Embedding Virtual Machines in ATS

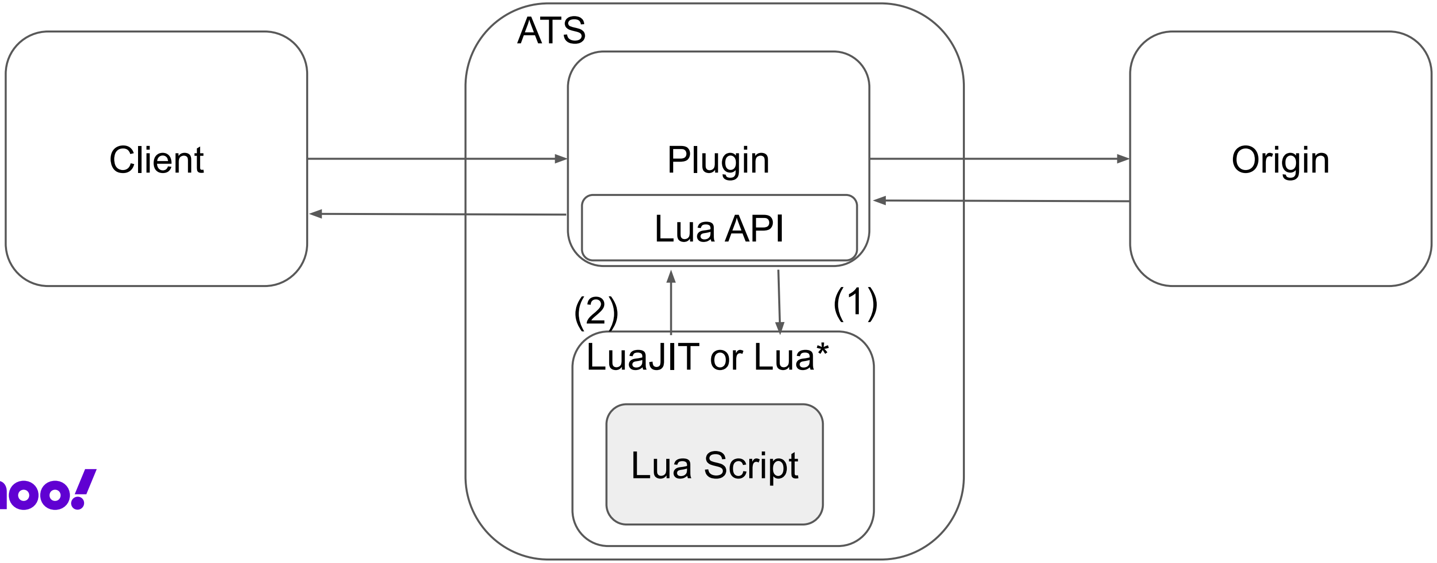
Shu Kit Chan
Yahoo
11/14/2023

Clarification on Definitions

- System Virtual Machine
  - Allows the running of a complete OS
  - E.g. VMWare, Docker, etc
  - Not what we want to talk about today!!!
- Process Virtual Machine
  - Application to provide programmable environment on a system
  - E.g. JVM, .NET framework
  - In this talk - LuaJIT or Wasm in ATS

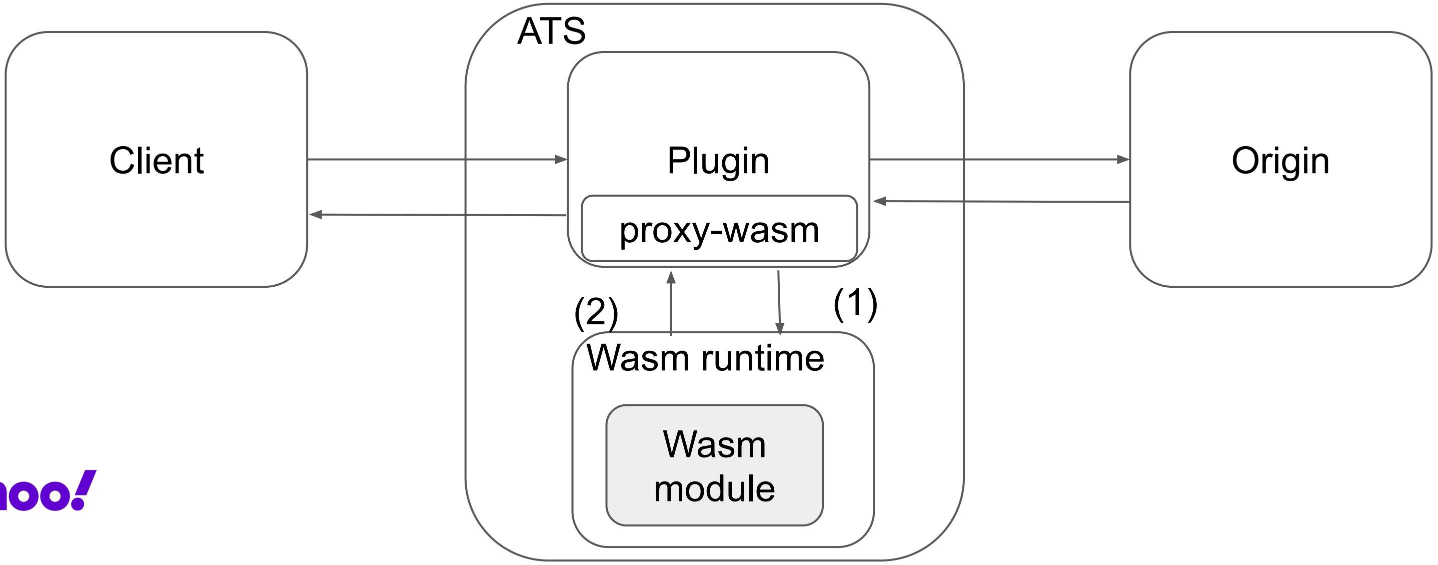
Architecture for Lua Plugin

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



Architecture for Wasm Plugin

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



Different Wasm Runtimes

WAMR

- Bytecode Alliance project
- Written in C
- Interpreter or JIT / LLVM JIT
- Configurable options at compile time
- Low memory footprint

Wasmtime

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

Different Wasm Runtimes

WasmEdge

- Written in C++
- LLVM JIT
- High memory footprint
- Lots of focus on AI Inference use cases

V8

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

Big Decision to Choose a Wasm Runtime

- 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 (extensions) supported by different runtime
- Trust in Security
  - Choice of implementation language
  - Maturity of processes handling CVE
- Performance !!!

yahoo!

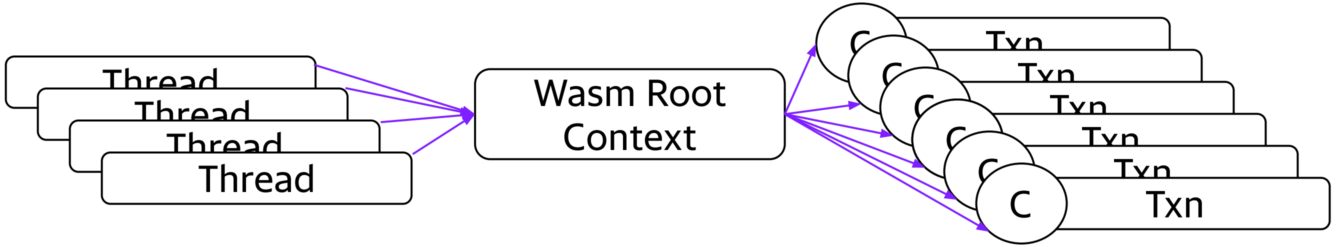
Quick Notes on Performance Testing

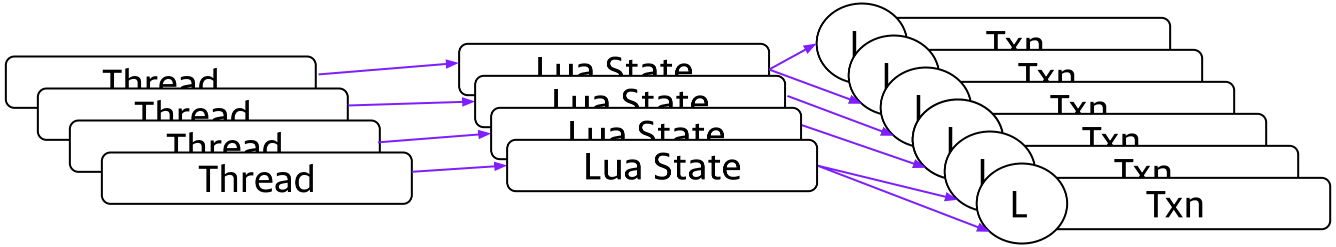
- Test: a small program to read a request header and add a request header
- Use h2load from a separate box

- 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

More Performance Testing

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





Downside of supporting Multiple VM instances

- No more single shared state. E.g.

```
local test = 0

function do_global_read_request()
  ts.debug(“test: “ .. test)
  test = 1
end
```

- Same thing will happen for Wasm plugin if we support Multiple Root Context

Downside of supporting Multiple VM instances

- More VM instance more memory usage
  - LuaJIT has a 2GB memory limit
- Wasm plugin will be similar
  - Worse with wasmtime/WasmEdge with high memory footprint

LuaJIT Memory Limitation

- What?
  - 2GB limit - per process, regardless of number of Lua VM
  - Only able use address values in the low 31 bit space for memory used by GC
  - "`PANIC: unprotected error in call to Lua API (not enough memory)`"
- GC64 mode
  - Since 2016
  - Memory usage can be a tag bit larger
  - No visible performance impact - experiments done by OpenResty group
- GC64 with ATS Lua Plugin
  - Thanks to Wikipedia team!
  - Significant mmap overhead - https://github.com/apache/trafficserver/issues/7423
  - Turning off JIT fixed the performance issue
  - Theory - LuaJIT wasted too much to do JIT repeatedly and unsuccessfully
  - More Investigation needed!

yahoo!

References

- OpenResty
  - LuaJIT GC64 mode - https://blog.openresty.com/en/luajit-gc64-mode/
- Apache APISIX
  - “Cloud Native API Gateway”
  - Built on top of Nginx/OpenResty
  - Programmable through LuaJIT and Wasm
  - Details of LuaJIT usage and comparison with Wasm - https://api7.ai/blog/apisix-chooses-luajit

Conclusion

- LuaJIT is awesome!!!
  - Memory limit gone
  - Performance
  - Proven
- But Wasm is still the future
  - Language support
  - Interoperability
  - Safety first / Sandboxed approach
  - Performance / Memory Usage still to catch up!

To Do - Wasm Plugin

- 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
- Future Changes on Wasm

yahoo!

To Do - Lua Plugin

- mmap issue with GC64 mode
- Adding support for TLS User Agent Hooks
- Any contributions/suggestions are welcome!