BlueSSLService

Swift@IBM

Agenda

- What is BlueSSLService?
- How did we arrive at the design?
- Challenges we faced.

What is BlueSSLService?

- Cross-platform Swift based framework that provides an SSL/TLS add-in framework for Sockets using a delegate/protocol model.

- Currently directly supports BlueSocket however, it can be used with any Swift Socket framework that implements the protocol.

- Works on macOS using Secure Transport and on Linux using OpenSSL.

How did we arrive at the design?

- We already had an existing socket implementation (BlueSocket).

- We did not want to introduce a new dependency for that socket implementation based on the new package. It needed to be an “add-in” not a requirement.

- We wanted to minimize the platform dependencies, in other words, use what was available and not require additional downloads.

- Once attached to a Socket, it should be virtually transparent to the user of the Socket. All the heavy lifting should be done in the framework so the developer just has to configure it, assign it to a socket and forget about it.

Challenges we faced

- API
    - The APIs for Secure Transport and OpenSSL are significantly different. Finding a common ground was somewhat difficult.
- Configuration
    - OpenSSL provides more flexibility with regard to configuration.
    - Secure Transport is less flexible. At this point, the framework only supports certificates in PKCS#12 format on macOS.
- Cipher Suites
    - Secure Transport and OpenSSL handle cipher suites in an entirely different manner. In particular, the documentation for selecting cipher suites for Secure Transport is lacking.
    - As a result, we currently allow selection of cipher suites on the Linux platform only. For macOS, only the default set of ciphers is supported.

Conclusions
&
Recommendations

- Any solution should take advantage of the existing infrastructure provided by the platform. This will avoid having to be concerned with certifications (e.g. FIPS).

- None of the challenges were insurmountable:

    - API was unified, exposing an easy to use way to use the framework.

    - Cipher suites will be handled in a future release based on a potential solution that we're exploring.

    - Configuration, on macOS, is limited to the APIs exposed by Apple's Secure Transport.

- The delegate model allows for easy integration into existing socket packages.

- For more information, read the blog found at https://developer.ibm.com/swift/2016/12/12/securing-kitura-cross-platform-challenges