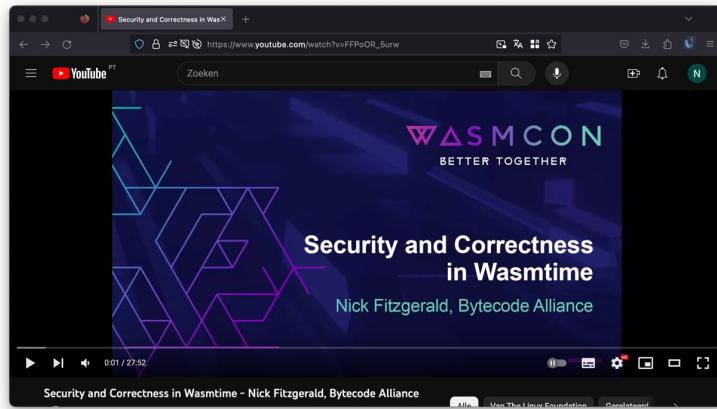
- Most security research published focusses on correctness of WASM runtimes/VM's
- Bytecode Alliance Blogpost September 2022:
 - "Security and Correctness in Wasmtime"
 - Written in Rust → Using all its LangSec features
 - Continues Fuzzing & formal verification
 - Security process & vulnerability disclosure

ElevateDev'24

 @nielstanis@infosec.exchange

<https://bytecodealliance.org/articles/security-and-correctness-in-wasmtime>

Runtimes and Security



ElevateDev'24

 @nielstanis@infosec.exchange

https://www.youtube.com/watch?v=FFPoOR_5urw

Conclusion

- Cloud Native ❤️ WebAssembly
- WebAssembly has a lot of potential to be used to run, extend, and secure your applications!
- Its as secure as the WebAssembly runtime implementation!
- WASI Preview 2 big milestone; now tooling can be implemented!

ElevateDev'24

 @nielstanis@infosec.exchange

Questions?

- <https://github.com/nielstanis/elevatedev24wasm/>
- ntanis at Veracode.com
- @nielstanis@infosec.exchange
- <https://blog.fennec.dev>
- Merci! Thank you!

ElevateDev'24

 @nielstanis@infosec.exchange