ns-3

3-dev

Generated by Doxygen 1.8.17

1 Module Documentation

1.1 Application-helper

Be sure to read repository README.md before going down to the API.

Classes

- class ns3::SensorNodeHelper

  Helper class for the creation of SensorNode applications.
- class ns3::SinkHelper

  Helper class for the creation of Sink applications.

1.1.1 Detailed Description

Be sure to read repository README.md before going down to the API.

1.2 Enumerators

Be sure to read repository README.md before going down to the API.

Enumerations

- enum ns3::BodyOptions { ns3::NO_Body = 0, ns3::Aggregate, ns3::FixedSize, ns3::AggregateAndFixed }

  Specifies how the onion body must behave.
- enum ns3::IEEE_80211n { ns3::F_24GHz = 0, ns3::F_5GHz }

  Enumeration defining different carrier frequencies of the standard IEEE 802.11n.
- enum ns3::Routing { ns3::AODV = 0, ns3::DSR, ns3::OLSR, ns3::DSDV }

  Enumeration defining different routing algorithms that can be used in the simulator.
- enum ns3::Topology { ns3::GRID = 0, ns3::DISC }

  Enumeration defining different network topologies that can be used in the simulation.
- enum ns3::Verbosity { ns3::NO, ns3::ConsoleLog, ns3::PrintDescription, ns3::Both }

  Verbosity settings of the simulation.

1.2.1 Detailed Description

Be sure to read repository README.md before going down to the API.

1.2.2 Enumeration Type Documentation

1.2.2.1 BodyOptions `enum ns3::BodyOptions`

Specifies how the onion body must behave.

Enumerator

NO_Body	The onion message won't have an onion body.
Aggregate	The onion body will only aggregate a value.
FixedSize	the onion body will have a fixed size specified by the ns3::Sink::BodySize attribute
AggregateAndFixed	The onion body will aggregate a value and will maintain a fixed size apecified by the ns3::Sink::BodySize attribute.

Definition at line 97 of file enums.h.

1.2.2.2 IEEE_80211n `enum ns3::IEEE_80211n`

Enumeration defining different carrier frequencies of the standard IEEE 802.11n.

Enumerator

F_24GHz	2.4GHz
F_5GHz	5GHz

Definition at line 70 of file enums.h.

1.2.2.3 Routing `enum ns3::Routing`

Enumeration defining different routing algorithms that can be used in the simulator.

Enumerator

AODV	Ad Hoc On-Demand Distance Vector ns3::Aodv.
DSR	Dynamic Source Routing ns3::Dsr.
OLSR	Optimized Link State Routing Protocol ns3::Olsr.
DSDV	Destination-Sequenced Distance Vector routing ns3::Dsdv.

Definition at line 44 of file enums.h.

1.2.2.4 Topology `enum ns3::Topology`

Enumeration defining different network topologies that can be used in the simulation.

Enumerator

GRID	Grid topology.
DISC	Random disc topology.

Definition at line 58 of file enums.h.

1.2.2.5 Verbosity `enum ns3::Verbosity`

Verbosity settings of the simulation.

Enumerator

NO	No output on console log, except notifying simulation start and end, output data in csv file.
ConsoleLog	Output data and simulation description on console log, output data in csv file.
PrintDescription	No output on console log, print simulation description and data on csv file.
Both	print all simulation output on console log and csv file

Definition at line 82 of file enums.h.

1.3 Managers

Be sure to read repository README.md before going down to the API.

Classes

- class ns3::OnionManager

  Class that manages encryption keys and the encryption and decryption of layers of onion messagess The class is implementing the OnionRouting abstract class with the encryption and decryption methods.
- class ns3::OnionValidator

  Class shared between wsn nodes used to track how the onion is transiting in the WSN The class uses two counters `m_onionSeq` and `m_hopCount` to identify when an onion neeeds to be aborted.
- class ns3::OutputManager

  Class that manages the output of the simulation. ConsoleLog output and Output on .csv file

1.3.1 Detailed Description

Be sure to read repository README.md before going down to the API.

1.4 Node-application

Be sure to read repository README.md before going down to the API.

Classes

- class ns3::SensorNode

  The application of the sensor node.
- class ns3::Sink

  The application of the sink node. The node that generates onion messagess.
- class ns3::Wsn_node

  The wsn node base class that manages the sending and receiving of packets and basic configuration of nodes.

1.4.1 Detailed Description

Be sure to read repository README.md before going down to the API.

1.5 Onion-routing

This section documents the API of the ns-3 OnionRouting module. For a functional description, please refer to the ns-3 manual here: `http://www.nsnam.org/docs/models/html/onion-routing.html`.

Classes

- class ns3::OnionRouting

  Abstract class for creation and decryption of Onion messages.
- class ns3::OnionRoutingDummyEncryption

  class that implements the
- struct ns3::orLayer

  structure holding details resulting from layer decryption of an onion message

1.5.1 Detailed Description

This section documents the API of the ns-3 OnionRouting module. For a functional description, please refer to the ns-3 manual here: `http://www.nsnam.org/docs/models/html/onion-routing.html`.

Be sure to read the manual BEFORE going down to the API.

1.6 Onion_routing_wsn

Be sure to read repository README.md before going down to the API.

Classes

- class WsnConstructor

    The class that constructs the WSN, setup applications on nodes and starts the simulation.

1.6.1 Detailed Description

Be sure to read repository README.md before going down to the API.

1.7 Serialization

Be sure to read repository README.md before going down to the API.

Classes

- class ns3::SegmentNum

    Class for adding a tag to packets used to track different segments of packets packets are fragmented due to small MSS Just implementing methods from ns3 class ns3::Tag.

- class ns3::SerializationWrapper

    Class for the serialization-deserialization of the messagess to send in packets.

1.7.1 Detailed Description

Be sure to read repository README.md before going down to the API.

2 Namespace Documentation

2.1 internal Namespace Reference

2.2 ns3 Namespace Reference

Classes

- class OnionManager

  Class that manages encryption keys and the encryption and decryption of layers of onion messagess The class is implementing the OnionRouting abstract class with the encryption and decryption methods.
- class OnionRouting

  Abstract class for creation and decryption of Onion messages.
- class OnionRoutingDummyEncryption

  class that implements the
- class OnionValidator

  Class shared between wsn nodes used to track how the onion is transiting in the WSN The class uses two counters `m_onionSeq` and `m_hopCount` to identify when an onion neeeds to be aborted.
- struct orLayer

  structure holding details resulting from layer decryption of an onion message
- class OutputManager

  Class that manages the output of the simulation. ConsoleLog output and Output on .csv file
- class SegmentNum

  Class for adding a tag to packets used to track different segments of packets packets are fragmented due to small MSS Just implementing methods from ns3 class ns3::Tag.
- class SensorNode

  The application of the sensor node.
- class SensorNodeHelper

  Helper class for the creation of SensorNode applications.
- class SerializationWrapper

  Class for the serialization-deserialization of the messagess to send in packets.
- class Sink

  The application of the sink node. The node that generates onion messagess.
- class SinkHelper

  Helper class for the creation of Sink applications.
- class Wsn_node

  The wsn node base class that manages the sending and receiving of packets and basic configuration of nodes.

Enumerations

- enum BodyOptions { NO_Body = 0, Aggregate, FixedSize, AggregateAndFixed }

  Specifies how the onion body must behave.
- enum IEEE_80211n { F_24GHz = 0, F_5GHz }

  Enumeration defining different carrier frequencies of the standard IEEE 802.11n.
- enum Routing { AODV = 0, DSR, OLSR, DSDV }

  Enumeration defining different routing algorithms that can be used in the simulator.
- enum Topology { GRID = 0, DISC }

  Enumeration defining different network topologies that can be used in the simulation.
- enum Verbosity { NO, ConsoleLog, PrintDescription, Both }

  Verbosity settings of the simulation.

2.3 protomessage Namespace Reference

Classes

- class ProtoPacket
- class ProtoPacket_Handshake
- struct ProtoPacket_HandshakeDefaultTypeInternal
- class ProtoPacket_OnionBody
- struct ProtoPacket_OnionBodyDefaultTypeInternal
- class ProtoPacket_OnionHead
- struct ProtoPacket_OnionHeadDefaultTypeInternal
- struct ProtoPacketDefaultTypeInternal

Variables

- PROTOBUF_ATTRIBUTE_NO_DESTROY PROTOBUF_CONSTINIT ProtoPacketDefaultTypeInternal _ProtoPacket_default_instance_
- PROTOBUF_ATTRIBUTE_NO_DESTROY PROTOBUF_CONSTINIT ProtoPacket_HandshakeDefaultTypeInternal _ProtoPacket_Handshake_default_instance_
- PROTOBUF_ATTRIBUTE_NO_DESTROY PROTOBUF_CONSTINIT ProtoPacket_OnionBodyDefaultTypeInternal _ProtoPacket_OnionBody_default_instance_
- PROTOBUF_ATTRIBUTE_NO_DESTROY PROTOBUF_CONSTINIT ProtoPacket_OnionHeadDefaultTypeInternal _ProtoPacket_OnionHead_default_instance_

2.3.1 Variable Documentation

2.3.1.1 _ProtoPacket_default_instance_ `ProtoPacketDefaultTypeInternal protomessage::_ProtoPacket_default_instance_`

Definition at line 72 of file proto-packet.pb.cc.

Referenced by protomessage::ProtoPacket::internal_default_instance().

2.3.1.2 _ProtoPacket_Handshake_default_instance_ `ProtoPacket_HandshakeDefaultTypeInternal protomessage::_ProtoPacket_Handshake_default_instance_`

Definition at line 58 of file proto-packet.pb.cc.

Referenced by protomessage::ProtoPacket::_internal_h_shake(), and protomessage::ProtoPacket_Handshake::internal_default_instance().

2.3.1.3 _ProtoPacket_OnionBody_default_instance_ `ProtoPacket_OnionBodyDefaultTypeInternal protomessage::_ProtoPacket_OnionBody_default_instance_`

Definition at line 46 of file proto-packet.pb.cc.

Referenced by protomessage::ProtoPacket::_internal_o_body(), and protomessage::ProtoPacket_OnionBody↩ ::internal_default_instance().

2.3.1.4 _ProtoPacket_OnionHead_default_instance_ `ProtoPacket_OnionHeadDefaultTypeInternal protomessage::_ProtoPacket_OnionHead_default_instance_`

Definition at line 33 of file proto-packet.pb.cc.

Referenced by protomessage::ProtoPacket::_internal_o_head(), and protomessage::ProtoPacket_OnionHead↩ ::internal_default_instance().

3 Class Documentation

3.1 protomessage::ProtoPacket_OnionHead::_Internal Class Reference

Public Types

- using HasBits = decltype(std::declval< ProtoPacket_OnionHead >()._has_bits_)

Static Public Member Functions

- static void set_has_onion_message (HasBits ∗has_bits)
- static void set_has_onionid (HasBits ∗has_bits)
- static void set_has_padding (HasBits ∗has_bits)

3.1.1 Detailed Description

Definition at line 169 of file proto-packet.pb.cc.

3.1.2 Member Typedef Documentation

3.1.2.1 HasBits `using protomessage::ProtoPacket_OnionHead::_Internal::HasBits = decltype(std↩ ::declval<ProtoPacket_OnionHead>()._has_bits_)`

Definition at line 171 of file proto-packet.pb.cc.

3.1.3 Member Function Documentation

3.1.3.1 set_has_onion_message()

```
static void protomessage::ProtoPacket_OnionHead::_Internal↩
::set_has_onion_message (
            HasBits * has_bits )  [inline], [static]
```

Definition at line 175 of file proto-packet.pb.cc.

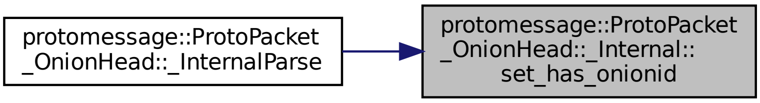
3.1.3.2 set_has_onionid()

```
static void protomessage::ProtoPacket_OnionHead::_Internal::set_has↩
_onionid (
            HasBits * has_bits )  [inline], [static]
```

Definition at line 172 of file proto-packet.pb.cc.

Referenced by protomessage::ProtoPacket_OnionHead::_InternalParse().

Here is the caller graph for this function:


protomessage::ProtoPacket_OnionHead::_InternalParse
protomessage::ProtoPacket_OnionHead::_Internal::set_has_onionid

3.1.3.3 set_has_padding()

```
static void protomessage::ProtoPacket_OnionHead::_Internal::set_has↩
_padding (
            HasBits * has_bits )  [inline], [static]
```

Definition at line 178 of file proto-packet.pb.cc.

The documentation for this class was generated from the following file:

- src/onion_routing_wsn/protobuf/proto-packet.pb.cc

3.2 protomessage::ProtoPacket_OnionBody::_Internal Class Reference

Public Types

- using HasBits = decltype(std::declval< ProtoPacket_OnionBody >()._has_bits_)

Static Public Member Functions

- static void set_has_aggregatedvalue (HasBits ∗has_bits)
- static void set_has_padding (HasBits ∗has_bits)

3.2.1 Detailed Description

Definition at line 467 of file proto-packet.pb.cc.

3.2.2 Member Typedef Documentation

3.2.2.1 HasBits

```
using protomessage::ProtoPacket_OnionBody::_Internal::HasBits = decltype(std←
::declval<ProtoPacket_OnionBody>()._has_bits_)
```

Definition at line 469 of file proto-packet.pb.cc.

3.2.3 Member Function Documentation

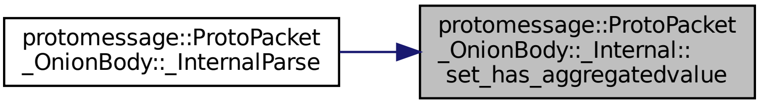
3.2.3.1 set_has_aggregatedvalue()

```
static void protomessage::ProtoPacket_OnionBody::_Internal←
::set_has_aggregatedvalue (
            HasBits ∗ has_bits )  [inline], [static]
```

Definition at line 470 of file proto-packet.pb.cc.

Referenced by protomessage::ProtoPacket_OnionBody::_InternalParse().

Here is the caller graph for this function:


protomessage::ProtoPacket
_OnionBody::_InternalParse
protomessage::ProtoPacket
_OnionBody::_Internal::
set_has_aggregatedvalue

3.2.3.2 set_has_padding() `static void protomessage::ProtoPacket_OnionBody::_Internal::set_has↩ _padding ( HasBits * has_bits ) [inline], [static]`

Definition at line 473 of file proto-packet.pb.cc.

The documentation for this class was generated from the following file:

- src/onion_routing_wsn/protobuf/proto-packet.pb.cc

3.3 protomessage::ProtoPacket_Handshake::_Internal Class Reference

Public Types

- using HasBits = decltype(std::declval< ProtoPacket_Handshake >()._has_bits_)

Static Public Member Functions

- static void set_has_publickey (HasBits *has_bits)

3.3.1 Detailed Description

Definition at line 714 of file proto-packet.pb.cc.

3.3.2 Member Typedef Documentation

3.3.2.1 HasBits `using protomessage::ProtoPacket_Handshake::_Internal::HasBits = decltype(std↩ ::declval<ProtoPacket_Handshake>()._has_bits_)`

Definition at line 716 of file proto-packet.pb.cc.

3.3.3 Member Function Documentation

3.3.3.1 set_has_publickey() `static void protomessage::ProtoPacket_Handshake::_Internal::set_↩ has_publickey ( HasBits * has_bits ) [inline], [static]`

Definition at line 717 of file proto-packet.pb.cc.

The documentation for this class was generated from the following file:

- src/onion_routing_wsn/protobuf/proto-packet.pb.cc

3.4 protomessage::ProtoPacket::_Internal Class Reference

Public Types

- using HasBits = decltype(std::declval< ProtoPacket >()._has_bits_)

Static Public Member Functions

- static const ::protomessage::ProtoPacket_Handshake & h_shake (const ProtoPacket ∗msg)
- static const ::protomessage::ProtoPacket_OnionBody & o_body (const ProtoPacket ∗msg)
- static const ::protomessage::ProtoPacket_OnionHead & o_head (const ProtoPacket ∗msg)
- static void set_has_h_shake (HasBits ∗has_bits)
- static void set_has_o_body (HasBits ∗has_bits)
- static void set_has_o_head (HasBits ∗has_bits)

3.4.1 Detailed Description

Definition at line 925 of file proto-packet.pb.cc.

3.4.2 Member Typedef Documentation

3.4.2.1 HasBits `using protomessage::ProtoPacket::_Internal::HasBits = decltype(std::declval<ProtoPacket>()._h`

Definition at line 927 of file proto-packet.pb.cc.

3.4.3 Member Function Documentation

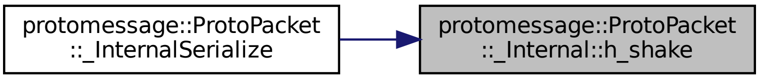
3.4.3.1 h_shake() `const ::protomessage::ProtoPacket_Handshake & protomessage::ProtoPacket::_↩ Internal::h_shake ( const ProtoPacket ∗ msg ) [static]`

Definition at line 943 of file proto-packet.pb.cc.

References protomessage::ProtoPacket::h_shake_.

Referenced by protomessage::ProtoPacket::_InternalSerialize().

Here is the caller graph for this function:


protomessage::ProtoPacket ::_InternalSerialize
protomessage::ProtoPacket ::_Internal::h_shake

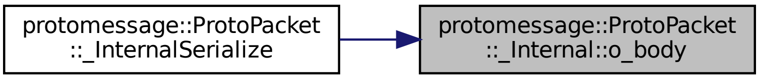
3.4.3.2 o_body() const ::protomessage::ProtoPacket_OnionBody & protomessage::ProtoPacket::_↩ Internal::o_body (const ProtoPacket * *msg*) [static]

Definition at line 951 of file proto-packet.pb.cc.

References protomessage::ProtoPacket::o_body_.

Referenced by protomessage::ProtoPacket::_InternalSerialize().

Here is the caller graph for this function:


protomessage::ProtoPacket ::_InternalSerialize
protomessage::ProtoPacket ::_Internal::o_body

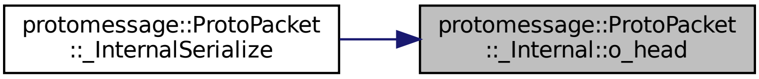
3.4.3.3 o_head() const ::protomessage::ProtoPacket_OnionHead & protomessage::ProtoPacket::_↩ Internal::o_head (const ProtoPacket * *msg*) [static]

Definition at line 947 of file proto-packet.pb.cc.

References protomessage::ProtoPacket::o_head_.

Referenced by protomessage::ProtoPacket::_InternalSerialize().

Here is the caller graph for this function:


protomessage::ProtoPacket ::_InternalSerialize
protomessage::ProtoPacket ::_Internal::o_head

3.4.3.4 set_has_h_shake() static void protomessage::ProtoPacket::_Internal::set_has_h_shake (HasBits * *has_bits*) [inline], [static]

Definition at line 929 of file proto-packet.pb.cc.

3.4.3.5 set_has_o_body() static void protomessage::ProtoPacket::_Internal::set_has_o_body (HasBits * *has_bits*) [inline], [static]

Definition at line 937 of file proto-packet.pb.cc.

3.4.3.6 set_has_o_head() static void protomessage::ProtoPacket::_Internal::set_has_o_head (HasBits * *has_bits*) [inline], [static]

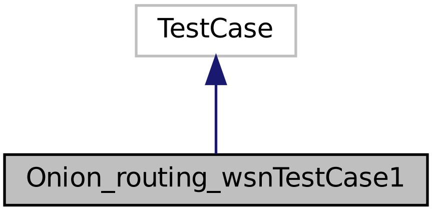
Definition at line 933 of file proto-packet.pb.cc.

The documentation for this class was generated from the following file:

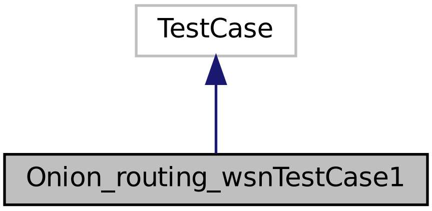
- src/onion_routing_wsn/protobuf/proto-packet.pb.cc

3.5 Onion_routing_wsnTestCase1 Class Reference

Inheritance diagram for Onion_routing_wsnTestCase1:



Collaboration diagram for Onion_routing_wsnTestCase1:



Public Member Functions

- Onion_routing_wsnTestCase1 ()
- virtual ∼Onion_routing_wsnTestCase1 ()

Private Member Functions

- virtual void DoRun (void)

3.5.1 Detailed Description

Definition at line 14 of file onion_routing_wsn-test-suite.cc.

3.5.2 Constructor & Destructor Documentation

3.5.2.1 Onion_routing_wsnTestCase1() `Onion_routing_wsnTestCase1::Onion_routing_wsnTestCase1 ( )`

Definition at line 25 of file onion_routing_wsn-test-suite.cc.

3.5.2.2 ∼Onion_routing_wsnTestCase1() `Onion_routing_wsnTestCase1::∼Onion_routing_wsnTestCase1 ( ) [virtual]`

Definition at line 32 of file onion_routing_wsn-test-suite.cc.

3.5.3 Member Function Documentation

3.5.3.1 DoRun() `void Onion_routing_wsnTestCase1::DoRun ( void ) [private], [virtual]`

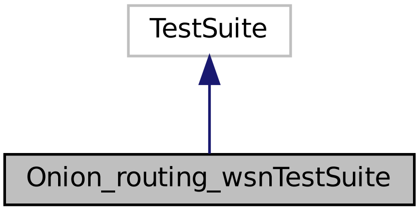
Definition at line 41 of file onion_routing_wsn-test-suite.cc.

The documentation for this class was generated from the following file:

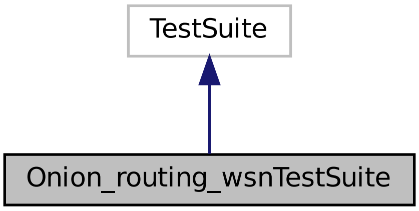
- src/onion_routing_wsn/test/onion_routing_wsn-test-suite.cc

3.6 Onion_routing_wsnTestSuite Class Reference

Inheritance diagram for Onion_routing_wsnTestSuite:



Collaboration diagram for Onion_routing_wsnTestSuite:



Public Member Functions

- Onion_routing_wsnTestSuite ()

3.6.1 Detailed Description

Definition at line 53 of file onion_routing_wsn-test-suite.cc.

3.6.2 Constructor & Destructor Documentation

3.6.2.1 Onion_routing_wsnTestSuite() `Onion_routing_wsnTestSuite::Onion_routing_wsnTestSuite ( )`

Definition at line 59 of file onion_routing_wsn-test-suite.cc.

The documentation for this class was generated from the following file:

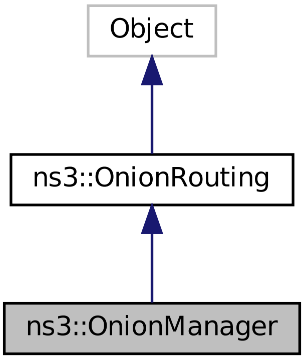
- src/onion_routing_wsn/test/onion_routing_wsn-test-suite.cc

3.7 ns3::OnionManager Class Reference

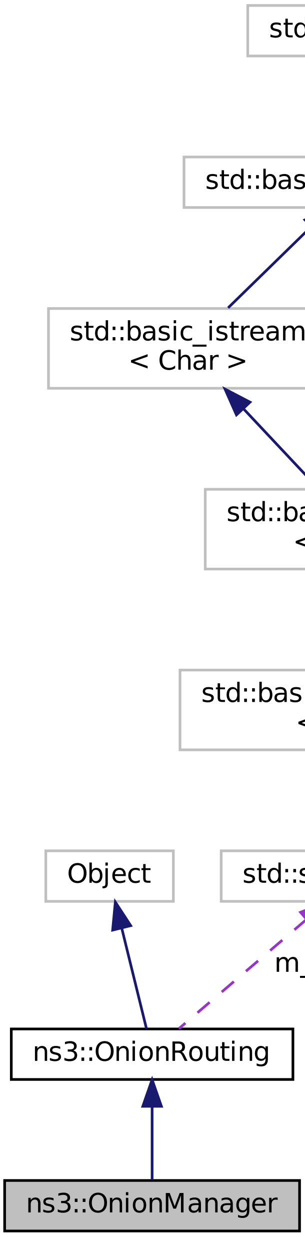
Class that manages encryption keys and the encryption and decryption of layers of onion messagess The class is implementing the OnionRouting abstract class with the encryption and decryption methods.

```
#include "onionmanager.h"
```

Inheritance diagram for ns3::OnionManager:



Collaboration diagram for ns3::OnionManager:


std::ios_base
std::basic_ios< Char >
std::basic_istream < Char >
std::basic_ostream < Char >
std::basic_iostream < Char >
std::basic_stringstream < char >
Object
std::stringstream
m_onionStream
ns3::OnionRouting
ns3::OnionManager

Public Member Functions

- OnionManager ()

  Default constructor.
- ∼OnionManager ()

  Default destructor.
- virtual void DecryptLayer (unsigned char ∗innerLayer, unsigned char ∗onion, uint16_t onionLen, unsigned char ∗pk, unsigned char ∗sk) const

Implementing decryption using the libsodium library.

- virtual void EncryptLayer (unsigned char ∗ciphertext, unsigned char ∗message, int len, unsigned char ∗key) const

  Implementing encryption using the libsodium library.
- void GenerateNewKeyPair (void)

  Generate a new public/private keypair using the libsodium library.
- unsigned char ∗ GetPK (void)

  accessor
- std::string GetPKtoString ()

  accessor
- unsigned char ∗ GetSK (void)

  accessor
- std::string GetSKtoString ()

  accessor
- unsigned char ∗ IpToBuff (uint32_t in)

  Convert an Ipv4 address given as an unsigned integer value to buffer array of 4Bytes.
- void SetPK (unsigned char ∗pk)

  setter
- void SetSK (unsigned char ∗sk)

  setter
- unsigned char ∗ StringToUchar (std::string in)

  Convert a string to an array of unsigned chars.
- std::string UcharToString (unsigned char ∗seq, int len)

  Convert an array of unsigned chars to a std::string.

Static Public Member Functions

- static TypeId GetTypeId (void)

  Register this type.

Private Attributes

- unsigned char m_publickey [crypto_box_PUBLICKEYBYTES]

  the public encryption key
- unsigned char m_secretkey [crypto_box_SECRETKEYBYTES]

  the secret encryption key

Additional Inherited Members

3.7.1 Detailed Description

Class that manages encryption keys and the encryption and decryption of layers of onion messagess The class is implementing the OnionRouting abstract class with the encryption and decryption methods.

Definition at line 48 of file onionmanager.h.

3.7.2 Constructor & Destructor Documentation

3.7.2.1 OnionManager() `ns3::OnionManager::OnionManager ( )`

Default constructor.

Definition at line 41 of file onionmanager.cc.

3.7.2.2 ∼OnionManager() `ns3::OnionManager::∼OnionManager ( )`

Default destructor.

Definition at line 45 of file onionmanager.cc.

3.7.3 Member Function Documentation

3.7.3.1 DecryptLayer() `void ns3::OnionManager::DecryptLayer (`

```
unsigned char * innerLayer,
unsigned char * onion,
uint16_t onionLen,
unsigned char * pk,
unsigned char * sk ) const  [virtual]
```

Implementing decryption using the libsodium library.

Parameters

in,out	*innerLayer*	memory on which the inner onion layer will be stored
in	*onion*	memory locations containing the data to be decrypted
in	*len*	length in bytes of the `onion`
in	*pk*	pointer to the public encryption key
in	*sk*	pointer to the secret encryption key

Definition at line 65 of file onionmanager.cc.

3.7.3.2 EncryptLayer() `void ns3::OnionManager::EncryptLayer (`

```
unsigned char * ciphertext,
unsigned char * message,
int len,
unsigned char * key ) const  [virtual]
```

Implementing encryption using the libsodium library.

implement onion routing class

Parameters

`in,out`	*ciphertext*	memory on which the ciphertext will be stored
`in`	*mesage*	memory locations containing the data to be encrypted
`in`	*len*	length in bytes of the `plaintext`
`in`	*key*	pointer to the encryption key

Definition at line 55 of file onionmanager.cc.

3.7.3.3 GenerateNewKeyPair() `void ns3::OnionManager::GenerateNewKeyPair ( void )`

Generate a new public/private keypair using the libsodium library.

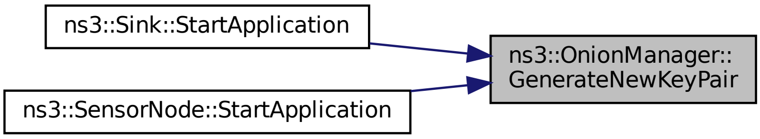
handling encryption keys

Definition at line 81 of file onionmanager.cc.

References m_publickey, and m_secretkey.

Referenced by ns3::Sink::StartApplication(), and ns3::SensorNode::StartApplication().

Here is the caller graph for this function:


ns3::Sink::StartApplication
ns3::SensorNode::StartApplication
ns3::OnionManager::
GenerateNewKeyPair

3.7.3.4 GetPK() `unsigned char * ns3::OnionManager::GetPK ( void )`

accessor

Returns

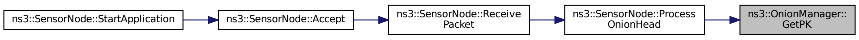
Return the pointer to the public key

Definition at line 88 of file onionmanager.cc.

References m_publickey.

Referenced by ns3::SensorNode::ProcessOnionHead().

Here is the caller graph for this function:



3.7.3.5 GetPKtoString() `std::string ns3::OnionManager::GetPKtoString ( )`

accessor

Returns

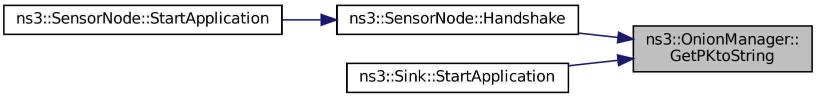
Return the public key as a string

Definition at line 103 of file onionmanager.cc.

References m_publickey.

Referenced by ns3::SensorNode::Handshake(), and ns3::Sink::StartApplication().

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Handshake
ns3::Sink::StartApplication
ns3::OnionManager::
GetPKtoString

3.7.3.6 GetSK() `unsigned char * ns3::OnionManager::GetSK ( void )`

accessor

Returns

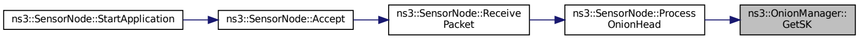
Return the pointer to the secret key

Definition at line 96 of file onionmanager.cc.

References m_secretkey.

Referenced by ns3::SensorNode::ProcessOnionHead().

Here is the caller graph for this function:



3.7.3.7 GetSKtoString() `std::string ns3::OnionManager::GetSKtoString ( )`

accessor

Returns

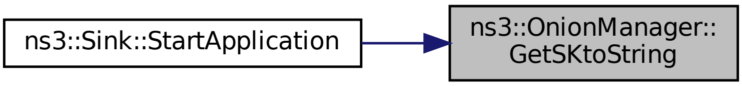
Return the secret key as a string

Definition at line 111 of file onionmanager.cc.

References m_secretkey.

Referenced by ns3::Sink::StartApplication().

Here is the caller graph for this function:



3.7.3.8 GetTypeId() `TypeId ns3::OnionManager::GetTypeId ( void ) [static]`

Register this type.

Returns

The object TypeId.

Definition at line 34 of file onionmanager.cc.

3.7.3.9 IpToBuff() `unsigned char * ns3::OnionManager::IpToBuff ( uint32_t in )`

Convert an Ipv4 address given as an unsigned integer value to buffer array of 4Bytes.

Parameters

in	*in*	an Ipv4 address given as an unsigned integer value

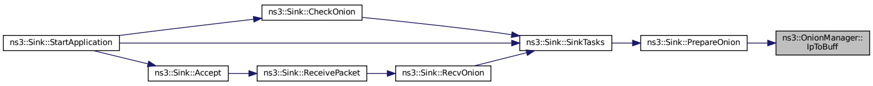
Returns

the pointer to the array of 4Bytes

Definition at line 155 of file onionmanager.cc.

Referenced by ns3::Sink::PrepareOnion().

Here is the caller graph for this function:



3.7.3.10 SetPK()

```
void ns3::OnionManager::SetPK (
    unsigned char * pk )
```

setter

Parameters

in	*pk*	pointer to the public encryption key

Definition at line 119 of file onionmanager.cc.

References m_publickey.

3.7.3.11 SetSK()

```
void ns3::OnionManager::SetSK (
    unsigned char * sk )
```

setter

Parameters

in	*pk*	pointer to the secret encryption key

Definition at line 126 of file onionmanager.cc.

References m_secretkey.

3.7.3.12 StringToUchar()

```
unsigned char * ns3::OnionManager::StringToUchar (
    std::string in )
```

Convert a string to an array of unsigned chars.

Other methods.

Parameters

`in`	*in*	a std::string

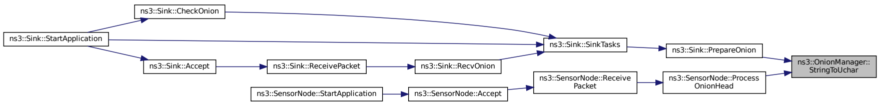
Returns

the pointer to the array of unsigned chars

Definition at line 138 of file onionmanager.cc.

Referenced by ns3::Sink::PrepareOnion(), and ns3::SensorNode::ProcessOnionHead().

Here is the caller graph for this function:



3.7.3.13 UcharToString()

```
std::string ns3::OnionManager::UcharToString (
    unsigned char * seq,
    int len )
```

Convert an array of unsigned chars to a std::string.

Parameters

`in`	*seq*	an array of unsigned chars
`in`	*len*	the length of the `seq` the array of unsigned chars

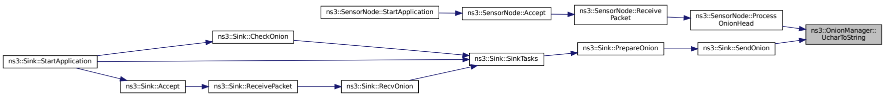
Returns

a std::string

Definition at line 147 of file onionmanager.cc.

Referenced by ns3::SensorNode::ProcessOnionHead(), and ns3::Sink::SendOnion().

Here is the caller graph for this function:



3.7.4 Member Data Documentation

3.7.4.1 m_publickey `unsigned char ns3::OnionManager::m_publickey[crypto_box_PUBLICKEYBYTES] [private]`

the public encryption key

Definition at line 192 of file onionmanager.h.

Referenced by GenerateNewKeyPair(), GetPK(), GetPKtoString(), and SetPK().

3.7.4.2 m_secretkey `unsigned char ns3::OnionManager::m_secretkey[crypto_box_SECRETKEYBYTES] [private]`

the secret encryption key

Definition at line 193 of file onionmanager.h.

Referenced by GenerateNewKeyPair(), GetSK(), GetSKtoString(), and SetSK().

The documentation for this class was generated from the following files:

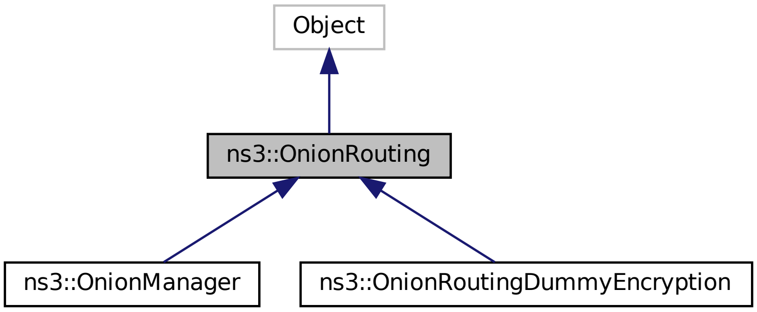
- src/onion_routing_wsn/managers/onionmanager.h
- src/onion_routing_wsn/managers/onionmanager.cc

3.8 ns3::OnionRouting Class Reference

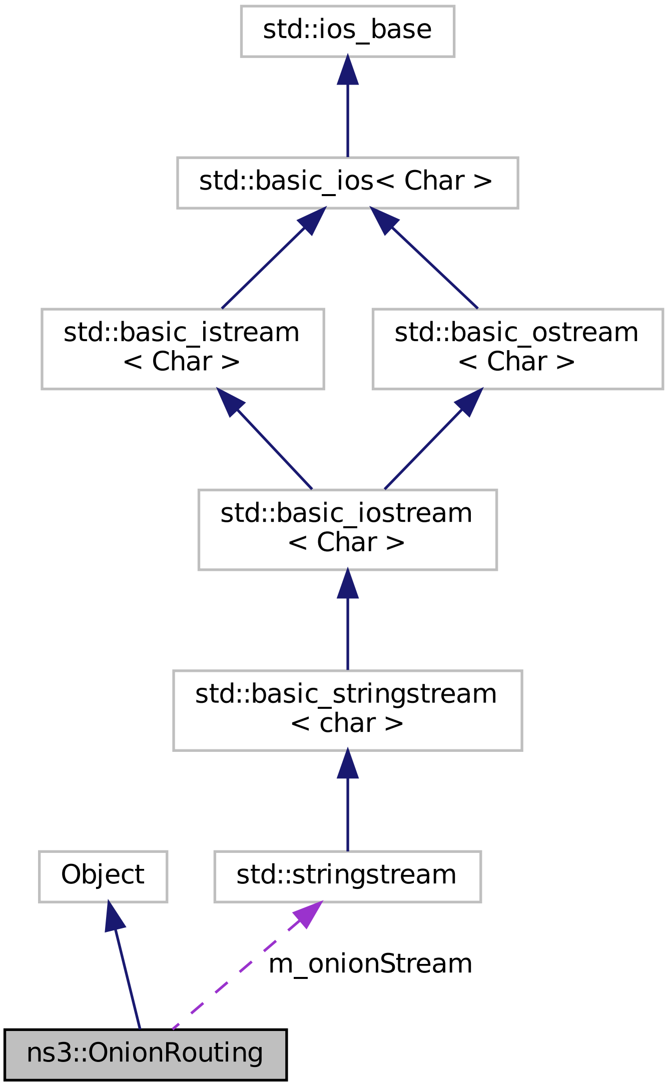
Abstract class for creation and decryption of Onion messages.

`#include "onion-routing.h"`

Inheritance diagram for ns3::OnionRouting:



Collaboration diagram for ns3::OnionRouting:


std::ios_base
std::basic_ios< Char >
std::basic_istream < Char >
std::basic_ostream < Char >
std::basic_iostream < Char >
std::basic_stringstream < char >
Object
std::stringstream
m_onionStream
ns3::OnionRouting

Public Types

- enum OnionErrno {
  ERROR_NOTERROR, ERROR_PROT_NUMBER, ERROR_ROUTE_TO_SHORT, ERROR_ENCRYPTION, ERROR_DECRYPTION }

  Enumeration of the possible errors using the class onion-routing.

Public Member Functions

- OnionRouting ()
- OnionRouting (uint16_t keySize, uint16_t sealPadding, uint16_t addressSize)

- OnionRouting (uint16_t sealPadding, const uint16_t protocolNumber)

  Constructor – Setup parameters for the creation of onions.
- virtual ∼OnionRouting ()
- void AddressToStream (uint8_t ∗ip)
- void AddressToStream (uint8_t ∗ip)

  Output an ip address to a stream variable, used to LOG the onion message.
- int BuildOnion (uint8_t ∗cipher, uint8_t ∗∗route, uint16_t routeLen, uint8_t ∗∗keys, uint8_t ∗content, uint16_t contentLen)
- void BuildOnion (uint8_t ∗cipher, uint8_t ∗∗route, uint8_t ∗∗keys, uint16_t routeLen)

  Manage construction of the onion ONION_NO_CONTENT.
- void BuildOnion (uint8_t ∗cipher, uint8_t ∗∗route, uint8_t ∗∗keys, uint16_t routeLen, uint8_t ∗endContent, uint16_t endContentLen)

  Manage construction of the onion ONION_ENDCONTENT.
- void BuildOnion (uint8_t ∗cipher, uint8_t ∗∗route, uint8_t ∗∗keys, uint8_t ∗∗layerContent, uint16_t layer←ContentLen, uint16_t routeLen)

  Manage construction of the onion ONION_LAYERCONTENT.
- void BuildOnion (uint8_t ∗cipher, uint8_t ∗∗route, uint8_t ∗∗keys, uint8_t ∗∗layerContent, uint16_t layer←ContentLen, uint16_t routeLen, uint8_t ∗endContent, uint16_t endContentLen)

  Manage construction of the onion ONION_LAYERCONTENT_ENDCONTENT.
- int CreateOnion (uint8_t ∗cipher, uint8_t ∗∗route, uint16_t index, uint16_t routeLen, uint8_t ∗∗keys, uint8_t ∗content, uint16_t contentLen)
- void CreateOnion (uint8_t ∗cipher, uint8_t ∗∗route, uint8_t ∗∗keys, uint16_t index, uint16_t routeLen, uint8_t ∗∗layerContent, uint16_t layerContentLen, uint8_t ∗endContent, uint16_t endContentLen)

  Constructs the onion message ∗.
- virtual void DecryptLayer (uint8_t ∗innerLayer, uint8_t ∗onion, uint16_t onionLen, uint8_t ∗pk, uint8_t ∗sk) const =0
- virtual void DecryptLayer (uint8_t ∗plaintext, uint8_t ∗ciphertext, uint16_t len, uint8_t ∗publicKey, uint8_←t ∗secretKey) const =0

  virtual method, implement decryption
- virtual void EncryptLayer (uint8_t ∗ciphertext, uint8_t ∗message, int len, uint8_t ∗key) const =0
- virtual void EncryptLayer (uint8_t ∗ciphertext, uint8_t ∗plaintext, int len, uint8_t ∗key) const =0

  virtual method, implement encryption
- enum OnionErrno GetErrno (void)

  Return the last error code of the OnionErrno enum.
- uint16_t OnionLength (uint16_t routeLen, uint16_t contentLen)
- uint16_t OnionLength (uint16_t routeLen, uint16_t layerContentLen, uint16_t endContentLen)

  Compute the length in bytes of the onion message at given parameters.
- orLayer ∗ PeelOnion (uint8_t ∗onion, uint16_t onionLen, uint8_t ∗pk, uint8_t ∗sk)
- orLayer ∗ PeelOnion (uint8_t ∗onion, uint16_t onionLen, uint8_t ∗publicKey, uint8_t ∗secretKey)

  Decipher the outer layer of the onion and return details.

Static Public Member Functions

- static TypeId GetTypeId (void)
- static TypeId GetTypeId (void)

  Register this type.

Public Attributes

- uint16_t m_addressSize

  size in bytes of the used address type (4-Ipv4, 16-Ipv6)
- enum OnionErrno m_errno

  error status while using the onion class
- uint16_t m_keySize
- std::stringstream m_onionStream

  stringstream used to LOG onion construction
- uint16_t m_sealPadding

  size increase of the ciphertext in bytes, intorduced by the encryption method

3.8.1 Detailed Description

Abstract class for creation and decryption of Onion messages.

The OnionRouting abstract class include useful methods for the creation and redirection of `Onion Messages`. The given class can be used to construct onion messages of the following features:
ONION_NO_CONTENT - onion message including only routing information
example: (((10.1.1.2) 10.1.1.1) 10.1.1.5)10.1.1.3

ONION_ENDCONTENT - onion message including content to be delivered to the last node in the path
example: ((((end_content,0.0.0.0) 10.1.1.2) 10.1.1.1) 10.1.1.5) 10.1.1.3

ONION_LAYERCONTENT - onion message including a content of fixed length (in bytes) in each layer
example: ((((layer_content,0.0.0.0) layer_content,10.1.1.2) layer_content,10.1.1.1) layer_content,10.1.1.5)10.1.1.3

ONION_LAYERCONTENT_ENDCONTENT - onion message including a content of fixed length in each layer and content of arbitrary length to be delivered to the last node in the path
example: ((((end_content,0.0.0.0) layer_content,10.1.1.2) layer_content,10.1.1.1) layer_content,10.1.1.5)10.1.1.3

The given class can be used to construct onion circuits as described in *Hiding Routing Information* by david M. Goldschlag, Micheal G. Reed, and Paul F. Syverson, May 1996

We designed an abstract class to allow the use of an arbitrary encryption suite, by implementing methods EncryptLayer & DecryptLayer

implementing dummy Encryption/Decryption methods.

The class simulates the use of encryption keys by including them into encryption layers of onion messages. A node deciphering a layer of the onion message will compare its encryption key with the encryption key included in the layer of the onion message. If the two keys match the layer is succesfully deciphered, otherwise the node is not the expected recipient of the onion message and the encryption will fail triggering an error message.

Since dummy encryption keys of 4B are included in each layer of the onion message, the parameter `m_sealPadding` must be set to at least 4 Bytes. The parameter `m_sealPadding` is used to emulate additional bytes introduced by a real encryption technique. This parameter is set in the constructor.

Definition at line 29 of file onion-routing.h.

3.8.2 Member Enumeration Documentation

3.8.2.1 OnionErrno `enum ns3::OnionRouting::OnionErrno`

Enumeration of the possible errors using the class onion-routing.

Enumerator

ERROR_NOTERROR	
ERROR_PROT_NUMBER	
ERROR_ROUTE_TO_SHORT	
ERROR_ENCRYPTION	
ERROR_DECRYPTION	

Definition at line 104 of file onion-routing.h.

3.8.3 Constructor & Destructor Documentation

3.8.3.1 OnionRouting() `[1/3]` `ns3::OnionRouting::OnionRouting ( )`

Definition at line 27 of file onion-routing.cc.

3.8.3.2 OnionRouting() `[2/3]`

```
ns3::OnionRouting::OnionRouting (
        uint16_t keySize,
        uint16_t sealPadding,
        uint16_t addressSize )
```

Definition at line 43 of file onion-routing.cc.

References m_addressSize, m_keySize, and m_sealPadding.

3.8.3.3 ∼OnionRouting() `ns3::OnionRouting::∼OnionRouting ( ) [virtual]`

Definition at line 50 of file onion-routing.cc.

3.8.3.4 OnionRouting() `[3/3]`

```
ns3::OnionRouting::OnionRouting (
        uint16_t sealPadding,
        const uint16_t protocolNumber )
```

Constructor – Setup parameters for the creation of onions.

Parameters

in	*sealPadding*	size increase of the ciphertext in bytes, intorduced by the encryption method
in	*protocolNumber*	value detailing the utilized IP protocol: IPv4–Ipv4L3Protocol::PROT_NUMBER, IPv6–Ipv6L3Protocol::PROT_NUMBER

Definition at line 51 of file onion-routing.cc.

References ERROR_NOTERROR, ERROR_PROT_NUMBER, m_addressSize, m_errno, and m_sealPadding.

3.8.4 Member Function Documentation

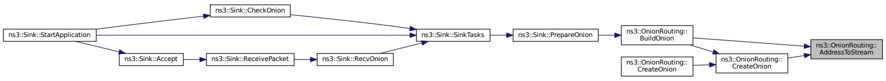
3.8.4.1 AddressToStream() [1/2] void ns3::OnionRouting::AddressToStream (uint8_t * *ip*)

Definition at line 200 of file onion-routing.cc.

References m_addressSize, and m_onionStream.

Referenced by BuildOnion(), and CreateOnion().

Here is the caller graph for this function:


ns3::Sink::CheckOnion
ns3::Sink::StartApplication
ns3::Sink::SinkTasks
ns3::Sink::PrepareOnion
ns3::OnionRouting::BuildOnion
ns3::OnionRouting::AddressToStream
ns3::Sink::Accept
ns3::Sink::ReceivePacket
ns3::Sink::RecvOnion
ns3::OnionRouting::CreateOnion
ns3::OnionRouting::CreateOnion

3.8.4.2 AddressToStream() [2/2] void ns3::OnionRouting::AddressToStream (uint8_t * *ip*)

Output an ip address to a stream variable, used to LOG the onion message.

Parameters

in	*ip*	serialized ip address

3.8.4.3 BuildOnion() [1/5] int ns3::OnionRouting::BuildOnion (

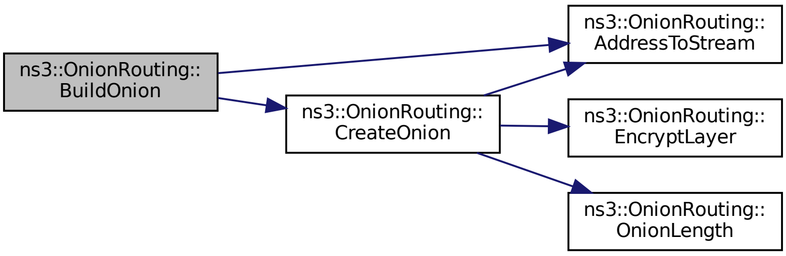
```
uint8_t * cipher,
uint8_t ** route,
uint16_t routeLen,
uint8_t ** keys,
uint8_t * content = nullptr,
uint16_t contentLen = 0 )
```

Definition at line 63 of file onion-routing.cc.

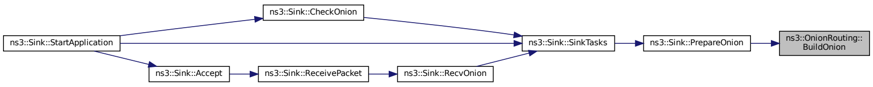
References AddressToStream(), CreateOnion(), and m_onionStream.

Referenced by ns3::Sink::PrepareOnion().

Here is the call graph for this function:



Here is the caller graph for this function:



3.8.4.4 BuildOnion() [2/5]

```
void ns3::OnionRouting::BuildOnion (
            uint8_t * cipher,
            uint8_t ** route,
            uint8_t ** keys,
            uint16_t routeLen )
```

Manage construction of the onion ONION_NO_CONTENT.

The resulting onion message include only routing information and the last hop in the onion path will not recieve content example: (((10.1.1.2) 10.1.1.1) 10.1.1.5)10.1.1.3

Allow the construction of onions of route length $>$ 4, and manage the LOG output

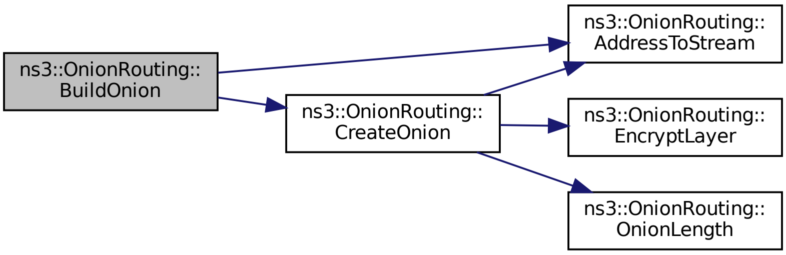
Parameters

`in,out`	*cipher*	memory locations on which the onion message will be stored
`in`	*route*	array of ip addresses defining the route of the onion message, ip addresses are stored in the serialized form
`in`	*keys*	array of encryption keys, keys are stored in the serialized form
`in`	*routeLen*	the length of the route that the onion message will travel (equal to the number of ip addresses stored in the route parameter)

Definition at line 77 of file onion-routing.cc.

References AddressToStream(), CreateOnion(), ERROR_NOTERROR, ERROR_ROUTE_TO_SHORT, m_errno, and m_onionStream.

Here is the call graph for this function:



3.8.4.5 BuildOnion() [3/5] `void ns3::OnionRouting::BuildOnion (`

```
uint8_t * cipher,
uint8_t ** route,
uint8_t ** keys,
uint16_t routeLen,
uint8_t * endContent,
uint16_t endContentLen )
```

Manage construction of the onion ONION_ENDCONTENT.

The resulting onion message include routing information and the last hop in the onion path recieve the given content
The zero address – 0.0.0.0 (ipv4) identifies the last hop in the path of the onion message
example: ((((end_content,0.0.0.0) 10.1.1.2) 10.1.1.1) 10.1.1.5) 10.1.1.3

Allow the construction of onions of route length $>$ 4, and manage the LOG output

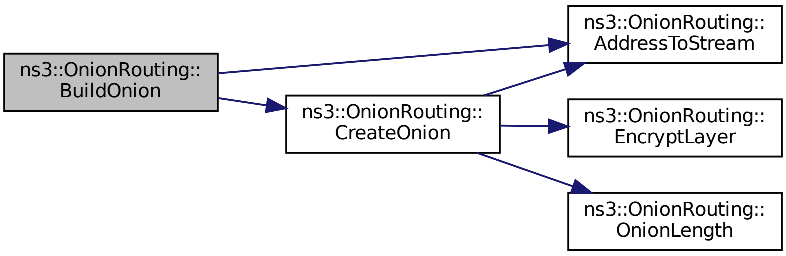
Parameters

in,out	*cipher*	memory locations on which the onion message will be stored
in	*route*	array of ip addresses defining the route of the onion message, ip addresses are stored in the serialized form
in	*keys*	array of encryption keys, keys are stored in the serialized form
in	*routeLen*	the length of the route that the onion message will travel (equal to the number of ip addresses stored in the `route`)
in	*endContent*	location of the content to forward to the last node in the onion message path
in	*endContentLen*	length in bytes of the data stored at `endContent`

Definition at line 105 of file onion-routing.cc.

References AddressToStream(), CreateOnion(), ERROR_NOTERROR, ERROR_ROUTE_TO_SHORT, m_errno, and m_onionStream.

Here is the call graph for this function:



3.8.4.6 BuildOnion() [4/5]

```
void ns3::OnionRouting::BuildOnion (
        uint8_t * cipher,
        uint8_t ** route,
        uint8_t ** keys,
        uint8_t ** layerContent,
        uint16_t layerContentLen,
        uint16_t routeLen )
```

Manage construction of the onion ONION_LAYERCONTENT.

The resulting onion message include only routing information, and each hop in the route will receive data stored in `layerContent`
The zero address – 0.0.0.0 (ipv4) identifies the last hop in the path of the onion message
example: ((((layer_content,0.0.0.0) layer_content,10.1.1.2) layer_content,10.1.1.1) layer_content,10.1.1.5)10.1.1.3

Allow the construction of onions of route length $>$ 4, and manage the LOG output

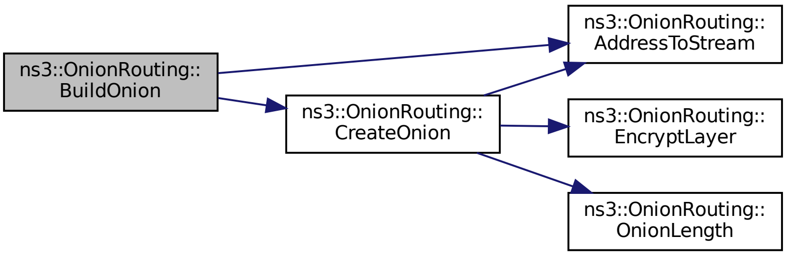
Parameters

in,out	*cipher*	memory locations on which the onion message will be stored
in	*route*	array of ip addresses defining the route of the onion message, ip addresses are stored in the serialized form
in	*keys*	array of encryption keys, keys are stored in the serialized form
in	*layerContent*	array of of pointers, pointing to data to be stored in a layer of the onion message the data is of fixed length in bytes
in	*layerContentLen*	length in bytes of the data to be stored in each layer of the onion message
in	*routeLen*	the length of the route that the onion message will travel (equal to the number of ip addresses stored in the `route`)

Definition at line 133 of file onion-routing.cc.

References AddressToStream(), CreateOnion(), ERROR_NOTERROR, ERROR_ROUTE_TO_SHORT, m_errno, and m_onionStream.

Here is the call graph for this function:



3.8.4.7 BuildOnion() [5/5]

```
void ns3::OnionRouting::BuildOnion (
        uint8_t * cipher,
        uint8_t ** route,
        uint8_t ** keys,
        uint8_t ** layerContent,
        uint16_t layerContentLen,
        uint16_t routeLen,
        uint8_t * endContent,
        uint16_t endContentLen )
```

Manage construction of the onion ONION_LAYERCONTENT_ENDCONTENT.

The resulting onion message include only routing information, each hop in the route will receive data stored in `layerContent`, the last hop in the onion path recieve content only data stored in `endContent`
The zero address – 0.0.0.0 (ipv4) identifies the last hop in the path of the onion message
example: ((((end_content,0.0.0.0) layer_content,10.1.1.2) layer_content,10.1.1.1) layer_content,10.1.1.5)10.1.1.3

Allow the construction of onions of route length > 4, and manage the LOG output

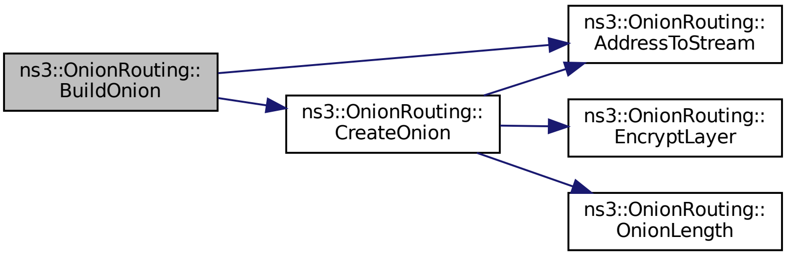
Parameters

`in,out`	*cipher*	memory locations on which the onion message will be stored
`in`	*route*	array of ip addresses defining the route of the onion message, ip addresses are stored in the serialized form
`in`	*keys*	array of encryption keys, keys are stored in the serialized form
`in`	*layerContent*	array of of pointers, pointing to data to be stored in a layer of the onion message the data is of fixed length in bytes
`in`	*layerContentLen*	length in bytes of the data to be stored in each layer of the onion message
`in`	*routeLen*	the length of the route that the onion message will travel (equal to the number of ip addresses stored in the `route`)
`in`	*endContent*	location of the content to forward to the last node in the onion message path
`in`	*endContentLen*	length in bytes of the data stored at `endContent`

Definition at line 160 of file onion-routing.cc.

References AddressToStream(), CreateOnion(), ERROR_NOTERROR, ERROR_ROUTE_TO_SHORT, m_errno, and m_onionStream.

Here is the call graph for this function:



3.8.4.8 CreateOnion() [1/2]

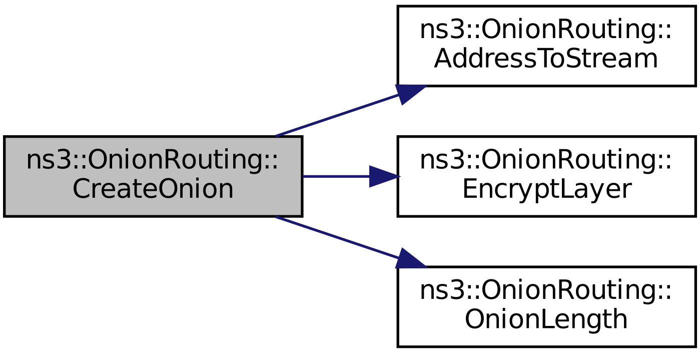
```
int ns3::OnionRouting::CreateOnion (
        uint8_t * cipher,
        uint8_t ** route,
        uint16_t index,
        uint16_t routeLen,
        uint8_t ** keys,
        uint8_t * content = nullptr,
        uint16_t contentLen = 0 )
```

Definition at line 103 of file onion-routing.cc.

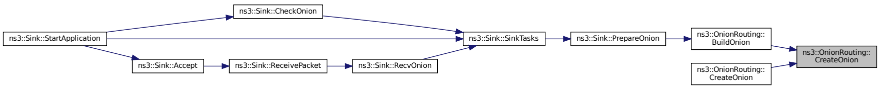
References AddressToStream(), EncryptLayer(), m_addressSize, m_onionStream, m_sealPadding, and OnionLength().

Referenced by BuildOnion(), and CreateOnion().

Here is the call graph for this function:



Here is the caller graph for this function:



3.8.4.9 CreateOnion() [2/2]

```
void ns3::OnionRouting::CreateOnion (
        uint8_t * cipher,
        uint8_t ** route,
        uint8_t ** keys,
        uint16_t index,
        uint16_t routeLen,
        uint8_t ** layerContent,
        uint16_t layerContentLen,
        uint8_t * endContent,
        uint16_t endContentLen )
```

Constructs the onion message *.

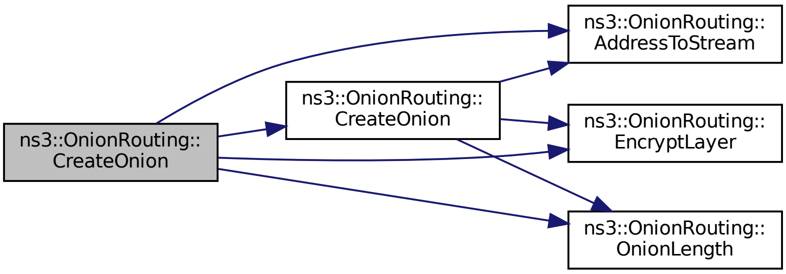
Parameters

in,out	*cipher*	memory on which the onion message will be stored
in	*route*	array of ip addresses defining the route of the onion message, ip addresses are stored in the serialized form
in	*keys*	array of encryption keys, keys are stored in the serialized form
in	*layerContent*	array of of pointers, pointing to data to be stored in a layer of the onion message the data is of fixed length in bytes
in	*layerContentLen*	length in bytes of the data to be stored in each layer of the onion message
in	*index*	additional parameter used for the construction of the onion message
in	*routeLen*	the length of the route that the onion message will travel (equal to the number of ip addresses stored in the `route`)
in	*endContent*	location of the content to forward to the last node in the onion message path
in	*endContentLen*	length in bytes of the data stored at `endContent`

Definition at line 185 of file onion-routing.cc.

References AddressToStream(), CreateOnion(), EncryptLayer(), m_addressSize, m_onionStream, m_sealPadding, and OnionLength().

Here is the call graph for this function:



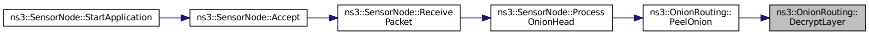
3.8.4.10 DecryptLayer() `[1/2]`

```
virtual void ns3::OnionRouting::DecryptLayer (
        uint8_t * innerLayer,
        uint8_t * onion,
        uint16_t onionLen,
        uint8_t * pk,
        uint8_t * sk ) const  [pure virtual]
```

Implemented in ns3::OnionRoutingDummyEncryption.

Referenced by PeelOnion().

Here is the caller graph for this function:



3.8.4.11 DecryptLayer() `[2/2]`

```
virtual void ns3::OnionRouting::DecryptLayer (
        uint8_t * plaintext,
        uint8_t * ciphertext,
        uint16_t len,
        uint8_t * publicKey,
        uint8_t * secretKey ) const  [pure virtual]
```

virtual method, implement decryption

Parameters

`in,out`	*plaintext*	memory locations containing the decrypted data
`in`	*ciphertext*	memory locations containing the encrypted data
`in`	*len*	length in bytes of the `ciphertext`
`in`	*publicKey*	encryption key
`in`	*secretKey*	encryption key

Implemented in ns3::OnionRoutingDummyEncryption.

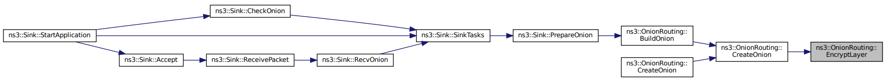
3.8.4.12 EncryptLayer() [1/2]

```
virtual void ns3::OnionRouting::EncryptLayer (
        uint8_t * ciphertext,
        uint8_t * message,
        int len,
        uint8_t * key ) const  [pure virtual]
```

Implemented in ns3::OnionRoutingDummyEncryption.

Referenced by CreateOnion().

Here is the caller graph for this function:


ns3::Sink::CheckOnion
ns3::Sink::StartApplication
ns3::Sink::SinkTasks
ns3::Sink::PrepareOnion
ns3::OnionRouting::BuildOnion
ns3::OnionRouting::CreateOnion
ns3::OnionRouting::EncryptLayer
ns3::OnionRouting::CreateOnion
ns3::Sink::Accept
ns3::Sink::ReceivePacket
ns3::Sink::RecvOnion

3.8.4.13 EncryptLayer() [2/2]

```
virtual void ns3::OnionRouting::EncryptLayer (
        uint8_t * ciphertext,
        uint8_t * plaintext,
        int len,
        uint8_t * key ) const  [pure virtual]
```

virtual method, implement encryption

Parameters

in,out	*ciphertext*	memory on which the ciphertext will be stored
in	*plaintext*	memory locations containing the data to be encrypted
in	*len*	length in bytes of the `plaintext`
in	*key*	encryption key

Implemented in ns3::OnionRoutingDummyEncryption.

3.8.4.14 GetErrno()

```
enum OnionRouting::OnionErrno ns3::OnionRouting::GetErrno (
        void  )
```

Return the last error code of the OnionErrno enum.

Returns

OnionErrno enum, if != 0 THEN signals ERROR

Definition at line 292 of file onion-routing.cc.

References m_errno.

3.8.4.15 GetTypeId() [1/2] `TypeId ns3::OnionRouting::GetTypeId ( void ) [static]`

Definition at line 17 of file onion-routing.cc.

3.8.4.16 GetTypeId() [2/2] `static TypeId ns3::OnionRouting::GetTypeId ( void ) [static]`

Register this type.

Returns

The object TypeId.

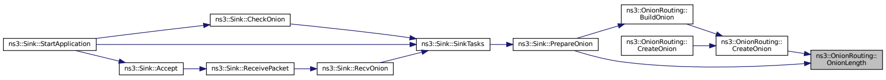
3.8.4.17 OnionLength() [1/2] `uint16_t ns3::OnionRouting::OnionLength ( uint16_t routeLen, uint16_t contentLen )`

Definition at line 180 of file onion-routing.cc.

References m_addressSize, and m_sealPadding.

Referenced by CreateOnion(), and ns3::Sink::PrepareOnion().

Here is the caller graph for this function:


ns3::Sink::StartApplication
ns3::Sink::CheckOnion
ns3::Sink::Accept
ns3::Sink::ReceivePacket
ns3::Sink::RecvOnion
ns3::Sink::SinkTasks
ns3::Sink::PrepareOnion
ns3::OnionRouting::BuildOnion
ns3::OnionRouting::CreateOnion
ns3::OnionRouting::CreateOnion
ns3::OnionRouting::OnionLength

3.8.4.18 OnionLength() [2/2] `uint16_t ns3::OnionRouting::OnionLength ( uint16_t routeLen, uint16_t layerContentLen, uint16_t endContentLen )`

Compute the length in bytes of the onion message at given parameters.

Parameters

in	*routeLen*	the length of the route that the onion message will travel (equal to the number of ip addresses stored in the `route`)
in	*layerContentLen*	length in bytes of the data to be stored in layers of the onion message
in	*endContentLen*	length in bytes of the data stored in the last hop's layer of the onion message

Returns

an integer detailing the length in bytes of the onion message at given parameters

Definition at line 259 of file onion-routing.cc.

References m_addressSize, and m_sealPadding.

3.8.4.19 PeelOnion() [1/2]

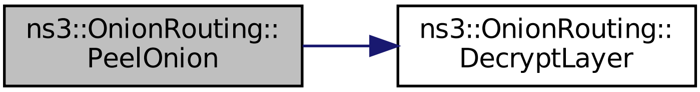
```
orLayer * ns3::OnionRouting::PeelOnion (
        uint8_t * onion,
        uint16_t onionLen,
        uint8_t * pk,
        uint8_t * sk )
```

Definition at line 158 of file onion-routing.cc.

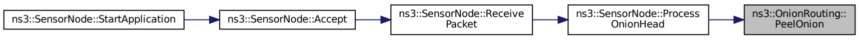
References DecryptLayer(), ns3::orLayer::innerLayer, ns3::orLayer::innerLayerLen, m_addressSize, m_sealPadding, and ns3::orLayer::nextHopIP.

Referenced by ns3::SensorNode::ProcessOnionHead().

Here is the call graph for this function:



Here is the caller graph for this function:



3.8.4.20 PeelOnion() [2/2] orLayer* ns3::OnionRouting::PeelOnion (uint8_t * *onion*, uint16_t *onionLen*, uint8_t * *publicKey*, uint8_t * *secretKey*)

Decipher the outer layer of the onion and return details.

Parameters

in	*onion*	the onion message
in	*onionLen*	the length in bytes of the onion message
in	*publicKey*	encryption key
in	*secretKey*	encryption key

Returns

orLayer * struct holding onion layer details

3.8.5 Member Data Documentation

3.8.5.1 m_addressSize uint16_t ns3::OnionRouting::m_addressSize

size in bytes of the used address type (4-Ipv4, 16-Ipv6)

Definition at line 52 of file onion-routing.h.

Referenced by AddressToStream(), CreateOnion(), OnionLength(), OnionRouting(), and PeelOnion().

3.8.5.2 m_errno enum OnionErrno ns3::OnionRouting::m_errno [mutable]

error status while using the onion class

Definition at line 320 of file onion-routing.h.

Referenced by BuildOnion(), ns3::OnionRoutingDummyEncryption::DecryptLayer(), GetErrno(), and OnionRouting().

3.8.5.3 m_keySize uint16_t ns3::OnionRouting::m_keySize

Definition at line 51 of file onion-routing.h.

Referenced by OnionRouting().

3.8.5.4 m_onionStream `std::stringstream ns3::OnionRouting::m_onionStream`

stringstream used to LOG onion construction

Definition at line 56 of file onion-routing.h.

Referenced by AddressToStream(), BuildOnion(), and CreateOnion().

3.8.5.5 m_sealPadding `uint16_t ns3::OnionRouting::m_sealPadding`

size increase of the ciphertext in bytes, intorduced by the encryption method

Definition at line 50 of file onion-routing.h.

Referenced by CreateOnion(), ns3::OnionRoutingDummyEncryption::DecryptLayer(), ns3::OnionRoutingDummy↩ Encryption::EncryptLayer(), OnionLength(), OnionRouting(), and PeelOnion().

The documentation for this class was generated from the following files:

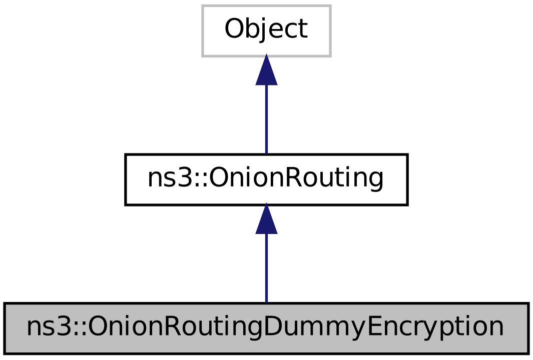
- src/onion-routing/model/bckp/onion-routing.h
- src/onion-routing/model/bckp/onion-routing.cc

3.9 ns3::OnionRoutingDummyEncryption Class Reference

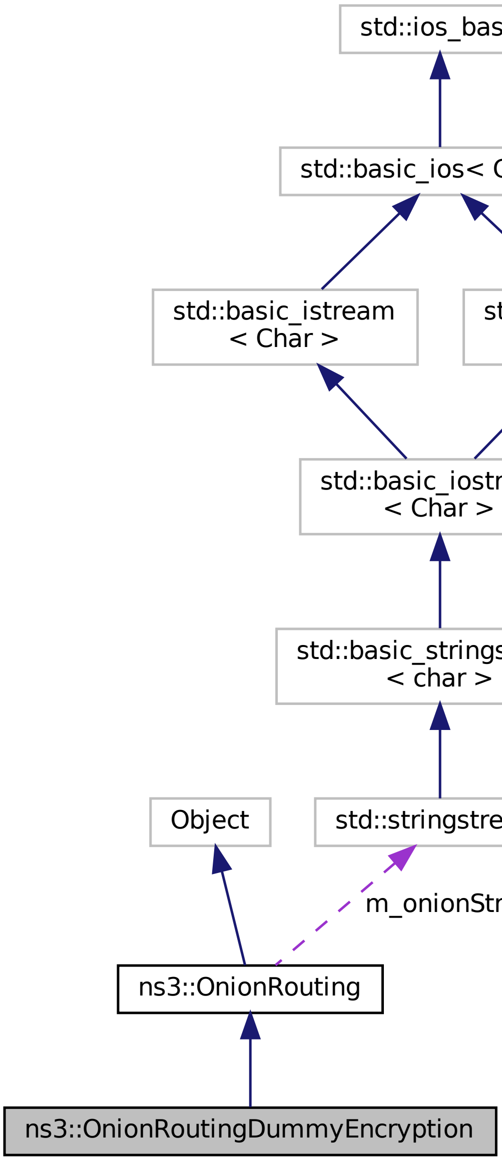
class that implements the

```
#include "onion-routing.h"
```

Inheritance diagram for ns3::OnionRoutingDummyEncryption:



Collaboration diagram for ns3::OnionRoutingDummyEncryption:



Public Member Functions

- OnionRoutingDummyEncryption (uint16_t sealPadding, const uint16_t protocolNumber)

    Constructor – Setup parameters for the creation of onions and check that `sealPadding` is greter than 4 Bytes.
- virtual void DecryptLayer (uint8_t ∗innerLayer, uint8_t ∗onion, uint16_t onionLen, uint8_t ∗pk, uint8_t ∗sk) const
- virtual void EncryptLayer (uint8_t ∗ciphertext, uint8_t ∗message, int len, uint8_t ∗key) const
- void GenerateNewKey (void)

    Generate a new dummy encryption key of 4Bytes using the uniform random generator.
- uint8_t ∗ GetEncryptionKey (void)

    Return the current encryption key.

Static Public Member Functions

- static TypeId GetTypeId (void)

  Register this type.

Public Attributes

- uint8_t m_encryptionkey [4]

  the current encryption key

Additional Inherited Members

3.9.1 Detailed Description

class that implements the

Definition at line 339 of file onion-routing.h.

3.9.2 Constructor & Destructor Documentation

3.9.2.1 OnionRoutingDummyEncryption() `ns3::OnionRoutingDummyEncryption::OnionRoutingDummy←Encryption (`

```
            uint16_t sealPadding,
            const uint16_t protocolNumber )
```

Constructor – Setup parameters for the creation of onions and check that `sealPadding` is greter than 4 Bytes.

Parameters

in	*sealPadding*	size increase of the ciphertext in bytes, intorduced by the simulated encryption method
in	*protocolNumber*	value detailing the utilized IP protocol: IPv4–Ipv4L3Protocol::PROT_NUMBER, IPv6–Ipv6L3Protocol::PROT_NUMBER

Definition at line 303 of file onion-routing.cc.

3.9.3 Member Function Documentation

3.9.3.1 DecryptLayer() `void ns3::OnionRoutingDummyEncryption::DecryptLayer (`

```
            uint8_t * innerLayer,
            uint8_t * onion,
            uint16_t onionLen,
            uint8_t * pk,
            uint8_t * sk ) const  [virtual]
```

Implements ns3::OnionRouting.

Definition at line 349 of file onion-routing.cc.

References ns3::OnionRouting::ERROR_DECRYPTION, ns3::OnionRouting::m_errno, and ns3::OnionRouting↩ ::m_sealPadding.

3.9.3.2 EncryptLayer()

```
void ns3::OnionRoutingDummyEncryption::EncryptLayer (
            uint8_t * ciphertext,
            uint8_t * message,
            int len,
            uint8_t * key ) const  [virtual]
```

Implements ns3::OnionRouting.

Definition at line 338 of file onion-routing.cc.

References ns3::OnionRouting::m_sealPadding.

3.9.3.3 GenerateNewKey()

```
void ns3::OnionRoutingDummyEncryption::GenerateNewKey (
            void  )
```

Generate a new dummy encryption key of 4Bytes using the uniform random generator.

Definition at line 323 of file onion-routing.cc.

References m_encryptionkey.

3.9.3.4 GetEncryptionKey()

```
uint8_t * ns3::OnionRoutingDummyEncryption::GetEncryptionKey (
            void  )
```

Return the current encryption key.

Returns

the encryption key in the form of a 4 bytes uint8_t array

Definition at line 331 of file onion-routing.cc.

References m_encryptionkey.

3.9.3.5 GetTypeId() `TypeId ns3::OnionRoutingDummyEncryption::GetTypeId ( void ) [static]`

Register this type.

Returns

The object TypeId.

Definition at line 313 of file onion-routing.cc.

3.9.4 Member Data Documentation

3.9.4.1 m_encryptionkey `uint8_t ns3::OnionRoutingDummyEncryption::m_encryptionkey[4]`

the current encryption key

Definition at line 381 of file onion-routing.h.

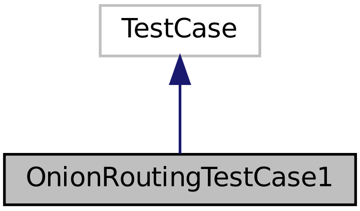
Referenced by GenerateNewKey(), and GetEncryptionKey().

The documentation for this class was generated from the following files:

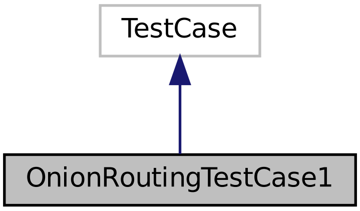
- src/onion-routing/model/onion-routing.h
- src/onion-routing/model/onion-routing.cc

3.10 OnionRoutingTestCase1 Class Reference

Inheritance diagram for OnionRoutingTestCase1:



Collaboration diagram for OnionRoutingTestCase1:



Public Member Functions

- OnionRoutingTestCase1 ()
- virtual ∼OnionRoutingTestCase1 ()

Private Member Functions

- virtual void DoRun (void)

3.10.1 Detailed Description

Definition at line 14 of file onion-routing-test-suite.cc.

3.10.2 Constructor & Destructor Documentation

3.10.2.1 OnionRoutingTestCase1() `OnionRoutingTestCase1::OnionRoutingTestCase1 ( )`

Definition at line 25 of file onion-routing-test-suite.cc.

3.10.2.2 ∼OnionRoutingTestCase1() `OnionRoutingTestCase1::∼OnionRoutingTestCase1 ( ) [virtual]`

Definition at line 32 of file onion-routing-test-suite.cc.

3.10.3 Member Function Documentation

3.10.3.1 DoRun() `void OnionRoutingTestCase1::DoRun ( void ) [private], [virtual]`

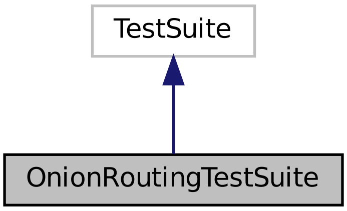
Definition at line 41 of file onion-routing-test-suite.cc.

The documentation for this class was generated from the following file:

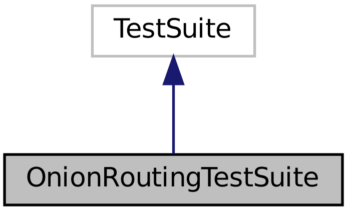
- src/onion-routing/test/onion-routing-test-suite.cc

3.11 OnionRoutingTestSuite Class Reference

Inheritance diagram for OnionRoutingTestSuite:



Collaboration diagram for OnionRoutingTestSuite:



Public Member Functions

- OnionRoutingTestSuite ()

3.11.1 Detailed Description

Definition at line 53 of file onion-routing-test-suite.cc.

3.11.2 Constructor & Destructor Documentation

3.11.2.1 OnionRoutingTestSuite() `OnionRoutingTestSuite::OnionRoutingTestSuite ( )`

Definition at line 59 of file onion-routing-test-suite.cc.

The documentation for this class was generated from the following file:

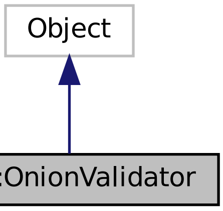
- src/onion-routing/test/onion-routing-test-suite.cc

3.12 ns3::OnionValidator Class Reference

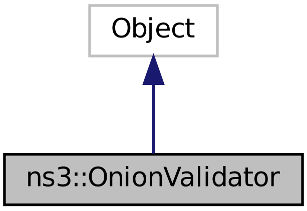
Class shared between wsn nodes used to track how the onion is transiting in the WSN The class uses two counters `m_onionSeq` and `m_hopCount` to identify when an onion neeeds to be aborted.

`#include "onionvalidator.h"`

Inheritance diagram for ns3::OnionValidator:



Collaboration diagram for ns3::OnionValidator:



Public Member Functions

- OnionValidator ()

  Default constructor.
- ∼OnionValidator ()

  Default destructor.
- bool CheckOnionReceived (int hop)

  Called by the sensor nodes when the `m_onionTimeout` elapses IF `m_hopCount` is equal to `hop` then the onion was not received by the next hop Therefore abort the onion.
- int GetOnionSeq (void)

  Return the current onion sequence number (the onionId)
- int OnionHopCount (void)

  Return the current hop sequence number.
- void OnionReceived (void)

  The onion was correctly received, increment the hop count.
- bool OnionStatus (void)

  Called by the sink node to check if the onion is still running or is aborted.
- void StartOnion (int seq_n)

  Called by the sink node, set up the `m_onionSeq` to the onionId value for keeping track of the onion.

Static Public Member Functions

- static TypeId GetTypeId (void)

Public Attributes

- int m_hopCount

  a sequence number incremented each time a node correctly receives an onion message
- int m_onionSeq = 0

  the onionId of the onion currently transiting the network

3.12.1 Detailed Description

Class shared between wsn nodes used to track how the onion is transiting in the WSN The class uses two counters `m_onionSeq` and `m_hopCount` to identify when an onion neeeds to be aborted.

Definition at line 55 of file onionvalidator.h.

3.12.2 Constructor & Destructor Documentation

3.12.2.1 OnionValidator() `ns3::OnionValidator::OnionValidator ( )`

Default constructor.

Definition at line 41 of file onionvalidator.cc.

3.12.2.2 ∼OnionValidator() `ns3::OnionValidator::∼OnionValidator ( )`

Default destructor.

Definition at line 45 of file onionvalidator.cc.

3.12.3 Member Function Documentation

3.12.3.1 CheckOnionReceived() `bool ns3::OnionValidator::CheckOnionReceived ( int hop )`

Called by the sensor nodes when the `m_onionTimeout` elapses IF `m_hopCount` is equal to `hop` then the onion was not received by the next hop Therefore abort the onion.

Parameters

`in`	*hop*	the `m_hopCount` number set when the onion was forwarded to the next node

Returns

TRUE if the onion is running, FALSE if the onion was aborted

Definition at line 78 of file onionvalidator.cc.

References m_hopCount, and m_onionSeq.

3.12.3.2 GetOnionSeq() `int ns3::OnionValidator::GetOnionSeq ( void )`

Return the current onion sequence number (the onionId)

Returns

return the onion sequence number

Definition at line 99 of file onionvalidator.cc.

References m_onionSeq.

3.12.3.3 GetTypeId() `TypeId ns3::OnionValidator::GetTypeId ( void ) [static]`

Returns

The object TypeId.

Definition at line 34 of file onionvalidator.cc.

3.12.3.4 OnionHopCount() `int ns3::OnionValidator::OnionHopCount ( void )`

Return the current hop sequence number.

Returns

return the hop count

Definition at line 92 of file onionvalidator.cc.

References m_hopCount.

3.12.3.5 OnionReceived() `void ns3::OnionValidator::OnionReceived ( void )`

The onion was correctly received, increment the hop count.

Definition at line 106 of file onionvalidator.cc.

References m_hopCount.

3.12.3.6 OnionStatus() `bool ns3::OnionValidator::OnionStatus ( void )`

Called by the sink node to check if the onion is still running or is aborted.

Returns

TRUE if the onion is running, FALSE if the onion was aborted

Definition at line 59 of file onionvalidator.cc.

References m_onionSeq.

3.12.3.7 StartOnion() `void ns3::OnionValidator::StartOnion ( int seq_n )`

Called by the sink node, set up the `m_onionSeq` to the onionId value for keeping track of the onion.

Parameters

in	*seq_n*	the onion ID

Definition at line 52 of file onionvalidator.cc.

References m_onionSeq.

3.12.4 Member Data Documentation

3.12.4.1 m_hopCount `int ns3::OnionValidator::m_hopCount`

Initial value:

```
=
      0
```

a sequence number incremented each time a node correctly receives an onion message

Definition at line 133 of file onionvalidator.h.

Referenced by CheckOnionReceived(), OnionHopCount(), and OnionReceived().

3.12.4.2 m_onionSeq `int ns3::OnionValidator::m_onionSeq = 0`

the onionId of the onion currently transiting the network

Definition at line 131 of file onionvalidator.h.

Referenced by CheckOnionReceived(), GetOnionSeq(), OnionStatus(), and StartOnion().

The documentation for this class was generated from the following files:

- src/onion_routing_wsn/managers/onionvalidator.h
- src/onion_routing_wsn/managers/onionvalidator.cc

3.13 ns3::orLayer Struct Reference

structure holding details resulting from layer decryption of an onion message

```
#include "onion-routing.h"
```

Public Attributes

- uint8_t ∗ innerLayer

  inner content of the onion message without the next hop address
- uint16_t innerLayerLen

  length of the inner content of the onion message
- uint8_t ∗ nextHopIP

  ip address given in the serialized form

3.13.1 Detailed Description

structure holding details resulting from layer decryption of an onion message

Definition at line 22 of file onion-routing.h.

3.13.2 Member Data Documentation

3.13.2.1 innerLayer `uint8_t * ns3::orLayer::innerLayer`

inner content of the onion message without the next hop address

Definition at line 24 of file onion-routing.h.

Referenced by ns3::OnionRouting::PeelOnion(), and ns3::SensorNode::ProcessOnionHead().

3.13.2.2 innerLayerLen `uint16_t ns3::orLayer::innerLayerLen`

length of the inner content of the onion message

Definition at line 25 of file onion-routing.h.

Referenced by ns3::OnionRouting::PeelOnion(), and ns3::SensorNode::ProcessOnionHead().

3.13.2.3 nextHopIP `uint8_t * ns3::orLayer::nextHopIP`

ip address given in the serialized form

Definition at line 23 of file onion-routing.h.

Referenced by ns3::OnionRouting::PeelOnion(), and ns3::SensorNode::ProcessOnionHead().

The documentation for this struct was generated from the following file:

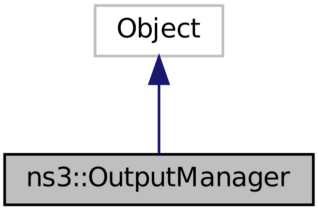
- src/onion-routing/model/bckp/onion-routing.h

3.14 ns3::OutputManager Class Reference

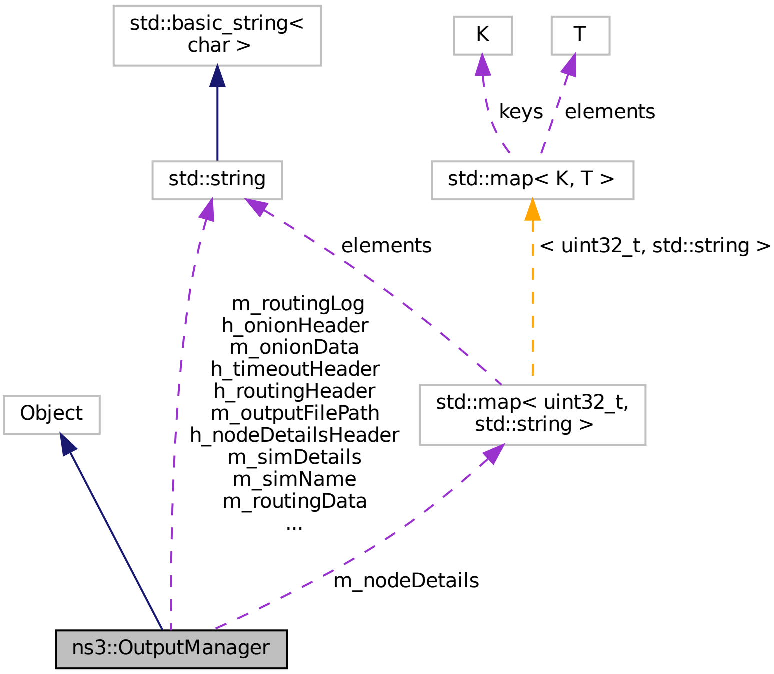
Class that manages the output of the simulation. ConsoleLog output and Output on .csv file

```
#include "outputmanager.h"
```

Inheritance diagram for ns3::OutputManager:



Collaboration diagram for ns3::OutputManager:



Public Member Functions

- OutputManager ()

  Default constructor.
- OutputManager (std::string name, uint16_t simNum, uint16_t numNodes, enum Topology, enum Routing, bool printDescription)

  constructor that takes simulation details to print on the csv file or on the console log
- ∼OutputManager ()

  Default destructor.
- void AbortOnion (Time abort_at)

  Called when an onion is deleted.
- void AddNodeDetails (Ipv4Address node_ip, int coord_x, int coord_y)

  register node details: IP, location on the network
- void AddNodeDetails (Ipv4Address node_ip, int coord_x, int coord_y, int n_degree)

  register node details: IP, location on the network and node degree for node degree reference check ns3::Wsn_node::NodeDegree()
- void CreateOutputFile ()

  Create stream wrapper and the output file, if the output file already exists, then delete the content.
- std::string CurrentTime (void)

  return the current time as a string
- enum Routing GetRouting (void)

  return the enum of the current routing algorithm used in the network
- std::string Ipv4ToString (Ipv4Address ip)

  Convert an IpV4 address to a string.
- void NewHandshake (int node_num, Ipv4Address node_ip, Time recv_at)

  Called when the sink receives a new handhake message.
- void OnionRoutingRecv (Time recv_at)

  Called by each node that receives an onion message.
- void OnionRoutingSend (Ipv4Address send_ip, Ipv4Address recv_ip, int packet_size, int head_size, int body_size, Time sent_at)

  Called by each node that sends an onion message.
- void PrintIntro (std::string intro)

  Print the simulation description containing simulation settings on the csv file or on the console log.
- void PrintLine (std::string line)

  print the given argument on the csv file
- void PrintNodeDetails (std::map< uint32_t, std::string > reachable)

  print node details on the csv file, print only nodes reachable by the sink node
- void RecvOnion (Time recv_at)

  Called by the sink node when it receives back the onion message.
- void SendOnion (int packet_size, int head_size, int body_size, int onion_path_len, Time sent_at)

  Called by the sink node when it sends a new onion message.
- void SetRouting (enum Routing routing)

  set the enum of the current routing algorithm used in the network
- void SimulationEnd (std::string end_at)

  Print at the end of the simulation.

Static Public Member Functions

- static TypeId GetTypeId (void)

  Register this type.

Public Attributes

- std::string h_nodeDetailsHeader

  header of CSV format
- std::string h_onionHeader

  header of CSV format
- std::string h_routingHeader

  header of CSV format
- std::string h_timeoutHeader

  header of CSV format
- std::map< uint32_t, std::string > m_nodeDetails

  holds details of nodes in the network for printing at the end of the csv file.
- std::string m_onionData

  holds data of the onion message currently executing in the network
- int m_onionId = 0

  identifies the onion message
- int m_onionPathLength = 0

  the onion path length
- std::string m_outputFilePath

  path to the directory where output files are stored
- bool m_printDescription

  boolean choice to print the description of the simulation parameters
- enum Routing m_routing

  information on the routing protocol
- std::string m_routingData

  holds data of the onion message traveling from hop to hop
- std::string m_routingLog

  holds data of the onion message traveling from hop to hop
- std::string m_simDetails

  holds details of the simulation
- std::string m_simName

  holds the name of the simulation
- Ptr< OutputStreamWrapper > m_simStreamWrapper

  stream wrapper to write on file
- double t_hopDelta = 0

  Hold time information of the onion message traveling from hop to hop.
- double t_onionDelta = 0

  Hold time information of the onion message traveling in the network.

3.14.1 Detailed Description

Class that manages the output of the simulation. ConsoleLog output and Output on .csv file

Definition at line 49 of file outputmanager.h.

3.14.2 Constructor & Destructor Documentation

3.14.2.1 OutputManager() [1/2] `ns3::OutputManager::OutputManager ( )`

Default constructor.

Definition at line 47 of file outputmanager.cc.

3.14.2.2 OutputManager() [2/2]

```
ns3::OutputManager::OutputManager (
        std::string name,
        uint16_t simNum,
        uint16_t numNodes,
        enum Topology topology,
        enum Routing routing,
        bool printDescription )
```

constructor that takes simulation details to print on the csv file or on the console log

Definition at line 51 of file outputmanager.cc.

References ns3::AODV, ns3::DISC, ns3::DSDV, ns3::DSR, ns3::GRID, m_printDescription, m_routing, m_simDetails, m_simName, and ns3::OLSR.

3.14.2.3 ∼OutputManager() `ns3::OutputManager::∼OutputManager ( )`

Default destructor.

Definition at line 101 of file outputmanager.cc.

3.14.3 Member Function Documentation

3.14.3.1 AbortOnion()

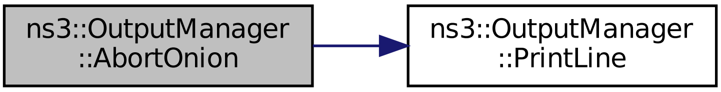
```
void ns3::OutputManager::AbortOnion (
        Time abort_at )
```

Called when an onion is deleted.

Definition at line 197 of file outputmanager.cc.

References m_onionId, m_onionPathLength, m_simDetails, m_simName, and PrintLine().

Here is the call graph for this function:


ns3::OutputManager ::AbortOnion
ns3::OutputManager ::PrintLine

3.14.3.2 AddNodeDetails() [1/2]

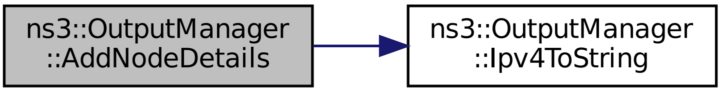
```
void ns3::OutputManager::AddNodeDetails (
        Ipv4Address node_ip,
        int coord_x,
        int coord_y )
```

register node details: IP, location on the network

Definition at line 217 of file outputmanager.cc.

References Ipv4ToString(), m_nodeDetails, m_simDetails, and m_simName.

Here is the call graph for this function:


ns3::OutputManager ::AddNodeDetails
ns3::OutputManager ::Ipv4ToString

3.14.3.3 AddNodeDetails() [2/2]

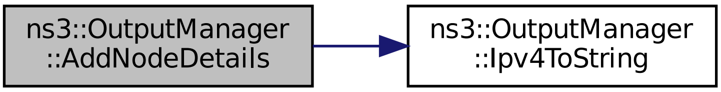
```
void ns3::OutputManager::AddNodeDetails (
        Ipv4Address node_ip,
        int coord_x,
        int coord_y,
        int n_degree )
```

register node details: IP, location on the network and node degree for node degree reference check ns3::Wsn_node::NodeDegree()

Definition at line 226 of file outputmanager.cc.

References Ipv4ToString(), m_nodeDetails, m_simDetails, and m_simName.

Here is the call graph for this function:


ns3::OutputManager ::AddNodeDetails
ns3::OutputManager ::Ipv4ToString

3.14.3.4 CreateOutputFile() `void ns3::OutputManager::CreateOutputFile ( )`

Create stream wrapper and the output file, if the output file already exists, then delete the content.

Definition at line 91 of file outputmanager.cc.

References m_outputFilePath, m_simName, and m_simStreamWrapper.

3.14.3.5 CurrentTime() `std::string ns3::OutputManager::CurrentTime ( void )`

return the current time as a string

Definition at line 262 of file outputmanager.cc.

3.14.3.6 GetRouting() `enum Routing ns3::OutputManager::GetRouting ( void )`

return the enum of the current routing algorithm used in the network

Definition at line 276 of file outputmanager.cc.

References m_routing.

3.14.3.7 GetTypeId() `TypeId ns3::OutputManager::GetTypeId ( void ) [static]`

Register this type.

Returns

The object TypeId.

Definition at line 34 of file outputmanager.cc.

References m_outputFilePath.

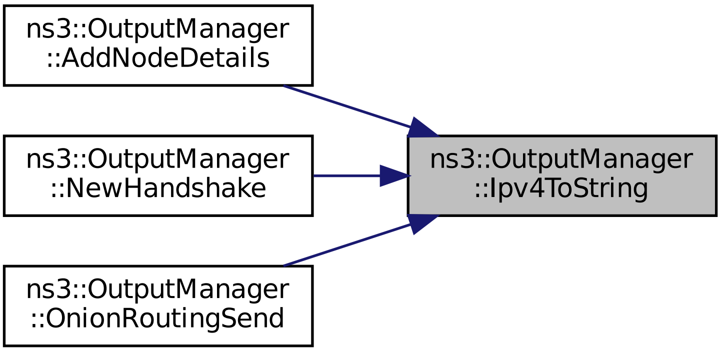
3.14.3.8 Ipv4ToString() `std::string ns3::OutputManager::Ipv4ToString ( Ipv4Address ip )`

Convert an IpV4 address to a string.

Definition at line 254 of file outputmanager.cc.

Referenced by AddNodeDetails(), NewHandshake(), and OnionRoutingSend().

Here is the caller graph for this function:


ns3::OutputManager ::AddNodeDetails
ns3::OutputManager ::NewHandshake
ns3::OutputManager ::OnionRoutingSend
ns3::OutputManager ::Ipv4ToString

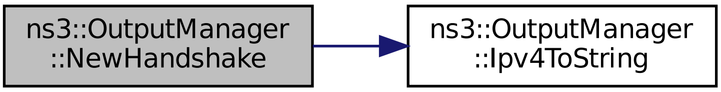
3.14.3.9 NewHandshake() `void ns3::OutputManager::NewHandshake ( int node_num, Ipv4Address node_ip, Time recv_at )`

Called when the sink receives a new handhake message.

Definition at line 210 of file outputmanager.cc.

References Ipv4ToString().

Here is the call graph for this function:


ns3::OutputManager ::NewHandshake
ns3::OutputManager ::Ipv4ToString

3.14.3.10 OnionRoutingRecv()

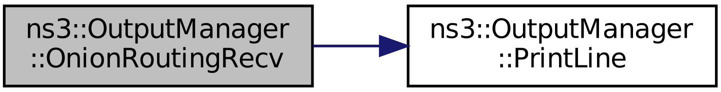
```
void ns3::OutputManager::OnionRoutingRecv (
        Time recv_at )
```

Called by each node that receives an onion message.

Definition at line 185 of file outputmanager.cc.

References m_routingData, m_routingLog, PrintLine(), and t_hopDelta.

Here is the call graph for this function:


ns3::OutputManager ::OnionRoutingRecv
ns3::OutputManager ::PrintLine

3.14.3.11 OnionRoutingSend()

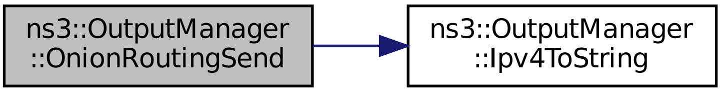
```
void ns3::OutputManager::OnionRoutingSend (
        Ipv4Address send_ip,
        Ipv4Address recv_ip,
        int packet_size,
        int head_size,
        int body_size,
        Time sent_at )
```

Called by each node that sends an onion message.

Definition at line 167 of file outputmanager.cc.

References Ipv4ToString(), m_onionId, m_routingData, m_routingLog, m_simDetails, m_simName, and t_hopDelta.

Here is the call graph for this function:


ns3::OutputManager ::OnionRoutingSend
ns3::OutputManager ::Ipv4ToString

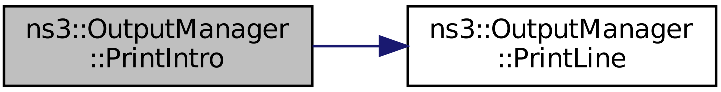
3.14.3.12 PrintIntro() `void ns3::OutputManager::PrintIntro ( std::string intro )`

Print the simulation description containing simulation settings on the csv file or on the console log.

Definition at line 106 of file outputmanager.cc.

References h_nodeDetailsHeader, h_onionHeader, h_routingHeader, h_timeoutHeader, m_printDescription, and PrintLine().

Here is the call graph for this function:


ns3::OutputManager ::PrintIntro
ns3::OutputManager ::PrintLine

3.14.3.13 PrintLine() `void ns3::OutputManager::PrintLine ( std::string line )`

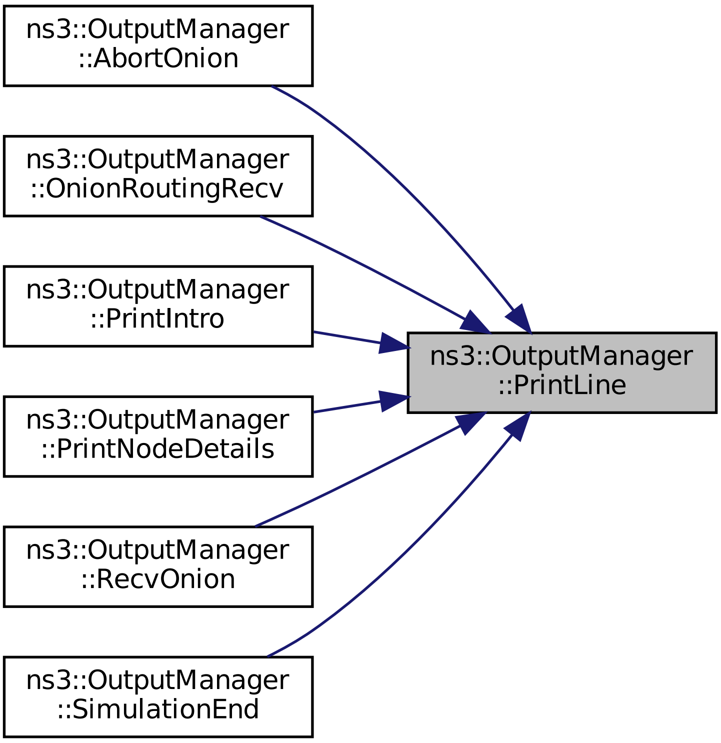
print the given argument on the csv file

Definition at line 247 of file outputmanager.cc.

References m_simStreamWrapper.

Referenced by AbortOnion(), OnionRoutingRecv(), PrintIntro(), PrintNodeDetails(), RecvOnion(), and SimulationEnd().

Here is the caller graph for this function:


ns3::OutputManager ::AbortOnion
ns3::OutputManager ::OnionRoutingRecv
ns3::OutputManager ::PrintIntro
ns3::OutputManager ::PrintNodeDetails
ns3::OutputManager ::RecvOnion
ns3::OutputManager ::SimulationEnd
ns3::OutputManager ::PrintLine

3.14.3.14 PrintNodeDetails() `void ns3::OutputManager::PrintNodeDetails ( std::map< uint32_t, std::string > reachable )`

print node details on the csv file, print only nodes reachable by the sink node

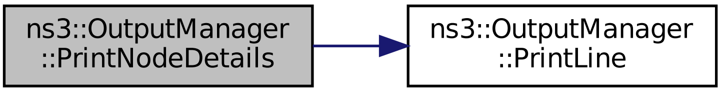
Parameters

in	*reachable*	the set of nodes reachable by the sink node

Definition at line 235 of file outputmanager.cc.

References m_nodeDetails, and PrintLine().

Here is the call graph for this function:



3.14.3.15 RecvOnion() `void ns3::OutputManager::RecvOnion (`

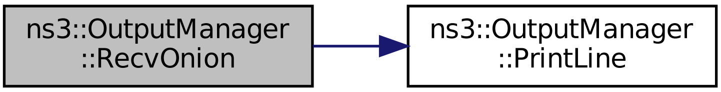
```
Time recv_at )
```

Called by the sink node when it receives back the onion message.

Definition at line 154 of file outputmanager.cc.

References m_onionData, m_onionId, PrintLine(), and t_onionDelta.

Here is the call graph for this function:



3.14.3.16 SendOnion() `void ns3::OutputManager::SendOnion (`

```
int packet_size,
int head_size,
int body_size,
int onion_path_len,
Time sent_at )
```

Called by the sink node when it sends a new onion message.

Definition at line 135 of file outputmanager.cc.

References m_onionData, m_onionId, m_onionPathLength, m_simDetails, m_simName, and t_onionDelta.

3.14.3.17 SetRouting() `void ns3::OutputManager::SetRouting ( enum Routing routing )`

set the enum of the current routing algorithm used in the network

Definition at line 270 of file outputmanager.cc.

References m_routing.

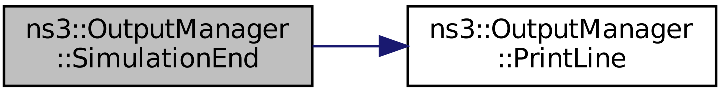
3.14.3.18 SimulationEnd() `void ns3::OutputManager::SimulationEnd ( std::string end_at )`

Print at the end of the simulation.

Definition at line 126 of file outputmanager.cc.

References m_printDescription, and PrintLine().

Here is the call graph for this function:


ns3::OutputManager ::SimulationEnd
ns3::OutputManager ::PrintLine

3.14.4 Member Data Documentation

3.14.4.1 h_nodeDetailsHeader `std::string ns3::OutputManager::h_nodeDetailsHeader`

Initial value:

```
= "node_details,sim_name,sim_num,num_of_nodes,topology,routing,"
                                    "coord_x,coord_y,node_degree"
```

header of CSV format

Definition at line 224 of file outputmanager.h.

Referenced by PrintIntro().

3.14.4.2 h_onionHeader `std::string ns3::OutputManager::h_onionHeader`

Initial value:

```
=
      "onion_details,sim_name,sim_num,num_of_nodes,topology,routing,onion_"
      "id,packet_size,onion_head_size,onion_body_size,onion_"
      "path_length,sent_at,recv_at,query_time_to_return"
```

header of CSV format

Definition at line 215 of file outputmanager.h.

Referenced by PrintIntro().

3.14.4.3 h_routingHeader `std::string ns3::OutputManager::h_routingHeader`

Initial value:

```
= "onion_routing,sim_name,sim_num,num_of_nodes,topology,routing,"
                                "onion_id,send_ip,recv_ip,packet_size,onion_head_size,onion_body_"
                                "size,sent_at,recv_at,hop_time"
```

header of CSV format

Definition at line 219 of file outputmanager.h.

Referenced by PrintIntro().

3.14.4.4 h_timeoutHeader `std::string ns3::OutputManager::h_timeoutHeader`

Initial value:

```
= "timeout,sim_name,sim_num,num_of_nodes,topology,routing,onion_id,"
                                "onion_path_length,abort_time"
```

header of CSV format

Definition at line 222 of file outputmanager.h.

Referenced by PrintIntro().

3.14.4.5 m_nodeDetails `std::map<uint32_t,std::string> ns3::OutputManager::m_nodeDetails`

holds details of nodes in the network for printing at the end of the csv file.

Definition at line 240 of file outputmanager.h.

Referenced by AddNodeDetails(), and PrintNodeDetails().

3.14.4.6 m_onionData `std::string ns3::OutputManager::m_onionData`

holds data of the onion message currently executing in the network

Definition at line 228 of file outputmanager.h.

Referenced by RecvOnion(), and SendOnion().

3.14.4.7 m_onionId `int ns3::OutputManager::m_onionId = 0`

identifies the onion message

Definition at line 231 of file outputmanager.h.

Referenced by AbortOnion(), OnionRoutingSend(), RecvOnion(), and SendOnion().

3.14.4.8 m_onionPathLength `int ns3::OutputManager::m_onionPathLength = 0`

the onion path length

Definition at line 232 of file outputmanager.h.

Referenced by AbortOnion(), and SendOnion().

3.14.4.9 m_outputFilePath `std::string ns3::OutputManager::m_outputFilePath`

path to the directory where output files are stored

Definition at line 242 of file outputmanager.h.

Referenced by CreateOutputFile(), and GetTypeId().

3.14.4.10 m_printDescription `bool ns3::OutputManager::m_printDescription`

boolean choice to print the description of the simulation parameters

Definition at line 208 of file outputmanager.h.

Referenced by OutputManager(), PrintIntro(), and SimulationEnd().

3.14.4.11 m_routing `enum Routing ns3::OutputManager::m_routing`

information on the routing protocol

Definition at line 237 of file outputmanager.h.

Referenced by GetRouting(), OutputManager(), and SetRouting().

3.14.4.12 m_routingData `std::string ns3::OutputManager::m_routingData`

holds data of the onion message traveling from hop to hop

Definition at line 229 of file outputmanager.h.

Referenced by OnionRoutingRecv(), and OnionRoutingSend().

3.14.4.13 m_routingLog `std::string ns3::OutputManager::m_routingLog`

holds data of the onion message traveling from hop to hop

Definition at line 230 of file outputmanager.h.

Referenced by OnionRoutingRecv(), and OnionRoutingSend().

3.14.4.14 m_simDetails `std::string ns3::OutputManager::m_simDetails`

holds details of the simulation

Definition at line 212 of file outputmanager.h.

Referenced by AbortOnion(), AddNodeDetails(), OnionRoutingSend(), OutputManager(), and SendOnion().

3.14.4.15 m_simName `std::string ns3::OutputManager::m_simName`

holds the name of the simulation

Definition at line 210 of file outputmanager.h.

Referenced by AbortOnion(), AddNodeDetails(), CreateOutputFile(), OnionRoutingSend(), OutputManager(), and SendOnion().

3.14.4.16 m_simStreamWrapper `Ptr<OutputStreamWrapper> ns3::OutputManager::m_simStream↩ Wrapper`

stream wrapper to write on file

Definition at line 206 of file outputmanager.h.

Referenced by CreateOutputFile(), and PrintLine().

3.14.4.17 t_hopDelta `double ns3::OutputManager::t_hopDelta = 0`

Hold time information of the onion message traveling from hop to hop.

Definition at line 235 of file outputmanager.h.

Referenced by OnionRoutingRecv(), and OnionRoutingSend().

3.14.4.18 t_onionDelta `double ns3::OutputManager::t_onionDelta = 0`

Hold time information of the onion message traveling in the network.

Definition at line 234 of file outputmanager.h.

Referenced by RecvOnion(), and SendOnion().

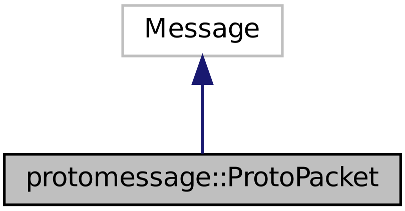
The documentation for this class was generated from the following files:

- src/onion_routing_wsn/managers/outputmanager.h
- src/onion_routing_wsn/managers/outputmanager.cc

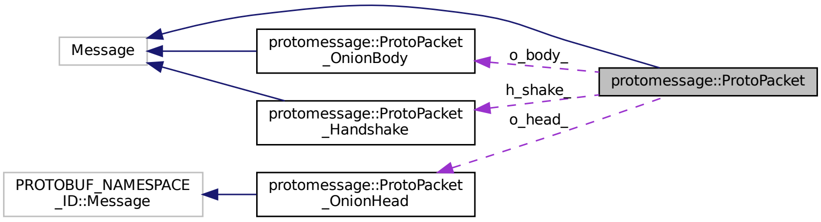
3.15 protomessage::ProtoPacket Class Reference

```
#include "proto-packet.pb.h"
```

Inheritance diagram for protomessage::ProtoPacket:



Collaboration diagram for protomessage::ProtoPacket:


Message
protomessage::ProtoPacket _OnionBody
protomessage::ProtoPacket _Handshake
PROTOBUF_NAMESPACE _ID::Message
protomessage::ProtoPacket _OnionHead
protomessage::ProtoPacket
o_body_
h_shake_
o_head_

Classes

- class _Internal

Public Types

- enum : int { kHShakeFieldNumber = 1, kOHeadFieldNumber = 2, kOBodyFieldNumber = 3 }
- typedef ProtoPacket_Handshake Handshake
- typedef ProtoPacket_OnionBody OnionBody
- typedef ProtoPacket_OnionHead OnionHead

Public Member Functions

- ProtoPacket ()
- constexpr ProtoPacket (::PROTOBUF_NAMESPACE_ID::internal::ConstantInitialized)
- ProtoPacket (const ProtoPacket &from)
- ProtoPacket (ProtoPacket &&from) noexcept
- ∼ProtoPacket () override
- const char ∗ _InternalParse (const char ∗ptr, ::PROTOBUF_NAMESPACE_ID::internal::ParseContext ∗ctx) final
- uint8_t ∗ _InternalSerialize (uint8_t ∗target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream ∗stream) const final
- size_t ByteSizeLong () const final
- PROTOBUF_ATTRIBUTE_REINITIALIZES void Clear () final
- void clear_h_shake ()
- void clear_o_body ()
- void clear_o_head ()
- void CopyFrom (const ProtoPacket &from)
- int GetCachedSize () const final
- const ::PROTOBUF_NAMESPACE_ID::Message::ClassData ∗ GetClassData () const final
- ::PROTOBUF_NAMESPACE_ID::Metadata GetMetadata () const final
- const ::protomessage::ProtoPacket_Handshake & h_shake () const
- bool has_h_shake () const
- bool has_o_body () const
- bool has_o_head () const
- bool IsInitialized () const final
- void MergeFrom (const ProtoPacket &from)
- ::protomessage::ProtoPacket_Handshake ∗ mutable_h_shake ()
- ::protomessage::ProtoPacket_OnionBody ∗ mutable_o_body ()

- ::protomessage::ProtoPacket_OnionHead ∗ mutable_o_head ()
- ProtoPacket ∗ New (::PROTOBUF_NAMESPACE_ID::Arena ∗arena=nullptr) const final
- const ::protomessage::ProtoPacket_OnionBody & o_body () const
- const ::protomessage::ProtoPacket_OnionHead & o_head () const
- ProtoPacket & operator= (const ProtoPacket &from)
- ProtoPacket & operator= (ProtoPacket &&from) noexcept
- PROTOBUF_NODISCARD ::protomessage::ProtoPacket_Handshake ∗ release_h_shake ()
- PROTOBUF_NODISCARD ::protomessage::ProtoPacket_OnionBody ∗ release_o_body ()
- PROTOBUF_NODISCARD ::protomessage::ProtoPacket_OnionHead ∗ release_o_head ()
- void set_allocated_h_shake (::protomessage::ProtoPacket_Handshake ∗h_shake)
- void set_allocated_o_body (::protomessage::ProtoPacket_OnionBody ∗o_body)
- void set_allocated_o_head (::protomessage::ProtoPacket_OnionHead ∗o_head)
- void Swap (ProtoPacket ∗other)
- ::protomessage::ProtoPacket_Handshake ∗ unsafe_arena_release_h_shake ()
- ::protomessage::ProtoPacket_OnionBody ∗ unsafe_arena_release_o_body ()
- ::protomessage::ProtoPacket_OnionHead ∗ unsafe_arena_release_o_head ()
- void unsafe_arena_set_allocated_h_shake (::protomessage::ProtoPacket_Handshake ∗h_shake)
- void unsafe_arena_set_allocated_o_body (::protomessage::ProtoPacket_OnionBody ∗o_body)
- void unsafe_arena_set_allocated_o_head (::protomessage::ProtoPacket_OnionHead ∗o_head)
- void UnsafeArenaSwap (ProtoPacket ∗other)

Static Public Member Functions

- static const ProtoPacket & default_instance ()
- static const ::PROTOBUF_NAMESPACE_ID::Descriptor ∗ descriptor ()
- static const ::PROTOBUF_NAMESPACE_ID::Descriptor ∗ GetDescriptor ()
- static const ::PROTOBUF_NAMESPACE_ID::Reflection ∗ GetReflection ()
- static const ProtoPacket ∗ internal_default_instance ()

Static Public Attributes

- static const ClassData _class_data_
- static constexpr int kIndexInFileMessages

Protected Member Functions

- ProtoPacket (::PROTOBUF_NAMESPACE_ID::Arena ∗arena, bool is_message_owned=false)

Private Types

- typedef void DestructorSkippable_
- typedef void InternalArenaConstructable_

Private Member Functions

- const ::protomessage::ProtoPacket_Handshake & _internal_h_shake () const
- bool _internal_has_h_shake () const
- bool _internal_has_o_body () const
- bool _internal_has_o_head () const
- ::protomessage::ProtoPacket_Handshake ∗ _internal_mutable_h_shake ()
- ::protomessage::ProtoPacket_OnionBody ∗ _internal_mutable_o_body ()
- ::protomessage::ProtoPacket_OnionHead ∗ _internal_mutable_o_head ()
- const ::protomessage::ProtoPacket_OnionBody & _internal_o_body () const
- const ::protomessage::ProtoPacket_OnionHead & _internal_o_head () const
- void InternalSwap (ProtoPacket ∗other)
- void RegisterArenaDtor (::PROTOBUF_NAMESPACE_ID::Arena ∗arena)
- void SetCachedSize (int size) const final
- void SharedCtor ()
- void SharedDtor ()

Static Private Member Functions

- static void ArenaDtor (void ∗object)
- ::PROTOBUF_NAMESPACE_ID::StringPiece FullMessageName ()
- static void MergeImpl (::PROTOBUF_NAMESPACE_ID::Message ∗to, const ::PROTOBUF_NAMESPACE_ID::Message &from)

Private Attributes

- mutable ::PROTOBUF_NAMESPACE_ID::internal::CachedSize _cached_size_
- ::PROTOBUF_NAMESPACE_ID::internal::HasBits< 1 > _has_bits_
- ::protomessage::ProtoPacket_Handshake ∗ h_shake_
- ::protomessage::ProtoPacket_OnionBody ∗ o_body_
- ::protomessage::ProtoPacket_OnionHead ∗ o_head_

Friends

- template<typename T >
  class ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper
- class ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata
- struct ::TableStruct_proto_2dpacket_2eproto
- void swap (ProtoPacket &a, ProtoPacket &b)

3.15.1 Detailed Description

Definition at line 598 of file proto-packet.pb.h.

3.15.2 Member Typedef Documentation

3.15.2.1 DestructorSkippable_ `typedef void protomessage::ProtoPacket::DestructorSkippable_ [private]`

Definition at line 788 of file proto-packet.pb.h.

3.15.2.2 Handshake `typedef ProtoPacket_Handshake protomessage::ProtoPacket::Handshake`

Definition at line 719 of file proto-packet.pb.h.

3.15.2.3 InternalArenaConstructable_ `typedef void protomessage::ProtoPacket::InternalArenaConstructable_ [private]`

Definition at line 787 of file proto-packet.pb.h.

3.15.2.4 OnionBody `typedef ProtoPacket_OnionBody protomessage::ProtoPacket::OnionBody`

Definition at line 718 of file proto-packet.pb.h.

3.15.2.5 OnionHead `typedef ProtoPacket_OnionHead protomessage::ProtoPacket::OnionHead`

Definition at line 717 of file proto-packet.pb.h.

3.15.3 Member Enumeration Documentation

3.15.3.1 anonymous enum `anonymous enum : int`

Enumerator

kHShakeFieldNumber	
kOHeadFieldNumber	
kOBodyFieldNumber	

Definition at line 723 of file proto-packet.pb.h.

3.15.4 Constructor & Destructor Documentation

3.15.4.1 ProtoPacket() [1/5] protomessage::ProtoPacket::ProtoPacket ( ) [inline]

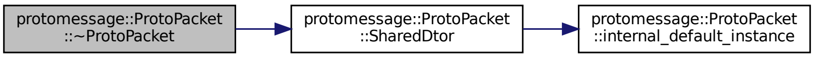
Definition at line 601 of file proto-packet.pb.h.

3.15.4.2 ∼ProtoPacket() protomessage::ProtoPacket::∼ProtoPacket ( ) [override]

Definition at line 992 of file proto-packet.pb.cc.

References SharedDtor().

Here is the call graph for this function:


protomessage::ProtoPacket
::∼ProtoPacket
protomessage::ProtoPacket
::SharedDtor
protomessage::ProtoPacket
::internal_default_instance

3.15.4.3 ProtoPacket() [2/5] constexpr protomessage::ProtoPacket::ProtoPacket (::PROTOBUF_NAMESPACE_ID::internal::ConstantInitialized) [explicit], [constexpr]

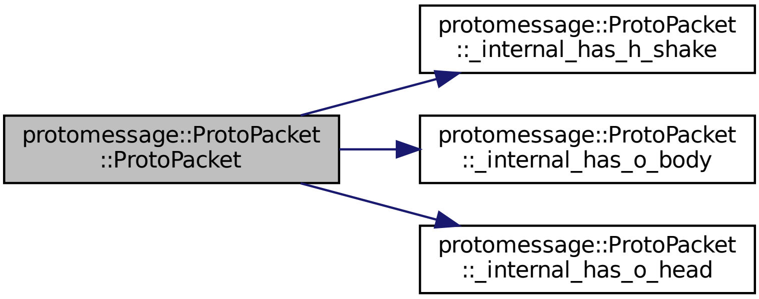
Definition at line 59 of file proto-packet.pb.cc.

3.15.4.4 ProtoPacket() [3/5] protomessage::ProtoPacket::ProtoPacket (const ProtoPacket & *from*)

Definition at line 963 of file proto-packet.pb.cc.

References _internal_has_h_shake(), _internal_has_o_body(), _internal_has_o_head(), h_shake_, o_body_, and o_head_.

Here is the call graph for this function:


protomessage::ProtoPacket
::ProtoPacket
protomessage::ProtoPacket
::_internal_has_h_shake
protomessage::ProtoPacket
::_internal_has_o_body
protomessage::ProtoPacket
::_internal_has_o_head

3.15.4.5 ProtoPacket() [4/5] protomessage::ProtoPacket::ProtoPacket (ProtoPacket && *from*) [inline], [noexcept]

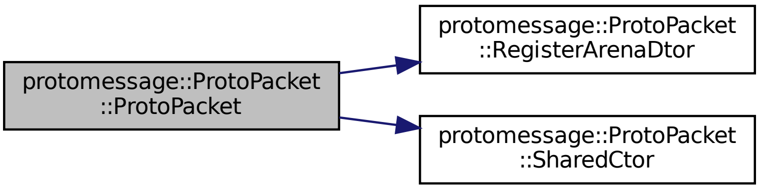
Definition at line 606 of file proto-packet.pb.h.

3.15.4.6 ProtoPacket() [5/5] protomessage::ProtoPacket::ProtoPacket (::PROTOBUF_NAMESPACE_ID::Arena * *arena*, bool *is_message_owned* = *false*) [explicit], [protected]

Definition at line 954 of file proto-packet.pb.cc.

References RegisterArenaDtor(), and SharedCtor().

Here is the call graph for this function:


protomessage::ProtoPacket ::ProtoPacket
protomessage::ProtoPacket ::RegisterArenaDtor
protomessage::ProtoPacket ::SharedCtor

3.15.5 Member Function Documentation

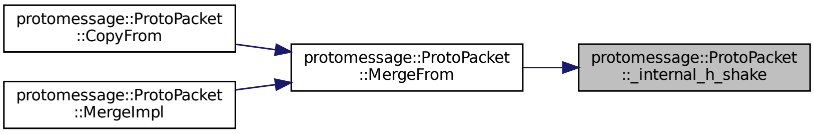
3.15.5.1 _internal_h_shake() const ::protomessage::ProtoPacket_Handshake & protomessage::ProtoPacket::_internal_h_shake ( ) const [inline], [private]

Definition at line 1164 of file proto-packet.pb.h.

References protomessage::_ProtoPacket_Handshake_default_instance_.

Referenced by MergeFrom().

Here is the caller graph for this function:


protomessage::ProtoPacket ::CopyFrom
protomessage::ProtoPacket ::MergeImpl
protomessage::ProtoPacket ::MergeFrom
protomessage::ProtoPacket ::_internal_h_shake

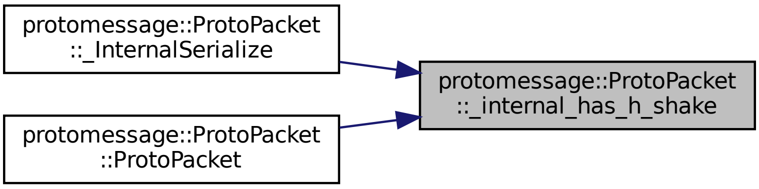
3.15.5.2 _internal_has_h_shake() `bool protomessage::ProtoPacket::_internal_has_h_shake ( ) const [inline], [private]`

Definition at line 1152 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

Referenced by _InternalSerialize(), and ProtoPacket().

Here is the caller graph for this function:


protomessage::ProtoPacket ::_InternalSerialize
protomessage::ProtoPacket ::ProtoPacket
protomessage::ProtoPacket ::_internal_has_h_shake

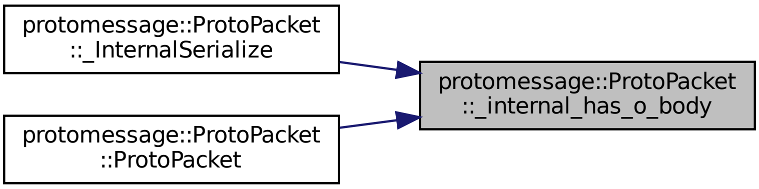
3.15.5.3 _internal_has_o_body() `bool protomessage::ProtoPacket::_internal_has_o_body ( ) const [inline], [private]`

Definition at line 1332 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

Referenced by _InternalSerialize(), and ProtoPacket().

Here is the caller graph for this function:


protomessage::ProtoPacket ::_InternalSerialize
protomessage::ProtoPacket ::ProtoPacket
protomessage::ProtoPacket ::_internal_has_o_body

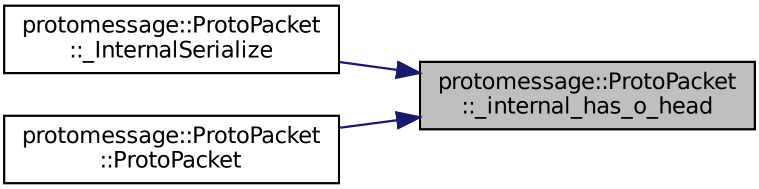
3.15.5.4 _internal_has_o_head() `bool protomessage::ProtoPacket::_internal_has_o_head ( ) const [inline], [private]`

Definition at line 1242 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

Referenced by _InternalSerialize(), and ProtoPacket().

Here is the caller graph for this function:


protomessage::ProtoPacket ::_InternalSerialize
protomessage::ProtoPacket ::ProtoPacket
protomessage::ProtoPacket ::_internal_has_o_head

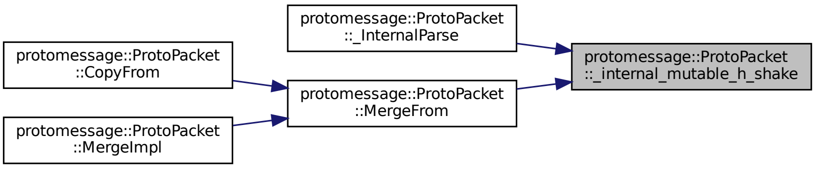
3.15.5.5 _internal_mutable_h_shake() `protomessage::ProtoPacket_Handshake * protomessage::↩ ProtoPacket::_internal_mutable_h_shake ( ) [inline], [private]`

Definition at line 1208 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

Referenced by _InternalParse(), and MergeFrom().

Here is the caller graph for this function:


protomessage::ProtoPacket ::_InternalParse
protomessage::ProtoPacket ::CopyFrom
protomessage::ProtoPacket ::MergeFrom
protomessage::ProtoPacket ::MergeImpl
protomessage::ProtoPacket ::_internal_mutable_h_shake

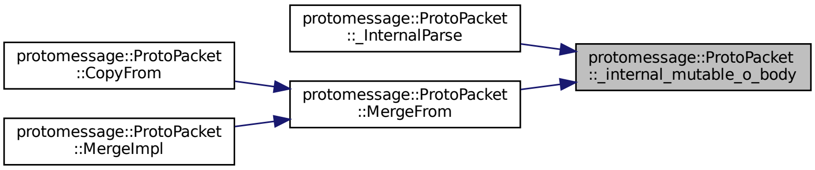
3.15.5.6 _internal_mutable_o_body() protomessage::ProtoPacket_OnionBody * protomessage::Proto←↩ Packet::_internal_mutable_o_body ( ) [inline], [private]

Definition at line 1388 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

Referenced by _InternalParse(), and MergeFrom().

Here is the caller graph for this function:


protomessage::ProtoPacket ::_InternalParse
protomessage::ProtoPacket ::CopyFrom
protomessage::ProtoPacket ::MergeFrom
protomessage::ProtoPacket ::MergeImpl
protomessage::ProtoPacket ::_internal_mutable_o_body

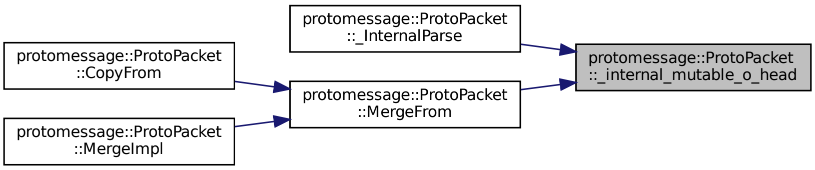
3.15.5.7 _internal_mutable_o_head() protomessage::ProtoPacket_OnionHead * protomessage::Proto←↩ Packet::_internal_mutable_o_head ( ) [inline], [private]

Definition at line 1298 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

Referenced by _InternalParse(), and MergeFrom().

Here is the caller graph for this function:


protomessage::ProtoPacket ::_InternalParse
protomessage::ProtoPacket ::CopyFrom
protomessage::ProtoPacket ::MergeFrom
protomessage::ProtoPacket ::MergeImpl
protomessage::ProtoPacket ::_internal_mutable_o_head

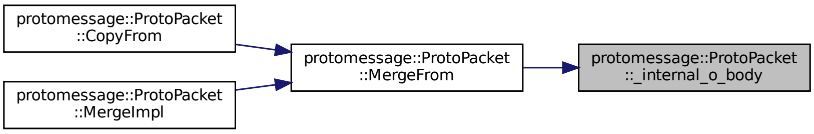
3.15.5.8 _internal_o_body() const ::protomessage::ProtoPacket_OnionBody & protomessage::Proto↩ Packet::_internal_o_body ( ) const [inline], [private]

Definition at line 1344 of file proto-packet.pb.h.

References protomessage::_ProtoPacket_OnionBody_default_instance_.

Referenced by MergeFrom().

Here is the caller graph for this function:


protomessage::ProtoPacket ::CopyFrom
protomessage::ProtoPacket ::MergeImpl
protomessage::ProtoPacket ::MergeFrom
protomessage::ProtoPacket ::_internal_o_body

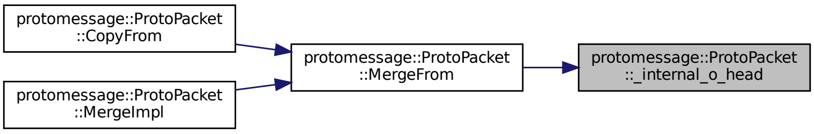
3.15.5.9 _internal_o_head() const ::protomessage::ProtoPacket_OnionHead & protomessage::Proto↩ Packet::_internal_o_head ( ) const [inline], [private]

Definition at line 1254 of file proto-packet.pb.h.

References protomessage::_ProtoPacket_OnionHead_default_instance_.

Referenced by MergeFrom().

Here is the caller graph for this function:


protomessage::ProtoPacket ::CopyFrom
protomessage::ProtoPacket ::MergeImpl
protomessage::ProtoPacket ::MergeFrom
protomessage::ProtoPacket ::_internal_o_head

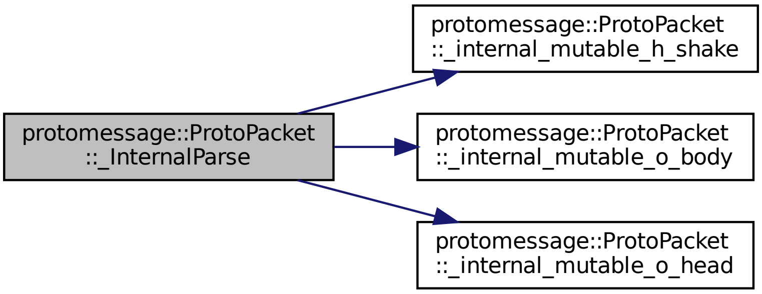
3.15.5.10 _InternalParse()

```
const char * protomessage::ProtoPacket::_InternalParse (
    const char * ptr,
    ::PROTOBUF_NAMESPACE_ID::internal::ParseContext * ctx )  [final]
```

Definition at line 1041 of file proto-packet.pb.cc.

References _has_bits_, _internal_mutable_h_shake(), _internal_mutable_o_body(), _internal_mutable_o_head(), and CHK_.

Here is the call graph for this function:


protomessage::ProtoPacket ::_InternalParse
protomessage::ProtoPacket ::_internal_mutable_h_shake
protomessage::ProtoPacket ::_internal_mutable_o_body
protomessage::ProtoPacket ::_internal_mutable_o_head

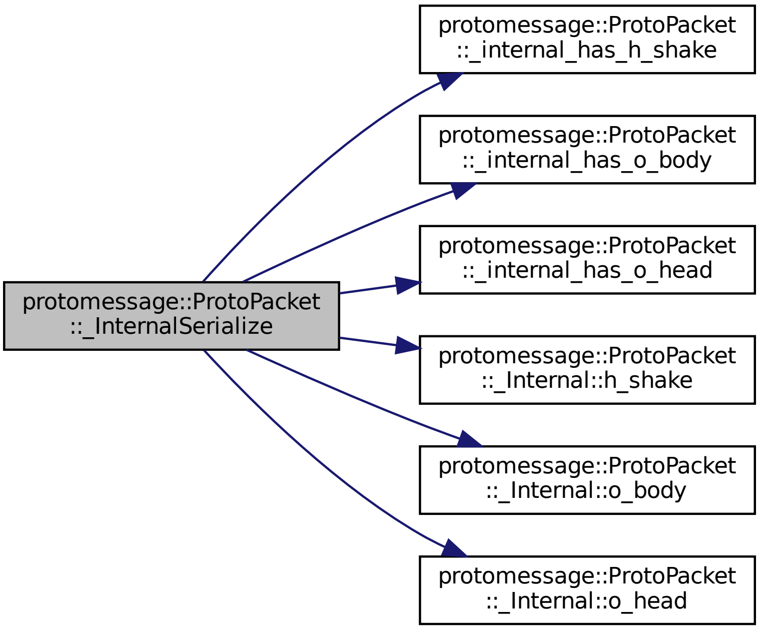
3.15.5.11 _InternalSerialize()

```
uint8_t * protomessage::ProtoPacket::_InternalSerialize (
    uint8_t * target,
    ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream * stream ) const  [final]
```

Definition at line 1096 of file proto-packet.pb.cc.

References _internal_has_h_shake(), _internal_has_o_body(), _internal_has_o_head(), protomessage::ProtoPacket::_Internal::h_shake(), protomessage::ProtoPacket::_Internal::o_body(), and protomessage::ProtoPacket::_Internal::o_head().

Here is the call graph for this function:


protomessage::ProtoPacket ::_InternalSerialize
protomessage::ProtoPacket ::_internal_has_h_shake
protomessage::ProtoPacket ::_internal_has_o_body
protomessage::ProtoPacket ::_internal_has_o_head
protomessage::ProtoPacket ::_Internal::h_shake
protomessage::ProtoPacket ::_Internal::o_body
protomessage::ProtoPacket ::_Internal::o_head

3.15.5.12 ArenaDtor() `void protomessage::ProtoPacket::ArenaDtor ( void * object ) [static], [private]`

Definition at line 1006 of file proto-packet.pb.cc.

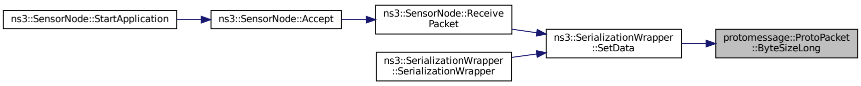
3.15.5.13 ByteSizeLong() `size_t protomessage::ProtoPacket::ByteSizeLong ( ) const [final]`

Definition at line 1134 of file proto-packet.pb.cc.

References _cached_size_, _has_bits_, h_shake_, o_body_, and o_head_.

Referenced by ns3::SerializationWrapper::SetData().

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::SerializationWrapper ::SerializationWrapper
ns3::SerializationWrapper ::SetData
protomessage::ProtoPacket ::ByteSizeLong

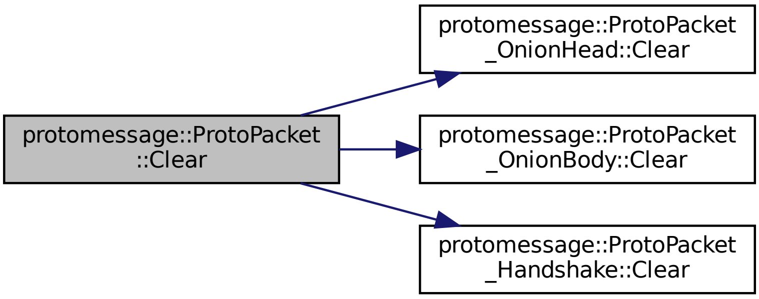
3.15.5.14 Clear() `void protomessage::ProtoPacket::Clear ( ) [final]`

Definition at line 1016 of file proto-packet.pb.cc.

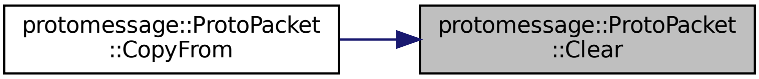
References _has_bits_, protomessage::ProtoPacket_OnionHead::Clear(), protomessage::ProtoPacket_Onion←Body::Clear(), protomessage::ProtoPacket_Handshake::Clear(), h_shake_, o_body_, and o_head_.

Referenced by CopyFrom().

Here is the call graph for this function:


protomessage::ProtoPacket _OnionHead::Clear
protomessage::ProtoPacket ::Clear
protomessage::ProtoPacket _OnionBody::Clear
protomessage::ProtoPacket _Handshake::Clear

Here is the caller graph for this function:


protomessage::ProtoPacket ::CopyFrom
protomessage::ProtoPacket ::Clear

3.15.5.15 clear_h_shake() `void protomessage::ProtoPacket::clear_h_shake ( ) [inline]`

Definition at line 1160 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.15.5.16 clear_o_body() `void protomessage::ProtoPacket::clear_o_body ( ) [inline]`

Definition at line 1340 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.15.5.17 clear_o_head() `void protomessage::ProtoPacket::clear_o_head ( ) [inline]`

Definition at line 1250 of file proto-packet.pb.h.

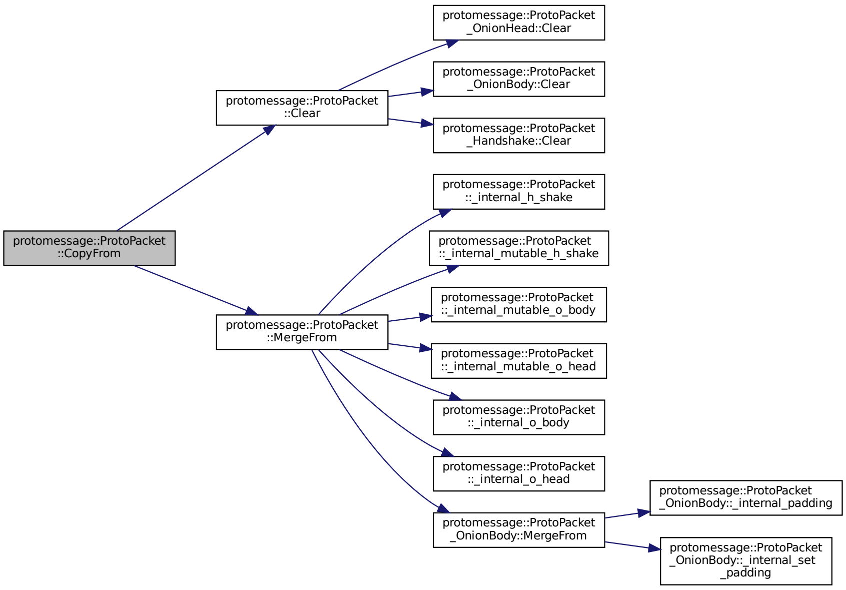
References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.15.5.18 CopyFrom() `void protomessage::ProtoPacket::CopyFrom ( const ProtoPacket & from )`

Definition at line 1203 of file proto-packet.pb.cc.

References Clear(), and MergeFrom().

Here is the call graph for this function:

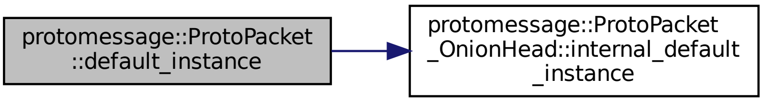

protomessage::ProtoPacket ::CopyFrom
protomessage::ProtoPacket ::Clear
protomessage::ProtoPacket _OnionHead::Clear
protomessage::ProtoPacket _OnionBody::Clear
protomessage::ProtoPacket _Handshake::Clear
protomessage::ProtoPacket ::MergeFrom
protomessage::ProtoPacket ::_internal_h_shake
protomessage::ProtoPacket ::_internal_mutable_h_shake
protomessage::ProtoPacket ::_internal_mutable_o_body
protomessage::ProtoPacket ::_internal_mutable_o_head
protomessage::ProtoPacket ::_internal_o_body
protomessage::ProtoPacket ::_internal_o_head
protomessage::ProtoPacket _OnionBody::MergeFrom
protomessage::ProtoPacket _OnionBody::_internal_padding
protomessage::ProtoPacket _OnionBody::_internal_set _padding

3.15.5.19 default_instance() `static const ProtoPacket& protomessage::ProtoPacket::default_instance ( ) [inline], [static]`

Definition at line 638 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::internal_default_instance().

Here is the call graph for this function:

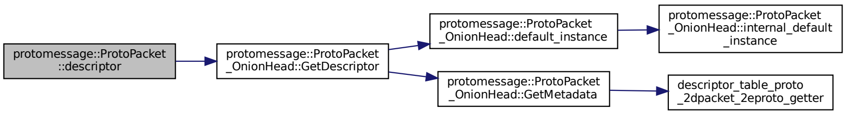

protomessage::ProtoPacket ::default_instance
protomessage::ProtoPacket _OnionHead::internal_default _instance

3.15.5.20 descriptor() `static const ::PROTOBUF_NAMESPACE_ID::Descriptor* protomessage::ProtoPacket::descriptor ( ) [inline], [static]`

Definition at line 629 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::GetDescriptor().

Here is the call graph for this function:


protomessage::ProtoPacket ::descriptor
protomessage::ProtoPacket _OnionHead::GetDescriptor
protomessage::ProtoPacket _OnionHead::default_instance
protomessage::ProtoPacket _OnionHead::internal_default _instance
protomessage::ProtoPacket _OnionHead::GetMetadata
descriptor_table_proto _2dpacket_2eproto_getter

3.15.5.21 FullMessageName() `::PROTOBUF_NAMESPACE_ID::StringPiece protomessage::ProtoPacket::FullMessageName ( ) [inline], [static], [private]`

Definition at line 699 of file proto-packet.pb.h.

3.15.5.22 GetCachedSize() `int protomessage::ProtoPacket::GetCachedSize ( ) const [inline], [final]`

Definition at line 689 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_cached_size_.

3.15.5.23 GetClassData() `const ::PROTOBUF_NAMESPACE_ID::Message::ClassData * protomessage::ProtoPacket::GetClassData ( ) const [final]`

Definition at line 1173 of file proto-packet.pb.cc.

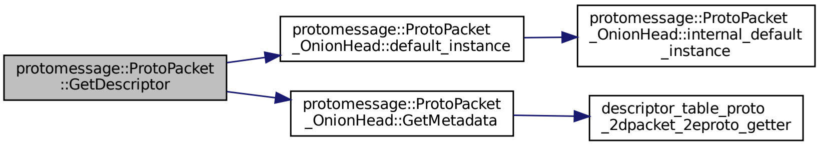
References _class_data_.

3.15.5.24 GetDescriptor() `static const ::PROTOBUF_NAMESPACE_ID::Descriptor* protomessage::↩ ProtoPacket::GetDescriptor ( ) [inline], [static]`

Definition at line 632 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::default_instance(), and protomessage::ProtoPacket_Onion↩ Head::GetMetadata().

Here is the call graph for this function:

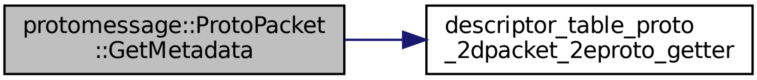

protomessage::ProtoPacket ::GetDescriptor
protomessage::ProtoPacket _OnionHead::default_instance
protomessage::ProtoPacket _OnionHead::internal_default _instance
protomessage::ProtoPacket _OnionHead::GetMetadata
descriptor_table_proto _2dpacket_2eproto_getter

3.15.5.25 GetMetadata() `PROTOBUF_NAMESPACE_ID::Metadata protomessage::ProtoPacket::GetMetadata ( ) const [final]`

Definition at line 1226 of file proto-packet.pb.cc.

References descriptor_table_proto_2dpacket_2eproto_getter(), descriptor_table_proto_2dpacket_2eproto_once, and file_level_metadata_proto_2dpacket_2eproto.

Here is the call graph for this function:

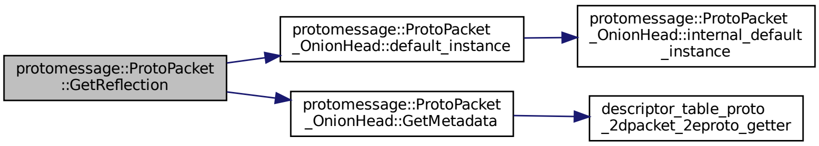

protomessage::ProtoPacket ::GetMetadata
descriptor_table_proto _2dpacket_2eproto_getter

3.15.5.26 GetReflection() `static const ::PROTOBUF_NAMESPACE_ID::Reflection* protomessage::↩ ProtoPacket::GetReflection ( ) [inline], [static]`

Definition at line 635 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::default_instance(), and protomessage::ProtoPacket_Onion↩ Head::GetMetadata().

Here is the call graph for this function:


protomessage::ProtoPacket ::GetReflection
protomessage::ProtoPacket _OnionHead::default_instance
protomessage::ProtoPacket _OnionHead::internal_default _instance
protomessage::ProtoPacket _OnionHead::GetMetadata
descriptor_table_proto _2dpacket_2eproto_getter

3.15.5.27 h_shake()

`const ::protomessage::ProtoPacket_Handshake & protomessage::ProtoPacket::h_shake ( ) const [inline]`

Definition at line 1169 of file proto-packet.pb.h.

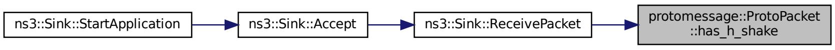
3.15.5.28 has_h_shake()

`bool protomessage::ProtoPacket::has_h_shake ( ) const [inline]`

Definition at line 1157 of file proto-packet.pb.h.

Referenced by ns3::Sink::ReceivePacket().

Here is the caller graph for this function:


ns3::Sink::StartApplication
ns3::Sink::Accept
ns3::Sink::ReceivePacket
protomessage::ProtoPacket ::has_h_shake

3.15.5.29 has_o_body()

`bool protomessage::ProtoPacket::has_o_body ( ) const [inline]`

Definition at line 1337 of file proto-packet.pb.h.

3.15.5.30 has_o_head()

`bool protomessage::ProtoPacket::has_o_head ( ) const [inline]`

Definition at line 1247 of file proto-packet.pb.h.

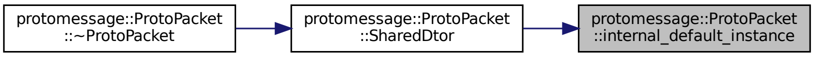
3.15.5.31 internal_default_instance() `static const ProtoPacket* protomessage::ProtoPacket::internal_default_instance ( ) [inline], [static]`

Definition at line 641 of file proto-packet.pb.h.

References protomessage::_ProtoPacket_default_instance_.

Referenced by SharedDtor().

Here is the caller graph for this function:


protomessage::ProtoPacket ::~ProtoPacket
protomessage::ProtoPacket ::SharedDtor
protomessage::ProtoPacket ::internal_default_instance

3.15.5.32 InternalSwap() `void protomessage::ProtoPacket::InternalSwap ( ProtoPacket * other ) [private]`

Definition at line 1214 of file proto-packet.pb.cc.

References _has_bits_, h_shake_, o_body_, and swap.

3.15.5.33 IsInitialized() `bool protomessage::ProtoPacket::IsInitialized ( ) const [final]`

Definition at line 1210 of file proto-packet.pb.cc.

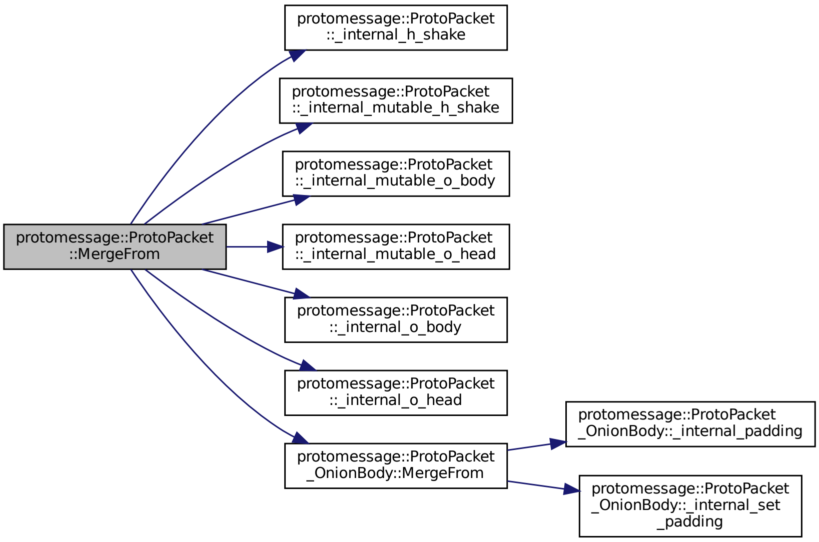
3.15.5.34 MergeFrom() `void protomessage::ProtoPacket::MergeFrom ( const ProtoPacket & from )`

Definition at line 1182 of file proto-packet.pb.cc.

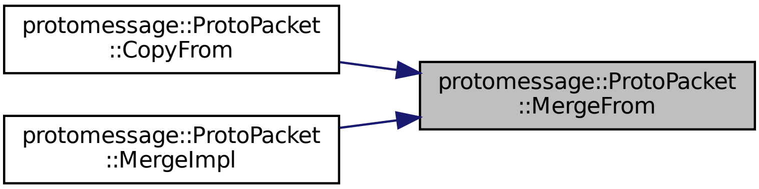
References _has_bits_, _internal_h_shake(), _internal_mutable_h_shake(), _internal_mutable_o_body(), _internal_mutable_o_head(), _internal_o_body(), _internal_o_head(), and protomessage::ProtoPacket_OnionBody::MergeFrom().

Referenced by CopyFrom(), and MergeImpl().

Here is the call graph for this function:


protomessage::ProtoPacket ::MergeFrom
protomessage::ProtoPacket ::_internal_h_shake
protomessage::ProtoPacket ::_internal_mutable_h_shake
protomessage::ProtoPacket ::_internal_mutable_o_body
protomessage::ProtoPacket ::_internal_mutable_o_head
protomessage::ProtoPacket ::_internal_o_body
protomessage::ProtoPacket ::_internal_o_head
protomessage::ProtoPacket _OnionBody::MergeFrom
protomessage::ProtoPacket _OnionBody::_internal_padding
protomessage::ProtoPacket _OnionBody::_internal_set _padding

Here is the caller graph for this function:


protomessage::ProtoPacket ::CopyFrom
protomessage::ProtoPacket ::MergeImpl
protomessage::ProtoPacket ::MergeFrom

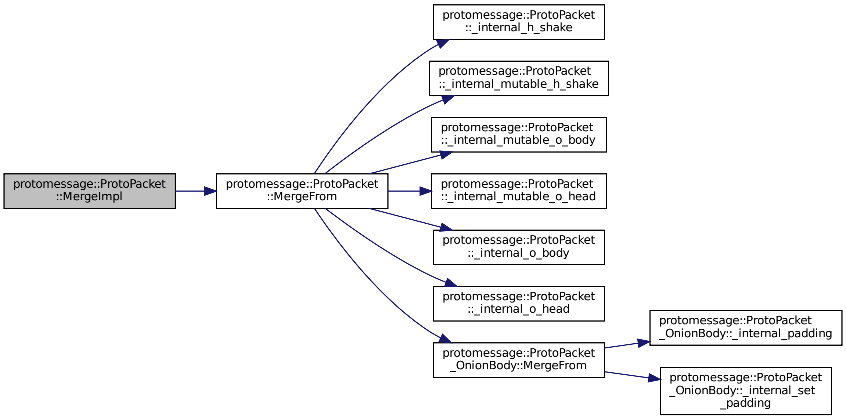
3.15.5.35 MergeImpl()

```
void protomessage::ProtoPacket::MergeImpl (
    ::PROTOBUF_NAMESPACE_ID::Message * to,
    const ::PROTOBUF_NAMESPACE_ID::Message & from )  [static], [private]
```

Definition at line 1175 of file proto-packet.pb.cc.

References MergeFrom().

Here is the call graph for this function:

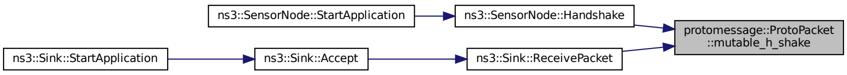

protomessage::ProtoPacket
::MergeImpl
protomessage::ProtoPacket
::MergeFrom
protomessage::ProtoPacket
::_internal_h_shake
protomessage::ProtoPacket
::_internal_mutable_h_shake
protomessage::ProtoPacket
::_internal_mutable_o_body
protomessage::ProtoPacket
::_internal_mutable_o_head
protomessage::ProtoPacket
::_internal_o_body
protomessage::ProtoPacket
::_internal_o_head
protomessage::ProtoPacket
_OnionBody::MergeFrom
protomessage::ProtoPacket
_OnionBody::_internal_padding
protomessage::ProtoPacket
_OnionBody::_internal_set
_padding

3.15.5.36 mutable_h_shake() protomessage::ProtoPacket_Handshake * protomessage::ProtoPacket↩ ::mutable_h_shake ( ) [inline]

Definition at line 1216 of file proto-packet.pb.h.

Referenced by ns3::SensorNode::Handshake(), and ns3::Sink::ReceivePacket().

Here is the caller graph for this function:

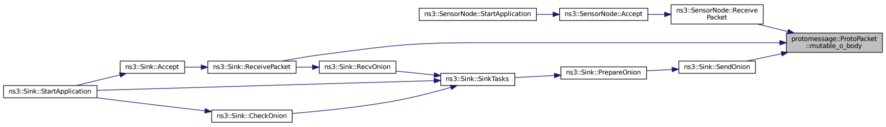

ns3::SensorNode::StartApplication
ns3::SensorNode::Handshake
ns3::Sink::StartApplication
ns3::Sink::Accept
ns3::Sink::ReceivePacket
protomessage::ProtoPacket
::mutable_h_shake

3.15.5.37 mutable_o_body() protomessage::ProtoPacket_OnionBody * protomessage::ProtoPacket↩ ::mutable_o_body ( ) [inline]

Definition at line 1396 of file proto-packet.pb.h.

Referenced by ns3::SensorNode::ReceivePacket(), ns3::Sink::ReceivePacket(), and ns3::Sink::SendOnion().

Here is the caller graph for this function:



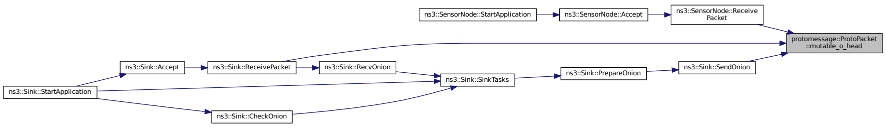
3.15.5.38 mutable_o_head()

protomessage::ProtoPacket_OnionHead ∗ protomessage::ProtoPacket↩ ::mutable_o_head ( ) [inline]

Definition at line 1306 of file proto-packet.pb.h.

Referenced by ns3::SensorNode::ReceivePacket(), ns3::Sink::ReceivePacket(), and ns3::Sink::SendOnion().

Here is the caller graph for this function:



3.15.5.39 New()

ProtoPacket∗ protomessage::ProtoPacket::New (::PROTOBUF_NAMESPACE_ID::Arena ∗ *arena* = *nullptr*) const [inline], [final]

Definition at line 672 of file proto-packet.pb.h.

3.15.5.40 o_body()

const ::protomessage::ProtoPacket_OnionBody & protomessage::ProtoPacket↩ ::o_body ( ) const [inline]

Definition at line 1349 of file proto-packet.pb.h.

3.15.5.41 o_head()

const ::protomessage::ProtoPacket_OnionHead & protomessage::ProtoPacket↩ ::o_head ( ) const [inline]

Definition at line 1259 of file proto-packet.pb.h.

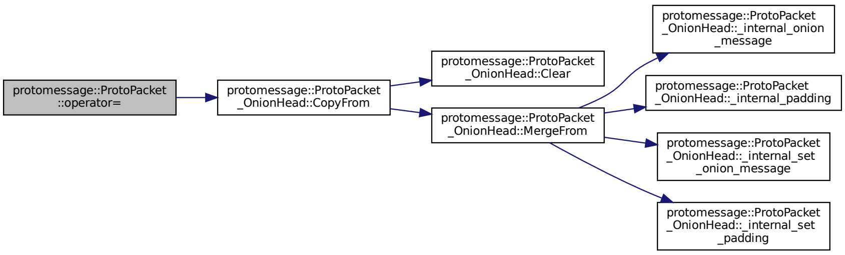
3.15.5.42 operator=() [1/2]

ProtoPacket& protomessage::ProtoPacket::operator= (const ProtoPacket & *from*) [inline]

Definition at line 611 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::CopyFrom().

Here is the call graph for this function:

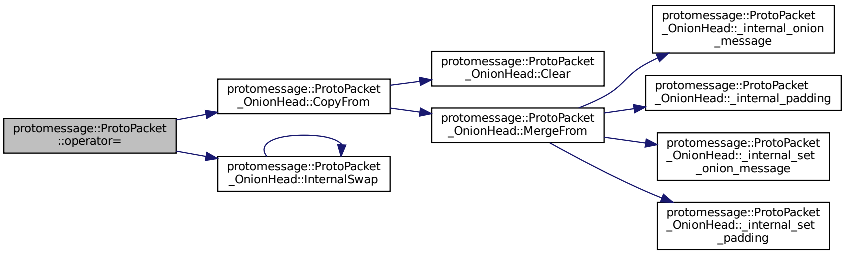

protomessage::ProtoPacket ::operator=
protomessage::ProtoPacket _OnionHead::CopyFrom
protomessage::ProtoPacket _OnionHead::Clear
protomessage::ProtoPacket _OnionHead::MergeFrom
protomessage::ProtoPacket _OnionHead::_internal_onion _message
protomessage::ProtoPacket _OnionHead::_internal_padding
protomessage::ProtoPacket _OnionHead::_internal_set _onion_message
protomessage::ProtoPacket _OnionHead::_internal_set _padding

3.15.5.43 operator=() [2/2] ProtoPacket& protomessage::ProtoPacket::operator= (ProtoPacket && *from*) [inline], [noexcept]

Definition at line 615 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::CopyFrom(), and protomessage::ProtoPacket_OnionHead↩ ::InternalSwap().

Here is the call graph for this function:

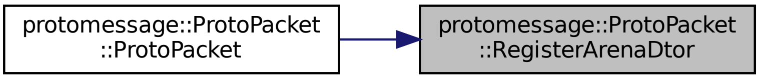

protomessage::ProtoPacket ::operator=
protomessage::ProtoPacket _OnionHead::CopyFrom
protomessage::ProtoPacket _OnionHead::InternalSwap
protomessage::ProtoPacket _OnionHead::Clear
protomessage::ProtoPacket _OnionHead::MergeFrom
protomessage::ProtoPacket _OnionHead::_internal_onion _message
protomessage::ProtoPacket _OnionHead::_internal_padding
protomessage::ProtoPacket _OnionHead::_internal_set _onion_message
protomessage::ProtoPacket _OnionHead::_internal_set _padding

3.15.5.44 RegisterArenaDtor() void protomessage::ProtoPacket::RegisterArenaDtor (::PROTOBUF_NAMESPACE_ID::Arena * *arena*) [inline], [private]

Definition at line 1010 of file proto-packet.pb.cc.

Referenced by ProtoPacket().

Here is the caller graph for this function:


protomessage::ProtoPacket ::ProtoPacket
protomessage::ProtoPacket ::RegisterArenaDtor

3.15.5.45 release_h_shake() protomessage::ProtoPacket_Handshake * protomessage::ProtoPacket↩ ::release_h_shake ( ) [inline]

Definition at line 1186 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.15.5.46 release_o_body() protomessage::ProtoPacket_OnionBody ∗ protomessage::ProtoPacket↩ ::release_o_body ( ) [inline]

Definition at line 1366 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.15.5.47 release_o_head() protomessage::ProtoPacket_OnionHead ∗ protomessage::ProtoPacket↩ ::release_o_head ( ) [inline]

Definition at line 1276 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.15.5.48 set_allocated_h_shake() void protomessage::ProtoPacket::set_allocated_h_shake (::protomessage::ProtoPacket_Handshake ∗ *h_shake*) [inline]

Definition at line 1221 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.15.5.49 set_allocated_o_body() void protomessage::ProtoPacket::set_allocated_o_body (::protomessage::ProtoPacket_OnionBody ∗ *o_body*) [inline]

Definition at line 1401 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.15.5.50 set_allocated_o_head() void protomessage::ProtoPacket::set_allocated_o_head (::protomessage::ProtoPacket_OnionHead ∗ *o_head*) [inline]

Definition at line 1311 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.15.5.51 SetCachedSize() void protomessage::ProtoPacket::SetCachedSize (int *size*) const [final], [private]

Definition at line 1012 of file proto-packet.pb.cc.

References _cached_size_.

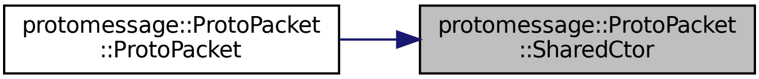
3.15.5.52 SharedCtor() `void protomessage::ProtoPacket::SharedCtor ( ) [inline], [private]`

Definition at line 985 of file proto-packet.pb.cc.

References h_shake_, and o_body_.

Referenced by ProtoPacket().

Here is the caller graph for this function:


protomessage::ProtoPacket ::ProtoPacket
protomessage::ProtoPacket ::SharedCtor

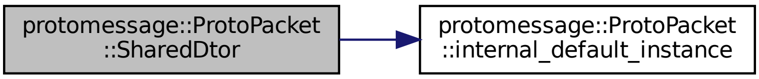
3.15.5.53 SharedDtor() `void protomessage::ProtoPacket::SharedDtor ( ) [inline], [private]`

Definition at line 999 of file proto-packet.pb.cc.

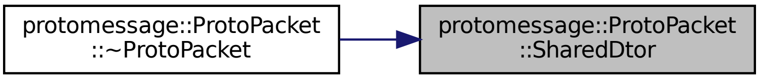
References h_shake_, internal_default_instance(), o_body_, and o_head_.

Referenced by ∼ProtoPacket().

Here is the call graph for this function:


protomessage::ProtoPacket ::SharedDtor
protomessage::ProtoPacket ::internal_default_instance

Here is the caller graph for this function:

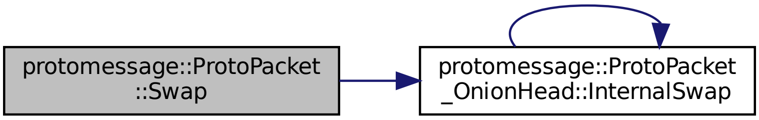

protomessage::ProtoPacket ::∼ProtoPacket
protomessage::ProtoPacket ::SharedDtor

3.15.5.54 Swap() `void protomessage::ProtoPacket::Swap ( ProtoPacket * other ) [inline]`

Definition at line 651 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::InternalSwap().

Here is the call graph for this function:



3.15.5.55 unsafe_arena_release_h_shake() `protomessage::ProtoPacket_Handshake * protomessage::↩ ProtoPacket::unsafe_arena_release_h_shake ( ) [inline]`

Definition at line 1201 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.15.5.56 unsafe_arena_release_o_body() `protomessage::ProtoPacket_OnionBody * protomessage::↩ ProtoPacket::unsafe_arena_release_o_body ( ) [inline]`

Definition at line 1381 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.15.5.57 unsafe_arena_release_o_head() `protomessage::ProtoPacket_OnionHead * protomessage::↩ ProtoPacket::unsafe_arena_release_o_head ( ) [inline]`

Definition at line 1291 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.15.5.58 unsafe_arena_set_allocated_h_shake() `void protomessage::ProtoPacket::unsafe_arena_↩ set_allocated_h_shake ( ::protomessage::ProtoPacket_Handshake * h_shake ) [inline]`

Definition at line 1173 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.15.5.59 unsafe_arena_set_allocated_o_body() `void protomessage::ProtoPacket::unsafe_arena_↩ set_allocated_o_body ( ::protomessage::ProtoPacket_OnionBody * o_body ) [inline]`

Definition at line 1353 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.15.5.60 unsafe_arena_set_allocated_o_head() `void protomessage::ProtoPacket::unsafe_arena_↩ set_allocated_o_head ( ::protomessage::ProtoPacket_OnionHead * o_head ) [inline]`

Definition at line 1263 of file proto-packet.pb.h.

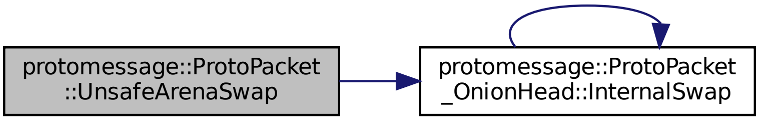
References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.15.5.61 UnsafeArenaSwap() `void protomessage::ProtoPacket::UnsafeArenaSwap ( ProtoPacket * other ) [inline]`

Definition at line 664 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::InternalSwap().

Here is the call graph for this function:


protomessage::ProtoPacket ::UnsafeArenaSwap
protomessage::ProtoPacket _OnionHead::InternalSwap

3.15.6 Friends And Related Function Documentation

3.15.6.1 ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper

`template<typename T > friend class ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper [friend]`

Definition at line 784 of file proto-packet.pb.h.

3.15.6.2 ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata

`friend class ::PROTOBUF_NAMESPACE_↩ ID::internal::AnyMetadata [friend]`

Definition at line 698 of file proto-packet.pb.h.

3.15.6.3 ::TableStruct_proto_2dpacket_2eproto

`friend struct ::TableStruct_proto_2dpacket_2eproto [friend]`

Definition at line 794 of file proto-packet.pb.h.

3.15.6.4 swap

```
void swap (
            ProtoPacket & a,
            ProtoPacket & b ) [friend]
```

Definition at line 648 of file proto-packet.pb.h.

Referenced by InternalSwap().

3.15.7 Member Data Documentation

3.15.7.1 _cached_size_

`mutable ::PROTOBUF_NAMESPACE_ID::internal::CachedSize protomessage::↩ ProtoPacket::_cached_size_ [private]`

Definition at line 790 of file proto-packet.pb.h.

Referenced by ByteSizeLong(), and SetCachedSize().

3.15.7.2 _class_data_

`const ::PROTOBUF_NAMESPACE_ID::Message::ClassData protomessage::Proto↩ Packet::_class_data_ [static]`

Initial value:

```
= {
    ::PROTOBUF_NAMESPACE_ID::Message::CopyWithSizeCheck,
    ProtoPacket::MergeImpl
}
```

Definition at line 710 of file proto-packet.pb.h.

Referenced by GetClassData().

3.15.7.3 _has_bits_ `::PROTOBUF_NAMESPACE_ID::internal::HasBits<1> protomessage::ProtoPacket↩ ::_has_bits_ [private]`

Definition at line 789 of file proto-packet.pb.h.

Referenced by _InternalParse(), ByteSizeLong(), Clear(), InternalSwap(), and MergeFrom().

3.15.7.4 h_shake_ `::protomessage::ProtoPacket_Handshake* protomessage::ProtoPacket::h_shake_ [private]`

Definition at line 791 of file proto-packet.pb.h.

Referenced by ByteSizeLong(), Clear(), protomessage::ProtoPacket::_Internal::h_shake(), InternalSwap(), ProtoPacket(), SharedCtor(), and SharedDtor().

3.15.7.5 kIndexInFileMessages `constexpr int protomessage::ProtoPacket::kIndexInFileMessages [static], [constexpr]`

Initial value:

```
=
    3
```

Definition at line 645 of file proto-packet.pb.h.

3.15.7.6 o_body_ `::protomessage::ProtoPacket_OnionBody* protomessage::ProtoPacket::o_body_↩ [private]`

Definition at line 793 of file proto-packet.pb.h.

Referenced by ByteSizeLong(), Clear(), InternalSwap(), protomessage::ProtoPacket::_Internal::o_body(), ProtoPacket(), SharedCtor(), and SharedDtor().

3.15.7.7 o_head_ `::protomessage::ProtoPacket_OnionHead* protomessage::ProtoPacket::o_head_↩ [private]`

Definition at line 792 of file proto-packet.pb.h.

Referenced by ByteSizeLong(), Clear(), protomessage::ProtoPacket::_Internal::o_head(), ProtoPacket(), and SharedDtor().

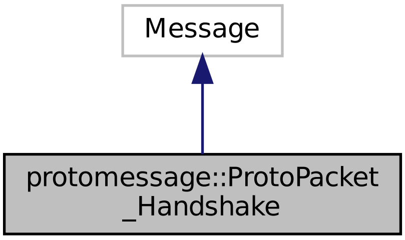
The documentation for this class was generated from the following files:

- src/onion_routing_wsn/protobuf/proto-packet.pb.h
- src/onion_routing_wsn/protobuf/proto-packet.pb.cc

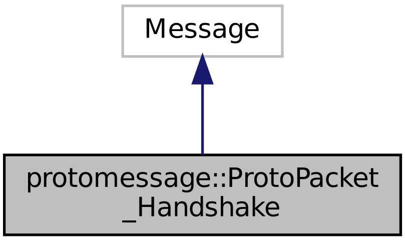
3.16 protomessage::ProtoPacket_Handshake Class Reference

```
#include "proto-packet.pb.h"
```

Inheritance diagram for protomessage::ProtoPacket_Handshake:



Collaboration diagram for protomessage::ProtoPacket_Handshake:



Classes

- class _Internal

Public Types

- enum : int { kPublickeyFieldNumber = 1 }

Public Member Functions

- ProtoPacket_Handshake ()
- constexpr ProtoPacket_Handshake (::PROTOBUF_NAMESPACE_ID::internal::ConstantInitialized)
- ProtoPacket_Handshake (const ProtoPacket_Handshake &from)
- ProtoPacket_Handshake (ProtoPacket_Handshake &&from) noexcept
- ∼ProtoPacket_Handshake () override
- const char ∗ _InternalParse (const char ∗ptr, ::PROTOBUF_NAMESPACE_ID::internal::ParseContext ∗ctx) final
- uint8_t ∗ _InternalSerialize (uint8_t ∗target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream ∗stream) const final
- size_t ByteSizeLong () const final
- PROTOBUF_ATTRIBUTE_REINITIALIZES void Clear () final
- void clear_publickey ()
- void CopyFrom (const ProtoPacket_Handshake &from)
- int GetCachedSize () const final
- const ::PROTOBUF_NAMESPACE_ID::Message::ClassData ∗ GetClassData () const final
- ::PROTOBUF_NAMESPACE_ID::Metadata GetMetadata () const final
- bool has_publickey () const
- bool IsInitialized () const final
- void MergeFrom (const ProtoPacket_Handshake &from)
- std::string ∗ mutable_publickey ()
- ProtoPacket_Handshake ∗ New (::PROTOBUF_NAMESPACE_ID::Arena ∗arena=nullptr) const final
- ProtoPacket_Handshake & operator= (const ProtoPacket_Handshake &from)
- ProtoPacket_Handshake & operator= (ProtoPacket_Handshake &&from) noexcept
- const std::string & publickey () const
- PROTOBUF_NODISCARD std::string ∗ release_publickey ()
- void set_allocated_publickey (std::string ∗publickey)
- template<typename ArgT0 = const std::string&, typename... ArgT>
  void set_publickey (ArgT0 &&arg0, ArgT... args)
- template<typename ArgT0 , typename... ArgT>
  PROTOBUF_ALWAYS_INLINE void set_publickey (ArgT0 &&arg0, ArgT... args)
- void Swap (ProtoPacket_Handshake ∗other)
- void UnsafeArenaSwap (ProtoPacket_Handshake ∗other)

Static Public Member Functions

- static const ProtoPacket_Handshake & default_instance ()
- static const ::PROTOBUF_NAMESPACE_ID::Descriptor ∗ descriptor ()
- static const ::PROTOBUF_NAMESPACE_ID::Descriptor ∗ GetDescriptor ()
- static const ::PROTOBUF_NAMESPACE_ID::Reflection ∗ GetReflection ()
- static const ProtoPacket_Handshake ∗ internal_default_instance ()

Static Public Attributes

- static const ClassData _class_data_
- static constexpr int kIndexInFileMessages

Protected Member Functions

- ProtoPacket_Handshake (::PROTOBUF_NAMESPACE_ID::Arena ∗arena, bool is_message_owned=false)

Private Types

- typedef void DestructorSkippable_
- typedef void InternalArenaConstructable_

Private Member Functions

- bool _internal_has_publickey () const
- std::string ∗ _internal_mutable_publickey ()
- const std::string & _internal_publickey () const
- PROTOBUF_ALWAYS_INLINE void _internal_set_publickey (const std::string &value)
- void InternalSwap (ProtoPacket_Handshake ∗other)
- void RegisterArenaDtor (::PROTOBUF_NAMESPACE_ID::Arena ∗arena)
- void SetCachedSize (int size) const final
- void SharedCtor ()
- void SharedDtor ()

Static Private Member Functions

- static void ArenaDtor (void ∗object)
- ::PROTOBUF_NAMESPACE_ID::StringPiece FullMessageName ()
- static void MergeImpl (::PROTOBUF_NAMESPACE_ID::Message ∗to, const ::PROTOBUF_NAMESPACE↩ _ID::Message &from)

Private Attributes

- mutable ::PROTOBUF_NAMESPACE_ID::internal::CachedSize _cached_size_
- ::PROTOBUF_NAMESPACE_ID::internal::HasBits< 1 > _has_bits_
- ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr publickey_

Friends

- template<typename T >
  class ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper
- class ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata
- struct ::TableStruct_proto_2dpacket_2eproto
- void swap (ProtoPacket_Handshake &a, ProtoPacket_Handshake &b)

3.16.1 Detailed Description

Definition at line 442 of file proto-packet.pb.h.

3.16.2 Member Typedef Documentation

3.16.2.1 DestructorSkippable_ `typedef void protomessage::ProtoPacket_Handshake::DestructorSkippable_ [private]`

Definition at line 590 of file proto-packet.pb.h.

3.16.2.2 InternalArenaConstructable_ `typedef void protomessage::ProtoPacket_Handshake::InternalArenaConstructab [private]`

Definition at line 589 of file proto-packet.pb.h.

3.16.3 Member Enumeration Documentation

3.16.3.1 anonymous enum `anonymous enum : int`

Enumerator

kPublickeyFieldNumber	

Definition at line 563 of file proto-packet.pb.h.

3.16.4 Constructor & Destructor Documentation

3.16.4.1 ProtoPacket_Handshake() [1/5] `protomessage::ProtoPacket_Handshake::ProtoPacket_↩ Handshake ( ) [inline]`

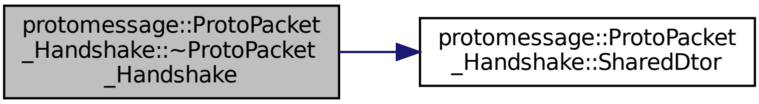
Definition at line 445 of file proto-packet.pb.h.

3.16.4.2 ∼ProtoPacket_Handshake() `protomessage::ProtoPacket_Handshake::∼ProtoPacket_Handshake ( ) [override]`

Definition at line 753 of file proto-packet.pb.cc.

References SharedDtor().

Here is the call graph for this function:



3.16.4.3 ProtoPacket_Handshake() [2/5] constexpr protomessage::ProtoPacket_Handshake::ProtoPacket_Handshake (::PROTOBUF_NAMESPACE_ID::internal::ConstantInitialized) [explicit], [constexpr]

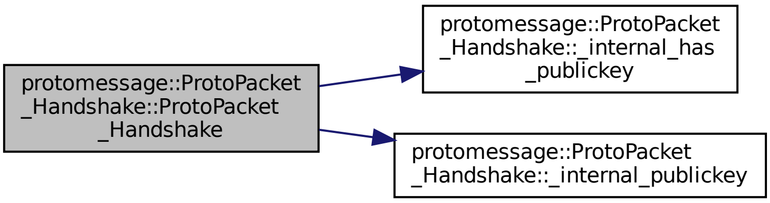
Definition at line 47 of file proto-packet.pb.cc.

3.16.4.4 ProtoPacket_Handshake() [3/5] protomessage::ProtoPacket_Handshake::ProtoPacket_Handshake (const ProtoPacket_Handshake & *from*)

Definition at line 731 of file proto-packet.pb.cc.

References _internal_has_publickey(), _internal_publickey(), and publickey_.

Here is the call graph for this function:


protomessage::ProtoPacket
_Handshake::ProtoPacket
_Handshake
protomessage::ProtoPacket
_Handshake::_internal_has
_publickey
protomessage::ProtoPacket
_Handshake::_internal_publickey

3.16.4.5 ProtoPacket_Handshake() [4/5] protomessage::ProtoPacket_Handshake::ProtoPacket_Handshake (ProtoPacket_Handshake && *from*) [inline], [noexcept]

Definition at line 450 of file proto-packet.pb.h.

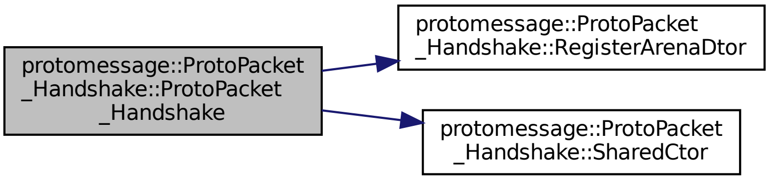
3.16.4.6 ProtoPacket_Handshake() [5/5] protomessage::ProtoPacket_Handshake::ProtoPacket_↩ Handshake (

```
Handshake (
            ::PROTOBUF_NAMESPACE_ID::Arena * arena,
            bool is_message_owned = false )  [explicit], [protected]
```

Definition at line 722 of file proto-packet.pb.cc.

References RegisterArenaDtor(), and SharedCtor().

Here is the call graph for this function:


protomessage::ProtoPacket _Handshake::ProtoPacket _Handshake
protomessage::ProtoPacket _Handshake::RegisterArenaDtor
protomessage::ProtoPacket _Handshake::SharedCtor

3.16.5 Member Function Documentation

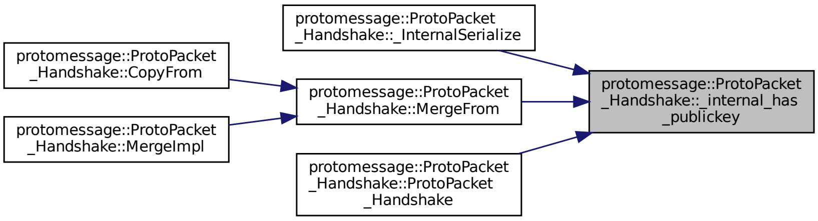
3.16.5.1 _internal_has_publickey() bool protomessage::ProtoPacket_Handshake::_internal_has_↩ publickey ( ) const [inline], [private]

Definition at line 1079 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

Referenced by _InternalSerialize(), MergeFrom(), and ProtoPacket_Handshake().

Here is the caller graph for this function:


protomessage::ProtoPacket _Handshake::_InternalSerialize
protomessage::ProtoPacket _Handshake::CopyFrom
protomessage::ProtoPacket _Handshake::MergeFrom
protomessage::ProtoPacket _Handshake::_internal_has _publickey
protomessage::ProtoPacket _Handshake::MergeImpl
protomessage::ProtoPacket _Handshake::ProtoPacket _Handshake

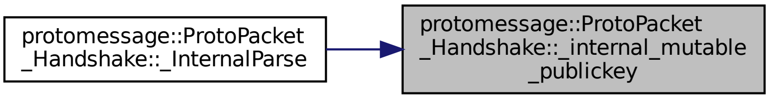
3.16.5.2 _internal_mutable_publickey() `std::string * protomessage::ProtoPacket_Handshake::_↩ internal_mutable_publickey ( ) [inline], [private]`

Definition at line 1113 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

Referenced by _InternalParse().

Here is the caller graph for this function:

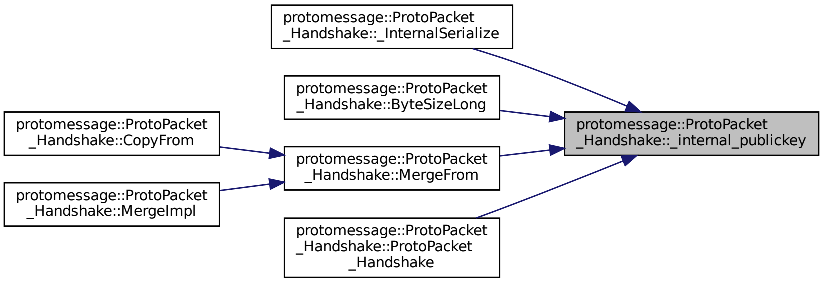

protomessage::ProtoPacket
_Handshake::_InternalParse
protomessage::ProtoPacket
_Handshake::_internal_mutable
_publickey

3.16.5.3 _internal_publickey() `const std::string & protomessage::ProtoPacket_Handshake::_↩ internal_publickey ( ) const [inline], [private]`

Definition at line 1106 of file proto-packet.pb.h.

Referenced by _InternalSerialize(), ByteSizeLong(), MergeFrom(), and ProtoPacket_Handshake().

Here is the caller graph for this function:


protomessage::ProtoPacket
_Handshake::_InternalSerialize
protomessage::ProtoPacket
_Handshake::ByteSizeLong
protomessage::ProtoPacket
_Handshake::CopyFrom
protomessage::ProtoPacket
_Handshake::MergeFrom
protomessage::ProtoPacket
_Handshake::MergeImpl
protomessage::ProtoPacket
_Handshake::ProtoPacket
_Handshake
protomessage::ProtoPacket
_Handshake::_internal_publickey

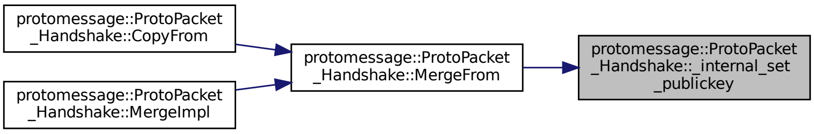
3.16.5.4 _internal_set_publickey() `void protomessage::ProtoPacket_Handshake::_internal_set_↩ publickey ( const std::string & value ) [inline], [private]`

Definition at line 1109 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

Referenced by MergeFrom().

Here is the caller graph for this function:

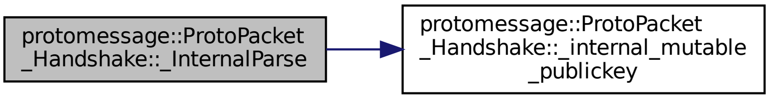

protomessage::ProtoPacket _Handshake::CopyFrom
protomessage::ProtoPacket _Handshake::MergeImpl
protomessage::ProtoPacket _Handshake::MergeFrom
protomessage::ProtoPacket _Handshake::_internal_set _publickey

3.16.5.5 _InternalParse() `const char * protomessage::ProtoPacket_Handshake::_InternalParse ( const char * ptr, ::PROTOBUF_NAMESPACE_ID::internal::ParseContext * ctx ) [final]`

Definition at line 789 of file proto-packet.pb.cc.

References _has_bits_, _internal_mutable_publickey(), and CHK_.

Here is the call graph for this function:

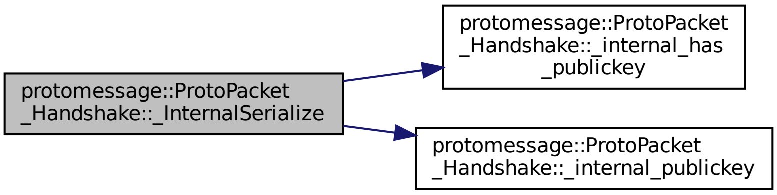

protomessage::ProtoPacket _Handshake::_InternalParse
protomessage::ProtoPacket _Handshake::_internal_mutable _publickey

3.16.5.6 _InternalSerialize() `uint8_t * protomessage::ProtoPacket_Handshake::_InternalSerialize ( uint8_t * target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream * stream ) const [final]`

Definition at line 829 of file proto-packet.pb.cc.

References _internal_has_publickey(), and _internal_publickey().

Here is the call graph for this function:


protomessage::ProtoPacket _Handshake::_InternalSerialize
protomessage::ProtoPacket _Handshake::_internal_has _publickey
protomessage::ProtoPacket _Handshake::_internal_publickey

3.16.5.7 ArenaDtor() `void protomessage::ProtoPacket_Handshake::ArenaDtor ( void * object ) [static], [private]`

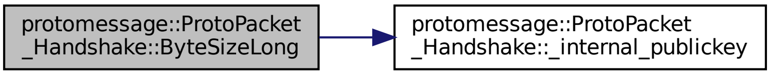
Definition at line 765 of file proto-packet.pb.cc.

3.16.5.8 ByteSizeLong() `size_t protomessage::ProtoPacket_Handshake::ByteSizeLong ( ) const [final]`

Definition at line 849 of file proto-packet.pb.cc.

References _cached_size_, _has_bits_, and _internal_publickey().

Here is the call graph for this function:


protomessage::ProtoPacket _Handshake::ByteSizeLong
protomessage::ProtoPacket _Handshake::_internal_publickey

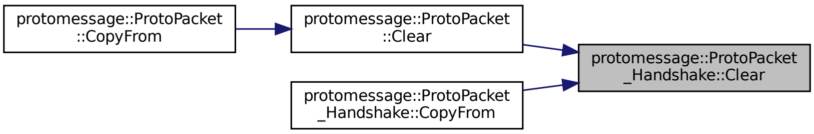
3.16.5.9 Clear() `void protomessage::ProtoPacket_Handshake::Clear ( ) [final]`

Definition at line 775 of file proto-packet.pb.cc.

References _has_bits_, and publickey_.

Referenced by protomessage::ProtoPacket::Clear(), and CopyFrom().

Here is the caller graph for this function:


protomessage::ProtoPacket ::CopyFrom
protomessage::ProtoPacket ::Clear
protomessage::ProtoPacket _Handshake::CopyFrom
protomessage::ProtoPacket _Handshake::Clear

3.16.5.10 clear_publickey() `void protomessage::ProtoPacket_Handshake::clear_publickey ( ) [inline]`

Definition at line 1086 of file proto-packet.pb.h.

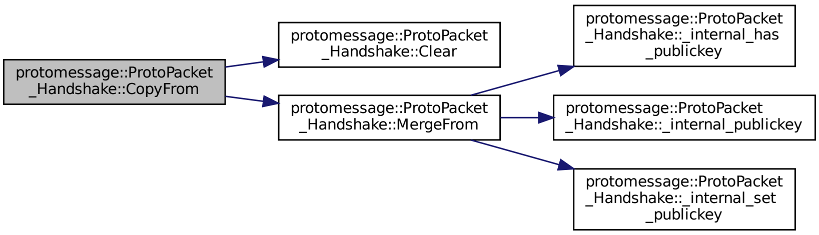
References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.16.5.11 CopyFrom() `void protomessage::ProtoPacket_Handshake::CopyFrom ( const ProtoPacket_Handshake & from )`

Definition at line 893 of file proto-packet.pb.cc.

References Clear(), and MergeFrom().

Here is the call graph for this function:


protomessage::ProtoPacket _Handshake::CopyFrom
protomessage::ProtoPacket _Handshake::Clear
protomessage::ProtoPacket _Handshake::MergeFrom
protomessage::ProtoPacket _Handshake::_internal_has _publickey
protomessage::ProtoPacket _Handshake::_internal_publickey
protomessage::ProtoPacket _Handshake::_internal_set _publickey

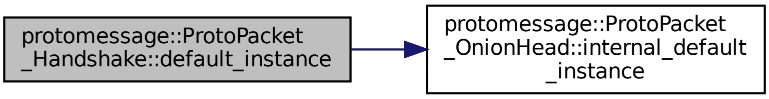
3.16.5.12 default_instance()

`static const ProtoPacket_Handshake& protomessage::ProtoPacket_Handshake::default_instance ( ) [inline], [static]`

Definition at line 482 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::internal_default_instance().

Here is the call graph for this function:


protomessage::ProtoPacket
_Handshake::default_instance
protomessage::ProtoPacket
_OnionHead::internal_default
_instance

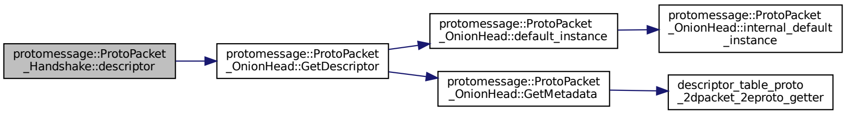
3.16.5.13 descriptor()

`static const ::PROTOBUF_NAMESPACE_ID::Descriptor* protomessage::ProtoPacket_Handshake::descriptor ( ) [inline], [static]`

Definition at line 473 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::GetDescriptor().

Here is the call graph for this function:


protomessage::ProtoPacket
_Handshake::descriptor
protomessage::ProtoPacket
_OnionHead::GetDescriptor
protomessage::ProtoPacket
_OnionHead::default_instance
protomessage::ProtoPacket
_OnionHead::internal_default
_instance
protomessage::ProtoPacket
_OnionHead::GetMetadata
descriptor_table_proto
_2dpacket_2eproto_getter

3.16.5.14 FullMessageName()

`::PROTOBUF_NAMESPACE_ID::StringPiece protomessage::ProtoPacket_Handshake::FullMessageName ( ) [inline], [static], [private]`

Definition at line 543 of file proto-packet.pb.h.

3.16.5.15 GetCachedSize()

`int protomessage::ProtoPacket_Handshake::GetCachedSize ( ) const [inline], [final]`

Definition at line 533 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_cached_size_.

3.16.5.16 GetClassData() `const ::PROTOBUF_NAMESPACE_ID::Message::ClassData * protomessage::ProtoPacket_Handshake::GetClassData ( ) const [final]`

Definition at line 872 of file proto-packet.pb.cc.

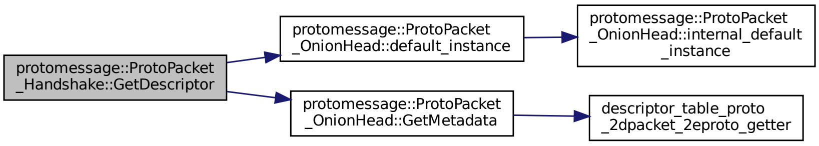
References _class_data_.

3.16.5.17 GetDescriptor() `static const ::PROTOBUF_NAMESPACE_ID::Descriptor* protomessage::ProtoPacket_Handshake::GetDescriptor ( ) [inline], [static]`

Definition at line 476 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::default_instance(), and protomessage::ProtoPacket_OnionHead::GetMetadata().

Here is the call graph for this function:

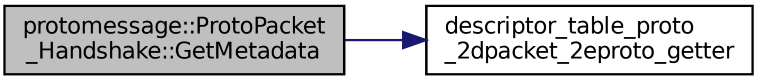

protomessage::ProtoPacket
_Handshake::GetDescriptor
protomessage::ProtoPacket
_OnionHead::default_instance
protomessage::ProtoPacket
_OnionHead::internal_default
_instance
protomessage::ProtoPacket
_OnionHead::GetMetadata
descriptor_table_proto
_2dpacket_2eproto_getter

3.16.5.18 GetMetadata() `PROTOBUF_NAMESPACE_ID::Metadata protomessage::ProtoPacket_Handshake::GetMetadata ( ) const [final]`

Definition at line 917 of file proto-packet.pb.cc.

References descriptor_table_proto_2dpacket_2eproto_getter(), descriptor_table_proto_2dpacket_2eproto_once, and file_level_metadata_proto_2dpacket_2eproto.

Here is the call graph for this function:

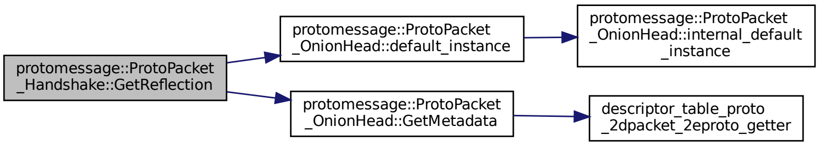


3.16.5.19 GetReflection() `static const ::PROTOBUF_NAMESPACE_ID::Reflection* protomessage::ProtoPacket_Handshake::GetReflection ( ) [inline], [static]`

Definition at line 479 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::default_instance(), and protomessage::ProtoPacket_OnionHead::GetMetadata().

Here is the call graph for this function:


protomessage::ProtoPacket _Handshake::GetReflection
protomessage::ProtoPacket _OnionHead::default_instance
protomessage::ProtoPacket _OnionHead::internal_default _instance
protomessage::ProtoPacket _OnionHead::GetMetadata
descriptor_table_proto _2dpacket_2eproto_getter

3.16.5.20 has_publickey() `bool protomessage::ProtoPacket_Handshake::has_publickey ( ) const [inline]`

Definition at line 1083 of file proto-packet.pb.h.

3.16.5.21 internal_default_instance() `static const ProtoPacket_Handshake* protomessage::ProtoPacket_Handshake::internal_default_instance ( ) [inline], [static]`

Definition at line 485 of file proto-packet.pb.h.

References protomessage::_ProtoPacket_Handshake_default_instance_.

3.16.5.22 InternalSwap() `void protomessage::ProtoPacket_Handshake::InternalSwap ( ProtoPacket_Handshake * other ) [private]`

Definition at line 904 of file proto-packet.pb.cc.

References _has_bits_, InternalSwap(), publickey_, and swap.

Referenced by InternalSwap().

Here is the call graph for this function:

protomessage::ProtoPacket
_Handshake::InternalSwap

Here is the caller graph for this function:

protomessage::ProtoPacket
_Handshake::InternalSwap

3.16.5.23 IsInitialized() `bool protomessage::ProtoPacket_Handshake::IsInitialized ( ) const [final]`

Definition at line 900 of file proto-packet.pb.cc.

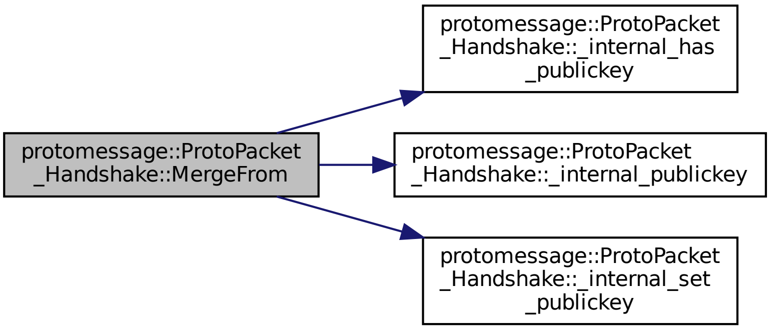
3.16.5.24 MergeFrom() `void protomessage::ProtoPacket_Handshake::MergeFrom ( const ProtoPacket_Handshake & from )`

Definition at line 881 of file proto-packet.pb.cc.

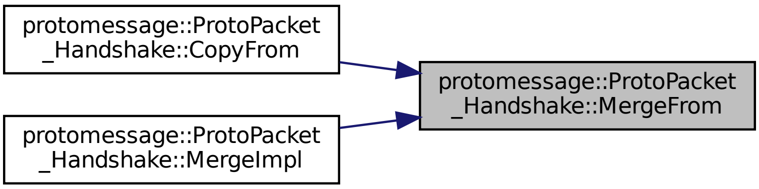
References _internal_has_publickey(), _internal_publickey(), and _internal_set_publickey().

Referenced by CopyFrom(), and MergeImpl().

Here is the call graph for this function:


protomessage::ProtoPacket _Handshake::MergeFrom
protomessage::ProtoPacket _Handshake::_internal_has _publickey
protomessage::ProtoPacket _Handshake::_internal_publickey
protomessage::ProtoPacket _Handshake::_internal_set _publickey

Here is the caller graph for this function:


protomessage::ProtoPacket _Handshake::CopyFrom
protomessage::ProtoPacket _Handshake::MergeImpl
protomessage::ProtoPacket _Handshake::MergeFrom

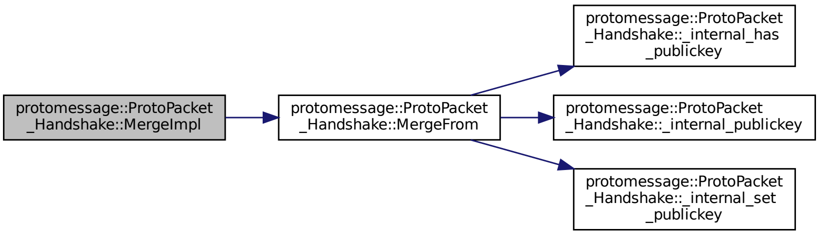
3.16.5.25 MergeImpl()

```
void protomessage::ProtoPacket_Handshake::MergeImpl (
    ::PROTOBUF_NAMESPACE_ID::Message * to,
    const ::PROTOBUF_NAMESPACE_ID::Message & from ) [static], [private]
```

Definition at line 874 of file proto-packet.pb.cc.

References MergeFrom().

Here is the call graph for this function:


protomessage::ProtoPacket _Handshake::MergeImpl
protomessage::ProtoPacket _Handshake::MergeFrom
protomessage::ProtoPacket _Handshake::_internal_has _publickey
protomessage::ProtoPacket _Handshake::_internal_publickey
protomessage::ProtoPacket _Handshake::_internal_set _publickey

3.16.5.26 mutable_publickey() std::string * protomessage::ProtoPacket_Handshake::mutable_↩ publickey ( ) [inline]

Definition at line 1101 of file proto-packet.pb.h.

3.16.5.27 New() ProtoPacket_Handshake* protomessage::ProtoPacket_Handshake::New (::PROTOBUF_NAMESPACE_ID::Arena * *arena* = *nullptr*) const [inline], [final]

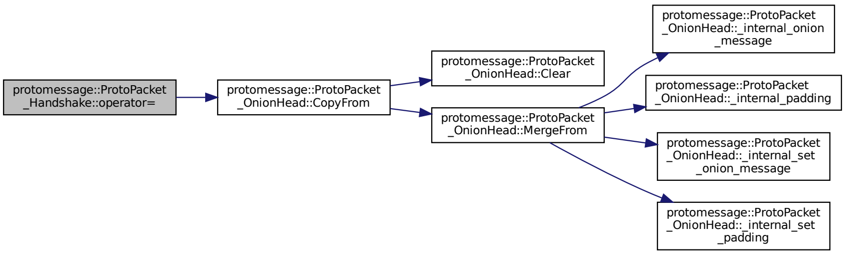
Definition at line 516 of file proto-packet.pb.h.

3.16.5.28 operator=() [1/2] ProtoPacket_Handshake& protomessage::ProtoPacket_Handshake::operator= (const ProtoPacket_Handshake & *from*) [inline]

Definition at line 455 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::CopyFrom().

Here is the call graph for this function:


protomessage::ProtoPacket_Handshake::operator=
protomessage::ProtoPacket_OnionHead::CopyFrom
protomessage::ProtoPacket_OnionHead::Clear
protomessage::ProtoPacket_OnionHead::MergeFrom
protomessage::ProtoPacket_OnionHead::_internal_onion_message
protomessage::ProtoPacket_OnionHead::_internal_padding
protomessage::ProtoPacket_OnionHead::_internal_set_onion_message
protomessage::ProtoPacket_OnionHead::_internal_set_padding

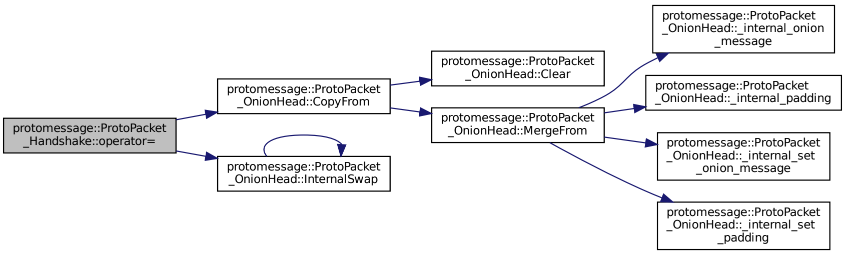
3.16.5.29 operator=() [2/2] ProtoPacket_Handshake& protomessage::ProtoPacket_Handshake::operator= (

ProtoPacket_Handshake && *from*) [inline], [noexcept]

Definition at line 459 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::CopyFrom(), and protomessage::ProtoPacket_OnionHead↩ ::InternalSwap().

Here is the call graph for this function:

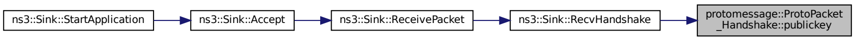

protomessage::ProtoPacket_Handshake::operator=
protomessage::ProtoPacket_OnionHead::CopyFrom
protomessage::ProtoPacket_OnionHead::InternalSwap
protomessage::ProtoPacket_OnionHead::Clear
protomessage::ProtoPacket_OnionHead::MergeFrom
protomessage::ProtoPacket_OnionHead::_internal_onion_message
protomessage::ProtoPacket_OnionHead::_internal_padding
protomessage::ProtoPacket_OnionHead::_internal_set_onion_message
protomessage::ProtoPacket_OnionHead::_internal_set_padding

3.16.5.30 publickey() const std::string & protomessage::ProtoPacket_Handshake::publickey ( ) const [inline]

Definition at line 1090 of file proto-packet.pb.h.

Referenced by ns3::Sink::RecvHandshake().

Here is the caller graph for this function:

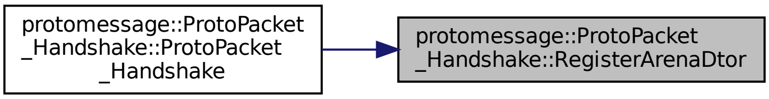

ns3::Sink::StartApplication
ns3::Sink::Accept
ns3::Sink::ReceivePacket
ns3::Sink::RecvHandshake
protomessage::ProtoPacket_Handshake::publickey

3.16.5.31 RegisterArenaDtor() void protomessage::ProtoPacket_Handshake::RegisterArenaDtor (::PROTOBUF_NAMESPACE_ID::Arena * *arena*) [inline], [private]

Definition at line 769 of file proto-packet.pb.cc.

Referenced by ProtoPacket_Handshake().

Here is the caller graph for this function:


protomessage::ProtoPacket_Handshake::ProtoPacket_Handshake
protomessage::ProtoPacket_Handshake::RegisterArenaDtor

3.16.5.32 release_publickey() `std::string * protomessage::ProtoPacket_Handshake::release_↩ publickey ( ) [inline]`

Definition at line 1117 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.16.5.33 set_allocated_publickey() `void protomessage::ProtoPacket_Handshake::set_allocated_↩ publickey ( std::string * publickey ) [inline]`

Definition at line 1131 of file proto-packet.pb.h.

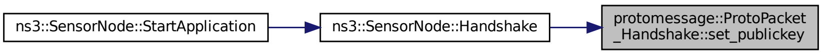
References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.16.5.34 set_publickey() [1/2]

```
template<typename ArgT0 = const std::string&, typename... ArgT>
void protomessage::ProtoPacket_Handshake::set_publickey (
            ArgT0 && arg0,
            ArgT... args )
```

Referenced by ns3::SensorNode::Handshake().

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Handshake
protomessage::ProtoPacket
_Handshake::set_publickey

3.16.5.35 set_publickey() [2/2]

```
template<typename ArgT0 , typename... ArgT>
PROTOBUF_ALWAYS_INLINE void protomessage::ProtoPacket_Handshake::set_publickey (
            ArgT0 && arg0,
            ArgT... args ) [inline]
```

Definition at line 1096 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.16.5.36 SetCachedSize()

```
void protomessage::ProtoPacket_Handshake::SetCachedSize (
            int size ) const [final], [private]
```

Definition at line 771 of file proto-packet.pb.cc.

References _cached_size_.

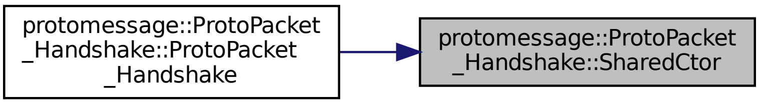
3.16.5.37 SharedCtor() `void protomessage::ProtoPacket_Handshake::SharedCtor ( ) [inline], [private]`

Definition at line 746 of file proto-packet.pb.cc.

References publickey_.

Referenced by ProtoPacket_Handshake().

Here is the caller graph for this function:



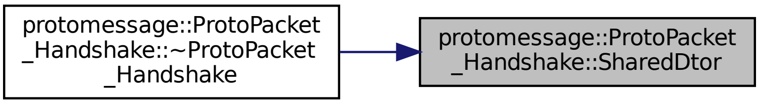
3.16.5.38 SharedDtor() `void protomessage::ProtoPacket_Handshake::SharedDtor ( ) [inline], [private]`

Definition at line 760 of file proto-packet.pb.cc.

References publickey_.

Referenced by ∼ProtoPacket_Handshake().

Here is the caller graph for this function:

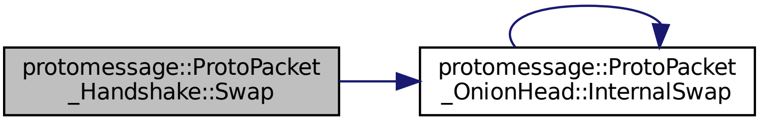


3.16.5.39 Swap() `void protomessage::ProtoPacket_Handshake::Swap ( ProtoPacket_Handshake * other ) [inline]`

Definition at line 495 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::InternalSwap().

Here is the call graph for this function:

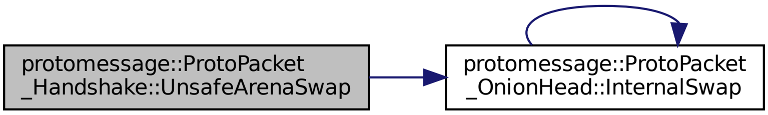


3.16.5.40 UnsafeArenaSwap() `void protomessage::ProtoPacket_Handshake::UnsafeArenaSwap ( ProtoPacket_Handshake * other ) [inline]`

Definition at line 508 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::InternalSwap().

Here is the call graph for this function:


protomessage::ProtoPacket
_Handshake::UnsafeArenaSwap
protomessage::ProtoPacket
_OnionHead::InternalSwap

3.16.6 Friends And Related Function Documentation

3.16.6.1 ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper `template<typename T > friend class ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper [friend]`

Definition at line 586 of file proto-packet.pb.h.

3.16.6.2 ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata

`friend class ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata [friend]`

Definition at line 542 of file proto-packet.pb.h.

3.16.6.3 ::TableStruct_proto_2dpacket_2eproto

`friend struct ::TableStruct_proto_2dpacket_2eproto [friend]`

Definition at line 594 of file proto-packet.pb.h.

3.16.6.4 swap

```
void swap (
        ProtoPacket_Handshake & a,
        ProtoPacket_Handshake & b )  [friend]
```

Definition at line 492 of file proto-packet.pb.h.

Referenced by InternalSwap().

3.16.7 Member Data Documentation

3.16.7.1 _cached_size_

`mutable ::PROTOBUF_NAMESPACE_ID::internal::CachedSize protomessage::ProtoPacket_Handshake::_cached_size_ [private]`

Definition at line 592 of file proto-packet.pb.h.

Referenced by ByteSizeLong(), and SetCachedSize().

3.16.7.2 _class_data_

`const ::PROTOBUF_NAMESPACE_ID::Message::ClassData protomessage::ProtoPacket_Handshake::_class_data_ [static]`

Initial value:

```
= {
    ::PROTOBUF_NAMESPACE_ID::Message::CopyWithSizeCheck,
    ProtoPacket_Handshake::MergeImpl
}
```

Definition at line 554 of file proto-packet.pb.h.

Referenced by GetClassData().

3.16.7.3 _has_bits_ `::PROTOBUF_NAMESPACE_ID::internal::HasBits<1> protomessage::ProtoPacket_Handshake::_has_bits_ [private]`

Definition at line 591 of file proto-packet.pb.h.

Referenced by _InternalParse(), ByteSizeLong(), Clear(), and InternalSwap().

3.16.7.4 kIndexInFileMessages `constexpr int protomessage::ProtoPacket_Handshake::kIndexInFileMessages [static], [constexpr]`

Initial value:

```
=
    2
```

Definition at line 489 of file proto-packet.pb.h.

3.16.7.5 publickey_ `::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr protomessage::ProtoPacket_Handshake::publickey_ [private]`

Definition at line 593 of file proto-packet.pb.h.

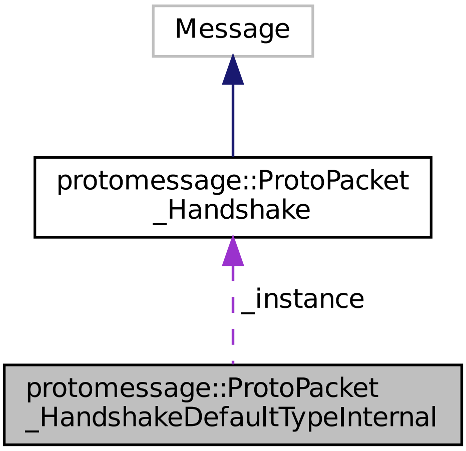
Referenced by Clear(), InternalSwap(), ProtoPacket_Handshake(), SharedCtor(), and SharedDtor().

The documentation for this class was generated from the following files:

- src/onion_routing_wsn/protobuf/proto-packet.pb.h
- src/onion_routing_wsn/protobuf/proto-packet.pb.cc

3.17 protomessage::ProtoPacket_HandshakeDefaultTypeInternal Struct Reference

Collaboration diagram for protomessage::ProtoPacket_HandshakeDefaultTypeInternal:


Message
protomessage::ProtoPacket_Handshake
_instance
protomessage::ProtoPacket_HandshakeDefaultTypeInternal

Public Member Functions

- constexpr ProtoPacket_HandshakeDefaultTypeInternal ()
- ∼ProtoPacket_HandshakeDefaultTypeInternal ()

Public Attributes

- union {
  ProtoPacket_Handshake _instance
  };

3.17.1 Detailed Description

Definition at line 50 of file proto-packet.pb.cc.

3.17.2 Constructor & Destructor Documentation

3.17.2.1 ProtoPacket_HandshakeDefaultTypeInternal()

`constexpr protomessage::ProtoPacket_Handshake↵ DefaultTypeInternal::ProtoPacket_HandshakeDefaultTypeInternal ( ) [inline], [constexpr]`

Definition at line 51 of file proto-packet.pb.cc.

3.17.2.2 ∼ProtoPacket_HandshakeDefaultTypeInternal()

`protomessage::ProtoPacket_Handshake↵ DefaultTypeInternal::∼ProtoPacket_HandshakeDefaultTypeInternal ( ) [inline]`

Definition at line 53 of file proto-packet.pb.cc.

3.17.3 Member Data Documentation

3.17.3.1 "@5

`union { ... }`

3.17.3.2 _instance

`ProtoPacket_Handshake protomessage::ProtoPacket_HandshakeDefaultType↵ Internal::_instance`

Definition at line 55 of file proto-packet.pb.cc.

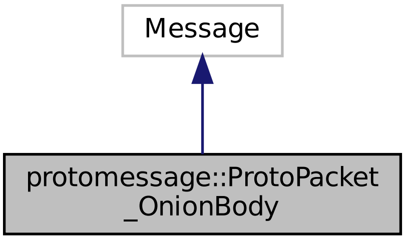
The documentation for this struct was generated from the following file:

- src/onion_routing_wsn/protobuf/proto-packet.pb.cc

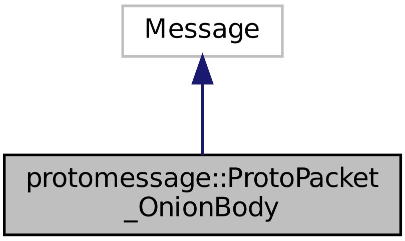
3.18 protomessage::ProtoPacket_OnionBody Class Reference

```
#include "proto-packet.pb.h"
```

Inheritance diagram for protomessage::ProtoPacket_OnionBody:



Collaboration diagram for protomessage::ProtoPacket_OnionBody:



Classes

- class _Internal

Public Types

- enum : int { kPaddingFieldNumber = 2, kAggregatedValueFieldNumber = 1 }

Public Member Functions

- ProtoPacket_OnionBody ()
- constexpr ProtoPacket_OnionBody (::PROTOBUF_NAMESPACE_ID::internal::ConstantInitialized)
- ProtoPacket_OnionBody (const ProtoPacket_OnionBody &from)
- ProtoPacket_OnionBody (ProtoPacket_OnionBody &&from) noexcept
- ∼ProtoPacket_OnionBody () override
- const char ∗ _InternalParse (const char ∗ptr, ::PROTOBUF_NAMESPACE_ID::internal::ParseContext ∗ctx) final
- uint8_t ∗ _InternalSerialize (uint8_t ∗target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream ∗stream) const final
- int32_t aggregatedvalue () const
- size_t ByteSizeLong () const final
- PROTOBUF_ATTRIBUTE_REINITIALIZES void Clear () final
- void clear_aggregatedvalue ()
- void clear_padding ()
- void CopyFrom (const ProtoPacket_OnionBody &from)
- int GetCachedSize () const final
- const ::PROTOBUF_NAMESPACE_ID::Message::ClassData ∗ GetClassData () const final
- ::PROTOBUF_NAMESPACE_ID::Metadata GetMetadata () const final
- bool has_aggregatedvalue () const
- bool has_padding () const
- bool IsInitialized () const final
- void MergeFrom (const ProtoPacket_OnionBody &from)
- std::string ∗ mutable_padding ()
- ProtoPacket_OnionBody ∗ New (::PROTOBUF_NAMESPACE_ID::Arena ∗arena=nullptr) const final
- ProtoPacket_OnionBody & operator= (const ProtoPacket_OnionBody &from)
- ProtoPacket_OnionBody & operator= (ProtoPacket_OnionBody &&from) noexcept
- const std::string & padding () const
- PROTOBUF_NODISCARD std::string ∗ release_padding ()
- void set_aggregatedvalue (int32_t value)
- void set_allocated_padding (std::string ∗padding)
- template<typename ArgT0 = const std::string&, typename... ArgT>
  void set_padding (ArgT0 &&arg0, ArgT... args)
- template<typename ArgT0 , typename... ArgT>
  PROTOBUF_ALWAYS_INLINE void set_padding (ArgT0 &&arg0, ArgT... args)
- void Swap (ProtoPacket_OnionBody ∗other)
- void UnsafeArenaSwap (ProtoPacket_OnionBody ∗other)

Static Public Member Functions

- static const ProtoPacket_OnionBody & default_instance ()
- static const ::PROTOBUF_NAMESPACE_ID::Descriptor ∗ descriptor ()
- static const ::PROTOBUF_NAMESPACE_ID::Descriptor ∗ GetDescriptor ()
- static const ::PROTOBUF_NAMESPACE_ID::Reflection ∗ GetReflection ()
- static const ProtoPacket_OnionBody ∗ internal_default_instance ()

Static Public Attributes

- static const ClassData _class_data_
- static constexpr int kIndexInFileMessages

Protected Member Functions

- ProtoPacket_OnionBody (::PROTOBUF_NAMESPACE_ID::Arena ∗arena, bool is_message_owned=false)

Private Types

- typedef void DestructorSkippable_
- typedef void InternalArenaConstructable_

Private Member Functions

- int32_t _internal_aggregatedvalue () const
- bool _internal_has_aggregatedvalue () const
- bool _internal_has_padding () const
- std::string ∗ _internal_mutable_padding ()
- const std::string & _internal_padding () const
- void _internal_set_aggregatedvalue (int32_t value)
- PROTOBUF_ALWAYS_INLINE void _internal_set_padding (const std::string &value)
- void InternalSwap (ProtoPacket_OnionBody ∗other)
- void RegisterArenaDtor (::PROTOBUF_NAMESPACE_ID::Arena ∗arena)
- void SetCachedSize (int size) const final
- void SharedCtor ()
- void SharedDtor ()

Static Private Member Functions

- static void ArenaDtor (void ∗object)
- ::PROTOBUF_NAMESPACE_ID::StringPiece FullMessageName ()
- static void MergeImpl (::PROTOBUF_NAMESPACE_ID::Message ∗to, const ::PROTOBUF_NAMESPACE←_ID::Message &from)

Private Attributes

- mutable ::PROTOBUF_NAMESPACE_ID::internal::CachedSize _cached_size_
- ::PROTOBUF_NAMESPACE_ID::internal::HasBits< 1 > _has_bits_
- int32_t aggregatedvalue_
- ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr padding_

Friends

- template<typename T >
  class ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper
- class ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata
- struct ::TableStruct_proto_2dpacket_2eproto
- void swap (ProtoPacket_OnionBody &a, ProtoPacket_OnionBody &b)

3.18.1 Detailed Description

Definition at line 271 of file proto-packet.pb.h.

3.18.2 Member Typedef Documentation

3.18.2.1 DestructorSkippable_ `typedef void protomessage::ProtoPacket_OnionBody::DestructorSkippable_ [private]`

Definition at line 433 of file proto-packet.pb.h.

3.18.2.2 InternalArenaConstructable_ `typedef void protomessage::ProtoPacket_OnionBody::InternalArenaConstructab`
`[private]`

Definition at line 432 of file proto-packet.pb.h.

3.18.3 Member Enumeration Documentation

3.18.3.1 anonymous enum `anonymous enum : int`

Enumerator

kPaddingFieldNumber	
kAggregatedValueFieldNumber	

Definition at line 392 of file proto-packet.pb.h.

3.18.4 Constructor & Destructor Documentation

3.18.4.1 ProtoPacket_OnionBody() [1/5] `protomessage::ProtoPacket_OnionBody::ProtoPacket_↩ OnionBody ( ) [inline]`

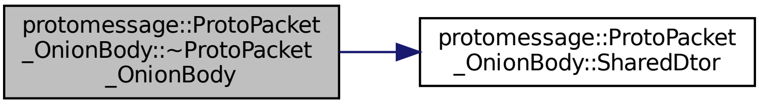
Definition at line 274 of file proto-packet.pb.h.

3.18.4.2 ∼ProtoPacket_OnionBody() `protomessage::ProtoPacket_OnionBody::∼ProtoPacket_OnionBody ( ) [override]`

Definition at line 511 of file proto-packet.pb.cc.

References SharedDtor().

Here is the call graph for this function:


protomessage::ProtoPacket_OnionBody::~ProtoPacket_OnionBody
protomessage::ProtoPacket_OnionBody::SharedDtor

3.18.4.3 ProtoPacket_OnionBody() [2/5] `constexpr protomessage::ProtoPacket_OnionBody::Proto↩ Packet_OnionBody ( ::PROTOBUF_NAMESPACE_ID::internal::ConstantInitialized ) [explicit], [constexpr]`

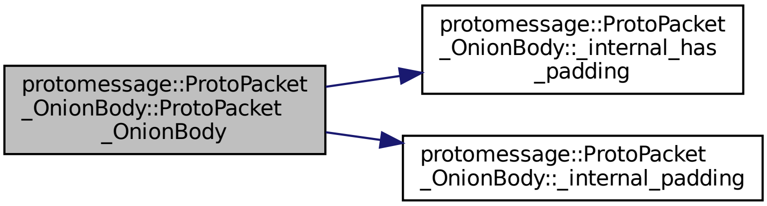
Definition at line 34 of file proto-packet.pb.cc.

3.18.4.4 ProtoPacket_OnionBody() [3/5] `protomessage::ProtoPacket_OnionBody::ProtoPacket_↩ OnionBody ( const ProtoPacket_OnionBody & from )`

Definition at line 487 of file proto-packet.pb.cc.

References _internal_has_padding(), _internal_padding(), aggregatedvalue_, and padding_.

Here is the call graph for this function:


protomessage::ProtoPacket_OnionBody::ProtoPacket_OnionBody
protomessage::ProtoPacket_OnionBody::_internal_has_padding
protomessage::ProtoPacket_OnionBody::_internal_padding

3.18.4.5 ProtoPacket_OnionBody() [4/5] protomessage::ProtoPacket_OnionBody::ProtoPacket_↩ OnionBody (

```
            ProtoPacket_OnionBody && from )  [inline], [noexcept]
```

Definition at line 279 of file proto-packet.pb.h.

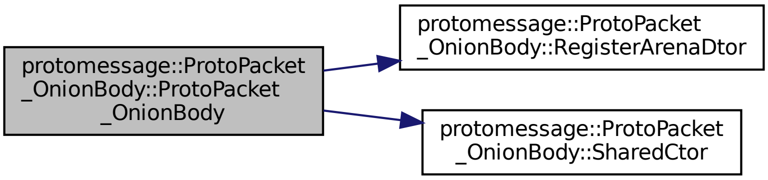
3.18.4.6 ProtoPacket_OnionBody() [5/5] protomessage::ProtoPacket_OnionBody::ProtoPacket_↩ OnionBody (

```
            ::PROTOBUF_NAMESPACE_ID::Arena * arena,
            bool is_message_owned = false )  [explicit], [protected]
```

Definition at line 478 of file proto-packet.pb.cc.

References RegisterArenaDtor(), and SharedCtor().

Here is the call graph for this function:



protomessage::ProtoPacket _OnionBody::ProtoPacket _OnionBody
protomessage::ProtoPacket _OnionBody::RegisterArenaDtor
protomessage::ProtoPacket _OnionBody::SharedCtor

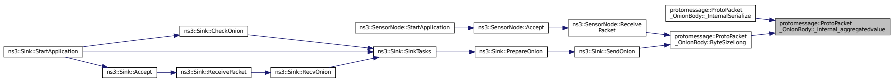
3.18.5 Member Function Documentation

3.18.5.1 _internal_aggregatedvalue() int32_t protomessage::ProtoPacket_OnionBody::_internal_↩ aggregatedvalue ( ) const [inline], [private]

Definition at line 989 of file proto-packet.pb.h.

Referenced by _InternalSerialize(), and ByteSizeLong().

Here is the caller graph for this function:



ns3::Sink::StartApplication
ns3::Sink::CheckOnion
ns3::Sink::Accept
ns3::Sink::ReceivePacket
ns3::Sink::RecvOnion
ns3::SensorNode::StartApplication
ns3::Sink::SinkTasks
ns3::SensorNode::Accept
ns3::Sink::PrepareOnion
ns3::SensorNode::Receive Packet
ns3::Sink::SendOnion
protomessage::ProtoPacket _OnionBody::_InternalSerialize
protomessage::ProtoPacket _OnionBody::ByteSizeLong
protomessage::ProtoPacket _OnionBody::_internal_aggregatedvalue

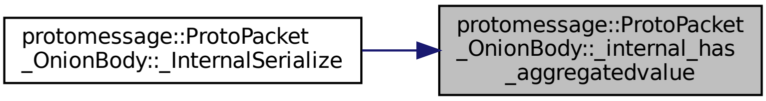
3.18.5.2 _internal_has_aggregatedvalue() `bool protomessage::ProtoPacket_OnionBody::_internal_has_aggregatedvalue ( ) const [inline], [private]`

Definition at line 978 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

Referenced by _InternalSerialize().

Here is the caller graph for this function:


protomessage::ProtoPacket _OnionBody::_InternalSerialize
protomessage::ProtoPacket _OnionBody::_internal_has _aggregatedvalue

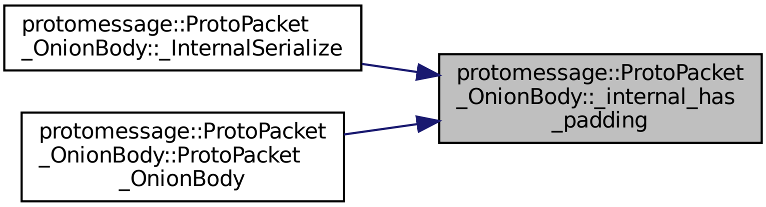
3.18.5.3 _internal_has_padding() `bool protomessage::ProtoPacket_OnionBody::_internal_has_padding ( ) const [inline], [private]`

Definition at line 1006 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

Referenced by _InternalSerialize(), and ProtoPacket_OnionBody().

Here is the caller graph for this function:


protomessage::ProtoPacket _OnionBody::_InternalSerialize
protomessage::ProtoPacket _OnionBody::ProtoPacket _OnionBody
protomessage::ProtoPacket _OnionBody::_internal_has _padding

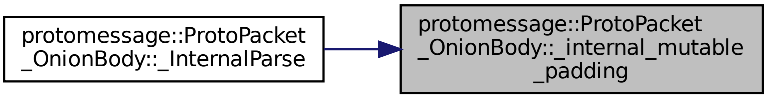
3.18.5.4 _internal_mutable_padding() `std::string * protomessage::ProtoPacket_OnionBody::_internal_mutable_padding ( ) [inline], [private]`

Definition at line 1040 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_, and protomessage::ProtoPacket_OnionHead::padding_.

Referenced by _InternalParse().

Here is the caller graph for this function:


protomessage::ProtoPacket _OnionBody::_InternalParse
protomessage::ProtoPacket _OnionBody::_internal_mutable _padding

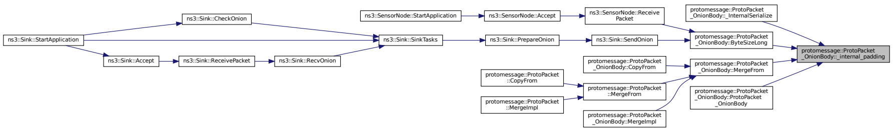
3.18.5.5 _internal_padding() `const std::string & protomessage::ProtoPacket_OnionBody::_internal_padding ( ) const [inline], [private]`

Definition at line 1033 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::padding_.

Referenced by _InternalSerialize(), ByteSizeLong(), MergeFrom(), and ProtoPacket_OnionBody().

Here is the caller graph for this function:


ns3::Sink::StartApplication
ns3::Sink::CheckOnion
ns3::Sink::Accept
ns3::Sink::ReceivePacket
ns3::Sink::RecvOnion
ns3::SensorNode::StartApplication
ns3::Sink::SinkTasks
ns3::SensorNode::Accept
ns3::Sink::PrepareOnion
ns3::SensorNode::Receive Packet
ns3::Sink::SendOnion
protomessage::ProtoPacket _OnionBody::CopyFrom
protomessage::ProtoPacket ::CopyFrom
protomessage::ProtoPacket ::MergeFrom
protomessage::ProtoPacket ::MergeImpl
protomessage::ProtoPacket _OnionBody::MergeImpl
protomessage::ProtoPacket _OnionBody::_InternalSerialize
protomessage::ProtoPacket _OnionBody::ByteSizeLong
protomessage::ProtoPacket _OnionBody::MergeFrom
protomessage::ProtoPacket _OnionBody::ProtoPacket _OnionBody
protomessage::ProtoPacket _OnionBody::_internal_padding

3.18.5.6 _internal_set_aggregatedvalue() `void protomessage::ProtoPacket_OnionBody::_internal_set_aggregatedvalue ( int32_t value ) [inline], [private]`

Definition at line 996 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

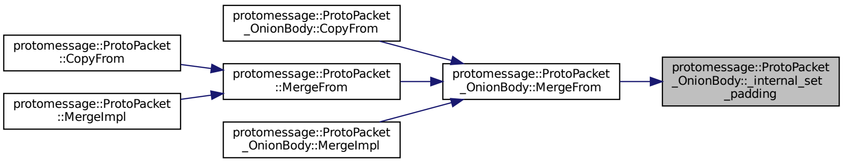
3.18.5.7 _internal_set_padding() `void protomessage::ProtoPacket_OnionBody::_internal_set_padding ( const std::string & value ) [inline], [private]`

Definition at line 1036 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_, and protomessage::ProtoPacket_OnionHead↩ ::padding_.

Referenced by MergeFrom().

Here is the caller graph for this function:

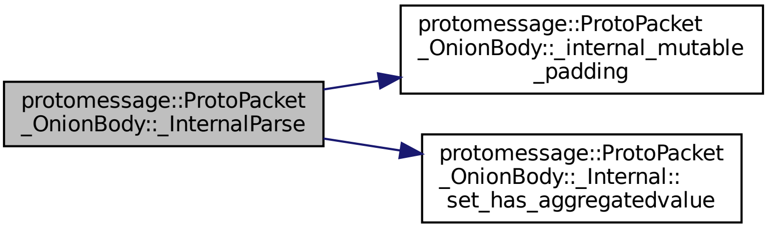

protomessage::ProtoPacket ::CopyFrom
protomessage::ProtoPacket ::MergeImpl
protomessage::ProtoPacket _OnionBody::CopyFrom
protomessage::ProtoPacket ::MergeFrom
protomessage::ProtoPacket _OnionBody::MergeImpl
protomessage::ProtoPacket _OnionBody::MergeFrom
protomessage::ProtoPacket _OnionBody::_internal_set _padding

3.18.5.8 _InternalParse() `const char * protomessage::ProtoPacket_OnionBody::_InternalParse ( const char * ptr, ::PROTOBUF_NAMESPACE_ID::internal::ParseContext * ctx ) [final]`

Definition at line 548 of file proto-packet.pb.cc.

References _has_bits_, _internal_mutable_padding(), aggregatedvalue_, CHK_, and protomessage::Proto↩ Packet_OnionBody::_Internal::set_has_aggregatedvalue().

Here is the call graph for this function:

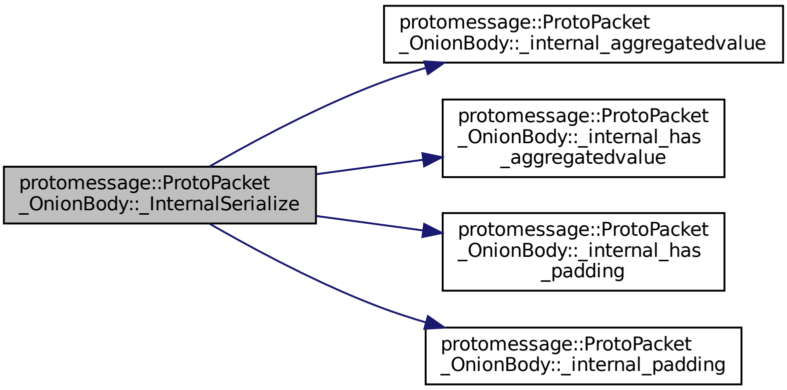

protomessage::ProtoPacket _OnionBody::_InternalParse
protomessage::ProtoPacket _OnionBody::_internal_mutable _padding
protomessage::ProtoPacket _OnionBody::_Internal:: set_has_aggregatedvalue

3.18.5.9 _InternalSerialize() `uint8_t * protomessage::ProtoPacket_OnionBody::_InternalSerialize ( uint8_t * target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream * stream ) const [final]`

Definition at line 597 of file proto-packet.pb.cc.

References _internal_aggregatedvalue(), _internal_has_aggregatedvalue(), _internal_has_padding(), and _internal_padding().

Here is the call graph for this function:

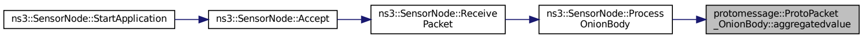

protomessage::ProtoPacket _OnionBody::_InternalSerialize
protomessage::ProtoPacket _OnionBody::_internal_aggregatedvalue
protomessage::ProtoPacket _OnionBody::_internal_has _aggregatedvalue
protomessage::ProtoPacket _OnionBody::_internal_has _padding
protomessage::ProtoPacket _OnionBody::_internal_padding

3.18.5.10 aggregatedvalue() `int32_t protomessage::ProtoPacket_OnionBody::aggregatedvalue ( ) const [inline]`

Definition at line 992 of file proto-packet.pb.h.

Referenced by ns3::SensorNode::ProcessOnionBody().

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::SensorNode::Process OnionBody
protomessage::ProtoPacket _OnionBody::aggregatedvalue

3.18.5.11 ArenaDtor() `void protomessage::ProtoPacket_OnionBody::ArenaDtor ( void * object ) [static], [private]`

Definition at line 523 of file proto-packet.pb.cc.

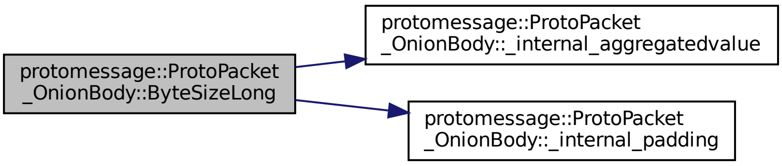
3.18.5.12 ByteSizeLong() `size_t protomessage::ProtoPacket_OnionBody::ByteSizeLong ( ) const [final]`

Definition at line 623 of file proto-packet.pb.cc.

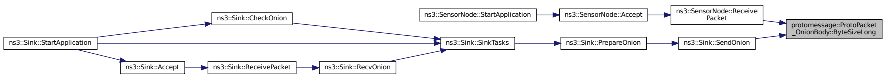
References _cached_size_, _has_bits_, _internal_aggregatedvalue(), and _internal_padding().

Referenced by ns3::SensorNode::ReceivePacket(), and ns3::Sink::SendOnion().

Here is the call graph for this function:


protomessage::ProtoPacket
_OnionBody::ByteSizeLong
protomessage::ProtoPacket
_OnionBody::_internal_aggregatedvalue
protomessage::ProtoPacket
_OnionBody::_internal_padding

Here is the caller graph for this function:



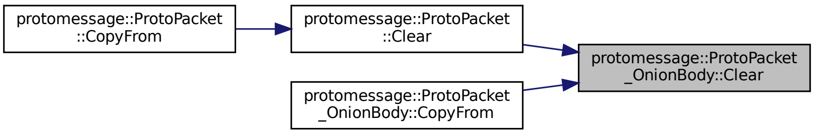
3.18.5.13 Clear() `void protomessage::ProtoPacket_OnionBody::Clear ( ) [final]`

Definition at line 533 of file proto-packet.pb.cc.

References _has_bits_, aggregatedvalue_, and padding_.

Referenced by protomessage::ProtoPacket::Clear(), and CopyFrom().

Here is the caller graph for this function:


protomessage::ProtoPacket
::CopyFrom
protomessage::ProtoPacket
::Clear
protomessage::ProtoPacket
_OnionBody::CopyFrom
protomessage::ProtoPacket
_OnionBody::Clear

3.18.5.14 clear_aggregatedvalue() `void protomessage::ProtoPacket_OnionBody::clear_aggregatedvalue ( ) [inline]`

Definition at line 985 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_.

3.18.5.15 clear_padding() `void protomessage::ProtoPacket_OnionBody::clear_padding ( ) [inline]`

Definition at line 1013 of file proto-packet.pb.h.

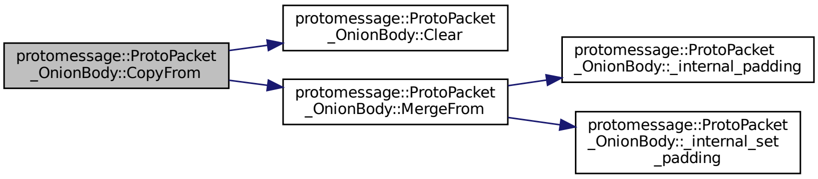
References protomessage::ProtoPacket_OnionHead::_has_bits_, and protomessage::ProtoPacket_OnionHead↩ ::padding_.

3.18.5.16 CopyFrom() `void protomessage::ProtoPacket_OnionBody::CopyFrom ( const ProtoPacket_OnionBody & from )`

Definition at line 681 of file proto-packet.pb.cc.

References Clear(), and MergeFrom().

Here is the call graph for this function:

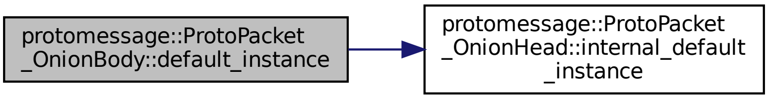

protomessage::ProtoPacket _OnionBody::CopyFrom
protomessage::ProtoPacket _OnionBody::Clear
protomessage::ProtoPacket _OnionBody::MergeFrom
protomessage::ProtoPacket _OnionBody::_internal_padding
protomessage::ProtoPacket _OnionBody::_internal_set _padding

3.18.5.17 default_instance() `static const ProtoPacket_OnionBody& protomessage::ProtoPacket_↩ OnionBody::default_instance ( ) [inline], [static]`

Definition at line 311 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::internal_default_instance().

Here is the call graph for this function:

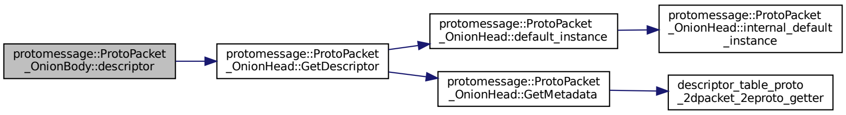

protomessage::ProtoPacket _OnionBody::default_instance
protomessage::ProtoPacket _OnionHead::internal_default _instance

3.18.5.18 descriptor() `static const ::PROTOBUF_NAMESPACE_ID::Descriptor* protomessage::ProtoPacket_OnionBody::descriptor ( ) [inline], [static]`

Definition at line 302 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::GetDescriptor().

Here is the call graph for this function:


protomessage::ProtoPacket_OnionBody::descriptor
protomessage::ProtoPacket_OnionHead::GetDescriptor
protomessage::ProtoPacket_OnionHead::default_instance
protomessage::ProtoPacket_OnionHead::internal_default_instance
protomessage::ProtoPacket_OnionHead::GetMetadata
descriptor_table_proto_2dpacket_2eproto_getter

3.18.5.19 FullMessageName() `::PROTOBUF_NAMESPACE_ID::StringPiece protomessage::ProtoPacket_OnionBody::FullMessageName ( ) [inline], [static], [private]`

Definition at line 372 of file proto-packet.pb.h.

3.18.5.20 GetCachedSize() `int protomessage::ProtoPacket_OnionBody::GetCachedSize ( ) const [inline], [final]`

Definition at line 362 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_cached_size_.

3.18.5.21 GetClassData() `const ::PROTOBUF_NAMESPACE_ID::Message::ClassData * protomessage::ProtoPacket_OnionBody::GetClassData ( ) const [final]`

Definition at line 653 of file proto-packet.pb.cc.

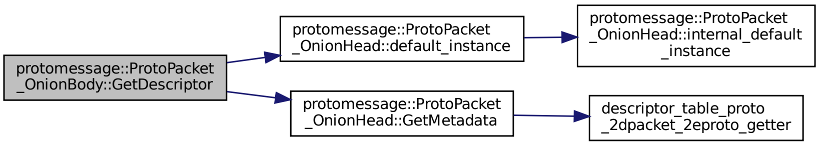
References _class_data_.

3.18.5.22 GetDescriptor() `static const ::PROTOBUF_NAMESPACE_ID::Descriptor* protomessage::ProtoPacket_OnionBody::GetDescriptor ( ) [inline], [static]`

Definition at line 305 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::default_instance(), and protomessage::ProtoPacket_OnionHead::GetMetadata().

Here is the call graph for this function:

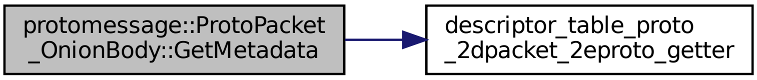

protomessage::ProtoPacket _OnionBody::GetDescriptor
protomessage::ProtoPacket _OnionHead::default_instance
protomessage::ProtoPacket _OnionHead::internal_default _instance
protomessage::ProtoPacket _OnionHead::GetMetadata
descriptor_table_proto _2dpacket_2eproto_getter

3.18.5.23 GetMetadata() `PROTOBUF_NAMESPACE_ID::Metadata protomessage::ProtoPacket_OnionBody::GetMetadata ( ) const [final]`

Definition at line 706 of file proto-packet.pb.cc.

References descriptor_table_proto_2dpacket_2eproto_getter(), descriptor_table_proto_2dpacket_2eproto_once, and file_level_metadata_proto_2dpacket_2eproto.

Here is the call graph for this function:

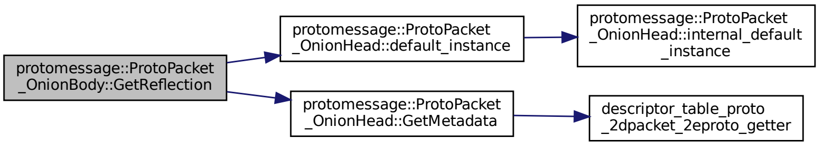

protomessage::ProtoPacket _OnionBody::GetMetadata
descriptor_table_proto _2dpacket_2eproto_getter

3.18.5.24 GetReflection() `static const ::PROTOBUF_NAMESPACE_ID::Reflection* protomessage::ProtoPacket_OnionBody::GetReflection ( ) [inline], [static]`

Definition at line 308 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::default_instance(), and protomessage::ProtoPacket_OnionHead::GetMetadata().

Here is the call graph for this function:

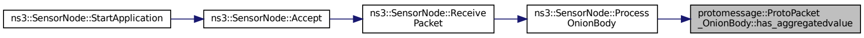

protomessage::ProtoPacket
_OnionBody::GetReflection
protomessage::ProtoPacket
_OnionHead::default_instance
protomessage::ProtoPacket
_OnionHead::internal_default
_instance
protomessage::ProtoPacket
_OnionHead::GetMetadata
descriptor_table_proto
_2dpacket_2eproto_getter

3.18.5.25 has_aggregatedvalue() `bool protomessage::ProtoPacket_OnionBody::has_aggregatedvalue ( ) const [inline]`

Definition at line 982 of file proto-packet.pb.h.

Referenced by ns3::SensorNode::ProcessOnionBody().

Here is the caller graph for this function:

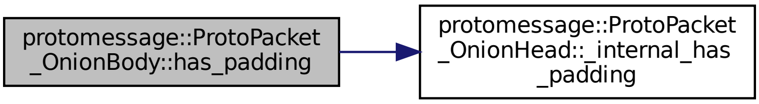

ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive
Packet
ns3::SensorNode::Process
OnionBody
protomessage::ProtoPacket
_OnionBody::has_aggregatedvalue

3.18.5.26 has_padding() `bool protomessage::ProtoPacket_OnionBody::has_padding ( ) const [inline]`

Definition at line 1010 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_internal_has_padding().

Here is the call graph for this function:


protomessage::ProtoPacket
_OnionBody::has_padding
protomessage::ProtoPacket
_OnionHead::_internal_has
_padding

3.18.5.27 internal_default_instance() static const ProtoPacket_OnionBody* protomessage::Proto←↩ Packet_OnionBody::internal_default_instance ( ) [inline], [static]

Definition at line 314 of file proto-packet.pb.h.

References protomessage::_ProtoPacket_OnionBody_default_instance_.

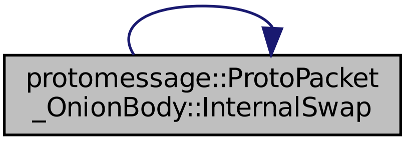
3.18.5.28 InternalSwap() void protomessage::ProtoPacket_OnionBody::InternalSwap (ProtoPacket_OnionBody * *other*) [private]

Definition at line 692 of file proto-packet.pb.cc.

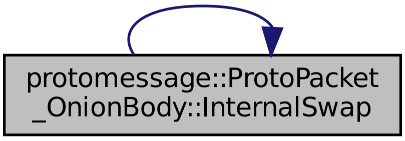
References _has_bits_, aggregatedvalue_, InternalSwap(), padding_, and swap.

Referenced by InternalSwap().

Here is the call graph for this function:



Here is the caller graph for this function:



3.18.5.29 IsInitialized() bool protomessage::ProtoPacket_OnionBody::IsInitialized ( ) const [final]

Definition at line 688 of file proto-packet.pb.cc.

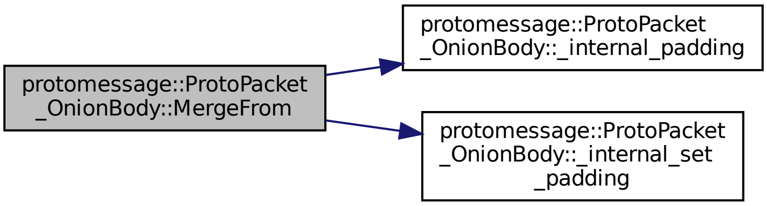
3.18.5.30 MergeFrom() `void protomessage::ProtoPacket_OnionBody::MergeFrom ( const ProtoPacket_OnionBody & from )`

Definition at line 662 of file proto-packet.pb.cc.

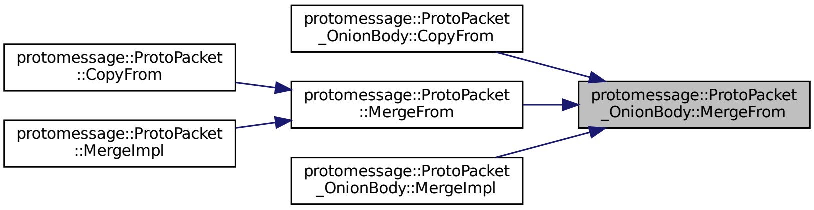
References _has_bits_, _internal_padding(), _internal_set_padding(), and aggregatedvalue_.

Referenced by CopyFrom(), protomessage::ProtoPacket::MergeFrom(), and MergeImpl().

Here is the call graph for this function:


protomessage::ProtoPacket
_OnionBody::MergeFrom
protomessage::ProtoPacket
_OnionBody::_internal_padding
protomessage::ProtoPacket
_OnionBody::_internal_set
_padding

Here is the caller graph for this function:

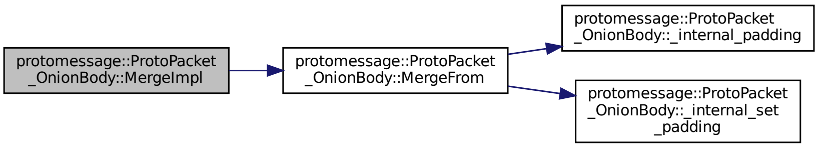

protomessage::ProtoPacket
::CopyFrom
protomessage::ProtoPacket
::MergeImpl
protomessage::ProtoPacket
_OnionBody::CopyFrom
protomessage::ProtoPacket
::MergeFrom
protomessage::ProtoPacket
_OnionBody::MergeImpl
protomessage::ProtoPacket
_OnionBody::MergeFrom

3.18.5.31 MergeImpl() `void protomessage::ProtoPacket_OnionBody::MergeImpl ( ::PROTOBUF_NAMESPACE_ID::Message * to, const ::PROTOBUF_NAMESPACE_ID::Message & from ) [static], [private]`

Definition at line 655 of file proto-packet.pb.cc.

References MergeFrom().

Here is the call graph for this function:

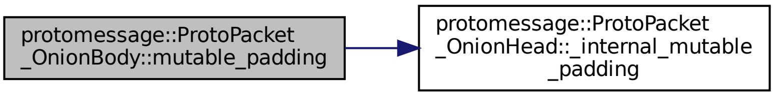

protomessage::ProtoPacket
_OnionBody::MergeImpl
protomessage::ProtoPacket
_OnionBody::MergeFrom
protomessage::ProtoPacket
_OnionBody::_internal_padding
protomessage::ProtoPacket
_OnionBody::_internal_set
_padding

3.18.5.32 mutable_padding() `std::string * protomessage::ProtoPacket_OnionBody::mutable_padding ( ) [inline]`

Definition at line 1028 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_internal_mutable_padding().

Here is the call graph for this function:


protomessage::ProtoPacket
_OnionBody::mutable_padding
protomessage::ProtoPacket
_OnionHead::_internal_mutable
_padding

3.18.5.33 New() `ProtoPacket_OnionBody* protomessage::ProtoPacket_OnionBody::New ( ::PROTOBUF_NAMESPACE_ID::Arena * arena = nullptr ) const [inline], [final]`

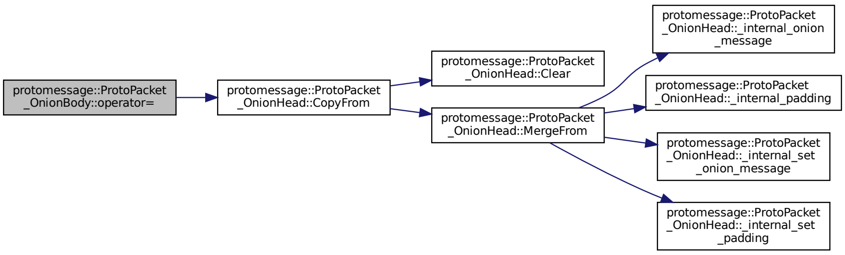
Definition at line 345 of file proto-packet.pb.h.

3.18.5.34 operator=() [1/2] `ProtoPacket_OnionBody& protomessage::ProtoPacket_OnionBody::operator= ( const ProtoPacket_OnionBody & from ) [inline]`

Definition at line 284 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::CopyFrom().

Here is the call graph for this function:


protomessage::ProtoPacket
_OnionBody::operator=
protomessage::ProtoPacket
_OnionHead::CopyFrom
protomessage::ProtoPacket
_OnionHead::Clear
protomessage::ProtoPacket
_OnionHead::MergeFrom
protomessage::ProtoPacket
_OnionHead::_internal_onion
_message
protomessage::ProtoPacket
_OnionHead::_internal_padding
protomessage::ProtoPacket
_OnionHead::_internal_set
_onion_message
protomessage::ProtoPacket
_OnionHead::_internal_set
_padding

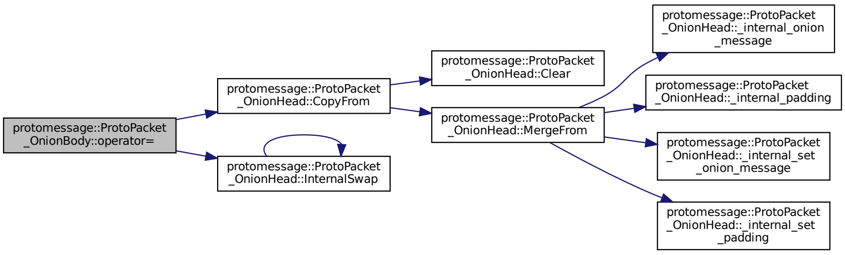
3.18.5.35 operator=() [2/2] ProtoPacket_OnionBody& protomessage::ProtoPacket_OnionBody::operator= (

ProtoPacket_OnionBody && *from*) [inline], [noexcept]

Definition at line 288 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::CopyFrom(), and protomessage::ProtoPacket_OnionHead↩ ::InternalSwap().

Here is the call graph for this function:

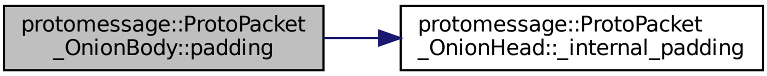

protomessage::ProtoPacket _OnionBody::operator=
protomessage::ProtoPacket _OnionHead::CopyFrom
protomessage::ProtoPacket _OnionHead::InternalSwap
protomessage::ProtoPacket _OnionHead::Clear
protomessage::ProtoPacket _OnionHead::MergeFrom
protomessage::ProtoPacket _OnionHead::_internal_onion _message
protomessage::ProtoPacket _OnionHead::_internal_padding
protomessage::ProtoPacket _OnionHead::_internal_set _onion_message
protomessage::ProtoPacket _OnionHead::_internal_set _padding

3.18.5.36 padding() const std::string & protomessage::ProtoPacket_OnionBody::padding ( ) const [inline]

Definition at line 1017 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_internal_padding().

Here is the call graph for this function:

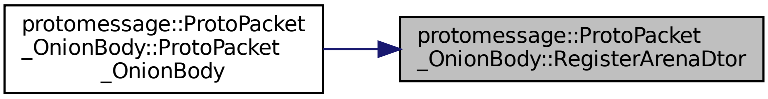

protomessage::ProtoPacket _OnionBody::padding
protomessage::ProtoPacket _OnionHead::_internal_padding

3.18.5.37 RegisterArenaDtor() `void protomessage::ProtoPacket_OnionBody::RegisterArenaDtor ( ::PROTOBUF_NAMESPACE_ID::Arena * arena ) [inline], [private]`

Definition at line 527 of file proto-packet.pb.cc.

Referenced by ProtoPacket_OnionBody().

Here is the caller graph for this function:

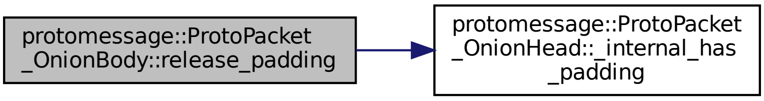

protomessage::ProtoPacket
_OnionBody::ProtoPacket
_OnionBody
protomessage::ProtoPacket
_OnionBody::RegisterArenaDtor

3.18.5.38 release_padding() `std::string * protomessage::ProtoPacket_OnionBody::release_padding ( ) [inline]`

Definition at line 1044 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_, protomessage::ProtoPacket_OnionHead::_internal_has_padding(), and protomessage::ProtoPacket_OnionHead::padding_.

Here is the call graph for this function:

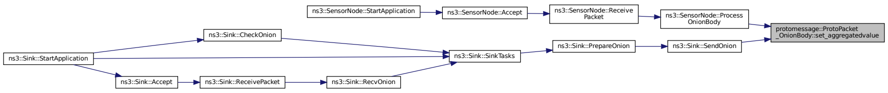

protomessage::ProtoPacket
_OnionBody::release_padding
protomessage::ProtoPacket
_OnionHead::_internal_has
_padding

3.18.5.39 set_aggregatedvalue() `void protomessage::ProtoPacket_OnionBody::set_aggregatedvalue ( int32_t value ) [inline]`

Definition at line 1000 of file proto-packet.pb.h.

Referenced by ns3::SensorNode::ProcessOnionBody(), and ns3::Sink::SendOnion().

Here is the caller graph for this function:



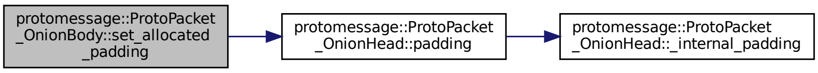
3.18.5.40 set_allocated_padding() `void protomessage::ProtoPacket_OnionBody::set_allocated_padding (`

```
            std::string * padding )  [inline]
```

Definition at line 1058 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_, protomessage::ProtoPacket_OnionHead::padding(), and protomessage::ProtoPacket_OnionHead::padding_.

Here is the call graph for this function:

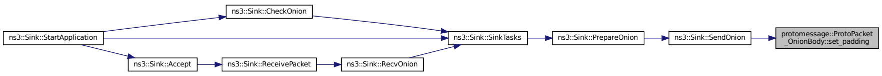

protomessage::ProtoPacket_OnionBody::set_allocated_padding
protomessage::ProtoPacket_OnionHead::padding
protomessage::ProtoPacket_OnionHead::_internal_padding

3.18.5.41 set_padding() [1/2] `template<typename ArgT0 = const std::string&, typename... ArgT>`

```
void protomessage::ProtoPacket_OnionBody::set_padding (
            ArgT0 && arg0,
            ArgT... args )
```

Referenced by ns3::Sink::SendOnion().

Here is the caller graph for this function:



3.18.5.42 set_padding() [2/2] `template<typename ArgT0 , typename... ArgT>`

```
PROTOBUF_ALWAYS_INLINE void protomessage::ProtoPacket_OnionBody::set_padding (
            ArgT0 && arg0,
            ArgT... args )  [inline]
```

Definition at line 1023 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::_has_bits_, and protomessage::ProtoPacket_OnionHead::padding_.

3.18.5.43 SetCachedSize() `void protomessage::ProtoPacket_OnionBody::SetCachedSize ( int size ) const [final], [private]`

Definition at line 529 of file proto-packet.pb.cc.

References _cached_size_.

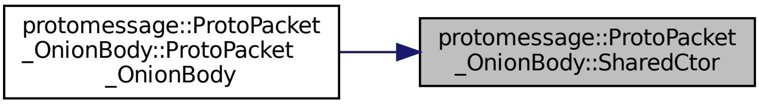
3.18.5.44 SharedCtor() `void protomessage::ProtoPacket_OnionBody::SharedCtor ( ) [inline], [private]`

Definition at line 503 of file proto-packet.pb.cc.

References aggregatedvalue_, and padding_.

Referenced by ProtoPacket_OnionBody().

Here is the caller graph for this function:



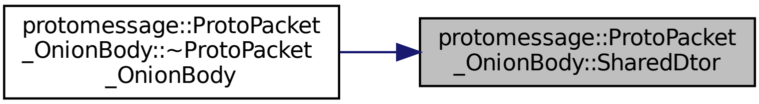
3.18.5.45 SharedDtor() `void protomessage::ProtoPacket_OnionBody::SharedDtor ( ) [inline], [private]`

Definition at line 518 of file proto-packet.pb.cc.

References padding_.

Referenced by ∼ProtoPacket_OnionBody().

Here is the caller graph for this function:

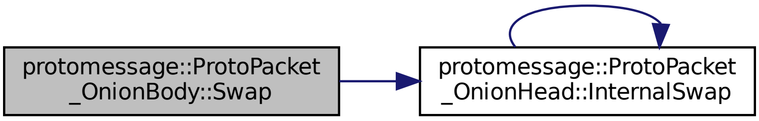


3.18.5.46 Swap() `void protomessage::ProtoPacket_OnionBody::Swap ( ProtoPacket_OnionBody * other ) [inline]`

Definition at line 324 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::InternalSwap().

Here is the call graph for this function:

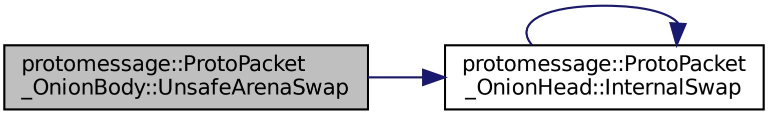


3.18.5.47 UnsafeArenaSwap() `void protomessage::ProtoPacket_OnionBody::UnsafeArenaSwap ( ProtoPacket_OnionBody * other ) [inline]`

Definition at line 337 of file proto-packet.pb.h.

References protomessage::ProtoPacket_OnionHead::InternalSwap().

Here is the call graph for this function:


protomessage::ProtoPacket
_OnionBody::UnsafeArenaSwap
protomessage::ProtoPacket
_OnionHead::InternalSwap

3.18.6 Friends And Related Function Documentation

3.18.6.1 ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper `template<typename T > friend class ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper [friend]`

Definition at line 429 of file proto-packet.pb.h.

3.18.6.2 ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata `friend class ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata [friend]`

Definition at line 371 of file proto-packet.pb.h.

3.18.6.3 ::TableStruct_proto_2dpacket_2eproto `friend struct ::TableStruct_proto_2dpacket_2eproto [friend]`

Definition at line 438 of file proto-packet.pb.h.

3.18.6.4 swap

```
void swap (
        ProtoPacket_OnionBody & a,
        ProtoPacket_OnionBody & b )  [friend]
```

Definition at line 321 of file proto-packet.pb.h.

Referenced by InternalSwap().

3.18.7 Member Data Documentation

3.18.7.1 _cached_size_ `mutable ::PROTOBUF_NAMESPACE_ID::internal::CachedSize protomessage::ProtoPacket_OnionBody::_cached_size_ [private]`

Definition at line 435 of file proto-packet.pb.h.

Referenced by ByteSizeLong(), and SetCachedSize().

3.18.7.2 _class_data_ `const ::PROTOBUF_NAMESPACE_ID::Message::ClassData protomessage::ProtoPacket_OnionBody::_class_data_ [static]`

Initial value:

```
= {
    ::PROTOBUF_NAMESPACE_ID::Message::CopyWithSizeCheck,
    ProtoPacket_OnionBody::MergeImpl
}
```

Definition at line 383 of file proto-packet.pb.h.

Referenced by GetClassData().

3.18.7.3 _has_bits_ `::PROTOBUF_NAMESPACE_ID::internal::HasBits<1> protomessage::ProtoPacket_OnionBody::_has_bits_ [private]`

Definition at line 434 of file proto-packet.pb.h.

Referenced by _InternalParse(), ByteSizeLong(), Clear(), InternalSwap(), and MergeFrom().

3.18.7.4 aggregatedvalue_ `int32_t protomessage::ProtoPacket_OnionBody::aggregatedvalue_ [private]`

Definition at line 437 of file proto-packet.pb.h.

Referenced by _InternalParse(), Clear(), InternalSwap(), MergeFrom(), ProtoPacket_OnionBody(), and SharedCtor().

3.18.7.5 kIndexInFileMessages `constexpr int protomessage::ProtoPacket_OnionBody::kIndexInFileMessages [static], [constexpr]`

Initial value:

```
=
    1
```

Definition at line 318 of file proto-packet.pb.h.

3.18.7.6 padding_ `::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr protomessage::ProtoPacket_OnionBody::padding_ [private]`

Definition at line 436 of file proto-packet.pb.h.

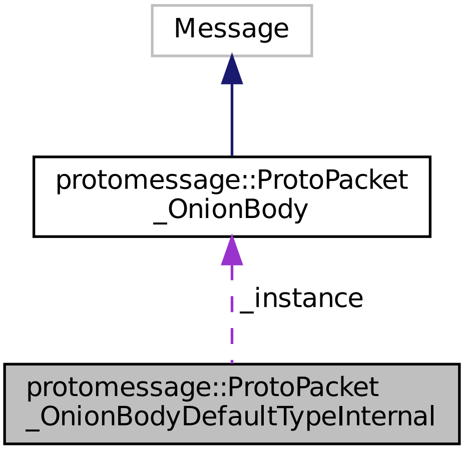
Referenced by Clear(), InternalSwap(), ProtoPacket_OnionBody(), SharedCtor(), and SharedDtor().

The documentation for this class was generated from the following files:

- src/onion_routing_wsn/protobuf/proto-packet.pb.h
- src/onion_routing_wsn/protobuf/proto-packet.pb.cc

3.19 protomessage::ProtoPacket_OnionBodyDefaultTypeInternal Struct Reference

Collaboration diagram for protomessage::ProtoPacket_OnionBodyDefaultTypeInternal:


Message
protomessage::ProtoPacket_OnionBody
_instance
protomessage::ProtoPacket_OnionBodyDefaultTypeInternal

Public Member Functions

- constexpr ProtoPacket_OnionBodyDefaultTypeInternal ()
- ∼ProtoPacket_OnionBodyDefaultTypeInternal ()

Public Attributes

- union {
  ProtoPacket_OnionBody _instance
  };

3.19.1 Detailed Description

Definition at line 38 of file proto-packet.pb.cc.

3.19.2 Constructor & Destructor Documentation

3.19.2.1 ProtoPacket_OnionBodyDefaultTypeInternal() `constexpr protomessage::ProtoPacket_OnionBodyDefaultTypeInternal::ProtoPacket_OnionBodyDefaultTypeInternal ( ) [inline], [constexpr]`

Definition at line 39 of file proto-packet.pb.cc.

3.19.2.2 ∼ProtoPacket_OnionBodyDefaultTypeInternal() protomessage::ProtoPacket_OnionBody←↩ DefaultTypeInternal::∼ProtoPacket_OnionBodyDefaultTypeInternal ( ) [inline]

Definition at line 41 of file proto-packet.pb.cc.

3.19.3 Member Data Documentation

3.19.3.1 "@3 union { ... }

3.19.3.2 _instance ProtoPacket_OnionBody protomessage::ProtoPacket_OnionBodyDefaultType←↩ Internal::_instance

Definition at line 43 of file proto-packet.pb.cc.

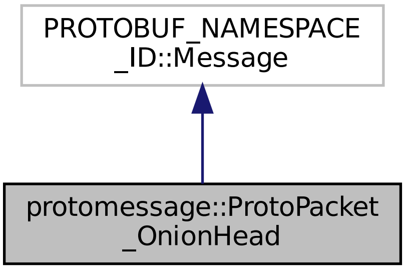
The documentation for this struct was generated from the following file:

- src/onion_routing_wsn/protobuf/proto-packet.pb.cc

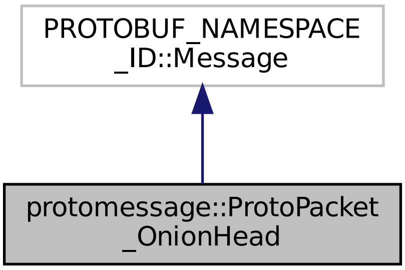
3.20 protomessage::ProtoPacket_OnionHead Class Reference

```
#include "proto-packet.pb.h"
```

Inheritance diagram for protomessage::ProtoPacket_OnionHead:



Collaboration diagram for protomessage::ProtoPacket_OnionHead:


PROTOBUF_NAMESPACE _ID::Message
protomessage::ProtoPacket _OnionHead

Classes

- class _Internal

Public Types

- enum : int { kOnionMessageFieldNumber = 2, kPaddingFieldNumber = 3, kOnionIdFieldNumber = 1 }

Public Member Functions

- ProtoPacket_OnionHead ()
- constexpr ProtoPacket_OnionHead (::PROTOBUF_NAMESPACE_ID::internal::ConstantInitialized)
- ProtoPacket_OnionHead (const ProtoPacket_OnionHead &from)
- ProtoPacket_OnionHead (ProtoPacket_OnionHead &&from) noexcept
- ∼ProtoPacket_OnionHead () override
- const char ∗ _InternalParse (const char ∗ptr, ::PROTOBUF_NAMESPACE_ID::internal::ParseContext ∗ctx) final
- uint8_t ∗ _InternalSerialize (uint8_t ∗target, ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream ∗stream) const final
- size_t ByteSizeLong () const final
- PROTOBUF_ATTRIBUTE_REINITIALIZES void Clear () final
- void clear_onion_message ()
- void clear_onionid ()
- void clear_padding ()
- void CopyFrom (const ProtoPacket_OnionHead &from)
- int GetCachedSize () const final
- const ::PROTOBUF_NAMESPACE_ID::Message::ClassData ∗ GetClassData () const final
- ::PROTOBUF_NAMESPACE_ID::Metadata GetMetadata () const final
- bool has_onion_message () const
- bool has_onionid () const
- bool has_padding () const
- bool IsInitialized () const final
- void MergeFrom (const ProtoPacket_OnionHead &from)
- std::string ∗ mutable_onion_message ()

- std::string ∗ mutable_padding ()
- ProtoPacket_OnionHead ∗ New (::PROTOBUF_NAMESPACE_ID::Arena ∗arena=nullptr) const final
- const std::string & onion_message () const
- int32_t onionid () const
- ProtoPacket_OnionHead & operator= (const ProtoPacket_OnionHead &from)
- ProtoPacket_OnionHead & operator= (ProtoPacket_OnionHead &&from) noexcept
- const std::string & padding () const
- PROTOBUF_NODISCARD std::string ∗ release_onion_message ()
- PROTOBUF_NODISCARD std::string ∗ release_padding ()
- void set_allocated_onion_message (std::string ∗onion_message)
- void set_allocated_padding (std::string ∗padding)
- template<typename ArgT0 = const std::string&, typename... ArgT>
  void set_onion_message (ArgT0 &&arg0, ArgT... args)
- template<typename ArgT0 , typename... ArgT>
  PROTOBUF_ALWAYS_INLINE void set_onion_message (ArgT0 &&arg0, ArgT... args)
- void set_onionid (int32_t value)
- template<typename ArgT0 = const std::string&, typename... ArgT>
  void set_padding (ArgT0 &&arg0, ArgT... args)
- template<typename ArgT0 , typename... ArgT>
  PROTOBUF_ALWAYS_INLINE void set_padding (ArgT0 &&arg0, ArgT... args)
- void Swap (ProtoPacket_OnionHead ∗other)
- void UnsafeArenaSwap (ProtoPacket_OnionHead ∗other)

Static Public Member Functions

- static const ProtoPacket_OnionHead & default_instance ()
- static const ::PROTOBUF_NAMESPACE_ID::Descriptor ∗ descriptor ()
- static const ::PROTOBUF_NAMESPACE_ID::Descriptor ∗ GetDescriptor ()
- static const ::PROTOBUF_NAMESPACE_ID::Reflection ∗ GetReflection ()
- static const ProtoPacket_OnionHead ∗ internal_default_instance ()

Static Public Attributes

- static const ClassData _class_data_
- static constexpr int kIndexInFileMessages

Protected Member Functions

- ProtoPacket_OnionHead (::PROTOBUF_NAMESPACE_ID::Arena ∗arena, bool is_message_owned=false)

Private Types

- typedef void DestructorSkippable_
- typedef void InternalArenaConstructable_

Private Member Functions

- bool _internal_has_onion_message () const
- bool _internal_has_onionid () const
- bool _internal_has_padding () const
- std::string ∗ _internal_mutable_onion_message ()
- std::string ∗ _internal_mutable_padding ()
- const std::string & _internal_onion_message () const
- int32_t _internal_onionid () const
- const std::string & _internal_padding () const
- PROTOBUF_ALWAYS_INLINE void _internal_set_onion_message (const std::string &value)
- void _internal_set_onionid (int32_t value)
- PROTOBUF_ALWAYS_INLINE void _internal_set_padding (const std::string &value)
- void InternalSwap (ProtoPacket_OnionHead ∗other)
- void RegisterArenaDtor (::PROTOBUF_NAMESPACE_ID::Arena ∗arena)
- void SetCachedSize (int size) const final
- void SharedCtor ()
- void SharedDtor ()

Static Private Member Functions

- static void ArenaDtor (void ∗object)
- ::PROTOBUF_NAMESPACE_ID::StringPiece FullMessageName ()
- static void MergeImpl (::PROTOBUF_NAMESPACE_ID::Message ∗to, const ::PROTOBUF_NAMESPACE←_ID::Message &from)

Private Attributes

- mutable ::PROTOBUF_NAMESPACE_ID::internal::CachedSize _cached_size_
- ::PROTOBUF_NAMESPACE_ID::internal::HasBits< 1 > _has_bits_
- ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr onion_message_
- int32_t onionid_
- ::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr padding_

Friends

- template<typename T >
  class ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper
- class ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata
- struct ::TableStruct_proto_2dpacket_2eproto
- void swap (ProtoPacket_OnionHead &a, ProtoPacket_OnionHead &b)

3.20.1 Detailed Description

Definition at line 80 of file proto-packet.pb.h.

3.20.2 Member Typedef Documentation

3.20.2.1 DestructorSkippable_ typedef void protomessage::ProtoPacket_OnionHead::DestructorSkippable_ [private]

Definition at line 261 of file proto-packet.pb.h.

3.20.2.2 InternalArenaConstructable_ typedef void protomessage::ProtoPacket_OnionHead::InternalArenaConstructab [private]

Definition at line 260 of file proto-packet.pb.h.

3.20.3 Member Enumeration Documentation

3.20.3.1 anonymous enum anonymous enum : int

Enumerator

kOnionMessageFieldNumber	
kPaddingFieldNumber	
kOnionIdFieldNumber	

Definition at line 201 of file proto-packet.pb.h.

3.20.4 Constructor & Destructor Documentation

3.20.4.1 ProtoPacket_OnionHead() [1/5] protomessage::ProtoPacket_OnionHead::ProtoPacket_↩ OnionHead ( ) [inline]

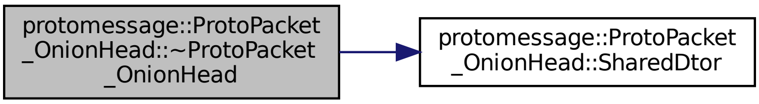
Definition at line 83 of file proto-packet.pb.h.

3.20.4.2 ∼ProtoPacket_OnionHead() protomessage::ProtoPacket_OnionHead::∼ProtoPacket_OnionHead ( ) [override]

Definition at line 228 of file proto-packet.pb.cc.

References SharedDtor().

Here is the call graph for this function:



3.20.4.3 ProtoPacket_OnionHead() `[2/5]`

```
constexpr protomessage::ProtoPacket_OnionHead::Proto↩
Packet_OnionHead (
            ::PROTOBUF_NAMESPACE_ID::internal::ConstantInitialized )  [explicit], [constexpr]
```

Definition at line 20 of file proto-packet.pb.cc.

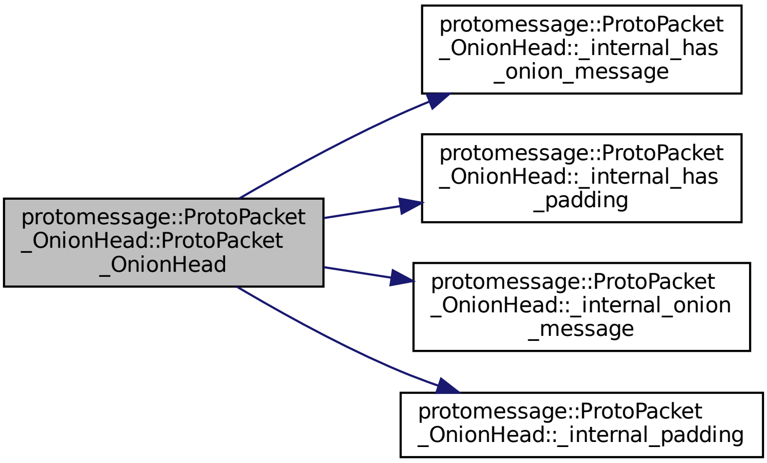
3.20.4.4 ProtoPacket_OnionHead() `[3/5]`

```
protomessage::ProtoPacket_OnionHead::ProtoPacket_↩
OnionHead (
            const ProtoPacket_OnionHead & from )
```

Definition at line 192 of file proto-packet.pb.cc.

References _internal_has_onion_message(), _internal_has_padding(), _internal_onion_message(), _internal_↩ padding(), onion_message_, onionid_, and padding_.

Here is the call graph for this function:


protomessage::ProtoPacket
_OnionHead::ProtoPacket
_OnionHead
protomessage::ProtoPacket
_OnionHead::_internal_has
_onion_message
protomessage::ProtoPacket
_OnionHead::_internal_has
_padding
protomessage::ProtoPacket
_OnionHead::_internal_onion
_message
protomessage::ProtoPacket
_OnionHead::_internal_padding

3.20.4.5 ProtoPacket_OnionHead() [4/5] protomessage::ProtoPacket_OnionHead::ProtoPacket_↩OnionHead (

```
            ProtoPacket_OnionHead && from )  [inline], [noexcept]
```

Definition at line 88 of file proto-packet.pb.h.

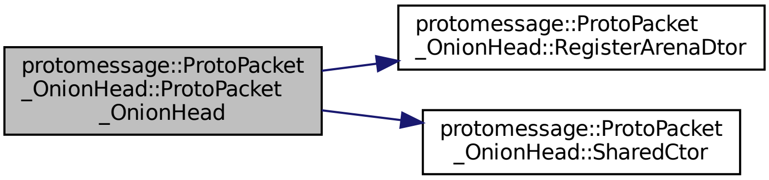
3.20.4.6 ProtoPacket_OnionHead() [5/5] protomessage::ProtoPacket_OnionHead::ProtoPacket_↩OnionHead (

```
            ::PROTOBUF_NAMESPACE_ID::Arena * arena,
            bool is_message_owned = false )  [explicit], [protected]
```

Definition at line 183 of file proto-packet.pb.cc.

References RegisterArenaDtor(), and SharedCtor().

Here is the call graph for this function:


protomessage::ProtoPacket_OnionHead::ProtoPacket_OnionHead
protomessage::ProtoPacket_OnionHead::RegisterArenaDtor
protomessage::ProtoPacket_OnionHead::SharedCtor

3.20.5 Member Function Documentation

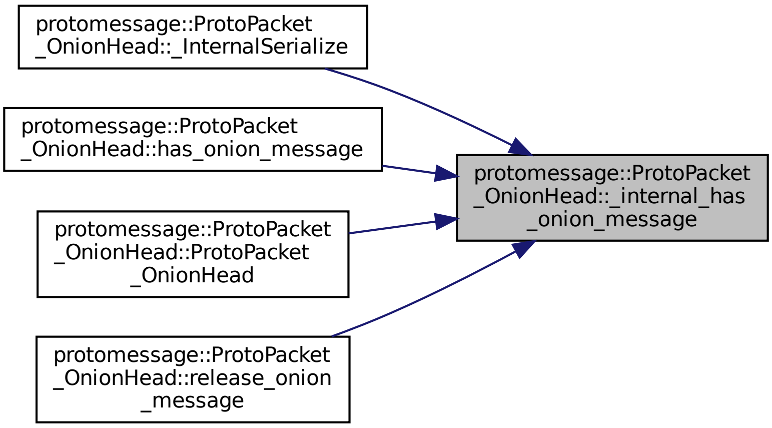
3.20.5.1 _internal_has_onion_message() bool protomessage::ProtoPacket_OnionHead::_internal_↩has_onion_message ( ) const [inline], [private]

Definition at line 836 of file proto-packet.pb.h.

References _has_bits_.

Referenced by _InternalSerialize(), has_onion_message(), ProtoPacket_OnionHead(), and release_onion_↩message().

Here is the caller graph for this function:


protomessage::ProtoPacket _OnionHead::_InternalSerialize
protomessage::ProtoPacket _OnionHead::has_onion_message
protomessage::ProtoPacket _OnionHead::ProtoPacket _OnionHead
protomessage::ProtoPacket _OnionHead::release_onion _message
protomessage::ProtoPacket _OnionHead::_internal_has _onion_message

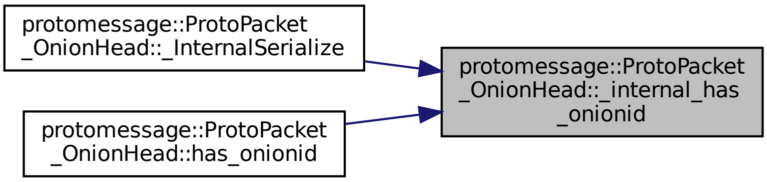
3.20.5.2 _internal_has_onionid() `bool protomessage::ProtoPacket_OnionHead::_internal_has_onionid ( ) const [inline], [private]`

Definition at line 808 of file proto-packet.pb.h.

References _has_bits_.

Referenced by _InternalSerialize(), and has_onionid().

Here is the caller graph for this function:


protomessage::ProtoPacket _OnionHead::_InternalSerialize
protomessage::ProtoPacket _OnionHead::has_onionid
protomessage::ProtoPacket _OnionHead::_internal_has _onionid

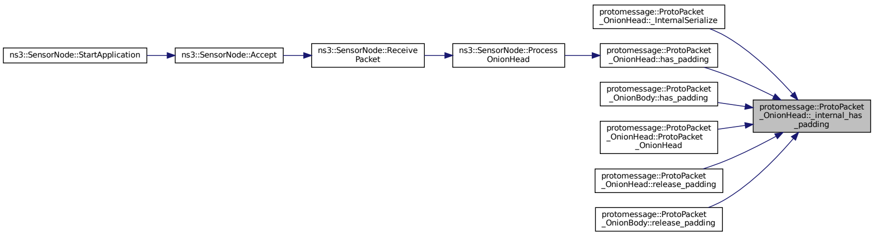
3.20.5.3 _internal_has_padding() `bool protomessage::ProtoPacket_OnionHead::_internal_has_↩ padding ( ) const [inline], [private]`

Definition at line 905 of file proto-packet.pb.h.

References _has_bits_.

Referenced by _InternalSerialize(), has_padding(), protomessage::ProtoPacket_OnionBody::has_padding(), ProtoPacket_OnionHead(), release_padding(), and protomessage::ProtoPacket_OnionBody::release_padding().

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::SensorNode::Process OnionHead
protomessage::ProtoPacket _OnionHead::_InternalSerialize
protomessage::ProtoPacket _OnionHead::has_padding
protomessage::ProtoPacket _OnionBody::has_padding
protomessage::ProtoPacket _OnionHead::ProtoPacket _OnionHead
protomessage::ProtoPacket _OnionHead::release_padding
protomessage::ProtoPacket _OnionBody::release_padding
protomessage::ProtoPacket _OnionHead::_internal_has _padding

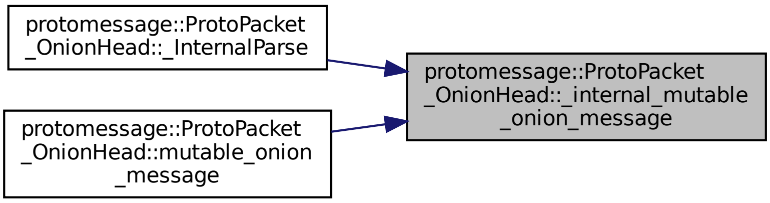
3.20.5.4 _internal_mutable_onion_message() `std::string * protomessage::ProtoPacket_OnionHead↩ ::_internal_mutable_onion_message ( ) [inline], [private]`

Definition at line 870 of file proto-packet.pb.h.

References _has_bits_, and onion_message_.

Referenced by _InternalParse(), and mutable_onion_message().

Here is the caller graph for this function:


protomessage::ProtoPacket _OnionHead::_InternalParse
protomessage::ProtoPacket _OnionHead::mutable_onion _message
protomessage::ProtoPacket _OnionHead::_internal_mutable _onion_message

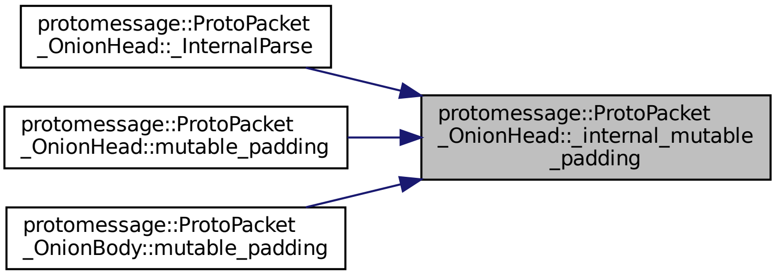
3.20.5.5 _internal_mutable_padding() `std::string * protomessage::ProtoPacket_OnionHead::_internal_mutable_padding ( ) [inline], [private]`

Definition at line 939 of file proto-packet.pb.h.

References _has_bits_, and padding_.

Referenced by _InternalParse(), mutable_padding(), and protomessage::ProtoPacket_OnionBody::mutable_padding().

Here is the caller graph for this function:


protomessage::ProtoPacket _OnionHead::_InternalParse
protomessage::ProtoPacket _OnionHead::mutable_padding
protomessage::ProtoPacket _OnionBody::mutable_padding
protomessage::ProtoPacket _OnionHead::_internal_mutable _padding

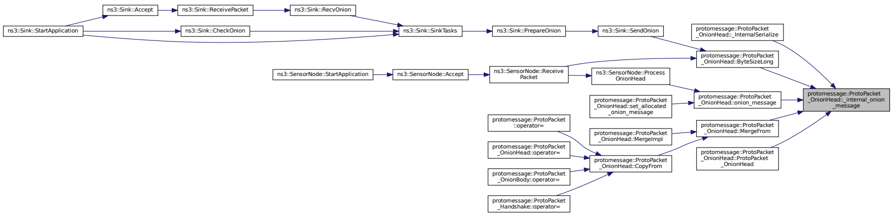
3.20.5.6 _internal_onion_message() `const std::string & protomessage::ProtoPacket_OnionHead::_internal_onion_message ( ) const [inline], [private]`

Definition at line 863 of file proto-packet.pb.h.

References onion_message_.

Referenced by _InternalSerialize(), ByteSizeLong(), MergeFrom(), onion_message(), and ProtoPacket_OnionHead().

Here is the caller graph for this function:


ns3::Sink::StartApplication
ns3::Sink::Accept
ns3::Sink::ReceivePacket
ns3::Sink::RecvOnion
ns3::Sink::CheckOnion
ns3::Sink::SinkTasks
ns3::Sink::PrepareOnion
ns3::Sink::SendOnion
ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::SensorNode::Process OnionHead
protomessage::ProtoPacket _OnionHead::_InternalSerialize
protomessage::ProtoPacket _OnionHead::ByteSizeLong
protomessage::ProtoPacket _OnionHead::set_allocated _onion_message
protomessage::ProtoPacket _OnionHead::onion_message
protomessage::ProtoPacket ::operator=
protomessage::ProtoPacket _OnionHead::MergeImpl
protomessage::ProtoPacket _OnionHead::MergeFrom
protomessage::ProtoPacket _OnionHead::operator=
protomessage::ProtoPacket _OnionHead::CopyFrom
protomessage::ProtoPacket _OnionHead::ProtoPacket _OnionHead
protomessage::ProtoPacket _OnionBody::operator=
protomessage::ProtoPacket _Handshake::operator=
protomessage::ProtoPacket _OnionHead::_internal_onion _message

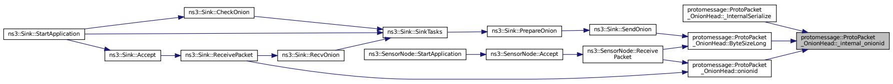
3.20.5.7 _internal_onionid() `int32_t protomessage::ProtoPacket_OnionHead::_internal_onionid ( ) const [inline], [private]`

Definition at line 819 of file proto-packet.pb.h.

References onionid_.

Referenced by _InternalSerialize(), ByteSizeLong(), and onionid().

Here is the caller graph for this function:


ns3::Sink::CheckOnion
protomessage::ProtoPacket _OnionHead::_InternalSerialize
ns3::Sink::StartApplication
ns3::Sink::SinkTasks
ns3::Sink::PrepareOnion
ns3::Sink::SendOnion
protomessage::ProtoPacket _OnionHead::ByteSizeLong
protomessage::ProtoPacket _OnionHead::_internal_onionid
ns3::Sink::Accept
ns3::Sink::ReceivePacket
ns3::Sink::RecvOnion
ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
protomessage::ProtoPacket _OnionHead::onionid

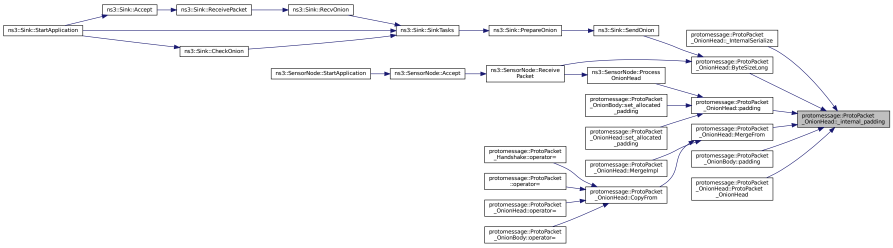
3.20.5.8 _internal_padding() `const std::string & protomessage::ProtoPacket_OnionHead::_internal_padding ( ) const [inline], [private]`

Definition at line 932 of file proto-packet.pb.h.

References padding_.

Referenced by _InternalSerialize(), ByteSizeLong(), MergeFrom(), padding(), protomessage::ProtoPacket_OnionBody::padding(), and ProtoPacket_OnionHead().

Here is the caller graph for this function:


ns3::Sink::Accept
ns3::Sink::ReceivePacket
ns3::Sink::RecvOnion
ns3::Sink::StartApplication
ns3::Sink::SinkTasks
ns3::Sink::PrepareOnion
ns3::Sink::SendOnion
protomessage::ProtoPacket _OnionHead::_InternalSerialize
ns3::Sink::CheckOnion
ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::SensorNode::Process OnionHead
protomessage::ProtoPacket _OnionHead::ByteSizeLong
protomessage::ProtoPacket _OnionBody::set_allocated _padding
protomessage::ProtoPacket _OnionHead::padding
protomessage::ProtoPacket _OnionHead::_internal_padding
protomessage::ProtoPacket _OnionHead::set_allocated _padding
protomessage::ProtoPacket _OnionHead::MergeFrom
protomessage::ProtoPacket _Handshake::operator=
protomessage::ProtoPacket _OnionHead::MergeImpl
protomessage::ProtoPacket _OnionBody::padding
protomessage::ProtoPacket ::operator=
protomessage::ProtoPacket _OnionHead::CopyFrom
protomessage::ProtoPacket _OnionHead::ProtoPacket _OnionHead
protomessage::ProtoPacket _OnionHead::operator=
protomessage::ProtoPacket _OnionBody::operator=

3.20.5.9 _internal_set_onion_message() `void protomessage::ProtoPacket_OnionHead::_internal_↩ set_onion_message (`

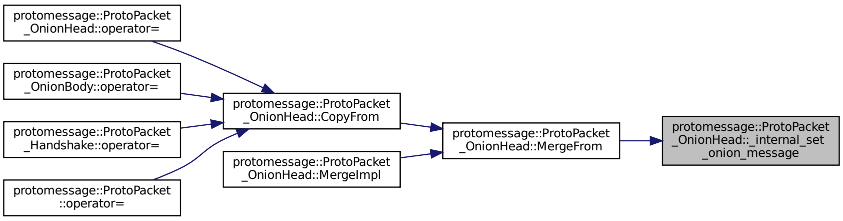
```
            const std::string & value )  [inline], [private]
```

Definition at line 866 of file proto-packet.pb.h.

References _has_bits_, and onion_message_.

Referenced by MergeFrom().

Here is the caller graph for this function:


protomessage::ProtoPacket _OnionHead::operator=
protomessage::ProtoPacket _OnionBody::operator=
protomessage::ProtoPacket _Handshake::operator=
protomessage::ProtoPacket ::operator=
protomessage::ProtoPacket _OnionHead::CopyFrom
protomessage::ProtoPacket _OnionHead::MergeImpl
protomessage::ProtoPacket _OnionHead::MergeFrom
protomessage::ProtoPacket _OnionHead::_internal_set _onion_message

3.20.5.10 _internal_set_onionid() `void protomessage::ProtoPacket_OnionHead::_internal_set_↩ onionid (`

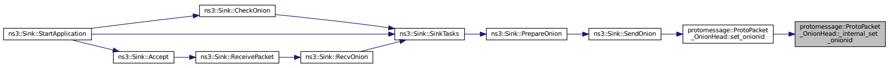
```
            int32_t value )  [inline], [private]
```

Definition at line 826 of file proto-packet.pb.h.

References _has_bits_, and onionid_.

Referenced by set_onionid().

Here is the caller graph for this function:



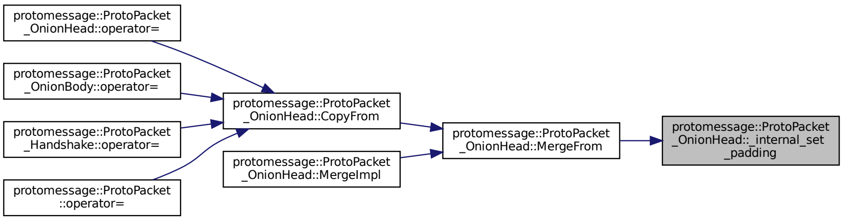
3.20.5.11 _internal_set_padding() `void protomessage::ProtoPacket_OnionHead::_internal_set_↩ padding ( const std::string & value ) [inline], [private]`

Definition at line 935 of file proto-packet.pb.h.

References _has_bits_, and padding_.

Referenced by MergeFrom().

Here is the caller graph for this function:

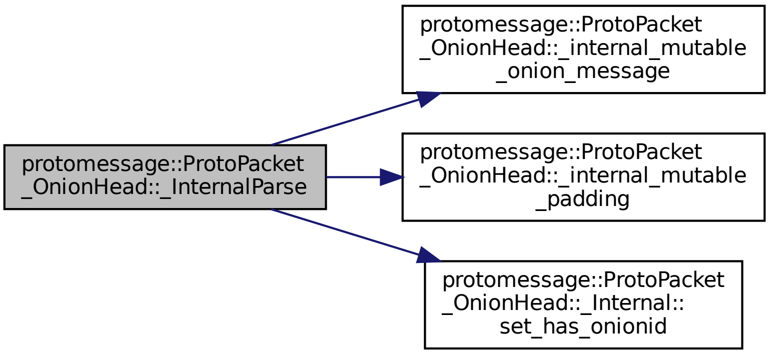

protomessage::ProtoPacket _OnionHead::operator=
protomessage::ProtoPacket _OnionBody::operator=
protomessage::ProtoPacket _Handshake::operator=
protomessage::ProtoPacket ::operator=
protomessage::ProtoPacket _OnionHead::CopyFrom
protomessage::ProtoPacket _OnionHead::MergeImpl
protomessage::ProtoPacket _OnionHead::MergeFrom
protomessage::ProtoPacket _OnionHead::_internal_set _padding

3.20.5.12 _InternalParse() `const char * protomessage::ProtoPacket_OnionHead::_InternalParse ( const char * ptr, ::PROTOBUF_NAMESPACE_ID::internal::ParseContext * ctx ) [final]`

Definition at line 271 of file proto-packet.pb.cc.

References _has_bits_, _internal_mutable_onion_message(), _internal_mutable_padding(), CHK_, onionid_, and protomessage::ProtoPacket_OnionHead::_Internal::set_has_onionid().

Here is the call graph for this function:


protomessage::ProtoPacket _OnionHead::_InternalParse
protomessage::ProtoPacket _OnionHead::_internal_mutable _onion_message
protomessage::ProtoPacket _OnionHead::_internal_mutable _padding
protomessage::ProtoPacket _OnionHead::_Internal:: set_has_onionid

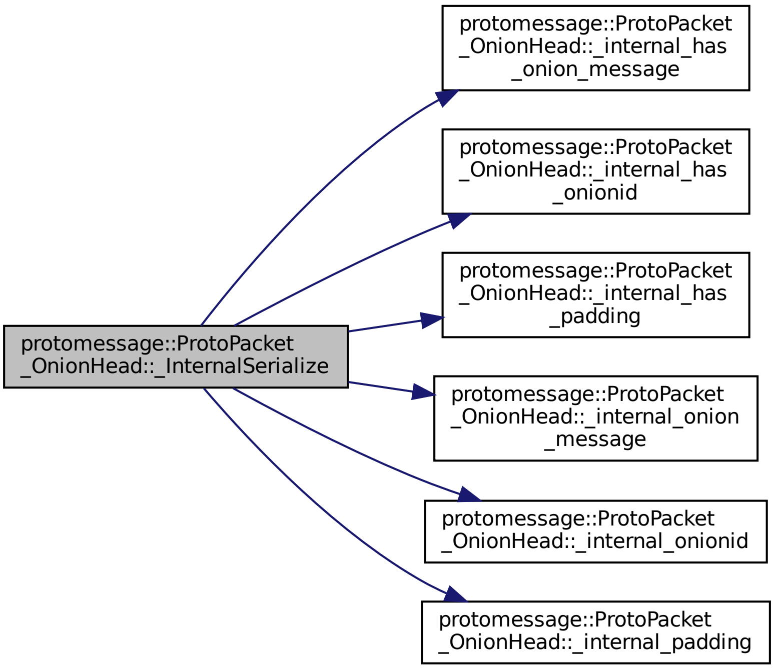
3.20.5.13 _InternalSerialize()

```
uint8_t * protomessage::ProtoPacket_OnionHead::_InternalSerialize
(
            uint8_t * target,
            ::PROTOBUF_NAMESPACE_ID::io::EpsCopyOutputStream * stream ) const [final]
```

Definition at line 329 of file proto-packet.pb.cc.

References _internal_has_onion_message(), _internal_has_onionid(), _internal_has_padding(), _internal_onion_message(), _internal_onionid(), and _internal_padding().

Here is the call graph for this function:


protomessage::ProtoPacket_OnionHead::_InternalSerialize
protomessage::ProtoPacket_OnionHead::_internal_has_onion_message
protomessage::ProtoPacket_OnionHead::_internal_has_onionid
protomessage::ProtoPacket_OnionHead::_internal_has_padding
protomessage::ProtoPacket_OnionHead::_internal_onion_message
protomessage::ProtoPacket_OnionHead::_internal_onionid
protomessage::ProtoPacket_OnionHead::_internal_padding

3.20.5.14 ArenaDtor()

```
void protomessage::ProtoPacket_OnionHead::ArenaDtor (
            void * object ) [static], [private]
```

Definition at line 241 of file proto-packet.pb.cc.

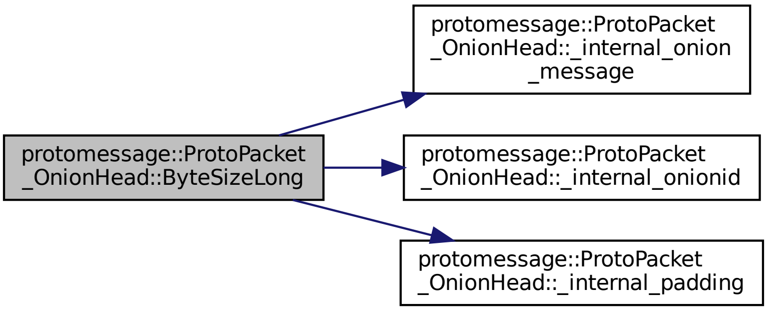
3.20.5.15 ByteSizeLong() `size_t protomessage::ProtoPacket_OnionHead::ByteSizeLong ( ) const [final]`

Definition at line 361 of file proto-packet.pb.cc.

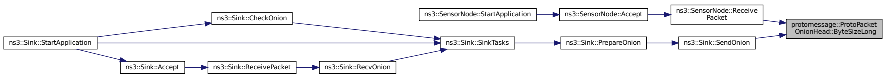
References _cached_size_, _has_bits_, _internal_onion_message(), _internal_onionid(), and _internal_padding().

Referenced by ns3::SensorNode::ReceivePacket(), and ns3::Sink::SendOnion().

Here is the call graph for this function:


protomessage::ProtoPacket _OnionHead::_internal_onion _message
protomessage::ProtoPacket _OnionHead::ByteSizeLong
protomessage::ProtoPacket _OnionHead::_internal_onionid
protomessage::ProtoPacket _OnionHead::_internal_padding

Here is the caller graph for this function:



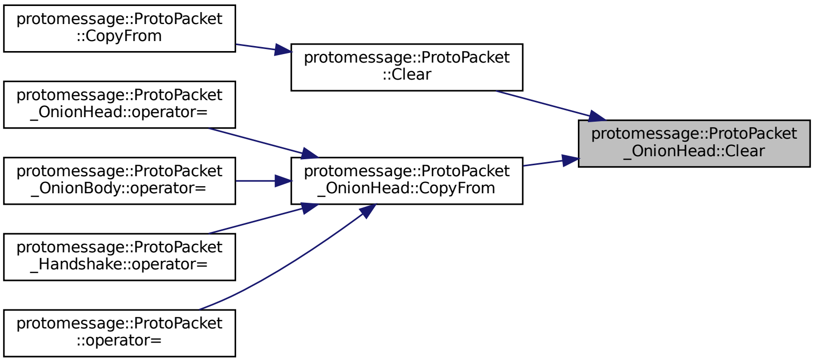
3.20.5.16 Clear() `void protomessage::ProtoPacket_OnionHead::Clear ( ) [final]`

Definition at line 251 of file proto-packet.pb.cc.

References _has_bits_, onion_message_, onionid_, and padding_.

Referenced by protomessage::ProtoPacket::Clear(), and CopyFrom().

Here is the caller graph for this function:


protomessage::ProtoPacket ::CopyFrom
protomessage::ProtoPacket _OnionHead::operator=
protomessage::ProtoPacket _OnionBody::operator=
protomessage::ProtoPacket _Handshake::operator=
protomessage::ProtoPacket ::operator=
protomessage::ProtoPacket ::Clear
protomessage::ProtoPacket _OnionHead::CopyFrom
protomessage::ProtoPacket _OnionHead::Clear

3.20.5.17 clear_onion_message() `void protomessage::ProtoPacket_OnionHead::clear_onion_message ( ) [inline]`

Definition at line 843 of file proto-packet.pb.h.

References _has_bits_, and onion_message_.

3.20.5.18 clear_onionid() `void protomessage::ProtoPacket_OnionHead::clear_onionid ( ) [inline]`

Definition at line 815 of file proto-packet.pb.h.

References _has_bits_, and onionid_.

3.20.5.19 clear_padding() `void protomessage::ProtoPacket_OnionHead::clear_padding ( ) [inline]`

Definition at line 912 of file proto-packet.pb.h.

References _has_bits_, and padding_.

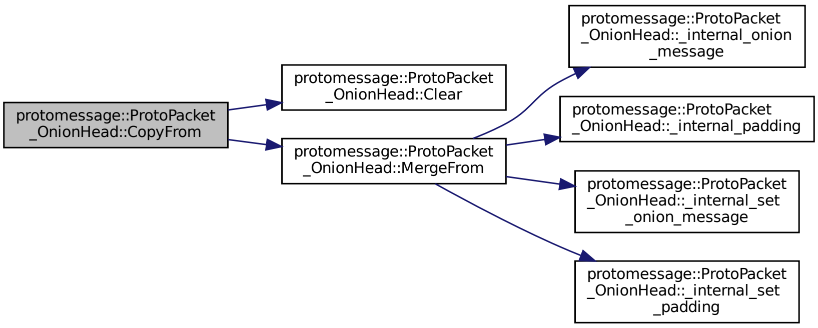
3.20.5.20 CopyFrom() `void protomessage::ProtoPacket_OnionHead::CopyFrom ( const ProtoPacket_OnionHead & from )`

Definition at line 429 of file proto-packet.pb.cc.

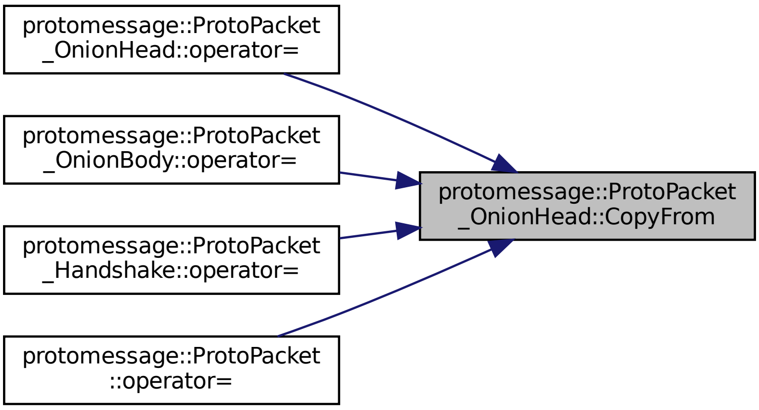
References Clear(), and MergeFrom().

Referenced by operator=(), protomessage::ProtoPacket_OnionBody::operator=(), protomessage::ProtoPacket_↩ Handshake::operator=(), and protomessage::ProtoPacket::operator=().

Here is the call graph for this function:


protomessage::ProtoPacket _OnionHead::CopyFrom
protomessage::ProtoPacket _OnionHead::Clear
protomessage::ProtoPacket _OnionHead::MergeFrom
protomessage::ProtoPacket _OnionHead::_internal_onion _message
protomessage::ProtoPacket _OnionHead::_internal_padding
protomessage::ProtoPacket _OnionHead::_internal_set _onion_message
protomessage::ProtoPacket _OnionHead::_internal_set _padding

Here is the caller graph for this function:


protomessage::ProtoPacket _OnionHead::operator=
protomessage::ProtoPacket _OnionBody::operator=
protomessage::ProtoPacket _Handshake::operator=
protomessage::ProtoPacket ::operator=
protomessage::ProtoPacket _OnionHead::CopyFrom

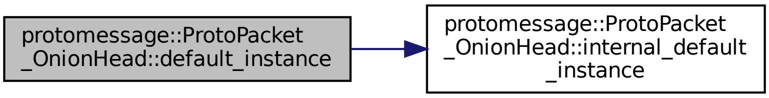
3.20.5.21 default_instance() `static const ProtoPacket_OnionHead& protomessage::ProtoPacket_↩ OnionHead::default_instance ( ) [inline], [static]`

Definition at line 120 of file proto-packet.pb.h.

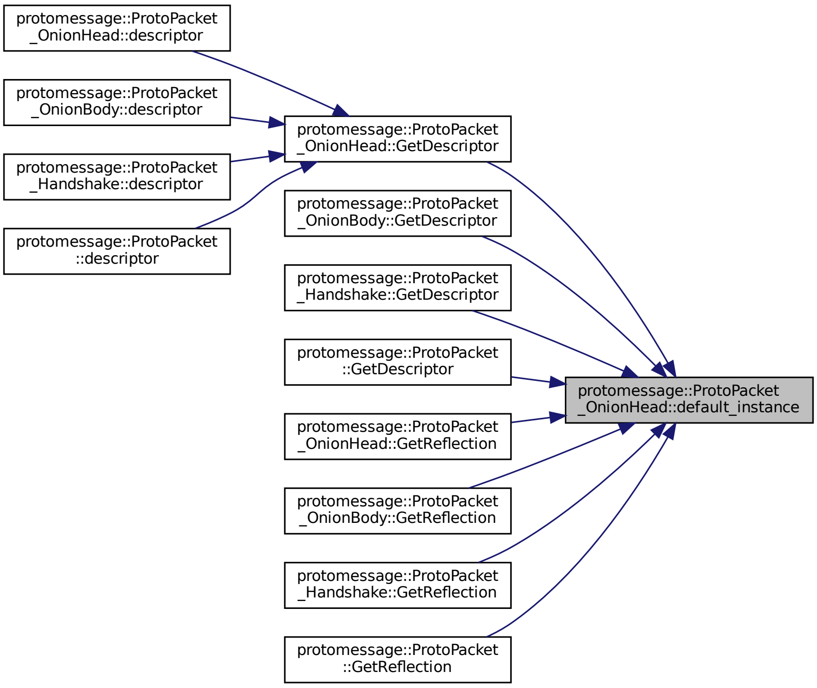
References internal_default_instance().

Referenced by GetDescriptor(), protomessage::ProtoPacket_OnionBody::GetDescriptor(), protomessage↩ ::ProtoPacket_Handshake::GetDescriptor(), protomessage::ProtoPacket::GetDescriptor(), GetReflection(), protomessage::ProtoPacket_OnionBody::GetReflection(), protomessage::ProtoPacket_Handshake::GetReflection(), and protomessage::ProtoPacket::GetReflection().

Here is the call graph for this function:


protomessage::ProtoPacket _OnionHead::default_instance
protomessage::ProtoPacket _OnionHead::internal_default _instance

Here is the caller graph for this function:

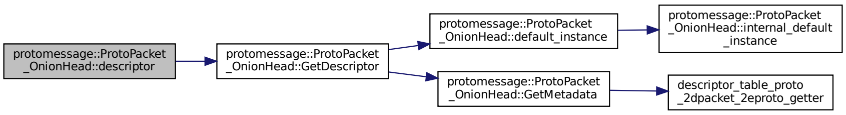

protomessage::ProtoPacket _OnionHead::descriptor
protomessage::ProtoPacket _OnionBody::descriptor
protomessage::ProtoPacket _Handshake::descriptor
protomessage::ProtoPacket ::descriptor
protomessage::ProtoPacket _OnionHead::GetDescriptor
protomessage::ProtoPacket _OnionBody::GetDescriptor
protomessage::ProtoPacket _Handshake::GetDescriptor
protomessage::ProtoPacket ::GetDescriptor
protomessage::ProtoPacket _OnionHead::GetReflection
protomessage::ProtoPacket _OnionBody::GetReflection
protomessage::ProtoPacket _Handshake::GetReflection
protomessage::ProtoPacket ::GetReflection
protomessage::ProtoPacket _OnionHead::default_instance

3.20.5.22 descriptor() `static const ::PROTOBUF_NAMESPACE_ID::Descriptor* protomessage::Proto↩ Packet_OnionHead::descriptor ( ) [inline], [static]`

Definition at line 111 of file proto-packet.pb.h.

References GetDescriptor().

Here is the call graph for this function:


protomessage::ProtoPacket _OnionHead::descriptor
protomessage::ProtoPacket _OnionHead::GetDescriptor
protomessage::ProtoPacket _OnionHead::default_instance
protomessage::ProtoPacket _OnionHead::internal_default _instance
protomessage::ProtoPacket _OnionHead::GetMetadata
descriptor_table_proto _2dpacket_2eproto_getter

3.20.5.23 FullMessageName() `::PROTOBUF_NAMESPACE_ID::StringPiece protomessage::ProtoPacket_↩ OnionHead::FullMessageName ( ) [inline], [static], [private]`

Definition at line 181 of file proto-packet.pb.h.

3.20.5.24 GetCachedSize() `int protomessage::ProtoPacket_OnionHead::GetCachedSize ( ) const [inline], [final]`

Definition at line 171 of file proto-packet.pb.h.

References _cached_size_.

3.20.5.25 GetClassData() `const ::PROTOBUF_NAMESPACE_ID::Message::ClassData * protomessage::↩ ProtoPacket_OnionHead::GetClassData ( ) const [final]`

Definition at line 398 of file proto-packet.pb.cc.

References _class_data_.

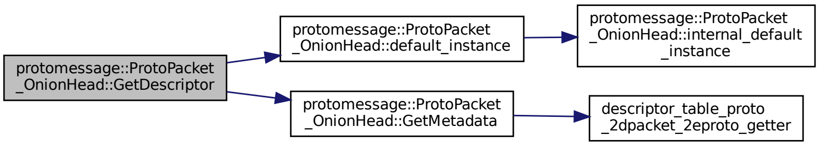
3.20.5.26 GetDescriptor() `static const ::PROTOBUF_NAMESPACE_ID::Descriptor* protomessage::↩ ProtoPacket_OnionHead::GetDescriptor ( ) [inline], [static]`

Definition at line 114 of file proto-packet.pb.h.

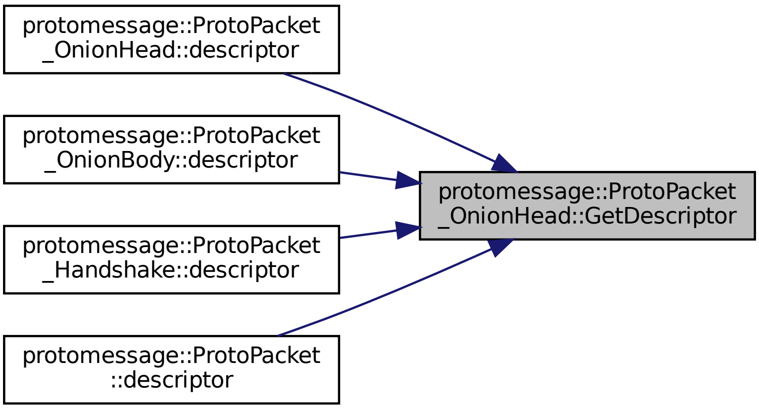
References default_instance(), and GetMetadata().

Referenced by descriptor(), protomessage::ProtoPacket_OnionBody::descriptor(), protomessage::ProtoPacket_↩ Handshake::descriptor(), and protomessage::ProtoPacket::descriptor().

Here is the call graph for this function:


protomessage::ProtoPacket _OnionHead::GetDescriptor
protomessage::ProtoPacket _OnionHead::default_instance
protomessage::ProtoPacket _OnionHead::internal_default _instance
protomessage::ProtoPacket _OnionHead::GetMetadata
descriptor_table_proto _2dpacket_2eproto_getter

Here is the caller graph for this function:


protomessage::ProtoPacket _OnionHead::descriptor
protomessage::ProtoPacket _OnionBody::descriptor
protomessage::ProtoPacket _Handshake::descriptor
protomessage::ProtoPacket ::descriptor
protomessage::ProtoPacket _OnionHead::GetDescriptor

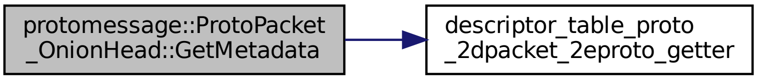
3.20.5.27 GetMetadata() `PROTOBUF_NAMESPACE_ID::Metadata protomessage::ProtoPacket_OnionHead::↩ GetMetadata ( ) const [final]`

Definition at line 459 of file proto-packet.pb.cc.

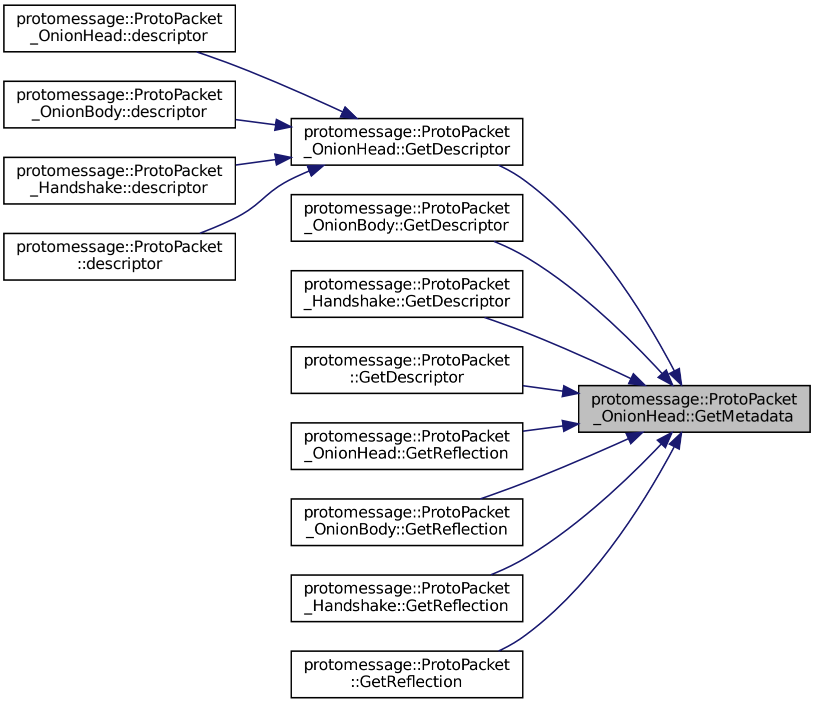
References descriptor_table_proto_2dpacket_2eproto_getter(), descriptor_table_proto_2dpacket_2eproto_once, and file_level_metadata_proto_2dpacket_2eproto.

Referenced by GetDescriptor(), protomessage::ProtoPacket_OnionBody::GetDescriptor(), protomessage↩ ::ProtoPacket_Handshake::GetDescriptor(), protomessage::ProtoPacket::GetDescriptor(), GetReflection(), protomessage::ProtoPacket_OnionBody::GetReflection(), protomessage::ProtoPacket_Handshake::GetReflection(), and protomessage::ProtoPacket::GetReflection().

Here is the call graph for this function:



Here is the caller graph for this function:

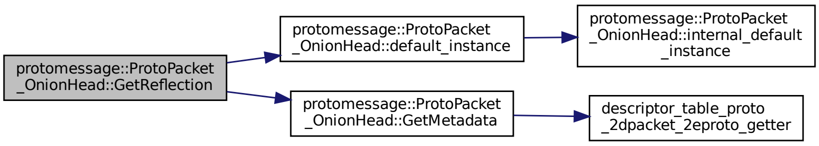

protomessage::ProtoPacket
_OnionHead::descriptor
protomessage::ProtoPacket
_OnionBody::descriptor
protomessage::ProtoPacket
_Handshake::descriptor
protomessage::ProtoPacket
::descriptor
protomessage::ProtoPacket
_OnionHead::GetDescriptor
protomessage::ProtoPacket
_OnionBody::GetDescriptor
protomessage::ProtoPacket
_Handshake::GetDescriptor
protomessage::ProtoPacket
::GetDescriptor
protomessage::ProtoPacket
_OnionHead::GetReflection
protomessage::ProtoPacket
_OnionBody::GetReflection
protomessage::ProtoPacket
_Handshake::GetReflection
protomessage::ProtoPacket
::GetReflection
protomessage::ProtoPacket
_OnionHead::GetMetadata

3.20.5.28 GetReflection() `static const ::PROTOBUF_NAMESPACE_ID::Reflection* protomessage::↩ ProtoPacket_OnionHead::GetReflection ( ) [inline], [static]`

Definition at line 117 of file proto-packet.pb.h.

References default_instance(), and GetMetadata().

Here is the call graph for this function:

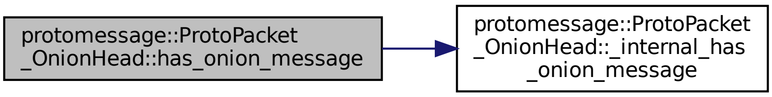

protomessage::ProtoPacket
_OnionHead::GetReflection
protomessage::ProtoPacket
_OnionHead::default_instance
protomessage::ProtoPacket
_OnionHead::internal_default
_instance
protomessage::ProtoPacket
_OnionHead::GetMetadata
descriptor_table_proto
_2dpacket_2eproto_getter

3.20.5.29 has_onion_message() `bool protomessage::ProtoPacket_OnionHead::has_onion_message ( ) const [inline]`

Definition at line 840 of file proto-packet.pb.h.

References _internal_has_onion_message().

Here is the call graph for this function:

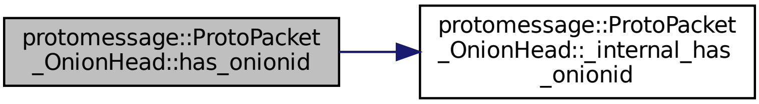

protomessage::ProtoPacket
_OnionHead::has_onion_message
protomessage::ProtoPacket
_OnionHead::_internal_has
_onion_message

3.20.5.30 has_onionid() `bool protomessage::ProtoPacket_OnionHead::has_onionid ( ) const [inline]`

Definition at line 812 of file proto-packet.pb.h.

References _internal_has_onionid().

Here is the call graph for this function:


protomessage::ProtoPacket
_OnionHead::has_onionid
protomessage::ProtoPacket
_OnionHead::_internal_has
_onionid

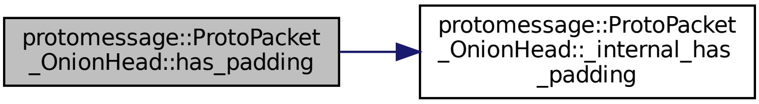
3.20.5.31 has_padding() `bool protomessage::ProtoPacket_OnionHead::has_padding ( ) const [inline]`

Definition at line 909 of file proto-packet.pb.h.

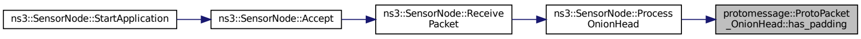
References _internal_has_padding().

Referenced by ns3::SensorNode::ProcessOnionHead().

Here is the call graph for this function:


protomessage::ProtoPacket _OnionHead::has_padding
protomessage::ProtoPacket _OnionHead::_internal_has _padding

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::SensorNode::Process OnionHead
protomessage::ProtoPacket _OnionHead::has_padding

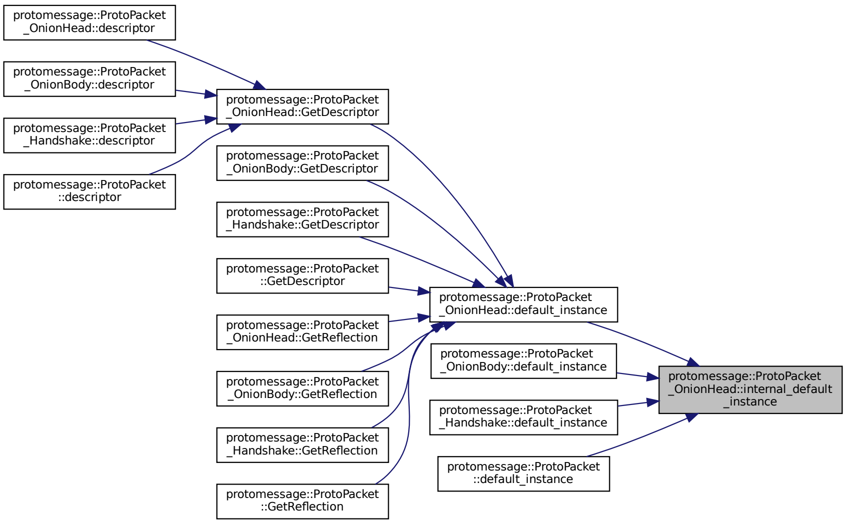
3.20.5.32 internal_default_instance() `static const ProtoPacket_OnionHead* protomessage::ProtoPacket_OnionHead::internal_default_instance ( ) [inline], [static]`

Definition at line 123 of file proto-packet.pb.h.

References protomessage::_ProtoPacket_OnionHead_default_instance_.

Referenced by default_instance(), protomessage::ProtoPacket_OnionBody::default_instance(), protomessage::ProtoPacket_Handshake::default_instance(), and protomessage::ProtoPacket::default_instance().

Here is the caller graph for this function:


protomessage::ProtoPacket _OnionHead::descriptor
protomessage::ProtoPacket _OnionBody::descriptor
protomessage::ProtoPacket _Handshake::descriptor
protomessage::ProtoPacket ::descriptor
protomessage::ProtoPacket _OnionHead::GetDescriptor
protomessage::ProtoPacket _OnionBody::GetDescriptor
protomessage::ProtoPacket _Handshake::GetDescriptor
protomessage::ProtoPacket ::GetDescriptor
protomessage::ProtoPacket _OnionHead::GetReflection
protomessage::ProtoPacket _OnionBody::GetReflection
protomessage::ProtoPacket _Handshake::GetReflection
protomessage::ProtoPacket ::GetReflection
protomessage::ProtoPacket _OnionHead::default_instance
protomessage::ProtoPacket _OnionBody::default_instance
protomessage::ProtoPacket _Handshake::default_instance
protomessage::ProtoPacket ::default_instance
protomessage::ProtoPacket _OnionHead::internal_default _instance

3.20.5.33 InternalSwap() `void protomessage::ProtoPacket_OnionHead::InternalSwap ( ProtoPacket_OnionHead * other ) [private]`

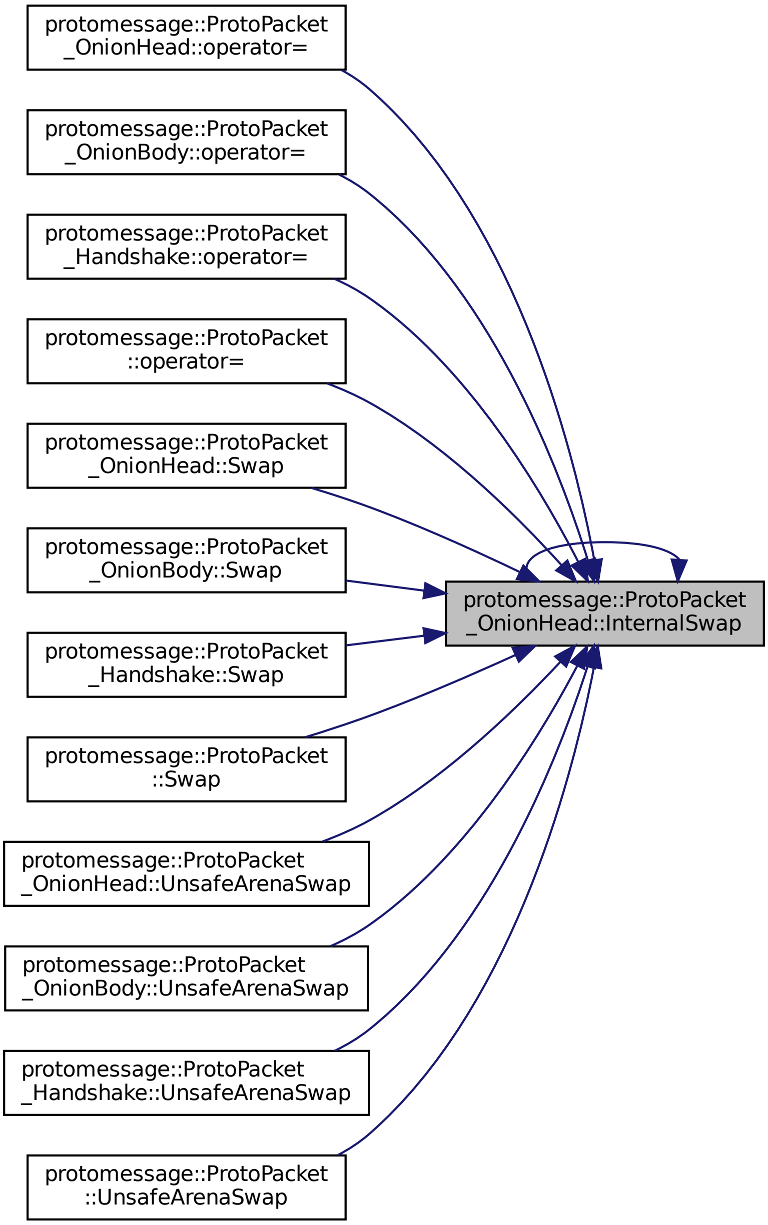
Definition at line 440 of file proto-packet.pb.cc.

References _has_bits_, InternalSwap(), onion_message_, onionid_, padding_, and swap.

Referenced by InternalSwap(), operator=(), protomessage::ProtoPacket_OnionBody::operator=(), protomessage::ProtoPacket_Handshake::operator=(), protomessage::ProtoPacket::operator=(), Swap(), protomessage::ProtoPacket_OnionBody::Swap(), protomessage::ProtoPacket_Handshake::Swap(), protomessage::ProtoPacket::Swap(), UnsafeArenaSwap(), protomessage::ProtoPacket_OnionBody::UnsafeArenaSwap(), protomessage::ProtoPacket_Handshake::UnsafeArenaSwap(), and protomessage::ProtoPacket::UnsafeArenaSwap().

Here is the call graph for this function:

Here is the caller graph for this function:


protomessage::ProtoPacket _OnionHead::operator=
protomessage::ProtoPacket _OnionBody::operator=
protomessage::ProtoPacket _Handshake::operator=
protomessage::ProtoPacket ::operator=
protomessage::ProtoPacket _OnionHead::Swap
protomessage::ProtoPacket _OnionBody::Swap
protomessage::ProtoPacket _Handshake::Swap
protomessage::ProtoPacket ::Swap
protomessage::ProtoPacket _OnionHead::UnsafeArenaSwap
protomessage::ProtoPacket _OnionBody::UnsafeArenaSwap
protomessage::ProtoPacket _Handshake::UnsafeArenaSwap
protomessage::ProtoPacket ::UnsafeArenaSwap
protomessage::ProtoPacket _OnionHead::InternalSwap

3.20.5.34 IsInitialized() `bool protomessage::ProtoPacket_OnionHead::IsInitialized ( ) const [final]`

Definition at line 436 of file proto-packet.pb.cc.

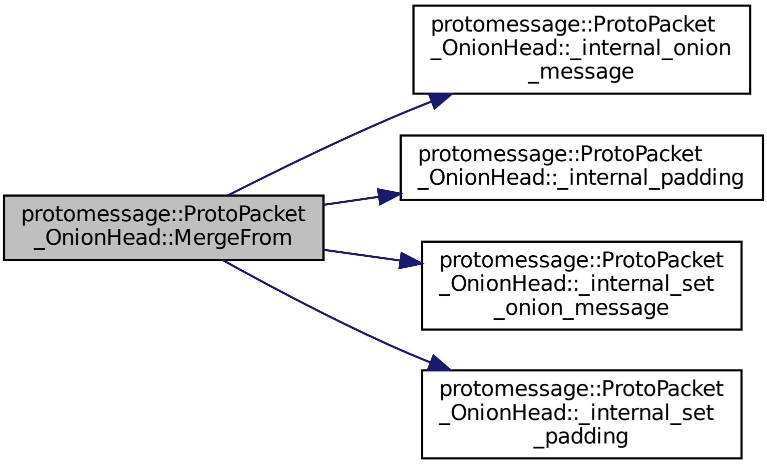
3.20.5.35 MergeFrom() `void protomessage::ProtoPacket_OnionHead::MergeFrom ( const ProtoPacket_OnionHead & from )`

Definition at line 407 of file proto-packet.pb.cc.

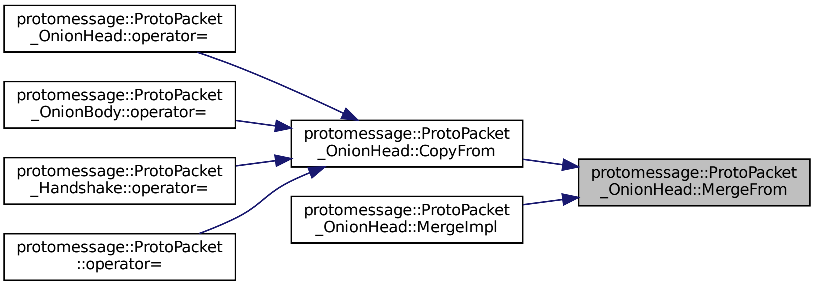
References _has_bits_, _internal_onion_message(), _internal_padding(), _internal_set_onion_message(), _internal_set_padding(), and onionid_.

Referenced by CopyFrom(), and MergeImpl().

Here is the call graph for this function:


protomessage::ProtoPacket_OnionHead::MergeFrom
protomessage::ProtoPacket_OnionHead::_internal_onion_message
protomessage::ProtoPacket_OnionHead::_internal_padding
protomessage::ProtoPacket_OnionHead::_internal_set_onion_message
protomessage::ProtoPacket_OnionHead::_internal_set_padding

Here is the caller graph for this function:

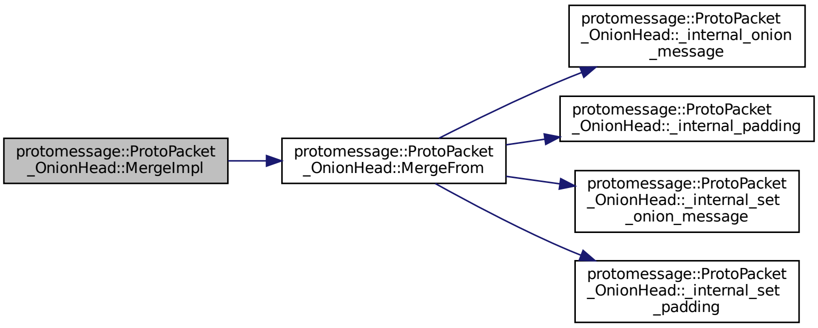

protomessage::ProtoPacket_OnionHead::operator=
protomessage::ProtoPacket_OnionBody::operator=
protomessage::ProtoPacket_Handshake::operator=
protomessage::ProtoPacket::operator=
protomessage::ProtoPacket_OnionHead::CopyFrom
protomessage::ProtoPacket_OnionHead::MergeImpl
protomessage::ProtoPacket_OnionHead::MergeFrom

3.20.5.36 MergeImpl() `void protomessage::ProtoPacket_OnionHead::MergeImpl ( ::PROTOBUF_NAMESPACE_ID::Message * to, const ::PROTOBUF_NAMESPACE_ID::Message & from ) [static], [private]`

Definition at line 400 of file proto-packet.pb.cc.

References MergeFrom().

Here is the call graph for this function:

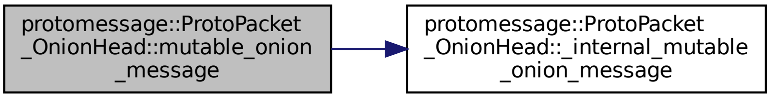

protomessage::ProtoPacket _OnionHead::MergeImpl
protomessage::ProtoPacket _OnionHead::MergeFrom
protomessage::ProtoPacket _OnionHead::_internal_onion _message
protomessage::ProtoPacket _OnionHead::_internal_padding
protomessage::ProtoPacket _OnionHead::_internal_set _onion_message
protomessage::ProtoPacket _OnionHead::_internal_set _padding

3.20.5.37 mutable_onion_message() `std::string * protomessage::ProtoPacket_OnionHead::mutable←↩ _onion_message ( ) [inline]`

Definition at line 858 of file proto-packet.pb.h.

References _internal_mutable_onion_message().

Here is the call graph for this function:

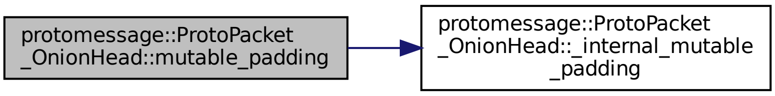

protomessage::ProtoPacket _OnionHead::mutable_onion _message
protomessage::ProtoPacket _OnionHead::_internal_mutable _onion_message

3.20.5.38 mutable_padding() `std::string * protomessage::ProtoPacket_OnionHead::mutable_padding ( ) [inline]`

Definition at line 927 of file proto-packet.pb.h.

References _internal_mutable_padding().

Here is the call graph for this function:


protomessage::ProtoPacket
_OnionHead::mutable_padding
protomessage::ProtoPacket
_OnionHead::_internal_mutable
_padding

3.20.5.39 New() `ProtoPacket_OnionHead* protomessage::ProtoPacket_OnionHead::New ( ::PROTOBUF_NAMESPACE_ID::Arena * arena = nullptr ) const [inline], [final]`

Definition at line 154 of file proto-packet.pb.h.

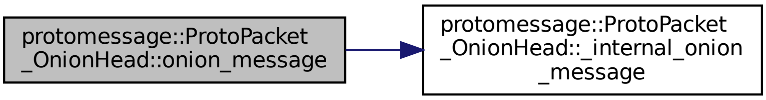
3.20.5.40 onion_message() `const std::string & protomessage::ProtoPacket_OnionHead::onion_message ( ) const [inline]`

Definition at line 847 of file proto-packet.pb.h.

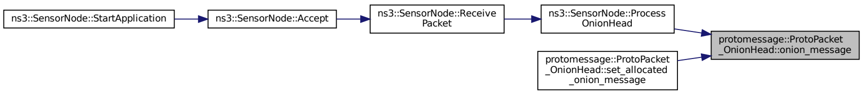
References _internal_onion_message().

Referenced by ns3::SensorNode::ProcessOnionHead(), and set_allocated_onion_message().

Here is the call graph for this function:


protomessage::ProtoPacket
_OnionHead::onion_message
protomessage::ProtoPacket
_OnionHead::_internal_onion
_message

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive
Packet
ns3::SensorNode::Process
OnionHead
protomessage::ProtoPacket
_OnionHead::set_allocated
_onion_message
protomessage::ProtoPacket
_OnionHead::onion_message

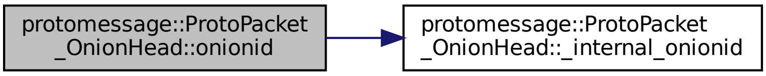
3.20.5.41 onionid() `int32_t protomessage::ProtoPacket_OnionHead::onionid ( ) const [inline]`

Definition at line 822 of file proto-packet.pb.h.

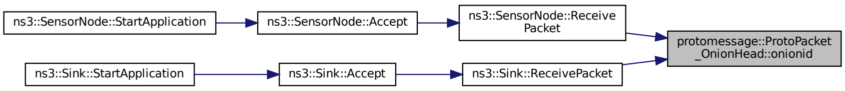
References _internal_onionid().

Referenced by ns3::SensorNode::ReceivePacket(), and ns3::Sink::ReceivePacket().

Here is the call graph for this function:


protomessage::ProtoPacket _OnionHead::onionid
protomessage::ProtoPacket _OnionHead::_internal_onionid

Here is the caller graph for this function:

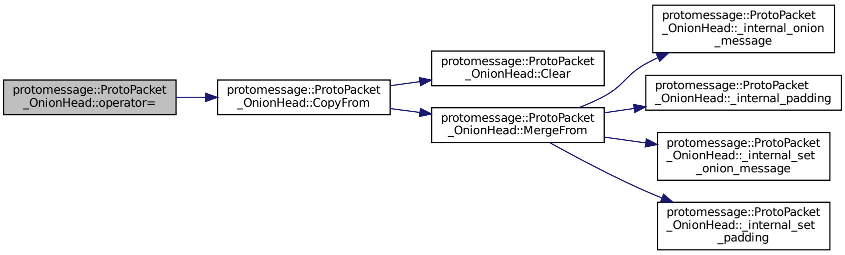

ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::Sink::StartApplication
ns3::Sink::Accept
ns3::Sink::ReceivePacket
protomessage::ProtoPacket _OnionHead::onionid

3.20.5.42 operator=() `[1/2]` `ProtoPacket_OnionHead& protomessage::ProtoPacket_OnionHead::operator= ( const ProtoPacket_OnionHead & from ) [inline]`

Definition at line 93 of file proto-packet.pb.h.

References CopyFrom().

Here is the call graph for this function:

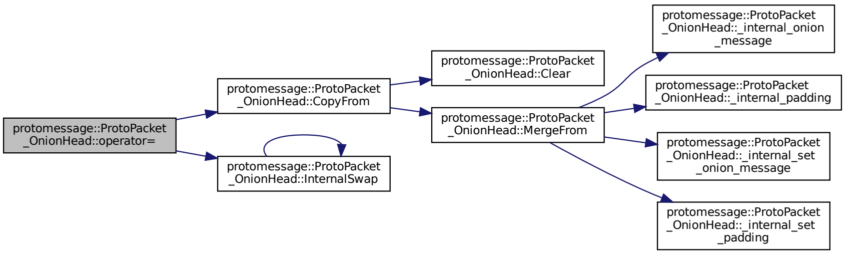

protomessage::ProtoPacket _OnionHead::operator=
protomessage::ProtoPacket _OnionHead::CopyFrom
protomessage::ProtoPacket _OnionHead::Clear
protomessage::ProtoPacket _OnionHead::MergeFrom
protomessage::ProtoPacket _OnionHead::_internal_onion _message
protomessage::ProtoPacket _OnionHead::_internal_padding
protomessage::ProtoPacket _OnionHead::_internal_set _onion_message
protomessage::ProtoPacket _OnionHead::_internal_set _padding

3.20.5.43 operator=() [2/2] ProtoPacket_OnionHead& protomessage::ProtoPacket_OnionHead::operator= (ProtoPacket_OnionHead && *from*) [inline], [noexcept]

Definition at line 97 of file proto-packet.pb.h.

References CopyFrom(), and InternalSwap().

Here is the call graph for this function:


protomessage::ProtoPacket
_OnionHead::operator=
protomessage::ProtoPacket
_OnionHead::CopyFrom
protomessage::ProtoPacket
_OnionHead::InternalSwap
protomessage::ProtoPacket
_OnionHead::Clear
protomessage::ProtoPacket
_OnionHead::MergeFrom
protomessage::ProtoPacket
_OnionHead::_internal_onion
_message
protomessage::ProtoPacket
_OnionHead::_internal_padding
protomessage::ProtoPacket
_OnionHead::_internal_set
_onion_message
protomessage::ProtoPacket
_OnionHead::_internal_set
_padding

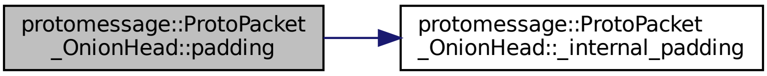
3.20.5.44 padding() const std::string & protomessage::ProtoPacket_OnionHead::padding ( ) const [inline]

Definition at line 916 of file proto-packet.pb.h.

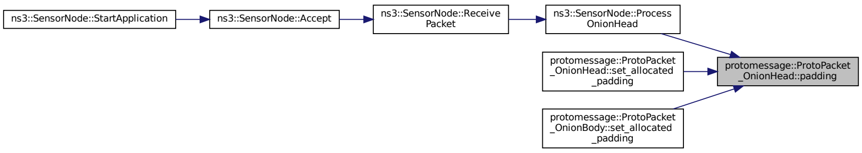
References _internal_padding().

Referenced by ns3::SensorNode::ProcessOnionHead(), set_allocated_padding(), and protomessage::Proto↵ Packet_OnionBody::set_allocated_padding().

Here is the call graph for this function:


protomessage::ProtoPacket
_OnionHead::padding
protomessage::ProtoPacket
_OnionHead::_internal_padding

Here is the caller graph for this function:

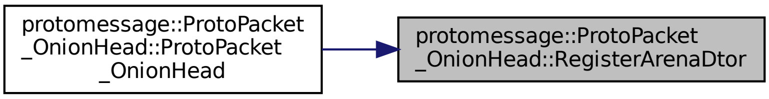

ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive
Packet
ns3::SensorNode::Process
OnionHead
protomessage::ProtoPacket
_OnionHead::set_allocated
_padding
protomessage::ProtoPacket
_OnionBody::set_allocated
_padding
protomessage::ProtoPacket
_OnionHead::padding

3.20.5.45 RegisterArenaDtor() `void protomessage::ProtoPacket_OnionHead::RegisterArenaDtor ( ::PROTOBUF_NAMESPACE_ID::Arena * arena ) [inline], [private]`

Definition at line 245 of file proto-packet.pb.cc.

Referenced by ProtoPacket_OnionHead().

Here is the caller graph for this function:

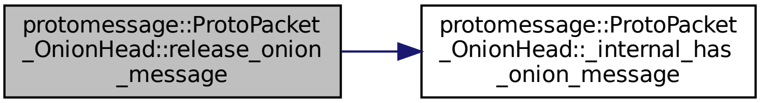

protomessage::ProtoPacket
_OnionHead::ProtoPacket
_OnionHead
protomessage::ProtoPacket
_OnionHead::RegisterArenaDtor

3.20.5.46 release_onion_message() `std::string * protomessage::ProtoPacket_OnionHead::release←_onion_message ( ) [inline]`

Definition at line 874 of file proto-packet.pb.h.

References _has_bits_, _internal_has_onion_message(), and onion_message_.

Here is the call graph for this function:

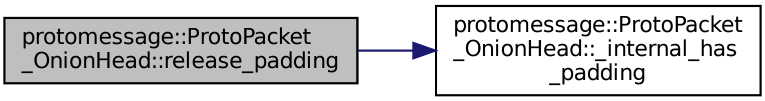

protomessage::ProtoPacket
_OnionHead::release_onion
_message
protomessage::ProtoPacket
_OnionHead::_internal_has
_onion_message

3.20.5.47 release_padding() `std::string * protomessage::ProtoPacket_OnionHead::release_padding ( ) [inline]`

Definition at line 943 of file proto-packet.pb.h.

References _has_bits_, _internal_has_padding(), and padding_.

Here is the call graph for this function:


protomessage::ProtoPacket
_OnionHead::release_padding
protomessage::ProtoPacket
_OnionHead::_internal_has
_padding

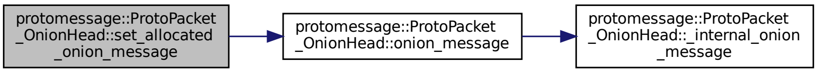
3.20.5.48 set_allocated_onion_message() `void protomessage::ProtoPacket_OnionHead::set_allocated←_onion_message (`

```
            std::string * onion_message )  [inline]
```

Definition at line 888 of file proto-packet.pb.h.

References _has_bits_, onion_message(), and onion_message_.

Here is the call graph for this function:


protomessage::ProtoPacket _OnionHead::set_allocated _onion_message
protomessage::ProtoPacket _OnionHead::onion_message
protomessage::ProtoPacket _OnionHead::_internal_onion _message

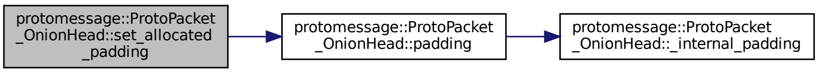
3.20.5.49 set_allocated_padding() `void protomessage::ProtoPacket_OnionHead::set_allocated_←padding (`

```
            std::string * padding )  [inline]
```

Definition at line 957 of file proto-packet.pb.h.

References _has_bits_, padding(), and padding_.

Here is the call graph for this function:

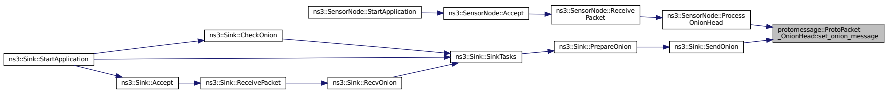

protomessage::ProtoPacket _OnionHead::set_allocated _padding
protomessage::ProtoPacket _OnionHead::padding
protomessage::ProtoPacket _OnionHead::_internal_padding

3.20.5.50 set_onion_message() [1/2] `template<typename ArgT0 = const std::string&, typename... ArgT>`

```
void protomessage::ProtoPacket_OnionHead::set_onion_message (
            ArgT0 && arg0,
            ArgT... args )
```

Referenced by ns3::SensorNode::ProcessOnionHead(), and ns3::Sink::SendOnion().

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::SensorNode::Process OnionHead
protomessage::ProtoPacket _OnionHead::set_onion_message
ns3::Sink::CheckOnion
ns3::Sink::StartApplication
ns3::Sink::SinkTasks
ns3::Sink::PrepareOnion
ns3::Sink::SendOnion
ns3::Sink::Accept
ns3::Sink::ReceivePacket
ns3::Sink::RecvOnion

3.20.5.51 set_onion_message() [2/2] `template<typename ArgT0 , typename... ArgT>`

```
PROTOBUF_ALWAYS_INLINE void protomessage::ProtoPacket_OnionHead::set_onion_message (
            ArgT0 && arg0,
            ArgT... args ) [inline]
```

Definition at line 853 of file proto-packet.pb.h.

References _has_bits_, and onion_message_.

3.20.5.52 set_onionid() `void protomessage::ProtoPacket_OnionHead::set_onionid (`

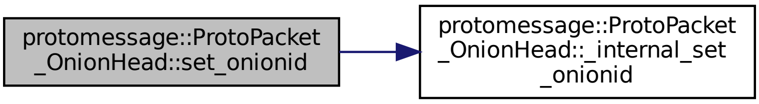
```
            int32_t value ) [inline]
```

Definition at line 830 of file proto-packet.pb.h.

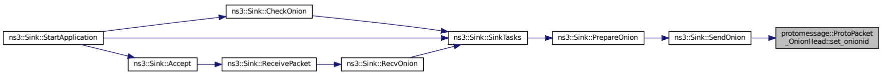
References _internal_set_onionid().

Referenced by ns3::Sink::SendOnion().

Here is the call graph for this function:


protomessage::ProtoPacket
_OnionHead::set_onionid
protomessage::ProtoPacket
_OnionHead::_internal_set
_onionid

Here is the caller graph for this function:

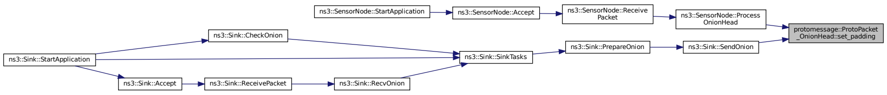


3.20.5.53 set_padding() [1/2] `template<typename ArgT0 = const std::string&, typename... ArgT>`

```
void protomessage::ProtoPacket_OnionHead::set_padding (
            ArgT0 && arg0,
            ArgT... args )
```

Referenced by ns3::SensorNode::ProcessOnionHead(), and ns3::Sink::SendOnion().

Here is the caller graph for this function:



3.20.5.54 set_padding() [2/2]

```
template<typename ArgT0 , typename... ArgT>
PROTOBUF_ALWAYS_INLINE void protomessage::ProtoPacket_OnionHead::set_padding (
            ArgT0 && arg0,
            ArgT... args ) [inline]
```

Definition at line 922 of file proto-packet.pb.h.

References _has_bits_, and padding_.

3.20.5.55 SetCachedSize()

```
void protomessage::ProtoPacket_OnionHead::SetCachedSize (
            int size ) const [final], [private]
```

Definition at line 247 of file proto-packet.pb.cc.

References _cached_size_.

3.20.5.56 SharedCtor()

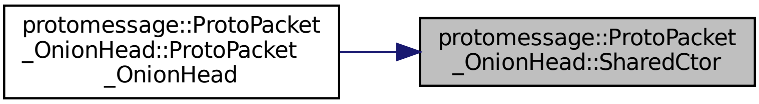
```
void protomessage::ProtoPacket_OnionHead::SharedCtor ( ) [inline],
[private]
```

Definition at line 216 of file proto-packet.pb.cc.

References onion_message_, onionid_, and padding_.

Referenced by ProtoPacket_OnionHead().

Here is the caller graph for this function:



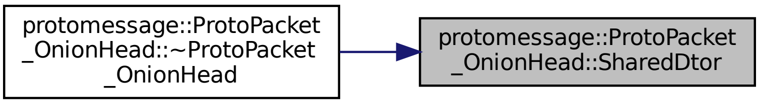
3.20.5.57 SharedDtor() `void protomessage::ProtoPacket_OnionHead::SharedDtor ( ) [inline], [private]`

Definition at line 235 of file proto-packet.pb.cc.

References onion_message_, and padding_.

Referenced by ∼ProtoPacket_OnionHead().

Here is the caller graph for this function:

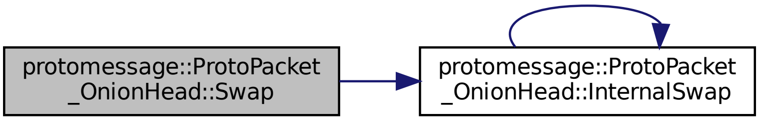


3.20.5.58 Swap() `void protomessage::ProtoPacket_OnionHead::Swap ( ProtoPacket_OnionHead * other ) [inline]`

Definition at line 133 of file proto-packet.pb.h.

References InternalSwap().

Here is the call graph for this function:

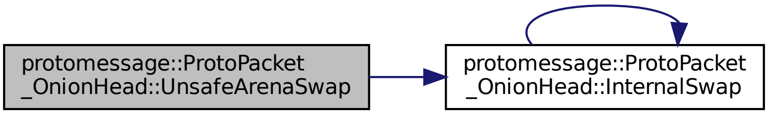


3.20.5.59 UnsafeArenaSwap() `void protomessage::ProtoPacket_OnionHead::UnsafeArenaSwap ( ProtoPacket_OnionHead * other ) [inline]`

Definition at line 146 of file proto-packet.pb.h.

References InternalSwap().

Here is the call graph for this function:


protomessage::ProtoPacket _OnionHead::UnsafeArenaSwap
protomessage::ProtoPacket _OnionHead::InternalSwap

3.20.6 Friends And Related Function Documentation

3.20.6.1 ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper `template<typename T > friend class ::PROTOBUF_NAMESPACE_ID::Arena::InternalHelper [friend]`

Definition at line 257 of file proto-packet.pb.h.

3.20.6.2 ::PROTOBUF_NAMESPACE_ID::internal::AnyMetadata `friend class ::PROTOBUF_NAMESPACE_↩ ID::internal::AnyMetadata [friend]`

Definition at line 180 of file proto-packet.pb.h.

3.20.6.3 ::TableStruct_proto_2dpacket_2eproto `friend struct ::TableStruct_proto_2dpacket_2eproto [friend]`

Definition at line 267 of file proto-packet.pb.h.

3.20.6.4 swap `void swap ( ProtoPacket_OnionHead & a, ProtoPacket_OnionHead & b ) [friend]`

Definition at line 130 of file proto-packet.pb.h.

Referenced by InternalSwap().

3.20.7 Member Data Documentation

3.20.7.1 _cached_size_

`mutable ::PROTOBUF_NAMESPACE_ID::internal::CachedSize protomessage::ProtoPacket_OnionHead::_cached_size_ [private]`

Definition at line 263 of file proto-packet.pb.h.

Referenced by ByteSizeLong(), GetCachedSize(), protomessage::ProtoPacket_OnionBody::GetCachedSize(), protomessage::ProtoPacket_Handshake::GetCachedSize(), protomessage::ProtoPacket::GetCachedSize(), and SetCachedSize().

3.20.7.2 _class_data_

`const ::PROTOBUF_NAMESPACE_ID::Message::ClassData protomessage::ProtoPacket_OnionHead::_class_data_ [static]`

Initial value:

```
= {
    ::PROTOBUF_NAMESPACE_ID::Message::CopyWithSizeCheck,
    ProtoPacket_OnionHead::MergeImpl
}
```

Definition at line 192 of file proto-packet.pb.h.

Referenced by GetClassData().

3.20.7.3 _has_bits_

`::PROTOBUF_NAMESPACE_ID::internal::HasBits<1> protomessage::ProtoPacket_OnionHead::_has_bits_ [private]`

Definition at line 262 of file proto-packet.pb.h.

Referenced by protomessage::ProtoPacket_OnionBody::_internal_has_aggregatedvalue(), protomessage::ProtoPacket::_internal_has_h_shake(), protomessage::ProtoPacket::_internal_has_o_body(), protomessage::ProtoPacket::_internal_has_o_head(), _internal_has_onion_message(), _internal_has_onionid(), _internal_has_padding(), protomessage::ProtoPacket_OnionBody::_internal_has_padding(), protomessage::ProtoPacket_Handshake::_internal_has_publickey(), protomessage::ProtoPacket::_internal_mutable_h_shake(), protomessage::ProtoPacket::_internal_mutable_o_body(), protomessage::ProtoPacket::_internal_mutable_o_head(), _internal_mutable_onion_message(), _internal_mutable_padding(), protomessage::ProtoPacket_OnionBody::_internal_mutable_padding(), protomessage::ProtoPacket_Handshake::_internal_mutable_publickey(), protomessage::ProtoPacket_OnionBody::_internal_set_aggregatedvalue(), _internal_set_onion_message(), _internal_set_onionid(), _internal_set_padding(), protomessage::ProtoPacket_OnionBody::_internal_set_padding(), protomessage::ProtoPacket_Handshake::_internal_set_publickey(), _InternalParse(), ByteSizeLong(), Clear(), protomessage::ProtoPacket_OnionBody::clear_aggregatedvalue(), protomessage::ProtoPacket::clear_h_shake(), protomessage::ProtoPacket::clear_o_body(), protomessage::ProtoPacket::clear_o_head(), clear_onion_message(), clear_onionid(), clear_padding(), protomessage::ProtoPacket_OnionBody::clear_padding(), protomessage::ProtoPacket_Handshake::clear_publickey(), InternalSwap(), MergeFrom(), protomessage::ProtoPacket::release_h_shake(), protomessage::ProtoPacket::release_o_body(), protomessage::ProtoPacket::release_o_head(), release_onion_message(), release_padding(), protomessage::ProtoPacket_OnionBody::release_padding(), protomessage::ProtoPacket_Handshake::release_publickey(), protomessage::ProtoPacket::set_allocated_h_shake(), protomessage::ProtoPacket::set_allocated_o_body(), protomessage::ProtoPacket::set_allocated_o_head(), set_allocated_onion_message(), set_allocated_padding(), protomessage::ProtoPacket_OnionBody::set_allocated_padding(), protomessage::ProtoPacket_Handshake::set_allocated_publickey(), set_onion_message(), set_padding(), protomessage::ProtoPacket_OnionBody::set_padding(), protomessage::ProtoPacket_Handshake::set_publickey(), protomessage::ProtoPacket::unsafe_arena_release_h_shake(), protomessage::ProtoPacket::unsafe_arena_release_o_body(), protomessage::ProtoPacket::unsafe_arena_release_o_head(), protomessage::ProtoPacket::unsafe_arena_set_allocated_h_shake(), protomessage::ProtoPacket::unsafe_arena_set_allocated_o_body(), and protomessage::ProtoPacket::unsafe_arena_set_allocated_o_head().

3.20.7.4 kIndexInFileMessages `constexpr int protomessage::ProtoPacket_OnionHead::kIndexInFileMessages [static], [constexpr]`

Initial value:

```
=
    0
```

Definition at line 127 of file proto-packet.pb.h.

3.20.7.5 onion_message_ `::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr protomessage::ProtoPacket_OnionHead::onion_message_ [private]`

Definition at line 264 of file proto-packet.pb.h.

Referenced by _internal_mutable_onion_message(), _internal_onion_message(), _internal_set_onion_message(), Clear(), clear_onion_message(), InternalSwap(), ProtoPacket_OnionHead(), release_onion_message(), set_allocated_onion_message(), set_onion_message(), SharedCtor(), and SharedDtor().

3.20.7.6 onionid_ `int32_t protomessage::ProtoPacket_OnionHead::onionid_ [private]`

Definition at line 266 of file proto-packet.pb.h.

Referenced by _internal_onionid(), _internal_set_onionid(), _InternalParse(), Clear(), clear_onionid(), InternalSwap(), MergeFrom(), ProtoPacket_OnionHead(), and SharedCtor().

3.20.7.7 padding_ `::PROTOBUF_NAMESPACE_ID::internal::ArenaStringPtr protomessage::ProtoPacket_OnionHead::padding_ [private]`

Definition at line 265 of file proto-packet.pb.h.

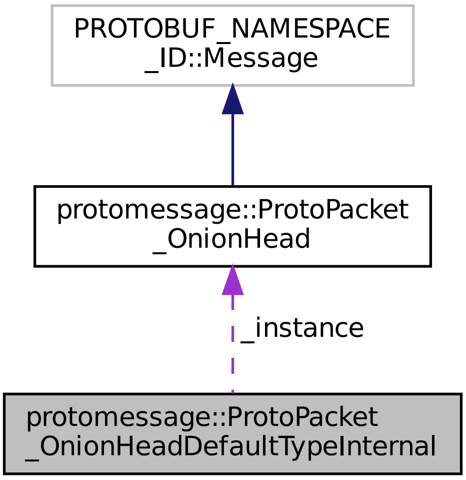
Referenced by _internal_mutable_padding(), protomessage::ProtoPacket_OnionBody::_internal_mutable_padding(), _internal_padding(), protomessage::ProtoPacket_OnionBody::_internal_padding(), _internal_set_padding(), protomessage::ProtoPacket_OnionBody::_internal_set_padding(), Clear(), clear_padding(), protomessage::ProtoPacket_OnionBody::clear_padding(), InternalSwap(), ProtoPacket_OnionHead(), release_padding(), protomessage::ProtoPacket_OnionBody::release_padding(), set_allocated_padding(), protomessage::ProtoPacket_OnionBody::set_allocated_padding(), set_padding(), protomessage::ProtoPacket_OnionBody::set_padding(), SharedCtor(), and SharedDtor().

The documentation for this class was generated from the following files:

- src/onion_routing_wsn/protobuf/proto-packet.pb.h
- src/onion_routing_wsn/protobuf/proto-packet.pb.cc

3.21 protomessage::ProtoPacket_OnionHeadDefaultTypeInternal Struct Reference

Collaboration diagram for protomessage::ProtoPacket_OnionHeadDefaultTypeInternal:


PROTOBUF_NAMESPACE
_ID::Message
protomessage::ProtoPacket
_OnionHead
_instance
protomessage::ProtoPacket
_OnionHeadDefaultTypeInternal

Public Member Functions

- constexpr ProtoPacket_OnionHeadDefaultTypeInternal ()
- ∼ProtoPacket_OnionHeadDefaultTypeInternal ()

Public Attributes

- union {
  ProtoPacket_OnionHead _instance
  };

3.21.1 Detailed Description

Definition at line 25 of file proto-packet.pb.cc.

3.21.2 Constructor & Destructor Documentation

3.21.2.1 ProtoPacket_OnionHeadDefaultTypeInternal() `constexpr protomessage::ProtoPacket_Onion←HeadDefaultTypeInternal::ProtoPacket_OnionHeadDefaultTypeInternal ( ) [inline], [constexpr]`

Definition at line 26 of file proto-packet.pb.cc.

3.21.2.2 ∼ProtoPacket_OnionHeadDefaultTypeInternal() `protomessage::ProtoPacket_OnionHead←↩ DefaultTypeInternal::∼ProtoPacket_OnionHeadDefaultTypeInternal ( ) [inline]`

Definition at line 28 of file proto-packet.pb.cc.

3.21.3 Member Data Documentation

3.21.3.1 "@1 `union { ... }`

3.21.3.2 _instance `ProtoPacket_OnionHead protomessage::ProtoPacket_OnionHeadDefaultType←↩ Internal::_instance`

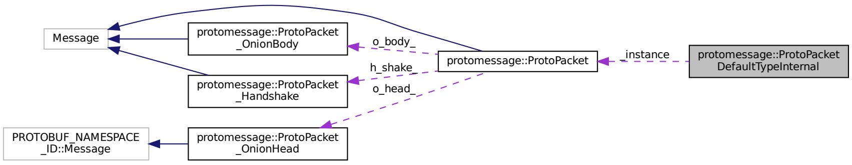
Definition at line 30 of file proto-packet.pb.cc.

The documentation for this struct was generated from the following file:

- src/onion_routing_wsn/protobuf/proto-packet.pb.cc

3.22 protomessage::ProtoPacketDefaultTypeInternal Struct Reference

Collaboration diagram for protomessage::ProtoPacketDefaultTypeInternal:


Message
protomessage::ProtoPacket _OnionBody
protomessage::ProtoPacket _Handshake
PROTOBUF_NAMESPACE _ID::Message
protomessage::ProtoPacket _OnionHead
o_body_
h_shake_
o_head_
protomessage::ProtoPacket
_instance
protomessage::ProtoPacket DefaultTypeInternal

Public Member Functions

- constexpr ProtoPacketDefaultTypeInternal ()
- ∼ProtoPacketDefaultTypeInternal ()

Public Attributes

- union {
  ProtoPacket _instance
  };

3.22.1 Detailed Description

Definition at line 64 of file proto-packet.pb.cc.

3.22.2 Constructor & Destructor Documentation

3.22.2.1 ProtoPacketDefaultTypeInternal() `constexpr protomessage::ProtoPacketDefaultTypeInternal↩ ::ProtoPacketDefaultTypeInternal ( ) [inline], [constexpr]`

Definition at line 65 of file proto-packet.pb.cc.

3.22.2.2 ∼ProtoPacketDefaultTypeInternal() `protomessage::ProtoPacketDefaultTypeInternal::∼↩ ProtoPacketDefaultTypeInternal ( ) [inline]`

Definition at line 67 of file proto-packet.pb.cc.

3.22.3 Member Data Documentation

3.22.3.1 "@7 `union { ... }`

3.22.3.2 _instance `ProtoPacket protomessage::ProtoPacketDefaultTypeInternal::_instance`

Definition at line 69 of file proto-packet.pb.cc.

The documentation for this struct was generated from the following file:

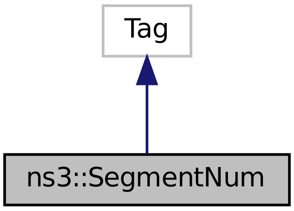
- src/onion_routing_wsn/protobuf/proto-packet.pb.cc

3.23 ns3::SegmentNum Class Reference

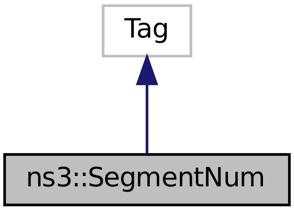
Class for adding a tag to packets used to track different segments of packets packets are fragmented due to small MSS Just implementing methods from ns3 class ns3::Tag.

```
#include "segmentnum.h"
```

Inheritance diagram for ns3::SegmentNum:



Collaboration diagram for ns3::SegmentNum:



Public Member Functions

- SegmentNum ()
- SegmentNum (uint32_t seg_num)
- ∼SegmentNum ()
- virtual void Deserialize (TagBuffer i)
- virtual TypeId GetInstanceTypeId (void) const
- uint32_t GetSegNum ()
- virtual uint32_t GetSerializedSize (void) const
- virtual void Print (std::ostream &os) const
- virtual void Serialize (TagBuffer i) const
- void SetSegNum (uint32_t seg_num)

Static Public Member Functions

- static TypeId GetTypeId (void)

Public Attributes

- uint32_t s_num

3.23.1 Detailed Description

Class for adding a tag to packets used to track different segments of packets packets are fragmented due to small MSS Just implementing methods from ns3 class ns3::Tag.

Definition at line 41 of file segmentnum.h.

3.23.2 Constructor & Destructor Documentation

3.23.2.1 SegmentNum() [1/2] `ns3::SegmentNum::SegmentNum ( )`

Definition at line 48 of file segmentnum.cc.

References s_num.

3.23.2.2 SegmentNum() [2/2] `ns3::SegmentNum::SegmentNum (`

```
            uint32_t seg_num )
```

Definition at line 53 of file segmentnum.cc.

References s_num.

3.23.2.3 ∼SegmentNum() `ns3::SegmentNum::∼SegmentNum ( )`

Definition at line 58 of file segmentnum.cc.

3.23.3 Member Function Documentation

3.23.3.1 Deserialize() `void ns3::SegmentNum::Deserialize ( TagBuffer i ) [virtual]`

Definition at line 82 of file segmentnum.cc.

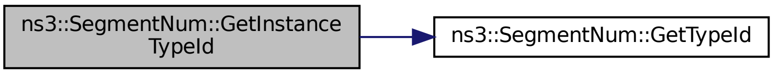
References s_num.

3.23.3.2 GetInstanceTypeId() `TypeId ns3::SegmentNum::GetInstanceTypeId ( void ) const [virtual]`

Definition at line 42 of file segmentnum.cc.

References GetTypeId().

Here is the call graph for this function:


ns3::SegmentNum::GetInstance TypeId
ns3::SegmentNum::GetTypeId

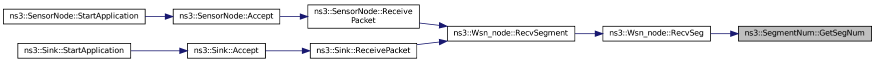
3.23.3.3 GetSegNum() `uint32_t ns3::SegmentNum::GetSegNum ( )`

Definition at line 60 of file segmentnum.cc.

References s_num.

Referenced by ns3::Wsn_node::RecvSeg().

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::Sink::StartApplication
ns3::Sink::Accept
ns3::Sink::ReceivePacket
ns3::Wsn_node::RecvSegment
ns3::Wsn_node::RecvSeg
ns3::SegmentNum::GetSegNum

3.23.3.4 GetSerializedSize() `uint32_t ns3::SegmentNum::GetSerializedSize ( void ) const [virtual]`

Definition at line 72 of file segmentnum.cc.

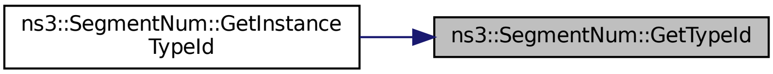
3.23.3.5 GetTypeId()

```
TypeId ns3::SegmentNum::GetTypeId (
    void ) [static]
```

Definition at line 31 of file segmentnum.cc.

Referenced by GetInstanceTypeId().

Here is the caller graph for this function:


ns3::SegmentNum::GetInstance TypeId
ns3::SegmentNum::GetTypeId

3.23.3.6 Print()

```
void ns3::SegmentNum::Print (
    std::ostream & os ) const [virtual]
```

Definition at line 87 of file segmentnum.cc.

References s_num.

3.23.3.7 Serialize()

```
void ns3::SegmentNum::Serialize (
    TagBuffer i ) const [virtual]
```

Definition at line 77 of file segmentnum.cc.

References s_num.

3.23.3.8 SetSegNum()

```
void ns3::SegmentNum::SetSegNum (
    uint32_t seg_num )
```

Definition at line 65 of file segmentnum.cc.

References s_num.

3.23.4 Member Data Documentation

3.23.4.1 s_num `uint32_t ns3::SegmentNum::s_num`

Definition at line 61 of file segmentnum.h.

Referenced by Deserialize(), GetSegNum(), Print(), SegmentNum(), Serialize(), and SetSegNum().

The documentation for this class was generated from the following files:

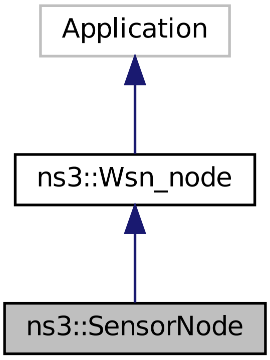
- src/onion_routing_wsn/protocol/segmentnum.h
- src/onion_routing_wsn/protocol/segmentnum.cc

3.24 ns3::SensorNode Class Reference

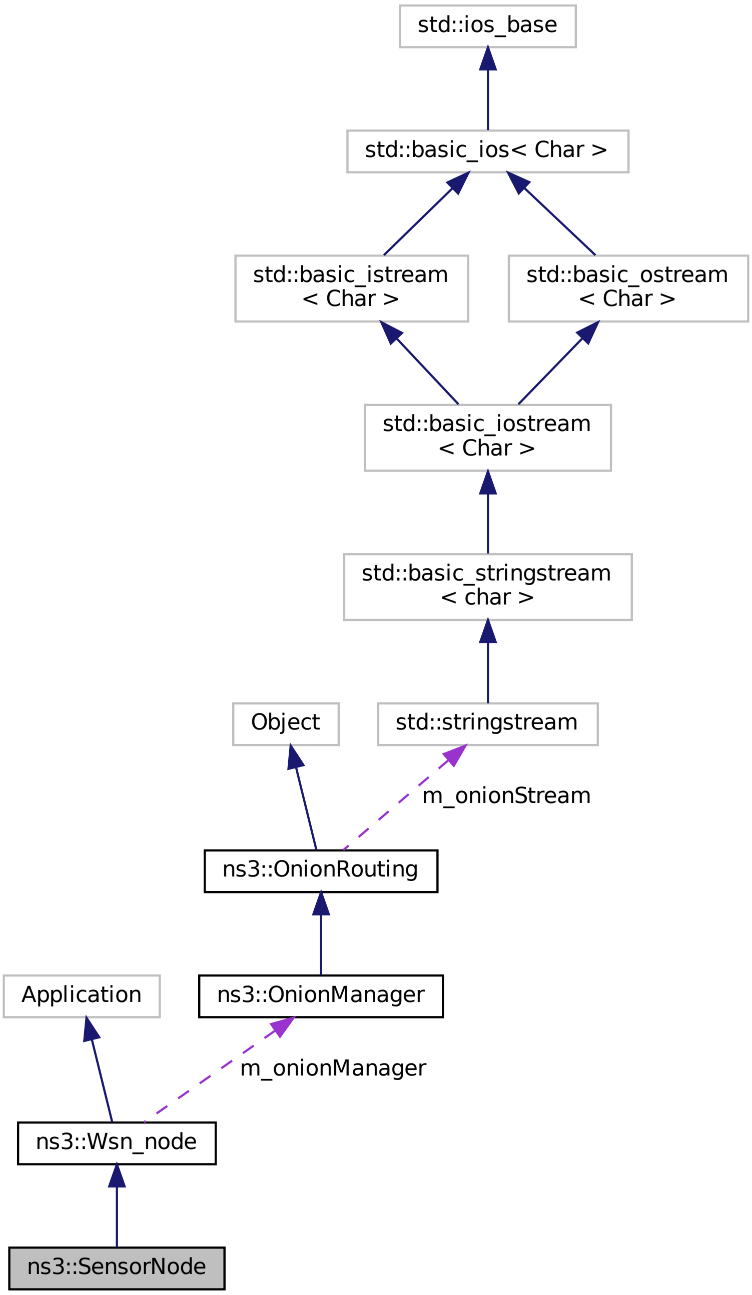
The application of the sensor node.

```
#include "sensornode.h"
```

Inheritance diagram for ns3::SensorNode:



Collaboration diagram for ns3::SensorNode:



Public Member Functions

- SensorNode ()

  Default constructor.
- virtual ∼SensorNode ()

  Default destructor.
- void Accept (Ptr< Socket > socket, const ns3::Address &from)

Accept new TCP connections.

- uint32_t DeserializeIpv4ToInt (uint8_t ∗buff)

  Convert an IPV4 address given as a buffer.
- void ProcessOnionBody (protomessage::ProtoPacket_OnionBody ∗onionbody)

  If the onion body contains the aggregated value, then aggregate the sensor (dummy) value to the value carried in the onion body.
- uint32_t ProcessOnionHead (protomessage::ProtoPacket_OnionHead ∗onionHead)

  Decrypt the outer layer of the onion head, obtain the information of the next IP address, reasemble the onion head by including padding if required.
- void ReceivePacket (Ptr< Socket > socket)

  Executed when a new onion is received.

Static Public Member Functions

- static TypeId GetTypeId (void)

  Register this type.

Private Member Functions

- void Handshake (void)

  Construct a new protobuf object containing the node publickey and send it to the sink node.
- virtual void StartApplication (void)

  1.Start the application run ns3::Wsn_node::Configure() 2.Generate new encryption keys 3.Schedule the execution of ns3::SensorNode::Handshake() after `delay` milliseconds, the delay is computed based on the node ip address 4.Set callback at new socket connection
- virtual void StopApplication (void)

  Stop the application.

Private Attributes

- uint32_t m_sensorValue = 20

  dummy reading of a sensor equipped on the node
- Ipv4Address m_sinkAddress

  address of the sink node

Additional Inherited Members

3.24.1 Detailed Description

The application of the sensor node.

Definition at line 46 of file sensornode.h.

3.24.2 Constructor & Destructor Documentation

3.24.2.1 SensorNode() `ns3::SensorNode::SensorNode ( )`

Default constructor.

Definition at line 47 of file sensornode.cc.

3.24.2.2 ∼SensorNode() `ns3::SensorNode::∼SensorNode ( ) [virtual]`

Default destructor.

Definition at line 51 of file sensornode.cc.

References ns3::Wsn_node::m_socket.

3.24.3 Member Function Documentation

3.24.3.1 Accept()

```
void ns3::SensorNode::Accept (
        Ptr< Socket > socket,
        const ns3::Address & from )
```

Accept new TCP connections.

Parameters

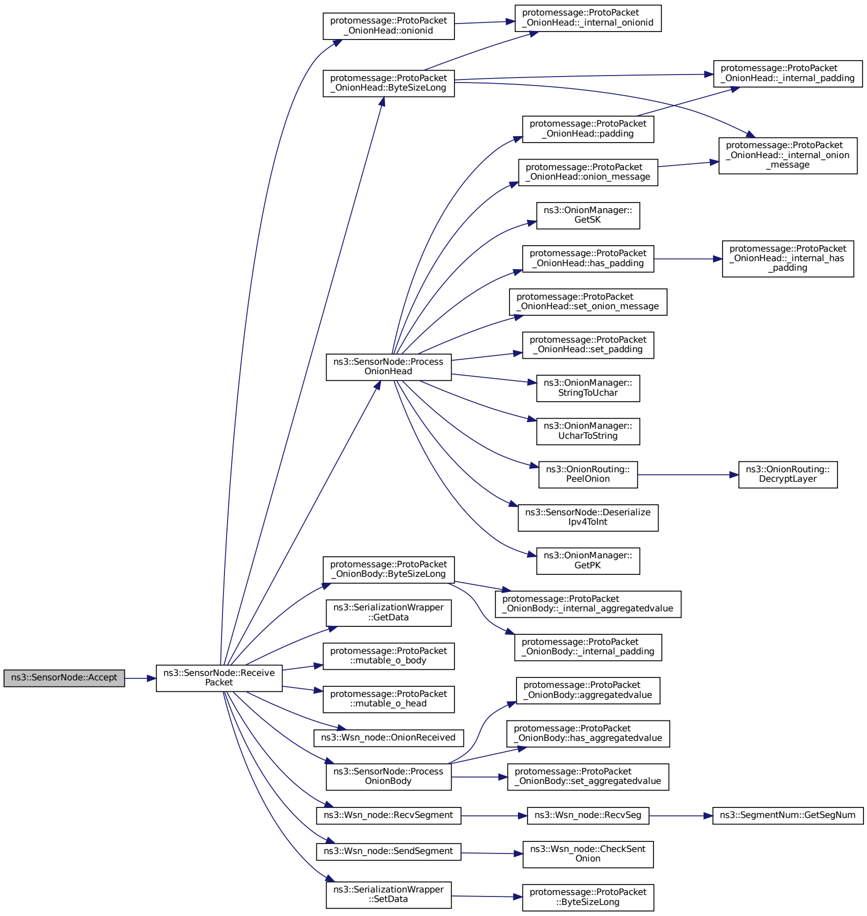
in	*socket*	
in	*from*	sending address

Definition at line 197 of file sensornode.cc.

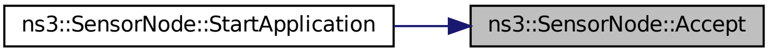
References ReceivePacket().

Referenced by StartApplication().

Here is the call graph for this function:


protomessage::ProtoPacket _OnionHead::onionid
protomessage::ProtoPacket _OnionHead::_internal_onionid
protomessage::ProtoPacket _OnionHead::ByteSizeLong
protomessage::ProtoPacket _OnionHead::_internal_padding
protomessage::ProtoPacket _OnionHead::padding
protomessage::ProtoPacket _OnionHead::_internal_onion _message
protomessage::ProtoPacket _OnionHead::onion_message
ns3::OnionManager:: GetSK
protomessage::ProtoPacket _OnionHead::has_padding
protomessage::ProtoPacket _OnionHead::_internal_has _padding
protomessage::ProtoPacket _OnionHead::set_onion_message
protomessage::ProtoPacket _OnionHead::set_padding
ns3::SensorNode::Process OnionHead
ns3::OnionManager:: StringToUchar
ns3::OnionManager:: UcharToString
ns3::OnionRouting:: PeelOnion
ns3::OnionRouting:: DecryptLayer
ns3::SensorNode::Deserialize Ipv4ToInt
ns3::OnionManager:: GetPK
protomessage::ProtoPacket _OnionBody::ByteSizeLong
protomessage::ProtoPacket _OnionBody::_internal_aggregatedvalue
ns3::SerializationWrapper ::GetData
protomessage::ProtoPacket _OnionBody::_internal_padding
protomessage::ProtoPacket ::mutable_o_body
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
protomessage::ProtoPacket ::mutable_o_head
protomessage::ProtoPacket _OnionBody::aggregatedvalue
protomessage::ProtoPacket _OnionBody::has_aggregatedvalue
ns3::Wsn_node::OnionReceived
ns3::SensorNode::Process OnionBody
protomessage::ProtoPacket _OnionBody::set_aggregatedvalue
ns3::Wsn_node::RecvSegment
ns3::Wsn_node::RecvSeg
ns3::SegmentNum::GetSegNum
ns3::Wsn_node::SendSegment
ns3::Wsn_node::CheckSent Onion
ns3::SerializationWrapper ::SetData
protomessage::ProtoPacket ::ByteSizeLong

Here is the caller graph for this function:



3.24.3.2 DeserializeIpv4ToInt() `uint32_t ns3::SensorNode::DeserializeIpv4ToInt ( uint8_t * buff )`

Convert an IPV4 address given as a buffer.

Parameters

in	*buff*	pointer to an array containing an IpV4 address

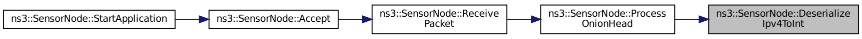
Returns

an IpV4 address as an unisgned integer of 32b

Definition at line 182 of file sensornode.cc.

Referenced by ProcessOnionHead().

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::SensorNode::Process OnionHead
ns3::SensorNode::Deserialize Ipv4ToInt

3.24.3.3 GetTypeId() `TypeId ns3::SensorNode::GetTypeId ( void ) [static]`

Register this type.

Returns

The object TypeId.

Definition at line 34 of file sensornode.cc.

References m_sinkAddress.

3.24.3.4 Handshake() `void ns3::SensorNode::Handshake ( void ) [private]`

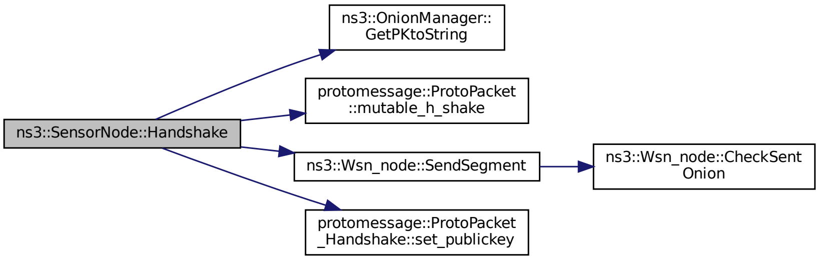
Construct a new protobuf object containing the node publickey and send it to the sink node.

Definition at line 59 of file sensornode.cc.

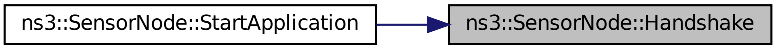
References ns3::OnionManager::GetPKtoString(), ns3::Wsn_node::m_onionManager, ns3::Wsn_node::m_←port, m_sinkAddress, protomessage::ProtoPacket::mutable_h_shake(), ns3::Wsn_node::SendSegment(), and protomessage::ProtoPacket_Handshake::set_publickey().

Referenced by StartApplication().

Here is the call graph for this function:


ns3::OnionManager:: GetPKtoString
protomessage::ProtoPacket ::mutable_h_shake
ns3::SensorNode::Handshake
ns3::Wsn_node::SendSegment
ns3::Wsn_node::CheckSent Onion
protomessage::ProtoPacket _Handshake::set_publickey

Here is the caller graph for this function:



3.24.3.5 ProcessOnionBody() `void ns3::SensorNode::ProcessOnionBody ( protomessage::ProtoPacket_OnionBody * onionbody )`

If the onion body contains the aggregated value, then aggregate the sensor (dummy) value to the value carried in the onion body.

Parameters

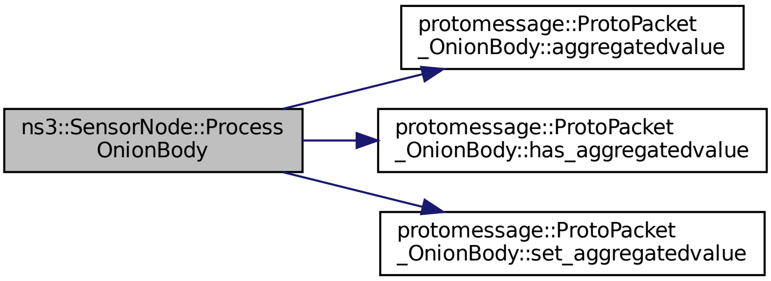
in	*onionBody*	pionter to the protobuf object holding informations of the onion body

Definition at line 168 of file sensornode.cc.

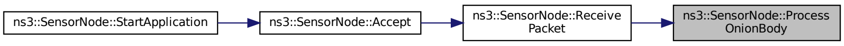
References protomessage::ProtoPacket_OnionBody::aggregatedvalue(), protomessage::ProtoPacket_OnionBody::has_aggregatedvalue(), m_sensorValue, and protomessage::ProtoPacket_OnionBody::set_aggregatedvalue().

Referenced by ReceivePacket().

Here is the call graph for this function:


ns3::SensorNode::Process OnionBody
protomessage::ProtoPacket _OnionBody::aggregatedvalue
protomessage::ProtoPacket _OnionBody::has_aggregatedvalue
protomessage::ProtoPacket _OnionBody::set_aggregatedvalue

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::SensorNode::Process OnionBody

3.24.3.6 ProcessOnionHead() `uint32_t ns3::SensorNode::ProcessOnionHead ( protomessage::ProtoPacket_OnionHead * onionHead )`

Decrypt the outer layer of the onion head, obtain the information of the next IP address, reasemble the onion head by including padding if required.

Parameters

in	*onionHead*	pionter to the protobuf object holding informations of the onion head

Returns

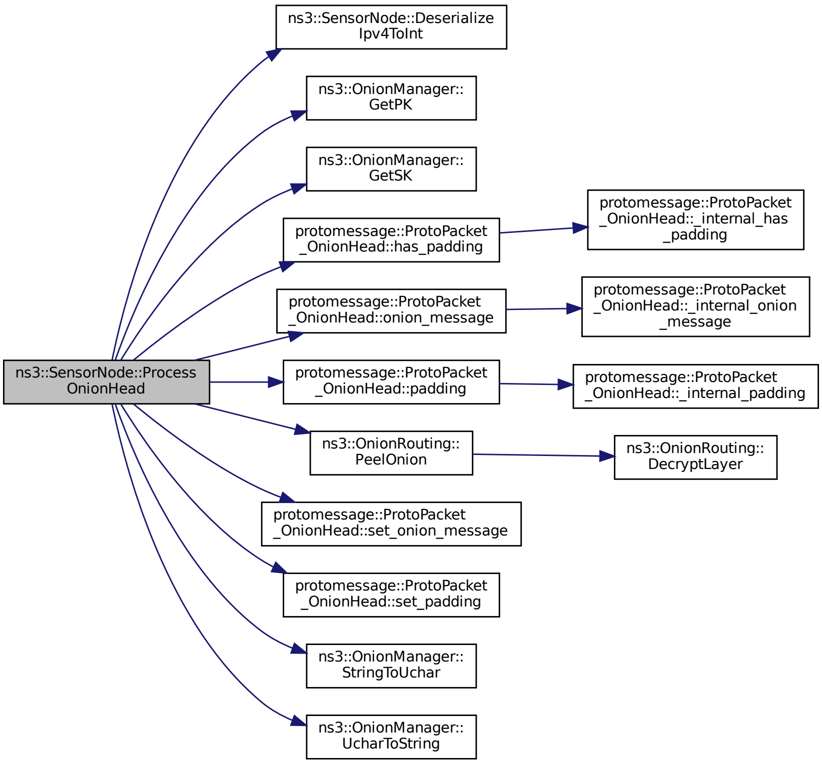
an IpV4 address as an unisgned integer of 32b

Definition at line 131 of file sensornode.cc.

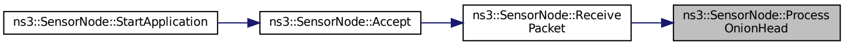
References DeserializeIpv4ToInt(), ns3::OnionManager::GetPK(), ns3::OnionManager::GetSK(), protomessage::ProtoPacket_OnionHead::has_padding(), ns3::orLayer::innerLayer, ns3::orLayer::innerLayerLen, ns3::Wsn_node::m_onionManager, ns3::orLayer::nextHopIP, protomessage::ProtoPacket_OnionHead::onion_message(), protomessage::ProtoPacket_OnionHead::padding(), ns3::OnionRouting::PeelOnion(), protomessage::ProtoPacket_OnionHead::set_onion_message(), protomessage::ProtoPacket_OnionHead::set_padding(), ns3::OnionManager::StringToUchar(), and ns3::OnionManager::UcharToString().

Referenced by ReceivePacket().

Here is the call graph for this function:


ns3::SensorNode::Deserialize Ipv4ToInt
ns3::OnionManager:: GetPK
ns3::OnionManager:: GetSK
protomessage::ProtoPacket _OnionHead::has_padding
protomessage::ProtoPacket _OnionHead::_internal_has _padding
protomessage::ProtoPacket _OnionHead::onion_message
protomessage::ProtoPacket _OnionHead::_internal_onion _message
ns3::SensorNode::Process OnionHead
protomessage::ProtoPacket _OnionHead::padding
protomessage::ProtoPacket _OnionHead::_internal_padding
ns3::OnionRouting:: PeelOnion
ns3::OnionRouting:: DecryptLayer
protomessage::ProtoPacket _OnionHead::set_onion_message
protomessage::ProtoPacket _OnionHead::set_padding
ns3::OnionManager:: StringToUchar
ns3::OnionManager:: UcharToString

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::SensorNode::Process OnionHead

3.24.3.7 ReceivePacket() `void ns3::SensorNode::ReceivePacket ( Ptr< Socket > socket )`

Executed when a new onion is received.

Repeatedly call ns3::Wsn_node::RecvSegment() until the whole packet is received. Then, check if the onion is valid by comparing the onionID of the onion and the onion sequence number in the ns3::OnionValidator::GetOnionSeq(). If the onion is not valid then delete the onion. Otherwise: Signal that the onion was received. Process the onion head and retrieve the next hop ip value. (ns3::SensorNode::ProcessOnionHead()) Processs the onion body. (ns3::SensorNode::ProcessOnionBody()) Send the onion meesage to the next hop ip.

Parameters

in	*socket*	the incoming connection

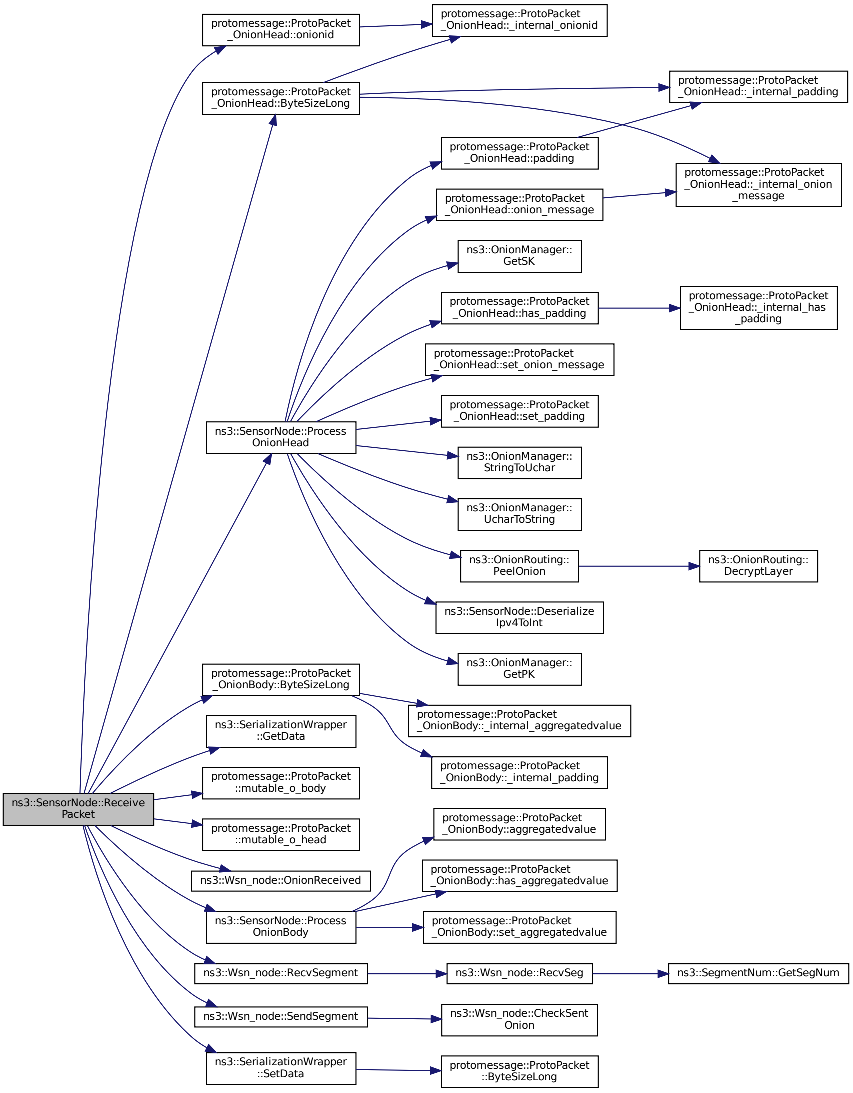
Log details about the onion

Definition at line 77 of file sensornode.cc.

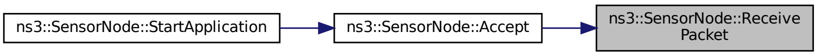
References protomessage::ProtoPacket_OnionHead::ByteSizeLong(), protomessage::ProtoPacket_OnionBody::ByteSizeLong(), ns3::SerializationWrapper::GetData(), ns3::Wsn_node::m_address, ns3::Wsn_node::m_onionValidator, ns3::Wsn_node::m_outputManager, ns3::Wsn_node::m_port, protomessage::ProtoPacket::mutable_o_body(), protomessage::ProtoPacket::mutable_o_head(), ns3::Wsn_node::o_sequenceNum, protomessage::ProtoPacket_OnionHead::onionid(), ns3::Wsn_node::OnionReceived(), ProcessOnionBody(), ProcessOnionHead(), ns3::Wsn_node::RecvSegment(), ns3::Wsn_node::SendSegment(), and ns3::SerializationWrapper::SetData().

Referenced by Accept().

Here is the call graph for this function:


ns3::SensorNode::Receive Packet
protomessage::ProtoPacket _OnionHead::onionid
protomessage::ProtoPacket _OnionHead::_internal_onionid
protomessage::ProtoPacket _OnionHead::ByteSizeLong
protomessage::ProtoPacket _OnionHead::_internal_padding
protomessage::ProtoPacket _OnionHead::padding
protomessage::ProtoPacket _OnionHead::_internal_onion _message
protomessage::ProtoPacket _OnionHead::onion_message
ns3::OnionManager:: GetSK
protomessage::ProtoPacket _OnionHead::has_padding
protomessage::ProtoPacket _OnionHead::_internal_has _padding
protomessage::ProtoPacket _OnionHead::set_onion_message
protomessage::ProtoPacket _OnionHead::set_padding
ns3::SensorNode::Process OnionHead
ns3::OnionManager:: StringToUchar
ns3::OnionManager:: UcharToString
ns3::OnionRouting:: PeelOnion
ns3::OnionRouting:: DecryptLayer
ns3::SensorNode::Deserialize Ipv4ToInt
ns3::OnionManager:: GetPK
protomessage::ProtoPacket _OnionBody::ByteSizeLong
protomessage::ProtoPacket _OnionBody::_internal_aggregatedvalue
ns3::SerializationWrapper ::GetData
protomessage::ProtoPacket _OnionBody::_internal_padding
protomessage::ProtoPacket ::mutable_o_body
protomessage::ProtoPacket _OnionBody::aggregatedvalue
protomessage::ProtoPacket ::mutable_o_head
protomessage::ProtoPacket _OnionBody::has_aggregatedvalue
ns3::Wsn_node::OnionReceived
ns3::SensorNode::Process OnionBody
protomessage::ProtoPacket _OnionBody::set_aggregatedvalue
ns3::Wsn_node::RecvSegment
ns3::Wsn_node::RecvSeg
ns3::SegmentNum::GetSegNum
ns3::Wsn_node::SendSegment
ns3::Wsn_node::CheckSent Onion
ns3::SerializationWrapper ::SetData
protomessage::ProtoPacket ::ByteSizeLong

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet

3.24.3.8 StartApplication() `void ns3::SensorNode::StartApplication ( void ) [private], [virtual]`

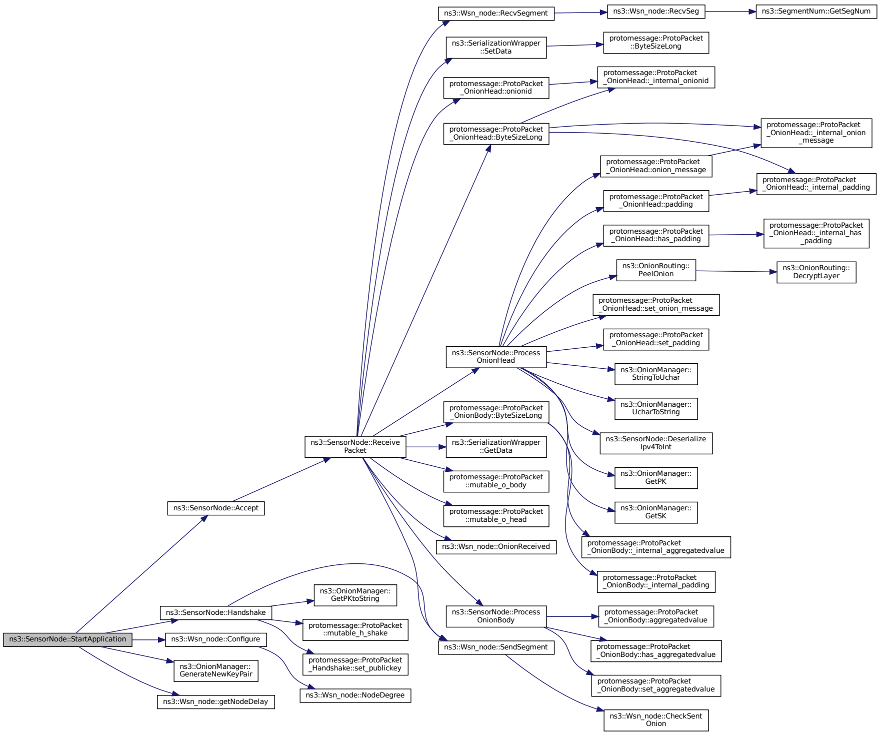
1.Start the application run ns3::Wsn_node::Configure() 2.Generate new encryption keys 3.Schedule the execution of ns3::SensorNode::Handshake() after `delay` milliseconds, the delay is computed based on the node ip address 4.Set callback at new socket connection

Reimplemented from ns3::Wsn_node.

Definition at line 204 of file sensornode.cc.

References Accept(), ns3::Wsn_node::Configure(), ns3::OnionManager::GenerateNewKeyPair(), ns3::Wsn_node↩ ::getNodeDelay(), Handshake(), ns3::Wsn_node::m_address, ns3::Wsn_node::m_onionManager, and ns3::Wsn↩ _node::m_socket.

Here is the call graph for this function:


ns3::Wsn_node::RecvSegment
ns3::Wsn_node::RecvSeg
ns3::SegmentNum::GetSegNum
ns3::SerializationWrapper ::SetData
protomessage::ProtoPacket ::ByteSizeLong
protomessage::ProtoPacket _OnionHead::onionid
protomessage::ProtoPacket _OnionHead::_internal_onionid
protomessage::ProtoPacket _OnionHead::ByteSizeLong
protomessage::ProtoPacket _OnionHead::_internal_onion _message
protomessage::ProtoPacket _OnionHead::onion_message
protomessage::ProtoPacket _OnionHead::_internal_padding
protomessage::ProtoPacket _OnionHead::padding
protomessage::ProtoPacket _OnionHead::has_padding
protomessage::ProtoPacket _OnionHead::_internal_has _padding
ns3::OnionRouting:: PeelOnion
ns3::OnionRouting:: DecryptLayer
protomessage::ProtoPacket _OnionHead::set_onion_message
protomessage::ProtoPacket _OnionHead::set_padding
ns3::SensorNode::Process OnionHead
ns3::OnionManager:: StringToUchar
ns3::OnionManager:: UcharToString
protomessage::ProtoPacket _OnionBody::ByteSizeLong
ns3::SensorNode::Deserialize Ipv4ToInt
ns3::SensorNode::Receive Packet
ns3::SerializationWrapper ::GetData
ns3::OnionManager:: GetPK
protomessage::ProtoPacket ::mutable_o_body
ns3::OnionManager:: GetSK
protomessage::ProtoPacket ::mutable_o_head
ns3::SensorNode::Accept
ns3::Wsn_node::OnionReceived
protomessage::ProtoPacket _OnionBody::_internal_aggregatedvalue
protomessage::ProtoPacket _OnionBody::_internal_padding
ns3::OnionManager:: GetPKtoString
ns3::SensorNode::Handshake
ns3::SensorNode::Process OnionBody
protomessage::ProtoPacket _OnionBody::aggregatedvalue
protomessage::ProtoPacket ::mutable_h_shake
ns3::SensorNode::StartApplication
ns3::Wsn_node::Configure
ns3::Wsn_node::SendSegment
protomessage::ProtoPacket _OnionBody::has_aggregatedvalue
ns3::OnionManager:: GenerateNewKeyPair
protomessage::ProtoPacket _Handshake::set_publickey
protomessage::ProtoPacket _OnionBody::set_aggregatedvalue
ns3::Wsn_node::NodeDegree
ns3::Wsn_node::getNodeDelay
ns3::Wsn_node::CheckSent Onion

3.24.3.9 StopApplication() `void ns3::SensorNode::StopApplication ( void ) [private], [virtual]`

Stop the application.

Reimplemented from ns3::Wsn_node.

Definition at line 222 of file sensornode.cc.

References ns3::Wsn_node::m_socket.

3.24.4 Member Data Documentation

3.24.4.1 m_sensorValue `uint32_t ns3::SensorNode::m_sensorValue = 20 [private]`

dummy reading of a sensor equipped on the node

Definition at line 162 of file sensornode.h.

Referenced by ProcessOnionBody().

3.24.4.2 m_sinkAddress `Ipv4Address ns3::SensorNode::m_sinkAddress [private]`

address of the sink node

Definition at line 160 of file sensornode.h.

Referenced by GetTypeId(), and Handshake().

The documentation for this class was generated from the following files:

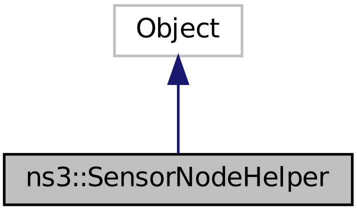
- src/onion_routing_wsn/model/sensornode.h
- src/onion_routing_wsn/model/sensornode.cc

3.25 ns3::SensorNodeHelper Class Reference

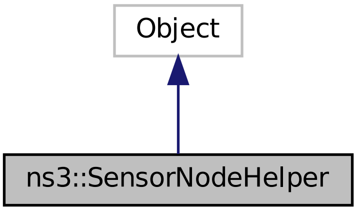
Helper class for the creation of SensorNode applications.

```
#include "sensornode-helper.h"
```

Inheritance diagram for ns3::SensorNodeHelper:



Collaboration diagram for ns3::SensorNodeHelper:



Public Member Functions

- SensorNodeHelper ()

  Default constructor.
- SensorNodeHelper (Ipv4Address address, Ptr< OutputManager > outputManager, Ptr< OnionValidator > onionValidator)

  Constructor with attributes to set basic attributes of sensor nodes.
- ∼SensorNodeHelper ()

  Default constructor.
- ApplicationContainer Install (NodeContainer c) const
- Ptr< Application > InstallPriv (Ptr< Node > node) const
- void SetAttribute (std::string name, const AttributeValue &value)

  Setter of individual attributes.

Static Public Member Functions

- static TypeId GetTypeId (void)

  Register this type.

Private Attributes

- ObjectFactory m_factory

  factory object
- uint16_t m_onionTimeout

  A watchdog timer set to abort onion messagess, if the timer elepses before the onion returns back to the sink node.
- Ptr< OnionValidator > m_onionValidator

  Manage onions and when to abort them.
- Ptr< OutputManager > m_outputManager

  Manage the output of the simulation.
- Ipv4Address m_sinkAddress

  Ipv4 address of the sink node.

3.25.1 Detailed Description

Helper class for the creation of SensorNode applications.

Definition at line 56 of file sensornode-helper.h.

3.25.2 Constructor & Destructor Documentation

3.25.2.1 SensorNodeHelper() [1/2] ns3::SensorNodeHelper::SensorNodeHelper ( )

Default constructor.

Definition at line 39 of file sensornode-helper.cc.

References m_factory.

3.25.2.2 SensorNodeHelper() [2/2] ns3::SensorNodeHelper::SensorNodeHelper (

```
Ipv4Address address,
Ptr< OutputManager > outputManager,
Ptr< OnionValidator > onionValidator )
```

Constructor with attributes to set basic attributes of sensor nodes.

Definition at line 44 of file sensornode-helper.cc.

References m_factory, m_onionValidator, and m_outputManager.

3.25.2.3 ∼SensorNodeHelper() ns3::SensorNodeHelper::∼SensorNodeHelper ( )

Default constructor.

Definition at line 54 of file sensornode-helper.cc.

3.25.3 Member Function Documentation

3.25.3.1 GetTypeId() TypeId ns3::SensorNodeHelper::GetTypeId (

```
void ) [static]
```

Register this type.

Returns

The object TypeId.

Definition at line 32 of file sensornode-helper.cc.

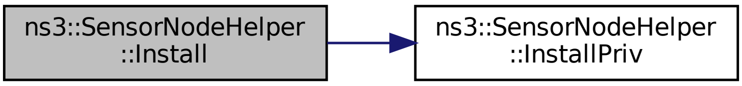
3.25.3.2 Install() `ApplicationContainer ns3::SensorNodeHelper::Install ( NodeContainer c ) const`

Definition at line 65 of file sensornode-helper.cc.

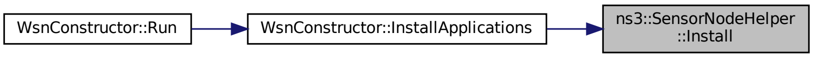
References InstallPriv().

Referenced by WsnConstructor::InstallApplications().

Here is the call graph for this function:



Here is the caller graph for this function:



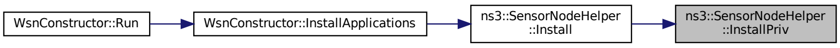
3.25.3.3 InstallPriv() `Ptr< Application > ns3::SensorNodeHelper::InstallPriv ( Ptr< Node > node ) const`

Definition at line 77 of file sensornode-helper.cc.

References m_factory.

Referenced by Install().

Here is the caller graph for this function:



3.25.3.4 SetAttribute() `void ns3::SensorNodeHelper::SetAttribute ( std::string name, const AttributeValue & value )`

Setter of individual attributes.

Definition at line 59 of file sensornode-helper.cc.

References m_factory.

3.25.4 Member Data Documentation

3.25.4.1 m_factory `ObjectFactory ns3::SensorNodeHelper::m_factory [private]`

factory object

Definition at line 116 of file sensornode-helper.h.

Referenced by InstallPriv(), SensorNodeHelper(), and SetAttribute().

3.25.4.2 m_onionTimeout `uint16_t ns3::SensorNodeHelper::m_onionTimeout [private]`

A watchdog timer set to abort onion messagess, if the timer elepses before the onion returns back to the sink node.

Definition at line 115 of file sensornode-helper.h.

3.25.4.3 m_onionValidator `Ptr<OnionValidator> ns3::SensorNodeHelper::m_onionValidator [private]`

Manage onions and when to abort them.

Definition at line 113 of file sensornode-helper.h.

Referenced by SensorNodeHelper().

3.25.4.4 m_outputManager `Ptr<OutputManager> ns3::SensorNodeHelper::m_outputManager [private]`

Manage the output of the simulation.

Definition at line 112 of file sensornode-helper.h.

Referenced by SensorNodeHelper().

3.25.4.5 m_sinkAddress `Ipv4Address ns3::SensorNodeHelper::m_sinkAddress [private]`

Ipv4 address of the sink node.

Definition at line 111 of file sensornode-helper.h.

The documentation for this class was generated from the following files:

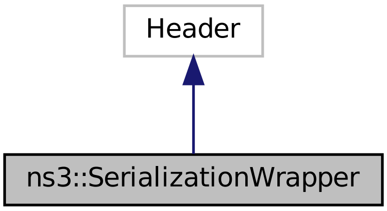
- src/onion_routing_wsn/helper/sensornode-helper.h
- src/onion_routing_wsn/helper/sensornode-helper.cc

3.26 ns3::SerializationWrapper Class Reference

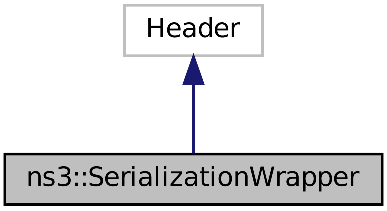
Class for the serialization-deserialization of the messagess to send in packets.

```
#include "serializationwrapper.h"
```

Inheritance diagram for ns3::SerializationWrapper:



Collaboration diagram for ns3::SerializationWrapper:



Public Member Functions

- SerializationWrapper ()

  Default constructor.
- SerializationWrapper (protomessage::ProtoPacket message)

  Constructor with argument.
- virtual ∼SerializationWrapper ()
- virtual uint32_t Deserialize (Buffer::Iterator start)

  deserialize the data
- void GetData (protomessage::ProtoPacket ∗message)

  Getter of the data in the protocol header.
- virtual TypeId GetInstanceTypeId (void) const
- virtual uint32_t GetSerializedSize (void) const

  compute the serialized size of the data
- virtual void Print (std::ostream &os) const

  dummy printing of the serialized data
- virtual void Serialize (Buffer::Iterator start) const

  serialize the data
- void SetData (protomessage::ProtoPacket message)

  Setter of the data in the protocol header.

Static Public Member Functions

- static TypeId GetTypeId (void)

  Register this type.

Public Attributes

- uint8_t ∗ m_data

  the serialized data
- int m_dataSize = 0

  holds the size of the serialized data in bytes

3.26.1 Detailed Description

Class for the serialization-deserialization of the messagess to send in packets.

Definition at line 54 of file serializationwrapper.h.

3.26.2 Constructor & Destructor Documentation

3.26.2.1 SerializationWrapper() [1/2] ns3::SerializationWrapper::SerializationWrapper ( )

Default constructor.

Definition at line 47 of file serializationwrapper.cc.

3.26.2.2 ∼SerializationWrapper() `ns3::SerializationWrapper::∼SerializationWrapper ( ) [virtual]`

Definition at line 51 of file serializationwrapper.cc.

References m_data.

3.26.2.3 SerializationWrapper() [2/2] `ns3::SerializationWrapper::SerializationWrapper ( protomessage::ProtoPacket message )`

Constructor with argument.

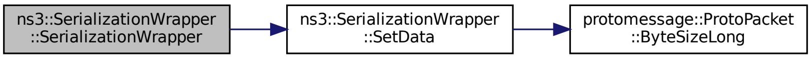
Parameters

in	*message*	the protobuff object containing the data to transmit

Definition at line 81 of file serializationwrapper.cc.

References SetData().

Here is the call graph for this function:


ns3::SerializationWrapper ::SerializationWrapper
ns3::SerializationWrapper ::SetData
protomessage::ProtoPacket ::ByteSizeLong

3.26.3 Member Function Documentation

3.26.3.1 Deserialize() `uint32_t ns3::SerializationWrapper::Deserialize ( Buffer::Iterator start ) [virtual]`

deserialize the data

Parameters

in,out	*start*	an iterator which points to where the data should read from.

Returns

the number of bytes read.

Definition at line 105 of file serializationwrapper.cc.

References m_data, and m_dataSize.

3.26.3.2 GetData() `void ns3::SerializationWrapper::GetData ( protomessage::ProtoPacket * message )`

Getter of the data in the protocol header.

Parameters

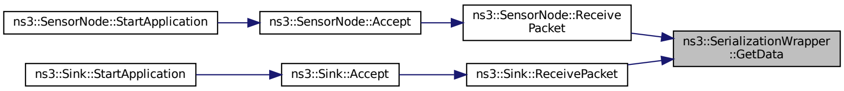
in,out	*message*	pointer to the protobuff object for storing data

Definition at line 75 of file serializationwrapper.cc.

References m_data, and m_dataSize.

Referenced by ns3::SensorNode::ReceivePacket(), and ns3::Sink::ReceivePacket().

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::SerializationWrapper ::GetData
ns3::Sink::StartApplication
ns3::Sink::Accept
ns3::Sink::ReceivePacket

3.26.3.3 GetInstanceTypeId() `TypeId ns3::SerializationWrapper::GetInstanceTypeId ( void ) const [virtual]`

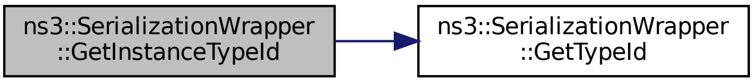
Returns

The object TypeId.

Definition at line 42 of file serializationwrapper.cc.

References GetTypeId().

Here is the call graph for this function:


ns3::SerializationWrapper ::GetInstanceTypeId
ns3::SerializationWrapper ::GetTypeId

3.26.3.4 GetSerializedSize() `uint32_t ns3::SerializationWrapper::GetSerializedSize ( void ) const [virtual]`

compute the serialized size of the data

Returns

return the size of the serialized data in bytes

Definition at line 87 of file serializationwrapper.cc.

References m_dataSize.

3.26.3.5 GetTypeId() `TypeId ns3::SerializationWrapper::GetTypeId ( void ) [static]`

Register this type.

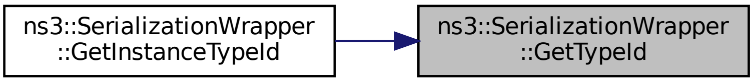
Returns

The object TypeId.

Definition at line 31 of file serializationwrapper.cc.

Referenced by GetInstanceTypeId().

Here is the caller graph for this function:


ns3::SerializationWrapper ::GetInstanceTypeId
ns3::SerializationWrapper ::GetTypeId

3.26.3.6 Print() `void ns3::SerializationWrapper::Print ( std::ostream & os ) const [virtual]`

dummy printing of the serialized data

Definition at line 123 of file serializationwrapper.cc.

3.26.3.7 Serialize() `void ns3::SerializationWrapper::Serialize ( Buffer::Iterator start ) const [virtual]`

serialize the data

Parameters

in,out	*start*	an iterator which points to where the data should be written.

Definition at line 98 of file serializationwrapper.cc.

References m_data, and m_dataSize.

3.26.3.8 SetData() `void ns3::SerializationWrapper::SetData ( protomessage::ProtoPacket message )`

Setter of the data in the protocol header.

Parameters

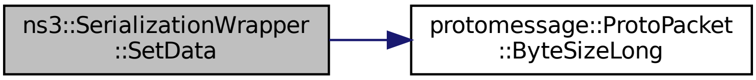
in	*message*	the protobuff object containing the data to transmit

Definition at line 58 of file serializationwrapper.cc.

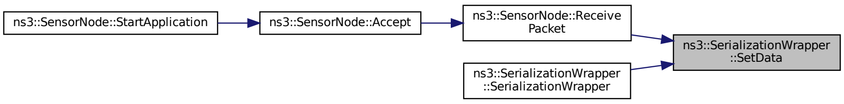
References protomessage::ProtoPacket::ByteSizeLong(), m_data, and m_dataSize.

Referenced by ns3::SensorNode::ReceivePacket(), and SerializationWrapper().

Here is the call graph for this function:



Here is the caller graph for this function:



3.26.4 Member Data Documentation

3.26.4.1 m_data `uint8_t* ns3::SerializationWrapper::m_data`

the serialized data

Definition at line 142 of file serializationwrapper.h.

Referenced by Deserialize(), GetData(), Serialize(), SetData(), and ∼SerializationWrapper().

3.26.4.2 m_dataSize `int ns3::SerializationWrapper::m_dataSize = 0`

holds the size of the serialized data in bytes

Definition at line 140 of file serializationwrapper.h.

Referenced by Deserialize(), GetData(), GetSerializedSize(), Serialize(), and SetData().

The documentation for this class was generated from the following files:

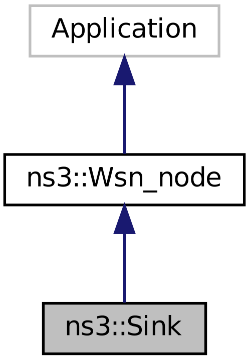
- src/onion_routing_wsn/protocol/serializationwrapper.h
- src/onion_routing_wsn/protocol/serializationwrapper.cc

3.27 ns3::Sink Class Reference

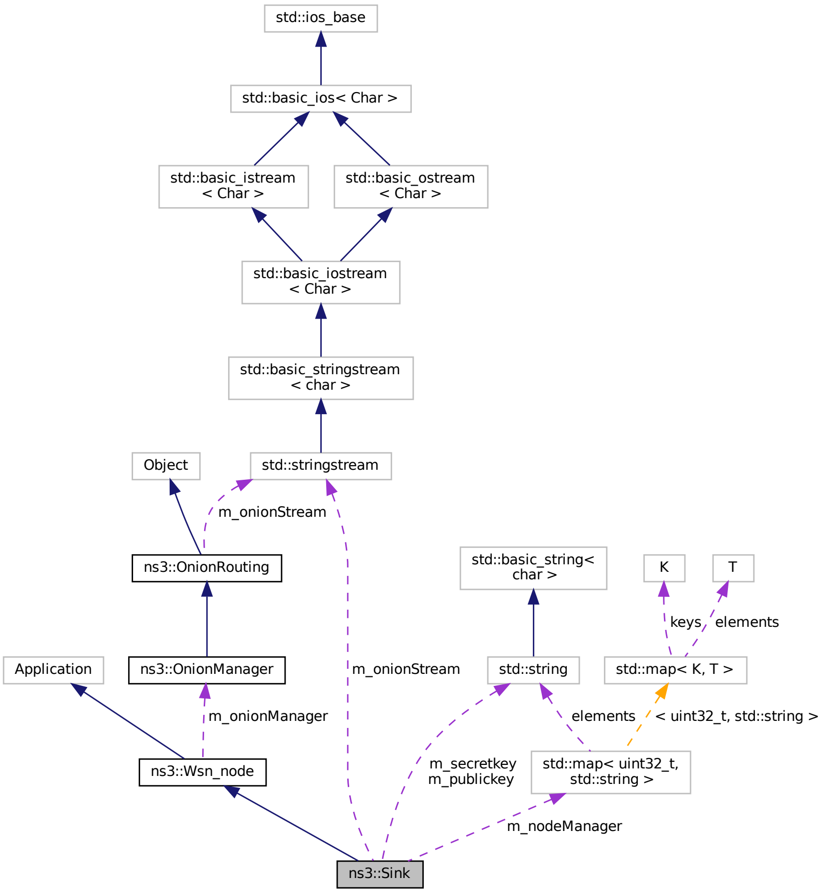
The application of the sink node. The node that generates onion messagess.

```
#include "sink.h"
```

Inheritance diagram for ns3::Sink:



Collaboration diagram for ns3::Sink:



Public Member Functions

- Sink ()

  Default constructor.
- virtual ∼Sink ()

  Default destructor.
- void Accept (Ptr< Socket > socket, const ns3::Address &from)

  Accept new TCP connections.
- void CheckOnion (void)

  Each five seconds call the ns3::OnionValidator and check if at least one onion message is alive Otherwise the onion was aborted, decrease the value of `m_repeateCount` and schedule a new execution of the function ns3::Sink::SinkTasks()
- void Setup (uint16_t ∗onionPathlengths, uint16_t numOnionLengths, int repeateTimes)

  Setup sink node parameters after the application installation.
- void SinkTasks ()

  Schedule the creation of a new onion based on the path length specified in `m_onionPathLengths` If all onions specified in `m_onionPathLengths` were executed for `m_repeateTimes` Then end the simulation.

Static Public Member Functions

- static TypeId GetTypeId (void)

  Register this type.

Private Member Functions

- void PrepareOnion (int ∗route, int routeLen)

  Method that constructs the onion head from `route` and `routeLen` parameters.
- void ReceivePacket (Ptr< Socket > socket)

  Recieve a new packet from `socket` Check if the packet is a handshake packet or a packet containing an onion message.
- void RecvHandshake (protomessage::ProtoPacket_Handshake ∗handshake_data, InetSocketAddress from)

  When receiving a new handshake with a node.
- void RecvOnion (protomessage::ProtoPacket_OnionBody ∗onion_body)

  Triggered when an onion message is received back at the sink node Data about the message is captured and a new onion message is scheduled.
- void SelectRoute (int ∗route, int routeLen)

  The method builds the path of the onion message by randomly selecting sensor nodes from the `m_nodeManager` structure.
- void SendOnion (uint32_t firstHop, int routeLen, unsigned char ∗cipher, int cipherLen)

  The method constructs the onion message as a protobuf object.
- virtual void StartApplication (void)

  1.Start the application run ns3::Wsn_node::Configure() 2.Generate new encryption keys 3.Set callback at new socket connection 4.Schedule the execution of ns3::Sink::SinkTasks() after `m_onionDelay` milliseconds 5.Schedule the execution of ns3::Sink::CheckOnion() after `m_onionDelay` milliseconds and 5 seconds
- virtual void StopApplication (void)

  Stop the application.

Private Attributes

- enum BodyOptions m_bodyOptions
- uint16_t m_bodySize
- uint32_t m_decoyNum

  dummy decoy value used to obfuscate the value carried in the onion body
- bool m_fixedOnionSize

  maintain the onion size fixed by adding padding (after layer decryption)
- std::map< uint32_t, std::string > m_nodeManager

  hashmap to manage data about nodes in the WSN// pair <IP,publickey>
- uint16_t m_numnodes

  The number of sensor nodes in the simulation.
- uint16_t m_numOnionLengths

  size of the array m_onionPathsLengths
- uint32_t m_onionDelay

  The sink will start sending onion messagess after OnionDelay seconds.
- int m_onionId = 1

  onion ID incremented each time a new onion is issued
- int m_onionLengthIndex = 0

  index of the current onion path length
- uint16_t ∗ m_onionPathLengths

  array holding onion path lengths
- std::stringstream m_onionStream

  a string stream holding the onion represented as a string
- std::string m_publickey

  the encryption key: publickey
- int m_repeateCount = 0

  how many times the onion was sent at the current path length
- int m_repeateTimes

  Integer specifying the number of times to generate the onion message for each value in the m_onionPathsLengths.
- std::string m_secretkey

  the encryption key: secretkey

Additional Inherited Members

3.27.1 Detailed Description

The application of the sink node. The node that generates onion messagess.

Definition at line 49 of file sink.h.

3.27.2 Constructor & Destructor Documentation

3.27.2.1 Sink() `ns3::Sink::Sink ( )`

Default constructor.

Definition at line 63 of file sink.cc.

3.27.2.2 ∼Sink() `ns3::Sink::∼Sink ( ) [virtual]`

Default destructor.

Definition at line 67 of file sink.cc.

References ns3::Wsn_node::m_socket.

3.27.3 Member Function Documentation

3.27.3.1 Accept()

```
void ns3::Sink::Accept (
    Ptr< Socket > socket,
    const ns3::Address & from )
```

Accept new TCP connections.

Parameters

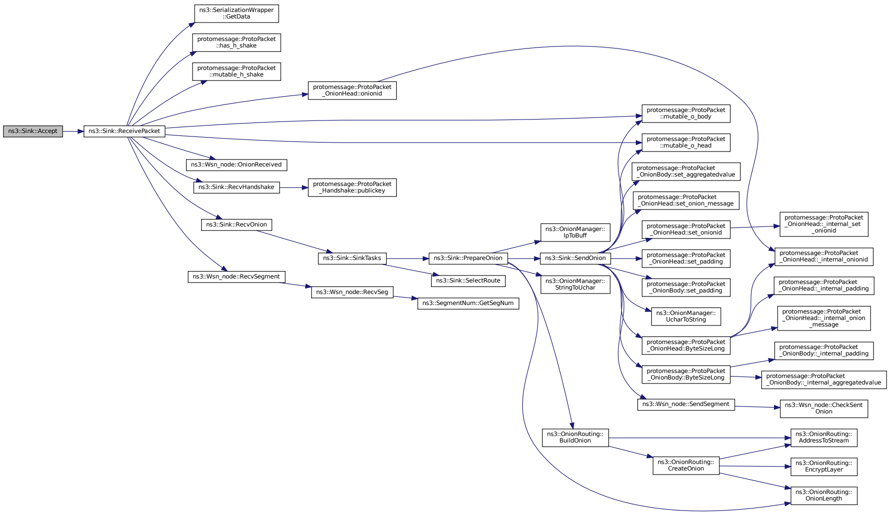
in	*socket*	
in	*from*	sending address

Definition at line 87 of file sink.cc.

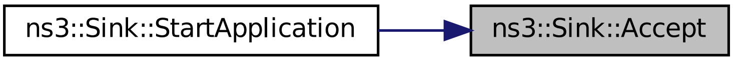
References ReceivePacket().

Referenced by StartApplication().

Here is the call graph for this function:


ns3::Sink::Accept
ns3::Sink::ReceivePacket
ns3::SerializationWrapper ::GetData
protomessage::ProtoPacket ::has_h_shake
protomessage::ProtoPacket ::mutable_h_shake
protomessage::ProtoPacket _OnionHead::onionid
ns3::Wsn_node::OnionReceived
ns3::Sink::RecvHandshake
protomessage::ProtoPacket _Handshake::publickey
ns3::Sink::RecvOnion
ns3::Sink::SinkTasks
ns3::Wsn_node::RecvSegment
ns3::Wsn_node::RecvSeg
ns3::SegmentNum::GetSegNum
ns3::Sink::PrepareOnion
ns3::Sink::SelectRoute
ns3::OnionManager:: IpToBuff
ns3::Sink::SendOnion
ns3::OnionManager:: StringToUchar
protomessage::ProtoPacket ::mutable_o_body
protomessage::ProtoPacket ::mutable_o_head
protomessage::ProtoPacket _OnionBody::set_aggregatedvalue
protomessage::ProtoPacket _OnionHead::set_onion_message
protomessage::ProtoPacket _OnionHead::set_onionid
protomessage::ProtoPacket _OnionHead::set_padding
protomessage::ProtoPacket _OnionBody::set_padding
ns3::OnionManager:: UcharToString
protomessage::ProtoPacket _OnionHead::ByteSizeLong
protomessage::ProtoPacket _OnionBody::ByteSizeLong
ns3::Wsn_node::SendSegment
protomessage::ProtoPacket _OnionHead::_internal_set _onionid
protomessage::ProtoPacket _OnionHead::_internal_onionid
protomessage::ProtoPacket _OnionHead::_internal_padding
protomessage::ProtoPacket _OnionHead::_internal_onion _message
protomessage::ProtoPacket _OnionBody::_internal_padding
protomessage::ProtoPacket _OnionBody::_internal_aggregatedvalue
ns3::Wsn_node::CheckSent Onion
ns3::OnionRouting:: BuildOnion
ns3::OnionRouting:: CreateOnion
ns3::OnionRouting:: AddressToStream
ns3::OnionRouting:: EncryptLayer
ns3::OnionRouting:: OnionLength

Here is the caller graph for this function:



3.27.3.2 CheckOnion() `void ns3::Sink::CheckOnion ( void )`

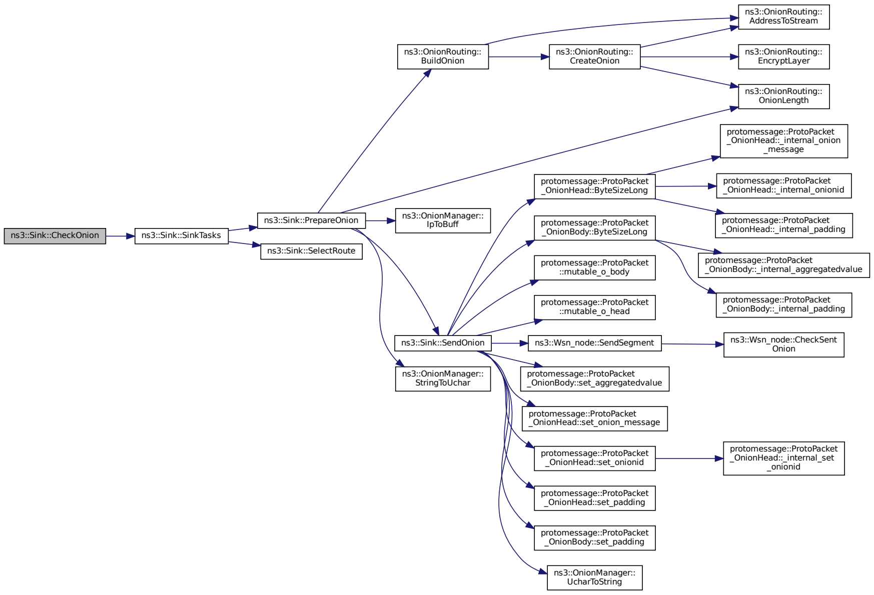
Each five seconds call the ns3::OnionValidator and check if at least one onion message is alive Otherwise the onion was aborted, decrease the value of `m_repeateCount` and schedule a new execution of the function ns3::Sink::SinkTasks()

Definition at line 322 of file sink.cc.

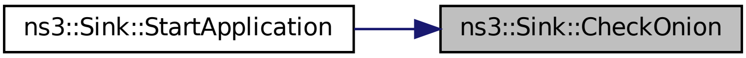
References ns3::Wsn_node::m_onionValidator, m_repeateCount, and SinkTasks().

Referenced by StartApplication().

Here is the call graph for this function:


ns3::Sink::CheckOnion
ns3::Sink::SinkTasks
ns3::Sink::PrepareOnion
ns3::Sink::SelectRoute
ns3::OnionRouting::BuildOnion
ns3::OnionRouting::CreateOnion
ns3::OnionRouting::AddressToStream
ns3::OnionRouting::EncryptLayer
ns3::OnionRouting::OnionLength
ns3::OnionManager::IpToBuff
ns3::Sink::SendOnion
ns3::OnionManager::StringToUchar
protomessage::ProtoPacket_OnionHead::_internal_onion_message
protomessage::ProtoPacket_OnionHead::ByteSizeLong
protomessage::ProtoPacket_OnionHead::_internal_onionid
protomessage::ProtoPacket_OnionHead::_internal_padding
protomessage::ProtoPacket_OnionBody::ByteSizeLong
protomessage::ProtoPacket_OnionBody::_internal_aggregatedvalue
protomessage::ProtoPacket::mutable_o_body
protomessage::ProtoPacket_OnionBody::_internal_padding
protomessage::ProtoPacket::mutable_o_head
ns3::Wsn_node::SendSegment
ns3::Wsn_node::CheckSentOnion
protomessage::ProtoPacket_OnionBody::set_aggregatedvalue
protomessage::ProtoPacket_OnionHead::set_onion_message
protomessage::ProtoPacket_OnionHead::set_onionid
protomessage::ProtoPacket_OnionHead::_internal_set_onionid
protomessage::ProtoPacket_OnionHead::set_padding
protomessage::ProtoPacket_OnionBody::set_padding
ns3::OnionManager::UcharToString

Here is the caller graph for this function:



3.27.3.3 GetTypeId() `TypeId ns3::Sink::GetTypeId ( void ) [static]`

Register this type.

Returns

The object TypeId.

Definition at line 32 of file sink.cc.

References ns3::Aggregate, ns3::AggregateAndFixed, ns3::FixedSize, m_bodyOptions, m_bodySize, m_fixedOnionSize, m_numnodes, m_onionDelay, and ns3::NO_Body.

3.27.3.4 PrepareOnion() `void ns3::Sink::PrepareOnion ( int * route, int routeLen ) [private]`

Method that constructs the onion head from `route` and `routeLen` parameters.

The method constructs two arrays `keys` , `ipRoute` containing respectively encryption keys and IP addreses of sensor nodes in the `m_nodeManager` structure at indexes specified by the `route` array. The onion head is constructed by calling ns3::OnionRouting::BuildOnion().

Parameters

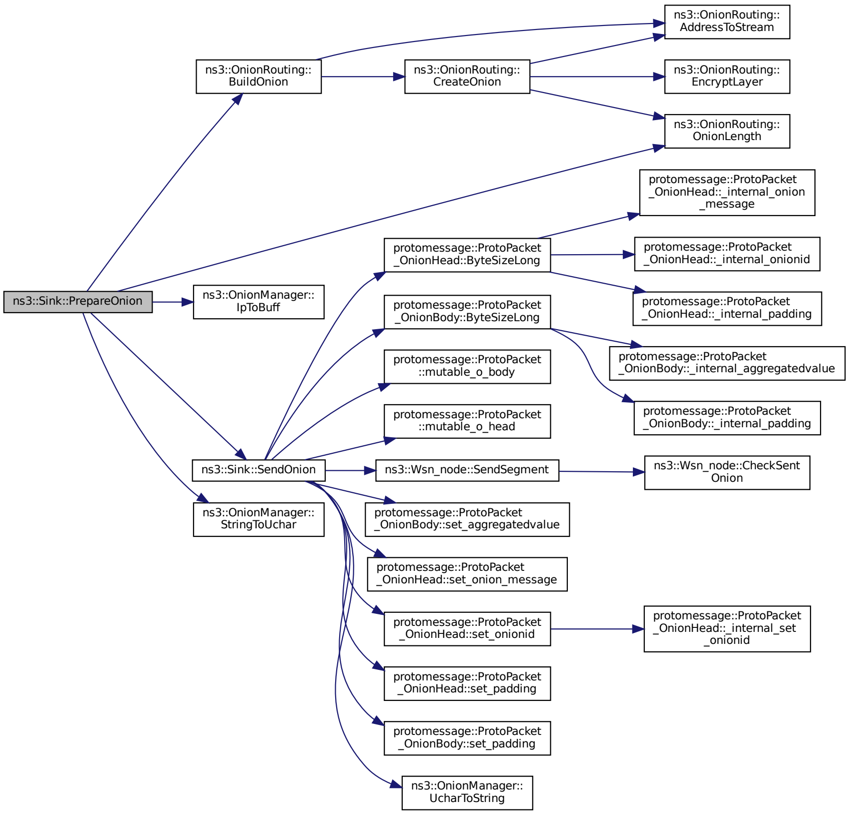
`in,out`	*route*	pointer to an array of length `routeLen` cointaining indexes of sensor nodes in the `m_nodeManager` structure.
`in`	*routeLen*	length of the array `route`

Definition at line 229 of file sink.cc.

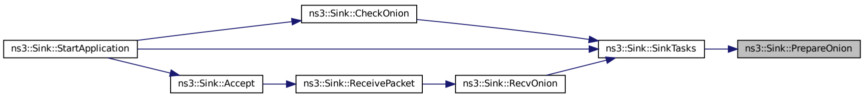
References ns3::OnionRouting::BuildOnion(), ns3::OnionManager::IpToBuff(), ns3::Wsn_node::m_address, m_↩ nodeManager, ns3::Wsn_node::m_onionManager, m_publickey, ns3::OnionRouting::OnionLength(), SendOnion(), and ns3::OnionManager::StringToUchar().

Referenced by SinkTasks().

Here is the call graph for this function:


ns3::OnionRouting:: AddressToStream
ns3::OnionRouting:: BuildOnion
ns3::OnionRouting:: CreateOnion
ns3::OnionRouting:: EncryptLayer
ns3::OnionRouting:: OnionLength
protomessage::ProtoPacket _OnionHead::_internal_onion _message
protomessage::ProtoPacket _OnionHead::ByteSizeLong
protomessage::ProtoPacket _OnionHead::_internal_onionid
ns3::Sink::PrepareOnion
ns3::OnionManager:: IpToBuff
protomessage::ProtoPacket _OnionBody::ByteSizeLong
protomessage::ProtoPacket _OnionHead::_internal_padding
protomessage::ProtoPacket ::mutable_o_body
protomessage::ProtoPacket _OnionBody::_internal_aggregatedvalue
protomessage::ProtoPacket ::mutable_o_head
protomessage::ProtoPacket _OnionBody::_internal_padding
ns3::Sink::SendOnion
ns3::Wsn_node::SendSegment
ns3::Wsn_node::CheckSent Onion
ns3::OnionManager:: StringToUchar
protomessage::ProtoPacket _OnionBody::set_aggregatedvalue
protomessage::ProtoPacket _OnionHead::set_onion_message
protomessage::ProtoPacket _OnionHead::set_onionid
protomessage::ProtoPacket _OnionHead::_internal_set _onionid
protomessage::ProtoPacket _OnionHead::set_padding
protomessage::ProtoPacket _OnionBody::set_padding
ns3::OnionManager:: UcharToString

Here is the caller graph for this function:



3.27.3.5 ReceivePacket() `void ns3::Sink::ReceivePacket ( Ptr< Socket > socket ) [private]`

Recieve a new packet from `socket` Check if the packet is a handshake packet or a packet containing an onion message.

1. Handshake packet -> call ns3::Sink::RecvHandshake()
2. Onion message -> call ns3::Sink::RecvOnion()

Parameters

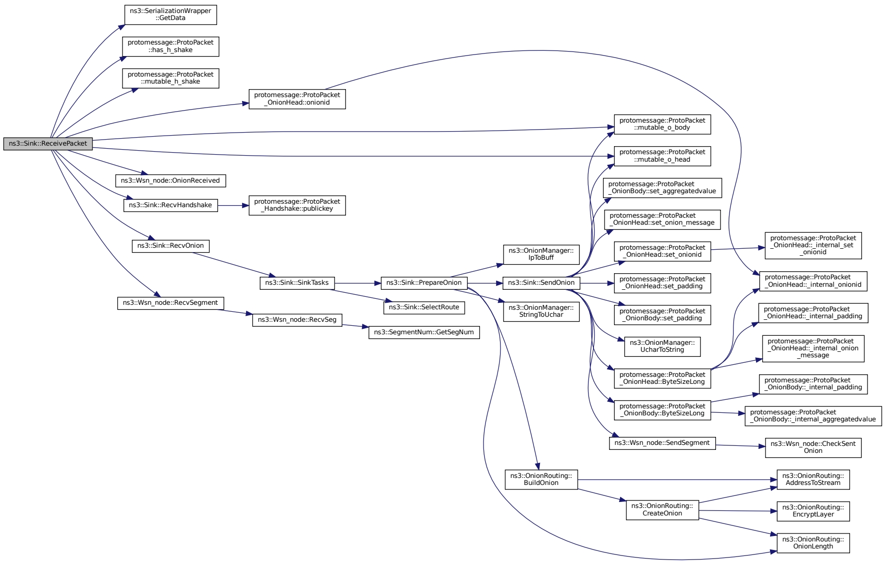
in	*socket*	the receiving tcp socket

Definition at line 95 of file sink.cc.

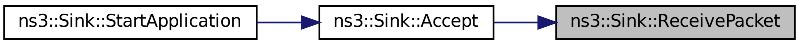
References ns3::SerializationWrapper::GetData(), protomessage::ProtoPacket::has_h_shake(), ns3::Wsn_node::m_address, m_onionId, protomessage::ProtoPacket::mutable_h_shake(), protomessage::ProtoPacket::mutable_o_body(), protomessage::ProtoPacket::mutable_o_head(), ns3::Wsn_node::o_sequenceNum, protomessage::ProtoPacket_OnionHead::onionid(), ns3::Wsn_node::OnionReceived(), RecvHandshake(), RecvOnion(), and ns3::Wsn_node::RecvSegment().

Referenced by Accept().

Here is the call graph for this function:


ns3::Sink::ReceivePacket
ns3::SerializationWrapper ::GetData
protomessage::ProtoPacket ::has_h_shake
protomessage::ProtoPacket ::mutable_h_shake
protomessage::ProtoPacket _OnionHead::onionid
ns3::Wsn_node::OnionReceived
ns3::Sink::RecvHandshake
protomessage::ProtoPacket _Handshake::publickey
ns3::Sink::RecvOnion
ns3::Wsn_node::RecvSegment
ns3::Sink::SinkTasks
ns3::Wsn_node::RecvSeg
ns3::Sink::PrepareOnion
ns3::Sink::SelectRoute
ns3::SegmentNum::GetSegNum
ns3::OnionManager:: IpToBuff
ns3::Sink::SendOnion
ns3::OnionManager:: StringToUchar
protomessage::ProtoPacket ::mutable_o_body
protomessage::ProtoPacket ::mutable_o_head
protomessage::ProtoPacket _OnionBody::set_aggregatedvalue
protomessage::ProtoPacket _OnionHead::set_onion_message
protomessage::ProtoPacket _OnionHead::set_onionid
protomessage::ProtoPacket _OnionHead::set_padding
protomessage::ProtoPacket _OnionBody::set_padding
ns3::OnionManager:: UcharToString
protomessage::ProtoPacket _OnionHead::ByteSizeLong
protomessage::ProtoPacket _OnionBody::ByteSizeLong
ns3::Wsn_node::SendSegment
protomessage::ProtoPacket _OnionHead::_internal_set _onionid
protomessage::ProtoPacket _OnionHead::_internal_onionid
protomessage::ProtoPacket _OnionHead::_internal_padding
protomessage::ProtoPacket _OnionHead::_internal_onion _message
protomessage::ProtoPacket _OnionBody::_internal_padding
protomessage::ProtoPacket _OnionBody::_internal_aggregatedvalue
ns3::Wsn_node::CheckSent Onion
ns3::OnionRouting:: BuildOnion
ns3::OnionRouting:: CreateOnion
ns3::OnionRouting:: AddressToStream
ns3::OnionRouting:: EncryptLayer
ns3::OnionRouting:: OnionLength

Here is the caller graph for this function:



3.27.3.6 RecvHandshake()

`void ns3::Sink::RecvHandshake ( protomessage::ProtoPacket_Handshake * handshake_data, InetSocketAddress from ) [private]`

When receiving a new handshake with a node.

The sink node stores the sensor node IP address and publickey (PK) in std::map structure `m_nodeManager` , where IP is the key and PK is the value

Parameters

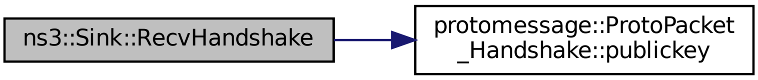
in	*handshake_data*	pointer to the protobuf object holding message data
in	*from*	the IP address of the sensor node

Definition at line 141 of file sink.cc.

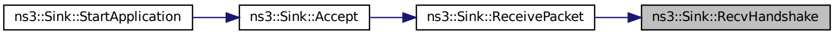
References m_nodeManager, ns3::Wsn_node::m_outputManager, and protomessage::ProtoPacket_Handshake↩ ::publickey().

Referenced by ReceivePacket().

Here is the call graph for this function:



Here is the caller graph for this function:



3.27.3.7 RecvOnion() `void ns3::Sink::RecvOnion ( protomessage::ProtoPacket_OnionBody * onion_body ) [private]`

Triggered when an onion message is received back at the sink node Data about the message is captured and a new onion message is scheduled.

Parameters

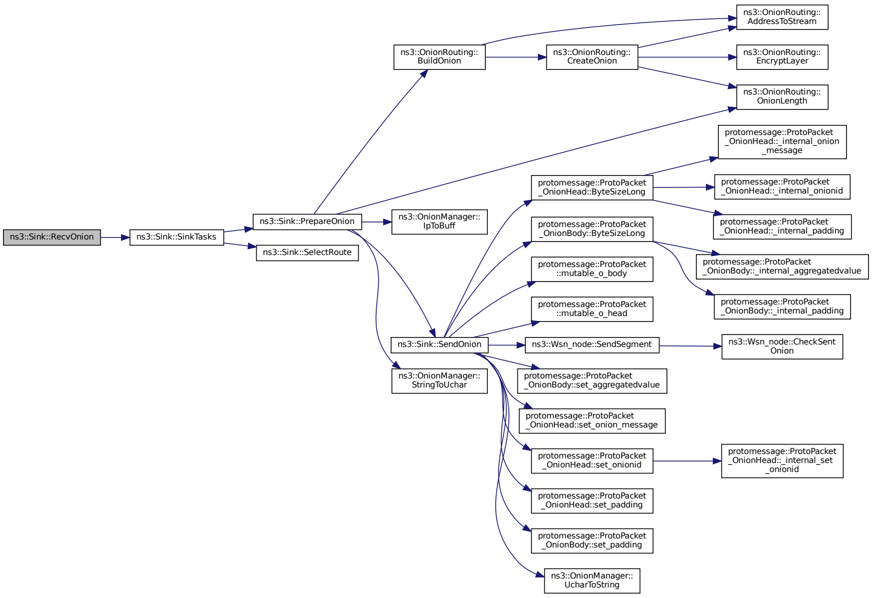
in	*onion_body*	pointer to the protobuf object holding onion data

Definition at line 153 of file sink.cc.

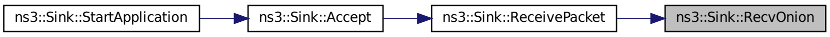
References ns3::Wsn_node::m_outputManager, and SinkTasks().

Referenced by ReceivePacket().

Here is the call graph for this function:


ns3::Sink::RecvOnion
ns3::Sink::SinkTasks
ns3::Sink::PrepareOnion
ns3::Sink::SelectRoute
ns3::OnionRouting::BuildOnion
ns3::OnionRouting::CreateOnion
ns3::OnionRouting::AddressToStream
ns3::OnionRouting::EncryptLayer
ns3::OnionRouting::OnionLength
ns3::OnionManager::IpToBuff
ns3::OnionManager::StringToUchar
ns3::Sink::SendOnion
protomessage::ProtoPacket_OnionHead::ByteSizeLong
protomessage::ProtoPacket_OnionHead::_internal_onion_message
protomessage::ProtoPacket_OnionHead::_internal_onionid
protomessage::ProtoPacket_OnionHead::_internal_padding
protomessage::ProtoPacket_OnionBody::ByteSizeLong
protomessage::ProtoPacket_OnionBody::_internal_aggregatedvalue
protomessage::ProtoPacket_OnionBody::_internal_padding
protomessage::ProtoPacket::mutable_o_body
protomessage::ProtoPacket::mutable_o_head
ns3::Wsn_node::SendSegment
ns3::Wsn_node::CheckSentOnion
protomessage::ProtoPacket_OnionBody::set_aggregatedvalue
protomessage::ProtoPacket_OnionHead::set_onion_message
protomessage::ProtoPacket_OnionHead::set_onionid
protomessage::ProtoPacket_OnionHead::_internal_set_onionid
protomessage::ProtoPacket_OnionHead::set_padding
protomessage::ProtoPacket_OnionBody::set_padding
ns3::OnionManager::UcharToString

Here is the caller graph for this function:



3.27.3.8 SelectRoute()

```
void ns3::Sink::SelectRoute (
    int * route,
    int routeLen ) [private]
```

The method builds the path of the onion message by randomly selecting sensor nodes from the `m_nodeManager` structure.

The path can have loops, but the same node cannot be placed in two consequent postions in the onion message path. The onion path must be of length >= 3.

Parameters

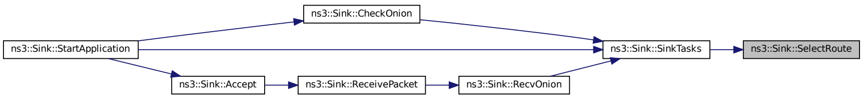
in,out	*route*	pointer to an array of length `routeLen`
in	*routeLen*	length of the array `route`

Definition at line 201 of file sink.cc.

References m_nodeManager.

Referenced by SinkTasks().

Here is the caller graph for this function:



3.27.3.9 SendOnion()

```
void ns3::Sink::SendOnion (
        uint32_t firstHop,
        int routeLen,
        unsigned char * cipher,
        int cipherLen ) [private]
```

The method constructs the onion message as a protobuf object.

The method sends the onion message to the sesor node at the `firstHop` address The method sets the tracking of the onion by calling ns3::OnionValidator::StartOnion()

Parameters

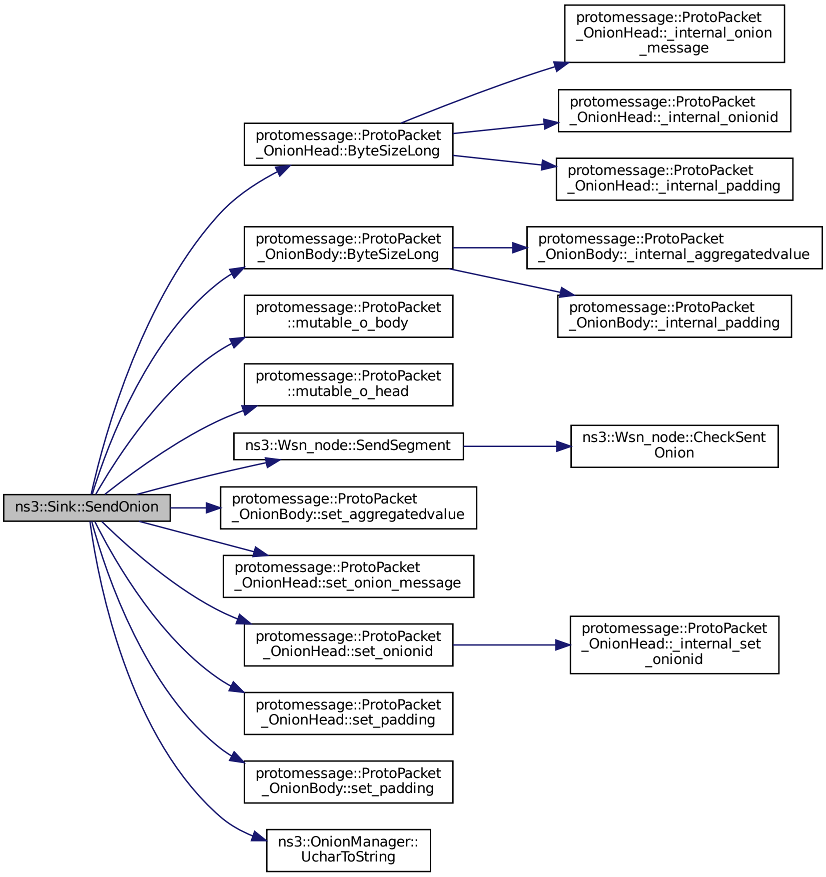
`in`	*firstHop*	IP address of the first sensor node in the onion path
`in`	*routeLen*	length of the onion path
`in,out`	*cipher*	pointer to the ciphertext of the onion head
`in`	*cipherLen*	length in the number of bytes of the `cipher`

Definition at line 265 of file sink.cc.

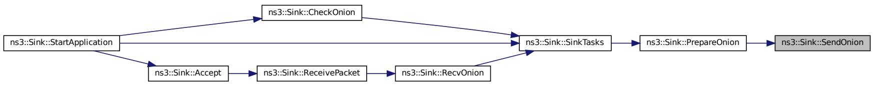
References ns3::Aggregate, ns3::AggregateAndFixed, protomessage::ProtoPacket_OnionHead::ByteSizeLong(), protomessage::ProtoPacket_OnionBody::ByteSizeLong(), ns3::FixedSize, ns3::Wsn_node::m_address, m_bodyOptions, m_bodySize, m_decoyNum, m_fixedOnionSize, m_onionId, ns3::Wsn_node::m_onionManager, ns3::Wsn_node::m_onionValidator, ns3::Wsn_node::m_outputManager, ns3::Wsn_node::m_port, protomessage::ProtoPacket::mutable_o_body(), protomessage::ProtoPacket::mutable_o_head(), ns3::NO_Body, ns3::Wsn_node::SendSegment(), protomessage::ProtoPacket_OnionBody::set_aggregatedvalue(), protomessage::ProtoPacket_OnionHead::set_onion_message(), protomessage::ProtoPacket_OnionHead::set_onionid(), protomessage::ProtoPacket_OnionHead::set_padding(), protomessage::ProtoPacket_OnionBody::set_padding(), and ns3::OnionManager::UcharToString().

Referenced by PrepareOnion().

Here is the call graph for this function:


protomessage::ProtoPacket _OnionHead::_internal_onion _message
protomessage::ProtoPacket _OnionHead::_internal_onionid
protomessage::ProtoPacket _OnionHead::ByteSizeLong
protomessage::ProtoPacket _OnionHead::_internal_padding
protomessage::ProtoPacket _OnionBody::ByteSizeLong
protomessage::ProtoPacket _OnionBody::_internal_aggregatedvalue
protomessage::ProtoPacket ::mutable_o_body
protomessage::ProtoPacket _OnionBody::_internal_padding
protomessage::ProtoPacket ::mutable_o_head
ns3::Wsn_node::SendSegment
ns3::Wsn_node::CheckSent Onion
ns3::Sink::SendOnion
protomessage::ProtoPacket _OnionBody::set_aggregatedvalue
protomessage::ProtoPacket _OnionHead::set_onion_message
protomessage::ProtoPacket _OnionHead::set_onionid
protomessage::ProtoPacket _OnionHead::_internal_set _onionid
protomessage::ProtoPacket _OnionHead::set_padding
protomessage::ProtoPacket _OnionBody::set_padding
ns3::OnionManager:: UcharToString

Here is the caller graph for this function:



3.27.3.10 Setup() `void ns3::Sink::Setup (`

```
uint16_t * onionPathlengths,
uint16_t numOnionLengths,
int repeateTimes )
```

Setup sink node parameters after the application installation.

Parameters

`in`	*onionPathLengths*	pointer to the array where each value is representing the number of hops the onion will travel to return back to the sink node issuer of the onion.
`in`	*numOnionLengths*	number of cells of the array `onionPathLengths`
`in`	*repeateTimes*	integer specifying the number of times to generate the onion message for each value of the array `onionPathsLengths`

Definition at line 73 of file sink.cc.

References m_numOnionLengths, m_onionPathLengths, and m_repeateTimes.

3.27.3.11 SinkTasks() `void ns3::Sink::SinkTasks ( )`

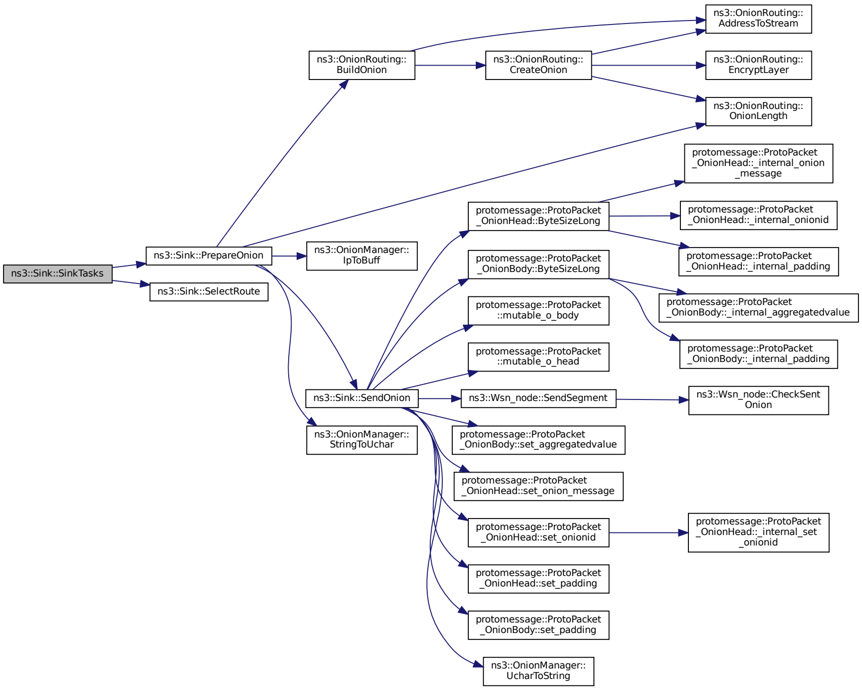
Schedule the creation of a new onion based on the path length specified in `m_onionPathLengths` If all onions specified in `m_onionPathLengths` were executed for `m_repeateTimes` Then end the simulation.

Definition at line 162 of file sink.cc.

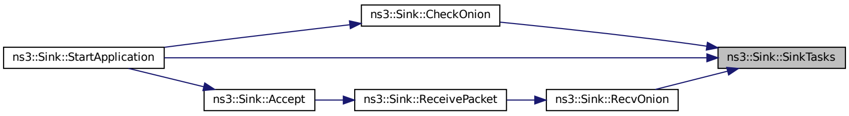
References m_nodeManager, m_numOnionLengths, m_onionLengthIndex, m_onionPathLengths, ns3::Wsn_↩ node::m_outputManager, m_repeateCount, m_repeateTimes, PrepareOnion(), and SelectRoute().

Referenced by CheckOnion(), RecvOnion(), and StartApplication().

Here is the call graph for this function:


ns3::Sink::SinkTasks
ns3::Sink::PrepareOnion
ns3::Sink::SelectRoute
ns3::OnionRouting::BuildOnion
ns3::OnionRouting::CreateOnion
ns3::OnionRouting::AddressToStream
ns3::OnionRouting::EncryptLayer
ns3::OnionRouting::OnionLength
ns3::OnionManager::IpToBuff
ns3::Sink::SendOnion
ns3::OnionManager::StringToUchar
protomessage::ProtoPacket_OnionHead::_internal_onion_message
protomessage::ProtoPacket_OnionHead::ByteSizeLong
protomessage::ProtoPacket_OnionHead::_internal_onionid
protomessage::ProtoPacket_OnionBody::ByteSizeLong
protomessage::ProtoPacket_OnionHead::_internal_padding
protomessage::ProtoPacket_OnionBody::_internal_aggregatedvalue
protomessage::ProtoPacket::mutable_o_body
protomessage::ProtoPacket_OnionBody::_internal_padding
protomessage::ProtoPacket::mutable_o_head
ns3::Wsn_node::SendSegment
ns3::Wsn_node::CheckSentOnion
protomessage::ProtoPacket_OnionBody::set_aggregatedvalue
protomessage::ProtoPacket_OnionHead::set_onion_message
protomessage::ProtoPacket_OnionHead::set_onionid
protomessage::ProtoPacket_OnionHead::_internal_set_onionid
protomessage::ProtoPacket_OnionHead::set_padding
protomessage::ProtoPacket_OnionBody::set_padding
ns3::OnionManager::UcharToString

Here is the caller graph for this function:



3.27.3.12 StartApplication() `void ns3::Sink::StartApplication ( void ) [private], [virtual]`

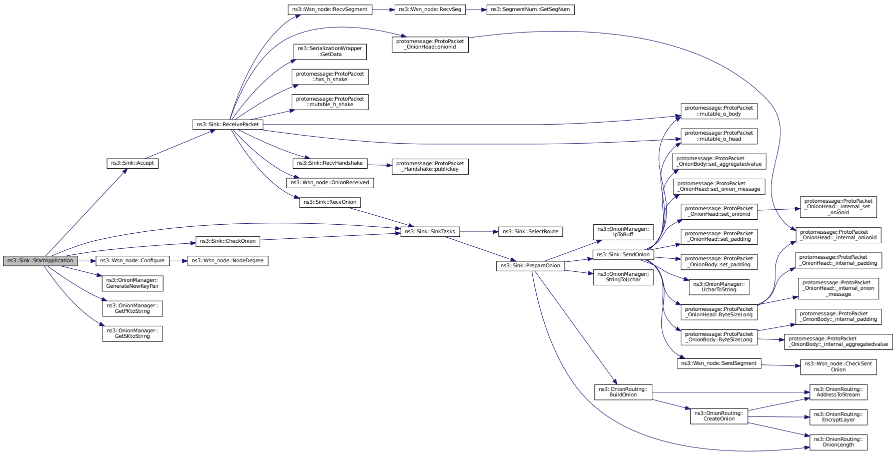
1.Start the application run ns3::Wsn_node::Configure() 2.Generate new encryption keys 3.Set callback at new socket connection 4.Schedule the execution of ns3::Sink::SinkTasks() after `m_onionDelay` milliseconds 5.↩ Schedule the execution of ns3::Sink::CheckOnion() after `m_onionDelay` milliseconds and 5 seconds

Reimplemented from ns3::Wsn_node.

Definition at line 338 of file sink.cc.

References Accept(), CheckOnion(), ns3::Wsn_node::Configure(), ns3::OnionManager::GenerateNewKeyPair(), ns3::OnionManager::GetPKtoString(), ns3::OnionManager::GetSKtoString(), ns3::Wsn_node::m_delay, m_↩ numnodes, m_onionDelay, ns3::Wsn_node::m_onionManager, m_publickey, m_secretkey, ns3::Wsn_node::m↩ _socket, and SinkTasks().

Here is the call graph for this function:


ns3::Wsn_node::RecvSegment
ns3::Wsn_node::RecvSeg
ns3::SegmentNum::GetSegNum
protomessage::ProtoPacket _OnionHead::onionid
ns3::SerializationWrapper ::GetData
protomessage::ProtoPacket ::has_h_shake
protomessage::ProtoPacket ::mutable_h_shake
ns3::Sink::ReceivePacket
protomessage::ProtoPacket ::mutable_o_body
protomessage::ProtoPacket ::mutable_o_head
ns3::Sink::Accept
ns3::Sink::RecvHandshake
protomessage::ProtoPacket _Handshake::publickey
protomessage::ProtoPacket _OnionBody::set_aggregatedvalue
ns3::Wsn_node::OnionReceived
protomessage::ProtoPacket _OnionHead::set_onion_message
ns3::Sink::RecvOnion
protomessage::ProtoPacket _OnionHead::set_onionid
protomessage::ProtoPacket _OnionHead::_internal_set _onionid
ns3::OnionManager:: IpToBuff
ns3::Sink::SinkTasks
ns3::Sink::SelectRoute
protomessage::ProtoPacket _OnionHead::set_padding
protomessage::ProtoPacket _OnionHead::_internal_onionid
ns3::Sink::CheckOnion
ns3::Sink::SendOnion
protomessage::ProtoPacket _OnionBody::set_padding
protomessage::ProtoPacket _OnionHead::_internal_padding
ns3::Sink::StartApplication
ns3::Wsn_node::Configure
ns3::Wsn_node::NodeDegree
ns3::Sink::PrepareOnion
ns3::OnionManager:: StringToUchar
ns3::OnionManager:: UcharToString
protomessage::ProtoPacket _OnionHead::_internal_onion _message
ns3::OnionManager:: GenerateNewKeyPair
ns3::OnionManager:: GetPKtoString
protomessage::ProtoPacket _OnionHead::ByteSizeLong
protomessage::ProtoPacket _OnionBody::_internal_padding
ns3::OnionManager:: GetSKtoString
protomessage::ProtoPacket _OnionBody::ByteSizeLong
protomessage::ProtoPacket _OnionBody::_internal_aggregatedvalue
ns3::Wsn_node::SendSegment
ns3::Wsn_node::CheckSent Onion
ns3::OnionRouting:: BuildOnion
ns3::OnionRouting:: AddressToStream
ns3::OnionRouting:: CreateOnion
ns3::OnionRouting:: EncryptLayer
ns3::OnionRouting:: OnionLength

3.27.3.13 StopApplication() `void ns3::Sink::StopApplication ( void ) [private], [virtual]`

Stop the application.

Reimplemented from ns3::Wsn_node.

Definition at line 359 of file sink.cc.

References ns3::Wsn_node::m_socket.

3.27.4 Member Data Documentation

3.27.4.1 m_bodyOptions `enum BodyOptions ns3::Sink::m_bodyOptions [private]`

Definition at line 214 of file sink.h.

Referenced by GetTypeId(), and SendOnion().

3.27.4.2 m_bodySize `uint16_t ns3::Sink::m_bodySize [private]`

Definition at line 215 of file sink.h.

Referenced by GetTypeId(), and SendOnion().

3.27.4.3 m_decoyNum `uint32_t ns3::Sink::m_decoyNum [private]`

Initial value:

```
=
      1203
```

dummy decoy value used to obfuscate the value carried in the onion body

Definition at line 210 of file sink.h.

Referenced by SendOnion().

3.27.4.4 m_fixedOnionSize `bool ns3::Sink::m_fixedOnionSize [private]`

maintain the onion size fixed by adding padding (after layer decryption)

Definition at line 213 of file sink.h.

Referenced by GetTypeId(), and SendOnion().

3.27.4.5 m_nodeManager `std::map<uint32_t, std::string> ns3::Sink::m_nodeManager [private]`

hashmap to manage data about nodes in the WSN// pair <IP,publickey>

Definition at line 209 of file sink.h.

Referenced by PrepareOnion(), RecvHandshake(), SelectRoute(), and SinkTasks().

3.27.4.6 m_numnodes `uint16_t ns3::Sink::m_numnodes [private]`

The number of sensor nodes in the simulation.

Definition at line 206 of file sink.h.

Referenced by GetTypeId(), and StartApplication().

3.27.4.7 m_numOnionLengths `uint16_t ns3::Sink::m_numOnionLengths [private]`

size of the array m_onionPathsLengths

Definition at line 222 of file sink.h.

Referenced by Setup(), and SinkTasks().

3.27.4.8 m_onionDelay `uint32_t ns3::Sink::m_onionDelay [private]`

The sink will start sending onion messagess after OnionDelay seconds.

Definition at line 207 of file sink.h.

Referenced by GetTypeId(), and StartApplication().

3.27.4.9 m_onionId `int ns3::Sink::m_onionId = 1 [private]`

onion ID incremented each time a new onion is issued

Definition at line 225 of file sink.h.

Referenced by ReceivePacket(), and SendOnion().

3.27.4.10 m_onionLengthIndex `int ns3::Sink::m_onionLengthIndex = 0 [private]`

index of the current onion path length

Definition at line 220 of file sink.h.

Referenced by SinkTasks().

3.27.4.11 m_onionPathLengths `uint16_t* ns3::Sink::m_onionPathLengths [private]`

array holding onion path lengths

Definition at line 221 of file sink.h.

Referenced by Setup(), and SinkTasks().

3.27.4.12 m_onionStream `std::stringstream ns3::Sink::m_onionStream [private]`

a string stream holding the onion represented as a string

Definition at line 228 of file sink.h.

3.27.4.13 m_publickey `std::string ns3::Sink::m_publickey [private]`

the encryption key: publickey

Definition at line 231 of file sink.h.

Referenced by PrepareOnion(), and StartApplication().

3.27.4.14 m_repeateCount `int ns3::Sink::m_repeateCount = 0 [private]`

how many times the onion was sent at the current path length

Definition at line 219 of file sink.h.

Referenced by CheckOnion(), and SinkTasks().

3.27.4.15 m_repeateTimes `int ns3::Sink::m_repeateTimes [private]`

Integer specifying the number of times to generate the onion message for each value in the m_onionPathsLengths.

Definition at line 218 of file sink.h.

Referenced by Setup(), and SinkTasks().

3.27.4.16 m_secretkey `std::string ns3::Sink::m_secretkey [private]`

the encryption key: secretkey

Definition at line 232 of file sink.h.

Referenced by StartApplication().

The documentation for this class was generated from the following files:

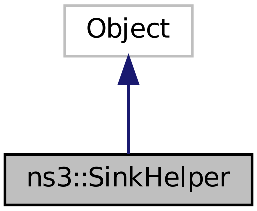
- src/onion_routing_wsn/model/sink.h
- src/onion_routing_wsn/model/sink.cc

3.28 ns3::SinkHelper Class Reference

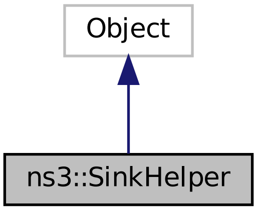
Helper class for the creation of Sink applications.

```
#include "sink-helper.h"
```

Inheritance diagram for ns3::SinkHelper:



Collaboration diagram for ns3::SinkHelper:



Public Member Functions

- SinkHelper ()

  Default constructor.
- SinkHelper (uint16_t numnodes, Ptr< OutputManager > outputManager, Ptr< OnionValidator > onionValidator, uint16_t ∗m_onionPathLengths)

  Constructor with attributes to set basic attributes of the sink node.
- ∼SinkHelper ()

  Default constructor.
- ApplicationContainer Install (NodeContainer c) const
- Ptr< Application > InstallPriv (Ptr< Node > node) const
- void SetAttribute (std::string name, const AttributeValue &value)

  Setter of individual attributes.

Static Public Member Functions

- static TypeId GetTypeId (void)

  Register this type.

Private Attributes

- ObjectFactory m_factory

  factory object
- uint16_t m_numnodes

  Number of sensor nodes in the network.
- uint16_t m_onionTimeout

  A watchdog timer set to abort onion messagess, if the timer elepses before the onion returns back to the sink node.
- Ptr< OnionValidator > m_onionValidator

  Manage onions and when to abort them.
- Ptr< OutputManager > m_outputManager

  Manage the output of the simulation.

3.28.1 Detailed Description

Helper class for the creation of Sink applications.

Definition at line 66 of file sink-helper.h.

3.28.2 Constructor & Destructor Documentation

3.28.2.1 SinkHelper() [1/2] `ns3::SinkHelper::SinkHelper ( )`

Default constructor.

Definition at line 39 of file sink-helper.cc.

References m_factory.

3.28.2.2 SinkHelper() [2/2]

```
ns3::SinkHelper::SinkHelper (
        uint16_t numnodes,
        Ptr< OutputManager > outputManager,
        Ptr< OnionValidator > onionValidator,
        uint16_t * m_onionPathLengths )
```

Constructor with attributes to set basic attributes of the sink node.

Definition at line 44 of file sink-helper.cc.

References m_factory, m_numnodes, m_onionValidator, and m_outputManager.

3.28.2.3 ∼SinkHelper()

```
ns3::SinkHelper::∼SinkHelper ( )
```

Default constructor.

Definition at line 54 of file sink-helper.cc.

3.28.3 Member Function Documentation

3.28.3.1 GetTypeId()

```
TypeId ns3::SinkHelper::GetTypeId (
        void ) [static]
```

Register this type.

Returns

The object TypeId.

Definition at line 32 of file sink-helper.cc.

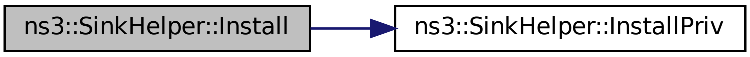
3.28.3.2 Install() `ApplicationContainer ns3::SinkHelper::Install ( NodeContainer c ) const`

Definition at line 65 of file sink-helper.cc.

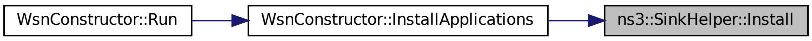
References InstallPriv().

Referenced by WsnConstructor::InstallApplications().

Here is the call graph for this function:



Here is the caller graph for this function:



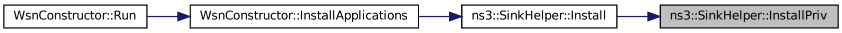
3.28.3.3 InstallPriv() `Ptr< Application > ns3::SinkHelper::InstallPriv ( Ptr< Node > node ) const`

Definition at line 77 of file sink-helper.cc.

References m_factory.

Referenced by Install().

Here is the caller graph for this function:


WsnConstructor::Run
WsnConstructor::InstallApplications
ns3::SinkHelper::Install
ns3::SinkHelper::InstallPriv

3.28.3.4 SetAttribute() `void ns3::SinkHelper::SetAttribute ( std::string name, const AttributeValue & value )`

Setter of individual attributes.

Definition at line 59 of file sink-helper.cc.

References m_factory.

3.28.4 Member Data Documentation

3.28.4.1 m_factory `ObjectFactory ns3::SinkHelper::m_factory [private]`

factory object

Definition at line 124 of file sink-helper.h.

Referenced by InstallPriv(), SetAttribute(), and SinkHelper().

3.28.4.2 m_numnodes `uint16_t ns3::SinkHelper::m_numnodes [private]`

Number of sensor nodes in the network.

Definition at line 123 of file sink-helper.h.

Referenced by SinkHelper().

3.28.4.3 m_onionTimeout `uint16_t ns3::SinkHelper::m_onionTimeout [private]`

A watchdog timer set to abort onion messagess, if the timer elepses before the onion returns back to the sink node.

Definition at line 128 of file sink-helper.h.

3.28.4.4 m_onionValidator `Ptr<OnionValidator> ns3::SinkHelper::m_onionValidator [private]`

Manage onions and when to abort them.

Definition at line 126 of file sink-helper.h.

Referenced by SinkHelper().

3.28.4.5 m_outputManager `Ptr<OutputManager> ns3::SinkHelper::m_outputManager [private]`

Manage the output of the simulation.

Definition at line 125 of file sink-helper.h.

Referenced by SinkHelper().

The documentation for this class was generated from the following files:

- src/onion_routing_wsn/helper/sink-helper.h
- src/onion_routing_wsn/helper/sink-helper.cc

3.29 TableStruct_proto_2dpacket_2eproto Struct Reference

`#include "proto-packet.pb.h"`

Static Public Member Functions

- static const ::PROTOBUF_NAMESPACE_ID::internal::ParseTableField entries[ ] PROTOBUF_SECTION_VARIABLE (protodesc_cold)
- static const ::PROTOBUF_NAMESPACE_ID::internal::AuxiliaryParseTableField aux[ ] PROTOBUF_SECTION_VARIABLE (protodesc_cold)
- static const ::PROTOBUF_NAMESPACE_ID::internal::ParseTable schema[4] PROTOBUF_SECTION_VARIABLE (protodesc_cold)

Static Public Attributes

- static const ::PROTOBUF_NAMESPACE_ID::internal::FieldMetadata field_metadata [ ]
- static const uint32_t offsets [ ]
- static const ::PROTOBUF_NAMESPACE_ID::internal::SerializationTable serialization_table [ ]

3.29.1 Detailed Description

Definition at line 44 of file proto-packet.pb.h.

3.29.2 Member Function Documentation

3.29.2.1 PROTOBUF_SECTION_VARIABLE() [1/3] `static const ::PROTOBUF_NAMESPACE_ID::internal↩ ::ParseTableField entries [ ] TableStruct_proto_2dpacket_2eproto::PROTOBUF_SECTION_VARIABLE ( protodesc_cold ) [static]`

3.29.2.2 PROTOBUF_SECTION_VARIABLE() [2/3]

```
static const ::PROTOBUF_NAMESPACE_ID::internal::AuxiliaryParseTableField aux [] TableStruct_proto_2dpacket_2eproto::PROTOBUF_SECTION_VARIABLE (
            protodesc_cold ) [static]
```

3.29.2.3 PROTOBUF_SECTION_VARIABLE() [3/3]

```
static const ::PROTOBUF_NAMESPACE_ID::internal::ParseTable schema [4] TableStruct_proto_2dpacket_2eproto::PROTOBUF_SECTION_VARIABLE (
            protodesc_cold ) [static]
```

3.29.3 Member Data Documentation

3.29.3.1 field_metadata

```
const ::PROTOBUF_NAMESPACE_ID::internal::FieldMetadata TableStruct_proto_2dpacket_2eproto::field_metadata[ ] [static]
```

Definition at line 51 of file proto-packet.pb.h.

3.29.3.2 offsets

```
const uint32_t TableStruct_proto_2dpacket_2eproto::offsets[ ] [static]
```

Definition at line 53 of file proto-packet.pb.h.

3.29.3.3 serialization_table

```
const ::PROTOBUF_NAMESPACE_ID::internal::SerializationTable TableStruct_proto_2dpacket_2eproto::serialization_table[ ] [static]
```

Definition at line 52 of file proto-packet.pb.h.

The documentation for this struct was generated from the following file:

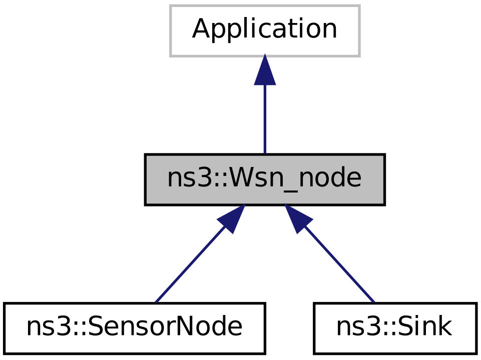
- src/onion_routing_wsn/protobuf/proto-packet.pb.h

3.30 ns3::Wsn_node Class Reference

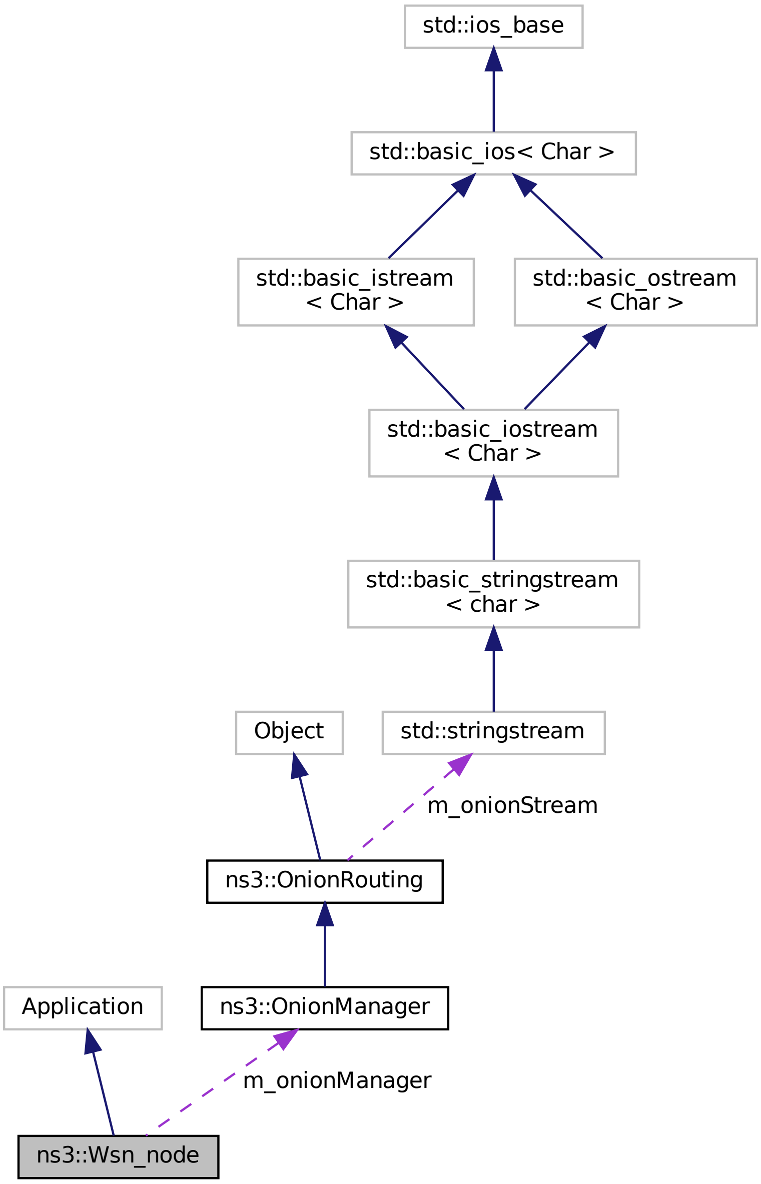
The wsn node base class that manages the sending and receiving of packets and basic configuration of nodes.

```
#include "wsn_node.h"
```

Inheritance diagram for ns3::Wsn_node:



Collaboration diagram for ns3::Wsn_node:



Public Member Functions

- Wsn_node ()

  Default constructor.
- void CheckSentOnion (int count)

  The method checks if the onion was received by the next receiving node If the onion was not received then abort the current onion and schedule the sending of another onion with equal parametrs as the aborted onion.
- void Configure (void)

  1.
- uint32_t getNodeDelay (Ipv4Address node_address)

compute when the node should start the handshake process from the node ip address Used to not generate a network congestion if all nodes will strat the handshake at the same time

- void NodeDegree (double coord_x, double coord_y)

  Retrieve the degreee this node from the OLSR routing protocol.
- void OnionReceived (void)

  Signal to the ns3::OnionValidator that the onion was corrctly received.
- Ptr< Packet > RecvSeg (Ptr< Socket > socket, Ptr< Packet > p, Address from)

  method for receiving packets able to merge segment fragments if a packet was split into multiple segments due to being larger than the MSS Use the packet tag, to merge packet fragments into a single packet.
- Ptr< Packet > RecvSegment (Ptr< Socket > socket)

  method for receiving a segment calls ns3::Wsn_node::RecvSeg()
- Ptr< Packet > RecvSegment (Ptr< Socket > socket, Address &from)

  method for receiving a segment calls ns3::Wsn_node::RecvSeg()
- void SendSegment (InetSocketAddress remote, Ptr< Packet > packet, bool b_onion)

  Send a packet through a TCP connection to the remote address.

Static Public Member Functions

- static TypeId GetTypeId (void)

  Register this type.

Protected Attributes

- uint16_t f_mss

  maximum segment size
- Ptr< Packet > f_pendingPacket

  pointer to the packet where received segment fragments are stored
- Ipv4Address f_receivingAddress = Ipv4Address::GetAny ()

  the receiving address
- int f_segmentSize

  the size of the whole packet
- Ipv4Address m_address

  ns3::Ipv4Address of this node
- uint16_t m_delay

  delay after which the handshake process will start
- OnionManager m_onionManager

  The ns3::OnionManager object.
- uint16_t m_onionTimeout

  timer in seconds, if elepsed and the onion was not recieved by the next receiver, then delete the onion
- Ptr< OnionValidator > m_onionValidator

  Pointer to the ns3::OnionValidator.
- Ptr< OutputManager > m_outputManager

  Pointer to the ns3::OutputManager.
- uint16_t m_port

  port of the application
- Ptr< Socket > m_socket

  listening socket
- int o_hopCount = 0

  track how the onion is is transiting in the network
- int o_sequenceNum = 0

  sequence number of the onion, should be same as onion_id

Private Member Functions

- virtual void StartApplication (void)

  Start the application.
- virtual void StopApplication (void)

  Stop the application.

3.30.1 Detailed Description

The wsn node base class that manages the sending and receiving of packets and basic configuration of nodes.

Definition at line 62 of file wsn_node.h.

3.30.2 Constructor & Destructor Documentation

3.30.2.1 Wsn_node() `ns3::Wsn_node::Wsn_node ( )`

Default constructor.

Definition at line 64 of file wsn_node.cc.

3.30.3 Member Function Documentation

3.30.3.1 CheckSentOnion() `void ns3::Wsn_node::CheckSentOnion ( int count )`

The method checks if the onion was received by the next receiving node If the onion was not received then abort the current onion and schedule the sending of another onion with equal parametrs as the aborted onion.

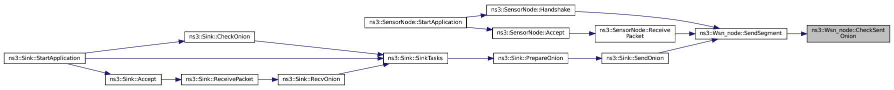
Checking onion.

Definition at line 277 of file wsn_node.cc.

References m_onionValidator, and m_outputManager.

Referenced by SendSegment().

Here is the caller graph for this function:


ns3::SensorNode::Handshake
ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::Wsn_node::SendSegment
ns3::Wsn_node::CheckSent Onion
ns3::Sink::CheckOnion
ns3::Sink::StartApplication
ns3::Sink::SinkTasks
ns3::Sink::PrepareOnion
ns3::Sink::SendOnion
ns3::Sink::Accept
ns3::Sink::ReceivePacket
ns3::Sink::RecvOnion

3.30.3.2 Configure() `void ns3::Wsn_node::Configure ( void )`

1.

configure basic attributes of nodes

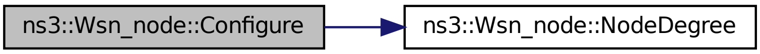
1. create socket and listen for connections
2. get node position in the form of x and y coordinates

Definition at line 79 of file wsn_node.cc.

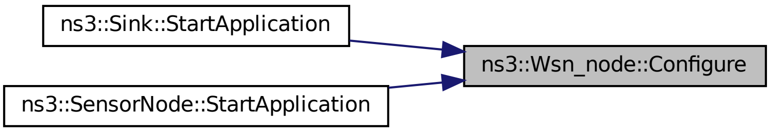
References m_address, m_outputManager, m_port, m_socket, NodeDegree(), and ns3::OLSR.

Referenced by ns3::Sink::StartApplication(), and ns3::SensorNode::StartApplication().

Here is the call graph for this function:



Here is the caller graph for this function:


ns3::Sink::StartApplication
ns3::SensorNode::StartApplication
ns3::Wsn_node::Configure

3.30.3.3 getNodeDelay() `uint32_t ns3::Wsn_node::getNodeDelay ( Ipv4Address node_address )`

compute when the node should start the handshake process from the node ip address Used to not generate a network congestion if all nodes will strat the handshake at the same time

Parameters

in	*node_address*	the ipv4 address

Returns

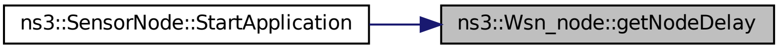
return the delay in milliseconds

Definition at line 154 of file wsn_node.cc.

References m_delay.

Referenced by ns3::SensorNode::StartApplication().

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::Wsn_node::getNodeDelay

3.30.3.4 GetTypeId()

```
TypeId ns3::Wsn_node::GetTypeId (
        void ) [static]
```

Register this type.

Returns

The object TypeId.

Definition at line 30 of file wsn_node.cc.

References f_mss, m_delay, m_onionTimeout, m_onionValidator, m_outputManager, and m_port.

3.30.3.5 NodeDegree()

```
void ns3::Wsn_node::NodeDegree (
        double coord_x,
        double coord_y )
```

Retrieve the degreee this node from the OLSR routing protocol.

Works only if using the OLSR protocol. We refer to the degreee of the current node as the number of nodes that are in the one-hop neighbourhood of the current node with respect to the multi-hop communication.

Parameters

in	*coord↵ _x*	x coordinate of the current node
out	*coord↵ _y*	y coordinate of the current node

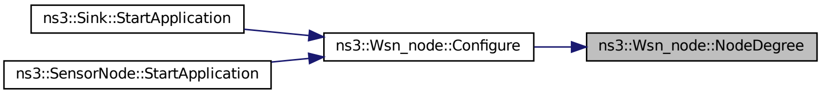
Get one hop neighbours:

Definition at line 119 of file wsn_node.cc.

References m_address, and m_outputManager.

Referenced by Configure().

Here is the caller graph for this function:


ns3::Sink::StartApplication
ns3::SensorNode::StartApplication
ns3::Wsn_node::Configure
ns3::Wsn_node::NodeDegree

3.30.3.6 OnionReceived() `void ns3::Wsn_node::OnionReceived ( void )`

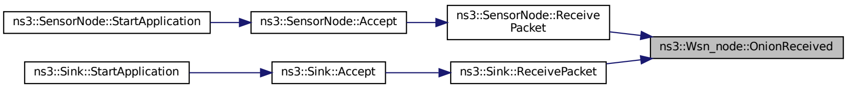
Signal to the ns3::OnionValidator that the onion was corrctly received.

Definition at line 287 of file wsn_node.cc.

References m_onionValidator.

Referenced by ns3::SensorNode::ReceivePacket(), and ns3::Sink::ReceivePacket().

Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::Wsn_node::OnionReceived
ns3::Sink::StartApplication
ns3::Sink::Accept
ns3::Sink::ReceivePacket

3.30.3.7 RecvSeg()

```
Ptr< Packet > ns3::Wsn_node::RecvSeg (
            Ptr< Socket > socket,
            Ptr< Packet > p,
            Address from )
```

method for receiving packets able to merge segment fragments if a packet was split into multiple segments due to being larger than the MSS Use the packet tag, to merge packet fragments into a single packet.

Parameters

in	*socket*	the receiving socket
in	*packet*	pointer to the receiving packet
in,out	*from*	extract the sender IP from the receiving socket

Returns

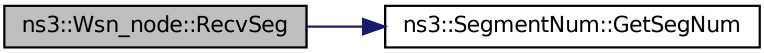
> return pointer to packet if the whole packet is received OR return NULL value if only packet fragment is received

Definition at line 235 of file wsn_node.cc.

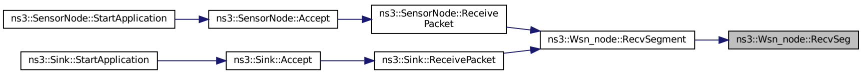
References f_pendingPacket, f_receivingAddress, f_segmentSize, and ns3::SegmentNum::GetSegNum().

Referenced by RecvSegment().

Here is the call graph for this function:



Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::Sink::StartApplication
ns3::Sink::Accept
ns3::Sink::ReceivePacket
ns3::Wsn_node::RecvSegment
ns3::Wsn_node::RecvSeg

3.30.3.8 RecvSegment() `[1/2]` `Ptr< Packet > ns3::Wsn_node::RecvSegment ( Ptr< Socket > socket )`

method for receiving a segment calls ns3::Wsn_node::RecvSeg()

Parameters

in	*socket*	the receiving socket

Returns

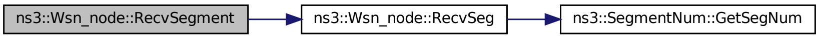
> the result of the ns3::Wsn_node::RecvSeg() function

Definition at line 205 of file wsn_node.cc.

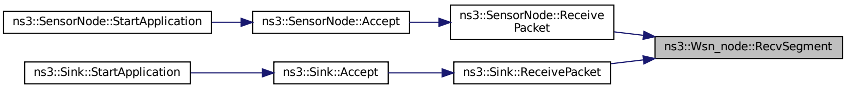
References RecvSeg().

Referenced by ns3::SensorNode::ReceivePacket(), and ns3::Sink::ReceivePacket().

Here is the call graph for this function:



Here is the caller graph for this function:


ns3::SensorNode::StartApplication
ns3::SensorNode::Accept
ns3::SensorNode::Receive Packet
ns3::Wsn_node::RecvSegment
ns3::Sink::StartApplication
ns3::Sink::Accept
ns3::Sink::ReceivePacket

3.30.3.9 RecvSegment() `[2/2]`

```
Ptr< Packet > ns3::Wsn_node::RecvSegment (
        Ptr< Socket > socket,
        Address & from )
```

method for receiving a segment calls ns3::Wsn_node::RecvSeg()

Parameters

`in`	*socket*	the receiving socket
`in,out`	*from*	extract the sender IP from the receiving socket

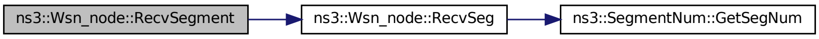
Returns

the result of the ns3::Wsn_node::RecvSeg() function

Definition at line 218 of file wsn_node.cc.

References RecvSeg().

Here is the call graph for this function:



3.30.3.10 SendSegment() `void ns3::Wsn_node::SendSegment ( InetSocketAddress remote, Ptr< Packet > packet, bool b_onion )`

Send a packet through a TCP connection to the remote address.

Set b_onion to true to send an onion message. If b_onion is true, the method sets a callback after `m_onionTimeout` seconds The callback triggers the function ns3::Wsn_node::CheckSentOnion(). If the packe size is larger than the MSS. Then add a tag to the packet, which specifies the size of the whole packet. The tag is used by ns3::Wsn_node::RecvSeg() to reconstruct a packet that was split into multiple segments.

Parameters

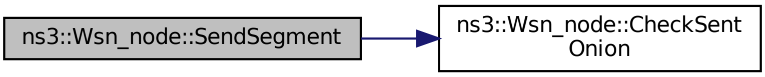
in	*remote*	the receiving address
in	*packet*	the packet to send
in	*b_onion*	boolean value, set to true if sending an onion message. Adds the packet TAG to the packet

Definition at line 173 of file wsn_node.cc.

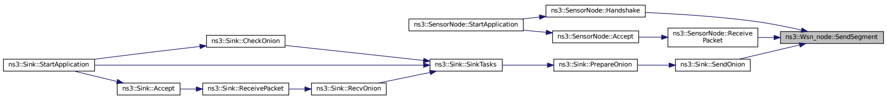
References CheckSentOnion(), f_mss, m_onionTimeout, m_onionValidator, and o_hopCount.

Referenced by ns3::SensorNode::Handshake(), ns3::SensorNode::ReceivePacket(), and ns3::Sink::SendOnion().

Here is the call graph for this function:



Here is the caller graph for this function:



3.30.3.11 StartApplication() `void ns3::Wsn_node::StartApplication ( void ) [private], [virtual]`

Start the application.

Reimplemented in ns3::SensorNode, and ns3::Sink.

Definition at line 69 of file wsn_node.cc.

3.30.3.12 StopApplication() `void ns3::Wsn_node::StopApplication ( void ) [private], [virtual]`

Stop the application.

Reimplemented in ns3::SensorNode, and ns3::Sink.

Definition at line 74 of file wsn_node.cc.

3.30.4 Member Data Documentation

3.30.4.1 f_mss `uint16_t ns3::Wsn_node::f_mss [protected]`

maximum segment size

Definition at line 197 of file wsn_node.h.

Referenced by GetTypeId(), and SendSegment().

3.30.4.2 f_pendingPacket `Ptr<Packet> ns3::Wsn_node::f_pendingPacket [protected]`

pointer to the packet where received segment fragments are stored

Definition at line 200 of file wsn_node.h.

Referenced by RecvSeg().

3.30.4.3 f_receivingAddress `Ipv4Address ns3::Wsn_node::f_receivingAddress = Ipv4Address::GetAny () [protected]`

the receiving address

Definition at line 201 of file wsn_node.h.

Referenced by RecvSeg().

3.30.4.4 f_segmentSize `int ns3::Wsn_node::f_segmentSize [protected]`

the size of the whole packet

Definition at line 198 of file wsn_node.h.

Referenced by RecvSeg().

3.30.4.5 m_address `Ipv4Address ns3::Wsn_node::m_address [protected]`

ns3::Ipv4Address of this node

Definition at line 191 of file wsn_node.h.

Referenced by Configure(), NodeDegree(), ns3::Sink::PrepareOnion(), ns3::SensorNode::ReceivePacket(), ns3::↩ Sink::ReceivePacket(), ns3::Sink::SendOnion(), and ns3::SensorNode::StartApplication().

3.30.4.6 m_delay `uint16_t ns3::Wsn_node::m_delay [protected]`

delay after which the handshake process will start

Definition at line 193 of file wsn_node.h.

Referenced by getNodeDelay(), GetTypeId(), and ns3::Sink::StartApplication().

3.30.4.7 m_onionManager `OnionManager ns3::Wsn_node::m_onionManager [protected]`

The ns3::OnionManager object.

Definition at line 194 of file wsn_node.h.

Referenced by ns3::SensorNode::Handshake(), ns3::Sink::PrepareOnion(), ns3::SensorNode::ProcessOnion↩ Head(), ns3::Sink::SendOnion(), ns3::Sink::StartApplication(), and ns3::SensorNode::StartApplication().

3.30.4.8 m_onionTimeout `uint16_t ns3::Wsn_node::m_onionTimeout [protected]`

timer in seconds, if elepsed and the onion was not recieved by the next receiver, then delete the onion

Definition at line 208 of file wsn_node.h.

Referenced by GetTypeId(), and SendSegment().

3.30.4.9 m_onionValidator `Ptr<OnionValidator> ns3::Wsn_node::m_onionValidator [protected]`

Pointer to the ns3::OnionValidator.

Definition at line 190 of file wsn_node.h.

Referenced by ns3::Sink::CheckOnion(), CheckSentOnion(), GetTypeId(), OnionReceived(), ns3::SensorNode::↩ ReceivePacket(), ns3::Sink::SendOnion(), and SendSegment().

3.30.4.10 m_outputManager `Ptr<OutputManager> ns3::Wsn_node::m_outputManager [protected]`

Pointer to the ns3::OutputManager.

Definition at line 189 of file wsn_node.h.

Referenced by CheckSentOnion(), Configure(), GetTypeId(), NodeDegree(), ns3::SensorNode::ReceivePacket(), ns3::Sink::RecvHandshake(), ns3::Sink::RecvOnion(), ns3::Sink::SendOnion(), and ns3::Sink::SinkTasks().

3.30.4.11 m_port `uint16_t ns3::Wsn_node::m_port [protected]`

port of the application

Definition at line 188 of file wsn_node.h.

Referenced by Configure(), GetTypeId(), ns3::SensorNode::Handshake(), ns3::SensorNode::ReceivePacket(), and ns3::Sink::SendOnion().

3.30.4.12 m_socket `Ptr<Socket> ns3::Wsn_node::m_socket [protected]`

listening socket

Definition at line 192 of file wsn_node.h.

Referenced by Configure(), ns3::Sink::StartApplication(), ns3::SensorNode::StartApplication(), ns3::Sink::Stop↩ Application(), ns3::SensorNode::StopApplication(), ns3::SensorNode::∼SensorNode(), and ns3::Sink::∼Sink().

3.30.4.13 o_hopCount `int ns3::Wsn_node::o_hopCount = 0 [protected]`

track how the onion is is transiting in the network

Definition at line 204 of file wsn_node.h.

Referenced by SendSegment().

3.30.4.14 o_sequenceNum `int ns3::Wsn_node::o_sequenceNum = 0 [protected]`

sequence number of the onion, should be same as onion_id

Definition at line 205 of file wsn_node.h.

Referenced by ns3::SensorNode::ReceivePacket(), and ns3::Sink::ReceivePacket().

The documentation for this class was generated from the following files:

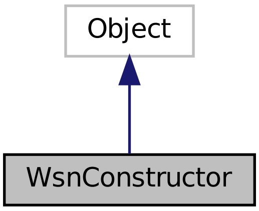
- src/onion_routing_wsn/model/wsn_node.h
- src/onion_routing_wsn/model/wsn_node.cc

3.31 WsnConstructor Class Reference

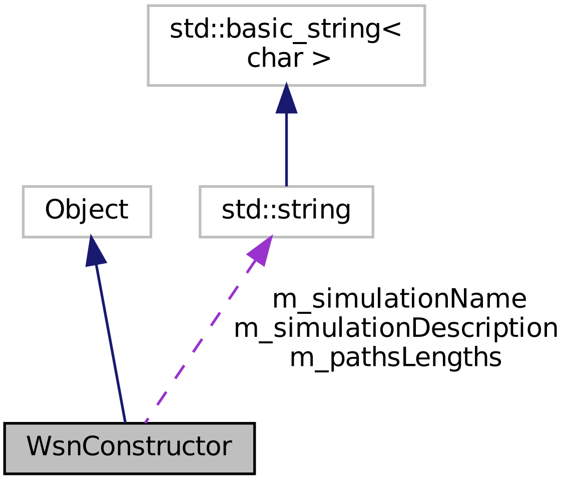
The class that constructs the WSN, setup applications on nodes and starts the simulation.

`#include "wsnconstructor.h"`

Inheritance diagram for WsnConstructor:



Collaboration diagram for WsnConstructor:



Public Member Functions

- WsnConstructor ()

  Default constructor.
- void Configure ()

  Generate a description of attributes given from the config file and from CLI set the simulation seed and verbosity.
- void Run ()

  Construct the WSN, install applications on nodes and start the simulation.

Static Public Member Functions

- static TypeId GetTypeId (void)

  Register this type.

Private Member Functions

- void AODVrouting ()

  Install AODV routing.
- void BuildDiscTopology ()

  Deploy nodes at random positions on a disc shaped plane.
- void BuildGridTopology ()

  Sensor nodes are deployed according to a grid structure; each sensor node is equidistant from the closest sensor nodes in cardinal directions.
- void CreateDevices ()

  Create Network devices and setup the wireless communication based on the IEEE 802.11n Allow to select from 2.4GHz to 5GHz and setup different MCS using the attribute system.
- void CreateNodes ()

  Create node objects.
- void DSDVrouting ()

  Install DSDV routing.
- void DSRrouting ()

  Install DSR routing.
- void InstallApplications ()

  Install applications on nodes and setup starting time of the handshake and the onion start time.
- void InstallInternetStack ()

  Installing the internet stack on nodes and setting up IP-addresses.
- void OLSRrouting ()

  Install OLSR routing.
- void ProcessPathString ()

  Split the string `m_pathsLengths` by the delimiter (,) and save each value into `m_onionPathsLengths` and the number of values into `m_numOnionPaths`.

Private Attributes

- uint16_t m_cellSide

  Parameter for the setup of the grid topology.
- enum IEEE_80211n m_mac

  Carrier frequency of the IEEE 802.11n.
- uint16_t m_mss

  maximum segment size
- uint16_t m_numNodes

  number of sensor nodes in the WSN
- uint16_t m_numOnionPaths = 0

  Number of different onion paths.
- uint16_t ∗ m_onionPathsLengths

  Array containing one onion path length in each cell.
- uint16_t m_onionRepeate

  Number of times to generate the onion for each value of path length.
- Ptr< OnionValidator > m_onionValidator

Checks if the onion messagess transiting in the network are valid.

- Ptr< OutputManager > m_outputManager

  Manages the output of the simulation.
- std::string m_pathsLengths

  String of values delimited by the symbol "," each value representing the number of hops the onion will travel to return back to the sink node issuer of the onion.
- uint16_t m_radius

  Parameter for the setup of the random disc topology.
- enum Routing m_routing

  routing algorithm for wireless multihop networks
- std::string m_simulationDescription

  String holding a description of parameters used in the simulation.
- std::string m_simulationName

  name of the simulation
- uint32_t m_simulationSeed

  seed to use for the random generation of numbers during the simulation
- NodeContainer m_sink

  Container of the sink node.
- enum Topology m_topology

  network topology
- enum Verbosity m_verbosity

  verbosity of the simulation
- MobilityHelper mobility

  Topology helper.
- ApplicationContainer sensornodeApps

  Container of sensor node applications.
- NodeContainer sensornodes

  Container of sensor nodes.
- ApplicationContainer sinkApps

  Container of sink node applications.
- NetDeviceContainer wifiDevices

  Container of wireless devices.
- Ipv4InterfaceContainer wifiInterfaces

  Container of netork interfaces.
- NodeContainer wifiNodes

  Container of wireless nodes.

3.31.1 Detailed Description

The class that constructs the WSN, setup applications on nodes and starts the simulation.

Definition at line 97 of file wsnconstructor.h.

3.31.2 Constructor & Destructor Documentation

3.31.2.1 WsnConstructor() `WsnConstructor::WsnConstructor ( )`

Default constructor.

Definition at line 116 of file wsnconstructor.cc.

3.31.3 Member Function Documentation

3.31.3.1 AODVrouting() `void WsnConstructor::AODVrouting ( ) [private]`

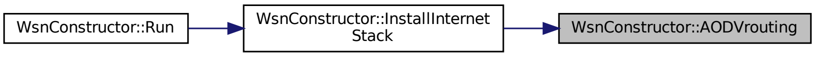
Install AODV routing.

Definition at line 511 of file wsnconstructor.cc.

References m_simulationDescription, and wifiNodes.

Referenced by InstallInternetStack().

Here is the caller graph for this function:


WsnConstructor::Run
WsnConstructor::InstallInternet Stack
WsnConstructor::AODVrouting

3.31.3.2 BuildDiscTopology() `void WsnConstructor::BuildDiscTopology ( ) [private]`

Deploy nodes at random positions on a disc shaped plane.

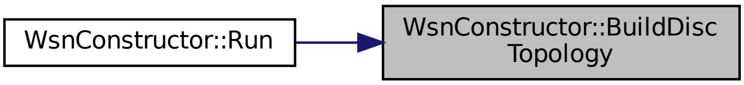
The radius of the disc is selected based on $r\_disc^2 * \pi = A$. A being the sum of circular areas covered by `m_numNodes` at radius `m_radius` Therefore the average sensor node density is maintained fixed if the parameter `m_numNodes` changes. The sink node is deployed in the center of the disc.

Definition at line 374 of file wsnconstructor.cc.

References m_numNodes, m_radius, m_simulationDescription, m_sink, mobility, and sensornodes.

Referenced by Run().

Here is the caller graph for this function:



3.31.3.3 BuildGridTopology() `void WsnConstructor::BuildGridTopology ( ) [private]`

Sensor nodes are deployed according to a grid structure; each sensor node is equidistant from the closest sensor nodes in cardinal directions.

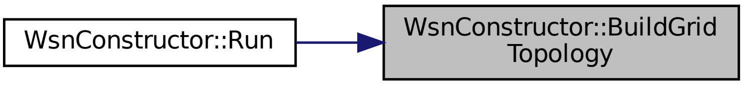
The distance between sensor nodes is defined by the parameter `m_cellSide` The sink node is deployed in the center of the grid repleacing a sensor node.

Definition at line 419 of file wsnconstructor.cc.

References m_cellSide, m_numNodes, m_simulationDescription, m_sink, mobility, and sensornodes.

Referenced by Run().

Here is the caller graph for this function:



3.31.3.4 Configure() `void WsnConstructor::Configure ( )`

Generate a description of attributes given from the config file and from CLI set the simulation seed and verbosity.

Definition at line 120 of file wsnconstructor.cc.

References ns3::AODV, ns3::Both, ns3::ConsoleLog, ns3::DISC, ns3::DSDV, ns3::DSR, ns3::GRID, m_mss, m_numNodes, m_onionValidator, m_outputManager, m_routing, m_simulationDescription, m_simulationName, m_simulationSeed, m_topology, m_verbosity, ns3::OLSR, and ns3::PrintDescription.

3.31.3.5 CreateDevices() `void WsnConstructor::CreateDevices ( ) [private]`

Create Network devices and setup the wireless communication based on the IEEE 802.11n Allow to select from 2.4GHz to 5GHz and setup different MCS using the attribute system.

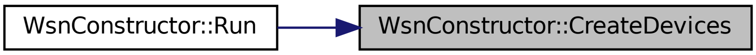
Set up the network: configure the physical mode, the wi-fi parameters (seting an adhoc wifi), etc.

Definition at line 296 of file wsnconstructor.cc.

References ns3::F_24GHz, ns3::F_5GHz, m_mac, m_mss, m_simulationDescription, wifiDevices, and wifiNodes.

Referenced by Run().

Here is the caller graph for this function:



3.31.3.6 CreateNodes() `void WsnConstructor::CreateNodes ( ) [private]`

Create node objects.

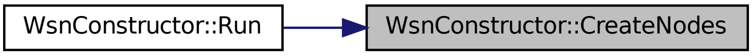
Create nodes and split them between node containers, to split nodes by functionality in the application.

Definition at line 277 of file wsnconstructor.cc.

References m_numNodes, m_simulationDescription, m_sink, sensornodes, and wifiNodes.

Referenced by Run().

Here is the caller graph for this function:



3.31.3.7 DSDVrouting() `void WsnConstructor::DSDVrouting ( ) [private]`

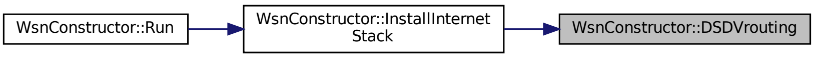
Install DSDV routing.

Definition at line 549 of file wsnconstructor.cc.

References m_simulationDescription, and wifiNodes.

Referenced by InstallInternetStack().

Here is the caller graph for this function:


WsnConstructor::Run
WsnConstructor::InstallInternet Stack
WsnConstructor::DSDVrouting

3.31.3.8 DSRrouting() `void WsnConstructor::DSRrouting ( ) [private]`

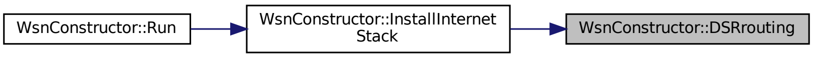
Install DSR routing.

Definition at line 497 of file wsnconstructor.cc.

References m_simulationDescription, and wifiNodes.

Referenced by InstallInternetStack().

Here is the caller graph for this function:


WsnConstructor::Run
WsnConstructor::InstallInternet Stack
WsnConstructor::DSRrouting

3.31.3.9 GetTypeId() `TypeId WsnConstructor::GetTypeId ( void ) [static]`

Register this type.

Returns

The object TypeId.

Definition at line 47 of file wsnconstructor.cc.

References ns3::AODV, ns3::Both, ns3::ConsoleLog, ns3::DISC, ns3::DSDV, ns3::DSR, ns3::F_24GHz, ns3::F_↩ 5GHz, ns3::GRID, m_cellSide, m_mac, m_mss, m_numNodes, m_onionRepeate, m_pathsLengths, m_radius, m↩ _routing, m_simulationName, m_simulationSeed, m_topology, m_verbosity, ns3::NO, ns3::OLSR, and ns3::Print↩ Description.

3.31.3.10 InstallApplications() `void WsnConstructor::InstallApplications ( ) [private]`

Install applications on nodes and setup starting time of the handshake and the onion start time.

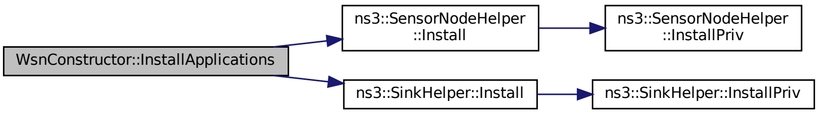
Install an aplication on network.

Definition at line 571 of file wsnconstructor.cc.

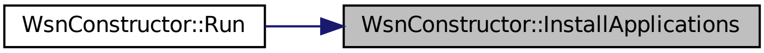
References ns3::DSDV, ns3::SensorNodeHelper::Install(), ns3::SinkHelper::Install(), m_numNodes, m_num↩ OnionPaths, m_onionPathsLengths, m_onionRepeate, m_onionValidator, m_outputManager, m_routing, m_↩ simulationDescription, m_sink, ns3::OLSR, sensornodeApps, sensornodes, sinkApps, and wifiInterfaces.

Referenced by Run().

Here is the call graph for this function:


WsnConstructor::InstallApplications
ns3::SensorNodeHelper ::Install
ns3::SensorNodeHelper ::InstallPriv
ns3::SinkHelper::Install
ns3::SinkHelper::InstallPriv

Here is the caller graph for this function:



3.31.3.11 InstallInternetStack() `void WsnConstructor::InstallInternetStack ( ) [private]`

Installing the internet stack on nodes and setting up IP-addresses.

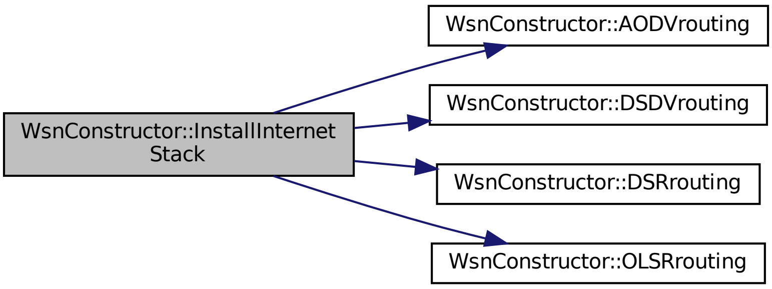
Set internet stack (ipv4) and its addresses.

Definition at line 464 of file wsnconstructor.cc.

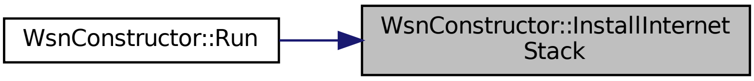
References ns3::AODV, AODVrouting(), ns3::DSDV, DSDVrouting(), ns3::DSR, DSRrouting(), m_outputManager, m_routing, ns3::OLSR, OLSRrouting(), wifiDevices, and wifiInterfaces.

Referenced by Run().

Here is the call graph for this function:


WsnConstructor::InstallInternet Stack
WsnConstructor::AODVrouting
WsnConstructor::DSDVrouting
WsnConstructor::DSRrouting
WsnConstructor::OLSRrouting

Here is the caller graph for this function:



3.31.3.12 OLSRrouting() `void WsnConstructor::OLSRrouting ( ) [private]`

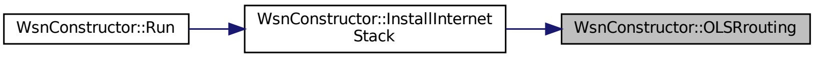
Install OLSR routing.

Definition at line 530 of file wsnconstructor.cc.

References m_simulationDescription, and wifiNodes.

Referenced by InstallInternetStack().

Here is the caller graph for this function:


WsnConstructor::Run
WsnConstructor::InstallInternet Stack
WsnConstructor::OLSRrouting

3.31.3.13 ProcessPathString() `void WsnConstructor::ProcessPathString ( ) [private]`

Split the string `m_pathsLengths` by the delimiter (,) and save each value into `m_onionPathsLengths` and the number of values into `m_numOnionPaths`.

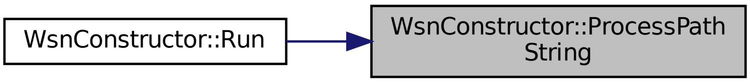
Convert the string given as parameter containing path lenghts in the array m_onionPathsLenghts containing a path length in each cell.

Definition at line 244 of file wsnconstructor.cc.

References m_numOnionPaths, m_onionPathsLengths, and m_pathsLengths.

Referenced by Run().

Here is the caller graph for this function:



3.31.3.14 Run() `void WsnConstructor::Run ( )`

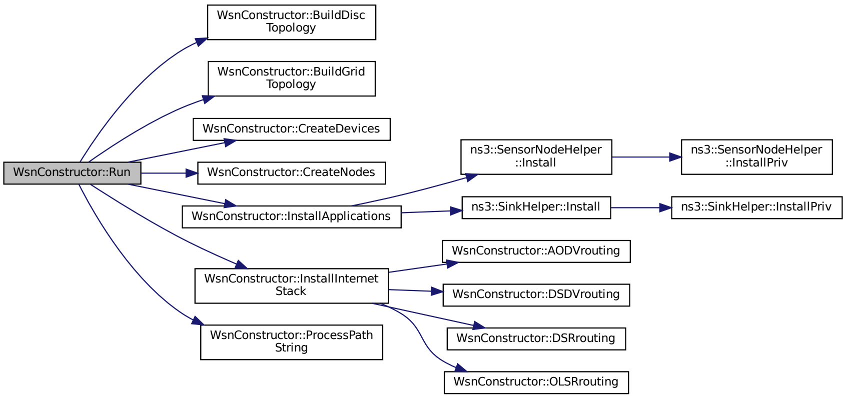
Construct the WSN, install applications on nodes and start the simulation.

Run the simulation, showing the progress of it.

Definition at line 194 of file wsnconstructor.cc.

References BuildDiscTopology(), BuildGridTopology(), CreateDevices(), CreateNodes(), ns3::DISC, ns3::GRID, InstallApplications(), InstallInternetStack(), m_outputManager, m_simulationDescription, m_simulationName, m←_topology, and ProcessPathString().

Here is the call graph for this function:


WsnConstructor::BuildDisc Topology
WsnConstructor::BuildGrid Topology
WsnConstructor::CreateDevices
WsnConstructor::Run
WsnConstructor::CreateNodes
WsnConstructor::InstallApplications
WsnConstructor::InstallInternet Stack
WsnConstructor::ProcessPath String
ns3::SensorNodeHelper ::Install
ns3::SensorNodeHelper ::InstallPriv
ns3::SinkHelper::Install
ns3::SinkHelper::InstallPriv
WsnConstructor::AODVrouting
WsnConstructor::DSDVrouting
WsnConstructor::DSRrouting
WsnConstructor::OLSRrouting

3.31.4 Member Data Documentation

3.31.4.1 m_cellSide `uint16_t WsnConstructor::m_cellSide [private]`

Parameter for the setup of the grid topology.

Definition at line 140 of file wsnconstructor.h.

Referenced by BuildGridTopology(), and GetTypeId().

3.31.4.2 m_mac `enum IEEE_80211n WsnConstructor::m_mac [private]`

Carrier frequency of the IEEE 802.11n.

Definition at line 136 of file wsnconstructor.h.

Referenced by CreateDevices(), and GetTypeId().

3.31.4.3 m_mss `uint16_t WsnConstructor::m_mss [private]`

maximum segment size

Definition at line 138 of file wsnconstructor.h.

Referenced by Configure(), CreateDevices(), and GetTypeId().

3.31.4.4 m_numNodes `uint16_t WsnConstructor::m_numNodes [private]`

number of sensor nodes in the WSN

Definition at line 133 of file wsnconstructor.h.

Referenced by BuildDiscTopology(), BuildGridTopology(), Configure(), CreateNodes(), GetTypeId(), and InstallApplications().

3.31.4.5 m_numOnionPaths `uint16_t WsnConstructor::m_numOnionPaths = 0 [private]`

Number of different onion paths.

Definition at line 151 of file wsnconstructor.h.

Referenced by InstallApplications(), and ProcessPathString().

3.31.4.6 m_onionPathsLengths `uint16_t* WsnConstructor::m_onionPathsLengths [private]`

Array containing one onion path length in each cell.

Definition at line 152 of file wsnconstructor.h.

Referenced by InstallApplications(), and ProcessPathString().

3.31.4.7 m_onionRepeate `uint16_t WsnConstructor::m_onionRepeate [private]`

Number of times to generate the onion for each value of path length.

Definition at line 153 of file wsnconstructor.h.

Referenced by GetTypeId(), and InstallApplications().

3.31.4.8 m_onionValidator `Ptr<OnionValidator> WsnConstructor::m_onionValidator [private]`

Checks if the onion messagess transiting in the network are valid.

Definition at line 145 of file wsnconstructor.h.

Referenced by Configure(), and InstallApplications().

3.31.4.9 m_outputManager `Ptr<OutputManager> WsnConstructor::m_outputManager [private]`

Manages the output of the simulation.

Definition at line 143 of file wsnconstructor.h.

Referenced by Configure(), InstallApplications(), InstallInternetStack(), and Run().

3.31.4.10 m_pathsLengths `std::string WsnConstructor::m_pathsLengths [private]`

String of values delimited by the symbol "," each value representing the number of hops the onion will travel to return back to the sink node issuer of the onion.

Definition at line 155 of file wsnconstructor.h.

Referenced by GetTypeId(), and ProcessPathString().

3.31.4.11 m_radius `uint16_t WsnConstructor::m_radius [private]`

Parameter for the setup of the random disc topology.

Definition at line 139 of file wsnconstructor.h.

Referenced by BuildDiscTopology(), and GetTypeId().

3.31.4.12 m_routing `enum Routing WsnConstructor::m_routing [private]`

routing algorithm for wireless multihop networks

Definition at line 134 of file wsnconstructor.h.

Referenced by Configure(), GetTypeId(), InstallApplications(), and InstallInternetStack().

3.31.4.13 m_simulationDescription `std::string WsnConstructor::m_simulationDescription [private]`

String holding a description of parameters used in the simulation.

Definition at line 147 of file wsnconstructor.h.

Referenced by AODVrouting(), BuildDiscTopology(), BuildGridTopology(), Configure(), CreateDevices(), CreateNodes(), DSDVrouting(), DSRrouting(), InstallApplications(), OLSRrouting(), and Run().

3.31.4.14 m_simulationName `std::string WsnConstructor::m_simulationName [private]`

name of the simulation

Definition at line 148 of file wsnconstructor.h.

Referenced by Configure(), GetTypeId(), and Run().

3.31.4.15 m_simulationSeed `uint32_t WsnConstructor::m_simulationSeed [private]`

seed to use for the random generation of numbers during the simulation

Definition at line 132 of file wsnconstructor.h.

Referenced by Configure(), and GetTypeId().

3.31.4.16 m_sink `NodeContainer WsnConstructor::m_sink [private]`

Container of the sink node.

Definition at line 241 of file wsnconstructor.h.

Referenced by BuildDiscTopology(), BuildGridTopology(), CreateNodes(), and InstallApplications().

3.31.4.17 m_topology `enum Topology WsnConstructor::m_topology [private]`

network topology

Definition at line 135 of file wsnconstructor.h.

Referenced by Configure(), GetTypeId(), and Run().

3.31.4.18 m_verbosity `enum Verbosity WsnConstructor::m_verbosity [private]`

verbosity of the simulation

Definition at line 137 of file wsnconstructor.h.

Referenced by Configure(), and GetTypeId().

3.31.4.19 mobility `MobilityHelper WsnConstructor::mobility [private]`

Topology helper.

Definition at line 239 of file wsnconstructor.h.

Referenced by BuildDiscTopology(), and BuildGridTopology().

3.31.4.20 sensornodeApps `ApplicationContainer WsnConstructor::sensornodeApps [private]`

Container of sensor node applications.

Definition at line 246 of file wsnconstructor.h.

Referenced by InstallApplications().

3.31.4.21 sensornodes `NodeContainer WsnConstructor::sensornodes [private]`

Container of sensor nodes.

Definition at line 242 of file wsnconstructor.h.

Referenced by BuildDiscTopology(), BuildGridTopology(), CreateNodes(), and InstallApplications().

3.31.4.22 sinkApps `ApplicationContainer WsnConstructor::sinkApps [private]`

Container of sink node applications.

Definition at line 245 of file wsnconstructor.h.

Referenced by InstallApplications().

3.31.4.23 wifiDevices `NetDeviceContainer WsnConstructor::wifiDevices [private]`

Container of wireless devices.

Definition at line 243 of file wsnconstructor.h.

Referenced by CreateDevices(), and InstallInternetStack().

3.31.4.24 wifiInterfaces `Ipv4InterfaceContainer WsnConstructor::wifiInterfaces [private]`

Container of netork interfaces.

Definition at line 244 of file wsnconstructor.h.

Referenced by InstallApplications(), and InstallInternetStack().

3.31.4.25 wifiNodes `NodeContainer WsnConstructor::wifiNodes [private]`

Container of wireless nodes.

Definition at line 240 of file wsnconstructor.h.

Referenced by AODVrouting(), CreateDevices(), CreateNodes(), DSDVrouting(), DSRrouting(), and OLSRrouting().

The documentation for this class was generated from the following files:

- src/onion_routing_wsn/wsnconstructor.h
- src/onion_routing_wsn/wsnconstructor.cc

4 File Documentation

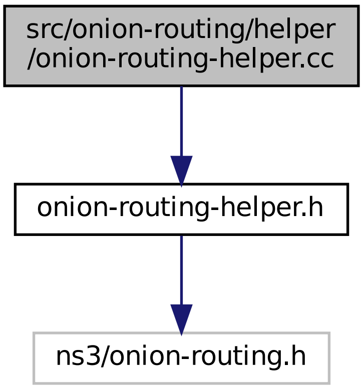
4.1 src/onion-routing/examples/onion-routing-dummy-encryption-example.cc File Reference

4.2 src/onion-routing/examples/onion-routing-example.cc File Reference

4.3 src/onion-routing/helper/onion-routing-helper.cc File Reference

```
#include "onion-routing-helper.h"
```

Include dependency graph for onion-routing-helper.cc:



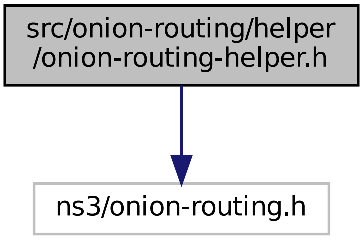
Namespaces

- ns3

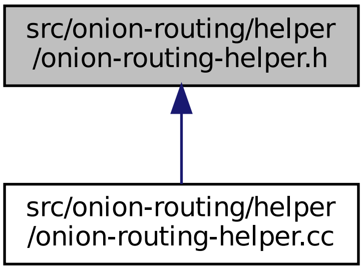
4.4 src/onion-routing/helper/onion-routing-helper.h File Reference

```
#include "ns3/onion-routing.h"
```

Include dependency graph for onion-routing-helper.h:



This graph shows which files directly or indirectly include this file:



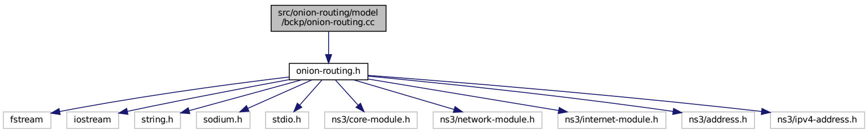
Namespaces

- ns3

4.5 src/onion-routing/model/bckp/onion-routing.cc File Reference

```
#include "onion-routing.h"
```

Include dependency graph for onion-routing.cc:



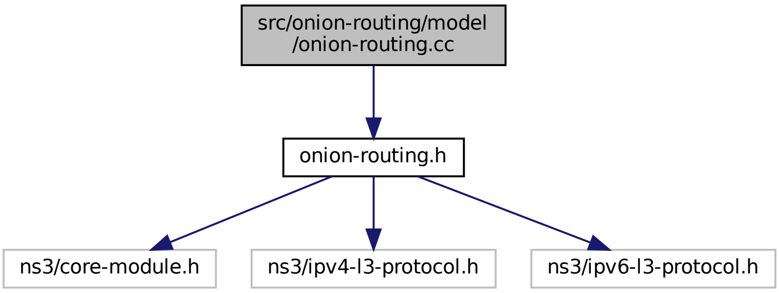
Namespaces

- ns3

4.6 src/onion-routing/model/onion-routing.cc File Reference

```
#include "onion-routing.h"
```

Include dependency graph for onion-routing.cc:



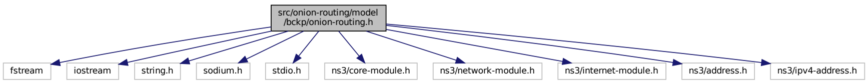
Namespaces

- ns3

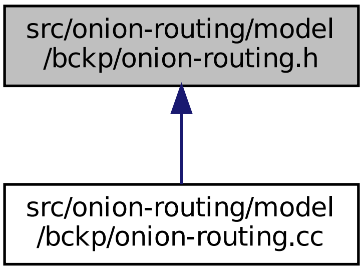
4.7 src/onion-routing/model/bckp/onion-routing.h File Reference

```
#include <fstream>
#include <iostream>
#include <string.h>
#include <sodium.h>
#include <stdio.h>
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/address.h"
#include "ns3/ipv4-address.h"
```

Include dependency graph for onion-routing.h:



This graph shows which files directly or indirectly include this file:



Classes

- class ns3::OnionRouting

    Abstract class for creation and decryption of Onion messages.
- struct ns3::orLayer

    structure holding details resulting from layer decryption of an onion message

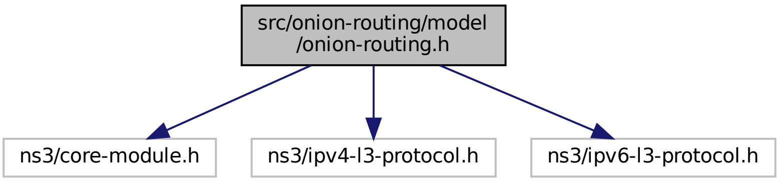
Namespaces

- ns3

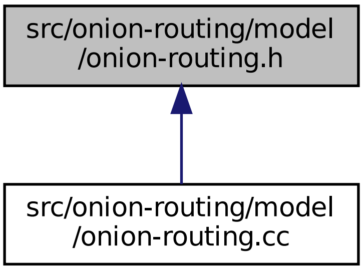
4.8 src/onion-routing/model/onion-routing.h File Reference

```
#include "ns3/core-module.h"
#include "ns3/ipv4-l3-protocol.h"
#include "ns3/ipv6-l3-protocol.h"
```

Include dependency graph for onion-routing.h:



This graph shows which files directly or indirectly include this file:



Classes

- class ns3::OnionRouting

  Abstract class for creation and decryption of Onion messages.
- class ns3::OnionRoutingDummyEncryption

  class that implements the
- struct ns3::orLayer

  structure holding details resulting from layer decryption of an onion message

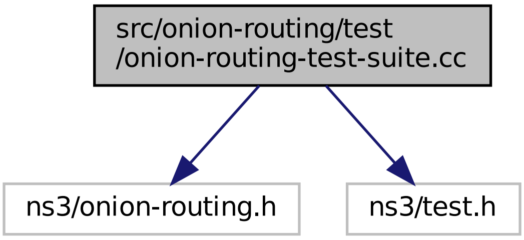
Namespaces

- ns3

4.9 src/onion-routing/test/onion-routing-test-suite.cc File Reference

```
#include "ns3/onion-routing.h"
#include "ns3/test.h"
```

Include dependency graph for onion-routing-test-suite.cc:



Classes

- class OnionRoutingTestCase1
- class OnionRoutingTestSuite

Variables

- static OnionRoutingTestSuite sonionRoutingTestSuite

4.9.1 Variable Documentation

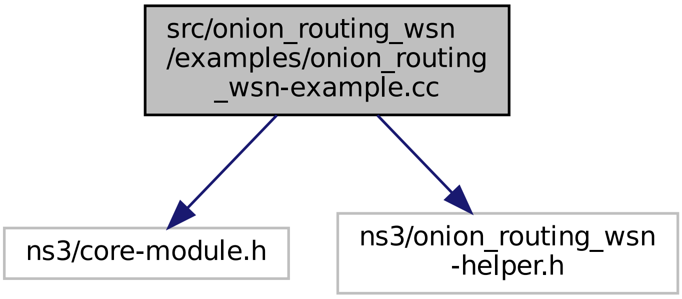
4.9.1.1 sonionRoutingTestSuite `OnionRoutingTestSuite sonionRoutingTestSuite [static]`

Definition at line 67 of file onion-routing-test-suite.cc.

4.10 src/onion_routing_wsn/examples/onion_routing_wsn-example.cc File Reference

```
#include "ns3/core-module.h"
#include "ns3/onion_routing_wsn-helper.h"
```

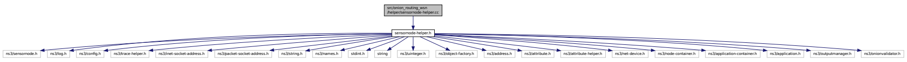
Include dependency graph for onion_routing_wsn-example.cc:



4.11 src/onion_routing_wsn/helper/sensornode-helper.cc File Reference

```
#include "sensornode-helper.h"
```

Include dependency graph for sensornode-helper.cc:



Namespaces

- ns3

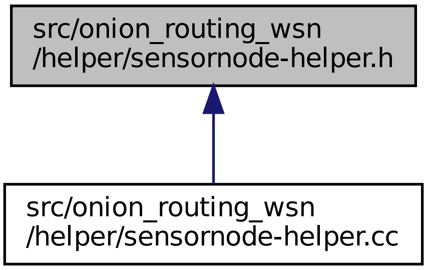
4.12 src/onion_routing_wsn/helper/sensornode-helper.h File Reference

```
#include "ns3/sensornode.h"
#include "ns3/log.h"
#include "ns3/config.h"
#include "ns3/trace-helper.h"
#include "ns3/inet-socket-address.h"
#include "ns3/packet-socket-address.h"
#include "ns3/string.h"
#include "ns3/names.h"
#include <stdint.h>
#include <string>
#include "ns3/uinteger.h"
#include "ns3/object-factory.h"
#include "ns3/address.h"
#include "ns3/attribute.h"
#include "ns3/attribute-helper.h"
#include "ns3/net-device.h"
#include "ns3/node-container.h"
#include "ns3/application-container.h"
#include "ns3/application.h"
#include "ns3/outputmanager.h"
#include "ns3/onionvalidator.h"
```

Include dependency graph for sensornode-helper.h:

src/onion_routing_wsn
/helper/sensornode-helper.h
ns3/sensornode.h
ns3/log.h
ns3/config.h
ns3/trace-helper.h
ns3/inet-socket-address.h
ns3/packet-socket-address.h
ns3/string.h
ns3/names.h
stdint.h
string
ns3/uinteger.h
ns3/object-factory.h
ns3/address.h
ns3/attribute.h
ns3/attribute-helper.h
ns3/net-device.h
ns3/node-container.h
ns3/application-container.h
ns3/application.h
ns3/outputmanager.h
ns3/onionvalidator.h

This graph shows which files directly or indirectly include this file:



Classes

- class ns3::SensorNodeHelper

  Helper class for the creation of SensorNode applications.

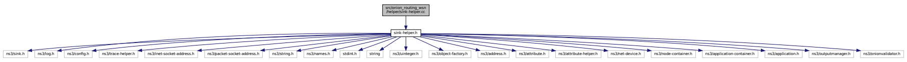
Namespaces

- ns3

4.13 src/onion_routing_wsn/helper/sink-helper.cc File Reference

```
#include "sink-helper.h"
```

Include dependency graph for sink-helper.cc:



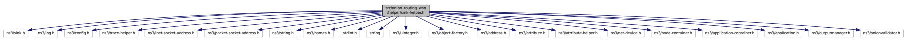
Namespaces

- ns3

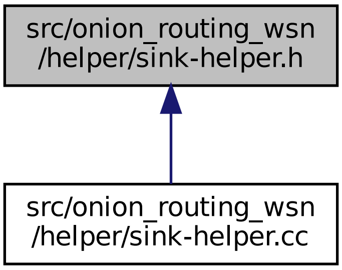
4.14 src/onion_routing_wsn/helper/sink-helper.h File Reference

```
#include "ns3/sink.h"
#include "ns3/log.h"
#include "ns3/config.h"
#include "ns3/trace-helper.h"
#include "ns3/inet-socket-address.h"
#include "ns3/packet-socket-address.h"
#include "ns3/string.h"
#include "ns3/names.h"
#include <stdint.h>
#include <string>
#include "ns3/uinteger.h"
#include "ns3/object-factory.h"
#include "ns3/address.h"
#include "ns3/attribute.h"
#include "ns3/attribute-helper.h"
#include "ns3/net-device.h"
#include "ns3/node-container.h"
#include "ns3/application-container.h"
#include "ns3/application.h"
#include "ns3/outputmanager.h"
#include "ns3/onionvalidator.h"
```

Include dependency graph for sink-helper.h:



This graph shows which files directly or indirectly include this file:



Classes

- class ns3::SinkHelper

  Helper class for the creation of Sink applications.

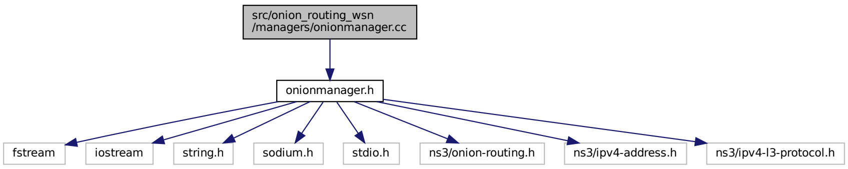
Namespaces

- ns3

4.15 src/onion_routing_wsn/managers/onionmanager.cc File Reference

```
#include "onionmanager.h"
```

Include dependency graph for onionmanager.cc:



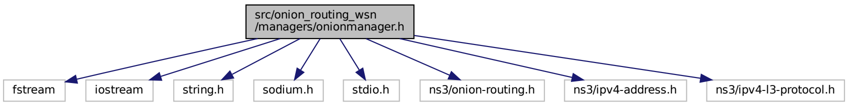
Namespaces

- ns3

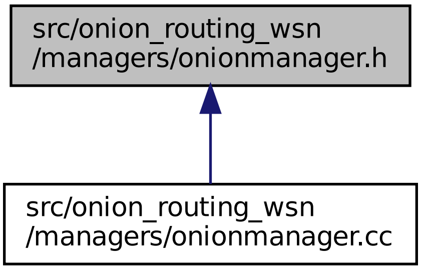
4.16 src/onion_routing_wsn/managers/onionmanager.h File Reference

```
#include <fstream>
#include <iostream>
#include <string.h>
#include <sodium.h>
#include <stdio.h>
#include "ns3/onion-routing.h"
#include "ns3/ipv4-address.h"
#include "ns3/ipv4-l3-protocol.h"
```

Include dependency graph for onionmanager.h:



This graph shows which files directly or indirectly include this file:



Classes

- class ns3::OnionManager

  Class that manages encryption keys and the encryption and decryption of layers of onion messagess The class is implementing the OnionRouting abstract class with the encryption and decryption methods.

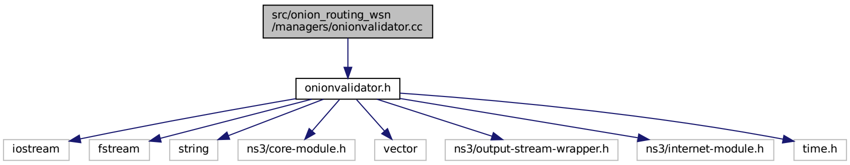
Namespaces

- ns3

4.17 src/onion_routing_wsn/managers/onionvalidator.cc File Reference

```
#include "onionvalidator.h"
```

Include dependency graph for onionvalidator.cc:



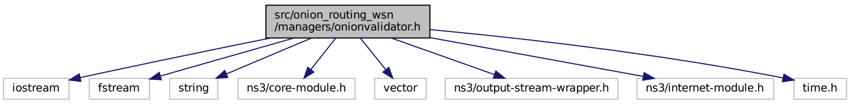
Namespaces

- ns3

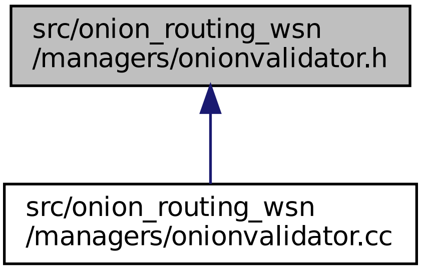
4.18 src/onion_routing_wsn/managers/onionvalidator.h File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include "ns3/core-module.h"
#include <vector>
#include "ns3/output-stream-wrapper.h"
#include "ns3/internet-module.h"
#include <time.h>
```

Include dependency graph for onionvalidator.h:



This graph shows which files directly or indirectly include this file:



Classes

- class ns3::OnionValidator

  Class shared between wsn nodes used to track how the onion is transiting in the WSN The class uses two counters `m_onionSeq` and `m_hopCount` to identify when an onion neeeds to be aborted.

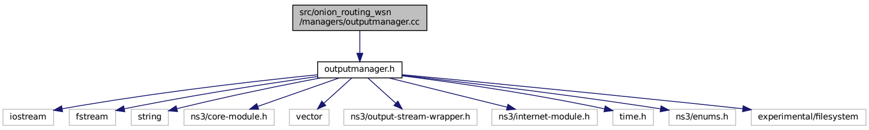
Namespaces

- ns3

4.19 src/onion_routing_wsn/managers/outputmanager.cc File Reference

```
#include "outputmanager.h"
```

Include dependency graph for outputmanager.cc:



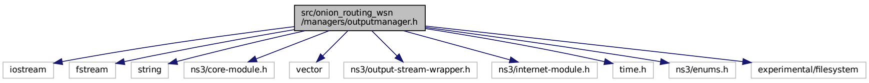
Namespaces

- ns3

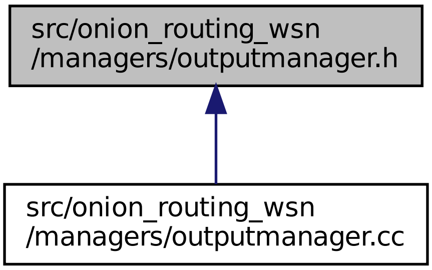
4.20 src/onion_routing_wsn/managers/outputmanager.h File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include "ns3/core-module.h"
#include <vector>
#include "ns3/output-stream-wrapper.h"
#include "ns3/internet-module.h"
#include <time.h>
#include "ns3/enums.h"
#include <experimental/filesystem>
```

Include dependency graph for outputmanager.h:



This graph shows which files directly or indirectly include this file:



Classes

- class ns3::OutputManager

  Class that manages the output of the simulation. ConsoleLog output and Output on .csv file

Namespaces

- ns3

4.21 src/onion_routing_wsn/model/enums.h File Reference

Namespaces

- ns3

Enumerations

- enum ns3::BodyOptions { ns3::NO_Body = 0, ns3::Aggregate, ns3::FixedSize, ns3::AggregateAndFixed }

  Specifies how the onion body must behave.
- enum ns3::IEEE_80211n { ns3::F_24GHz = 0, ns3::F_5GHz }

  Enumeration defining different carrier frequencies of the standard IEEE 802.11n.
- enum ns3::Routing { ns3::AODV = 0, ns3::DSR, ns3::OLSR, ns3::DSDV }

  Enumeration defining different routing algorithms that can be used in the simulator.
- enum ns3::Topology { ns3::GRID = 0, ns3::DISC }

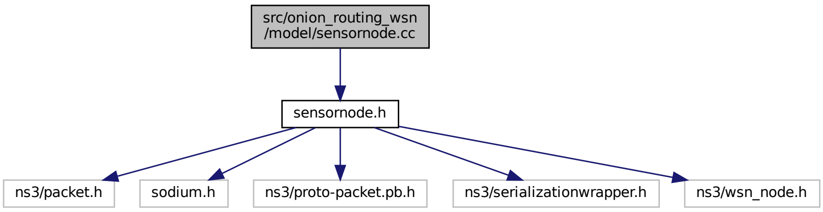
  Enumeration defining different network topologies that can be used in the simulation.
- enum ns3::Verbosity { ns3::NO, ns3::ConsoleLog, ns3::PrintDescription, ns3::Both }

  Verbosity settings of the simulation.

4.22 src/onion_routing_wsn/model/sensornode.cc File Reference

```
#include "sensornode.h"
```

Include dependency graph for sensornode.cc:



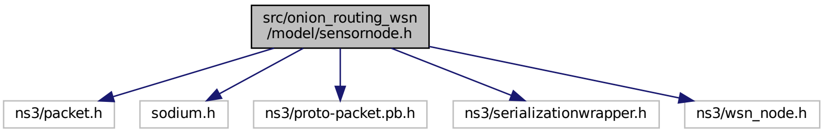
Namespaces

- ns3

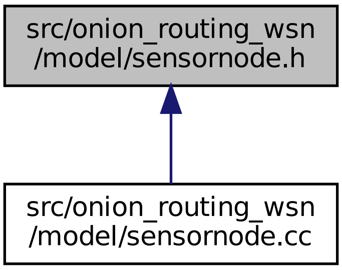
4.23 src/onion_routing_wsn/model/sensornode.h File Reference

```
#include "ns3/packet.h"
#include <sodium.h>
#include "ns3/proto-packet.pb.h"
#include "ns3/serializationwrapper.h"
#include "ns3/wsn_node.h"
```

Include dependency graph for sensornode.h:



This graph shows which files directly or indirectly include this file:



Classes

- class ns3::SensorNode

    The application of the sensor node.

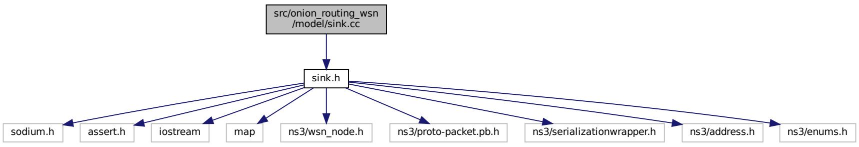
Namespaces

- ns3

4.24 src/onion_routing_wsn/model/sink.cc File Reference

```
#include "sink.h"
```

Include dependency graph for sink.cc:



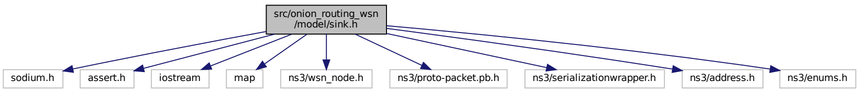
Namespaces

- ns3

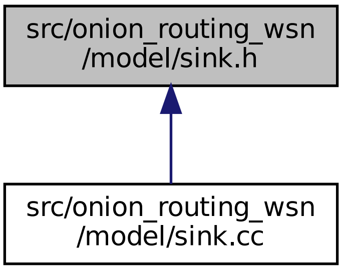
4.25 src/onion_routing_wsn/model/sink.h File Reference

```
#include <sodium.h>
#include <assert.h>
#include <iostream>
#include <map>
#include "ns3/wsn_node.h"
#include "ns3/proto-packet.pb.h"
#include "ns3/serializationwrapper.h"
#include "ns3/address.h"
#include "ns3/enums.h"
```

Include dependency graph for sink.h:



This graph shows which files directly or indirectly include this file:



Classes

- class ns3::Sink

  The application of the sink node. The node that generates onion messagess.

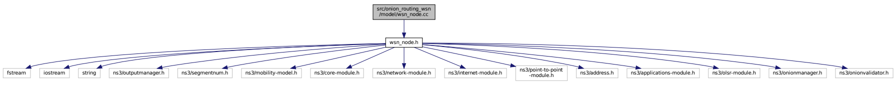
Namespaces

- ns3

4.26 src/onion_routing_wsn/model/wsn_node.cc File Reference

```
#include "wsn_node.h"
```

Include dependency graph for wsn_node.cc:



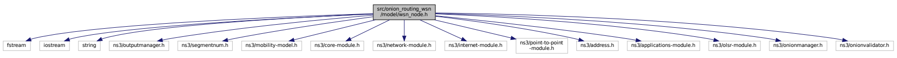
Namespaces

- ns3

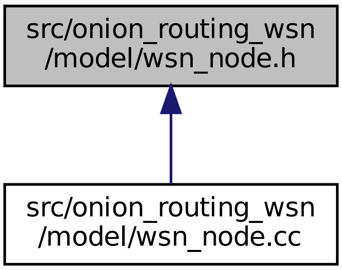
4.27 src/onion_routing_wsn/model/wsn_node.h File Reference

```
#include <fstream>
#include <iostream>
#include <string>
#include "ns3/outputmanager.h"
#include "ns3/segmentnum.h"
#include "ns3/mobility-model.h"
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/address.h"
#include "ns3/applications-module.h"
#include "ns3/olsr-module.h"
#include "ns3/onionmanager.h"
#include "ns3/onionvalidator.h"
```

Include dependency graph for wsn_node.h:



This graph shows which files directly or indirectly include this file:



Classes

- class ns3::Wsn_node

  The wsn node base class that manages the sending and receiving of packets and basic configuration of nodes.

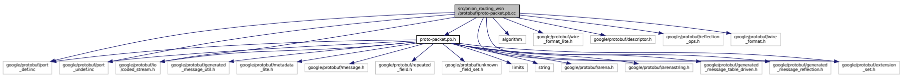
Namespaces

- ns3

4.28 src/onion_routing_wsn/protobuf/proto-packet.pb.cc File Reference

```
#include "proto-packet.pb.h"
#include <algorithm>
#include <google/protobuf/io/coded_stream.h>
#include <google/protobuf/extension_set.h>
#include <google/protobuf/wire_format_lite.h>
#include <google/protobuf/descriptor.h>
#include <google/protobuf/generated_message_reflection.h>
#include <google/protobuf/reflection_ops.h>
#include <google/protobuf/wire_format.h>
#include <google/protobuf/port_def.inc>
#include <google/protobuf/port_undef.inc>
```

Include dependency graph for proto-packet.pb.cc:



Classes

- class protomessage::ProtoPacket::_Internal
- class protomessage::ProtoPacket_Handshake::_Internal
- class protomessage::ProtoPacket_OnionBody::_Internal
- class protomessage::ProtoPacket_OnionHead::_Internal
- struct protomessage::ProtoPacket_HandshakeDefaultTypeInternal
- struct protomessage::ProtoPacket_OnionBodyDefaultTypeInternal
- struct protomessage::ProtoPacket_OnionHeadDefaultTypeInternal
- struct protomessage::ProtoPacketDefaultTypeInternal

Namespaces

- protomessage

Macros

- #define CHK_(x) if (PROTOBUF_PREDICT_FALSE(!(x))) goto failure
- #define CHK_(x) if (PROTOBUF_PREDICT_FALSE(!(x))) goto failure
- #define CHK_(x) if (PROTOBUF_PREDICT_FALSE(!(x))) goto failure
- #define CHK_(x) if (PROTOBUF_PREDICT_FALSE(!(x))) goto failure

Functions

- template<> PROTOBUF_NOINLINE ::protomessage::ProtoPacket ∗ Arena::CreateMaybeMessage< ::protomessage::ProtoPa
(Arena ∗arena)
- template<> PROTOBUF_NOINLINE ::protomessage::ProtoPacket_Handshake ∗ Arena::CreateMaybeMessage< ::protomess
(Arena ∗arena)
- template<> PROTOBUF_NOINLINE ::protomessage::ProtoPacket_OnionBody ∗ Arena::CreateMaybeMessage< ::protomess
(Arena ∗arena)
- template<> PROTOBUF_NAMESPACE_OPEN PROTOBUF_NOINLINE ::protomessage::ProtoPacket_OnionHead ∗ Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_OnionHead > (Arena ∗arena)
- const PROTOBUF_ATTRIBUTE_WEAK ::PROTOBUF_NAMESPACE_ID::internal::DescriptorTable ∗ descriptor_table_proto_2dpacket_2eproto_getter ()
- const uint32_t TableStruct_proto_2dpacket_2eproto::offsets[ ] PROTOBUF_SECTION_VARIABLE (protodesc←
_cold)

Variables

- PROTOBUF_ATTRIBUTE_NO_DESTROY PROTOBUF_CONSTINIT ProtoPacketDefaultTypeInternal protomessage::_ProtoPacket_default_instance_
- PROTOBUF_ATTRIBUTE_NO_DESTROY PROTOBUF_CONSTINIT ProtoPacket_HandshakeDefault↩ TypeInternal protomessage::_ProtoPacket_Handshake_default_instance_
- PROTOBUF_ATTRIBUTE_NO_DESTROY PROTOBUF_CONSTINIT ProtoPacket_OnionBodyDefault↩ TypeInternal protomessage::_ProtoPacket_OnionBody_default_instance_
- PROTOBUF_ATTRIBUTE_NO_DESTROY PROTOBUF_CONSTINIT ProtoPacket_OnionHeadDefault↩ TypeInternal protomessage::_ProtoPacket_OnionHead_default_instance_
- const ::PROTOBUF_NAMESPACE_ID::internal::DescriptorTable descriptor_table_proto_2dpacket_2eproto
- ::PROTOBUF_NAMESPACE_ID::internal::once_flag descriptor_table_proto_2dpacket_2eproto_once
- const ::PROTOBUF_NAMESPACE_ID::Message ∗const file_default_instances [ ]
- static constexpr ::PROTOBUF_NAMESPACE_ID::EnumDescriptor const ∗∗ file_level_enum_descriptors_proto_2dpacket_2epr
  = nullptr
- ::PROTOBUF_NAMESPACE_ID::Metadata file_level_metadata_proto_2dpacket_2eproto [4]
- static constexpr ::PROTOBUF_NAMESPACE_ID::ServiceDescriptor const ∗∗ file_level_service_descriptors_proto_2dpacket_2
  = nullptr

4.28.1 Macro Definition Documentation

4.28.1.1 CHK_ [1/4]

```
#define CHK_(
      x ) if (PROTOBUF_PREDICT_FALSE(!(x))) goto failure
```

4.28.1.2 CHK_ [2/4]

```
#define CHK_(
      x ) if (PROTOBUF_PREDICT_FALSE(!(x))) goto failure
```

4.28.1.3 CHK_ [3/4]

```
#define CHK_(
      x ) if (PROTOBUF_PREDICT_FALSE(!(x))) goto failure
```

4.28.1.4 CHK_ [4/4]

```
#define CHK_(
      x ) if (PROTOBUF_PREDICT_FALSE(!(x))) goto failure
```

4.28.2 Function Documentation

4.28.2.1 Arena::CreateMaybeMessage< ::protomessage::ProtoPacket >() template<>

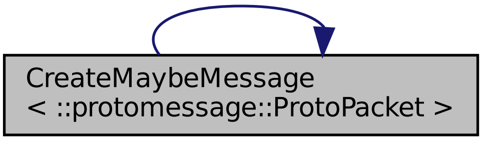
```
PROTOBUF_NOINLINE ::protomessage::ProtoPacket* Arena::CreateMaybeMessage< ::protomessage::ProtoPacket
> (
            Arena * arena )
```

Definition at line 1244 of file proto-packet.pb.cc.

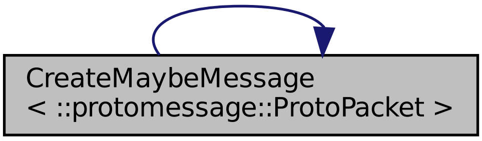
References Arena::CreateMaybeMessage< ::protomessage::ProtoPacket >().

Referenced by Arena::CreateMaybeMessage< ::protomessage::ProtoPacket >().

Here is the call graph for this function:



Here is the caller graph for this function:



4.28.2.2 Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_Handshake >() template<>

```
PROTOBUF_NOINLINE ::protomessage::ProtoPacket_Handshake* Arena::CreateMaybeMessage< ::protomessage::ProtoPacke
> (
            Arena * arena )
```

Definition at line 1241 of file proto-packet.pb.cc.

References Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_Handshake >().

Referenced by Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_Handshake >().

Here is the call graph for this function:

Here is the caller graph for this function:

4.28.2.3 Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_OnionBody >() `template<>`

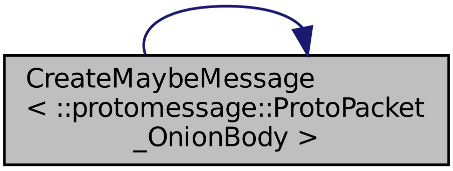
```
PROTOBUF_NOINLINE ::protomessage::ProtoPacket_OnionBody* Arena::CreateMaybeMessage< ::protomessage::ProtoPacke
> (
            Arena * arena )
```

Definition at line 1238 of file proto-packet.pb.cc.

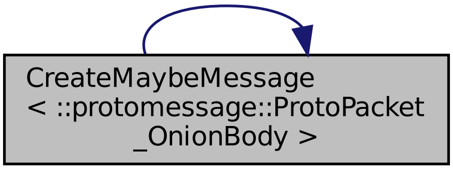
References Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_OnionBody >().

Referenced by Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_OnionBody >().

Here is the call graph for this function:



Here is the caller graph for this function:



4.28.2.4 Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_OnionHead >() template<>

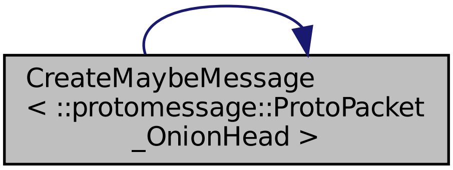
```
PROTOBUF_NAMESPACE_OPEN PROTOBUF_NOINLINE ::protomessage::ProtoPacket_OnionHead* Arena::↩
CreateMaybeMessage< ::protomessage::ProtoPacket_OnionHead > (
            Arena * arena )
```

Definition at line 1235 of file proto-packet.pb.cc.

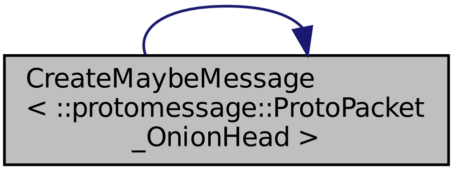
References Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_OnionHead >().

Referenced by Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_OnionHead >().

Here is the call graph for this function:



Here is the caller graph for this function:



4.28.2.5 descriptor_table_proto_2dpacket_2eproto_getter()

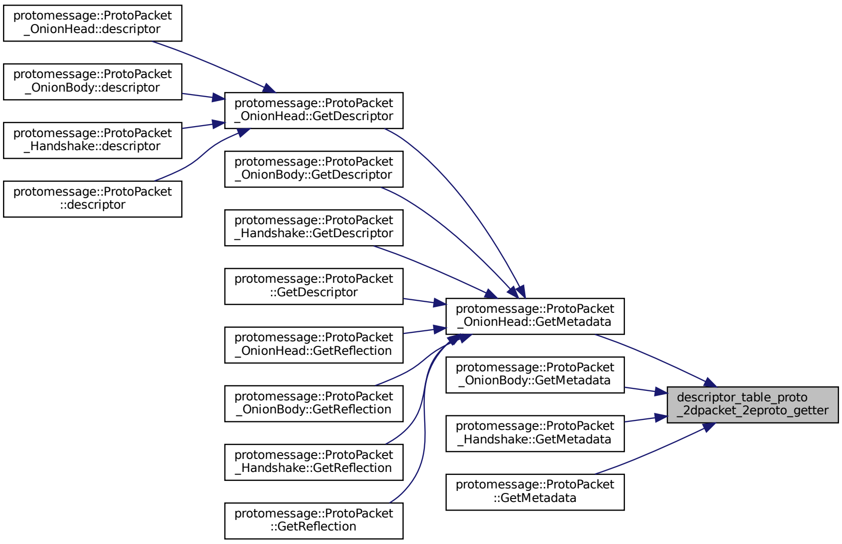
`const PROTOBUF_ATTRIBUTE_WEAK ::PROT↩ OBUF_NAMESPACE_ID::internal::DescriptorTable* descriptor_table_proto_2dpacket_2eproto_getter ( )`

Definition at line 159 of file proto-packet.pb.cc.

References descriptor_table_proto_2dpacket_2eproto.

Referenced by protomessage::ProtoPacket_OnionHead::GetMetadata(), protomessage::ProtoPacket_Onion↩ Body::GetMetadata(), protomessage::ProtoPacket_Handshake::GetMetadata(), and protomessage::ProtoPacket↩ ::GetMetadata().

Here is the caller graph for this function:


protomessage::ProtoPacket _OnionHead::descriptor
protomessage::ProtoPacket _OnionBody::descriptor
protomessage::ProtoPacket _Handshake::descriptor
protomessage::ProtoPacket ::descriptor
protomessage::ProtoPacket _OnionHead::GetDescriptor
protomessage::ProtoPacket _OnionBody::GetDescriptor
protomessage::ProtoPacket _Handshake::GetDescriptor
protomessage::ProtoPacket ::GetDescriptor
protomessage::ProtoPacket _OnionHead::GetReflection
protomessage::ProtoPacket _OnionBody::GetReflection
protomessage::ProtoPacket _Handshake::GetReflection
protomessage::ProtoPacket ::GetReflection
protomessage::ProtoPacket _OnionHead::GetMetadata
protomessage::ProtoPacket _OnionBody::GetMetadata
protomessage::ProtoPacket _Handshake::GetMetadata
protomessage::ProtoPacket ::GetMetadata
descriptor_table_proto _2dpacket_2eproto_getter

4.28.2.6 PROTOBUF_SECTION_VARIABLE()

`const uint32_t TableStruct_proto_2dpacket_2eproto::offsets [ ] PROTOBUF_SECTION_VARIABLE ( protodesc_cold )`

4.28.3 Variable Documentation

4.28.3.1 descriptor_table_proto_2dpacket_2eproto static PROTOBUF_ATTRIBUTE_INIT_PRIORITY::PROTOBUF_NAMESPACE_ID::internal::AddDescriptorsRunner dynamic_init_dummy_proto_2dpacket_2eproto & descriptor_table_proto_2dpacket_2eproto

Initial value:

```
= {
  false, false, 542, descriptor_table_protodef_proto_2dpacket_2eproto, "proto-packet.proto",
  &descriptor_table_proto_2dpacket_2eproto_once, nullptr, 0, 4,
  schemas, file_default_instances, TableStruct_proto_2dpacket_2eproto::offsets,
  file_level_metadata_proto_2dpacket_2eproto, file_level_enum_descriptors_proto_2dpacket_2eproto,
       file_level_service_descriptors_proto_2dpacket_2eproto,
}
```

Definition at line 153 of file proto-packet.pb.cc.

Referenced by descriptor_table_proto_2dpacket_2eproto_getter().

4.28.3.2 descriptor_table_proto_2dpacket_2eproto_once ::PROTOBUF_NAMESPACE_ID::internal::once_flag descriptor_table_proto_2dpacket_2eproto_once [static]

Definition at line 152 of file proto-packet.pb.cc.

Referenced by protomessage::ProtoPacket_OnionHead::GetMetadata(), protomessage::ProtoPacket_OnionBody::GetMetadata(), protomessage::ProtoPacket_Handshake::GetMetadata(), and protomessage::ProtoPacket::GetMetadata().

4.28.3.3 file_default_instances const ::PROTOBUF_NAMESPACE_ID::Message* const file_default_instances[] [static]

Initial value:

```
= {
  reinterpret_cast<const
       ::PROTOBUF_NAMESPACE_ID::Message*>(&::protomessage::_ProtoPacket_OnionHead_default_instance_),
  reinterpret_cast<const
       ::PROTOBUF_NAMESPACE_ID::Message*>(&::protomessage::_ProtoPacket_OnionBody_default_instance_),
  reinterpret_cast<const
       ::PROTOBUF_NAMESPACE_ID::Message*>(&::protomessage::_ProtoPacket_Handshake_default_instance_),
  reinterpret_cast<const
       ::PROTOBUF_NAMESPACE_ID::Message*>(&::protomessage::_ProtoPacket_default_instance_),
}
```

Definition at line 129 of file proto-packet.pb.cc.

4.28.3.4 file_level_enum_descriptors_proto_2dpacket_2eproto constexpr ::PROTOBUF_NAMESPACE_ID::EnumDescriptor const** file_level_enum_descriptors_proto_2dpacket_2eproto = nullptr [static]

Definition at line 75 of file proto-packet.pb.cc.

4.28.3.5 file_level_metadata_proto_2dpacket_2eproto `::PROTOBUF_NAMESPACE_ID::Metadata file_level_metadata_proto_2dpacket_2eproto[4] [static]`

Definition at line 74 of file proto-packet.pb.cc.

Referenced by protomessage::ProtoPacket_OnionHead::GetMetadata(), protomessage::ProtoPacket_OnionBody::GetMetadata(), protomessage::ProtoPacket_Handshake::GetMetadata(), and protomessage::ProtoPacket::GetMetadata().

4.28.3.6 file_level_service_descriptors_proto_2dpacket_2eproto `constexpr ::PROTOBUF_NAMESPACE_ID::ServiceDescriptor const** file_level_service_descriptors_proto_2dpacket_2eproto = nullptr [static]`

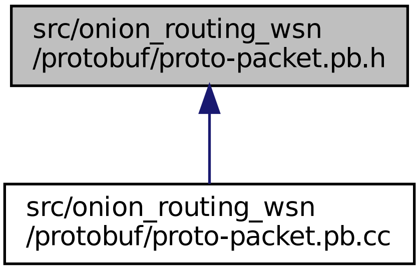
Definition at line 76 of file proto-packet.pb.cc.

4.29 src/onion_routing_wsn/protobuf/proto-packet.pb.h File Reference

```
#include <limits>
#include <string>
#include <google/protobuf/port_def.inc>
#include <google/protobuf/port_undef.inc>
#include <google/protobuf/io/coded_stream.h>
#include <google/protobuf/arena.h>
#include <google/protobuf/arenastring.h>
#include <google/protobuf/generated_message_table_driven.h>
#include <google/protobuf/generated_message_util.h>
#include <google/protobuf/metadata_lite.h>
#include <google/protobuf/generated_message_reflection.h>
#include <google/protobuf/message.h>
#include <google/protobuf/repeated_field.h>
#include <google/protobuf/extension_set.h>
#include <google/protobuf/unknown_field_set.h>
```

Include dependency graph for proto-packet.pb.h:

This graph shows which files directly or indirectly include this file:



Classes

- class protomessage::ProtoPacket
- class protomessage::ProtoPacket_Handshake
- class protomessage::ProtoPacket_OnionBody
- class protomessage::ProtoPacket_OnionHead
- struct TableStruct_proto_2dpacket_2eproto

Namespaces

- internal
- protomessage

Macros

- #define PROTOBUF_INTERNAL_EXPORT_proto_2dpacket_2eproto

Functions

- template<> PROTOBUF_NAMESPACE_OPEN ::protomessage::ProtoPacket ∗ Arena::CreateMaybeMessage< ::protomessag
  (Arena ∗)
- template<> ::protomessage::ProtoPacket_Handshake ∗ Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_Handsl
  (Arena ∗)
- template<> ::protomessage::ProtoPacket_OnionBody ∗ Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_OnionE
  (Arena ∗)
- template<> ::protomessage::ProtoPacket_OnionHead ∗ Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_OnionH
  (Arena ∗)

Variables

- const ::PROTOBUF_NAMESPACE_ID::internal::DescriptorTable descriptor_table_proto_2dpacket_2eproto

4.29.1 Macro Definition Documentation

4.29.1.1 PROTOBUF_INTERNAL_EXPORT_proto_2dpacket_2eproto `#define PROTOBUF_INTERNAL_EXPO↩ RT_proto_2dpacket_2eproto`

Definition at line 36 of file proto-packet.pb.h.

4.29.2 Function Documentation

4.29.2.1 Arena::CreateMaybeMessage< ::protomessage::ProtoPacket >() template<>

```
PROTOBUF_NAMESPACE_OPEN ::protomessage::ProtoPacket* Arena::CreateMaybeMessage<::protomessage::ProtoPacket
> (
            Arena * )
```

4.29.2.2 Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_Handshake >() template<>

```
::protomessage::ProtoPacket_Handshake* Arena::CreateMaybeMessage<::protomessage::ProtoPacket_Handshake
> (
            Arena * )
```

4.29.2.3 Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_OnionBody >() template<>

```
::protomessage::ProtoPacket_OnionBody* Arena::CreateMaybeMessage<::protomessage::ProtoPacket_OnionBody
> (
            Arena * )
```

4.29.2.4 Arena::CreateMaybeMessage< ::protomessage::ProtoPacket_OnionHead >() template<>

```
::protomessage::ProtoPacket_OnionHead* Arena::CreateMaybeMessage<::protomessage::ProtoPacket_OnionHead
> (
            Arena * )
```

4.29.3 Variable Documentation

4.29.3.1 descriptor_table_proto_2dpacket_2eproto const ::PROTOBUF_NAMESPACE_ID::internal::←

```
DescriptorTable descriptor_table_proto_2dpacket_2eproto
```

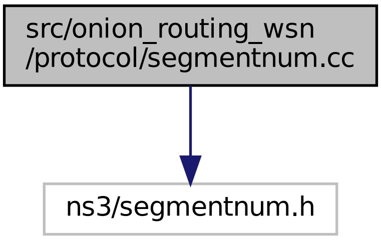
Definition at line 153 of file proto-packet.pb.cc.

Referenced by descriptor_table_proto_2dpacket_2eproto_getter().

4.30 src/onion_routing_wsn/protocol/segmentnum.cc File Reference

```
#include "ns3/segmentnum.h"
```

Include dependency graph for segmentnum.cc:



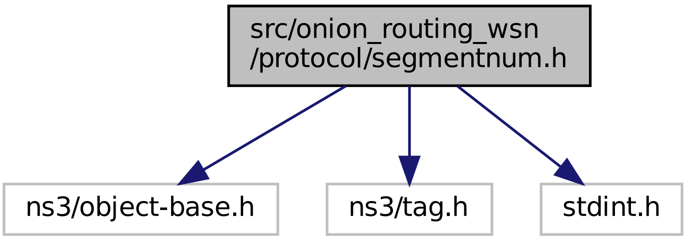
Namespaces

- ns3

4.31 src/onion_routing_wsn/protocol/segmentnum.h File Reference

```
#include "ns3/object-base.h"
#include "ns3/tag.h"
#include <stdint.h>
```

Include dependency graph for segmentnum.h:



Classes

- class ns3::SegmentNum

    Class for adding a tag to packets used to track different segments of packets packets are fragmented due to small MSS Just implementing methods from ns3 class ns3::Tag.

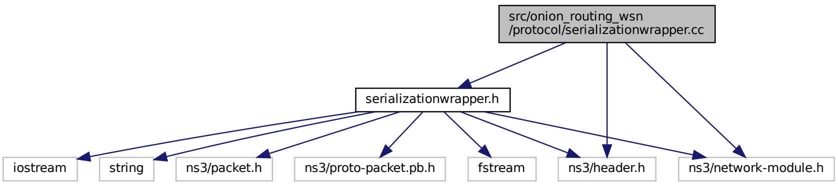
Namespaces

- ns3

4.32 src/onion_routing_wsn/protocol/serializationwrapper.cc File Reference

```
#include "serializationwrapper.h"
#include "ns3/header.h"
#include "ns3/network-module.h"
```

Include dependency graph for serializationwrapper.cc:



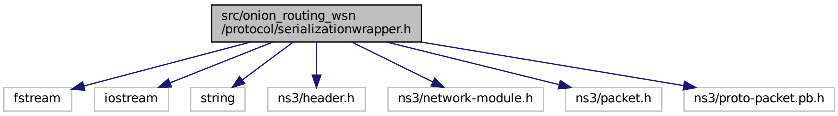
Namespaces

- ns3

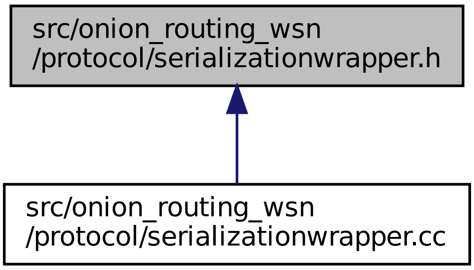
4.33 src/onion_routing_wsn/protocol/serializationwrapper.h File Reference

```
#include <fstream>
#include <iostream>
#include <string>
#include "ns3/header.h"
#include "ns3/network-module.h"
#include "ns3/packet.h"
#include "ns3/proto-packet.pb.h"
```

Include dependency graph for serializationwrapper.h:



This graph shows which files directly or indirectly include this file:


src/onion_routing_wsn
/protocol/serializationwrapper.h
src/onion_routing_wsn
/protocol/serializationwrapper.cc

Classes

- class ns3::SerializationWrapper

  Class for the serialization-deserialization of the messagess to send in packets.

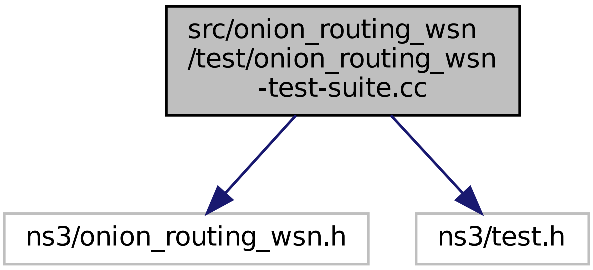
Namespaces

- ns3

4.34 src/onion_routing_wsn/test/onion_routing_wsn-test-suite.cc File Reference

```
#include "ns3/onion_routing_wsn.h"
#include "ns3/test.h"
```

Include dependency graph for onion_routing_wsn-test-suite.cc:



Classes

- class Onion_routing_wsnTestCase1
- class Onion_routing_wsnTestSuite

Variables

- static Onion_routing_wsnTestSuite sonion_routing_wsnTestSuite

4.34.1 Variable Documentation

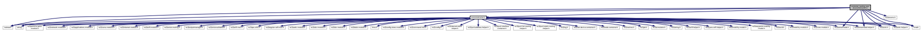
4.34.1.1 sonion_routing_wsnTestSuite `Onion_routing_wsnTestSuite sonion_routing_wsnTestSuite [static]`

Definition at line 67 of file onion_routing_wsn-test-suite.cc.

4.35 src/onion_routing_wsn/wsnconstructor.cc File Reference

```
#include "wsnconstructor.h"
#include "ns3/sensornode.h"
#include "ns3/sensornode-helper.h"
#include "ns3/sink.h"
#include "ns3/sink-helper.h"
#include <iostream>
#include <string>
#include <cmath>
#include "ns3/enums.h"
```

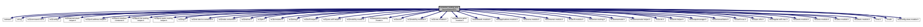
Include dependency graph for wsnconstructor.cc:



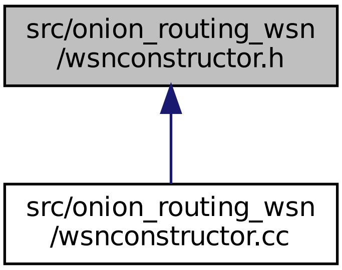
4.36 src/onion_routing_wsn/wsnconstructor.h File Reference

```
#include <iostream>
#include <string>
#include "ns3/command-line.h"
#include "ns3/config.h"
#include "ns3/internet-stack-helper.h"
#include "ns3/ipv4-address-helper.h"
#include "ns3/ipv4-interface-container.h"
#include "ns3/ipv4-list-routing-helper.h"
#include "ns3/ipv4-static-routing-helper.h"
#include "ns3/log.h"
#include "ns3/net-device-container.h"
#include "ns3/node-container.h"
#include "ns3/nstime.h"
#include "ns3/ptr.h"
#include "ns3/simulator.h"
#include "ns3/string.h"
#include "ns3/wifi-helper.h"
#include "ns3/yans-wifi-helper.h"
#include "ns3/mobility-model.h"
#include "ns3/waypoint-mobility-model.h"
#include "ns3/ssid.h"
#include "ns3/mobility-module.h"
#include "ns3/core-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/network-module.h"
#include "ns3/applications-module.h"
#include "ns3/csma-module.h"
#include "ns3/internet-module.h"
#include "ns3/wifi-module.h"
#include "ns3/onion-routing.h"
#include "ns3/sensornode.h"
#include "ns3/sensornode-helper.h"
#include "ns3/sink.h"
#include "ns3/sink-helper.h"
#include "ns3/outputmanager.h"
#include "ns3/onionvalidator.h"
#include "ns3/wifi-mac.h"
#include "ns3/qos-utils.h"
#include "ns3/regular-wifi-mac.h"
#include "ns3/aodv-module.h"
#include "ns3/olsr-module.h"
#include "ns3/dsr-module.h"
#include "ns3/dsdv-module.h"
#include <time.h>
#include <cmath>
#include "ns3/config-store-module.h"
```

Include dependency graph for wsnconstructor.h:



This graph shows which files directly or indirectly include this file:



Classes

- class WsnConstructor

  The class that constructs the WSN, setup applications on nodes and starts the simulation.

Index