WASI certification tests

Current state and possible directions for the future

Current state

- WebAssembly/WASI: #9
  - github.com/caspervonb/wasi-test-suite
- Wasmtime: github.com/bytecodealliance/wasmtime/tree/main/crates/test-programs
- wasmer: github.com/wasmerio/wasi-tests
- WasmEdge: github.com/WasmEdge/wasi-test
- wasm3: github.com/wasm3/wasm3/tree/main/test/wasi
- WAMR: github.com/wasm-micro-runtime/wamr-test

Goals

- One place for all the tests (ideally under Bytecode Alliance umbrella)
- No toolchain dependency for test execution
- Minimal adoption effort
- Extensible test suite (e.g. nonstandard APIs)
- Different types of tests supported
- Ability to measure the coverage

Tests

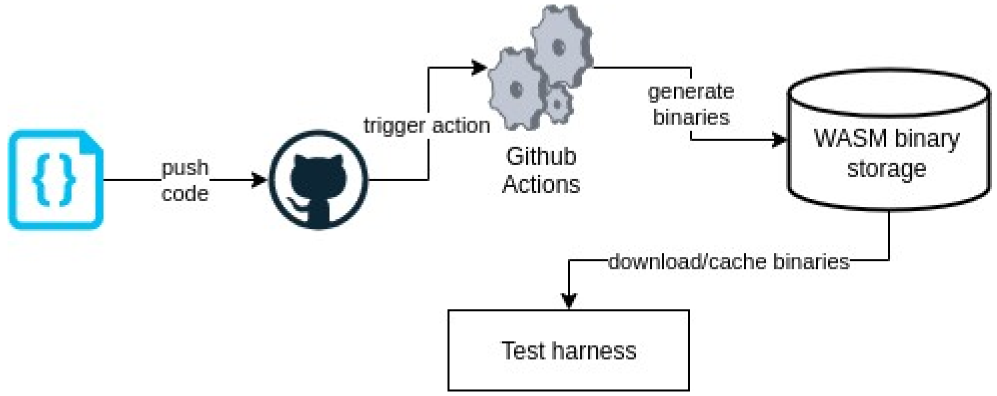
- WASM module per test
- Test suites
  - Core tests
    - No external dependencies (only WASI syscalls)
    - Written in one language (C/WAT/AssemblyScript/Rust?)
    - Unit-test like tests
  - Integration tests <- **Focus on those first**
    - Written in multiple languages
    - Can use standard libraries (e.g. libc) but it’s purpose is not to test them
      - Hide WASI snapshot preview versions
    - Real-life scenarios
  - Other types of tests (?):
    - Fuzz tests
    - Benchmarks

Git repositories

- Repository per test suite?
- Separate repository for test harness?
- Start simple:
  - Single repository for tooling and standard test suites
  - Separate repositories for non-standarized APIs

Precompiled binaries

- .wasm in git repository is simple, but
- repository can grow over time
- binaries are hard to review
  - Malicious code
  - Human errors

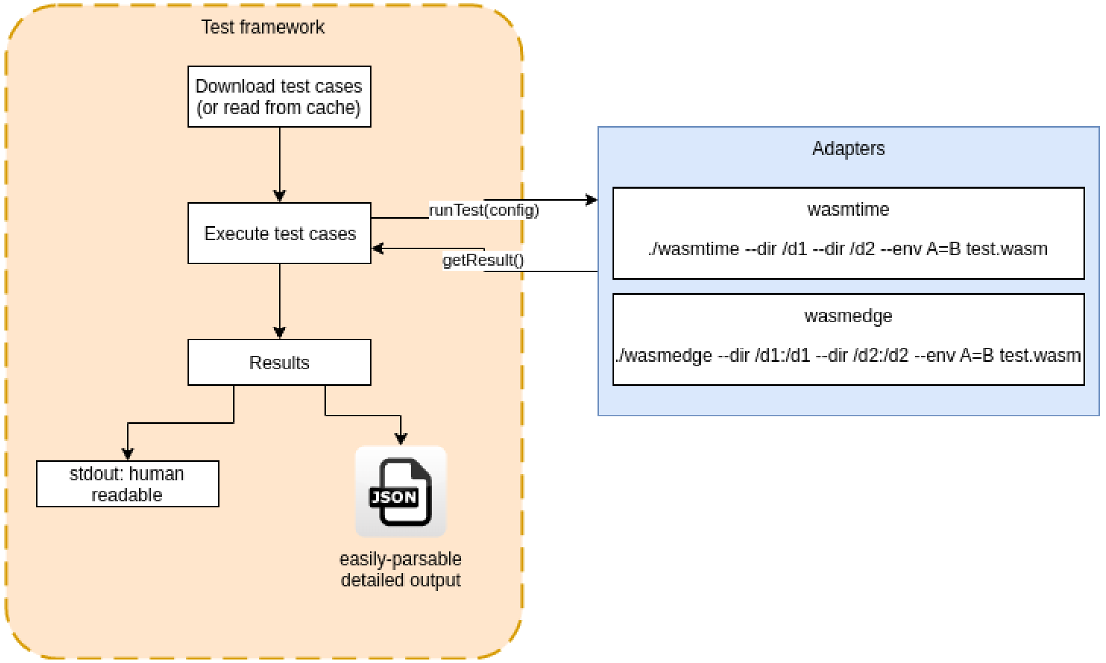


Test configuration

```
{
    "status": 0, // expected status code, defaults to 0
    "stdin": "string", // standard input passed to the test, defaults to empty string
    "stdout": "string", // expected standard output, by default not checked
    "stderr": "string", // expected standard error, by default not checked
    "env": [{"name": "value"}], // environment variables defined for a given test, defaults to empty list
    "arg": "string", // arguments passed to the program at execution, defaults to empty string
    "dir": ["string"], // list of pre-open directories, defaults to empty list
    "network": ["CDIR"], // a list of allowlisted IP address ranges, defaults to empty list
    "ns-lookup": ["domain"], // alist of allowlisted domains to lookup for, defaults to empty list
    "min-wasi-version": ["wasi_snapshot_preview1"], // a min version of WASI that te test is compatible with, defaults to first versions
    "max-wasi-version": ["wasi_snapshot_preview1"] // a max version of WASI that te test is compatible with, defaults to last versions
}
```

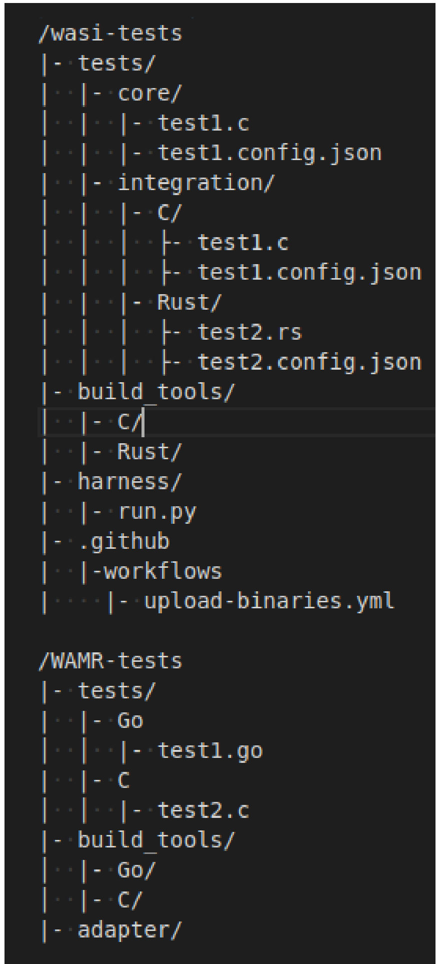
Test framework

- Runtime-owned adapters
- Reusable for different test suites
  - Tests for proposals
  - Runtime-specific APIs
- Python3 (?) for implementation
- Detailed JSON output (execution time, diff for non-matching outputs)
- Filtering
- Generate coverage


Test framework
Download test cases (or read from cache)
Execute test cases
runTest(config)
getResult()
Results
stdout: human readable
JSON
easily-parsable detailed output
Adapters
wasmtime
./wasmtime --dir /d1 --dir /d2 --env A=B test.wasm
wasmedge
./wasmedge --dir /d1:/d1 --dir /d2:/d2 --env A=B test.wasm

Proposed solution - summary

- Single repository in github.com/BytecodeAlliance
  - Test harness
  - Core & integration tests
- Runtime-agnostic test harness (each runtime provides the adapter)
- Precompiled .wasm binaries auto-generated by CI and stored outside of git repository
- Multiple test suites supported
- JSON test configuration and JSON output



```
/wasi-tests
|- tests/
|  |- core/
|  |  |- test1.c
|  |  |- test1.config.json
|  |- integration/
|  |  |- C/
|  |  |  ├- test1.c
|  |  |  ├- test1.config.json
|  |  |- Rust/
|  |  |  ├- test2.rs
|  |  |  ├- test2.config.json
|- build_tools/
|  |- C/
|  |- Rust/
|- harness/
|  |- run.py
|- .github
|  |-workflows
|     |- upload-binaries.yml

/WAMR-tests
|- tests/
|  |- Go
|  |  |- test1.go
|  |- C
|  |  |- test2.c
|- build_tools/
|  |- Go/
|  |- C/
|- adapter/
```

Questions? Suggestions?