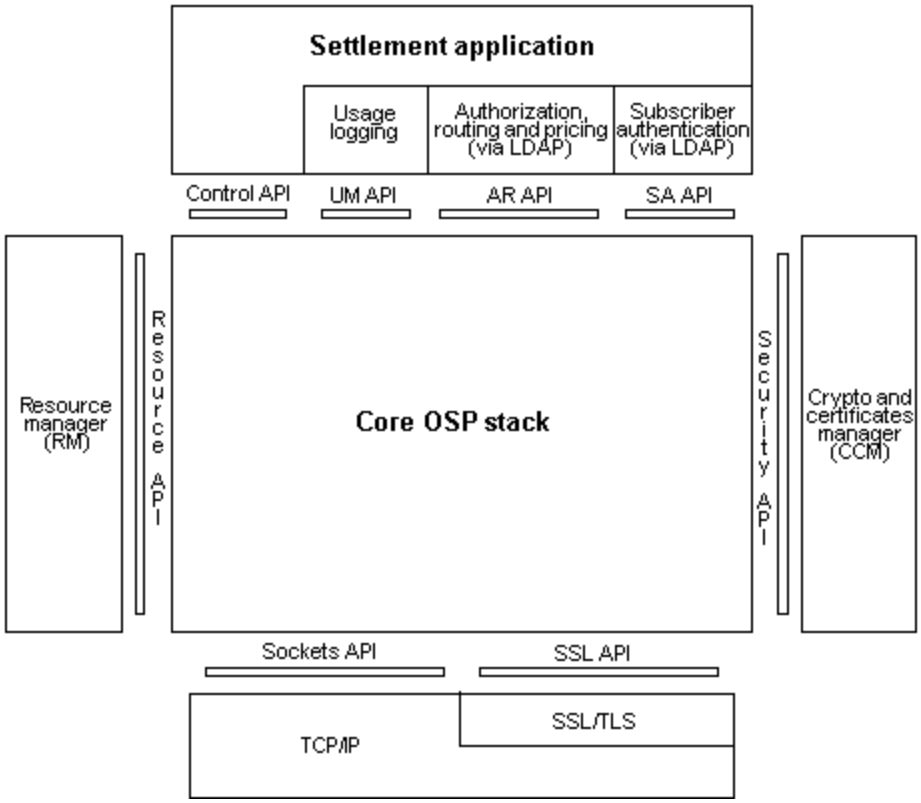
Summary of OpenOSP Interfaces

The following diagram illustrates the core OpenOSP server stack together with the application programming interfaces (APIs) and the sample implementations of the other components.



At the top of the diagram are the four Open Settlement Protocol (OSP) APIs, which provide access to the core OpenOSP server function from your settlement application.

- The **Control API** is used by the application to start and stop the OSP stack, to verify incoming client connections, and to gather statistical information about its operation.
- The **Usage Metering (UM) API** allows your application to collect and store usage information about each call that has been made using the server (for example the call duration and calling / called numbers), for use in billing.
- The **Authorization and Routing (AR) API** allows the OSP stack to request authorization and routing information from your application for a call. This API is also used to pass pricing and capabilities information (gathered from OSP clients) to your application, for use in routing and authorization decisions.
- The **Subscriber Authentication (SA) API** allows the OSP stack to request your application to authenticate subscribers and check whether they are entitled to use the facilities that OSP provides.

The information passed on these APIs is based on the contents of the messages between OSP servers and clients, OpenOSP breaks these messages out into individual fields and data structures for ease of use by the application.

Depending on your requirements, your settlement application may not need to use all of these APIs. For example, you may have multiple OSP servers each providing a subset of these functions, or you may have no requirement to support subscriber authentication.

The remaining APIs are used by OpenOSP to access utility functions.

- The **Security API** allows OpenOSP to manage signatures and certificates.
- The **Resource API** allows OpenOSP to make requests for operating system resources (memory and threads), and to free these resources when no longer required. The supplied Resource Manager code maps these requests to basic operating system functions; the API allows you to change this mapping to manage your own resource allocation if required. For example, you can pre-allocate resources, or allocate and free them in larger "chunks" rather than in response to individual requests.
- The **Secure Sockets Layer (SSL) API** provides access to OSP clients through SSL / TLS. It is a subset of the API defined by the OpenSSL open source SSL / TLS implementation, allowing you to use this implementation without change. If you intend to use a different Secure Socket Layer/Transport Layer Security (SSL / TLS) implementation with OpenOSP, the implementation must provide this interface.
- The **Sockets API** is the standard Berkeley Software Distribution (BSD) sockets interface as implemented on Sun SPARC Solaris, and provides access to OSP clients through Transmission Control Protocol (TCP). Refer to the Sun Solaris documentation for details of this API.

Note that OpenOSP also implements the Simple Certificate Enrollment Protocol (SCEP), but no additional API is required for its operation.