CryptoGateway Documentation

Adrian Bedard Jonathan Bedard

February 22, 2016

Contents

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. CrytpoGateway 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:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Namespace Documentation

4.1 crypto Namespace Reference

Classes

- class **actionOnFileClosed**

    File closed error.
- class **actionOnFileError**

    File error.
- class **avlKeyBank**

    AVL key back.
- 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**
- class **hashCompareError**

  Hash mis-match.
- class **hashGenerationError**

  Hash generation error.
- class **illegalAlgorithmBind**

  Algorithm bound failure.
- class **insertionFailed**

  ADS Insertion Failed.
- class **integer**
- class **interior_message**
- class **keyBank**

  Key bank interface.
- 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**
- class **passwordLargeError**

  Symmetric key too big.
- class **passwordSmallError**

  Symmetric key too small.
- class **publicField**
- class **publicKey**
- class **publicKeyPackage**
- class **publicKeyPackageFrame**
- class **publicKeySizeWrong**

  Public-key size error.
- class **publicKeyTypeBank**
- class **publicRSA**

- class **rc4Hash**
- class **RCFour**
- class **RSAKeyGenerator**
- class **security_gateway**
- class **streamCipher**
- class **streamDecrypter**
- class **streamEncrypter**
- class **streamPackage**
- class **streamPackageFrame**
- class **streamPackageTypeBank**
- class **streamPacket**
- class **unknownErrorType**

  Unknown error.
- class **xorHash**

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)
- std::istream & **operator**>> (std::istream &is, **hash** &num)
- template<class hashClass >
  hashClass **hashData** (uint16_t hashType, const unsigned char ∗data, uint32_t length)
- std::ostream & **cryptoout_func** ()
- std::ostream & **cryptoerr_func** ()
- std::ostream & **operator**<< (std::ostream &os, const **number** &num)
- std::istream & **operator**>> (std::istream &is, **number** &num)
- static uint16_t **to_comp_mode_sgtw** (uint16_t i)
- static uint16_t **from_comp_mode_sgtw** (uint16_t i)
- static uint32_t **to_comp_mode_sgtw** (uint32_t i)
- static uint32_t **from_comp_mode_sgtw** (uint32_t i)
- static uint64_t **to_comp_mode_sgtw** (uint64_t i)
- static uint64_t **from_comp_mode_sgtw** (uint64_t i)
- static bool **file_exists** (const std::string &file_name)
- static uint64_t **get_timestamp** ()
- static std::string **convertTimestamp** (uint64_t stamp)
- static bool **check_numeric** (const char char_to_check)
- static int **conver_char_int** (const char char_to_check)
- static uint64_t **convert_64** (const std::string &str)
- 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, os::smart_ptr< **publicKey** > pbk, os::smart_ptr< **streamPackageFrame** > spf)
- os::smartXMLNode **EXML_Input** (std::string path, std::string password)
- os::smartXMLNode **EXML_Input** (std::string path, os::smart_ptr< **publicKey** > pbk)

Variables

- const unsigned int **PUBLIC_FIELD_NO_TYPE** =0
- bool **global_logging** = false
- os::smart_ptr< std::ostream > **cryptoout_ptr** = &(std::cout)
- os::smart_ptr< std::ostream > **cryptoerr_ptr** = &(std::cerr)
- 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

static bool crypto::check_numeric (const char char_to_check) `[static]`

static int crypto::conver_char_int (const char char_to_check) `[static]`

static uint64_t crypto::convert_64 (const std::string & str) `[static]`

static std::string crypto::convertTimestamp (uint64_t stamp) `[static]`

std::ostream & crypto::cryptoerr_func ( )

std::ostream & crypto::cryptoout_func ( )

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

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, os::smart_ptr< **publicKey** > pbk, os::smart_ptr< **streamPackageFrame** > spf)

static bool crypto::file_exists (const std::string & file_name) `[static]`

static uint16_t crypto::from_comp_mode_sgtw (uint16_t i) `[static]`

static uint32_t crypto::from_comp_mode_sgtw (uint32_t i) `[static]`

static uint64_t crypto::from_comp_mode_sgtw (uint64_t i) `[static]`

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]`

static uint64_t crypto::get_timestamp ( ) `[static]`

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

bool crypto::isHexCharacter (char c)

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

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

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

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

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]`

static uint16_t crypto::to_comp_mode_sgtw (uint16_t i) `[static]`

static uint32_t crypto::to_comp_mode_sgtw (uint32_t i) `[static]`

static uint64_t crypto::to_comp_mode_sgtw (uint64_t i) `[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 = &(std::cerr)

os::smart_ptr< std::ostream > crypto::cryptoout_ptr = &(std::cout)

bool crypto::global_logging = false

const unsigned int crypto::LARGE_NUMBER_SIZE =32

const unsigned int crypto::MESSAGE_MAX =512

const unsigned int crypto::PRIME_TEST_ITERATION =10

const unsigned int crypto::PUBLIC_FIELD_NO_TYPE =0

Chapter 5

Class Documentation

Chapter 6

File Documentation