libFuzzer Integration

Custom builds of the Swift toolchain (including development snapshots) have a built-in `libFuzzer` integration. In order to use it on a file `myfile.swift`, define an entry point fuzzing function with a `@_cdecl("LLVMFuzzerTestOneInput")` annotation:

```
@_cdecl("LLVMFuzzerTestOneInput")
public func test(_ start: UnsafeRawPointer, _ count: Int) -> CInt {
  let bytes = UnsafeRawBufferPointer(start: start, count: count)
  // TODO: Test the code using the provided bytes.
  return 0
}
```

To compile it, use the `-sanitize=fuzzer` flag to link `libFuzzer` and enable code coverage information; and the `-parse-as-library` flag to omit the `main` symbol, so that the fuzzer entry point can be used:

```
% swiftc -sanitize=fuzzer -parse-as-library myfile.swift
```

`libFuzzer` can be combined with other sanitizers:

```
% swiftc -sanitize=fuzzer,address -parse-as-library myfile.swift
```

Finally, launch the fuzzing process:

```
% ./myfile
```

Refer to the official `libFuzzer` documentation at https://llvm.org/docs/LibFuzzer.html#options for a description of the fuzzer's command line options.