Zero Trust in the Cloud

With WebAssembly and wasmCloud

About Me

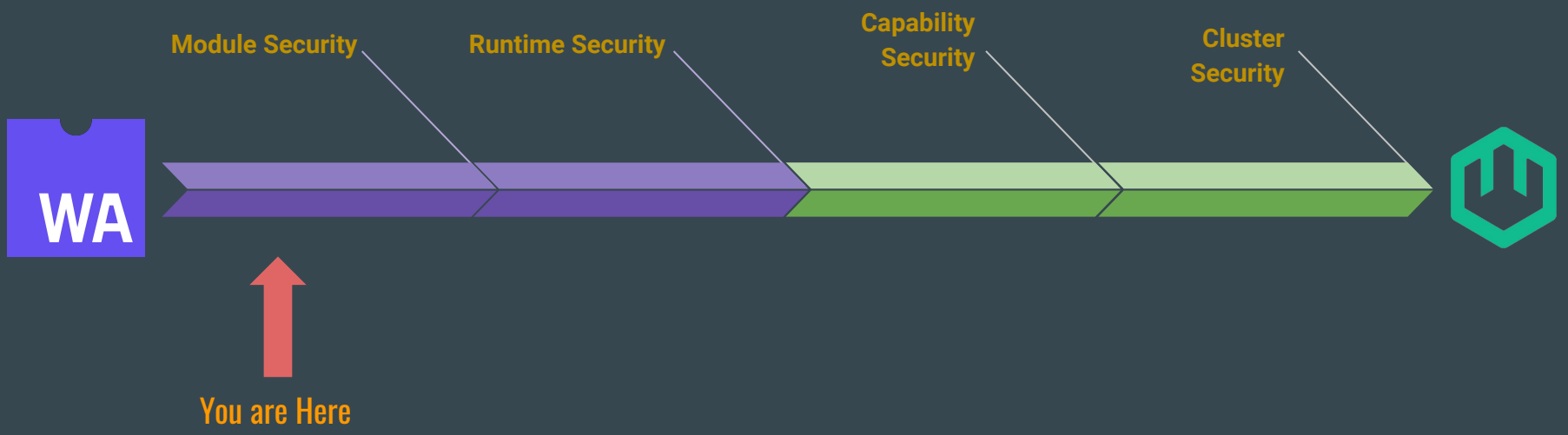
- Author
  - “Programming WebAssembly with Rust”
  - Cloud Native Go
  - Building Microservices with ASP.NET Core
  - ... a lot more
- Creator of the CNCF project wasmCloud
- Co-founder of Cosmonic - a provider of managed wasmCloud services
- Paranoid / Security Proponent
- Big WebAssembly Fan*
- Github: autodidaddict, Fediverse: @autodidaddict@mastodon.world

Constraints Liberate, Liberties Constrain

Runar Bjarnason

@runarorama

Trust No One - the Agenda



Module Security

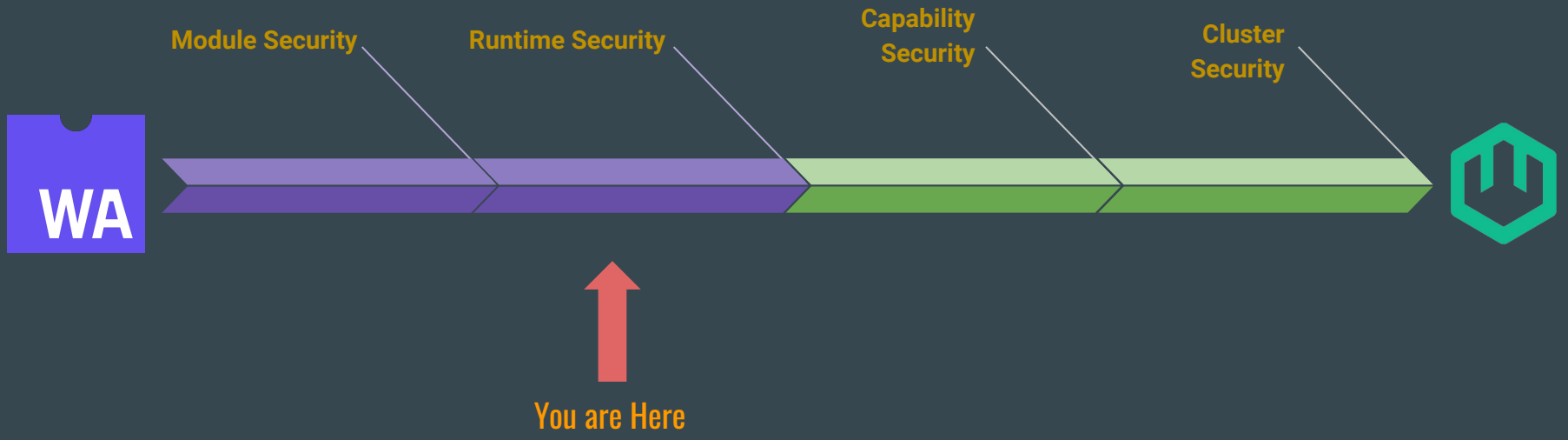
- Call stack can’t buffer overflow
  - https://www.youtube.com/watch?v=fj9u00PMkYU
- Cannot write code that uses a syscall
- Branches always point to valid destinations
- Trap (exception) instead of undefined behavior
  - Bad array index, out of bounds in linear memory
  - Sneaky indirect function all
  - Exceeding call stack size
  - Illegal math operations (/0 etc)

Control Flow Security

- Functions & fixed static scope variables >> pointers
  - Cannot dereference invalid memory
- No Return-oriented Programming (ROP) attacks
- Direct function calls
- Indirect function calls
- Returns
- Safe jumps

WebAssembly Demo

Trust No One


Module Security
Runtime Security
Capability Security
Cluster Security
WA
You are Here

Runtime Security

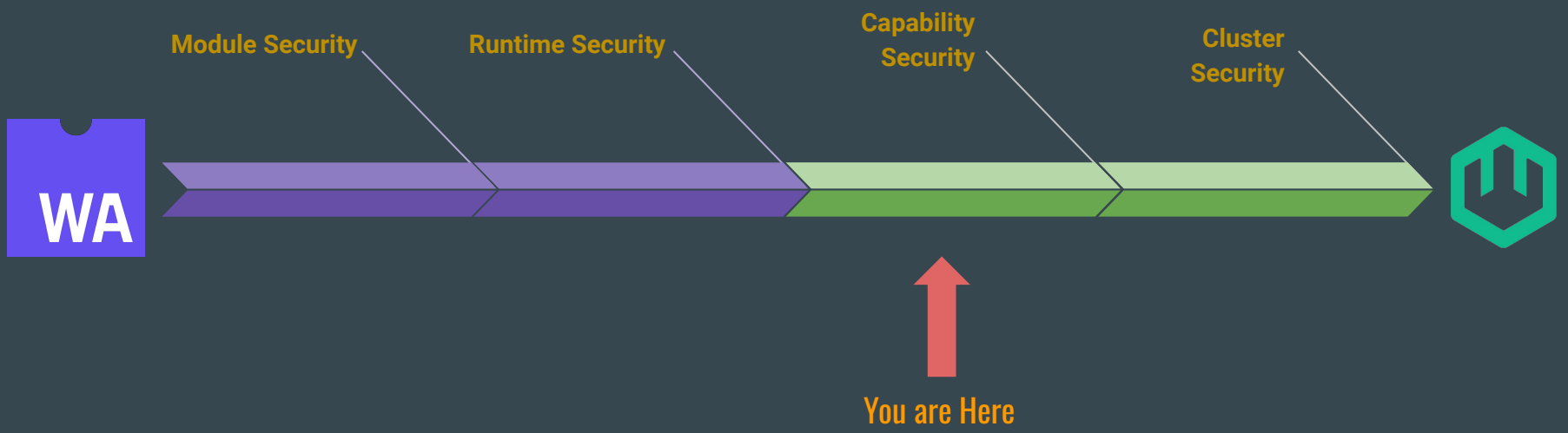
- Host memory is off limits
  - Host and guest module use isolated linear memory block
- Host supplies implementation for module imports
- Only host can trigger code in module
- No I/O opcodes in wasm
- Anti-forgery checks
- JIT up to the host

Runtime Security - WASI

- WASI != POSIX
- Host runtime gets right of refusal
  - I/O
  - All imports and implementation
- WASI requires files and directories to be pre-approved
- Socket I/O but *host creates socket*

WASI Demo

Trust No One - the Agenda



Capability Security

- *Why* do you want that socket?
- High-level abstractions
  - HTTP server
  - HTTP Client
  - Message broker
  - Data stores - SQL, KV, NoSQL
  - Time, random, crypto, etc

Capability Security - Constraints Influence Better Design

```
let x = keyvalue().increment(“securityday”, 1)?;
let pong = messaging().request(“security.day”, b”ping!”)?;
```

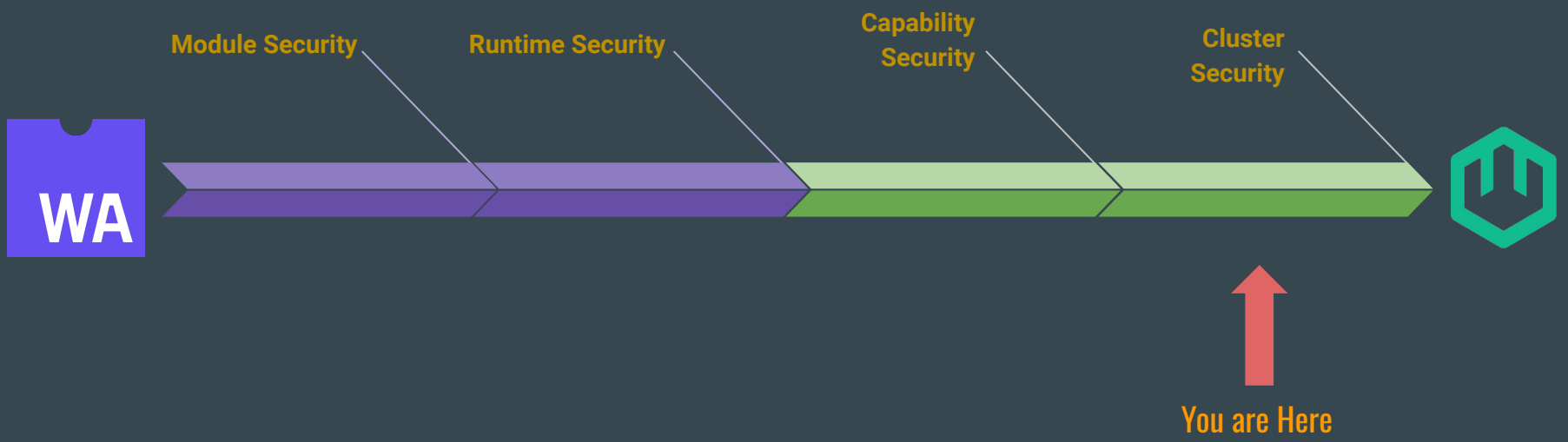
```
impl HttpServer for MyActor {

  fn handle_request(&self, value: &HttpRequest) -> Result<HttpResponse>
  {
    Ok(HttpResponse::ok(“PONG”))
  }

}
```

Enhanced Module Security

- Embed JWT in WebAssembly module
  - Verifiable in isolation / offline
  - Cryptographic signature
  - Tamper proof module (via hash)
- Claims & Attestations
  - Capability allow list
  - Issuer & subject
  - Expiration and valid-after

wasmCloud Capabilities Demo

Trust No One - the Agenda



Cluster Security

- Each host in the cluster has a key pair
- Each host trusts only a set of public keys
- Invocations & Comms all signed and hash verified
- Policy enforcement
  - Control what can start / load
  - Control which actors (.wasm) can talk to which other actors

wasmCloud Cluster Security Demo

Q&A

- https://webassembly.org/
- https://github.com/WebAssembly/WASI
- https://wasmcloud.com
- https://github.com/wasmcloud
- https://cosmonic.com
- https://mastodon.world/@autodidaddict

SUBMIT FEEDBACK ⬇️