(This document is available as a Google Doc: https://goo.gl/BZdVhM)

Modularization of Specification

Proposal: 0002
Authors: Dr. Brandon Wiley
Status: Awaiting implementation

Implementation: TBD

Introduction

The modularization of the specification refers to breaking the overall specification up into separate parts, which will have their own version numbers. No source code changes will be required as this only changes how the specification is formatted and how the revision process is managed.

Motivation

The PT 1.0 specification only consisted of documentation for the IPC protocol. There was no standardized language-level modularity in the specification. In the PT 2.0 specification, both the IPC protocol and the new Go API were documented. After the drafting of the PT 2.0 specification, some PT implementers expressed interest exclusively in the IPC protocol, and some exclusively in the Go API, while some were interested in both. Implementers interested in just one of these sections in the specification expressed confusion at all of the additional documentation extraneous to their needs, as well as a greater cognitive load due to following all of the proposed changes to the PT specification not related to their section of interest.

Proposed solution

By splitting the PT specification into separate documents, implementers can only read the sections of interest. They can also engage in tracking of proposed changes only for sections of interest. This also allows for new sections to be added as additional separate documents. The PT specification as a whole will be considered to be the current set of documents for each section. This follows the POSIX model for standard development. For instance, the POSIX.1-2008 standard consists of four separate documents, each with their own version number.

Design

The PT specification will be split into the following documents:

- Base specification - an architectural overview of Pluggable Transports
- IPC protocol
- Go API

Additional documents will be added as new parts of the specification are created. In particular, additional language bindings may be specified.

The overall specification will be given a version number and defined as the set of versions for corresponding documents. For instance, and just as a hypothetical example, the PT 2.1 specification might be defined as follows:

- Base specification - v2.0
- IPC protocol - v2.0
- Go API - v2.1
- Swift API - v1.0

For an example of how this kind of multi-document standard is handled in other projects, look at the POSIX standard: https://en.wikipedia.org/wiki/POSIX

Effect on API Compatibility

This change only affects formatting of the specification documents and the process for revising them. There is no effect on API compatibility.

Effect on IPC Compatibility

This change only affects formatting of the specification documents and the process for revising them. There is no effect on IPC compatibility.

Alternatives considered

Another option that was discussed at the Pluggable Transport Implementers Meeting in Boston were making some sections likely to change in the future into appendices such that the main specification could remain unchanged while the appendices could be modified and more appendices added. However, this solution does not handle how version numbering would work for appendices.

There was some concern expressed with the current proposal that giving each section its own version number will be more confusing that having just one version number. However, the alternatives are either to update the overall version number any time one section is changed or to make changes but not update the version number. Each of these alternatives is more confusing to implementers as it does not map version numbers cleanly to changes of interest. In the new system, for instance, Go developers can only pay attention to the current version of the Go API. The overall specification version number therefore becomes a tracker more of the timeline of the overall revision effort instead of tracking small changes to each document.