
FlexTLS Documentation

Release 0.3

DinoTools

March 07, 2015

1	Features	1
2	Installation	3
2.1	Introduction	3
2.2	Changelog	4
3	API Documentation	5
3.1	Connection	5
3.2	Exceptions	5
3.3	Fields	5
3.4	Helpers	10
3.5	Protocol	10
4	Indices and tables	13
	Python Module Index	15

Features

- Supported cryptographic protocols:
 - SSLv2, SSLv3, TLS 1.0, TLS 1.1 and TLS 1.2
 - DTLS 1.0 and DTLS 1.2
- Decode and encode SSL/TLS/DTLS records
- Handle fragmentation
 - TLS - Handle fragmentation on the record layer
 - DTLS - Handle fragmented handshake messages
- Handle connection state

Installation

You can install FlexTLS with pip:

```
$ pip install flextls
```

See [Introduction](#) for more information.

Contents:

2.1 Introduction

2.1.1 Installation

As a Python egg

You can install the most recent FlexTLS version using pip

```
$ pip install flextls
```

From a tarball release

Download the most recent tarball from github, unpack it and run the following command on the command-line.

```
$ python setup.py install
```

Install the development version

Install git and run the following commands on the command-line.

```
$ git clone https://github.com/DinoTools/python-flextls.git
$ cd python-flextls
$ python setup.py install
```

2.2 Changelog

2.2.1 0.3 - 2015-03-07

- Add support for DTLS 1.0 and DTLS 1.2
- Add support to handle fragmentation on the record layer (TLS)
- Add support to handle fragmented handshake messages (DTLS)
- Add support to handle connection state
- Add support to decode ServerKeyExchange messages
- Change class names for consistent names
- Add additional tests
- Remove deprecated and unused code

2.2.2 0.2 - 2014-11-17

- Add Registry to store global information
 - Add SSL and TLS cipher suites
 - Add named curves
 - Add signature and hash algorithms
- Add support to parse SSLv2 ClientHello and ServerHello packages
- Fixes (Thanks to Till Maas)

2.2.3 0.1 - 2014-10-15

Proof of concept

- Initial release.

API Documentation

3.1 Connection

The class in this python module can be used to handle SSL/TLS/DTLS connections.

class `flextls.connection.BaseConnection` (*protocol_version*)

Base class to handle SSL/TLS/DTLS connections and its state.

class `flextls.connection.BaseDTLSConnection` (*protocol_version*)

Base class for DTLS connections.

class `flextls.connection.BaseTLSConnection` (*protocol_version*)

Class to handle SSL/TLS connections.

class `flextls.connection.DTLsv10Connection` (*protocol_version*)

Class to handle DTLS 1.0 and DTLS 1.2 connections.

class `flextls.connection.SSLv30Connection` (*protocol_version*)

Class to handle SSLv3.0, TLS 1.0, TLS 1.1 and TLS 1.2 connections.

3.2 Exceptions

exception `flextls.exception.NotEnoughData`

Not enough data to decode the next record or field.

exception `flextls.exception.WrongProtocolVersion` (*msg=None*, *record=None*, *protocol_version=None*)

Raised during a connection if the server/client returns a wrong protocol version.

Parameters

- **msg** (*String*) – Message
- **record** (*flextls.protocol.Protocol*) – The decoded record
- **protocol_version** (*Integer*) – Internal ID of the expected protocol version

3.3 Fields

class `flextls.field.CertificateField` (*name='certificate'*)

A certificate.

Parameters **name** (*String*) – The name of the field

class `flextls.field.CertificateListField(name)`
List of certificates

Parameters **name** (*String*) – The name of the field

class `flextls.field.CipherSuiteField(name='unnamed')`
A cipher suite

class `flextls.field.CipherSuitesField(name)`
List of cipher suites.

Parameters **name** (*String*) – The name of the field

class `flextls.field.CompressionMethodField(name='unnamed')`
Compression method

class `flextls.field.CompressionMethodsField(name)`
List of compression methods

Parameters **name** (*String*) – The name of the field

class `flextls.field.ECParametersNamedCurveField(name)`
RFC4492 ECC Cipher Suites for TLS

class `flextls.field.ECPointField(name)`
RFC4492 ECC Cipher Suites for TLS

class `flextls.field.EnumField(name, default, enums, fmt='H')`
The field should only use the defined values.

Parameters

- **name** (*String*) – The name of the field
- **default** (*Mixed*) – A value defined in the enums list
- **enums** (*Dict*) – List of possible values.
- **fmt** (*String*) – The format string

get_value_name (*pretty=False*)
Get the name of the value

Parameters **pretty** (*Boolean*) – Return the name in a pretty format

Returns The name

Return type String

set_value (*value, force=False*)
Set the value.

Parameters

- **value** (*String|Integer*) – The value to set. Must be in the enum list.
- **force** (*Boolean*) – Set the value without checking it

Raises

- **ValueError** – If value name given but it isn't available
- **TypeError** – If value is not String or Integer

value
Return the field value.

Returns The value of the field

Return type Mixed

class flextls.field.**ExtensionsField**(*name*)
List of extensions

Parameters *name* (*String*) – The name of the field

class flextls.field.**Field**(*name*, *default*, *fmt*='H')
Base class for all fields. Used to extract additional information.

Parameters

- **name** (*String*) – Name of the field
- **default** (*Mixed*) – Default field value
- **fmt** (*String*) – Format string used to decode the data

assemble()
Assemble the field by using the given value.

Returns The assembled data

Return type bytes

dissect(*data*)
Dissect the field.

Parameters *data* (*bytes*) – The data to extract the field value from

Returns The rest of the data not used to dissect the field value

Return type bytes

get_value()
Return the field value.

Returns The value of the field

Return type Mixed

set_value(*value*)
Set the value of the field

Parameters *value* (*Mixed*) – The value

value
Return the field value.

Returns The value of the field

Return type Mixed

class flextls.field.**HostNameField**(*name*)
The hostname.

class flextls.field.**MultiPartField**(*name*, *fields*=[])
A field consisting of more than one value.

Parameters

- **name** (*String*) – The name of the field
- **fields** – List of sub fields

class flextls.field.**RandomField**(*name*)
Random data.

class flextls.field.**SSLv2CipherSuiteField**(*name*='unnamed')
A cipher suite for SSLv2

class flextls.field.**ServerDHParamsField**(*name*)
RFC5246 Section 7.4.3. Server Key Exchange Message

class flextls.field.**ServerECDHParamsField**(*name*)
RFC4492 ECC Cipher Suites for TLS

class flextls.field.**ServerNameField**(*name*='test', ***kwargs*)
The server name

class flextls.field.**ServerNameListField**(*name*)
List of server names

Parameters *name* (*String*) – The name of the field

class flextls.field.**SignatureAndHashAlgorithmField**(*name*)
Representing a signature and hash algorithm

class flextls.field.**UInt16EnumField**(*name*, *default*, *enums*)
The field should only use the defined values. The value must be an 16-Bit unsigned integer.

Parameters

- **name** (*String*) – The name of the field
- **default** (*Mixed*) – A value defined in the enums list
- **enums** (*Dict*) – List of possible values.

class flextls.field.**UInt16Field**(*name*, *default*)
Field representing an 16-bit unsigned integer value(range: 0 through 65535 decimal).

class flextls.field.**UInt24Field**(*name*, *default*)
Field representing an 16-bit unsigned integer value.

class flextls.field.**UInt48Field**(*name*, *default*)
Field representing an 48-bit unsigned integer value.

class flextls.field.**UInt8EnumField**(*name*, *default*, *enums*)
The field should only use the defined values. The value must be an 8-Bit unsigned integer.

Parameters

- **name** (*String*) – The name of the field
- **default** (*Mixed*) – A value defined in the enums list
- **enums** (*Dict*) – List of possible values.

class flextls.field.**UInt8Field**(*name*, *default*)
Field representing an 8-bit unsigned integer value(range: 0 through 255 decimal).

class flextls.field.**VectorBaseField**(*name*, *default*='', *fmt*='H', *connection*=None)
A vector as defined by the RFC is a single dimensioned array.

Parameters

- **name** (*String*) – The name of the field
- **default** (*Bytes*) – Default value of the field
- **fmt** (*String*) – The format string of the length identifier

class `flextls.field.VectorInt24Field(name)`

A vector as defined by the RFC is a single dimensioned array. The length identifier of this vector is a 24-bit unsigned integer.

Parameters

- **name** (*String*) – The name of the field
- **fmt** (*String*) – The format string of the length identifier

class `flextls.field.VectorListBaseField(name, item_class=None, item_class_args=None, fmt='H')`

A vector as defined by the RFC is a single dimensioned array.

Parameters

- **name** (*String*) – The name of the field
- **item_class** (*flextls.field.Field*) –
- **item_class_args** (*List*) –
- **fmt** (*String*) – The format string

class `flextls.field.VectorListInt24Field(name, item_class=None, item_class_args=None)`

A vector as defined by the RFC is a single dimensioned array. The length identifier of this vector is a 24-bit unsigned integer.

Parameters

- **name** (*String*) – The name of the field
- **item_class** (*flextls.field.Field*) –
- **item_class_args** (*List*) –
- **fmt** (*String*) – The format string of the length identifier

class `flextls.field.VectorListUInt16Field(name, item_class=None, item_class_args=None)`

A vector as defined by the RFC is a single dimensioned array. The length identifier of this vector is a 16-bit unsigned integer.

Parameters

- **name** (*String*) – The name of the field
- **item_class** (*flextls.field.Field*) –
- **item_class_args** (*List*) –
- **fmt** (*String*) – The format string of the length identifier

class `flextls.field.VectorListUInt8Field(name, item_class=None, item_class_args=None)`

A vector as defined by the RFC is a single dimensioned array. The length identifier of this vector is a 8-bit unsigned integer.

Parameters

- **name** (*String*) – The name of the field
- **item_class** (*flextls.field.Field*) –
- **item_class_args** (*List*) –
- **fmt** (*String*) – The format string of the length identifier

class `flextls.field.VectorUInt16Field(name)`

A vector as defined by the RFC is a single dimensioned array. The length identifier of this vector is a 16-bit unsigned integer.

Parameters

- **name** (*String*) – The name of the field
- **fmt** (*String*) – The format string of the length identifier

class `flextls.field.VectorUInt8Field(name)`

A vector as defined by the RFC is a single dimensioned array. The length identifier of this vector is a 8-bit unsigned integer.

Parameters

- **name** (*String*) – The name of the field
- **fmt** (*String*) – The format string of the length identifier

class `flextls.field.VersionField(name)`

The protocol version field.

Parameters **name** (*String*) – Name of the field

3.4 Helpers

`flextls.helper.get_version_by_version_id(version_id)`

Get the internal version ID be the version.

Parameters **version_id** (*Tuple*) – Major and minor version number

Returns Internal version ID

Return type `Integer!None`

`flextls.helper.get_version_id(protocol_version)`

Get a tuple with major and minor version number

Parameters **protocol_version** (*Integer*) – Internal version ID

Returns Tuple of major and minor protocol version

Return type `Tuple`

`flextls.helper.get_version_name(version_id)`

Get the name of a protocol version by the internal version ID.

Parameters **version_id** (*Integer*) – Internal protocol version ID

Returns Name of the version

Return type `String`

3.5 Protocol

The SSL/TLS Protocol

class `flextls.protocol.Protocol(connection=None)`

Base Class to decode protocols.

3.5.1 Alert

```
class flextls.protocol.alert.Alert(**kwargs)
    Handle Alert protocol
    •RFC5246 (Section 7.2)
```

3.5.2 Change Cipher Spec

```
class flextls.protocol.change_cipher_spec.ChangeCipherSpec(**kwargs)
    Handle Change Cipher Spec Protocol
```

3.5.3 Handshake

The SSL/TLS Handshake Protocol

```
class flextls.protocol.handshake.ClientHello(**kwargs)
    Handle SSLv3 and TLS 1.0, 1.1 and 1.2 Client Hello messages

class flextls.protocol.handshake.ClientKeyExchange(**kwargs)
    Handle SSLv3 and TLS 1.0, 1.1 and 1.2 and DLTS 1.0 and 1.2 Client Key Exchange messages

class flextls.protocol.handshake.DTLSTLSv10ClientHello(**kwargs)
    Handle DTLS 1.0 and 1.2 Client Hello messages

class flextls.protocol.handshake.DTLSTLSv10Handshake(**kwargs)
    Handle DTLS 1.0 and 1.2 Handshake protocol

class flextls.protocol.handshake.DTLSTLSv10HelloVerifyRequest(**kwargs)
    Handle DTLS 1.0 and 1.2 Hello Verify Request messages

class flextls.protocol.handshake.Handshake(**kwargs)
    Handle SSLv3 and TLS 1.0, 1.1 and 1.2 Handshake protocol

class flextls.protocol.handshake.SSLv2ClientHello(**kwargs)
    Handle SSLv2 Client Hello messages

class flextls.protocol.handshake.SSLv2ServerHello(**kwargs)
    Handle SSLv2 Server Hello messages

class flextls.protocol.handshake.ServerCertificate(**kwargs)
    Handle SSLv3 and TLS 1.0, 1.1 and 1.2 and DLTS 1.0 and 1.2 Server Certificate messages

class flextls.protocol.handshake.ServerHello(**kwargs)
    Handle SSLv3 and TLS 1.0, 1.1 and 1.2 Server Hello messages

class flextls.protocol.handshake.ServerHelloDone(**kwargs)
    Handle SSLv3 and TLS 1.0, 1.1 and 1.2 and DLTS 1.0 and 1.2 Server Hello Done messages

class flextls.protocol.handshake.ServerKeyExchange(**kwargs)
    Handle SSLv3 and TLS 1.0, 1.1 and 1.2 and DLTS 1.0 and 1.2 Server Key Exchange messages

class flextls.protocol.handshake.extension.ECPointFormats(**kwargs)
    Handle Elliptic Curves Point Format extension

class flextls.protocol.handshake.extension.EllipticCurves(**kwargs)
    Handle Elliptic Curves extension

class flextls.protocol.handshake.extension.Extension(**kwargs)
    Handle TLS and DTLS Extensions
```

```
class flextls.protocol.handshake.extension.Heartbeat (**kwargs)
    Handle Heartbeat extension

class flextls.protocol.handshake.extension.ServerNameIndication (**kwargs)
    Handle Server Name Indication extension
        •RFC6066 (Section 3)

class flextls.protocol.handshake.extension.SessionTicketTLS (**kwargs)
    Handle Session Ticket extension

class flextls.protocol.handshake.extension.SignatureAlgorithms (**kwargs)
    Handle Signature Algorithm extension
```

3.5.4 Heartbeat

```
class flextls.protocol.heartbeat.Heartbeat (**kwargs)
    Handle Heartbeat Request and Response Messages
        •RFC6520
```

3.5.5 Record

The SSL/TLS Record Protocol

```
class flextls.protocol.record.DTLSv10Record (**kwargs)
    Handle DTLS 1.0 and DTLS 1.2 Record layer.

class flextls.protocol.record.SSLv2Record (**kwargs)
    Handle the SSLv2 Record layer.

class flextls.protocol.record.SSLv3Record (**kwargs)
    Handle the SSLv3 and TLS 1.0, 1.1 and 1.2 Record layer
```

Indices and tables

- *genindex*
- *modindex*
- *search*

f

- `flextls.connection`, [5](#)
- `flextls.exception`, [5](#)
- `flextls.field`, [5](#)
- `flextls.helper`, [10](#)
- `flextls.protocol`, [10](#)
- `flextls.protocol.alert`, [11](#)
- `flextls.protocol.change_cipher_spec`, [11](#)
- `flextls.protocol.handshake`, [11](#)
- `flextls.protocol.handshake.extension`,
[11](#)
- `flextls.protocol.heartbeat`, [12](#)
- `flextls.protocol.record`, [12](#)

A

Alert (class in flextls.protocol.alert), 11
assemble() (flextls.field.Field method), 7

B

BaseConnection (class in flextls.connection), 5
BaseDTLSConnection (class in flextls.connection), 5
BaseTLSConnection (class in flextls.connection), 5

C

CertificateField (class in flextls.field), 5
CertificateListField (class in flextls.field), 6
ChangeCipherSpec (class
flextls.protocol.change_cipher_spec), 11
CipherSuiteField (class in flextls.field), 6
CipherSuitesField (class in flextls.field), 6
ClientHello (class in flextls.protocol.handshake), 11
ClientKeyExchange (class in flextls.protocol.handshake),
11
CompressionMethodField (class in flextls.field), 6
CompressionMethodsField (class in flextls.field), 6

D

dissect() (flextls.field.Field method), 7
DTLSv10ClientHello (class
flextls.protocol.handshake), 11
DTLSv10Connection (class in flextls.connection), 5
DTLSv10Handshake (class
flextls.protocol.handshake), 11
DTLSv10HelloVerifyRequest (class
flextls.protocol.handshake), 11
DTLSv10Record (class in flextls.protocol.record), 12

E

ECParametersNamedCurveField (class in flextls.field), 6
ECPointField (class in flextls.field), 6
EcPointFormats (class
flextls.protocol.handshake.extension), 11
EllipticCurves (class
flextls.protocol.handshake.extension), 11

EnumField (class in flextls.field), 6
Extension (class in flextls.protocol.handshake.extension),
11
ExtensionsField (class in flextls.field), 7

F

Field (class in flextls.field), 7
flextls.connection (module), 5
flextls.exception (module), 5
flextls.field (module), 5
flextls.helper (module), 10
flextls.protocol (module), 10
flextls.protocol.alert (module), 11
flextls.protocol.change_cipher_spec (module), 11
flextls.protocol.handshake (module), 11
flextls.protocol.handshake.extension (module), 11
flextls.protocol.heartbeat (module), 12
flextls.protocol.record (module), 12

G

get_value() (flextls.field.Field method), 7
get_value_name() (flextls.field.EnumField method), 6
get_version_by_version_id() (in module flextls.helper),
10
get_version_id() (in module flextls.helper), 10
get_version_name() (in module flextls.helper), 10

H

Handshake (class in flextls.protocol.handshake), 11
Heartbeat (class in flextls.protocol.handshake.extension),
11
Heartbeat (class in flextls.protocol.heartbeat), 12
HostNameField (class in flextls.field), 7

M

MultiPartField (class in flextls.field), 7

N

NotEnoughData, 5

P

Protocol (class in flextls.protocol), 10

R

RandomField (class in flextls.field), 7

S

ServerCertificate (class in flextls.protocol.handshake), 11

ServerDHPParamsField (class in flextls.field), 8

ServerECDHParamsField (class in flextls.field), 8

ServerHello (class in flextls.protocol.handshake), 11

ServerHelloDone (class in flextls.protocol.handshake), 11

ServerKeyExchange (class in flextls.protocol.handshake),
11

ServerNameField (class in flextls.field), 8

ServerNameIndication (class in
flextls.protocol.handshake.extension), 12

ServerNameListField (class in flextls.field), 8

SessionTicketTLS (class in
flextls.protocol.handshake.extension), 12

set_value() (flextls.field.EnumField method), 6

set_value() (flextls.field.Field method), 7

SignatureAlgorithms (class in
flextls.protocol.handshake.extension), 12

SignatureAndHashAlgorithmField (class in flextls.field),
8

SSLv2CipherSuiteField (class in flextls.field), 8

SSLv2ClientHello (class in flextls.protocol.handshake),
11

SSLv2Record (class in flextls.protocol.record), 12

SSLv2ServerHello (class in flextls.protocol.handshake),
11

SSLv30Connection (class in flextls.connection), 5

SSLv3Record (class in flextls.protocol.record), 12

U

UInt16EnumField (class in flextls.field), 8

UInt16Field (class in flextls.field), 8

UInt24Field (class in flextls.field), 8

UInt48Field (class in flextls.field), 8

UInt8EnumField (class in flextls.field), 8

UInt8Field (class in flextls.field), 8

V

value (flextls.field.EnumField attribute), 6

value (flextls.field.Field attribute), 7

VectorBaseField (class in flextls.field), 8

VectorInt24Field (class in flextls.field), 8

VectorListBaseField (class in flextls.field), 9

VectorListInt24Field (class in flextls.field), 9

VectorListUInt16Field (class in flextls.field), 9

VectorListUInt8Field (class in flextls.field), 9

VectorUInt16Field (class in flextls.field), 9

VectorUInt8Field (class in flextls.field), 10

VersionField (class in flextls.field), 10

W

WrongProtocolVersion, 5