

WebAssembly Plugin for Apache Traffic Server

Shu Kit Chan

Yahoo

10/10/2023

yahoo!

Who? What?



Kit Chan

(kichan@apache.org / kichan@yahooinc.com)

- 18 years in Yahoo
- Software Architect in Media Infrastructure Team
- Volunteer in OSPO

- Apache Traffic Server PMC/Committer
- Wasm, Lua, ESI plugins

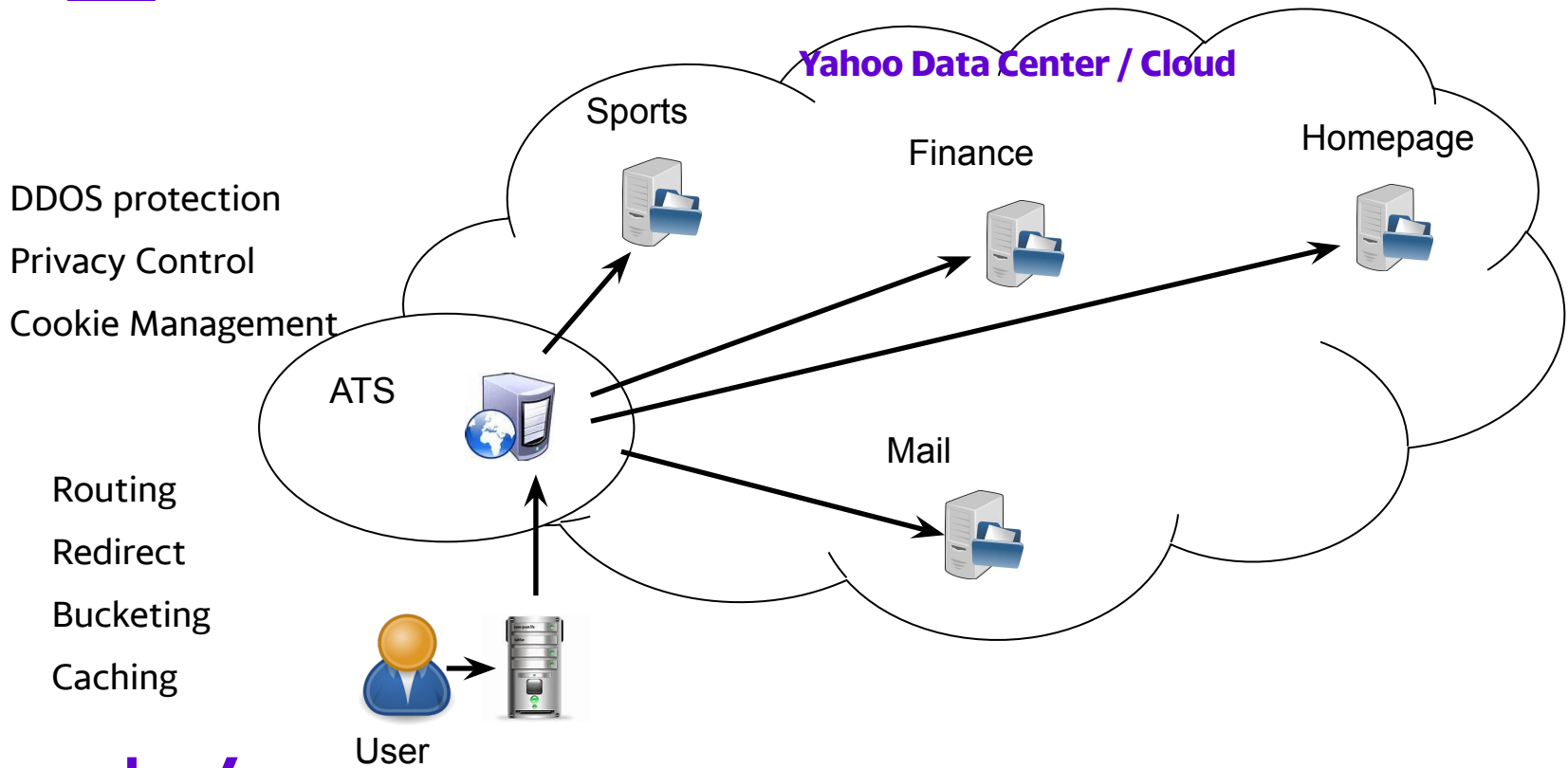
- WebAssembly + ATS == Framework to build functionality on your edge!

yahoo!

Apache Traffic Server

yahoo!

ATS & Yahoo!



Extending ATS / Building Features

- C++ plugins
 - Allow extension of HTTP/TLS handling for connections with clients and origins
 - Steep Learning curve
- Domain Specific Languages plugin (header_rewrite / txn_box / etc)
 - Invented language, Not turing complete, no unit test framework
 - Hard to expand
- Lua plugin
 - Easier to learn a scripting language
 - LuaJIT FFI allows integration with shared libraries (bindings needed)
 - Popularity?

Proxy-Wasm & Wasm Plugin

Proxy-Wasm

- WebAssembly for Proxies
- Specification
 - ongoing efforts!
 - Like a “WASI for Proxy”
 - Community with tie to “wasi-http”
- Partial WASI support is part of the Spec
 - `environ_get` / `environ_sizes_get`
 - `random_get`
 - `clock_time_get`
 - `fd_write` (for `stderr`, `stdout`)

Proxy-Wasm

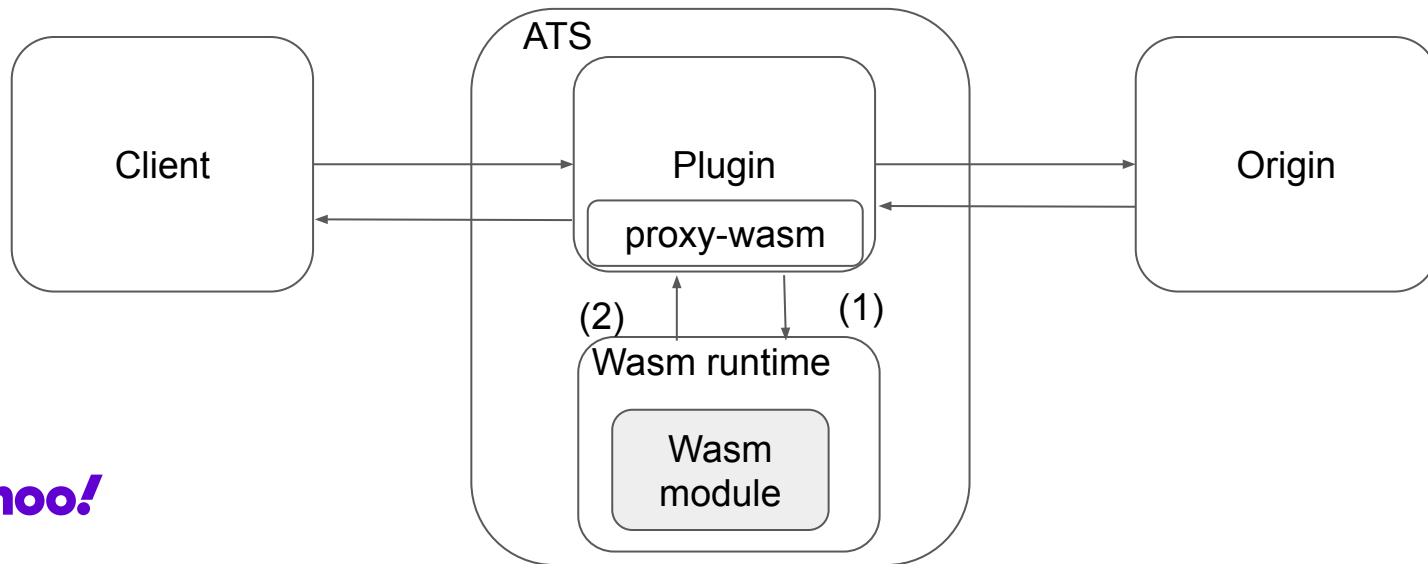
- Library
 - Implement the spec and provide integration with proxy
 - Integrate with different runtime - WAMR, Wasmtime, WasmEdge, V8
 - Existing Proxy Implementations - Envoy, MOSN, Nginx, ATS
- SDK - Help to compile programs to wasm modules following the spec
 - Official - C++, Rust
 - Third party - AssemblyScript, TinyGo, Zig

Benefits

- Support many programming languages
- Standard/specification promote interoperability
- Safety with Sandboxed approach
- Promising future
 - Vibrant/Passionate Communities
 - Upcoming proposals like Component Model, garbage collection, etc

ATS Plugin Architecture

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



Example in Rust (Snippet)

```
impl HttpContext for HttpHeaders {  
    fn on_http_request_headers(&mut self, _: usize, _: bool) -> Action {  
        for (name, value) in &self.get_http_request_headers() {  
            let s3 = format!("In WASM: #{} -> {}: {}", self.context_id, name,  
value);  
            trace!("{}", s3);  
        }  
        if let Some(ua) = self.get_http_request_header("User-Agent") {  
            if ua != "" {  
                trace!("UA is {}", ua);  
            }  
        }  
    }  
}
```

```
match self.get_http_request_header("token") {  
    Some(token) if token.parse::        trace!("It is prime!!!");  
        Action::Continue  
    }  
    _ => {  
        trace!("It is not prime!!! That's true.");  
        self.send_http_response(  
            403,  
            vec![("Powered-By", "proxy-wasm")],  
            Some(b"Access forbidden.\n"),  
        );  
        Action::Pause  
    }  
}  
}  
}
```

yahoo!

Real World Example 1 - WAF

- Coraza
 - Open Source WAF library in Go
 - Compatible with ModSecurity
- Coraza Proxy-wasm module
 - WASM module to be used with Envoy
 - Compiled with TinyGo SDK
- It works with ATS with the Wasm Plugin!

Real World Example 2- AI Inference with WASI-nn

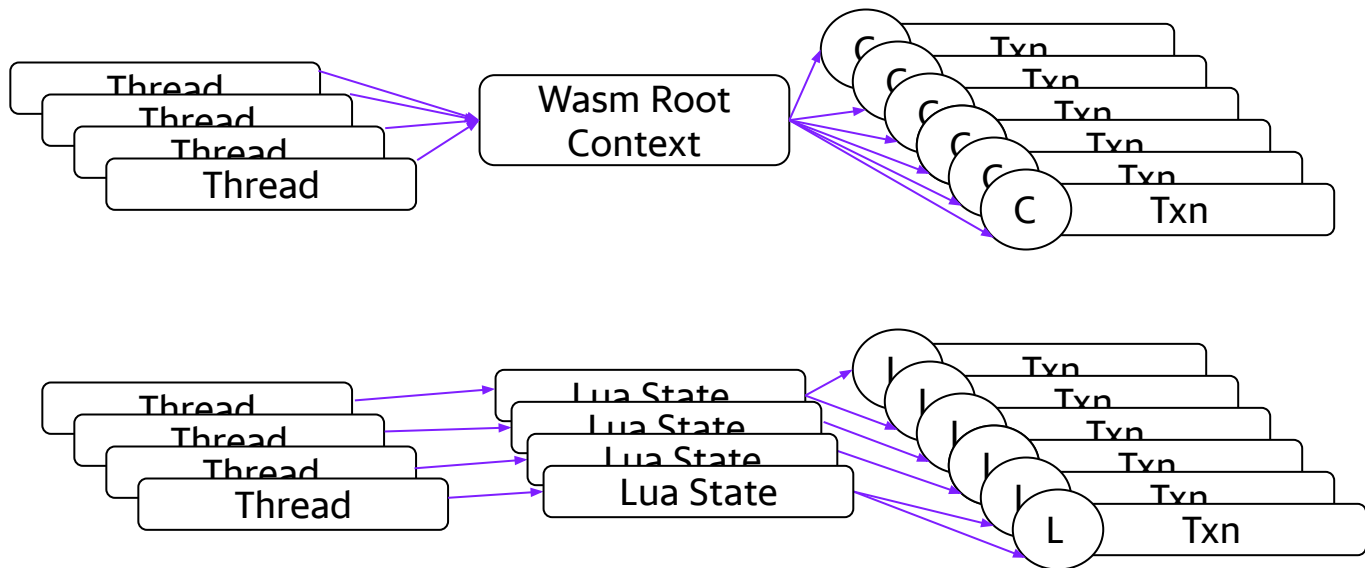
- WASI-nn - proposed WASI API for machine learning
- Proxy can do AI Inference with deployed model
- Need to be supported by underlying WASM runtime
- Example coming soon!

Limitations

- A few things we won't be able to support in proxy-wasm 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
- Extra functionality in ATS
 - E.g. caching API
 - Can be implemented outside of spec
 - But it will break interoperability

Performance Testing

- Experiments done between Lua script, DSL script and Wasm module
- Lua script / DSL script < Wasm module -> LuaJIT is AWESOME!!!
- Resource Contention inside Wasm plugin -



Other Tips / Techniques for Wasm Optimization

- Language Choice
- AOT - ahead of time compilation
- Compiler Flags
- wasm-opt
- Choice of Runtime?

Wasm Runtimes

Difficult to Choose

- The field evolves rapidly
- Each with different characteristics
- Change of runtime only possible for simple program
- Major investment involved when tools are used (e.g. profiling / debugging)
 - WAMR/Wasmtime - live debug support through lldb
 - Wasmtime - profiling with perf
- Different WASM proposals supported by different runtime
- Trust in Security
 - Choice of implementation language
 - Maturity of processes handling CVE

Runtime

WAMR

- Bytecode Alliance project
- Written in C
- Interpreter or JIT / LLVM JIT
- Configurable options at compile time - AOT, bulk memory, SIMD, tail call, thread/shared memory, multi-module, reference types
- Low memory footprint

Wasmtime

- Bytecode Alliance project
- Written in Rust
- Based on Cranelift
- High memory footprint

Runtime

WasmEdge

- Written in C++
- LLVM JIT
- High memory footprint

V8

- Not yet supported in ATS Wasm plugin
- Written in C++
- Many dependencies / Complicated to get it to work

Notes on Performance

- Preliminary testing shows WAMR is the fastest
- Inconclusive
 - each runtime has many configuration options
 - Default may not be suitable for proxy-wasm
 - More tests needed

Summary

ATS Wasm Plugin

- Available now / Another option for extending ATS
- Language supported - C++, Rust, TinyGo, AssemblyScript, Zig
- Runtime supported - WAMR, wasmtime, WasmEdge

To Do

- Performance Testing/Improvement
 - Resource contention
 - Test runtimes with different configuration options
- Tooling support
 - Profiling with perf
 - Debugging with lldb
- Use Cases
 - AI Inference with WASI-nn
- Runtime Support
 - V8
- Looking forward / Future
 - Component Model

Reference

- ATS Plugin development - <https://docs.trafficserver.apache.org/en/latest/developer-guide/plugins/index.en.html>
- ATS header_rewrite plugin - https://docs.trafficserver.apache.org/en/latest/admin-guide/plugins/header_rewrite.en.html
- ATS Lua plugin - <https://docs.trafficserver.apache.org/en/latest/admin-guide/plugins/lua.en.html>
- ATS Wasm plugin - <https://docs.trafficserver.apache.org/en/latest/admin-guide/plugins/wasm.en.html>
- Proxy-wasm - <https://github.com/proxy-wasm>
- Proxy-wasm spec - <https://github.com/proxy-wasm/spec>
- Proxy-wasm Library - <https://github.com/proxy-wasm/proxy-wasm-cpp-host>
- Proxy-wasm C++ SDK - <https://github.com/proxy-wasm/proxy-wasm-cpp-sdk>
- Proxy-wasm Rust SDK - <https://github.com/proxy-wasm/proxy-wasm-rust-sdk>
- Rust example - <https://github.com/apache/trafficserver/tree/master/plugins/experimental/wasm/examples/rust>
- Coraza - <https://github.com/corazawaf/coraza>
- Coraza Proxy-wasm - <https://github.com/corazawaf/coraza-proxy-wasm>
- Coraza Proxy-wasm in ATS - <https://github.com/apache/trafficserver/tree/master/plugins/experimental/wasm/examples/tinygo>
- Wasi-nn - <https://github.com/WebAssembly/wasi-nn>
- WAMR - <https://github.com/bytedcodealliance/wasm-micro-runtime>
- Wasmtime - <https://github.com/bytedcodealliance/wasmtime>
- WasmEdge - <https://github.com/WasmEdge/WasmEdge>

Thank you!

yahoo!

QR Codes

Linkedin Profile



Presentation link

