Improve Logging in IPC

Proposal: 0012
Authors: Dr. Brandon Wiley, Operator Foundation
Status: Initial proposal `

Implementation: TBD

Introduction

This proposal seeks to improve the way that logging is handled in the IPC protocol by providing a new IPC message to carry logging information from the dispatcher to the application.

Motivation

Better logging is a requested feature from developers of both applications and transports. The IPC protocol used to connect applications to transports through the dispatcher offers only very basic error handling. Other proposals exist to improve error handling generally. This proposal is specifically about how to handle logging messages which could be useful for debugging failing transports. Currently, transport developers are free to do their own transport-specific logging inside the dispatcher state directory. However, when testing a transport with the dispatcher or inside of an application, it would be convenient to have the transports logs go into the application's logs. Having convenient and pervasive logging for transports will make debugging transport failure easier. In the case of applications not writing in the same language as the transport, getting logs from the transport to the application requires changes to the IPC protocol.

Proposed solution

A new IPC protocol message will be introduced to carry logging information. Additionally, a new command line option will be introduced to control IPC logging, as opposed to other logging that the transports might do either internally or using the dispatcher's logging facilities.

Design

1. IPC Log Message

A new IPC message will be introduced, in the form "LOG [loglevel] [message]". For instance, "LOG DEBUG This is an example debug log message."

The allowed values for the loglevel parameter are as follows:

- ERROR
- WARN
- INFO
- DEBUG

The format for the message parameter is a UTF-8 string ending with a newline.

2. New Command Line Flag for IPC Logging

The new command line flag is -ipcLogLevel [loglevel]. Example usage: “-ipcLogLevel DEBUG”.

The allowed values for the loglevel parameter are as follows:

- NONE
- ERROR
- WARN
- INFO
- DEBUG

The -ipcLogLevel flag controls what log messages are sent to the application using IPC LOG messages. Messages at the same log level or above will be sent. For instance, if the -ipcLogLevel is set to ERROR then only ERROR messages will be sent to the application, whereas if the -ipcLogLevel is set to INFO then ERROR, WARN, and INFO (but not DEBUG) messages will be sent to the application. This convention is consistent with how log levels are handled in other applications. The NONE loglevel is a special case which indicates that no log messages should be sent to the application.

In order to maintain backwards compatibility with older applications which may fail to properly handle unknown IPC message types, the default IPC loglevel if no loglevel is specified on the command line is NONE.

Effect on API Compatibility

These changes only affect the IPC protocol.

Effect on IPC Compatibility

These changes only add new capabilities and do not remove any capabilities or add any new requirements. Therefore, this proposal is backwards-compatible and requires only a minor version change. Please note that this proposal adds a new IPC message type and so old applications will only be backwards-compatible if they ignore unknown IPC messages. However,

compatibility with broken old applications can be ensured by simply turning off IPC log messages when using those applications. As the default is for IPC log messages to be disabled, the default case is backwards compatibility with these applications.

Alternatives considered

There is another proposal for adding better error handling to the IPC protocol, proposal “#0004 - EVENT client message”. That proposal adds new messages which provide connection success and failure, including error reporting with specific error types. While the use cases overlap between these two proposals, they have different purposes. That proposal is for improved error handling, whereas this proposal is for improved logging. Each approach to debugging failure has its place and they are complementary. The two proposals are also compatible in that each can be adopted separately and they can be used at the same time.