ATS Summit Fall 2022
ATS Wasm Plugin

Kit Chan (kichan@apache.org)
Extending ATS

- **C/C++ plugins**
  - Allow extension of HTTP/TLS handling for connections with clients and origins
  - Steep Learning curve

- **header_rewrite scripts / txn box**
  - Domain Specific Language - not turing complete, no unit test framework
  - Cover some capabilities of the C/C++ plugins

- **Lua plugins**
  - Easier to learn a scripting language
  - Cover most of the capabilities of C/C++ plugins
  - LuaJIT FFI allows integration with shared libraries (bindings needed)
  - Not popular - fewer people learning the language / less support for bindings to newer libraries
Proxy-Wasm

- WebAssembly for Proxies
- Spec - https://github.com/proxy-wasm/spec
  - Classes for integration with proxy
  - Integrate with different runtime - V8, WAVM, WAMR, Wasmtime, WasmEdge
- SDK - Help to compile programs to wasm modules following the spec
  - C++ - https://github.com/proxy-wasm/proxy-wasm-cpp-sdk
  - Available for AssemblyScript, Tiny Go, Zig as well
- Proxy Implementations
  - Envoy
  - MOSN
  - Nginx
ATS Plugin Architecture

- With handler functions for proxy to call (1)
- Calling API functions that the proxy provides (2)
Code Structure

- Proxy-Wasm library provides base classes of WasmBase, ContextBase
- ATS Wasm Plugin provides extended classes of them
  - Wasm - initializing the runtime, load the module and configuration
  - Context - provide implementations for handler functions and API functions
    - Root Context - created from Wasm during ATS startup
    - Non-root context - created from root context for each transaction
Demo
Demo Summary

- **C++ Example** -
  - Demonstrate HTTP handler functions
  - Demonstrate logging
  - Demonstrate getting / setting HTTP headers
  - Demonstrate getting timestamp

- **Rust example** -
  - Demonstrate modules written in another language
Real World Example

- **Coraza** - [https://github.com/corazawaf/coraza](https://github.com/corazawaf/coraza)
  - Open Source WAF library
  - supports ModSecurity SecLang rulesets
  - Written in Go
- **Coraza Proxy WASM** - [https://github.com/corazawaf/coraza-proxy-wasm](https://github.com/corazawaf/coraza-proxy-wasm)
  - WASM filter to be used with Envoy
  - Compiled with TinyGo SDK
- We can download the wasm module and use on ATS with the wasm plugin
Open Source

- To-do list
  - Currently only the WAVM runtime is supported. We need to also support V8, WAMR, Wasmtime, and WasmEdge as well.
  - Need to support functionality for retrieving and setting request/response body
  - Need to support functionality for making async request call
  - Need to support L4 lifecycle handler functions
  - Support loading more than one Wasm module
- More production testing / performance testing
Limitations

● A few things we won’t be able to support in the spec
  ○ Getting and setting trailer request and response header
  ○ Getting and setting data in HTTP/2 meta data frame
  ○ Support on GRPC lifecycle handler functions
Use Cases

● Safety for complex plugins
  ○ Critical to business
  ○ Bugs can cause ATS to crash
  ○ With implementation as Wasm modules, bugs will only cause Wasm runtime to complain

● Allow us to use these plugins with Envoy and vice versa

● Programming in more popular languages - Rust / Go