

CryptoGateway Documentation

Adrian Bedard

Jonathan Bedard

March 10, 2016

Contents

I	CryptoGateway Library	2
1	Introduction	3
1.1	Namespace	3
2	Class Index	4
2.1	Class List	4
3	File Index	7
3.1	File List	7
4	Namespace Documentation	10
4.1	crypto Namespace Reference	10
4.1.1	Typedef Documentation	14
4.1.2	Function Documentation	14
4.1.3	Variable Documentation	16
5	Class Documentation	18
6	File Documentation	19

Part I

CryptoGateway Library

Chapter 1

Introduction

The CryptoGateway library contains classes which handle cryptography. CryptoGateway is designed as an open source library, so much of the cryptography within the library is relatively simple. CryptoGateway is not meant to define cryptography to be used widely, rather, it is meant to provide a series of generalized hooks and interfaces which can be extended to various cryptographic algorithms.

1.1 Namespace

CryptoGateway uses the crypto namespace. The crypto namespace is designed for class, functions and constants related to cryptography. CryptoGateway depends on many of the tools defined in the os namespace. Additionally, the crypto namespace contains a series of nested namespaces which help to disambiguate constants.

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

crypto::actionOnFileClosed	
File closed error	??
crypto::actionOnFileError	
File error	??
crypto::avlKeyBank	
AVL key bank	??
crypto::binaryDecryptor	
Encrypted binary file output	??
crypto::binaryEncryptor	
Encrypted binary file output	??
crypto::bufferLargeError	
Buffer too large	??
crypto::bufferSmallError	
Buffer too small	??
crypto::checksum_message	
crypto::customError	
Custom crypto::error (p. ??)	??
crypto::error	
Sortable exception	??
crypto::errorListener	
Crypto::error listener	??
crypto::errorSender	
Sends crypto::error (p. ??)	??
crypto::fileFormatError	
File format error	??
crypto::fileOpenError	
File open error	??
crypto::hash	
Base hash class	??
crypto::hashCompareError	
Hash mis-match	??

crypto::hashGenerationError		
Hash generation error	..	??
crypto::illegalAlgorithmBind		
Algorithm bound failure	..	??
crypto::insertionFailed		
ADS Insertion Failed	..	??
crypto::integer		
Integer number definition	..	??
crypto::interior_message		??
crypto::keyBank		
Key bank interface	..	??
crypto::keyMissing		
Key missing error	..	??
crypto::large_integer		??
crypto::large_number		??
crypto::masterMismatch		
Master mis-match	..	??
crypto::nodeGroup		
Node group	..	??
crypto::nodeKeyReference		
Key storage node	..	??
crypto::nodeNameReference		
Name storage node	..	??
crypto::NULLDataError		
NULL data error	..	??
crypto::NULLMaster		
NULL master error	..	??
crypto::NULLPublicKey		
NULL public-key error	..	??
crypto::number		
Basic number definition	..	??
numberType		
Number type function structure	..	??
crypto::passwordLargeError		
Symmetric key too big	..	??
crypto::passwordSmallError		
Symmetric key too small	..	??
crypto::publicKey		
Base public-key class	..	??
crypto::publicKeyPackage< pkType >		??
crypto::publicKeyPackageFrame		??
crypto::publicKeySizeWrong		
Public-key size error	..	??
crypto::publicKeyTypeBank		??
crypto::publicRSA		
RSA public-key encryption	..	??
crypto::rc4Hash		
RC-4 hash class	..	??
crypto::RCFour		??

crypto::RSAKeyGenerator		
Helper key generation class	..	??
crypto::security_gateway	..	??
crypto::streamCipher	..	??
crypto::streamDecrypter	..	??
crypto::streamEncrypter	..	??
crypto::streamPackage< streamType, hashType >	..	??
crypto::streamPackageFrame	..	??
crypto::streamPackageTypeBank	..	??
crypto::streamPacket	..	??
crypto::unknownErrorType		
Unknown error	..	??
crypto::user		
Primary user class	..	??
crypto::xorHash		
XOR hash class	..	??

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

binaryEncryption.cpp	Implementation of binary encryption files	??
binaryEncryption.h	Definition of binary encryption files	??
c_BaseTen.c	Implementation of base-10 algorithms	??
c_BaseTen.h	Base-10 number functions	??
c_cryptoTesting.cpp	Implementation for C file testing	??
c_cryptoTesting.h	Header for C file testing	??
c_numberDefinitions.c	Implementation of basic number	??
c_numberDefinitions.h	Basic number declarations	??
cryptoCConstants.h	Extern declarations of C constants	??
cryptoCHeaders.h	Collected headers for C source code	??
cryptoConstants.cpp	Implementation of CryptoGateway constants	??
cryptoConstants.h	Extern definitions of CryptoGateway constants	??
cryptoCSource.cpp	Implementation of all C code	??
cryptoError.cpp	Implementation of error sender and listener	??
cryptoError.h	Declaration of cryptographic errors	??

cryptoFileTest.cpp	Implementation for cryptographic file testing	??
cryptoFileTest.h	Header for cryptographic file testing	??
cryptoFrameworks.cpp	Deprecated public-key framework implementation	??
cryptoFrameworks.h	Deprecated public-key framework declaration	??
CryptoGateway.h	Global include file	??
cryptoHash.cpp	Implementation of crypto hashing	??
cryptoHash.h	Declaration of crypto hashing	??
cryptoLogging.cpp	Logging for crypto namespace, implementation	??
cryptoLogging.h	??
cryptoNumber.cpp	Implements basic number types	??
cryptoNumber.h	Defines basic number types	??
cryptoNumberTest.cpp	Testing crypto::number (p. ??) and crypto::integer (p. ??)	??
cryptoPublicKey.cpp	Generalized and RSA public key implementation	??
cryptoPublicKey.h	Generalized and RSA public keys	??
cryptoTest.cpp	CryptoGateway library test constructor	??
cryptoTest.h	CryptoGateway library test header	??
file_mechanics.h	Deprecated file functions	??
gateway.cpp	??
gateway.h	??
gatewayTest.cpp	Implementation for end-to-end gateway testing	??
gatewayTest.h	Header for end-to-end gateway testing	??
hashTest.cpp	Implementation for hash tests	??
hashTest.h	Header for hash testing	??
hexConversion.cpp	??
hexConversion.h	??
interior_message.cpp	??
interior_message.h	??
keyBank.cpp	Implimentation for the AVL tree based key bank	??

keyBank.h	
Header for the AVL tree based key bank	??
large_number.cpp	??
large_number.h	??
public_key.cpp	
Old RSA implementation	??
public_key.h	
Old RSA declaration	??
publicKeyPackage.cpp	??
publicKeyPackage.h	??
publicKeyTest.h	
Public Key tests	??
RC4_Hash.cpp	??
RC4_Hash.h	??
security_gateway.cpp	??
security_gateway.h	??
securitySpinLock.cpp	??
securitySpinLock.h	??
staticTestKeys.cpp	
Auto-generated	??
staticTestKeys.h	
Auto-generated	??
streamCipher.cpp	??
streamCipher.h	??
streamPackage.cpp	??
streamPackage.h	??
streamTest.cpp	
Implementation for stream tests	??
streamTest.h	
Header for stream testing	??
testKeyGeneration.cpp	??
testKeyGeneration.h	
Implementation of test key binding	??
user.cpp	
Implementation of the CryptoGateway user	??
user.h	
Definition of the CryptoGateway user	??
XMLEncryption.cpp	??
XMLEncryption.h	??

Chapter 4

Namespace Documentation

4.1 crypto Namespace Reference

Classes

- class **actionOnFileClosed**
File closed error.
- class **actionOnFileError**
File error.
- class **avlKeyBank**
AVL key bank.
- class **binaryDecryptor**
Encrypted binary file output.
- class **binaryEncryptor**
Encrypted binary file output.
- class **bufferLargeError**
Buffer too large.
- class **bufferSmallError**
Buffer too small.
- class **checksum_message**
- class **customError**
*Custom **crypto::error** (p. ??).*
- class **error**
Sortable exception.
- class **errorListener**
***crypto::error** (p. ??) listener*
- class **errorSender**
*Sends **crypto::error** (p. ??).*
- class **fileFormatError**
File format error.
- class **fileOpenError**

- File open error.*
- class **hash**
 - Base hash class.*
- class **hashCompareError**
 - Hash mis-match.*
- class **hashGenerationError**
 - Hash generation error.*
- class **illegalAlgorithmBind**
 - Algorithm bound failure.*
- class **insertionFailed**
 - ADS Insertion Failed.*
- class **integer**
 - Integer number definition.*
- class **interior_message**
- class **keyBank**
 - Key bank interface.*
- class **keyMissing**
 - Key missing error.*
- class **large_integer**
- class **large_number**
- class **masterMismatch**
 - Master mis-match.*
- class **nodeGroup**
 - Node group.*
- class **nodeKeyReference**
 - Key storage node.*
- class **nodeNameReference**
 - Name storage node.*
- class **NULLDataError**
 - NULL data error.*
- class **NULLMaster**
 - NULL master error.*
- class **NULLPublicKey**
 - NULL public-key error.*
- class **number**
 - Basic number definition.*
- class **passwordLargeError**
 - Symmetric key too big.*
- class **passwordSmallError**
 - Symmetric key too small.*
- class **publicKey**
 - Base public-key class.*
- class **publicKeyPackage**

- class **publicKeyPackageFrame**
- class **publicKeySizeWrong**
Public-key size error.
- class **publicKeyTypeBank**
- class **publicRSA**
RSA public-key encryption.
- class **rc4Hash**
RC-4 hash class.
- class **RCFour**
- class **RSAKeyGenerator**
Helper key generation class.
- class **security_gateway**
- class **streamCipher**
- class **streamDecrypter**
- class **streamEncrypter**
- class **streamPackage**
- class **streamPackageFrame**
- class **streamPackageTypeBank**
- class **streamPacket**
- class **unknownErrorType**
Unknown error.
- class **user**
Primary user class.
- class **xorHash**
XOR hash class.

Typedefs

- typedef os::smart_ptr< **error** > **errorPointer**
*Smart pointer to **crypto::error** (p. ??).*
- typedef os::smart_ptr< **interior_message** > **smartInteriorMessage**

Functions

- std::ostream & **operator**<< (std::ostream &os, const **hash** &num)
Output stream operator.
- std::istream & **operator**>> (std::istream &is, **hash** &num)
Input stream operator.
- template<class hashClass >
hashClass **hashData** (uint16_t hashType, const unsigned char *data, uint32_t length)
Hashes data with the specified algorithm.
- std::ostream & **cryptoout_func** ()
Standard out object for crypto namespace.
- std::ostream & **cryptoerr_func** ()

Standard error object for crypto namespace.

- `std::ostream & operator<< (std::ostream &os, const number &num)`

Output stream operator.

- `std::istream & operator>> (std::istream &is, number &num)`

Input stream operator.

- `bool isHexCharacter (char c)`
- `std::string toHex (unsigned char i)`
- `std::string toHex (uint32_t i)`
- `unsigned char fromHex8 (const std::string &str)`
- `uint32_t fromHex32 (const std::string &str)`
- `static std::vector< std::string > generateArgumentList (os::smartXMLNode head)`
- `static void recursiveXMLPrinting (os::smartXMLNode head, os::smart_ptr< streamCipher > strm, std::vector< std::string > args, std::ofstream &ofs)`
- `static os::smartXMLNode recursiveXMLBuilding (os::smart_ptr< streamCipher > strm, std::vector< std::string > args, std::ifstream &ifs)`
- `bool EXML_Output (std::string path, os::smartXMLNode head, std::string password, os::smart_ptr< streamPackageFrame > spf)`
- `bool EXML_Output (std::string path, os::smartXMLNode head, unsigned char *symKey, unsigned int passwordLength, os::smart_ptr< streamPackageFrame > spf)`
- `bool EXML_Output (std::string path, os::smartXMLNode head, os::smart_ptr< publicKey > pbk, unsigned int lockType, os::smart_ptr< streamPackageFrame > spf)`
- `bool EXML_Output (std::string path, os::smartXMLNode head, os::smart_ptr< number > publicKey, unsigned int pkAlgo, unsigned int pkSize, os::smart_ptr< streamPackageFrame > spf)`
- `os::smartXMLNode EXML_Input (std::string path, std::string password)`
- `os::smartXMLNode EXML_Input (std::string path, unsigned char *symKey, unsigned int passwordLength)`
- `os::smartXMLNode EXML_Input (std::string path, os::smart_ptr< publicKey > pbk, os::smart_ptr< keyBank > kyBank, os::smart_ptr< nodeGroup > &author)`
- `os::smartXMLNode EXML_Input (std::string path, os::smart_ptr< publicKey > pbk)`
- `os::smartXMLNode EXML_Input (std::string path, os::smart_ptr< keyBank > kyBank)`
- `os::smartXMLNode EXML_Input (std::string path, os::smart_ptr< keyBank > kyBank, os::smart_ptr< nodeGroup > &author)`

Variables

- `bool global_logging`
Deprecated logging flag.
- `os::smart_ptr< std::ostream > cryptoout_ptr`
Standard out pointer for crypto namespace.
- `os::smart_ptr< std::ostream > cryptoerr_ptr`
Standard error pointer for crypto namespace.
- `const unsigned int MESSAGE_MAX =512`
- `const unsigned int CHECKSUM_SIZE =4`
- `const unsigned int LARGE_NUMBER_SIZE =32`
- `const unsigned int PRIME_TEST_ITERATION =10`
- `static os::smart_ptr< publicKeyTypeBank > _singleton`
- `static os::smart_ptr< streamPackageTypeBank > _singleton`

4.1.1 Typedef Documentation

typedef os::smart_ptr<**error**> **crypto::errorPointer**

Smart pointer to **crypto::error** (p. ??).

typedef os::smart_ptr<**interior_message**> **crypto::smartInteriorMessage**

4.1.2 Function Documentation

std::ostream& crypto::cryptoerr_func ()

Standard error object for crypto namespace.

#define statements allow the user to call this function with "crypto::cryptoerr." Logging is achieved by using "crypto::cryptoerr" as one would use "std::cerr."

std::ostream& crypto::cryptoout_func ()

Standard out object for crypto namespace.

#define statements allow the user to call this function with "crypto::cryptoout." Logging is achieved by using "crypto::cryptoout" as one would use "std::cout."

os::smartXMLNode crypto::EXML_Input (std::string path, std::string password)

os::smartXMLNode crypto::EXML_Input (std::string path, unsigned char * symKey, unsigned int passwordLength)

os::smartXMLNode crypto::EXML_Input (std::string path, os::smart_ptr< **publicKey** > pbk, os::smart_ptr< **keyBank** > kyBank, os::smart_ptr< **nodeGroup** > & author)

os::smartXMLNode crypto::EXML_Input (std::string path, os::smart_ptr< **publicKey** > pbk)

os::smartXMLNode crypto::EXML_Input (std::string path, os::smart_ptr< **keyBank** > kyBank)

os::smartXMLNode crypto::EXML_Input (std::string path, os::smart_ptr< **keyBank** > kyBank, os::smart_ptr< **nodeGroup** > & author)

bool crypto::EXML_Output (std::string path, os::smartXMLNode head, std::string password, os::smart_ptr< **streamPackageFrame** > spf)

bool crypto::EXML_Output (std::string path, os::smartXMLNode head, unsigned char * symKey, unsigned int passwordLength, os::smart_ptr< **streamPackageFrame** > spf)

bool crypto::EXML_Output (std::string path, os::smartXMLNode head, os::smart_ptr< **publicKey** > pbk, unsigned int lockType, os::smart_ptr< **streamPackageFrame** > spf)

bool crypto::EXML_Output (std::string path, os::smartXMLNode head, os::smart_ptr< **number** > publicKey, unsigned int pkAlgo, unsigned int pkSize, os::smart_ptr< **streamPackageFrame** > spf)

uint32_t crypto::fromHex32 (const std::string & str)

unsigned char crypto::fromHex8 (const std::string & str)

```
static std::vector<std::string> crypto::generateArgumentList ( os::smartXMLNode head )
[static]
```

```
template<class hashClass > hashClass crypto::hashData ( uint16_t hashType, const unsigned
char * data, uint32_t length )
```

Hashes data with the specified algorithm.

Hashes the provided data array returning a hash of the specified algorithm. This is a template function, which calls the static hash function for the specified algorithm.

Parameters

in	<i>hashType</i>	Size of hash
in	<i>data</i>	Data array to be hashed
in	<i>length</i>	Length of data to be hashed

Returns

Hash for data array

```
bool crypto::isHexCharacter ( char c )
```

```
std::ostream& crypto::operator<< ( std::ostream & os, const number & num )
```

Output stream operator.

Parameters

	<i>[in/out]</i>	os Output stream
in	<i>num</i>	Number to be output

Returns

reference to std::ostream& os

```
std::ostream& crypto::operator<< ( std::ostream & os, const hash & num )
```

Output stream operator.

Outputs a hex version of the hash to the provided output stream. This output will look identical for two hashes which are equal but have different algorithms.

Parameters

	<i>[in/out]</i>	os Output stream
in	<i>num</i>	Hash to be printed return Reference to output stream

`std::istream& crypto::operator>> (std::istream & is, number & num)`

Input stream operator.

Parameters

	<i>[in/out]</i>	is Input stream
in	<i>num</i>	Number to set with the string

Returns

reference to `std::istream& is`

`std::istream& crypto::operator>> (std::istream & is, hash & num)`

Input stream operator.

Inputs a hex version of the hash from the provided output stream. This function must receive a constructed hash, although it will rebuild the provided hash with the stream data.

Parameters

	<i>[in/out]</i>	is Input stream
in	<i>num</i>	Hash to be created return Reference to input stream

`static os::smartXMLNode crypto::recursiveXMLBuilding (os::smart_ptr< streamCipher > strm, std::vector< std::string > args, std::ifstream & ifs) [static]`

`static void crypto::recursiveXMLPrinting (os::smartXMLNode head, os::smart_ptr< streamCipher > strm, std::vector< std::string > args, std::ofstream & ofs) [static]`

`std::string crypto::toHex (unsigned char i)`

`std::string crypto::toHex (uint32_t i)`

4.1.3 Variable Documentation

`os::smart_ptr<publicKeyTypeBank> crypto::_singleton [static]`

`os::smart_ptr<streamPackageTypeBank> crypto::_singleton [static]`

`const unsigned int crypto::CHECKSUM_SIZE =4`

`os::smart_ptr<std::ostream> crypto::cryptoerr_ptr`

Standard error pointer for crypto namespace.

This `std::ostream` is used as standard error for the crypto namespace. This pointer can be swapped out to programmatically redirect standard error for the crypto namespace.

`os::smart_ptr<std::ostream> crypto::cryptoout_ptr`

Standard out pointer for crypto namespace.

This `std::ostream` is used as standard out for the crypto namespace. This pointer can be swapped out to programmatically redirect standard out for the crypto namespace.

`bool crypto::global_logging`

Deprecated logging flag.

Old logging flag. Deprecated in the new CryptoGateway files. This has been replaced by the logging system outlined in this file.

`const unsigned int crypto::LARGE_NUMBER_SIZE =32`

`const unsigned int crypto::MESSAGE_MAX =512`

`const unsigned int crypto::PRIME_TEST_ITERATION =10`

Chapter 5

Class Documentation

Chapter 6

File Documentation