WIND RIVER

Wind River® SSL for VxWorks® 6

PROGRAMMER'S GUIDE

1.0

Corporate Headquarters
Wind River Systems, Inc.
500 Wind River Way
Alameda, CA 94501-1153
U.S.A.

toll free (U.S.): (800) 545-WIND
telephone: (510) 748-4100
facsimile: (510) 749-2010

For additional contact information, please visit the Wind River URL:

http://www.windriver.com

For information on how to contact Customer Support, please visit the following URL:

http://www.windriver.com/support

Wind River SSL for VxWorks 6 Programmer's Guide, 1.0

27 May 05
Part #: DOC-15481-ZD-00

Contents

1 Overview 1

1.1 Introduction 1

1.1.1 Technology Overview 1

1.1.2 Component Overview 2

1.1.3 Additional Documentation 3

Wind River Documentation 3
OpenSSL Documentation 4
Other Documentation 4

1.2 Configuration 5

1.3 Latest Release Information 5

2 Configuring VxWorks for Wind River SSL 7

2.1 Introduction 7

2.1.1 Including Wind River SSL in a VxWorks Image Project 8

2.2 Configuration Using Wind River Workbench 8

2.2.1 Including Wind River SSL 9

Locating Components with the Find Object Tool 9

2.2.2 Including Component Dependencies 10

2.2.3 Including Application Code 10
2.2.4 Setting Configuration Parameters 10

2.3 Configuration Using the Command Line 10
2.3.1 Configuring VxWorks for Wind River SSL 11
2.3.2 Managing Dependencies 12
2.3.3 Initializing Wind River SSL 12
2.3.4 Setting Configuration Parameters 12
2.3.5 Rebuilding VxWorks 12

3 Technical Background 13

3.1 Introduction 13

3.2 Description 14
3.2.1 OpenSSL 15
3.2.2 SSL Protocols 15
3.2.3 SSL Handshake Sequence 16
Cipher Suite Negotiation Step 16
Key Exchange Step 16
Authentication Step 17
3.2.4 Application Data Transmission 17
3.2.5 Cryptography 18
Public Key Cryptography 18
Symmetric Cryptography 18
Message Digests 18
3.2.6 Digital Signatures 19
3.2.7 Digital Certificates 19
SSL Certificates 19
Certificate Management Tasks 20

3.3 Implementation 20
3.3.1 Cryptography 20

3.3.2 Digital Certificates 21
3.3.3 Utilities 21

A Runtime Configuration Summary 23

A.1 Introduction 23

A.2 Locating This Component 24

A.3 Configuration Summary 24

A.3.1 Initialization Routine 24
A.3.2 Configlette 24
A.3.3 Parameters 24

A.4 Libraries 25

A.5 Reference Documentation 25

A.6 Dependencies 25

A Glossary 27

A.1 Terms 27

A.2 Abbreviations and Acronyms 27

1 Overview

1.1 Introduction 1
1.2 Configuration 5
1.3 Latest Release Information 5

1.1 Introduction

Secure Sockets Layer (SSL) enables TCP-based network applications to exchange data securely. Wind River SSL provides native SSL and Transport Layer Security (TLS) services to Wind River Platforms components.

NOTE: Throughout this guide, the term *SSL* refers to both SSL and TLS.

1.1.1 Technology Overview

Wind River SSL implements a subset of OpenSSL, the open source SSL development toolkit available at **http://www.openssl.org**.

By implementing OpenSSL, Wind River SSL implements the SSL v2, SSL v3, and TLS v1 protocols, as specified in the Internet Draft *The SSL Protocol Version 3.0*, and in RFC 2246: *The TLS Protocol Version 1.0*.

NOTE: This release of Wind River SSL includes OpenSSL version 0.9.7.e. Do not replace the OpenSSL code with any other version.

1.1.2 Component Overview

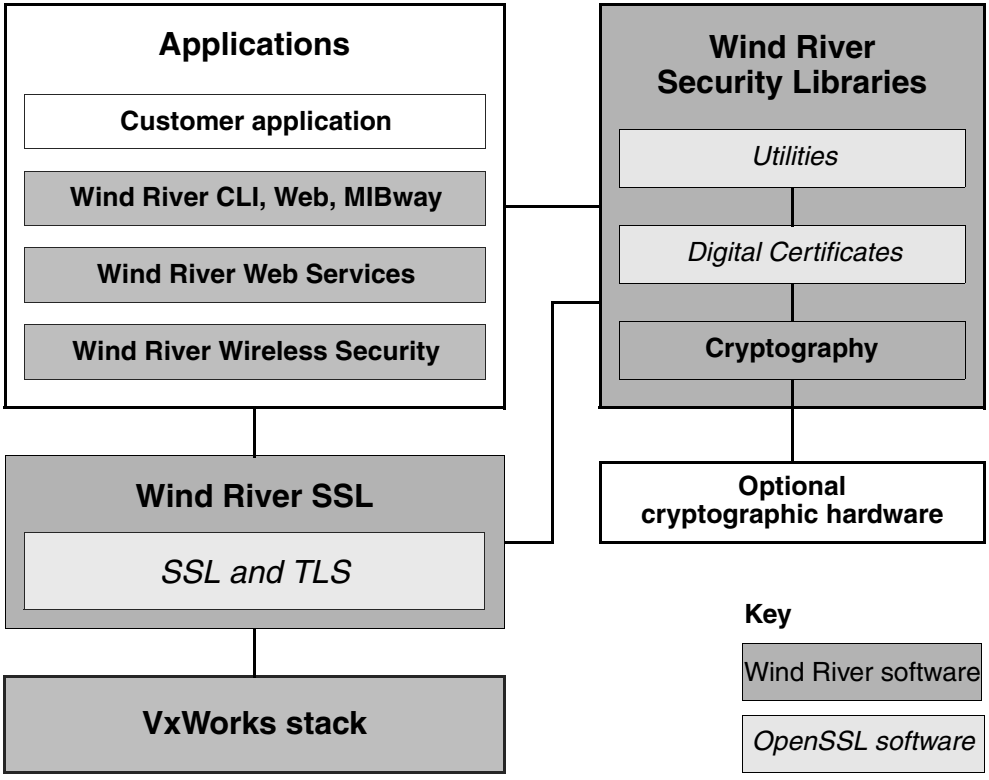
Wind River Platforms components implement OpenSSL functionality as follows:

- Certain functionality is implemented with modifications. The relevant code is part of Wind River Security Libraries. The APIs of this code are compatible with their equivalents in OpenSSL. Applications written for OpenSSL can be ported to Wind River SSL provided the implementation differences are taken into account. For details, see the *Wind River Security Libraries for VxWorks 6 Programmer's Guide*.
- The remaining functionality is unchanged. The relevant code is part of Wind River SSL. The APIs of this code are compatible with their equivalents in OpenSSL.

To use the security services of Wind River SSL in your application, you must code calls to API routines in Wind River SSL and Wind River Security Libraries.

Figure 1-1 summarizes the relationships of these components.

Figure 1-1 **Component Relationships**



1.1.3 Additional Documentation

This section lists additional documentation of interest to readers of this guide.

Wind River Documentation

For information on how Wind River implements OpenSSL's X>509 digital certificate functionality, and on Wind River's cryptography library, see the *Wind River Security Libraries for VxWorks 6 Programmer's Guide*.

OpenSSL Documentation

The focus of this guide is the configuration of the Wind River SSL component of the Wind River Platforms products. Although this guide contains relevant background information, it is beyond the scope of the guide to provide a complete description of OpenSSL and how to write OpenSSL applications. For any OpenSSL topic not covered in this guide, see the following book and web site:

- Viega, J., M. Messier, and P. Chandra, *Network Security with OpenSSL*, O'Reilly Media, Inc., 2002, ISBN 0-596-00270-X
  http://www.oreilly.com
- The OpenSSL Project
  http://www.openssl.org

For OpenSSL API reference documentation, including API references for routines that Wind River has implemented, see the OpenSSL Project.

NOTE: Wind River does not accept responsibility for the contents of non-Wind River sites.

Other Documentation

For a full description of the SSL and TLS protocols, consider the following Internet Engineering Task Force (IETF) papers and other sources:

- Chown, P., *AdvancedEncryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)*, RFC 3268, June 2002.
  http://www.ietf.org/rfc/rfc3602.txt
- Dierks, T., and C. Allen, *The TLS Protocol Version 1.0,* RFC 2246, January 1999.
  http://www.ietf.org/rfc/rfc2246.txt
- Freier, A., P. Karlton, and P. Kocher, *The SSL Protocol Version 3.0,* Internet Draft, March 1996.
  http://wp.netscape.com/eng/ssl3/ssl-toc.html
- Rescorla, Eric, *SSL and TLS: Designing and Building Secure Systems*, Addison-Wesley, 2001, ISBN 0-201-61598-3.
- *SSL 2.0 Protocol Specification*, November 1994.
  http://wp.netscape.com/eng/security/SSL_2.html

1.2 Configuration

Some components shipped with your Platform are provided as source code, which you must compile before you can use the component in a VxWorks image project. For source compile instructions, see *Wind River Platforms Getting Started*.

The configuration section of *Wind River Platforms Getting Started* contains instructions for a default or basic configuration of Wind River SSL. This guide contains a more thorough discussion of link-time and runtime configuration options in *2. Configuring VxWorks for Wind River SSL*.

1.3 Latest Release Information

The latest information on this release can be found in the *Wind River Platforms Release Notes*. Release notes are shipped with your Platform product and are also available from the Wind River Online Support site:

http://www.windriver.com/support/

In addition, this site includes links to topics such as known problems, fixed problems, documentation, and patches.

NOTE: Wind River strongly recommends that you visit the Online Support site before installing or using this component. The Online Support site may include important software patches or other critical information regarding this release.

2

Configuring VxWorks for Wind River SSL

2.1 Introduction 7

2.2 Configuration Using Wind River Workbench 8

2.3 Configuration Using the Command Line 10

2.1 Introduction

To configure Wind River SSL to run as a VxWorks kernel component, you must input configuration data in as many as three distinct places while building the VxWorks image.

- First, you may need to specify certain options when compiling the component source. For more information, see *Wind River Platforms Getting Started*
- Second, you need to explicitly include the component and its dependencies in your customized VxWorks image project.
- Third, you may need to write code to initialize the component at runtime.

NOTE: Before you can configure VxWorks for Wind River SSL, you must configure Wind River Security Libraries as described in the *Wind River Security Libraries for VxWorks 6 Programmer's Guide*.

2.1.1 Including Wind River SSL in a VxWorks Image Project

You can configure your VxWorks image project to include Wind River SSL in two ways: you can use Wind River Workbench to include components in the VxWorks image, or you can manually edit your BSP configuration files to include or exclude components, then build VxWorks from the command line.

When choosing a method for configuring and building Wind River SSL, consider the following criteria:

- Configuring and building using Wind River Workbench is an easy and accurate way to add needed components and ensure that other required dependencies are included. For instructions, see *2.2 Configuration Using Wind River Workbench*, p.8.
- Configuring and building from the command line involves editing text files that contain lists of dependencies and configuration parameters, and calling the **make** utility to build a system image. This process afford you the ability to automate the configuration and build, but it requires that you manage component dependencies. For instructions, see *2.3 Configuration Using the Command Line*, p.10.

NOTE: For detailed information about the use of Wind River Workbench, see the *Wind River Workbench User's Guide*. For information about the use of VxWorks facilities, target-resident tools, and optional components, see the *VxWorks Programmer's Guide*.

2.2 Configuration Using Wind River Workbench

This section describes how to use Wind River Workbench to perform the following tasks:

- Include Wind River SSL, its component dependencies, and application code in the VxWorks image.
- Set parameters that determine the runtime behavior of Wind River SSL.

2.2.1 Including Wind River SSL

To include Wind River SSL in your VxWorks image project:

1. Launch Workbench and open the workspace that contains your VxWorks image project.
1. Expand the project and right-click **Kernel Configuration**.
2. In the **Kernel Editor**, select **Components**.
3. Expand the component folders as follows:

   Network Components > Network Security Components > SSL/TLS
4. Include the required components. At a minimum you must include:

   - **SSL/TLS**

   You can optionally include:

   - **SSL/TLS Applications**

   Once you have located the component, right-click the component name and select **Include** *componentName* to include it in your customized VxWorks image. Workbench automatically calculates dependencies and prompts you to accept the inclusion of dependent components.

Locating Components with the Find Object Tool

The structure of the VxWorks component tree is subject to change. Updates or patches can change the location of components in the tree. If you cannot find an object, use the Find Object tool to locate it in the tree.

To find a component using the Find Object tool:

1. Right-click the project and select **Edit**.
2. In the **Kernel Editor**, select **Components**.
3. In the **Kernel Editor**, right-click a component tree node and select **Find Object**.
4. In the **Find Object** dialog, select **Component** from the **Type** list.
5. From the **Object** list, select the Wind River SSL components you want to include in the VxWorks image, then click **Find**.

   For the main SSL functionality, find:

   - **INCLUDE_SSL**

For the optional SSL Applications, find:

- **INCLUDE_SSL_APPS**

When Workbench locates the component, it highlights the component name in the tree.

6. Once **Find Object** has located the component, right-click the component name and select **Include** *componentName* to include it in the VxWorks image project.

2.2.2 Including Component Dependencies

Workbench automatically includes the dependencies listed in *A. Runtime Configuration Summary*.

2.2.3 Including Application Code

For instructions on including application code in the VxWorks image, see the *Wind River Workbench User's Guide*.

2.2.4 Setting Configuration Parameters

Wind River SSL has no configuration parameters.

2.3 Configuration Using the Command Line

The configuration procedure described in this section creates a VxWorks image that includes Wind River SSL, optional application code, and component dependencies.

NOTE: Before configuring Wind River SSL, you must configure Wind River Security Libraries as described in the *Wind River Security Libraries for VxWorks 6 Programmer's Guide*.

2.3.1 Configuring VxWorks for Wind River SSL

The steps below include Wind River SSL and optional application code.

1. Open a command window or prompt.
2. Change directory to *installDir***/vxworks-6.1/target/config/all**.
3. Edit **usrConfig.c**. In the **usrRoot( )** routine, define **INCLUDE_USER_APPL**.
4. In the **INCLUDE_USER_APPL** block, add a call to **sslInit( )** immediately after the call to **secLibInit( )**, as shown below. (For more information on the **secLibInit( )** call, see the *Wind River Security Libraries for VxWorks 6 Programmer's Guide*.)

```
#define INCLUDE_USER_APPL
#ifdef INCLUDE_USER_APPL \
   secLibInit(); \
   sslInit(); \
   USER_APPL_INIT
#endif
```

5. Define **INCLUDE_SSL_APPS** if you want to include the SSL command line applications.
6. Save **usrConfig.c**.
7. Change directory to *installDir***/vxworks-6.1/target/config/***bspName*, where *bspName* is your target type.
8. Copy *installDir***/vxworks-6.1/target/src/ssl/examples/sslInit.c** to the current directory.
9. Edit **config.h** and add the directives below to include network components. Depending on your BSP, these directives may already appear in **config.h**.

```
#define STANDALONE_NET
#define INCLUDE_NETWORK
#define INCLUDE_NET_INIT
```

10. If you want to include application code, for example, the object file **myApp.o** containing the main routine **myAppInit( )**, follow these steps:
    a. Copy **myApp.o** from your development location to the current directory.
    b. In **config.h**, the macro **USER_APPL_INIT** must be a valid C statement or block. This code will be included in the **usrRoot( )** routine if you have defined **INCLUDE_USER_APPL**. The code shown below is only an example; you must change it as needed. Place your code after any other code already present in the macro. If the **USER_APPL_INIT** section is not

already present in **config.h**, you can place it anywhere in the file. (The use of **taskSpawn( )** is recommended over direct execution of a user routine.)

```
#define USER_APPL_INIT \

{ \
   IMPORT int myAppInit(); \
   taskSpawn ("myApp", 30, 0, 5120, myAppInit, 0x1, 0x2, 0x3,
      0,0,0,0,0,0,0); \
}
```

c. Add additional directives to include any other functionality required by your application.

11. Save **config.h**.
12. Edit the **Makefile**, search for the **MACH_EXTRA** line, add additional objects as shown below, and save the file.

```
MACH_EXTRA += sslInit.o myApp.o
```

2.3.2 Managing Dependencies

The dependencies listed in *A. Runtime Configuration Summary* must be defined at link time.

2.3.3 Initializing Wind River SSL

To initialize Wind River SSL when the target boots, call **sslInit( )** as shown in *2.3.1 Configuring VxWorks for Wind River SSL*, p.11.

2.3.4 Setting Configuration Parameters

Wind River SSL has no configuration parameters.

2.3.5 Rebuilding VxWorks

Rebuild the VxWorks image as described in the *VxWorks Programmer's Guide*.

3 Technical Background

3.1 Introduction 13
3.2 Description 14
3.3 Implementation 20

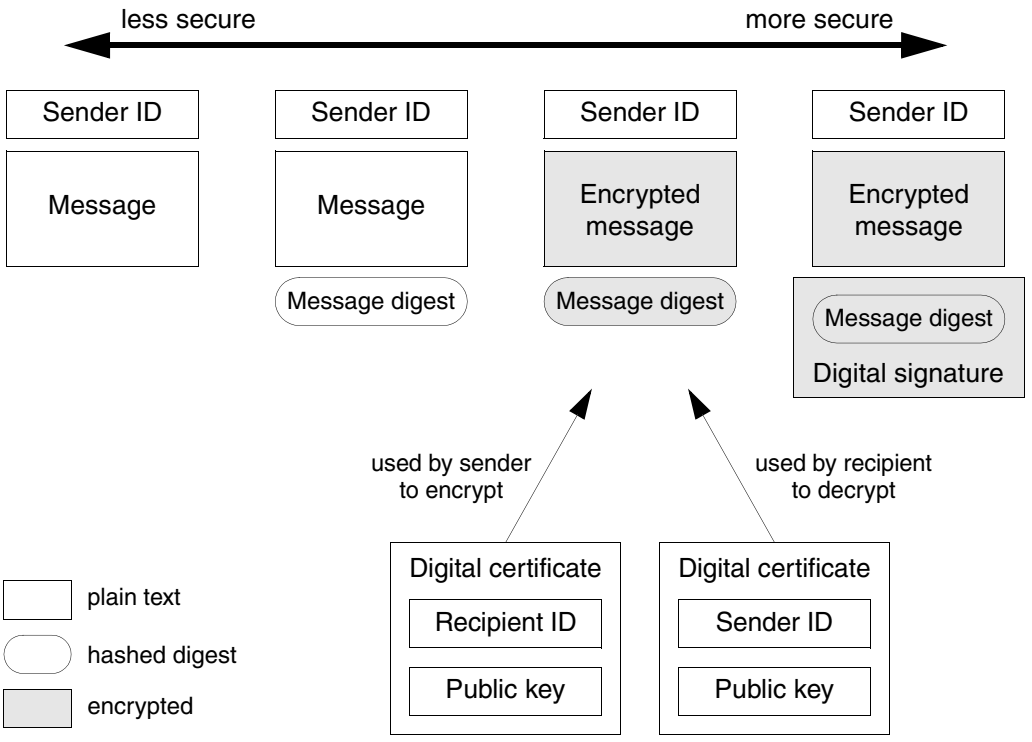
3.1 Introduction

This chapter gives brief descriptions of the technologies underlying Wind River SSL and, where applicable, descriptions of how Wind River has implemented these technologies.

The SSL protocols use cryptographic techniques such as public key cryptography, symmetric key cryptography, and hashed digests to guarantee data privacy. They use digital signatures to guarantee data integrity, and they use digital certificates to validate public keys.

Figure 3-1 gives a general summary, not limited to SSL, of how these technologies participate in data protection.

Figure 3-1 **Forms of Data Protection**



3.2 Description

This section gives a basic introduction to OpenSSL, the SSL protocols, and how SSL uses cryptography and digital certificates.

For detailed information on these and other OpenSSL topics, see the book *Network Security with OpenSSL*, listed under *Other Documentation*, p.4, and the OpenSSL Project Web site at **http://www.openssl.org**.

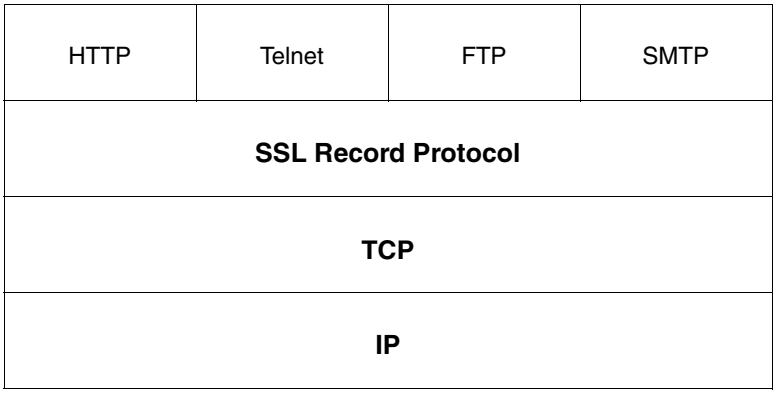
3.2.1 OpenSSL

The OpenSSL toolkit supports SSL 2.0, SSL 3.0, and TLS 1.0. It includes public key cryptography algorithms, symmetric key cryptography algorithms, message digest functions, and certificate management routines.

3.2.2 SSL Protocols

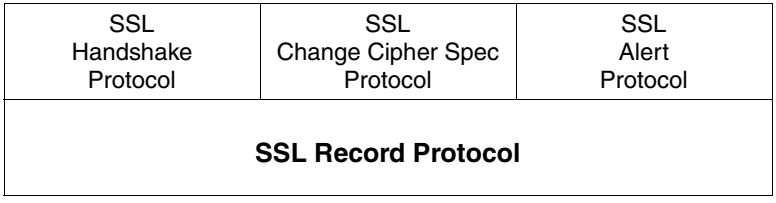
The SSL Record Protocol can add data security to any protocol that uses TCP/IP, as shown in Figure 3-2.

Figure 3-2 **SSL Protocol Stack**



The SSL Record Protocol encapsulates three separate control protocols as shown in Figure 3-3.

Figure 3-3 **SSL Handshake Protocol Stack**



The SSL Handshake Protocol performs the handshake sequence that sets up the SSL session. The SSL Change Cipher Spec Protocol updates the cipher suite to be used during the session. The SSL Alert Protocol conveys SSL-related alert messages.

3.2.3 SSL Handshake Sequence

When a client requests an SSL session, the server and client set up the session by means of a handshake sequence performed by the SSL Handshake Protocol. This section describes the steps of the handshake sequence.

Cipher Suite Negotiation Step

In the cipher suite negotiation step, the client and server establish the cipher suite to be used during the SSL session. A cipher suite consists of a key exchange method and a cipher spec.

Key Exchange Method

This part of the cipher suite determines how the shared key for symmetric cryptography will be agreed upon. The choice of key exchange method includes the choice of whether or not to use digital signatures during the key exchange.

Cipher Spec

A cipher spec includes the following items:

- Message digest algorithm to be used for signing record units during the key exchange. Signing the record units protects against a man-in-the-middle attack during the generation of the shared key.
- Cipher algorithm to be used for symmetric cryptography during the transmission of application data. SSL is designed to allow a variety of algorithms to be used.

Key Exchange Step

In the key exchange step, the server and client exchange the shared key to be used for symmetric cryptography. The most common form of key exchange in SSL uses the RSA algorithm.

Authentication Step

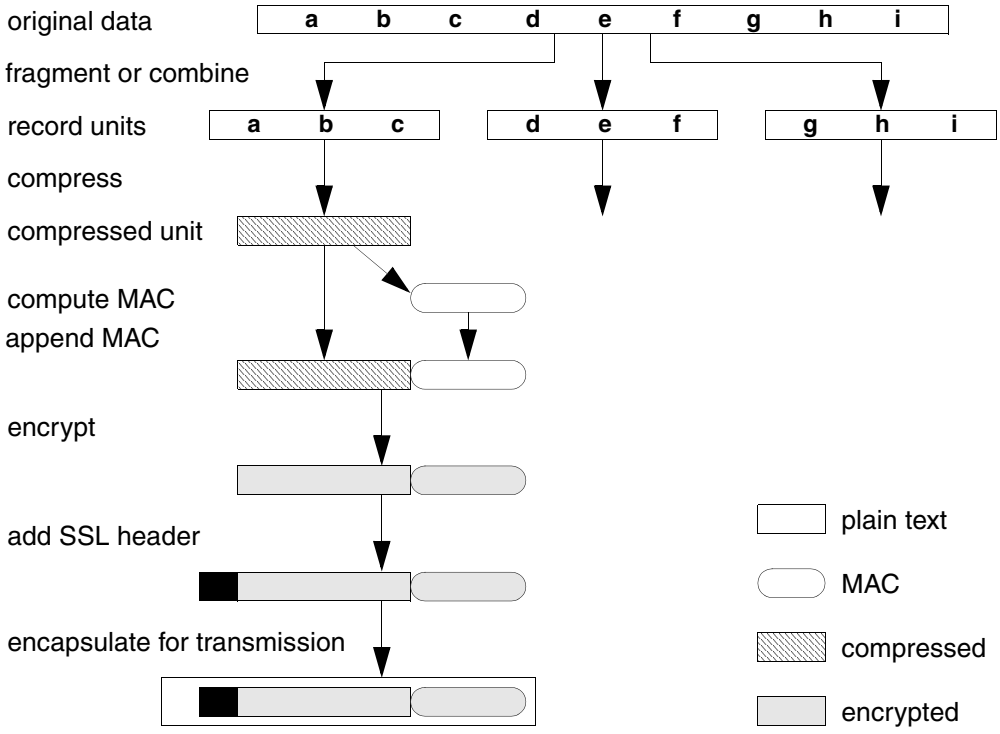
In the authentication step, the client authenticates the server, and the server optionally authenticates the client. The most common authentication method consists of the exchange of SSL certificates.

3.2.4 Application Data Transmission

The SSL Record Protocol transmits SSL control data and application data between the client and server in the form of SSL record units.

The protocol starts by fragmenting or combining the data into fixed-length units. In a typical scenario, it compresses these units, appends a MAC, and encrypts them, before transmitting them using the underlying reliable transport protocol. See Figure 3-4.

Figure 3-4 **SSL Record Unit Assembly**



3.2.5 Cryptography

SSL uses cryptographic techniques as described below.

Public Key Cryptography

Most types of public key cryptography use a pair of related keys: a private key and a public key. The key owner (a specific person, computer, or organization) keeps the private key secret, but distributes the public key to all parties who may want to communicate securely with the owner. Public keys must be distributed in such a way that there is no doubt about who owns them. Distributing a key in a digital certificate, issued by a recognized CA, ensures its trustworthiness.

An SSL client authenticates the server's SSL certificate using the public key of the CA that issued the certificate. This key is usually coded in the SSL implementation. The server may also authenticate the client's certificate.

Clients and servers encrypt and decrypt data during the SSL handshake sequence. The client obtains the server's public key from the server's SSL certificate.

Symmetric Cryptography

In symmetric cryptography, the parties agree on a single shared key before starting the secure communication.

SSL uses symmetric cryptography to guarantee data confidentiality. The SSL Record Protocol encrypts each record unit using a shared secret key defined by the SSL Handshake Protocol.

Message Digests

A message digest, or one-way hash, is a short, fixed-length representation of a longer, variable-length message. Digest algorithms produce a unique digest for every message; it is unfeasible to create two different messages that have the same digest. It is also unfeasible to determine the message from the digest.

SSL uses message digests to guarantee data integrity. The SSL Record Protocol signs each record unit with a message authentication code (MAC) containing a digest of the unit's data.

3.2.6 Digital Signatures

A digital signature consists of a hashed digest of a message, which has then been encrypted using the sender's private key.

When a sender transmits information in a signed message, the recipient can use the signature to determine whether the message really came from the apparent sender. The recipient decrypts the signature, using the sender's public key, and compares the result with the plain text version of the message. A match guarantees that the message came from the apparent sender (assuming that the sender is the true owner of the key pair).

3.2.7 Digital Certificates

Parties in a transaction can establish trust in each other by exchanging digital certificates. A certificate binds a party's public key to the party's identity. In order to be trusted, the certificate must itself be signed by a trusted Certification Authority (CA) or by a member of a chain of trust that ends at such an authority.

A public key certificate contains the following information:

- identity of the subject (key owner): can be a full X.509 distinguished name or certain parts as determined by the CA
- validity period of the certificate: a not-before date and a not-after date
- subject's public key
- identity of the issuer (CA): can be a full X.509 distinguished name or certain parts of a distinguished name, as determined by the issuer
- issuer's digital signature: a hashed digest of the certificate's contents, encrypted with the issuer's private key
- administrative information used by the CA, such as a serial number
- other application-specific information

SSL Certificates

The SSL protocol requires an SSL server to have an SSL certificate installed. An SSL certificate is a special kind of certificate containing the server's identity and the server's public key, all signed with a digital signature.

At the start of an SSL session, the server sends the client a certificate chain, starting with the server's own SSL certificate and ending with a CA's root certificate. In order to authenticate the server and trust the server's certificate, the client must check all the signatures and validity periods in the chain, and also determine whether it trusts the root CA.

The client then uses the server's public key to encrypt handshake messages and application data.

Certificate Management Tasks

Certificate management tasks include generating key pairs, obtaining SSL certificates from a CA, and downloading certificates to the target machine.

3.3 Implementation

Wind River Platforms components implement OpenSSL functionality as follows:

- Certain functionality is implemented with modifications. The relevant code is part of Wind River Security Libraries. The APIs of this code are compatible with their equivalents in OpenSSL. Applications written for OpenSSL can be ported to Wind River SSL provided the implementation differences are taken into account. For details, see the *Wind River Security Libraries for VxWorks 6 Programmer's Guide*.
- The remaining functionality is unchanged. The relevant code is part of Wind River SSL. The APIs of this code are compatible with their equivalents in OpenSSL.

For detailed information on OpenSSL itself, see the book *Network Security with OpenSSL*, listed under *Other Documentation*, p.4, and the OpenSSL Project Web site at **http://www.openssl.org**.

3.3.1 Cryptography

The Cryptography library replaces OpenSSL's ENGINE API with Wind River's Common Cryptographic Interface (CCI). Several alternative cryptographic

providers are available. CCI contains a software-based provider (the default provider), providers for specific hardware cryptographic engines, and a programming API with which you can write your own provider.

The following OpenSSL ciphers are not available in the Cryptography library: KRB5, CAST, IDEA, MDC2, RC2, BF, DESX, and Elliptic Curve.

The Cryptography library includes an implementation of OpenSSL's EVP (Digital Envelope) routines, modified so as to interface with CCI. The API of Wind River's EVP is identical to that of OpenSSL's EVP. Existing OpenSSL applications that call EVP do not need to be changed. However, if you have an application that calls OpenSSL's low-level cryptography routines, you must convert it to use EVP.

For more information on the Cryptography library and cryptographic providers, see the *Wind River Security Libraries for VxWorks 6 Programmer's Guide*.

3.3.2 Digital Certificates

The Digital Certificates library contains support routines for OpenSSL's X.509 certificate functionality.

3.3.3 Utilities

The Utilities library contains OpenSSL utility functions such as BIO and STACK.

A Runtime Configuration Summary

A.1 Introduction 23

A.2 Locating This Component 24

A.3 Configuration Summary 24

A.4 Libraries 25

A.5 Reference Documentation 25

A.6 Dependencies 25

A.1 Introduction

This section provides high-level information about the software elements of Wind River SSL as it pertains to VxWorks. It includes information such as dependencies, component initialization routines, libraries, and, where applicable, VxWorks component locations and component include macros.

NOTE: Before you can configure VxWorks for Wind River SSL, you must configure Wind River Security Libraries as described in the *Wind River Security Libraries for VxWorks 6 Programmer's Guide*.

A.2 Locating This Component

Component Name

SSL/TLS

VxWorks Component Names

SSL/TLS

SSL/TLS Applications

VxWorks Component Tree Location

Network Components > Network Security Components > SSL/TLS

Include Macro Names

#define INCLUDE_SSL

#define INCLUDE_SSL_APPS

A.3 Configuration Summary

A.3.1 Initialization Routine

installDir/**vxworks-6.1/target/src/ssl/examples/sslInit.c**

A.3.2 Configlette

installDir/**vxworks-6.1/target/config/comps/src/net/usrNetSSL.c**

A.3.3 Parameters

N/A

A.4 Libraries

libopenssl.a

A.5 Reference Documentation

For OpenSSL reference documentation, see **http://www.openssl.org**.

NOTE: Wind River does not accept responsibility for the contents of non-Wind River sites.

A.6 Dependencies

Application Interface

Default (Software)

CIPHER-AES - Advanced Encryption Standard

CIPHER-DES/3DES/DESX - Data Encryption Encryption

CIPHER-NULL - Encryption

CIPHER-RC4 - RC4 Encryption

CIPHER-RC4TKIP - Wireless Temporal Key Integrity Protocol

HASH-CRC32 - Authentication

HASH-MD2 - Authentication

HASH-MD4 - Authentication

HASH-MD5 - Authentication

HASH-RIP160 - Authentication

HASH-RIP128 - Authentication

HASH-SHA1 - Authentication

HASH-SHA256 - Authentication

HASH-SHA384 - Authentication

HASH-SHA512 - Authentication

HMAC-AESXCBC - Authentication using key

HMAC-MD4 - Authentication using key

HMAC-MD5 - Authentication using key

HMAC-RIP160 - Authentication using key

HMAC-RIP128 - Authentication using key

HMAC-SHA1 - Authentication using key

HMAC-SHA256 - Authentication using key

HMAC-SHA384 - Authentication using key

HMAC-SHA512 - Authentication using key

INTEGER - Precision Math Operations

KEYWRAP - AES - Keywrap Algorithm

PKI - RSA - Public Key Encryption

PRNG - Pseudo Random Number Generation

X.509 Certificate Support

A Glossary

A.1 Terms 27

A.2 Abbreviations and Acronyms 27

A.1 Terms

Certification Authority

A trusted organization responsible for issuing, revoking, renewing, and providing directories of digital certificates.

X.509

An ITU Recommendation used as a standard for defining digital certificates.

A.2 Abbreviations and Acronyms

ASN.1

Abstract Syntax Notation version 1

BER

Basic Encoding Rules

CA

Certification Authority

CCI

Common Cryptographic Interface

CRL

Certificate Revocation List

CSR

Certificate Signing Request

DER

Distinguished Encoding Rules

DSA

Digital Signature Algorithm

EVP

Digital Envelope Library, an abstract interface to the CCI library

ITU

International Telecommunication Union

MAC

Message Authentication Code

PEM

Privacy Enhanced Mail

PKI

Public Key Infrastructure

RSA

Rivest Shamir Adelman

SSL

Secure Sockets Layer

TLS

Transport Layer Security