Communication model of fuzzilli with V8

Source code

On a low level, Fuzzilli communicates with v8 through the REPRL protocol, implemented on the fuzzer side by the libreprl C library in `Sources/libreprl/`. The main way of using the library is through the following three functions:

`reprl_create_context()` this creates a new, empty REPRL context to be used by the following APIs.

`reprl_initialize_context(ctx, argv, envp)` this initializes the given context and sets the argv and envp vectors to use for the child processes.

`reprl_execute(ctx, code)` this executes the given code and returns the exit status. If necessary, a new child process is created for this. This involves creating pipes, forking itself, then setting filedescriptors, and using `execve` to execute the d8 binary. A child process can be reused for multiple executions, thus increasing fuzzing performance as the overhead of fork and execve are removed.

Coverage

Coverage information are being monitored through shared memory. On the side of v8 it is monitored through SanitizerCoverage module of Clang compiler (https://clang.llvm.org/docs/SanitizerCoverage.html) Through shared memory information about edges are shared with fuzzilli which implements counter for error and covered branches of the V8 code in Sources/libcoverage/coverage.c