

---

# **Aries Cloud Agent Python Documentation**

**Nicholas Rempel, Andrew Whitehead, Ian Costanzo, Stephen Klur**

**Apr 21, 2020**



---

## Aries Cloud Agent Python - Modules

---

<b>1</b>	<b>aries_cloudagent package</b>	<b>3</b>
1.1	Subpackages	3
1.1.1	aries_cloudagent.admin package	3
1.1.2	aries_cloudagent.cache package	6
1.1.3	aries_cloudagent.commands package	7
1.1.4	aries_cloudagent.config package	9
1.1.5	aries_cloudagent.core package	18
1.1.6	aries_cloudagent.holder package	24
1.1.7	aries_cloudagent.issuer package	27
1.1.8	aries_cloudagent.ledger package	31
1.1.9	aries_cloudagent.messaging package	39
1.1.10	aries_cloudagent.storage package	75
1.1.11	aries_cloudagent.transport package	83
1.1.12	aries_cloudagent.utils package	102
1.1.13	aries_cloudagent.verifier package	108
1.1.14	aries_cloudagent.wallet package	109
1.2	Submodules	126
1.3	aries_cloudagent.version module	126
<b>2</b>	<b>aries_cloudagent.connections package</b>	<b>127</b>
2.1	Subpackages	127
2.1.1	aries_cloudagent.connections.models package	127
<b>3</b>	<b>aries_cloudagent.protocols package</b>	<b>143</b>
3.1	Subpackages	143
3.1.1	aries_cloudagent.protocols.actionmenu package	143
3.1.2	aries_cloudagent.protocols.basicmessage package	144
3.1.3	aries_cloudagent.protocols.connections package	144
3.1.4	aries_cloudagent.protocols.credentials package	145
3.1.5	aries_cloudagent.protocols.discovery package	146
3.1.6	aries_cloudagent.protocols.introduction package	147
3.1.7	aries_cloudagent.protocols.issue_credential package	147
3.1.8	aries_cloudagent.protocols.present_proof package	170
3.1.9	aries_cloudagent.protocols.presentations package	191
3.1.10	aries_cloudagent.protocols.problem_report package	191
3.1.11	aries_cloudagent.protocols.routing package	192
3.1.12	aries_cloudagent.protocols.trustping package	193

<b>4 Indices and tables</b>	<b>195</b>
<b>Python Module Index</b>	<b>197</b>
<b>Index</b>	<b>201</b>

Hyperledger Aries Cloud Agent Python (ACA-Py) is a foundation for building decentralized identity applications and services running in non-mobile environments.

This is the Read The Docs site for the Hyperledger [Aries Cloud Agent Python](#). This site contains only the ACA-Py docstrings documentation extracted from the Python Code. For other documentation, please consult the links in the Readme for the [ACA-Py GitHub Repo](#).

If you are getting started with verifiable credentials or Aries, we recommend that you start with this [verifiable credentials and agents getting started guide](#).

Want to quick overview of the deployment model for ACA-Py? See [this document](#).

To investigate the code, use search or click the package links in the left menu to drill into the modules, subpackages and submodules that make up ACA-Py.

Developers that are interested in what DIDComm protocols are supported in ACA-Py should take a look at the [protocols](#) package. These should align with the corresponding [aries-rfcs protocols](#). Decorators defined in aries-rfcs and implemented in ACA-Py can be found [here](#). Some general purpose subpackages that might be of interest include [wallet](#) and [storage](#). For those agents playing different roles in a verifiable credential exchange, take a look at the [issuer](#), [holder](#) and [verifier](#) packages.

Please see the [ACA-Py Contribution guidelines](#) for how to contribute to ACA-Py, including for how to submit issues about ACA-Py.



---

## aries\_cloudagent package

---

Aries Cloud Agent.

## 1.1 Subpackages

### 1.1.1 aries\_cloudagent.admin package

#### Submodules

#### aries\_cloudagent.admin.base\_server module

Abstract admin server interface.

```
class aries_cloudagent.admin.base_server.BaseAdminServer
    Bases: abc.ABC
```

Admin HTTP server class.

```
add_webhook_target (target_url: str, topic_filter: Sequence[str] = None, max_attempts: int =
                    None)
    Add a webhook target.
```

```
remove_webhook_target (target_url: str)
    Remove a webhook target.
```

```
send_webhook (topic: str, payload: dict)
    Add a webhook to the queue, to send to all registered targets.
```

```
start () → None
    Start the webserver.
```

**Raises** AdminSetupError – If there was an error starting the webserver

```
stop () → None
    Stop the webserver.
```

## aries\_cloudagent.admin.error module

Admin error classes.

**exception** aries\_cloudagent.admin.error.**AdminError**(\*args, error\_code: str = None, \*\*kwargs)

Bases: *aries\_cloudagent.core.error.BaseError*

Base class for Admin-related errors.

**exception** aries\_cloudagent.admin.error.**AdminSetupError**(\*args, error\_code: str = None, \*\*kwargs)

Bases: *aries\_cloudagent.admin.error.AdminError*

Admin server setup or configuration error.

## aries\_cloudagent.admin.server module

Admin server classes.

**class** aries\_cloudagent.admin.server.**AdminModulesSchema**(\*args, \*\*kwargs)

Bases: sphinx.ext.autodoc.importer.\_MockObject

Schema for the modules endpoint.

**result**

Used by autodoc\_mock\_imports.

**class** aries\_cloudagent.admin.server.**AdminResponder**(context: *aries\_cloudagent.config.injection\_context.InjectionContext*, send: *Coroutine[T\_co, T\_contra, V\_co]*, webhook: *Coroutine[T\_co, T\_contra, V\_co]*, \*\*kwargs)

Bases: *aries\_cloudagent.messaging.responder.BaseResponder*

Handle outgoing messages from message handlers.

**send\_outbound**(message: *aries\_cloudagent.transport.outbound.message.OutboundMessage*)

Send outbound message.

**Parameters** **message** – The *OutboundMessage* to be sent

**send\_webhook**(topic: str, payload: dict)

Dispatch a webhook.

**Parameters**

- **topic** – the webhook topic identifier
- **payload** – the webhook payload value

**class** aries\_cloudagent.admin.server.**AdminServer**(host: str, port: int, context: *aries\_cloudagent.config.injection\_context.InjectionContext*, outbound\_message\_router: *Coroutine[T\_co, T\_contra, V\_co]*, webhook\_router: *Callable*, task\_queue: *aries\_cloudagent.utils.task\_queue.TaskQueue* = None, conductor\_stats: *Coroutine[T\_co, T\_contra, V\_co]* = None)

Bases: *aries\_cloudagent.admin.base\_server.BaseAdminServer*

Admin HTTP server class.



**add\_webhook\_target** (*target\_url: str, topic\_filter: Sequence[str] = None, max\_attempts: int = None*)  
Add a webhook target.

**make\_application** () → *<sphinx.ext.autodoc.importer.\_MockObject object at 0x7f7951f37f60>*  
Get the aiohttp application instance.

**on\_startup** (*app: <sphinx.ext.autodoc.importer.\_MockObject object at 0x7f7951f37f60>*)  
Perform webserver startup actions.

**plugins\_handler** (*request: <sphinx.ext.autodoc.importer.\_MockObject object at 0x7f7951f37f60>*)  
Request handler for the loaded plugins list.

**Parameters request** – aiohttp request object

**Returns** The module list response

**redirect\_handler** (*request: <sphinx.ext.autodoc.importer.\_MockObject object at 0x7f7951f37f60>*)  
Perform redirect to documentation.

**remove\_webhook\_target** (*target\_url: str*)  
Remove a webhook target.

**send\_webhook** (*topic: str, payload: dict*)  
Add a webhook to the queue, to send to all registered targets.

**start** () → None  
Start the webserver.

**Raises** `AdminSetupError` – If there was an error starting the webserver

**status\_handler** (*request: <sphinx.ext.autodoc.importer.\_MockObject object at 0x7f7951f37f60>*)  
Request handler for the server status information.

**Parameters request** – aiohttp request object

**Returns** The web response

**status\_reset\_handler** (*request: <sphinx.ext.autodoc.importer.\_MockObject object at 0x7f7951f37f60>*)  
Request handler for resetting the timing statistics.

**Parameters request** – aiohttp request object

**Returns** The web response

**stop** () → None  
Stop the webserver.

**websocket\_handler** (*request*)  
Send notifications to admin client over websocket.

**class** `aries_cloudagent.admin.server.AdminStatusSchema` (*\*args, \*\*kwargs*)  
Bases: `sphinx.ext.autodoc.importer._MockObject`  
Schema for the status endpoint.

**class** `aries_cloudagent.admin.server.WebhookTarget` (*endpoint: str, topic\_filter: Sequence[str] = None, max\_attempts: int = None*)  
Bases: `object`  
Class for managing webhook target information.

**topic\_filter**

Accessor for the target's topic filter.

## 1.1.2 aries\_cloudagent.cache package

### Submodules

#### aries\_cloudagent.cache.base module

Abstract base classes for cache.

**class** aries\_cloudagent.cache.base.BaseCache

Bases: `abc.ABC`

Abstract cache interface.

**acquire** (*key: str*)

Acquire a lock on a given cache key.

**clear** (*key: str*)

Remove an item from the cache, if present.

**Parameters** **key** – the key to remove

**flush** ()

Remove all items from the cache.

**get** (*key: str*)

Get an item from the cache.

**Parameters** **key** – the key to retrieve an item for

**Returns** The record found or *None*

**release** (*key: str*)

Release the lock on a given cache key.

**set** (*keys: Union[str, Sequence[str]], value: Any, ttl: int = None*)

Add an item to the cache with an optional ttl.

**Parameters**

- **keys** – the key or keys for which to set an item
- **value** – the value to store in the cache
- **ttl** – number of second that the record should persist

**exception** aries\_cloudagent.cache.base.CacheError (*\*args, error\_code: str = None, \*\*kwargs*)

Bases: `aries_cloudagent.core.error.BaseError`

Base class for cache-related errors.

**class** aries\_cloudagent.cache.base.CacheKeyLock (*cache: aries\_cloudagent.cache.base.BaseCache, key: str*)

Bases: `object`

A lock on a particular cache key.

Used to prevent multiple async threads from generating or querying the same semi-expensive data. Not thread safe.

**done**  
Accessor for the done state.

**future**  
Fetch the result in the form of an awaitable future.

**parent**  
Accessor for the parent key lock, if any.

**release()**  
Release the cache lock.

**result**  
Fetch the current result, if any.

**set\_result** (*value: Any, ttl: int = None*)  
Set the result, updating the cache and any waiters.

### **aries\_cloudagent.cache.basic module**

Basic in-memory cache implementation.

**class** `aries_cloudagent.cache.basic.BasicCache`  
Bases: `aries_cloudagent.cache.base.BaseCache`  
Basic in-memory cache class.

**clear** (*key: str*)  
Remove an item from the cache, if present.  
**Parameters** **key** – the key to remove

**flush** ()  
Remove all items from the cache.

**get** (*key: str*)  
Get an item from the cache.  
**Parameters** **key** – the key to retrieve an item for  
**Returns** The record found or *None*

**set** (*keys: Union[str, Sequence[str]], value: Any, ttl: int = None*)  
Add an item to the cache with an optional ttl.  
Overwrites existing cache entries.  
**Parameters**

- **keys** – the key or keys for which to set an item
- **value** – the value to store in the cache
- **ttl** – number of seconds that the record should persist

### **1.1.3 aries\_cloudagent.commands package**

Commands module common setup.

`aries_cloudagent.commands.available_commands()`  
Index available commands.

`aries_cloudagent.commands.load_command(command: str)`

Load the module corresponding with a named command.

`aries_cloudagent.commands.run_command(command: str, argv: Sequence[str] = None)`

Execute a named command with command line arguments.

### Submodules

#### `aries_cloudagent.commands.help` module

Help command for indexing available commands.

`aries_cloudagent.commands.help.execute(argv: Sequence[str] = None)`

Execute the help command.

#### `aries_cloudagent.commands.provision` module

Provision command for setting up agent settings before starting.

**exception** `aries_cloudagent.commands.provision.ProvisionError(*args, error_code: str = None, **kwargs)`

Bases: `aries_cloudagent.core.error.BaseError`

Base exception for provisioning errors.

`aries_cloudagent.commands.provision.execute(argv: Sequence[str] = None)`

Entrypoint.

`aries_cloudagent.commands.provision.init_argument_parser(parser: parse.ArgumentParser)`

Initialize an argument parser with the module's arguments.

`aries_cloudagent.commands.provision.provision(settings: dict)`

Perform provisioning.

#### `aries_cloudagent.commands.start` module

Entrypoint.

`aries_cloudagent.commands.start.execute(argv: Sequence[str] = None)`

Entrypoint.

`aries_cloudagent.commands.start.init_argument_parser(parser: parse.ArgumentParser)`

Initialize an argument parser with the module's arguments.

`aries_cloudagent.commands.start.run_loop(startup: Coroutine[T_co, T_contra, V_co], shutdown: Coroutine[T_co, T_contra, V_co])`

Execute the application, handling signals and ctrl-c.

`aries_cloudagent.commands.start.shutdown_app(conductor: aries_cloudagent.core.conductor.Conductor)`

Shut down.

`aries_cloudagent.commands.start.start_app(conductor: aries_cloudagent.core.conductor.Conductor)`

Start up.

## 1.1.4 aries\_cloudagent.config package

### Submodules

#### aries\_cloudagent.config.argparse module

Command line option parsing.

```
class aries_cloudagent.config.argparse.AdminGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup
    Admin server settings.

    CATEGORIES = ('start',)
    GROUP_NAME = 'Admin'

    add_arguments(parser: argparse.ArgumentParser)
        Add admin-specific command line arguments to the parser.

    get_settings(args: argparse.Namespace)
        Extract admin settings.
```

```
class aries_cloudagent.config.argparse.ArgumentGroup
    Bases: abc.ABC
    A class representing a group of related command line arguments.

    GROUP_NAME = None

    add_arguments(parser: argparse.ArgumentParser)
        Add arguments to the provided argument parser.

    get_settings(args: argparse.Namespace) → dict
        Extract settings from the parsed arguments.
```

```
class aries_cloudagent.config.argparse.DebugGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup
    Debug settings.

    CATEGORIES = ('start',)
    GROUP_NAME = 'Debug'

    add_arguments(parser: argparse.ArgumentParser)
        Add debug command line arguments to the parser.

    get_settings(args: argparse.Namespace) → dict
        Extract debug settings.
```

```
class aries_cloudagent.config.argparse.GeneralGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup
    General settings.

    CATEGORIES = ('general', 'start')
    GROUP_NAME = 'General'

    add_arguments(parser: argparse.ArgumentParser)
        Add general command line arguments to the parser.

    get_settings(args: argparse.Namespace) → dict
        Extract general settings.
```

```
class aries_cloudagent.config.argparse.LedgerGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup

    Ledger settings.

    CATEGORIES = ('start', 'general')

    GROUP_NAME = 'Ledger'

    add_arguments(parser: argparse.ArgumentParser)
        Add ledger-specific command line arguments to the parser.

    get_settings(args: argparse.Namespace) → dict
        Extract ledger settings.

class aries_cloudagent.config.argparse.LoggingGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup

    Logging settings.

    CATEGORIES = ('general', 'start')

    GROUP_NAME = 'Logging'

    add_arguments(parser: argparse.ArgumentParser)
        Add logging-specific command line arguments to the parser.

    get_settings(args: argparse.Namespace) → dict
        Extract logging settings.

class aries_cloudagent.config.argparse.ProtocolGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup

    Protocol settings.

    CATEGORIES = ('start',)

    GROUP_NAME = 'Protocol'

    add_arguments(parser: argparse.ArgumentParser)
        Add protocol-specific command line arguments to the parser.

    get_settings(args: argparse.Namespace) → dict
        Get protocol settings.

class aries_cloudagent.config.argparse.TransportGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup

    Transport settings.

    CATEGORIES = ('start',)

    GROUP_NAME = 'Transport'

    add_arguments(parser: argparse.ArgumentParser)
        Add transport-specific command line arguments to the parser.

    get_settings(args: argparse.Namespace)
        Extract transport settings.

class aries_cloudagent.config.argparse.WalletGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup

    Wallet settings.

    CATEGORIES = ('general', 'start')
```

**GROUP\_NAME** = 'Wallet'

**add\_arguments** (*parser: argparse.ArgumentParser*)  
Add wallet-specific command line arguments to the parser.

**get\_settings** (*args: argparse.Namespace*) → dict  
Extract wallet settings.

**class** aries\_cloudagent.config.argparse.**group** (\**categories*)  
Bases: `object`

Decorator for registering argument groups.

**classmethod** **get\_registered** (*category: str = None*)  
Fetch the set of registered classes in a category.

**aries\_cloudagent.config.argparse.load\_argument\_groups** (*parser: argparse.ArgumentParser, \*groups*)

Log a set of argument groups into a parser.

**Returns** A callable to convert loaded arguments into a settings dictionary

## aries\_cloudagent.config.base module

Configuration base classes.

**class** aries\_cloudagent.config.base.**BaseInjector**  
Bases: `abc.ABC`

Base injector class.

**copy** () → aries\_cloudagent.config.base.BaseInjector  
Produce a copy of the injector instance.

**inject** (*base\_cls: type, settings: Mapping[str, object] = None, \*, required: bool = True*) → object  
Get the provided instance of a given class identifier.

### Parameters

- **cls** – The base class to retrieve an instance of
- **settings** – An optional mapping providing configuration to the provider

**Returns** An instance of the base class, or None

**class** aries\_cloudagent.config.base.**BaseProvider**  
Bases: `abc.ABC`

Base provider class.

**provide** (*settings: aries\_cloudagent.config.base.BaseSettings, injector: aries\_cloudagent.config.base.BaseInjector*)  
Provide the object instance given a config and injector.

**class** aries\_cloudagent.config.base.**BaseSettings**  
Bases: `collections.abc.Mapping`, `typing.Generic`

Base settings class.

**copy** () → aries\_cloudagent.config.base.BaseSettings  
Produce a copy of the settings instance.

**extend** (*other: Mapping[str, object]*) → aries\_cloudagent.config.base.BaseSettings  
Merge another mapping to produce a new settings instance.

**get\_bool** (\*var\_names, default=None) → bool  
Fetch a setting as a boolean value.

**Parameters**

- **var\_names** – A list of variable name alternatives
- **default** – The default value to return if none are defined

**get\_int** (\*var\_names, default=None) → int  
Fetch a setting as an integer value.

**Parameters**

- **var\_names** – A list of variable name alternatives
- **default** – The default value to return if none are defined

**get\_str** (\*var\_names, default=None) → str  
Fetch a setting as a string value.

**Parameters**

- **var\_names** – A list of variable name alternatives
- **default** – The default value to return if none are defined

**get\_value** (\*var\_names, default=None)  
Fetch a setting.

**Parameters**

- **var\_names** – A list of variable name alternatives
- **default** – The default value to return if none are defined

**Returns** The setting value, if defined, otherwise the default value

**exception** aries\_cloudagent.config.base.**ConfigError** (\*args, error\_code: str = None, \*\*kwargs)

Bases: *aries\_cloudagent.core.error.BaseError*

A base exception for all configuration errors.

**exception** aries\_cloudagent.config.base.**InjectorError** (\*args, error\_code: str = None, \*\*kwargs)

Bases: *aries\_cloudagent.config.base.ConfigError*

The base exception raised by *BaseInjector* implementations.

**exception** aries\_cloudagent.config.base.**ProviderError** (\*args, error\_code: str = None, \*\*kwargs)

Bases: *aries\_cloudagent.config.base.ConfigError*

The base exception raised by *BaseProvider* implementations.

**exception** aries\_cloudagent.config.base.**SettingsError** (\*args, error\_code: str = None, \*\*kwargs)

Bases: *aries\_cloudagent.config.base.ConfigError*

The base exception raised by *BaseSettings* implementations.

## aries\_cloudagent.config.base\_context module

Base injection context builder classes.



```
class aries_cloudagent.config.base_context.ContextBuilder (settings: Mapping[str, object] = None)

    Bases: abc.ABC

    Base injection context builder class.

    build () → aries_cloudagent.config.injection_context.InjectionContext
        Build the new injection context.

    update_settings (settings: Mapping[str, object])
        Update the context builder with additional settings.
```

### aries\_cloudagent.config.default\_context module

Classes for configuring the default injection context.

```
class aries_cloudagent.config.default_context.DefaultContextBuilder (settings: Mapping[str, object] = None)

    Bases: aries_cloudagent.config.base_context.ContextBuilder

    Default context builder.

    bind_providers (context: aries_cloudagent.config.injection_context.InjectionContext)
        Bind various class providers.

    build () → aries_cloudagent.config.injection_context.InjectionContext
        Build the new injection context.

    load_plugins (context: aries_cloudagent.config.injection_context.InjectionContext)
        Set up plugin registry and load plugins.
```

### aries\_cloudagent.config.error module

Errors for config modules.

```
exception aries_cloudagent.config.error.ArgsParseError (*args, error_code: str = None, **kwargs)

    Bases: aries_cloudagent.config.base.ConfigError

    Error raised when there is a problem parsing the command-line arguments.
```

### aries\_cloudagent.config.injection\_context module

Injection context implementation.

```
class aries_cloudagent.config.injection_context.InjectionContext (*, settings: Mapping[str, object] = None, enforce_typing: bool = True)

    Bases: aries_cloudagent.config.base.BaseInjector

    Manager for configuration settings and class providers.

    ROOT_SCOPE = 'application'
```

**copy()** → aries\_cloudagent.config.injection\_context.InjectionContext  
Produce a copy of the injector instance.

**inject** (*base\_cls: type, settings: Mapping[str, object] = None, \*, required: bool = True*) → object  
Get the provided instance of a given class identifier.

**Parameters**

- **cls** – The base class to retrieve an instance of
- **settings** – An optional mapping providing configuration to the provider

**Returns** An instance of the base class, or None

**injector**  
Accessor for scope-specific injector.

**injector\_for\_scope** (*scope\_name: str*) → aries\_cloudagent.config.injector.Injector  
Fetch the injector for a specific scope.

**Parameters** **scope\_name** – The unique scope identifier

**scope\_name**  
Accessor for the current scope name.

**settings**  
Accessor for scope-specific settings.

**start\_scope** (*scope\_name: str, settings: Mapping[str, object] = None*) → aries\_cloudagent.config.injection\_context.InjectionContext  
Begin a new named scope.

**Parameters**

- **scope\_name** – The unique name for the scope being entered
- **settings** – An optional mapping of additional settings to apply

**Returns** A new injection context representing the scope

**update\_settings** (*settings: Mapping[str, object]*)  
Update the scope with additional settings.

**exception** aries\_cloudagent.config.injection\_context.**InjectionContextError** (\*args, error\_code: str = None, \*\*kwargs)

Bases: *aries\_cloudagent.config.base.InjectorError*

Base class for issues in the injection context.

**class** aries\_cloudagent.config.injection\_context.**Scope** (*name, injector*)  
Bases: *tuple*

**injector**  
Alias for field number 1

**name**  
Alias for field number 0

## aries\_cloudagent.config.injector module

Standard Injector implementation.

**class** aries\_cloudagent.config.injector.**Injector** (*settings: Mapping[str, object] = None, enforce\_typing: bool = True*)

Bases: *aries\_cloudagent.config.base.BaseInjector*

Injector implementation with static and dynamic bindings.

**bind\_instance** (*base\_cls: type, instance: object*)

Add a static instance as a class binding.

**bind\_provider** (*base\_cls: type, provider: aries\_cloudagent.config.base.BaseProvider, \*, cache: bool = False*)

Add a dynamic instance resolver as a class binding.

**clear\_binding** (*base\_cls: type*)

Remove a previously-added binding.

**copy** () → *aries\_cloudagent.config.base.BaseInjector*

Produce a copy of the injector instance.

**get\_provider** (*base\_cls: type*)

Find the provider associated with a class binding.

**inject** (*base\_cls: type, settings: Mapping[str, object] = None, \*, required: bool = True*)

Get the provided instance of a given class identifier.

### Parameters

- **cls** – The base class to retrieve an instance of
- **params** – An optional dict providing configuration to the provider

**Returns** An instance of the base class, or None

### settings

Accessor for scope-specific settings.

## aries\_cloudagent.config.ledger module

Ledger configuration.

*aries\_cloudagent.config.ledger.accept\_taa* (*ledger: aries\_cloudagent.ledger.base.BaseLedger, taa\_info, provision: bool = False*) → bool

Perform TAA acceptance.

*aries\_cloudagent.config.ledger.fetch\_genesis\_transactions* (*genesis\_url: str*) → str

Get genesis transactions.

*aries\_cloudagent.config.ledger.ledger\_config* (*context: aries\_cloudagent.config.injection\_context.InjectionContext, public\_id: str, provision: bool = False*) → bool

Perform Indy ledger configuration.

## aries\_cloudagent.config.logging module

Utilities related to logging.

**class** `aries_cloudagent.config.logging.LoggingConfigurator`

Bases: `object`

Utility class used to configure logging and print an informative start banner.

**classmethod** `configure` (*logging\_config\_path: str = None, log\_level: str = None, log\_file: str = None*)

Configure logger.

**Parameters**

- **logging\_config\_path** – str: (Default value = None) Optional path to custom logging config
- **log\_level** – str: (Default value = None)

**classmethod** `print_banner` (*agent\_label, inbound\_transports, outbound\_transports, public\_did, admin\_server=None, banner\_length=40, border\_character=':'*)

Print a startup banner describing the configuration.

**Parameters**

- **agent\_label** – Agent Label
- **inbound\_transports** – Configured inbound transports
- **outbound\_transports** – Configured outbound transports
- **admin\_server** – Admin server info
- **public\_did** – Public DID
- **banner\_length** – (Default value = 40) Length of the banner
- **border\_character** – (Default value = “:”) Character to use in banner
- **border** –

`aries_cloudagent.config.logging.load_resource` (*path: str, encoding: str = None*) → `TextIO`

Open a resource file located in a python package or the local filesystem.

**Parameters** **path** – The resource path in the form of *dir/file* or *package:dir/file*

**Returns** A file-like object representing the resource

## **aries\_cloudagent.config.provider module**

Service provider implementations.

**class** `aries_cloudagent.config.provider.CachedProvider` (*provider: aries\_cloudagent.config.base.BaseProvider*)

Bases: `aries_cloudagent.config.base.BaseProvider`

Cache the result of another provider.

**provide** (*config: aries\_cloudagent.config.base.BaseSettings, injector: aries\_cloudagent.config.base.BaseInjector*)

Provide the object instance given a config and injector.

**class** `aries_cloudagent.config.provider.ClassProvider` (*instance\_cls: Union[str, type], \*ctor\_args, async\_init: str = None, \*\*ctor\_kwargs*)

Bases: `aries_cloudagent.config.base.BaseProvider`

Provider for a particular class.

```

class Inject (base_cls: type)
    Bases: object

    A class for passing injected arguments to the constructor.

    provide (config: aries_cloudagent.config.base.BaseSettings, injector:
               aries_cloudagent.config.base.BaseInjector)
        Provide the object instance given a config and injector.

class aries_cloudagent.config.provider.InstanceProvider (instance)
    Bases: aries_cloudagent.config.base.BaseProvider

    Provider for a previously-created instance.

    provide (config: aries_cloudagent.config.base.BaseSettings, injector:
               aries_cloudagent.config.base.BaseInjector)
        Provide the object instance given a config and injector.

class aries_cloudagent.config.provider.StatsProvider (provider:
                                                       aries_cloudagent.config.base.BaseProvider,
                                                       methods: Sequence[str], *, ig-
                                                       nore_missing: bool = True)
    Bases: aries_cloudagent.config.base.BaseProvider

    Add statistics to the results of another provider.

    provide (config: aries_cloudagent.config.base.BaseSettings, injector:
               aries_cloudagent.config.base.BaseInjector)
        Provide the object instance given a config and injector.

```

## aries\_cloudagent.config.settings module

Settings implementation.

```

class aries_cloudagent.config.settings.Settings (values: Mapping[str, object] = None)
    Bases: aries_cloudagent.config.base.BaseSettings

    Mutable settings implementation.

    clear_value (var_name: str)
        Remove a setting.

        Parameters var_name – The name of the setting

    copy () → aries_cloudagent.config.base.BaseSettings
        Produce a copy of the settings instance.

    extend (other: Mapping[str, object]) → aries_cloudagent.config.base.BaseSettings
        Merge another settings instance to produce a new instance.

    get_value (*var_names, default=None)
        Fetch a setting.

        Parameters
            • var_names – A list of variable name alternatives
            • default – The default value to return if none are defined

    set_default (var_name: str, value)
        Add a setting if not currently defined.

        Parameters

```

- **var\_name** – The name of the setting
- **value** – The value to assign

**set\_value** (*var\_name: str, value*)

Add a setting.

### Parameters

- **var\_name** – The name of the setting
- **value** – The value to assign

## aries\_cloudagent.config.util module

Entrypoint.

**class** aries\_cloudagent.config.util.**ByteSize** (*min\_size: int = 0, max\_size: int = 0*)

Bases: `object`

Argument value parser for byte sizes.

aries\_cloudagent.config.util.**common\_config** (*settings: Mapping[str, Any]*)

Perform common app configuration.

## aries\_cloudagent.config.wallet module

Wallet configuration.

aries\_cloudagent.config.wallet.**wallet\_config** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, provision: bool = False*)

Initialize the wallet.

## 1.1.5 aries\_cloudagent.core package

### Submodules

#### aries\_cloudagent.core.conductor module

The Conductor.

The conductor is responsible for coordinating messages that are received over the network, communicating with the ledger, passing messages to handlers, instantiating concrete implementations of required modules and storing data in the wallet.

**class** aries\_cloudagent.core.conductor.**Conductor** (*context\_builder: aries\_cloudagent.config.base\_context.ContextBuilder*)

Bases: `object`

Conductor class.

Class responsible for initializing concrete implementations of our require interfaces and routing inbound and outbound message data.

**dispatch\_complete** (*message: aries\_cloudagent.transport.inbound.message.InboundMessage, completed: aries\_cloudagent.utils.task\_queue.CompletedTask*)

Handle completion of message dispatch.

**get\_stats** () → dict

Get the current stats tracked by the conductor.

**handle\_not\_delivered** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, outbound: aries\_cloudagent.transport.outbound.message.OutboundMessage*)  
 Handle a message that failed delivery via outbound transports.

**handle\_not\_returned** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, outbound: aries\_cloudagent.transport.outbound.message.OutboundMessage*)  
 Handle a message that failed delivery via an inbound session.

**inbound\_message\_router** (*message: aries\_cloudagent.transport.inbound.message.InboundMessage, can\_respond: bool = False*)  
 Route inbound messages.

#### Parameters

- **message** – The inbound message instance
- **can\_respond** – If the session supports return routing

**outbound\_message\_router** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, outbound: aries\_cloudagent.transport.outbound.message.OutboundMessage, inbound: aries\_cloudagent.transport.inbound.message.InboundMessage = None*) → None  
 Route an outbound message.

#### Parameters

- **context** – The request context
- **message** – An outbound message to be sent
- **inbound** – The inbound message that produced this response, if available

**queue\_outbound** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, outbound: aries\_cloudagent.transport.outbound.message.OutboundMessage, inbound: aries\_cloudagent.transport.inbound.message.InboundMessage = None*)  
 Queue an outbound message.

#### Parameters

- **context** – The request context
- **message** – An outbound message to be sent
- **inbound** – The inbound message that produced this response, if available

**setup** ()  
 Initialize the global request context.

**start** () → None  
 Start the agent.

**stop** (*timeout=1.0*)  
 Stop the agent.

**webhook\_router** (*topic: str, payload: dict, endpoint: str, max\_attempts: int = None*)  
 Route a webhook through the outbound transport manager.

#### Parameters

- **topic** – The webhook topic
- **payload** – The webhook payload
- **endpoint** – The endpoint of the webhook target
- **max\_attempts** – The maximum number of attempts

## aries\_cloudagent.core.dispatcher module

The Dispatcher.

The dispatcher is responsible for coordinating data flow between handlers, providing lifecycle hook callbacks storing state for message threads, etc.

**class** `aries_cloudagent.core.dispatcher.Dispatcher` (*context:*  
*aries\_cloudagent.config.injection\_context.InjectionContext*)

Bases: `object`

Dispatcher class.

Class responsible for dispatching messages to message handlers and responding to other agents.

**complete** (*timeout: float = 0.1*)

Wait for pending tasks to complete.

**handle\_message** (*inbound\_message: aries\_cloudagent.transport.inbound.message.InboundMessage,*  
*send\_outbound: Coroutine[T\_co, T\_contra, V\_co], send\_webhook: Corou-*  
*tine[T\_co, T\_contra, V\_co] = None*)

Configure responder and message context and invoke the message handler.

### Parameters

- **inbound\_message** – The inbound message instance
- **send\_outbound** – Async function to send outbound messages
- **send\_webhook** – Async function to dispatch a webhook

**Returns** The response from the handler

**log\_task** (*task: aries\_cloudagent.utils.task\_queue.CompletedTask*)

Log a completed task using the stats collector.

**make\_message** (*parsed\_msg: dict*) → `aries_cloudagent.messaging.agent_message.AgentMessage`

Deserialize a message dict into the appropriate message instance.

Given a dict describing a message, this method returns an instance of the related message class.

**Parameters** **parsed\_msg** – The parsed message

**Returns** An instance of the corresponding message class for this message

### Raises

- `MessageParseError` – If the message doesn't specify @type
- `MessageParseError` – If there is no message class registered to handle
- the given type

**put\_task** (*coro: Coroutine[T\_co, T\_contra, V\_co], complete: Callable = None, ident: str = None*) →  
`aries_cloudagent.utils.task_queue.PendingTask`

Run a task in the task queue, potentially blocking other handlers.

**queue\_message** (*inbound\_message: aries\_cloudagent.transport.inbound.message.InboundMessage,*  
*send\_outbound: Coroutine[T\_co, T\_contra, V\_co], send\_webhook: Corou-*  
*tine[T\_co, T\_contra, V\_co] = None, complete: Callable = None*) →  
`aries_cloudagent.utils.task_queue.PendingTask`

Add a message to the processing queue for handling.

### Parameters

- **inbound\_message** – The inbound message instance



- **send\_outbound** – Async function to send outbound messages
- **send\_webhook** – Async function to dispatch a webhook
- **complete** – Function to call when the handler has completed

**Returns** A pending task instance resolving to the handler task

**run\_task** (*coro: Coroutine[T\_co, T\_contra, V\_co]*, *complete: Callable = None*, *ident: str = None*) → *\_asyncio.Task*  
Run a task in the task queue, potentially blocking other handlers.

**setup** ()  
Perform async instance setup.

**class** *aries\_cloudagent.core.dispatcher.DispatcherResponder* (*context: aries\_cloudagent.messaging.request\_context.RequestContext*, *inbound\_message: aries\_cloudagent.transport.inbound.message.InboundMessage*, *send\_outbound: Coroutine[T\_co, T\_contra, V\_co]*, *send\_webhook: Coroutine[T\_co, T\_contra, V\_co] = None, \*\*kwargs*)

Bases: *aries\_cloudagent.messaging.responder.BaseResponder*

Handle outgoing messages from message handlers.

**create\_outbound** (*message: Union[aries\_cloudagent.messaging.agent\_message.AgentMessage, str, bytes]*, *\*\*kwargs*) → *aries\_cloudagent.transport.outbound.message.OutboundMessage*  
Create an OutboundMessage from a message body.

**Parameters** *message* – The message payload

**send\_outbound** (*message: aries\_cloudagent.transport.outbound.message.OutboundMessage*)  
Send outbound message.

**Parameters** *message* – The *OutboundMessage* to be sent

**send\_webhook** (*topic: str*, *payload: dict*)  
Dispatch a webhook.

**Parameters**

- **topic** – the webhook topic identifier
- **payload** – the webhook payload value

## aries\_cloudagent.core.error module

Common exception classes.

**exception** *aries\_cloudagent.core.error.BaseError* (*\*args*, *error\_code: str = None*, *\*\*kwargs*)

Bases: *Exception*

Generic exception class which other exceptions should inherit from.

**error\_code** = **None**

**message**

Accessor for the error message.

```
exception aries_cloudagent.core.error.ProtocolDefinitionValidationError(*args,
                                                                    er-
                                                                    ror_code:
                                                                    str
                                                                    =
                                                                    None,
                                                                    **kwargs)
```

Bases: `aries_cloudagent.core.error.BaseError`

Error raised when there is a problem validating a protocol definition.

```
exception aries_cloudagent.core.error.ProtocolMinorVersionNotSupported(*args,
                                                                    er-
                                                                    ror_code:
                                                                    str
                                                                    =
                                                                    None,
                                                                    **kwargs)
```

Bases: `aries_cloudagent.core.error.BaseError`

Minimum minor version protocol error.

Error raised when protocol support exists but minimum minor version is higher than in @type parameter.

```
exception aries_cloudagent.core.error.StartupError(*args, error_code: str = None,
                                                    **kwargs)
```

Bases: `aries_cloudagent.core.error.BaseError`

Error raised when there is a problem starting the conductor.

## aries\_cloudagent.core.plugin\_registry module

Handle registration of plugin modules for extending functionality.

```
class aries_cloudagent.core.plugin_registry.PluginRegistry
    Bases: object

    Plugin registry for indexing application plugins.

    init_context (context: aries_cloudagent.config.injection_context.InjectionContext)
        Call plugin setup methods on the current context.

    load_protocol_version (context: aries_cloudagent.config.injection_context.InjectionContext,
                           mod: module, version_definition: dict = None)
        Load a particular protocol version.

    load_protocols (context: aries_cloudagent.config.injection_context.InjectionContext, plugin: mod-
                    ule)
        For modules that don't implement setup, register protocols manually.

    plugin_names
        Accessor for a list of all plugin modules.

    plugins
        Accessor for a list of all plugin modules.

    register_admin_routes (app)
        Call route registration methods on the current context.
```

**register\_package** (*package\_name: str*) → Sequence[module]  
 Register all modules (sub-packages) under a given package name.

**register\_plugin** (*module\_name: str*) → module  
 Register a plugin module.

**validate\_version** (*version\_list, module\_name*)  
 Validate version dict format.

## aries\_cloudagent.core.protocol\_registry module

Handle registration and publication of supported protocols.

**class** aries\_cloudagent.core.protocol\_registry.**ProtocolRegistry**

Bases: `object`

Protocol registry for indexing message families.

**controllers**  
 Accessor for a list of all protocol controller functions.

**message\_types**  
 Accessor for a list of all message types.

**parse\_type\_string** (*message\_type*)  
 Parse message type string and return dict with info.

**prepare\_disclosed** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, protocols: Sequence[str]*)  
 Call controllers and return publicly supported message families and roles.

**protocols**  
 Accessor for a list of all message protocols.

**protocols\_matching\_query** (*query: str*) → Sequence[str]  
 Return a list of message protocols matching a query string.

**register\_controllers** (*\*controller\_sets, version\_definition=None*)  
 Add new controllers.

**Parameters controller\_sets** – Mappings of message families to coroutines

**register\_message\_types** (*\*typesets, version\_definition=None*)  
 Add new supported message types.

**Parameters**

- **typesets** – Mappings of message types to register
- **version\_definition** – Optional version definition dict

**resolve\_message\_class** (*message\_type: str*) → type  
 Resolve a message\_type to a message class.

Given a message type identifier, this method returns the corresponding registered message class.

**Parameters message\_type** – Message type to resolve

**Returns** The resolved message class

## 1.1.6 aries\_cloudagent.holder package

### Submodules

#### aries\_cloudagent.holder.base module

Base holder class.

**class** aries\_cloudagent.holder.base.**BaseHolder**

Bases: `abc.ABC`

Base class for holder.

**create\_credential\_request** (*credential\_offer: dict, credential\_definition: dict, holder\_did: str*)  
→ `Tuple[str, str]`

Create a credential request for the given credential offer.

#### Parameters

- **credential\_offer** – The credential offer to create request for
- **credential\_definition** – The credential definition to create an offer for
- **holder\_did** – the DID of the agent making the request

**Returns** A tuple of the credential request and credential request metadata

**create\_presentation** (*presentation\_request: dict, requested\_credentials: dict, schemas: dict, credential\_definitions: dict, rev\_states: dict = None*) → `str`

Get credentials stored in the wallet.

#### Parameters

- **presentation\_request** – Valid indy format presentation request
- **requested\_credentials** – Indy format requested credentials
- **schemas** – Indy formatted schemas JSON
- **credential\_definitions** – Indy formatted credential definitions JSON
- **rev\_states** – Indy format revocation states JSON

**create\_revocation\_state** (*cred\_rev\_id: str, rev\_reg\_def: dict, rev\_reg\_delta: dict, timestamp: int, tails\_file\_path: str*) → `str`

Create current revocation state for a received credential.

#### Parameters

- **cred\_rev\_id** – credential revocation id in revocation registry
- **rev\_reg\_def** – revocation registry definition
- **rev\_reg\_delta** – revocation delta
- **timestamp** – delta timestamp

**Returns** the revocation state

**delete\_credential** (*credential\_id: str*)

Remove a credential stored in the wallet.

**Parameters** **credential\_id** – Credential id to remove

**get\_credential** (*credential\_id: str*) → `str`

Get a credential stored in the wallet.

**Parameters** `credential_id` – Credential id to retrieve

**get\_mime\_type** (*credential\_id: str, attr: str = None*) → Union[dict, str]  
Get MIME type per attribute (or for all attributes).

**Parameters**

- `credential_id` – credential id
- `attr` – attribute of interest or omit for all

**Returns:** Attribute MIME type or dict mapping attribute names to MIME types `attr_meta_json = all_meta.tags.get(attr)`

**store\_credential** (*credential\_definition: dict, credential\_data: dict, credential\_request\_metadata: dict, credential\_attr\_mime\_types=None, credential\_id: str = None, rev\_reg\_def: dict = None*)  
Store a credential in the wallet.

**Parameters**

- `credential_definition` – Credential definition for this credential
- `credential_data` – Credential data generated by the issuer
- `credential_request_metadata` – credential request metadata generated by the issuer
- `credential_attr_mime_types` – dict mapping attribute names to (optional) MIME types to store as non-secret record, if specified
- `credential_id` – optionally override the stored credential id
- `rev_reg_def` – revocation registry definition in json

**Returns** the ID of the stored credential

**exception** `aries_cloudagent.holder.base.HolderError` (\*args, error\_code: str = None, \*\*kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Base class for holder exceptions.

## aries\_cloudagent.holder.indy module

Indy holder implementation.

**class** `aries_cloudagent.holder.indy.IndyHolder` (wallet)

Bases: `aries_cloudagent.holder.base.BaseHolder`

Indy holder class.

**RECORD\_TYPE\_MIME\_TYPES** = 'attribute-mime-types'

**create\_credential\_request** (*credential\_offer: dict, credential\_definition: dict, holder\_did: str*) → Tuple[str, str]

Create a credential request for the given credential offer.

**Parameters**

- `credential_offer` – The credential offer to create request for
- `credential_definition` – The credential definition to create an offer for
- `holder_did` – the DID of the agent making the request

**Returns** A tuple of the credential request and credential request metadata

**create\_presentation** (*presentation\_request: dict, requested\_credentials: dict, schemas: dict, credential\_definitions: dict, rev\_states: dict = None*) → str

Get credentials stored in the wallet.

**Parameters**

- **presentation\_request** – Valid indy format presentation request
- **requested\_credentials** – Indy format requested credentials
- **schemas** – Indy formatted schemas JSON
- **credential\_definitions** – Indy formatted credential definitions JSON
- **rev\_states** – Indy format revocation states JSON

**create\_revocation\_state** (*cred\_rev\_id: str, rev\_reg\_def: dict, rev\_reg\_delta: dict, timestamp: int, tails\_file\_path: str*) → str

Create current revocation state for a received credential.

**Parameters**

- **cred\_rev\_id** – credential revocation id in revocation registry
- **rev\_reg\_def** – revocation registry definition
- **rev\_reg\_delta** – revocation delta
- **timestamp** – delta timestamp

**Returns** the revocation state

**delete\_credential** (*credential\_id: str*)

Remove a credential stored in the wallet.

**Parameters** **credential\_id** – Credential id to remove

**get\_credential** (*credential\_id: str*) → str

Get a credential stored in the wallet.

**Parameters** **credential\_id** – Credential id to retrieve

**get\_credentials** (*start: int, count: int, wql: dict*)

Get credentials stored in the wallet.

**Parameters**

- **start** – Starting index
- **count** – Number of records to return
- **wql** – wql query dict

**get\_credentials\_for\_presentation\_request\_by\_referent** (*presentation\_request: dict, referents: Sequence[str], start: int, count: int, extra\_query: dict = {}*)

Get credentials stored in the wallet.

**Parameters**

- **presentation\_request** – Valid presentation request from issuer
- **referents** – Presentation request referents to use to search for creds

- **start** – Starting index
- **count** – Maximum number of records to return
- **extra\_query** – wql query dict

**get\_mime\_type** (*credential\_id: str, attr: str = None*) → Union[dict, str]

Get MIME type per attribute (or for all attributes).

#### Parameters

- **credential\_id** – credential id
- **attr** – attribute of interest or omit for all

**Returns:** Attribute MIME type or dict mapping attribute names to MIME types `attr_meta_json = all_meta.tags.get(attr)`

**store\_credential** (*credential\_definition: dict, credential\_data: dict, credential\_request\_metadata: dict, credential\_attr\_mime\_types=None, credential\_id: str = None, rev\_reg\_def: dict = None*) → str

Store a credential in the wallet.

#### Parameters

- **credential\_definition** – Credential definition for this credential
- **credential\_data** – Credential data generated by the issuer
- **credential\_request\_metadata** – credential request metadata generated by the issuer
- **credential\_attr\_mime\_types** – dict mapping attribute names to (optional) MIME types to store as non-secret record, if specified
- **credential\_id** – optionally override the stored credential id
- **rev\_reg\_def** – revocation registry definition in json

**Returns** the ID of the stored credential

## 1.1.7 aries\_cloudagent.issuer package

### Submodules

#### aries\_cloudagent.issuer.base module

Ledger issuer class.

**class** aries\_cloudagent.issuer.base.**BaseIssuer**

Bases: `abc.ABC`

Base class for issuer.

**create\_and\_store\_credential\_definition** (*origin\_did: str, schema: dict, signature\_type: str = None, tag: str = None, support\_revocation: bool = False*) → Tuple[str, str]

Create a new credential definition and store it in the wallet.

#### Parameters

- **origin\_did** – the DID issuing the credential definition

- **schema\_json** – the schema used as a basis
- **signature\_type** – the credential definition signature type (default ‘CL’)
- **tag** – the credential definition tag
- **support\_revocation** – whether to enable revocation for this credential def

**Returns** A tuple of the credential definition ID and JSON

**create\_and\_store\_revocation\_registry** (*origin\_did: str, cred\_def\_id: str, revoc\_def\_type: str, tag: str, max\_cred\_num: int, tails\_base\_path: str, issuance\_type: str = None*) → Tuple[str, str, str]

Create a new revocation registry and store it in the wallet.

#### Parameters

- **origin\_did** – the DID issuing the revocation registry
- **cred\_def\_id** – the identifier of the related credential definition
- **revoc\_def\_type** – the revocation registry type (default CL\_ACCUM)
- **tag** – the unique revocation registry tag
- **max\_cred\_num** – the number of credentials supported in the registry
- **tails\_base\_path** – where to store the tails file
- **issuance\_type** – optionally override the issuance type

**Returns** A tuple of the revocation registry ID, JSON, and entry JSON

**create\_and\_store\_schema** (*origin\_did: str, schema\_name: str, schema\_version: str, attribute\_names: Sequence[str]*) → Tuple[str, str]

Create a new credential schema and store it in the wallet.

#### Parameters

- **origin\_did** – the DID issuing the credential definition
- **schema\_name** – the schema name
- **schema\_version** – the schema version
- **attribute\_names** – a sequence of schema attribute names

**Returns** A tuple of the schema ID and JSON

**create\_credential** (*schema: dict, credential\_offer: dict, credential\_request: dict, credential\_values: dict, revoc\_reg\_id: str = None, tails\_reader\_handle: int = None*) → Tuple[str, str]

Create a credential.

**Args** schema: Schema to create credential for  
 credential\_offer: Credential Offer to create credential for  
 credential\_request: Credential request to create credential for  
 credential\_values: Values to go in credential  
 revoc\_reg\_id: ID of the revocation registry  
 tails\_reader\_handle: Handle for the tails file blob reader

**Returns** A tuple of created credential and revocation id

**create\_credential\_offer** (*credential\_definition\_id*) → str

Create a credential offer for the given credential definition id.



**Parameters** `credential_definition_id` – The credential definition to create an offer for

**Returns** The created credential offer

**`credential_definition_in_wallet`** (*credential\_definition\_id: str*) → bool  
Check whether a given credential definition ID is present in the wallet.

**Parameters** `credential_definition_id` – The credential definition ID to check

**`make_credential_definition_id`** (*origin\_id: str, schema: dict, signature\_type: str = None, tag: str = None*) → str  
Derive the ID for a credential definition.

**`make_schema_id`** (*origin\_id: str, schema\_name: str, schema\_version: str*) → str  
Derive the ID for a schema.

**`merge_revocation_registry_deltas`** (*fro\_delta: str, to\_delta: str*) → str  
Merge revocation registry deltas.

**Parameters**

- **`fro_delta`** – original delta in JSON format
- **`to_delta`** – incoming delta in JSON format

**Returns** Merged delta in JSON format

**`revoke_credentials`** (*revoc\_reg\_id: str, tails\_file\_path: str, cred\_revoc\_ids: Sequence[str]*) → str  
Revoke a set of credentials in a revocation registry.

**Parameters**

- **`revoc_reg_id`** – ID of the revocation registry
- **`tails_file_path`** – path to the local tails file
- **`cred_revoc_ids`** – sequences of credential indexes in the revocation registry

**Returns** the combined revocation delta

**exception** `aries_cloudagent.issuer.base.IssuerError` (\*args, error\_code: str = None, \*\*kwargs)  
Bases: `aries_cloudagent.core.error.BaseError`  
Generic issuer error.

**exception** `aries_cloudagent.issuer.base.IssuerRevocationRegistryFullError` (\*args, error\_code: str = None, \*\*kwargs)  
Bases: `aries_cloudagent.issuer.base.IssuerError`  
Revocation registry is full when issuing a new credential.

## aries\_cloudagent.issuer.indy module

Indy issuer implementation.

**class** `aries_cloudagent.issuer.indy.IndyIssuer` (*wallet*)

Bases: `aries_cloudagent.issuer.base.BaseIssuer`

Indy issuer class.

**create\_and\_store\_credential\_definition** (*origin\_id: str, schema: dict, signature\_type: str = None, tag: str = None, support\_revocation: bool = False*) → `Tuple[str, str]`

Create a new credential definition and store it in the wallet.

#### Parameters

- **origin\_id** – the DID issuing the credential definition
- **schema** – the schema used as a basis
- **signature\_type** – the credential definition signature type (default ‘CL’)
- **tag** – the credential definition tag
- **support\_revocation** – whether to enable revocation for this credential def

**Returns** A tuple of the credential definition ID and JSON

**create\_and\_store\_revocation\_registry** (*origin\_id: str, cred\_def\_id: str, revoc\_def\_type: str, tag: str, max\_cred\_num: int, tails\_base\_path: str, issuance\_type: str = None*) → `Tuple[str, str, str]`

Create a new revocation registry and store it in the wallet.

#### Parameters

- **origin\_id** – the DID issuing the revocation registry
- **cred\_def\_id** – the identifier of the related credential definition
- **revoc\_def\_type** – the revocation registry type (default CL\_ACCUM)
- **tag** – the unique revocation registry tag
- **max\_cred\_num** – the number of credentials supported in the registry
- **tails\_base\_path** – where to store the tails file
- **issuance\_type** – optionally override the issuance type

**Returns** A tuple of the revocation registry ID, JSON, and entry JSON

**create\_and\_store\_schema** (*origin\_id: str, schema\_name: str, schema\_version: str, attribute\_names: Sequence[str]*) → `Tuple[str, str]`

Create a new credential schema and store it in the wallet.

#### Parameters

- **origin\_id** – the DID issuing the credential definition
- **schema\_name** – the schema name
- **schema\_version** – the schema version
- **attribute\_names** – a sequence of schema attribute names

**Returns** A tuple of the schema ID and JSON

**create\_credential** (*schema: dict, credential\_offer: dict, credential\_request: dict, credential\_values: dict, revoc\_reg\_id: str = None, tails\_file\_path: str = None*) → Tuple[str, str]

Create a credential.

**Args** *schema*: Schema to create credential for *credential\_offer*: Credential Offer to create credential for *credential\_request*: Credential request to create credential for *credential\_values*: Values to go in credential *revoc\_reg\_id*: ID of the revocation registry *tails\_file\_path*: Path to the local tails file

**Returns** A tuple of created credential and revocation id

**create\_credential\_offer** (*credential\_definition\_id: str*) → str

Create a credential offer for the given credential definition id.

**Parameters** *credential\_definition\_id* – The credential definition to create an offer for

**Returns** The created credential offer

**credential\_definition\_in\_wallet** (*credential\_definition\_id: str*) → bool

Check whether a given credential definition ID is present in the wallet.

**Parameters** *credential\_definition\_id* – The credential definition ID to check

**make\_credential\_definition\_id** (*origin\_id: str, schema: dict, signature\_type: str = None, tag: str = None*) → str

Derive the ID for a credential definition.

**make\_schema\_id** (*origin\_id: str, schema\_name: str, schema\_version: str*) → str

Derive the ID for a schema.

**merge\_revocation\_registry\_deltas** (*fro\_delta: str, to\_delta: str*) → str

Merge revocation registry deltas.

**Parameters**

- **fro\_delta** – original delta in JSON format
- **to\_delta** – incoming delta in JSON format

**Returns** Merged delta in JSON format

**revoke\_credentials** (*revoc\_reg\_id: str, tails\_file\_path: str, cred\_revoc\_ids: Sequence[str]*) → str

Revoke a set of credentials in a revocation registry.

**Parameters**

- **revoc\_reg\_id** – ID of the revocation registry
- **tails\_file\_path** – path to the local tails file
- **cred\_revoc\_ids** – sequences of credential indexes in the revocation registry

**Returns** the combined revocation delta

## aries\_cloudagent.issuer.util module

### 1.1.8 aries\_cloudagent.ledger package

#### Submodules

## aries\_cloudagent.ledger.base module

Ledger base class.

```
class aries_cloudagent.ledger.base.BaseLedger
```

Bases: `abc.ABC`

Base class for ledger.

**LEDGER\_TYPE** = `None`

**accept\_txn\_author\_agreement** (*taa\_record: dict, mechanism: str, accept\_time: int = None*)

Save a new record recording the acceptance of the TAA.

**create\_and\_send\_credential\_definition** (*issuer: aries\_cloudagent.issuer.base.BaseIssuer, schema\_id: str, signature\_type: str = None, tag: str = None, support\_revocation: bool = False*)  
→ `Tuple[str, dict]`

Send credential definition to ledger and store relevant key matter in wallet.

### Parameters

- **issuer** – The issuer instance to use for credential definition creation
- **schema\_id** – The schema id of the schema to create cred def for
- **signature\_type** – The signature type to use on the credential definition
- **tag** – Optional tag to distinguish multiple credential definitions
- **support\_revocation** – Optional flag to enable revocation for this cred def

**create\_and\_send\_schema** (*issuer: aries\_cloudagent.issuer.base.BaseIssuer, schema\_name: str, schema\_version: str, attribute\_names: Sequence[str]*) → `Tuple[str, dict]`

Send schema to ledger.

### Parameters

- **issuer** – The issuer instance to use for schema creation
- **schema\_name** – The schema name
- **schema\_version** – The schema version
- **attribute\_names** – A list of schema attributes

**did\_to\_nym** (*did: str*) → `str`

Remove the ledger's DID prefix to produce a nym.

**fetch\_txn\_author\_agreement** ()

Fetch the current AML and TAA from the ledger.

**get\_credential\_definition** (*credential\_definition\_id: str*) → `dict`

Get a credential definition from the cache if available, otherwise the ledger.

**Parameters** **credential\_definition\_id** – The schema id of the schema to fetch cred def for

**get\_endpoint\_for\_did** (*did: str*) → `str`

Fetch the endpoint for a ledger DID.

**Parameters** **did** – The DID to look up on the ledger or in the cache

**get\_key\_for\_did** (*did: str*) → `str`

Fetch the verkey for a ledger DID.

**Parameters** **did** – The DID to look up on the ledger or in the cache

**get\_latest\_txn\_author\_acceptance** ()

Look up the latest TAA acceptance.

**get\_revoc\_reg\_def** (*revoc\_reg\_id: str*) → dict

Look up a revocation registry definition by ID.

**get\_revoc\_reg\_delta** (*revoc\_reg\_id: str, timestamp\_from=0, timestamp\_to=None*) -> (<class 'dict'>, <class 'int'>)

Look up a revocation registry delta by ID.

**get\_revoc\_reg\_entry** (*revoc\_reg\_id: str, timestamp: int*)

Get revocation registry entry by revocation registry ID and timestamp.

**get\_schema** (*schema\_id: str*) → dict

Get a schema from the cache if available, otherwise fetch from the ledger.

**Parameters** **schema\_id** – The schema id (or stringified sequence number) to retrieve

**get\_txn\_author\_agreement** (*reload: bool = False*)

Get the current transaction author agreement, fetching it if necessary.

**nym\_to\_did** (*nym: str*) → str

Format a nym with the ledger's DID prefix.

**register\_nym** (*did: str, verkey: str, alias: str = None, role: str = None*)

Register a nym on the ledger.

**Parameters**

- **did** – DID to register on the ledger.
- **verkey** – The verification key of the keypair.
- **alias** – Human-friendly alias to assign to the DID.
- **role** – For permissioned ledgers, what role should the new DID have.

**send\_revoc\_reg\_def** (*revoc\_reg\_def: dict, issuer\_did: str = None*)

Publish a revocation registry definition to the ledger.

**send\_revoc\_reg\_entry** (*revoc\_reg\_id: str, revoc\_def\_type: str, revoc\_reg\_entry: dict, issuer\_did: str = None*)

Publish a revocation registry entry to the ledger.

**taa\_digest** (*version: str, text: str*)

Generate the digest of a TAA record.

**update\_endpoint\_for\_did** (*did: str, endpoint: str*) → bool

Check and update the endpoint on the ledger.

**Parameters**

- **did** – The ledger DID
- **endpoint** – The endpoint address

## aries\_cloudagent.ledger.error module

Ledger related errors.

```
exception aries_cloudagent.ledger.error.BadLedgerRequestError(*args, error_code: str = None, **kwargs)
```

Bases: `aries_cloudagent.ledger.error.LedgerError`

The current request cannot proceed.

```
exception aries_cloudagent.ledger.error.ClosedPoolError(*args, error_code: str = None, **kwargs)
```

Bases: `aries_cloudagent.ledger.error.LedgerError`

Indy pool is closed.

```
exception aries_cloudagent.ledger.error.LedgerConfigError(*args, error_code: str = None, **kwargs)
```

Bases: `aries_cloudagent.ledger.error.LedgerError`

Base class for ledger configuration errors.

```
exception aries_cloudagent.ledger.error.LedgerError(*args, error_code: str = None, **kwargs)
```

Bases: `aries_cloudagent.core.error.BaseError`

Base class for ledger errors.

```
exception aries_cloudagent.ledger.error.LedgerTransactionError(*args, error_code: str = None, **kwargs)
```

Bases: `aries_cloudagent.ledger.error.LedgerError`

The ledger rejected the transaction.

## **aries\_cloudagent.ledger.indy module**

Indy ledger implementation.

```
class aries_cloudagent.ledger.indy.IndyLedger(pool_name: str, wallet: aries_cloudagent.wallet.base.BaseWallet, *, keepalive: int = 0, cache: aries_cloudagent.cache.base.BaseCache = None, cache_duration: int = 600, read_only: bool = False)
```

Bases: `aries_cloudagent.ledger.base.BaseLedger`

Indy ledger class.

**LEDGER\_TYPE** = 'indy'

```
accept_txn_author_agreement (taa_record: dict, mechanism: str, accept_time: int = None)  
    Save a new record recording the acceptance of the TAA.
```

```
check_existing_schema (public_did: str, schema_name: str, schema_version: str, attribute_names: Sequence[str]) → Tuple[str, dict]  
    Check if a schema has already been published.
```

```
check_pool_config () → bool  
    Check if a pool config has been created.
```

```
close ()  
    Close the pool ledger.
```

**create\_and\_send\_credential\_definition** (*issuer: aries\_cloudagent.issuer.base.BaseIssuer, schema\_id: str, signature\_type: str = None, tag: str = None, support\_revocation: bool = False*) → Tuple[str, dict]

Send credential definition to ledger and store relevant key matter in wallet.

#### Parameters

- **issuer** – The issuer instance to use for credential definition creation
- **schema\_id** – The schema id of the schema to create cred def for
- **signature\_type** – The signature type to use on the credential definition
- **tag** – Optional tag to distinguish multiple credential definitions
- **support\_revocation** – Optional flag to enable revocation for this cred def

**create\_and\_send\_schema** (*issuer: aries\_cloudagent.issuer.base.BaseIssuer, schema\_name: str, schema\_version: str, attribute\_names: Sequence[str]*) → Tuple[str, dict]

Send schema to ledger.

#### Parameters

- **issuer** – The issuer instance creating the schema
- **schema\_name** – The schema name
- **schema\_version** – The schema version
- **attribute\_names** – A list of schema attributes

**create\_pool\_config** (*genesis\_transactions: str, recreate: bool = False*)

Create the pool ledger configuration.

**credential\_definition\_id2schema\_id** (*credential\_definition\_id*)

From a credential definition, get the identifier for its schema.

**Parameters** **credential\_definition\_id** – The identifier of the credential definition from which to identify a schema

**fetch\_credential\_definition** (*credential\_definition\_id: str*) → dict

Get a credential definition from the ledger by id.

**Parameters** **credential\_definition\_id** – The cred def id of the cred def to fetch

**fetch\_schema\_by\_id** (*schema\_id: str*) → dict

Get schema from ledger.

**Parameters** **schema\_id** – The schema id (or stringified sequence number) to retrieve

**Returns** Indy schema dict

**fetch\_schema\_by\_seq\_no** (*seq\_no: int*)

Fetch a schema by its sequence number.

**Parameters** **seq\_no** – schema ledger sequence number

**Returns** Indy schema dict

**fetch\_txn\_author\_agreement** () → dict

Fetch the current AML and TAA from the ledger.

**get\_credential\_definition** (*credential\_definition\_id: str*) → dict

Get a credential definition from the cache if available, otherwise the ledger.

**Parameters** `credential_definition_id` – The schema id of the schema to fetch credential for

**get\_endpoint\_for\_did** (*did: str*) → str  
Fetch the endpoint for a ledger DID.

**Parameters** `did` – The DID to look up on the ledger or in the cache

**get\_indy\_storage** () → `aries_cloudagent.storage.indy.IndyStorage`  
Get an IndyStorage instance for the current wallet.

**get\_key\_for\_did** (*did: str*) → str  
Fetch the verkey for a ledger DID.

**Parameters** `did` – The DID to look up on the ledger or in the cache

**get\_latest\_txn\_author\_acceptance** () → dict  
Look up the latest TAA acceptance.

**get\_revoc\_reg\_def** (*revoc\_reg\_id: str*) → dict  
Get revocation registry definition by ID.

**get\_revoc\_reg\_delta** (*revoc\_reg\_id: str, timestamp\_from=0, timestamp\_to=None*) → (`<class 'dict'>`, `<class 'int'>`)  
Look up a revocation registry delta by ID.

:param `revoc_reg_id` revocation registry id :param `timestamp_from` from time. a total number of seconds from Unix Epoch :param `timestamp_to` to time. a total number of seconds from Unix Epoch

:returns delta response, delta timestamp

**get\_revoc\_reg\_entry** (*revoc\_reg\_id: str, timestamp: int*)  
Get revocation registry entry by revocation registry ID and timestamp.

**get\_schema** (*schema\_id: str*) → dict  
Get a schema from the cache if available, otherwise fetch from the ledger.

**Parameters** `schema_id` – The schema id (or stringified sequence number) to retrieve

**get\_txn\_author\_agreement** (*reload: bool = False*) → dict  
Get the current transaction author agreement, fetching it if necessary.

**nym\_to\_did** (*nym: str*) → str  
Format a nym with the ledger's DID prefix.

**open** ()  
Open the pool ledger, creating it if necessary.

**register\_nym** (*did: str, verkey: str, alias: str = None, role: str = None*)  
Register a nym on the ledger.

**Parameters**

- **did** – DID to register on the ledger.
- **verkey** – The verification key of the keypair.
- **alias** – Human-friendly alias to assign to the DID.
- **role** – For permissioned ledgers, what role should the new DID have.

**send\_revoc\_reg\_def** (*revoc\_reg\_def: dict, issuer\_did: str = None*)  
Publish a revocation registry definition to the ledger.



**send\_revoc\_reg\_entry** (*revoc\_reg\_id: str, revoc\_def\_type: str, revoc\_reg\_entry: dict, issuer\_did: str = None*)

Publish a revocation registry entry to the ledger.

**taa\_digest** (*version: str, text: str*)

Generate the digest of a TAA record.

**taa\_rough\_timestamp** () → int

Get a timestamp accurate to the day.

Anything more accurate is a privacy concern.

**update\_endpoint\_for\_did** (*did: str, endpoint: str*) → bool

Check and update the endpoint on the ledger.

#### Parameters

- **did** – The ledger DID
- **endpoint** – The endpoint address

## aries\_cloudagent.ledger.provider module

Default ledger provider classes.

**class** `aries_cloudagent.ledger.provider.LedgerProvider`

Bases: `aries_cloudagent.config.base.BaseProvider`

Provider for the default ledger implementation.

**LEDGER\_CLASSES** = {'indy': 'aries\_cloudagent.ledger.indy.IndyLedger'}

**provide** (*settings: aries\_cloudagent.config.base.BaseSettings, injector: aries\_cloudagent.config.base.BaseInjector*)

Create and open the ledger instance.

## aries\_cloudagent.ledger.routes module

Ledger admin routes.

**class** `aries_cloudagent.ledger.routes.AMLRecordSchema` (\*, *only=None, exclude=(), many=False, context=None, load\_only=(), dump\_only=(), partial=False, unknown=None*)

Bases: `marshmallow.schema.Schema`

Ledger AML record.

**opts** = <marshmallow.schema.SchemaOpts object>

**class** `aries_cloudagent.ledger.routes.TAAAcceptSchema` (\*, *only=None, exclude=(), many=False, context=None, load\_only=(), dump\_only=(), partial=False, unknown=None*)

Bases: `marshmallow.schema.Schema`

Input schema for accepting the TAA.

**opts** = <marshmallow.schema.SchemaOpts object>

```
class aries_cloudagent.ledger.routes.TAAAcceptanceSchema (*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)
```

Bases: `marshmallow.schema.Schema`

TAA acceptance record.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.ledger.routes.TAAInfoSchema (*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)
```

Bases: `marshmallow.schema.Schema`

Transaction author agreement info.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.ledger.routes.TAARecordSchema (*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)
```

Bases: `marshmallow.schema.Schema`

Ledger TAA record.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.ledger.routes.TAAResultSchema (*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)
```

Bases: `marshmallow.schema.Schema`

Result schema for a transaction author agreement.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
aries_cloudagent.ledger.routes.get_did_endpoint (request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f79514398d0>)
```

Request handler for getting a verkey for a DID from the ledger.

**Parameters** **request** – aiohttp request object

```
aries_cloudagent.ledger.routes.get_did_verkey (request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f79514398d0>)
```

Request handler for getting a verkey for a DID from the ledger.

**Parameters** **request** – aiohttp request object

```
aries_cloudagent.ledger.routes.ledger_accept_taa (request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f79514398d0>)
```

Request handler for accepting the current transaction author agreement.

**Parameters** **request** – aiohttp request object

**Returns** The DID list response

`aries_cloudagent.ledger.routes.ledger_get_taa` (*request: <sphinx.ext.autodoc.importer.\_MockObject object at 0x7f79514398d0>*)

Request handler for fetching the transaction author agreement.

**Parameters** `request` – aiohttp request object

**Returns** The TAA information including the AML

`aries_cloudagent.ledger.routes.register` (*app: <sphinx.ext.autodoc.importer.\_MockObject object at 0x7f79514398d0>*)

Register routes.

`aries_cloudagent.ledger.routes.register_ledger_nym` (*request: <sphinx.ext.autodoc.importer.\_MockObject object at 0x7f79514398d0>*)

Request handler for registering a NYM with the ledger.

**Parameters** `request` – aiohttp request object

## aries\_cloudagent.ledger.util module

Ledger utilities.

## 1.1.9 aries\_cloudagent.messaging package

### Subpackages

#### aries\_cloudagent.messaging.ack package

### Submodules

#### aries\_cloudagent.messaging.ack.message module

Represents an explicit ack message as per Aries RFC 15.

**class** `aries_cloudagent.messaging.ack.message.Ack` (*status: str = None, \*\*kwargs*)

Bases: `aries_cloudagent.messaging.agent_message.AgentMessage`

Base class representing an explicit ack message.

Subclass to adopt, specify Meta message type and handler class.

**class** `Meta`

Bases: `object`

Ack metadata.

**schema\_class** = 'AckSchema'

**class** `aries_cloudagent.messaging.ack.message.AckSchema` (*\*args, \*\*kwargs*)

Bases: `aries_cloudagent.messaging.agent_message.AgentMessageSchema`

Schema for Ack base class.

**class** `Meta`

Bases: `object`

Ack schema metadata.

**model\_class**  
alias of [Ack](#)

**status** = <fields.Constant (default='OK', attribute=None, validate=None, required=True, ...)

### **aries\_cloudagent.messaging.credential\_definitions package**

#### **Submodules**

#### **aries\_cloudagent.messaging.credential\_definitions.routes module**

Credential definition admin routes.

**class** aries\_cloudagent.messaging.credential\_definitions.routes.**CredentialDefinitionGetResu...**

Bases: `marshmallow.schema.Schema`

Results schema for schema get request.

**opts** = <marshmallow.schema.SchemaOpts object>

**class** aries\_cloudagent.messaging.credential\_definitions.routes.**CredentialDefinitionSchema** (

Bases: `marshmallow.schema.Schema`

Credential definition schema.

**opts** = <marshmallow.schema.SchemaOpts object>

```
class aries_cloudagent.messaging.credential_definitions.routes.CredentialDefinitionSendReq
```

Bases: `marshmallow.schema.Schema`

Request schema for schema send request.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.messaging.credential_definitions.routes.CredentialDefinitionSendRes
```

Bases: `marshmallow.schema.Schema`

Results schema for schema send request.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.messaging.credential_definitions.routes.CredentialDefinitionsCreated
```

Bases: `marshmallow.schema.Schema`

Results schema for cred-defs-created request.

**opts** = `<marshmallow.schema.SchemaOpts object>`

`aries_cloudagent.messaging.credential_definitions.routes.credential_definitions_created` (*request object at 0x7f7950a41198*)

Request handler for retrieving credential definitions that current agent created.

**Parameters** `request` – aiohttp request object

**Returns** The identifiers of matching credential definitions.

`aries_cloudagent.messaging.credential_definitions.routes.credential_definitions_get_credential_definition`

Request handler for getting a credential definition from the ledger.

**Parameters** `request` – aiohttp request object

**Returns** The credential definition details.

`aries_cloudagent.messaging.credential_definitions.routes.credential_definitions_send_credential_definition`

Request handler for sending a credential definition to the ledger.

**Parameters** `request` – aiohttp request object

**Returns** The credential definition identifier

`aries_cloudagent.messaging.credential_definitions.routes.register` (*app: <sphinx.ext.autodoc.importer.\_MockObject object at 0x7f7950a41198>*)

Register routes.

### `aries_cloudagent.messaging.credential_definitions.util` module

Credential definition utilities.

### `aries_cloudagent.messaging.decorators` package

#### Submodules

#### `aries_cloudagent.messaging.decorators.attach_decorator` module

A message decorator for attachments.

An attach decorator embeds content or specifies appended content.

```

class aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator(*,
                                                                            ident:
                                                                            str
                                                                            =
                                                                            None,
                                                                            de-
                                                                            scrip-
                                                                            tion:
                                                                            str
                                                                            =
                                                                            None,
                                                                            file-
                                                                            name:
                                                                            str
                                                                            =
                                                                            None,
                                                                            mime_type:
                                                                            str
                                                                            =
                                                                            None,
                                                                            last-
                                                                            mod_time:
                                                                            str
                                                                            =
                                                                            None,
                                                                            byte_count:
                                                                            int
                                                                            =
                                                                            None,
                                                                            data:
                                                                            aries_cloudagent.m
                                                                            **kwargs)

```

Bases: `aries_cloudagent.messaging.models.base.BaseModel`

Class representing attach decorator.

```
class Meta
```

Bases: `object`

AttachDecorator metadata.

```
schema_class = 'AttachDecoratorSchema'
```

```
classmethod from_indy_dict(indy_dict: dict, *, ident: str = None, description: str = None,
                           filename: str = None, lastmod_time: str = None, byte_count: int
                           = None)
```

Create *AttachDecorator* instance from indy object (dict).

Given indy object (dict), JSON dump, base64-encode, and embed it as data; mark *application/json* MIME type.

#### Parameters

- **indy\_dict** – indy (dict) data structure
- **ident** – optional attachment identifier (default random UUID4)
- **description** – optional attachment description
- **filename** – optional attachment filename

- **lastmod\_time** – optional attachment last modification time
- **byte\_count** – optional attachment byte count

#### **indy\_dict**

Return indy data structure encoded in attachment.

Returns: dict with indy object in data attachment

```
class aries_cloudagent.messaging.decorators.attach_decorator.AttachDecoratorData (*,
                                                                 jws_:
                                                                 aries_cloudagent.messaging.decorators.attach_decorator.AttachDecoratorData.jws_
                                                                 =
                                                                 None,
                                                                 sha256_:
                                                                 str
                                                                 =
                                                                 None,
                                                                 links_:
                                                                 Union[list,
                                                                 str]
                                                                 =
                                                                 None,
                                                                 base64_:
                                                                 str
                                                                 =
                                                                 None,
                                                                 json_:
                                                                 str
                                                                 =
                                                                 None)
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModel*

Attach decorator data.

#### **class Meta**

Bases: *object*

AttachDecoratorData metadata.

**schema\_class** = 'AttachDecoratorDataSchema'

#### **base64**

Accessor for base64 decorator data, or None.

**header\_map** (*idx: int = 0, jose: bool = True*) → Mapping[KT, VT\_co]

Accessor for header info at input index, default 0 or unique for singly-signed.

#### **Parameters**

- **idx** – index of interest, zero-based (default 0)
- **jose** – True to return unprotected header attributes, False for protected only

#### **json**

Accessor for json decorator data, or None.

#### **jws**

Accessor for JWS, or None.

#### **links**

Accessor for links decorator data, or None.



**sha256**

Accessor for sha256 decorator data, or None.

**sign** (*verkeys: Union[str, Sequence[str]]*, *wallet: aries\_cloudagent.wallet.base.BaseWallet*)

Sign base64 data value of attachment.

**Parameters**

- **verkeys** – verkey(s) of the signing party (in raw or DID key format)
- **wallet** – The wallet to use for the signature

**signatures**

Accessor for number of signatures.

**signed**

Accessor for signed content (payload), None for unsigned.

**verify** (*wallet: aries\_cloudagent.wallet.base.BaseWallet*) → bool

Verify the signature(s).

**Parameters** **wallet** – Wallet to use to verify signature

**Returns** True if verification succeeds else False

```
class aries_cloudagent.messaging.decorators.attach_decorator.AttachDecoratorData1JWS (*,
header:
    aries_cl
pro-
tected:
    str
    =
    None,
sig-
na-
ture:
    str)
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModel*

Single Detached JSON Web Signature for inclusion in attach decorator data.

**class Meta**

Bases: *object*

AttachDecoratorData1JWS metadata.

**schema\_class** = 'AttachDecoratorData1JWSSchema'

```
class aries_cloudagent.messaging.decorators.attach_decorator.AttachDecoratorData1JWSSchema
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModelSchema*

Single attach decorator data JWS schema.

**class Meta**

Bases: *object*

Single attach decorator data JWS schema metadata.

**model\_class**

alias of *AttachDecoratorData1JWS*

**header** = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None, ,

**protected** = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<ar

```
signature = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<ar
class aries_cloudagent.messaging.decorators.attach_decorator.AttachDecoratorDataJWS (*,
                                                    header:
                                                    aries_clo
                                                    =
                                                    None,
                                                    pro-
                                                    tected:
                                                    str
                                                    =
                                                    None,
                                                    sig-
                                                    na-
                                                    ture:
                                                    str
                                                    =
                                                    None,
                                                    sig-
                                                    na-
                                                    tures:
                                                    Se-
                                                    quence[a
                                                    =
                                                    None)
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModel*

Detached JSON Web Signature for inclusion in attach decorator data.

May hold one signature in flattened format, or multiple signatures in the “signatures” member.

**class Meta**

Bases: *object*

AttachDecoratorDataJWS metadata.

**schema\_class** = 'AttachDecoratorDataJWSSchema'

```
class aries_cloudagent.messaging.decorators.attach_decorator.AttachDecoratorDataJWSHeader (k
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModel*

Attach decorator data JWS header.

**class Meta**

Bases: *object*

AttachDecoratorDataJWS metadata.

**schema\_class** = 'AttachDecoratorDataJWSHeaderSchema'

```
class aries_cloudagent.messaging.decorators.attach_decorator.AttachDecoratorDataJWSHeaderSchema
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModelSchema*

Attach decorator data JWS header schema.

**class Meta**

Bases: *object*

Attach decorator data schema metadata.

```

    model_class
        alias of AttachDecoratorDataJWSHeader

    kid = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<aries_cl
class aries_cloudagent.messaging.decorators.attach_decorator.AttachDecoratorDataJWSSchema (
    Bases: aries_cloudagent.messaging.models.base.BaseModelSchema
    Schema for detached JSON Web Signature for inclusion in attach decorator data.

    class Meta
        Bases: object
        Metadata for schema for detached JWS for inclusion in attach deco data.

    model_class
        alias of AttachDecoratorDataJWS

    header = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None,
    protected = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<ar
    signature = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<ar
    signatures = <fields.List(default=<marshmallow.missing>, attribute=None, validate=None
    validate_single_xor_multi_sig (data: Mapping[KT, VT_co], **kwargs)
        Ensure model is for either 1 or many signatures, not mishmash of both.

class aries_cloudagent.messaging.decorators.attach_decorator.AttachDecoratorDataSchema (*args,
                                                                                       **kw
    Bases: aries_cloudagent.messaging.models.base.BaseModelSchema
    Attach decorator data schema.

    class Meta
        Bases: object
        Attach decorator data schema metadata.

    model_class
        alias of AttachDecoratorData

    base64_ = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<arie
    json_ = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, r
    jws_ = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None, re
    links_ = <fields.List(default=<marshmallow.missing>, attribute=None, validate=None, re
    sha256_ = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<arie
    validate_data_spec (data: Mapping[KT, VT_co], **kwargs)
        Ensure model chooses exactly one of base64, json, or links.

class aries_cloudagent.messaging.decorators.attach_decorator.AttachDecoratorSchema (*args,
                                                                                       **kwargs)
    Bases: aries_cloudagent.messaging.models.base.BaseModelSchema
    Attach decorator schema used in serialization/deserialization.

    class Meta
        Bases: object
        AttachDecoratorSchema metadata.

```

```
model_class
    alias of AttachDecorator

byte_count = <fields.Integer(default=<marshmallow.missing>, attribute=None, validate=None, re
data = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None, re
description = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None
filename = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None
ident = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, r
lastmod_time = <fields.String(default=<marshmallow.missing>, attribute=None, validate=
mime_type = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None

aries_cloudagent.messaging.decorators.attach_decorator.did_key(verkey: str) →
    str
    Qualify verkey into DID key if need be.

aries_cloudagent.messaging.decorators.attach_decorator.raw_key(verkey: str) →
    str
    Strip qualified key to raw key if need be.
```

## aries\_cloudagent.messaging.decorators.base module

Classes for managing a collection of decorators.

```
class aries_cloudagent.messaging.decorators.base.BaseDecoratorSet(models:
    dict =
    None)

Bases: collections.OrderedDict
Collection of decorators.

add_model(key: str, model: Type[aries_cloudagent.messaging.models.base.BaseModel])
    Add a registered decorator model.

copy() → aries_cloudagent.messaging.decorators.base.BaseDecoratorSet
    Return a copy of the decorator set.

extract_decorators(message: Mapping[KT, VT_co], schema:
    Type[<sphinx.ext.autodoc.importer._MockObject object at
    0x7f795197b400>] = None, serialized: bool = True, skip_attrs: Se-
    quence[str] = None) → collections.OrderedDict
    Extract decorators and return the remaining properties.

field(name: str) → aries_cloudagent.messaging.decorators.base.BaseDecoratorSet
    Access a named decorated field.

fields
    Accessor for the set of currently defined fields.

has_field(name: str) → bool
    Check for the existence of a named decorator field.

load_decorator(key: str, value, serialized=False)
    Convert a decorator value to its loaded representation.

models
    Accessor for the models dictionary.
```

**prefix**

Accessor for the decorator prefix.

**remove\_field** (*name: str*)

Remove a named decorated field.

**remove\_model** (*key: str*)

Remove a registered decorator model.

**to\_dict** (*prefix: str = None*) → collections.OrderedDict

Convert to a dictionary (serialize).

**Raises** BaseModelError – on decorator validation errors

```
exception aries_cloudagent.messaging.decorators.base.DecoratorError(*args,  
                                                                    er-  
                                                                    ror_code:  
                                                                    str      =  
                                                                    None,  
                                                                    **kwargs)
```

Bases: `aries_cloudagent.core.error.BaseError`

Base error for decorator issues.

**aries\_cloudagent.messaging.decorators.default module**

Default decorator set implementation.

```
class aries_cloudagent.messaging.decorators.default.DecoratorSet(models: dict  
                                                                = None)
```

Bases: `aries_cloudagent.messaging.decorators.base.BaseDecoratorSet`

Default decorator set implementation.

**aries\_cloudagent.messaging.decorators.localization\_decorator module**

The localization decorator (~110n) for message localization information.

```
class aries_cloudagent.messaging.decorators.localization_decorator.LocalizationDecorator(*,
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModel*

Class representing the localization decorator.

```
class Meta
```

Bases: *object*

LocalizationDecorator metadata.

```
schema_class = 'LocalizationDecoratorSchema'
```

```
class aries_cloudagent.messaging.decorators.localization_decorator.LocalizationDecoratorSchema
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModelSchema*

Localization decorator schema used in serialization/deserialization.

```
class Meta
```

Bases: *object*

LocalizationDecoratorSchema metadata.

```
model_class
```

alias of *LocalizationDecorator*

```
catalogs
```

Used by autodoc\_mock\_imports.

```
locale
```

Used by autodoc\_mock\_imports.

```
localizable
```

Used by autodoc\_mock\_imports.

## aries\_cloudagent.messaging.decorators.please\_ack\_decorator module

The please-ack decorator to request acknowledgement.

```

class aries_cloudagent.messaging.decorators.please_ack_decorator.PleaseAckDecorator (message_id: str,
                                             message: str,
                                             on: None,
                                             on: Sequence[str],
                                             on: None)

Bases: aries_cloudagent.messaging.models.base.BaseModel

Class representing the please-ack decorator.

class Meta
    Bases: object

    PleaseAckDecorator metadata.

    schema_class = 'PleaseAckDecoratorSchema'

class aries_cloudagent.messaging.decorators.please_ack_decorator.PleaseAckDecoratorSchema (message_id: str,
                                                message: str,
                                                on: None,
                                                on: Sequence[str],
                                                on: None)

Bases: aries_cloudagent.messaging.models.base.BaseModelSchema

PleaseAck decorator schema used in serialization/deserialization.

class Meta
    Bases: object

    PleaseAckDecoratorSchema metadata.

    model_class
        alias of PleaseAckDecorator

message_id = <fields.String (default=<marshmallow.missing>, attribute=None, validate=None)>
on = <fields.List (default=<marshmallow.missing>, attribute=None, validate=None, required=True)>

```

## aries\_cloudagent.messaging.decorators.signature\_decorator module

Model and schema for working with field signatures within message bodies.

```
class aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator(*,
signature_type: str = None, sig_data: str = None, signer: str = None)

Bases: aries_cloudagent.messaging.models.base.BaseModel
```

Class representing a field value signed by a known verkey.

**class Meta**

Bases: `object`

SignatureDecorator metadata.

**schema\_class** = 'SignatureDecoratorSchema'

**TYPE\_ED25519SHA512** = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/signature/1.0/ed25519Sha512\_'

**classmethod create** (value, signer: str, wallet: aries\_cloudagent.wallet.base.BaseWallet, timestamp=None) → aries\_cloudagent.messaging.decorators.signature\_decorator.SignatureDecorator

Create a Signature.

Sign a field value and return a newly constructed *SignatureDecorator* representing the resulting signature.

#### Parameters

- **value** – Value to sign
- **signer** – Verkey of the signing party
- **wallet** – The wallet to use for the signature

**Returns** The created *SignatureDecorator* object

**decode** () -> (<class 'object'>, <class 'int'>)

Decode the signature to its timestamp and value.

**Returns** A tuple of (decoded message, timestamp)

**verify** (wallet: aries\_cloudagent.wallet.base.BaseWallet) → bool

Verify the signature against the signer's public key.

**Parameters** **wallet** – Wallet to use to verify signature

**Returns** True if verification succeeds else False



```
class aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecoratorSchema (*args, **kwargs)
    Bases: aries_cloudagent.messaging.models.base.BaseModelSchema
    SignatureDecorator schema.

    class Meta
        Bases: object
        SignatureDecoratorSchema metadata.

    model_class
        alias of SignatureDecorator

    sig_data
        Used by autodoc_mock_imports.

    signature
        Used by autodoc_mock_imports.

    signature_type
        Used by autodoc_mock_imports.

    signer
        Used by autodoc_mock_imports.
```

## aries\_cloudagent.messaging.decorators.thread\_decorator module

A message decorator for threads.

A thread decorator identifies a message that may require additional context from previous messages.

```
class aries_cloudagent.messaging.decorators.thread_decorator.ThreadDecorator (*args, **kwargs)
    thid: str = None,
    pthid: str = None,
    sender_order: int = None,
    received_orders: Mapping[KT, VT_co] = None)

    Bases: aries_cloudagent.messaging.models.base.BaseModel
    Class representing thread decorator.

    class Meta
        Bases: object
        ThreadDecorator metadata.
```

```
    schema_class = 'ThreadDecoratorSchema'
```

**pthid**  
Accessor for parent thread identifier.

**Returns** This thread's *pthid*

**received\_orders**  
Get received orders.

**Returns** The highest sender\_order value that the sender has seen from other sender(s) on the thread.

**sender\_order**  
Get sender order.

**Returns** A number that tells where this message fits in the sequence of all messages that the current sender has contributed to this thread

**thid**  
Accessor for thread identifier.

**Returns** This thread's *thid*

```
class aries_cloudagent.messaging.decorators.thread_decorator.ThreadDecoratorSchema(*args,
                                                                                **kwargs)
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModelSchema*

Thread decorator schema used in serialization/deserialization.

```
class Meta
```

Bases: *object*

ThreadDecoratorSchema metadata.

```
    model_class
```

alias of *ThreadDecorator*

**pthid**  
Used by autodoc\_mock\_imports.

**received\_orders**  
Used by autodoc\_mock\_imports.

**sender\_order**  
Used by autodoc\_mock\_imports.

**thid**  
Used by autodoc\_mock\_imports.

## **aries\_cloudagent.messaging.decorators.timing\_decorator module**

The timing decorator (~timing).

This decorator allows the timing of agent messages to be communicated and constrained.

```

class aries_cloudagent.messaging.decorators.timing_decorator.TimingDecorator(*
    in_time:
    Union[str,
    date-
    time.datetime]
    =
    None,
    out_time:
    Union[str,
    date-
    time.datetime]
    =
    None,
    stale_time:
    Union[str,
    date-
    time.datetime]
    =
    None,
    expires_time:
    Union[str,
    date-
    time.datetime]
    =
    None,
    delay_milli:
    int
    =
    None,
    wait_until_time:
    Union[str,
    date-
    time.datetime]
    =
    None)

```

Bases: *aries\_cloudagent.messaging.models.base.BaseModel*

Class representing the timing decorator.

```

class Meta
    Bases: object
    TimingDecorator metadata.
    schema_class = 'TimingDecoratorSchema'

```

```

class aries_cloudagent.messaging.decorators.timing_decorator.TimingDecoratorSchema(*args,
    **kwargs)

```

Bases: *aries\_cloudagent.messaging.models.base.BaseModelSchema*

Timing decorator schema used in serialization/deserialization.

```

class Meta
    Bases: object
    TimingDecoratorSchema metadata.

```

```

    model_class
        alias of TimingDecorator

delay_milli
    Used by autodoc_mock_imports.

expires_time
    Used by autodoc_mock_imports.

in_time
    Used by autodoc_mock_imports.

out_time
    Used by autodoc_mock_imports.

stale_time
    Used by autodoc_mock_imports.

wait_until_time
    Used by autodoc_mock_imports.

```

## aries\_cloudagent.messaging.decorators.transport\_decorator module

The transport decorator (~transport).

This decorator allows changes to agent response behaviour and queue status updates.

```

class aries_cloudagent.messaging.decorators.transport_decorator.TransportDecorator (*,
                                                                 re-
                                                                 turn_route:
                                                                 str
                                                                 =
                                                                 None,
                                                                 re-
                                                                 turn_route:
                                                                 str
                                                                 =
                                                                 None,
                                                                 queued_me
                                                                 int
                                                                 =
                                                                 None)

```

Bases: *aries\_cloudagent.messaging.models.base.BaseModel*

Class representing the transport decorator.

```

class Meta
    Bases: object

    TransportDecorator metadata.

    schema_class = 'TransportDecoratorSchema'

```

```

class aries_cloudagent.messaging.decorators.transport_decorator.TransportDecoratorSchema (*,
                                                                 **

```

Bases: *aries\_cloudagent.messaging.models.base.BaseModelSchema*

Transport decorator schema used in serialization/deserialization.

```

class Meta
    Bases: object

```

TransportDecoratorSchema metadata.

**model\_class**

alias of *TransportDecorator*

**queued\_message\_count**

Used by autodoc\_mock\_imports.

**return\_route**

Used by autodoc\_mock\_imports.

**return\_route\_thread**

Used by autodoc\_mock\_imports.

## aries\_cloudagent.messaging.models package

### Submodules

### aries\_cloudagent.messaging.models.base module

Base classes for Models and Schemas.

**class** aries\_cloudagent.messaging.models.base.**BaseModel**

Bases: *abc.ABC*

Base model that provides convenience methods.

**class** **Meta**

Bases: *object*

BaseModel meta data.

**schema\_class** = **None**

**Schema**

Accessor for the model's schema class.

**Returns** The schema class

**classmethod** **deserialize** (*obj*)

Convert from JSON representation to a model instance.

**Parameters** *obj* – The dict to load into a model instance

**Returns** A model instance for this data

**classmethod** **from\_json** (*json\_repr: Union[str, bytes]*)

Parse a JSON string into a model instance.

**Parameters** *json\_repr* – JSON string

**Returns** A model instance representation of this JSON

**serialize** (*as\_string=False*) → dict

Create a JSON-compatible dict representation of the model instance.

**Parameters** *as\_string* – Return a string of JSON instead of a dict

**Returns** A dict representation of this model, or a JSON string if *as\_string* is True

**to\_json** () → str

Create a JSON representation of the model instance.

**Returns** A JSON representation of this message

**exception** `aries_cloudagent.messaging.models.base.BaseModelError` (\*args, error\_code: str = None, \*\*kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Base exception class for base model errors.

**class** `aries_cloudagent.messaging.models.base.BaseModelSchema` (\*args, \*\*kwargs)  
 Bases: `sphinx.ext.autodoc.importer._MockObject`

BaseModel schema.

**class** **Meta**

Bases: `object`

BaseModelSchema metadata.

**model\_class** = None

**ordered** = True

**skip\_values** = [None]

**Model**

Accessor for the schema's model class.

**Returns** The model class

**make\_model** (data: dict, \*\*kwargs)  
 Return model instance after loading.

**Returns** A model instance

**remove\_skipped\_values** (data, \*\*kwargs)  
 Remove values that are are marked to skip.

**Returns** Returns this modified data

**skip\_dump\_only** (data, \*\*kwargs)  
 Skip fields that are only expected during serialization.

**Parameters** **data** – The incoming data to clean

**Returns** The modified data

`aries_cloudagent.messaging.models.base.resolve_class` (the\_cls, relative\_cls: type = None)

Resolve a class.

**Parameters**

- **the\_cls** – The class to resolve
- **relative\_cls** – Relative class to resolve from

**Returns** The resolved class

**Raises** `ClassNotFoundError` – If the class could not be loaded

`aries_cloudagent.messaging.models.base.resolve_meta_property` (obj, prop\_name: str, defval=None)

Resolve a meta property.

**Parameters**

- **prop\_name** – The property to resolve
- **defval** – The default value

**Returns** The meta property

## aries\_cloudagent.messaging.models.base\_record module

Classes for BaseStorage-based record management.

```
class aries_cloudagent.messaging.models.base_record.BaseExchangeRecord(id:
    str
    =
    None,
    state:
    str
    =
    None,
    *,
    trace:
    bool
    =
    False,
    **kwargs)
```

Bases: *aries\_cloudagent.messaging.models.base\_record.BaseRecord*

Represents a base record with event tracing capability.

```
class aries_cloudagent.messaging.models.base_record.BaseExchangeSchema(*args,
    **kwargs)
```

Bases: *aries\_cloudagent.messaging.models.base\_record.BaseRecordSchema*

Base schema for exchange records.

```
class Meta
```

Bases: *object*

BaseExchangeSchema metadata.

```
model_class
```

alias of *BaseExchangeRecord*

```
trace = <fields.Boolean(default=False, attribute=None, validate=None, required=False,
```

```
class aries_cloudagent.messaging.models.base_record.BaseRecord(id:      str  =
    None,      state:
    str  =  None,
    *,  created_at:
    Union[str, date-
    time.datetime]
    =  None,  up-
    dated_at:
    Union[str, date-
    time.datetime]
    =  None)
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModel*

Represents a single storage record.

```
CACHE_ENABLED = False
```

```
CACHE_TTL = 60
LOG_STATE_FLAG = None

class Meta
    Bases: object

    BaseRecord metadata.

RECORD_ID_NAME = 'id'
RECORD_TYPE = None
TAG_NAMES = {'state'}
WEBHOOK_TOPIC = None

classmethod cache_key (record_id: str, record_type: str = None)
    Assemble a cache key.

    Parameters
        • record_id – The record identifier
        • The cache type identifier, defaulting to RECORD_TYPE
          (record_type) –

clear_cached (context: aries_cloudagent.config.injection_context.InjectionContext)
    Clear the cached value of this record, if any.

classmethod clear_cached_key (context: aries_cloudagent.config.injection_context.InjectionContext,
                             cache_key: str)
    Shortcut method to clear a cached key value, if any.

    Parameters
        • context – The injection context to use
        • cache_key – The unique cache identifier

delete_record (context: aries_cloudagent.config.injection_context.InjectionContext)
    Remove the stored record.

    Parameters context – The injection context to use

classmethod from_storage (record_id: str, record: Mapping[str, Any])
    Initialize a record from its stored representation.

    Parameters
        • record_id – The unique record identifier
        • record – The stored representation

classmethod get_cached_key (context: aries_cloudagent.config.injection_context.InjectionContext,
                           cache_key: str)
    Shortcut method to fetch a cached key value.

    Parameters
        • context – The injection context to use
        • cache_key – The unique cache identifier

classmethod get_tag_map () → Mapping[str, str]
    Accessor for the set of defined tags.
```



**classmethod log\_state** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, msg: str, params: dict = None, override: bool = False*)  
 Print a message with increased visibility (for testing).

**post\_save** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, new\_record: bool, last\_state: str, webhook: bool = None*)  
 Perform post-save actions.

#### Parameters

- **context** – The injection context to use
- **new\_record** – Flag indicating if the record was just created
- **last\_state** – The previous state value
- **webhook** – Adjust whether the webhook is called

**classmethod prefix\_tag\_filter** (*tag\_filter: dict*)  
 Prefix unencrypted tags used in the tag filter.

**classmethod query** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, tag\_filter: dict = None, post\_filter\_positive: dict = None, post\_filter\_negative: dict = None*) → Sequence[aries\_cloudagent.messaging.models.base\_record.BaseRecord]  
 Query stored records.

#### Parameters

- **context** – The injection context to use
- **tag\_filter** – An optional dictionary of tag filter clauses
- **post\_filter\_positive** – Additional value filters to apply matching positively
- **post\_filter\_negative** – Additional value filters to apply matching negatively

**record\_tags**  
 Accessor to define implementation-specific tags.

**record\_value**  
 Accessor to define custom properties for the JSON record value.

**classmethod retrieve\_by\_id** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, record\_id: str, cached: bool = True*) → aries\_cloudagent.messaging.models.base\_record.BaseRecord  
 Retrieve a stored record by ID.

#### Parameters

- **context** – The injection context to use
- **record\_id** – The ID of the record to find
- **cached** – Whether to check the cache for this record

**classmethod retrieve\_by\_tag\_filter** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, tag\_filter: dict, post\_filter: dict = None*) → aries\_cloudagent.messaging.models.base\_record.BaseRecord

Retrieve a record by tag filter.

#### Parameters

- **context** – The injection context to use
- **tag\_filter** – The filter dictionary to apply

- **post\_filter** – Additional value filters to apply after retrieval

**save** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, \*, reason: str = None, log\_params: Mapping[str, Any] = None, log\_override: bool = False, webhook: bool = None*)  
→ str  
Persist the record to storage.

#### Parameters

- **context** – The injection context to use
- **reason** – A reason to add to the log
- **log\_params** – Additional parameters to log
- **webhook** – Flag to override whether the webhook is sent

**send\_webhook** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, payload: Any, topic: str = None*)  
Send a standard webhook.

#### Parameters

- **context** – The injection context to use
- **payload** – The webhook payload
- **topic** – The webhook topic, defaulting to WEBHOOK\_TOPIC

**classmethod set\_cached\_key** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, cache\_key: str, value: Any, ttl=None*)  
Shortcut method to set a cached key value.

#### Parameters

- **context** – The injection context to use
- **cache\_key** – The unique cache identifier
- **value** – The value to cache
- **ttl** – The cache ttl

**storage\_record**  
Accessor for a *StorageRecord* representing this record.

**classmethod strip\_tag\_prefix** (*tags: dict*)  
Strip tilde from unencrypted tag names.

**tags**  
Accessor for the record tags generated for this record.

**value**  
Accessor for the JSON record value generated for this record.

**webhook\_payload**  
Return a JSON-serialized version of the record for the webhook.

**webhook\_topic**  
Return the webhook topic value.

**class** `aries_cloudagent.messaging.models.base_record.BaseRecordSchema` (*\*args, \*\*kwargs*)  
Bases: `aries_cloudagent.messaging.models.base.BaseModelSchema`  
Schema to allow serialization/deserialization of base records.

```

class Meta
    Bases: object

    BaseRecordSchema metadata.

    created_at = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<a
    state = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, r
    updated_at = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<a
aries_cloudagent.messaging.models.base_record.match_post_filter(record: dict,
                                post_filter:
                                dict, positive:
                                bool = True)
                                → bool

```

Determine if a record value matches the post-filter.

#### Parameters

- **record** – record to check
- **post\_filter** – filter to apply (empty or None filter matches everything)
- **positive** – whether matching all filter criteria positively or negatively

## aries\_cloudagent.messaging.schemas package

### Submodules

#### aries\_cloudagent.messaging.schemas.routes module

Credential schema admin routes.

```

class aries_cloudagent.messaging.schemas.routes.SchemaGetResultsSchema(*,
                                only=None,
                                ex-
                                clude=(),
                                many=False,
                                con-
                                text=None,
                                load_only=(),
                                dump_only=(),
                                par-
                                tial=False,
                                un-
                                known=None)

```

Bases: `marshmallow.schema.Schema`

Results schema for schema get request.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```

class aries_cloudagent.messaging.schemas.routes.SchemaSchema (*,
                                                                only=None,
                                                                exclude=(),
                                                                many=False,
                                                                context=None,
                                                                load_only=(),
                                                                dump_only=(),
                                                                partial=False,
                                                                unknown=None)

    Bases: marshmallow.schema.Schema

    Content for returned schema.

    opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.messaging.schemas.routes.SchemaSendRequestSchema (*,
                                                                            only=None,
                                                                            ex-
                                                                            clude=(),
                                                                            many=False,
                                                                            con-
                                                                            text=None,
                                                                            load_only=(),
                                                                            dump_only=(),
                                                                            par-
                                                                            tial=False,
                                                                            un-
                                                                            known=None)

    Bases: marshmallow.schema.Schema

    Request schema for schema send request.

    opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.messaging.schemas.routes.SchemaSendResultsSchema (*,
                                                                              only=None,
                                                                              ex-
                                                                              clude=(),
                                                                              many=False,
                                                                              con-
                                                                              text=None,
                                                                              load_only=(),
                                                                              dump_only=(),
                                                                              par-
                                                                              tial=False,
                                                                              un-
                                                                              known=None)

    Bases: marshmallow.schema.Schema

    Results schema for schema send request.

    opts = <marshmallow.schema.SchemaOpts object>

```

```
class aries_cloudagent.messaging.schemas.routes.SchemasCreatedResultsSchema (*,
                                                                    only=None,
                                                                    ex-
                                                                    clude=(),
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False,
                                                                    un-
                                                                    known=None)
```

Bases: `marshmallow.schema.Schema`

Results schema for a schemas-created request.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
aries_cloudagent.messaging.schemas.routes.register (app:
                                                                    <sphinx.ext.autodoc.importer._MockObject
                                                                    object at 0x7f79514cfa58>)
```

Register routes.

```
aries_cloudagent.messaging.schemas.routes.schemas_created (request:
                                                                    <sphinx.ext.autodoc.importer._MockObject
                                                                    object          at
                                                                    0x7f79514cfa58>)
```

Request handler for retrieving schemas that current agent created.

**Parameters** **request** – aiohttp request object

**Returns** The identifiers of matching schemas

```
aries_cloudagent.messaging.schemas.routes.schemas_get_schema (request:
                                                                    <sphinx.ext.autodoc.importer._MockObject
                                                                    object          at
                                                                    0x7f79514cfa58>)
```

Request handler for sending a credential offer.

**Parameters** **request** – aiohttp request object

**Returns** The schema details.

```
aries_cloudagent.messaging.schemas.routes.schemas_send_schema (request:
                                                                    <sphinx.ext.autodoc.importer._MockObject
                                                                    object          at
                                                                    0x7f79514cfa58>)
```

Request handler for sending a credential offer.

**Parameters** **request** – aiohttp request object

**Returns** The schema id sent

## aries\_cloudagent.messaging.schemas.util module

Schema utilities.

## Submodules

### aries\_cloudagent.messaging.agent\_message module

Agent message base class and schema.

```
class aries_cloudagent.messaging.agent_message.AgentMessage (_id: str = None,
                                                            _decorators:
                                                            aries_cloudagent.messaging.decorators.base.
                                                            = None)
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModel*

Agent message base class.

#### Handler

Accessor for the agent message's handler class.

**Returns** Handler class

#### class Meta

Bases: *object*

AgentMessage metadata.

**handler\_class** = None

**message\_type** = None

**schema\_class** = None

```
add_trace_decorator (target: str = 'log', full_thread: bool = True)
```

Create a new trace decorator.

#### Parameters

- **target** – The trace target
- **full\_thread** – Full thread flag

```
add_trace_report (val: Union[aries_cloudagent.messaging.decorators.trace_decorator.TraceReport,
                             dict])
```

Append a new trace report.

**Parameters val** – The trace target

```
assign_thread_from (msg: aries_cloudagent.messaging.agent_message.AgentMessage)
```

Copy thread information from a previous message.

**Parameters msg** – The received message containing optional thread information

```
assign_thread_id (thid: str, pthid: str = None)
```

Assign a specific thread ID.

#### Parameters

- **thid** – The thread identifier
- **pthid** – The parent thread identifier

```
assign_trace_decorator (context, trace)
```

Copy trace from a json structure.

**Parameters trace** – string containing trace json stucture

```
assign_trace_from (msg: aries_cloudagent.messaging.agent_message.AgentMessage)
```

Copy trace information from a previous message.

**Parameters** `msg` – The received message containing optional trace information

**get\_signature** (*field\_name: str*) → `aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator`  
Get the signature for a named field.

**Parameters** `field_name` – Field name to get the signature for

**Returns** A `SignatureDecorator` for the requested field name

**set\_signature** (*field\_name: str, signature: aries\_cloudagent.messaging.decorators.signature\_decorator.SignatureDecorator*)  
Add or replace the signature for a named field.

**Parameters**

- **field\_name** – Field to set signature on
- **signature** – Signature for the field

**sign\_field** (*field\_name: str, signer\_verkey: str, wallet: aries\_cloudagent.wallet.base.BaseWallet, timestamp=None*) → `aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator`  
Create and store a signature for a named field.

**Parameters**

- **field\_name** – Field to sign
- **signer\_verkey** – Verkey of signer
- **wallet** – Wallet to use for signature
- **timestamp** – Optional timestamp for signature

**Returns** A `SignatureDecorator` for newly created signature

**Raises** `ValueError` – If `field_name` doesn't exist on this message

**verify\_signatures** (*wallet: aries\_cloudagent.wallet.base.BaseWallet*) → `bool`  
Verify all associated field signatures.

**Parameters** `wallet` – Wallet to use in verification

**Returns** True if all signatures verify, else false

**verify\_signed\_field** (*field\_name: str, wallet: aries\_cloudagent.wallet.base.BaseWallet, signer\_verkey: str = None*) → `str`  
Verify a specific field signature.

**Parameters**

- **field\_name** – The field name to verify
- **wallet** – Wallet to use for the verification
- **signer\_verkey** – Verkey of signer to use

**Returns** The verkey of the signer

**Raises**

- `ValueError` – If `field_name` does not exist on this message
- `ValueError` – If the verification fails
- `ValueError` – If the verkey of the signature does not match the provided verkey

```
exception aries_cloudagent.messaging.agent_message.AgentMessageError (*args,  
                                                                    er-  
                                                                    ror_code:  
                                                                    str =  
                                                                    None,  
                                                                    **kwargs)
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModelError*

Base exception for agent message issues.

```
class aries_cloudagent.messaging.agent_message.AgentMessageSchema (*args,  
                                                                    **kwargs)
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModelSchema*

AgentMessage schema.

```
class Meta
```

Bases: *object*

AgentMessageSchema metadata.

**model\_class** = *None*

**signed\_fields** = *None*

```
check_dump_decorators (obj, **kwargs)
```

Pre-dump hook to validate and load the message decorators.

**Parameters** *obj* – The AgentMessage object

**Raises** *BaseModelError* – If a decorator does not validate

```
dump_decorators (data, **kwargs)
```

Post-dump hook to write the decorators to the serialized output.

**Parameters** *obj* – The serialized data

**Returns** The modified data

```
extract_decorators (data: Mapping[KT, VT_co], **kwargs)
```

Pre-load hook to extract the decorators and check the signed fields.

**Parameters** *data* – Incoming data to parse

**Returns** Parsed and modified data

**Raises**

- *ValidationError* – If a field signature does not correlate
- to a field in the message
- *ValidationError* – If the message defines both a field signature
- and a value for the same field
- *ValidationError* – If there is a missing field signature

```
populate_decorators (obj, **kwargs)
```

Post-load hook to populate decorators on the message.

**Parameters** *obj* – The AgentMessage object

**Returns** The AgentMessage object with populated decorators

```
replace_signatures (data, **kwargs)
```

Post-dump hook to write the signatures to the serialized output.



**Parameters** `obj` – The serialized data

**Returns** The modified data

## `aries_cloudagent.messaging.base_handler` module

A Base handler class for all message handlers.

**class** `aries_cloudagent.messaging.base_handler.BaseHandler`

Bases: `abc.ABC`

Abstract base class for handlers.

**handle** (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:* `aries_cloudagent.messaging.responder.BaseResponder`)  
Abstract method for handler logic.

**Parameters**

- **context** – Request context object
- **responder** – A responder object

**exception** `aries_cloudagent.messaging.base_handler.HandlerException` (*\*args*, *error\_code:* `str` = `None`, *\*\*kwargs*)

Bases: `aries_cloudagent.core.error.BaseError`

Exception base class for generic handler errors.

## `aries_cloudagent.messaging.error` module

Messaging-related error classes and codes.

**exception** `aries_cloudagent.messaging.error.MessageParseError` (*\*args*, *error\_code:* `str` = `None`, *\*\*kwargs*)

Bases: `aries_cloudagent.core.error.BaseError`

Message parse error.

**error\_code** = `'message_parse_error'`

**exception** `aries_cloudagent.messaging.error.MessagePrepareError` (*\*args*, *error\_code:* `str` = `None`, *\*\*kwargs*)

Bases: `aries_cloudagent.core.error.BaseError`

Message preparation error.

**error\_code** = `'message_prepare_error'`

## `aries_cloudagent.messaging.request_context` module

Request context class.

A request context provides everything required by handlers and other parts of the system to process a message.

```
class aries_cloudagent.messaging.request_context.RequestContext (*,
                                                                base_context:
                                                                aries_cloudagent.config.injection_context.InjectionContext
                                                                = None, settings: Mapping[str, object] =
                                                                None)
```

Bases: `aries_cloudagent.config.injection_context.InjectionContext`

Context established by the Conductor and passed into message handlers.

**connection\_ready**

Accessor for the flag indicating an active connection with the sender.

**Returns** True if the connection is active, else False

**connection\_record**

Accessor for the related connection record.

**copy** () → `aries_cloudagent.messaging.request_context.RequestContext`

Produce a copy of the request context instance.

**default\_endpoint**

Accessor for the default agent endpoint (from agent config).

**Returns** The default agent endpoint

**default\_label**

Accessor for the default agent label (from agent config).

**Returns** The default label

**message**

Accessor for the deserialized message instance.

**Returns** This context's agent message

**message\_receipt**

Accessor for the message receipt information.

**Returns** This context's message receipt information

## aries\_cloudagent.messaging.responder module

A message responder.

The responder is provided to message handlers to enable them to send a new message in response to the message being handled.

```
class aries_cloudagent.messaging.responder.BaseResponder (*, connection_id: str =
                                                                None, reply_session_id:
                                                                str = None, reply_to_verkey:
                                                                str = None)
```

Bases: `abc.ABC`

Interface for message handlers to send responses.

**create\_outbound** (*message: Union[aries\_cloudagent.messaging.agent\_message.AgentMessage, str, bytes], \*, connection\_id: str = None, reply\_session\_id: str = None, reply\_thread\_id: str = None, reply\_to\_verkey: str = None, target: aries\_cloudagent.connections.models.connection\_target.ConnectionTarget = None, target\_list: Sequence[aries\_cloudagent.connections.models.connection\_target.ConnectionTarget] = None, to\_session\_only: bool = False*) → *aries\_cloudagent.transport.outbound.message.OutboundMessage*  
 Create an OutboundMessage from a message payload.

**send** (*message: Union[aries\_cloudagent.messaging.agent\_message.AgentMessage, str, bytes], \*\*kwargs*)  
 Convert a message to an OutboundMessage and send it.

**send\_outbound** (*message: aries\_cloudagent.transport.outbound.message.OutboundMessage*)  
 Send an outbound message.

**Parameters message** – The *OutboundMessage* to be sent

**send\_reply** (*message: Union[aries\_cloudagent.messaging.agent\_message.AgentMessage, str, bytes], \*, connection\_id: str = None, target: aries\_cloudagent.connections.models.connection\_target.ConnectionTarget = None, target\_list: Sequence[aries\_cloudagent.connections.models.connection\_target.ConnectionTarget] = None*)  
 Send a reply to an incoming message.

**Parameters**

- **message** – the *AgentMessage*, or pre-packed str or bytes to reply with
- **connection\_id** – optionally override the target connection ID
- **target** – optionally specify a *ConnectionTarget* to send to

**Raises** *ResponderError* – If there is no active connection

**send\_webhook** (*topic: str, payload: dict*)  
 Dispatch a webhook.

**Parameters**

- **topic** – the webhook topic identifier
- **payload** – the webhook payload value

**class** *aries\_cloudagent.messaging.responder.MockResponder*  
 Bases: *aries\_cloudagent.messaging.responder.BaseResponder*  
 Mock responder implementation for use by tests.

**send** (*message: Union[aries\_cloudagent.messaging.agent\_message.AgentMessage, str, bytes], \*\*kwargs*)  
 Convert a message to an OutboundMessage and send it.

**send\_outbound** (*message: aries\_cloudagent.transport.outbound.message.OutboundMessage*)  
 Send an outbound message.

**send\_reply** (*message: Union[aries\_cloudagent.messaging.agent\_message.AgentMessage, str, bytes], \*\*kwargs*)  
 Send a reply to an incoming message.

**send\_webhook** (*topic: str, payload: dict*)  
 Send an outbound message.

**exception** `aries_cloudagent.messaging.responder.ResponderError` (\*args, error\_code: str = None, \*\*kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Responder error.

## **aries\_cloudagent.messaging.util module**

Utils for messages.

`aries_cloudagent.messaging.util.canon` (*raw\_attr\_name: str*) → str  
Canonicalize input attribute name for indy proofs and credential offers.

**Parameters** *raw\_attr\_name* – raw attribute name

**Returns** canonicalized attribute name

`aries_cloudagent.messaging.util.datetime_now` () → datetime.datetime  
Timestamp in UTC.

`aries_cloudagent.messaging.util.datetime_to_str` (*dt: Union[str, datetime.datetime]*) → str  
Convert a datetime object to an indy-standard datetime string.

**Parameters** *dt* – May be a string or datetime to allow automatic conversion

`aries_cloudagent.messaging.util.encode` (*orig: Any*) → str  
Encode a credential value as an int.

Encode credential attribute value, purely stringifying any int32 and leaving numeric int32 strings alone, but mapping any other input to a stringified 256-bit (but not 32-bit) integer. Predicates in indy-sdk operate on int32 values properly only when their encoded values match their raw values.

**Parameters** *orig* – original value to encode

**Returns** encoded value

`aries_cloudagent.messaging.util.epoch_to_str` (*epoch: int*) → str  
Convert epoch seconds to indy-standard datetime string.

**Parameters** *epoch* – epoch seconds

`aries_cloudagent.messaging.util.str_to_datetime` (*dt: Union[str, datetime.datetime]*) → datetime.datetime  
Convert an indy-standard datetime string to a datetime.

Using a fairly lax regex pattern to match slightly different formats. In Python 3.7 `datetime.fromisoformat` might be used.

**Parameters** *dt* – May be a string or datetime to allow automatic conversion

`aries_cloudagent.messaging.util.str_to_epoch` (*dt: Union[str, datetime.datetime]*) → int  
Convert an indy-standard datetime string to epoch seconds.

**Parameters** *dt* – May be a string or datetime to allow automatic conversion

`aries_cloudagent.messaging.util.time_now` () → str  
Timestamp in ISO format.

**aries\_cloudagent.messaging.valid module**

Validators for schema fields.

```
class aries_cloudagent.messaging.valid.Base58SHA256Hash
```

Bases: sphinx.ext.autodoc.importer.\_MockObject

Validate value against base58 encoding of SHA-256 hash.

```
EXAMPLE = 'H3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV'
```

```
PATTERN = '^(<sphinx.ext.autodoc.importer._MockObject object>){43,44}$'
```

```
class aries_cloudagent.messaging.valid.Base64
```

Bases: sphinx.ext.autodoc.importer.\_MockObject

Validate base64 value.

```
EXAMPLE = 'ey4uLn0='
```

```
PATTERN = '^([a-zA-Z0-9+/]{0,2})$'
```

```
class aries_cloudagent.messaging.valid.Base64URL
```

Bases: sphinx.ext.autodoc.importer.\_MockObject

Validate base64 value.

```
EXAMPLE = 'ey4uLn0='
```

```
PATTERN = '^([-_a-zA-Z0-9]{0,2})$'
```

```
class aries_cloudagent.messaging.valid.Base64URLNoPad
```

Bases: sphinx.ext.autodoc.importer.\_MockObject

Validate base64 value.

```
EXAMPLE = 'ey4uLn0'
```

```
PATTERN = '^([-_a-zA-Z0-9]*)$'
```

```
class aries_cloudagent.messaging.valid.DIDKey
```

Bases: sphinx.ext.autodoc.importer.\_MockObject

Validate value against DID key specification.

```
EXAMPLE = 'did:key:z6MkpTHR8VNsBxYAAWHut2Geadd9jSwuBV8xRoAnwWsdvktH'
```

```
PATTERN = '^did:key:z(<sphinx.ext.autodoc.importer._MockObject object>)+$'
```

```
class aries_cloudagent.messaging.valid.IndyCredDefId
```

Bases: sphinx.ext.autodoc.importer.\_MockObject

Validate value against indy credential definition identifier specification.

```
EXAMPLE = 'WgWxqztrNooG92RXvxSTWv:3:CL:20:tag'
```

```
PATTERN = '^([<sphinx.ext.autodoc.importer._MockObject object>]{21,22}):3:CL:([1-9][0-9]{0,2})$'
```

```
class aries_cloudagent.messaging.valid.IndyDID
```

Bases: sphinx.ext.autodoc.importer.\_MockObject

Validate value against indy DID.

```
EXAMPLE = 'WgWxqztrNooG92RXvxSTWv'
```

```
PATTERN = '^((did:sov:)?(<sphinx.ext.autodoc.importer._MockObject object>){21,22})$'
```

```
class aries_cloudagent.messaging.valid.IndyISO8601DateTime
```

```
    Bases: sphinx.ext.autodoc.importer._MockObject
```

```
    Validate value against ISO 8601 datetime format, indy profile.
```

```
    EXAMPLE = '2020-04-21 20:19:54Z'
```

```
    PATTERN = '^\\d{4}-\\d{2}-\\d{2}[T ]\\d{2}:\\d{2}:\\d{2}(?:\\.(?:\\d{1,6})?)?)?(?:
```

```
class aries_cloudagent.messaging.valid.IndyPredicate
```

```
    Bases: sphinx.ext.autodoc.importer._MockObject
```

```
    Validate value against indy predicate.
```

```
    EXAMPLE = '>='
```

```
class aries_cloudagent.messaging.valid.IndyRawPublicKey
```

```
    Bases: sphinx.ext.autodoc.importer._MockObject
```

```
    Validate value against indy (Ed25519VerificationKey2018) raw public key.
```

```
    EXAMPLE = 'H3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV'
```

```
    PATTERN = '^(<sphinx.ext.autodoc.importer._MockObject object>){43,44}$'
```

```
class aries_cloudagent.messaging.valid.IndyRevRegId
```

```
    Bases: sphinx.ext.autodoc.importer._MockObject
```

```
    Validate value against indy revocation registry identifier specification.
```

```
    EXAMPLE = 'WgWxqztrNooG92RXvxSTWv:4:WgWxqztrNooG92RXvxSTWv:3:CL:20:tag:CL_ACCUM:0'
```

```
    PATTERN = '^([<sphinx.ext.autodoc.importer._MockObject object>]{21,22}):4:([<sphinx.ex
```

```
class aries_cloudagent.messaging.valid.IndySchemaId
```

```
    Bases: sphinx.ext.autodoc.importer._MockObject
```

```
    Validate value against indy schema identifier specification.
```

```
    EXAMPLE = 'WgWxqztrNooG92RXvxSTWv:2:schema_name:1.0'
```

```
    PATTERN = '^(<sphinx.ext.autodoc.importer._MockObject object>){21,22}:2:.+: [0-9.]+$'
```

```
class aries_cloudagent.messaging.valid.IndyVersion
```

```
    Bases: sphinx.ext.autodoc.importer._MockObject
```

```
    Validate value against indy version specification.
```

```
    EXAMPLE = '1.0'
```

```
    PATTERN = '^ [0-9.]+$'
```

```
class aries_cloudagent.messaging.valid.IntEpoch
```

```
    Bases: sphinx.ext.autodoc.importer._MockObject
```

```
    Validate value against (integer) epoch format.
```

```
    EXAMPLE = 1587500394
```

```
class aries_cloudagent.messaging.valid.JSONWebToken
```

```
    Bases: sphinx.ext.autodoc.importer._MockObject
```

```
    Validate JSON Web Token.
```

```
    EXAMPLE = 'eyJhbGciOiJIJFZERTQ5Sj9.eyJhIjogIjAifQ.dBjftJeZ4CVP-mB92K27uhbUJU1p1r_wW1gFWFO
```

```
    PATTERN = '^[_a-zA-Z0-9]*\\. [_a-zA-Z0-9]*\\. [_a-zA-Z0-9]*$'
```

```

class aries_cloudagent.messaging.valid.JWSHeaderKid
    Bases: sphinx.ext.autodoc.importer._MockObject
    Validate value against JWS header kid.

    EXAMPLE = 'did:sov:LjgpST2rjsoxYegQDRm7EL#keys-4'
    PATTERN = '^did:(?:key:z[<sphinx.ext.autodoc.importer._MockObject object>]+|sov:[<sphinx.ext.autodoc.importer._MockObject object>]+'

class aries_cloudagent.messaging.valid.SHA256Hash
    Bases: sphinx.ext.autodoc.importer._MockObject
    Validate (binhex-encoded) SHA256 value.

    EXAMPLE = '617a48c7c8afe0521efdc03e5bb0ad9e655893e6b4b51f0e794d70fba132aacb'
    PATTERN = '^[a-fA-F0-9+/{64}]$'

class aries_cloudagent.messaging.valid.UUIDFour
    Bases: sphinx.ext.autodoc.importer._MockObject
    Validate UUID4: 8-4-4-12 hex digits, the 13th of which being 4.

    EXAMPLE = '3fa85f64-5717-4562-b3fc-2c963f66afa6'
    PATTERN = '[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-4[a-fA-F0-9]{3}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}'

```

## 1.1.10 aries\_cloudagent.storage package

### Submodules

#### aries\_cloudagent.storage.base module

Abstract base classes for non-secrets storage.

```

class aries_cloudagent.storage.base.BaseStorage
    Bases: abc.ABC
    Abstract Non-Secrets interface.

    add_record(record: aries_cloudagent.storage.record.StorageRecord)
        Add a new record to the store.

        Parameters record – StorageRecord to be stored

    delete_record(record: aries_cloudagent.storage.record.StorageRecord)
        Delete an existing record.

        Parameters record – StorageRecord to delete

    delete_record_tags(record: aries_cloudagent.storage.record.StorageRecord, tags: typing.Sequence, typing.Mapping)
        Update an existing stored record's tags.

        Parameters
        • record – StorageRecord to delete
        • tags – Tags

    get_record(record_type: str, record_id: str, options: Mapping[KT, VT_co] = None) → aries_cloudagent.storage.record.StorageRecord
        Fetch a record from the store by type and ID.

        Parameters

```

- **record\_type** – The record type
- **record\_id** – The record id
- **options** – A dictionary of backend-specific options

**Returns** A *StorageRecord* instance

**search\_records** (*type\_filter*: str, *tag\_query*: Mapping[KT, VT\_co] = None, *page\_size*: int = None, *options*: Mapping[KT, VT\_co] = None) → aries\_cloudagent.storage.base.BaseStorageRecordSearch  
Create a new record query.

**Parameters**

- **type\_filter** – Filter string
- **tag\_query** – Tags to query
- **page\_size** – Page size
- **options** – Dictionary of backend-specific options

**Returns** An instance of *BaseStorageRecordSearch*

**update\_record\_tags** (*record*: aries\_cloudagent.storage.record.StorageRecord, *tags*: Mapping[KT, VT\_co])  
Update an existing stored record's tags.

**Parameters**

- **record** – *StorageRecord* to update
- **tags** – New tags

**update\_record\_value** (*record*: aries\_cloudagent.storage.record.StorageRecord, *value*: str)  
Update an existing stored record's value.

**Parameters**

- **record** – *StorageRecord* to update
- **value** – The new value

```
class aries_cloudagent.storage.base.BaseStorageRecordSearch (store:
                                                                aries_cloudagent.storage.base.BaseStorage,
                                                                type_filter: str,
                                                                tag_query: Mapping[KT, VT_co],
                                                                page_size: int =
                                                                None, options: Mapping[KT, VT_co] =
                                                                None)
```

Bases: `abc.ABC`

Represent an active stored records search.

**close** ()  
Dispose of the search query.

**fetch** (*max\_count*: int) → Sequence[aries\_cloudagent.storage.record.StorageRecord]  
Fetch the next list of results from the store.

**Parameters** **max\_count** – Max number of records to return

**Returns** A list of *StorageRecord*



**fetch\_all()** → Sequence[aries\_cloudagent.storage.record.StorageRecord]

Fetch all records from the query.

**fetch\_single()** → aries\_cloudagent.storage.record.StorageRecord

Fetch a single query result.

**handle**

Handle a search request.

**open()**

Start the search query.

**opened**

Accessor for open state.

**Returns** True if opened, else False

**option** (name: str, default=None)

Fetch a named search option, if defined.

**Returns** The option value or default

**options**

Accessor for the search options.

**Returns** The search options

**page\_size**

Accessor for page size.

**Returns** The page size

**store**

BaseStorage backend for this implementation.

**Returns** The BaseStorage implementation being used

**tag\_query**

Accessor for tag query.

**Returns** The tag query

**type\_filter**

Accessor for type filter.

**Returns** The type filter

## aries\_cloudagent.storage.basic module

Basic in-memory storage implementation (non-wallet).

```
class aries_cloudagent.storage.basic.BasicStorage (_wallet:
                                                    aries_cloudagent.wallet.base.BaseWallet
                                                    = None)
```

Bases: *aries\_cloudagent.storage.base.BaseStorage*

Basic in-memory storage class.

**add\_record** (record: aries\_cloudagent.storage.record.StorageRecord)

Add a new record to the store.

**Parameters** **record** – StorageRecord to be stored

**Raises**

- `StorageError` – If no record is provided
- `StorageError` – If the record has no ID

**delete\_record** (*record: aries\_cloudagent.storage.record.StorageRecord*)

Delete a record.

**Parameters** **record** – *StorageRecord* to delete

**Raises** `StorageNotFoundError` – If record not found

**delete\_record\_tags** (*record: aries\_cloudagent.storage.record.StorageRecord, tags: (typing.Sequence, typing.Mapping)*)

Update an existing stored record's tags.

**Parameters**

- **record** – *StorageRecord* to delete
- **tags** – Tags

**Raises** `StorageNotFoundError` – If record not found

**get\_record** (*record\_type: str, record\_id: str, options: Mapping[KT, VT\_co] = None*) → *aries\_cloudagent.storage.record.StorageRecord*

Fetch a record from the store by type and ID.

**Parameters**

- **record\_type** – The record type
- **record\_id** – The record id
- **options** – A dictionary of backend-specific options

**Returns** A *StorageRecord* instance

**Raises** `StorageNotFoundError` – If the record is not found

**search\_records** (*type\_filter: str, tag\_query: Mapping[KT, VT\_co] = None, page\_size: int = None, options: Mapping[KT, VT\_co] = None*) → *aries\_cloudagent.storage.basic.BasicStorageRecordSearch*

Search stored records.

**Parameters**

- **type\_filter** – Filter string
- **tag\_query** – Tags to query
- **page\_size** – Page size
- **options** – Dictionary of backend-specific options

**Returns** An instance of *BaseStorageRecordSearch*

**update\_record\_tags** (*record: aries\_cloudagent.storage.record.StorageRecord, tags: Mapping[KT, VT\_co]*)

Update an existing stored record's tags.

**Parameters**

- **record** – *StorageRecord* to update
- **tags** – New tags

**Raises** `StorageNotFoundError` – If record not found

**update\_record\_value** (*record: aries\_cloudagent.storage.record.StorageRecord, value: str*)  
Update an existing stored record's value.

**Parameters**

- **record** – *StorageRecord* to update
- **value** – The new value

**Raises** *StorageNotFoundError* – If record not found

```
class aries_cloudagent.storage.basic.BasicStorageRecordSearch (store:
                                                    aries_cloudagent.storage.basic.BasicStorageRecordSearch,
                                                    type_filter: str,
                                                    tag_query: Mapping[KT, VT_co],
                                                    page_size: int =
                                                    None, options:
                                                    Mapping[KT,
                                                    VT_co] = None)
```

Bases: *aries\_cloudagent.storage.base.BaseStorageRecordSearch*

Represent an active stored records search.

**close** ()  
Dispose of the search query.

**fetch** (*max\_count: int*) → *Sequence[aries\_cloudagent.storage.record.StorageRecord]*  
Fetch the next list of results from the store.

**Parameters** **max\_count** – Max number of records to return

**Returns** A list of *StorageRecord*

**Raises** *StorageSearchError* – If the search query has not been opened

**open** ()  
Start the search query.

**opened**  
Accessor for open state.

**Returns** True if opened, else False

```
aries_cloudagent.storage.basic.basic_tag_query_match (tags: dict, tag_query: dict) →
                                                         bool
```

Match simple tag filters (string values).

```
aries_cloudagent.storage.basic.basic_tag_value_match (value: str, match: dict) → bool  
Match a single tag against a tag subquery.
```

TODO: What type coercion is needed? (support int or float values?)

## aries\_cloudagent.storage.error module

Storage-related exceptions.

```
exception aries_cloudagent.storage.error.StorageDuplicateError (*args,          er-
                                                                ror_code:
                                                                str = None,
                                                                **kwargs)
```

Bases: *aries\_cloudagent.storage.error.StorageError*

Duplicate record found in storage.

**exception** `aries_cloudagent.storage.error.StorageError(*args, error_code: str = None, **kwargs)`

Bases: `aries_cloudagent.core.error.BaseError`

Base class for Storage errors.

**exception** `aries_cloudagent.storage.error.StorageNotFoundError(*args, error_code: str = None, **kwargs)`

Bases: `aries_cloudagent.storage.error.StorageError`

Record not found in storage.

**exception** `aries_cloudagent.storage.error.StorageSearchError(*args, error_code: str = None, **kwargs)`

Bases: `aries_cloudagent.storage.error.StorageError`

General exception during record search.

## **aries\_cloudagent.storage.indy module**

Indy implementation of BaseStorage interface.

**class** `aries_cloudagent.storage.indy.IndyStorage(wallet: aries_cloudagent.wallet.indy.IndyWallet)`

Bases: `aries_cloudagent.storage.base.BaseStorage`

Indy Non-Secrets interface.

**add\_record** (*record: aries\_cloudagent.storage.record.StorageRecord*)  
Add a new record to the store.

**Parameters** **record** – *StorageRecord* to be stored

**delete\_record** (*record: aries\_cloudagent.storage.record.StorageRecord*)  
Delete a record.

**Parameters** **record** – *StorageRecord* to delete

**Raises**

- `StorageNotFoundError` – If record not found
- `StorageError` – If a libindy error occurs

**delete\_record\_tags** (*record: aries\_cloudagent.storage.record.StorageRecord, tags: (typing.Sequence, typing.Mapping)*)  
Update an existing stored record's tags.

**Parameters**

- **record** – *StorageRecord* to delete
- **tags** – Tags

**get\_record** (*record\_type: str, record\_id: str, options: Mapping[KT, VT\_co] = None*) → *aries\_cloudagent.storage.record.StorageRecord*  
Fetch a record from the store by type and ID.

**Parameters**

- **record\_type** – The record type
- **record\_id** – The record id

- **options** – A dictionary of backend-specific options

**Returns** A *StorageRecord* instance

**Raises**

- *StorageError* – If the record is not provided
- *StorageError* – If the record ID not provided
- *StorageNotFoundError* – If the record is not found
- *StorageError* – If record not found

**search\_records** (*type\_filter*: *str*, *tag\_query*: *Mapping[KT, VT\_co]* = *None*,  
*page\_size*: *int* = *None*, *options*: *Mapping[KT, VT\_co]* = *None*) →  
 aries\_cloudagent.storage.indy.IndyStorageRecordSearch  
 Search stored records.

**Parameters**

- **type\_filter** – Filter string
- **tag\_query** – Tags to query
- **page\_size** – Page size
- **options** – Dictionary of backend-specific options

**Returns** An instance of *IndyStorageRecordSearch*

**update\_record\_tags** (*record*: *aries\_cloudagent.storage.record.StorageRecord*, *tags*: *Mapping[KT, VT\_co]*)  
 Update an existing stored record's tags.

**Parameters**

- **record** – *StorageRecord* to update
- **tags** – New tags

**Raises**

- *StorageNotFoundError* – If record not found
- *StorageError* – If a libindy error occurs

**update\_record\_value** (*record*: *aries\_cloudagent.storage.record.StorageRecord*, *value*: *str*)  
 Update an existing stored record's value.

**Parameters**

- **record** – *StorageRecord* to update
- **value** – The new value

**Raises**

- *StorageNotFoundError* – If record not found
- *StorageError* – If a libindy error occurs

**wallet**

Accessor for *IndyWallet* instance.

```
class aries_cloudagent.storage.indy.IndyStorageRecordSearch (store:
    aries_cloudagent.storage.indy.IndyStorage,
    type_filter: str,
    tag_query: Mapping[KT, VT_co],
    page_size: int =
    None, options: Mapping[KT, VT_co] =
    None)

Bases: aries_cloudagent.storage.base.BaseStorageRecordSearch

Represent an active stored records search.

close ()
    Dispose of the search query.

fetch (max_count: int) → Sequence[aries_cloudagent.storage.record.StorageRecord]
    Fetch the next list of results from the store.

    Parameters max_count – Max number of records to return

    Returns A list of StorageRecord

    Raises StorageSearchError – If the search query has not been opened

handle
    Accessor for search handle.

    Returns The handle

open ()
    Start the search query.

opened
    Accessor for open state.

    Returns True if opened, else False
```

## aries\_cloudagent.storage.provider module

Default storage provider classes.

```
class aries_cloudagent.storage.provider.StorageProvider
    Bases: aries_cloudagent.config.base.BaseProvider

    Provider for the default configurable storage classes.

    STORAGE_TYPES = {'basic': 'aries_cloudagent.storage.basic.BasicStorage', 'indy': 'ar

    provide (settings: aries_cloudagent.config.base.BaseSettings, injector:
        aries_cloudagent.config.base.BaseInjector)
        Create and return the storage instance.
```

## aries\_cloudagent.storage.record module

Record instance stored and searchable by BaseStorage implementation.

```
class aries_cloudagent.storage.record.StorageRecord
    Bases: aries_cloudagent.storage.record.StorageRecord

    Storage record class.
```

### 1.1.11 aries\_cloudagent.transport package

#### Subpackages

[aries\\_cloudagent.transport.inbound package](#)

#### Submodules

[aries\\_cloudagent.transport.inbound.base module](#)

Base inbound transport class.

```
class aries_cloudagent.transport.inbound.base.BaseInboundTransport (scheme:
                                                                    str, create_session:
                                                                    Callable,
                                                                    *,
                                                                    max_message_size:
                                                                    int = 0,
                                                                    wire_format:
                                                                    aries_cloudagent.transport.wire_format.BaseWireFormat = None)
```

Bases: `abc.ABC`

Base inbound transport class.

**create\_session** (\*, *accept\_undelivered: bool = False, can\_respond: bool = False, client\_info: dict = None, wire\_format: aries\_cloudagent.transport.wire\_format.BaseWireFormat = None*) → Awaitable[aries\_cloudagent.transport.inbound.session.InboundSession]  
Create a new inbound session.

#### Parameters

- **accept\_undelivered** – Flag for accepting undelivered messages
- **can\_respond** – Flag indicating that the transport can send responses
- **client\_info** – Request-specific client information
- **wire\_format** – Optionally override the session wire format

#### **max\_message\_size**

Accessor for this transport's max message size.

#### **scheme**

Accessor for this transport's scheme.

**start** () → None

Start listening for on this transport.

**stop** () → None

Stop listening for on this transport.

```
class aries_cloudagent.transport.inbound.base.InboundTransportConfiguration (module,
                                                                    host,
                                                                    port)
```

Bases: `tuple`

#### **host**

Alias for field number 1

**module**

Alias for field number 0

**port**

Alias for field number 2

```
exception aries_cloudagent.transport.inbound.base.InboundTransportError (*args,
                                                                    er-
                                                                    ror_code:
                                                                    str
                                                                    =
                                                                    None,
                                                                    **kwargs)
```

Bases: *aries\_cloudagent.transport.error.TransportError*

Generic inbound transport error.

```
exception aries_cloudagent.transport.inbound.base.InboundTransportRegistrationError (*args,
                                                                    er-
                                                                    ror_code:
                                                                    str
                                                                    =
                                                                    None,
                                                                    **kwargs)
```

Bases: *aries\_cloudagent.transport.inbound.base.InboundTransportError*

Error in loading an inbound transport.

```
exception aries_cloudagent.transport.inbound.base.InboundTransportSetupError (*args,
                                                                    er-
                                                                    ror_code:
                                                                    str
                                                                    =
                                                                    None,
                                                                    **kwargs)
```

Bases: *aries\_cloudagent.transport.inbound.base.InboundTransportError*

Setup error for an inbound transport.

## aries\_cloudagent.transport.inbound.delivery\_queue module

The Delivery Queue.

The delivery queue holds and manages messages that have not yet been delivered to their intended destination.

```
class aries_cloudagent.transport.inbound.delivery_queue.DeliveryQueue
    Bases: object
```

DeliveryQueue class.

Manages undelivered messages.

```
add_message (msg: aries_cloudagent.transport.outbound.message.OutboundMessage)
    Add an OutboundMessage to delivery queue.
```

The message is added once per recipient key

**Parameters** *msg* – The OutboundMessage to add

```
expire_messages (ttl=None)
    Expire messages that are past the time limit.
```



**Parameters** `ttl` – Optional. Allows override of configured ttl

**get\_one\_message\_for\_key** (*key: str*)  
Remove and return a matching message.

**Parameters** `key` – The key to use for lookup

**has\_message\_for\_key** (*key: str*)  
Check for queued messages by key.

**Parameters** `key` – The key to use for lookup

**inspect\_all\_messages\_for\_key** (*key: str*)  
Return all messages for key.

**Parameters** `key` – The key to use for lookup

**message\_count\_for\_key** (*key: str*)  
Count of queued messages by key.

**Parameters** `key` – The key to use for lookup

**remove\_message\_for\_key** (*key: str, msg: aries\_cloudagent.transport.outbound.message.OutboundMessage*)  
Remove specified message from queue for key.

**Parameters**

- **key** – The key to use for lookup
- **msg** – The message to remove from the queue

**class** `aries_cloudagent.transport.inbound.delivery_queue.QueuedMessage` (*msg: aries\_cloudagent.transport.outbound.message.OutboundMessage*)

Bases: `object`

Wrapper Class for queued messages.

Allows tracking Metadata.

**older\_than** (*compare\_timestamp: float*) → bool  
Age Comparison.

Allows you to test age as compared to the provided timestamp.

**Parameters** `compare_timestamp` – The timestamp to compare

## `aries_cloudagent.transport.inbound.http` module

Http Transport classes and functions.

**class** `aries_cloudagent.transport.inbound.http.HttpTransport` (*host: str, port: int, create\_session, \*\*kwargs*)

Bases: `aries_cloudagent.transport.inbound.base.BaseInboundTransport`

Http Transport class.

**inbound\_message\_handler** (*request: <sphinx.ext.autodoc.importer.\_MockObject object at 0x7f79508ea0b8>*)  
Message handler for inbound messages.

**Parameters** `request` – aiohttp request object

**Returns** The web response

**invite\_message\_handler** (*request:* <sphinx.ext.autodoc.importer.\_MockObject object at 0x7f79508ea0b8>)

Message handler for invites.

**Parameters** **request** – aiohttp request object

**Returns** The web response

**make\_application** () → <sphinx.ext.autodoc.importer.\_MockObject object at 0x7f79508ea0b8>

Construct the aiohttp application.

**start** () → None

Start this transport.

**Raises** InboundTransportSetupError – If there was an error starting the webserver

**stop** () → None

Stop this transport.

## aries\_cloudagent.transport.inbound.manager module

Inbound transport manager.

**class** aries\_cloudagent.transport.inbound.manager.**InboundTransportManager** (*context:* aries\_cloudagent.config.inbound\_transport\_manager: ContextManager, *re-ceive\_inbound:* Callable[[T\_co, T\_contra, V\_co], Awaitable[Response]], *re-turn\_inbound:* Callable[[Response], Awaitable[Response]] = None)

Bases: `object`

Inbound transport manager class.

**closed\_session** (*session:* aries\_cloudagent.transport.inbound.session.InboundSession)

Clean up a closed session.

Returns an undelivered message to the caller if possible.

**create\_session** (*transport\_type:* str, \*, *accept\_undelivered:* bool = False, *can\_respond:* bool = False, *client\_info:* dict = None, *wire\_format:* aries\_cloudagent.transport.wire\_format.BaseWireFormat = None)

Create a new inbound session.

**Parameters**

- **transport\_type** – The inbound transport identifier
- **accept\_undelivered** – Flag for accepting undelivered messages
- **can\_respond** – Flag indicating that the transport can send responses
- **client\_info** – An optional dict describing the client
- **wire\_format** – Override the wire format for this session

**dispatch\_complete** (*message: aries\_cloudagent.transport.inbound.message.InboundMessage*,  
*completed: aries\_cloudagent.utils.task\_queue.CompletedTask*)  
 Handle completion of message dispatch.

**get\_transport\_instance** (*transport\_id: str*) → *aries\_cloudagent.transport.inbound.base.BaseInboundTransport*  
 Get an instance of a running transport by ID.

**process\_undelivered** (*session: aries\_cloudagent.transport.inbound.session.InboundSession*)  
 Interact with undelivered queue to find applicable messages.

**Parameters session** – The inbound session

**register** (*config: aries\_cloudagent.transport.inbound.base.InboundTransportConfiguration*) → *str*  
 Register transport module.

**Parameters config** – The inbound transport configuration

**register\_transport** (*transport: aries\_cloudagent.transport.inbound.base.BaseInboundTransport*,  
*transport\_id: str*) → *str*  
 Register a new inbound transport class.

**Parameters**

- **transport** – Transport instance to register
- **transport\_id** – The transport ID to register

**return\_to\_session** (*outbound: aries\_cloudagent.transport.outbound.message.OutboundMessage*)  
 → *bool*  
 Return an outbound message via an open session, if possible.

**return\_undelivered** (*outbound: aries\_cloudagent.transport.outbound.message.OutboundMessage*)  
 → *bool*  
 Add an undelivered message to the undelivered queue.

At this point the message could not be associated with an inbound session and could not be delivered via an outbound transport.

**setup** ()  
 Perform setup operations.

**start** ()  
 Start all registered transports.

**start\_transport** (*transport\_id: str*)  
 Start a registered inbound transport.

**Parameters transport\_id** – ID for the inbound transport to start

**stop** (*wait: bool = True*)  
 Stop all registered transports.

## aries\_cloudagent.transport.inbound.message module

Classes representing inbound messages.

```
class aries_cloudagent.transport.inbound.message.InboundMessage (payload:
    Union[str,
    bytes], receipt:
    aries_cloudagent.transport.inbound.rec
    *,      connec-
    tion_id:  str
    = None, ses-
    sion_id: str =
    None, trans-
    port_type: str
    = None)
```

Bases: `object`

Container class linking a message payload with its receipt details.

## aries\_cloudagent.transport.inbound.receipt module

Classes for representing message receipt details.

```
class aries_cloudagent.transport.inbound.receipt.MessageReceipt (*,      connec-
    tion_id:  str
    = None, di-
    rect_response_mode:
    str = None,
    in_time: date-
    time.datetime
    =      None,
    raw_message:
    str = None,
    recipi-
    ent_verkey: str
    = None, recipi-
    ent_id: str
    = None, recipi-
    ent_id_public:
    bool = None,
    sender_id:
    str = None,
    sender_verkey:
    str = None,
    thread_id: str
    = None)
```

Bases: `object`

Properties of an agent message's delivery.

**REPLY\_MODE\_ALL** = 'all'

**REPLY\_MODE\_NONE** = 'none'

**REPLY\_MODE\_THREAD** = 'thread'

**connection\_id**

Accessor for the pairwise connection identifier.

**Returns** This context's connection identifier

**direct\_response\_mode**

Accessor for the requested direct response mode.

**Returns** This context's requested direct response mode

**direct\_response\_requested**

Accessor for the the state of the direct response mode.

**Returns** This context's requested direct response mode

**in\_time**

Accessor for the datetime the message was received.

**Returns** This context's received time

**raw\_message**

Accessor for the raw message text.

**Returns** The raw message text

**recipient\_did**

Accessor for the recipient DID which corresponds with the verkey.

**Returns** The recipient DID

**recipient\_did\_public**

Check if the recipient did is public.

Indicates whether the message is associated with a public (ledger) recipient DID.

**Returns** True if the recipient's DID is public, else false

**recipient\_verkey**

Accessor for the recipient verkey key used to pack the incoming request.

**Returns** The recipient verkey

**sender\_did**

Accessor for the sender DID which corresponds with the verkey.

**Returns** The sender did

**sender\_verkey**

Accessor for the sender public key used to pack the incoming request.

**Returns** This context's sender's verkey

**thread\_id**

Accessor for the identifier of the message thread.

**Returns** The delivery thread ID

**aries\_cloudagent.transport.inbound.session module**

Inbound connection handling classes.

```
class aries_cloudagent.transport.inbound.session.AcceptResult (accepted: bool,  
retry: bool =  
False)
```

Bases: `object`

Represent the result of `accept_response`.

```
class aries_cloudagent.transport.inbound.session.InboundSession (*,      context:
                                                                    aries_cloudagent.config.injection_cont
                                                                    in-
                                                                    bound_handler:
                                                                    Callable, ses-
                                                                    sion_id:  str,
                                                                    wire_format:
                                                                    aries_cloudagent.transport.wire_forma
                                                                    ac-
                                                                    cept_undelivered:
                                                                    bool = False,
                                                                    can_respond:
                                                                    bool = False,
                                                                    client_info:
                                                                    dict = None,
                                                                    close_handler:
                                                                    Callable =
                                                                    None,      re-
                                                                    ply_mode: str
                                                                    = None, re-
                                                                    ply_thread_ids:
                                                                    Sequence[str]
                                                                    = None, re-
                                                                    ply_verkeys:
                                                                    Sequence[str]
                                                                    = None, trans-
                                                                    port_type: str
                                                                    = None)
```

Bases: `object`

Track an open transport connection for direct routing of outbound messages.

**accept\_response** (*message: aries\_cloudagent.transport.outbound.message.OutboundMessage*) →  
aries\_cloudagent.transport.inbound.session.AcceptResult  
Try to queue an outbound message if it applies to this session.

Returns: a tuple of (message buffered, retry later)

**add\_reply\_thread\_ids** (\**thids*)  
Add a thread ID to the set of potential reply targets.

**add\_reply\_verkeys** (\**verkeys*)  
Add a verkey to the set of potential reply targets.

**can\_respond**  
Accessor for the session can-respond state.

**clear\_response** ()  
Handle when the buffered response message has been delivered.

**close** ()  
Setter for the session closed state.

**closed**  
Accessor for the session closed state.

**encode\_outbound** (*outbound: aries\_cloudagent.transport.outbound.message.OutboundMessage*) →  
aries\_cloudagent.transport.outbound.message.OutboundMessage  
Apply wire formatting to an outbound message.

**parse\_inbound** (*payload\_enc: Union[str, bytes]*) → *aries\_cloudagent.transport.inbound.message.InboundMessage*  
 Convert a message payload and to an inbound message.

**process\_inbound** (*message: aries\_cloudagent.transport.inbound.message.InboundMessage*)  
 Process an incoming message and update the session metadata as necessary.

**Parameters** *message* – The inbound message instance

**receive** (*payload\_enc: Union[str, bytes]*) → *aries\_cloudagent.transport.inbound.message.InboundMessage*  
 Receive a new message payload and dispatch the message.

**receive\_inbound** (*message: aries\_cloudagent.transport.inbound.message.InboundMessage*)  
 Deliver the inbound message to the conductor.

**reply\_mode**  
 Accessor for the session reply mode.

**reply\_thread\_ids**  
 Accessor for the reply thread IDs.

**reply\_verkeys**  
 Accessor for the reply verkeys.

**response\_buffered**  
 Check if a response is currently buffered.

**select\_outbound** (*message: aries\_cloudagent.transport.outbound.message.OutboundMessage*) → *bool*  
 Determine if an outbound message should be sent to this session.

**Parameters** *message* – The outbound message to be checked

**set\_response** (*message: aries\_cloudagent.transport.outbound.message.OutboundMessage*)  
 Set the contents of the response message buffer.

**wait\_response** () → *Union[str, bytes]*  
 Wait for a response to be buffered and pack it.

## aries\_cloudagent.transport.inbound.ws module

Websockets Transport classes and functions.

**class** *aries\_cloudagent.transport.inbound.ws.WsTransport* (*host: str, port: int, create\_session, \*\*kwargs*)

Bases: *aries\_cloudagent.transport.inbound.base.BaseInboundTransport*

Websockets Transport class.

**inbound\_message\_handler** (*request*)  
 Message handler for inbound messages.

**Parameters** *request* – aiohttp request object

**Returns** The web response

**make\_application** () → *<sphinx.ext.autodoc.importer.\_MockObject object at 0x7f7950517128>*  
 Construct the aiohttp application.

**scheme**  
 Accessor for this transport's scheme.

**start** () → *None*  
 Start this transport.

**Raises** `InboundTransportSetupError` – If there was an error starting the webserver

**stop()** → None

Stop this transport.

## `aries_cloudagent.transport.outbound` package

### Submodules

#### `aries_cloudagent.transport.outbound.base` module

Base outbound transport.

```
class aries_cloudagent.transport.outbound.base.BaseOutboundTransport (wire_format:
                                                                    aries_cloudagent.transport.wire
                                                                    =
                                                                    None)
```

Bases: `abc.ABC`

Base outbound transport class.

**collector**

Accessor for the stats collector instance.

**handle\_message** (context: `aries_cloudagent.config.injection_context.InjectionContext`, payload: `Union[str, bytes]`, endpoint: `str`)

Handle message from queue.

#### Parameters

- **context** – the context that produced the message
- **payload** – message payload in string or byte format
- **endpoint** – URI endpoint for delivery

**start()**

Start the transport.

**stop()**

Shut down the transport.

**wire\_format**

Accessor for a custom wire format for the transport.

```
exception aries_cloudagent.transport.outbound.base.OutboundDeliveryError (*args,
                                                                    er-
                                                                    ror_code:
                                                                    str
                                                                    =
                                                                    None,
                                                                    **kwargs)
```

Bases: `aries_cloudagent.transport.outbound.base.OutboundTransportError`

Base exception when a message cannot be delivered via an outbound transport.



```
exception aries_cloudagent.transport.outbound.base.OutboundTransportError(*args,
                                                                           error_code:
                                                                           str
                                                                           =
                                                                           None,
                                                                           **kwargs)
```

Bases: `aries_cloudagent.transport.error.TransportError`

Generic outbound transport error.

```
exception aries_cloudagent.transport.outbound.base.OutboundTransportRegistrationError(*args,
                                                                                       error_code:
                                                                                       str
                                                                                       =
                                                                                       None,
                                                                                       **kwargs)
```

Bases: `aries_cloudagent.transport.outbound.base.OutboundTransportError`

Outbound transport registration error.

## aries\_cloudagent.transport.outbound.http module

Http outbound transport.

```
class aries_cloudagent.transport.outbound.http.HttpTransport
    Bases: aries_cloudagent.transport.outbound.base.BaseOutboundTransport
```

Http outbound transport class.

```
handle_message (context: aries_cloudagent.config.injection_context.InjectionContext, payload:
                  Union[str, bytes], endpoint: str)
    Handle message from queue.
```

### Parameters

- **context** – the context that produced the message
- **payload** – message payload in string or byte format
- **endpoint** – URI endpoint for delivery

```
schemes = ('http', 'https')
```

```
start ()
    Start the transport.
```

```
stop ()
    Stop the transport.
```

## aries\_cloudagent.transport.outbound.manager module

Outbound transport manager.

```
class aries_cloudagent.transport.outbound.manager.OutboundTransportManager (context:
                                                                    aries_cloudagent.config.
                                                                    handler.
                                                                    Callable
                                                                    =
                                                                    None)
```

Bases: `object`

Outbound transport manager class.

**deliver\_queued\_message** (queued: *aries\_cloudagent.transport.outbound.manager.QueuedOutboundMessage*)  
→ `_asyncio.Task`  
Kick off delivery of a queued message.

**encode\_queued\_message** (queued: *aries\_cloudagent.transport.outbound.manager.QueuedOutboundMessage*)  
→ `_asyncio.Task`  
Kick off encoding of a queued message.

**enqueue\_message** (context: *aries\_cloudagent.config.injection\_context.InjectionContext*, outbound: *aries\_cloudagent.transport.outbound.message.OutboundMessage*)  
Add an outbound message to the queue.

#### Parameters

- **context** – The context of the request
- **outbound** – The outbound message to deliver

**enqueue\_webhook** (topic: *str*, payload: *dict*, endpoint: *str*, max\_attempts: *int* = *None*)  
Add a webhook to the queue.

#### Parameters

- **topic** – The webhook topic
- **payload** – The webhook payload
- **endpoint** – The webhook endpoint
- **max\_attempts** – Override the maximum number of attempts

**Raises** `OutboundDeliveryError` – if the associated transport is not running

**finished\_deliver** (queued: *aries\_cloudagent.transport.outbound.manager.QueuedOutboundMessage*,  
completed: *aries\_cloudagent.utils.task\_queue.CompletedTask*)  
Handle completion of queued message delivery.

**finished\_encode** (queued: *aries\_cloudagent.transport.outbound.manager.QueuedOutboundMessage*,  
completed: *aries\_cloudagent.utils.task\_queue.CompletedTask*)  
Handle completion of queued message encoding.

**flush** ()  
Wait for any queued messages to be delivered.

**get\_registered\_transport\_for\_scheme** (scheme: *str*) → *str*  
Find the registered transport ID for a given scheme.

**get\_running\_transport\_for\_endpoint** (endpoint: *str*)  
Find the running transport ID to use for a given endpoint.

**get\_running\_transport\_for\_scheme** (scheme: *str*) → *str*  
Find the running transport ID for a given scheme.

**get\_transport\_instance** (transport\_id: *str*) → *aries\_cloudagent.transport.outbound.base.BaseOutboundTransport*  
Get an instance of a running transport by ID.

**perform\_encode** (*queued: aries\_cloudagent.transport.outbound.manager.QueuedOutboundMessage*)  
Perform message encoding.

**process\_queued** () → `_asyncio.Task`  
Start the process to deliver queued messages if necessary.  
  
Returns: the current queue processing task or None

**register** (*module: str*) → `str`  
Register a new outbound transport by module path.

**Parameters** **module** – Module name to register

**Raises**

- `OutboundTransportRegistrationError` – If the imported class cannot be located
- `OutboundTransportRegistrationError` – If the imported class does not specify a `schemes` attribute
- `OutboundTransportRegistrationError` – If the scheme has already been registered

**register\_class** (*transport\_class: Type[aries\_cloudagent.transport.outbound.base.BaseOutboundTransport]*,  
*transport\_id: str = None*) → `str`  
Register a new outbound transport class.

**Parameters** **transport\_class** – Transport class to register

**Raises**

- `OutboundTransportRegistrationError` – If the imported class does not specify a `schemes` attribute
- `OutboundTransportRegistrationError` – If the scheme has already been registered

**setup** ()  
Perform setup operations.

**start** ()  
Start all transports and feed messages from the queue.

**start\_transport** (*transport\_id: str*)  
Start a registered transport.

**stop** (*wait: bool = True*)  
Stop all running transports.

**class** `aries_cloudagent.transport.outbound.manager.QueuedOutboundMessage` (*context: aries\_cloudagent.config.injector.Injector, message: aries\_cloudagent.transport.outbound.base.OutboundMessage, target: str, get: Callable[[str], aries\_cloudagent.connection.Connection], transport\_id: str*)

Bases: `object`

Class representing an outbound message pending delivery.

```

STATE_DELIVER = 'deliver'
STATE_DONE = 'done'
STATE_ENCODE = 'encode'
STATE_NEW = 'new'
STATE_PENDING = 'pending'
STATE_RETRY = 'retry'

```

## aries\_cloudagent.transport.outbound.message module

Outbound message representation.

```

class aries_cloudagent.transport.outbound.message.OutboundMessage(*, connection_id:
    str = None,
    enc_payload:
    Union[str,
    bytes] =
    None, endpoint: str
    = None,
    payload:
    Union[str,
    bytes], reply_session_id:
    str =
    None, reply_thread_id:
    str =
    None, reply_to_verkey:
    str =
    None, reply_from_verkey:
    str = None,
    target:
    aries_cloudagent.connections.model.Connection =
    None,
    target_get_list: Sequence[aries_cloudagent.connections.model.Connection] =
    None,
    to_session_only:
    bool =
    False)

```

Bases: `object`

Represents an outgoing message.

## aries\_cloudagent.transport.outbound.ws module

Websockets outbound transport.

```
class aries_cloudagent.transport.outbound.ws.WsTransport
    Bases: aries_cloudagent.transport.outbound.base.BaseOutboundTransport

    Websockets outbound transport class.

    handle_message (context: aries_cloudagent.config.injection_context.InjectionContext, payload: Union[str, bytes], endpoint: str)
        Handle message from queue.

        Parameters

- context – the context that produced the message
- payload – message payload in string or byte format
- endpoint – URI endpoint for delivery

schemes = ('ws', 'wss')

    start ()
        Start the outbound transport.

    stop ()
        Stop the outbound transport.
```

## aries\_cloudagent.transport.queue package

### Submodules

#### aries\_cloudagent.transport.queue.base module

Abstract message queue.

```
class aries_cloudagent.transport.queue.base.BaseMessageQueue
    Bases: abc.ABC

    Abstract message queue class.

    dequeue (*, timeout: int = None)
        Dequeue a message.

        Returns The dequeued message, or None if a timeout occurs

        Raises

- asyncio.CancelledError if the queue has been stopped
- asyncio.TimeoutError if the timeout is reached

enqueue (message)
        Enqueue a message.

        Parameters message – The message to add to the end of the queue

        Raises asyncio.CancelledError if the queue has been stopped

    join ()
        Wait for the queue to empty.

    reset ()
        Empty the queue and reset the stop event.
```

**stop()**  
Cancel active iteration of the queue.

**task\_done()**  
Indicate that the current task is complete.

## aries\_cloudagent.transport.queue.basic module

Basic in memory queue.

**class** aries\_cloudagent.transport.queue.basic.**BasicMessageQueue**  
Bases: *aries\_cloudagent.transport.queue.base.BaseMessageQueue*

Basic in memory queue implementation class.

**dequeue** (\*, *timeout: int = None*)  
Dequeue a message.

**Returns** The dequeued message, or None if a timeout occurs

**Raises**

- `asyncio.CancelledError` if the queue has been stopped
- `asyncio.TimeoutError` if the timeout is reached

**enqueue** (*message*)  
Enqueue a message.

**Parameters** **message** – The message to add to the end of the queue

**Raises** `asyncio.CancelledError` if the queue has been stopped

**join**()  
Wait for the queue to empty.

**make\_queue**()  
Create the queue instance.

**reset**()  
Empty the queue and reset the stop event.

**stop**()  
Cancel active iteration of the queue.

**task\_done**()  
Indicate that the current task is complete.

## Submodules

### aries\_cloudagent.transport.error module

Transport-related error classes and codes.

**exception** aries\_cloudagent.transport.error.**MessageEncodeError** (\*args, *error\_code: str = None*, \*\*kwargs)

Bases: *aries\_cloudagent.transport.error.WireFormatError*

Message encoding error.

**error\_code** = 'message\_encode\_error'

```
exception aries_cloudagent.transport.error.MessageParseError(*args, error_code:
                                                                str = None,
                                                                **kwargs)
```

Bases: *aries\_cloudagent.transport.error.WireFormatError*

Message parse error.

```
error_code = 'message_parse_error'
```

```
exception aries_cloudagent.transport.error.TransportError(*args, error_code: str
                                                            = None, **kwargs)
```

Bases: *aries\_cloudagent.core.error.BaseError*

Base class for all transport errors.

```
exception aries_cloudagent.transport.error.WireFormatError(*args, error_code: str
                                                            = None, **kwargs)
```

Bases: *aries\_cloudagent.transport.error.TransportError*

Base class for wire-format errors.

## aries\_cloudagent.transport.pack\_format module

Standard packed message format classes.

```
class aries_cloudagent.transport.pack_format.PackWireFormat
    Bases: aries_cloudagent.transport.wire_format.BaseWireFormat
```

Standard DIDComm message parser and serializer.

```
encode_message (context:      aries_cloudagent.config.injection_context.InjectionContext,  mes-
                    sage_json: Union[str, bytes], recipient_keys: Sequence[str], routing_keys:
                    Sequence[str], sender_key: str) → Union[str, bytes]
```

Encode an outgoing message for transport.

### Parameters

- **context** – The injection context for settings and services
- **message\_json** – The message body to serialize
- **recipient\_keys** – A sequence of recipient verkeys
- **routing\_keys** – A sequence of routing verkeys
- **sender\_key** – The verification key of the sending agent

**Returns** The encoded message

**Raises** `MessageEncodeError` – If the message could not be encoded

```
pack (context: aries_cloudagent.config.injection_context.InjectionContext, message_json: Union[str,
    bytes], recipient_keys: Sequence[str], routing_keys: Sequence[str], sender_key: str)
```

Look up the wallet instance and perform the message pack.

```
parse_message (context:      aries_cloudagent.config.injection_context.InjectionContext,
                    message_body: Union[str, bytes]) → Tuple[dict,
                    aries_cloudagent.transport.inbound.receipt.MessageReceipt]
```

Deserialize an incoming message and further populate the request context.

### Parameters

- **context** – The injection context for settings and services
- **message\_body** – The body of the message

**Returns** A tuple of the parsed message and a message receipt instance

**Raises**

- `MessageParseError` – If the JSON parsing failed
- `MessageParseError` – If a wallet is required but can't be located

**unpack** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, message\_body: Union[str, bytes], receipt: aries\_cloudagent.transport.inbound.receipt.MessageReceipt*)  
Look up the wallet instance and perform the message unpack.

## aries\_cloudagent.transport.stats module

aiohttp stats collector support.

**class** `aries_cloudagent.transport.stats.StatsTracer` (*collector: aries\_cloudagent.utils.stats.Collector, prefix: str*)

Bases: `sphinx.ext.autodoc.importer._MockObject`

Attach hooks to client session events and report statistics.

**connection\_queued\_end** (*session, context, params*)  
Handle the end of a queued connection.

**connection\_queued\_start** (*session, context, params*)  
Handle the start of a queued connection.

**connection\_ready** (*session, context, params*)  
Handle the end of connection acquisition.

**dns\_resolvehost\_end** (*session, context, params*)  
Handle the end of a DNS resolution.

**dns\_resolvehost\_start** (*session, context, params*)  
Handle the start of a DNS resolution.

**request\_end** (*session, context, params*)  
Handle the end of request.

**request\_start** (*session, context, params*)  
Handle the start of a request.

**socket\_connect\_start** (*session, context, params*)  
Handle the start of a socket connection.

## aries\_cloudagent.transport.wire\_format module

Abstract wire format classes.

**class** `aries_cloudagent.transport.wire_format.BaseWireFormat`  
Bases: `object`

Abstract messaging wire format.

**encode\_message** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, message\_json: Union[str, bytes], recipient\_keys: Sequence[str], routing\_keys: Sequence[str], sender\_key: str*) → `Union[str, bytes]`  
Encode an outgoing message for transport.

**Parameters**



- **context** – The injection context for settings and services
- **message\_json** – The message body to serialize
- **recipient\_keys** – A sequence of recipient verkeys
- **routing\_keys** – A sequence of routing verkeys
- **sender\_key** – The verification key of the sending agent

**Returns** The encoded message

**Raises** `MessageEncodeError` – If the message could not be encoded

**parse\_message** (*context*: `aries_cloudagent.config.injection_context.InjectionContext`,  
*message\_body*: `Union[str, bytes]`) → `Tuple[dict,`  
`aries_cloudagent.transport.inbound.receipt.MessageReceipt]`  
 Deserialize an incoming message and further populate the request context.

#### Parameters

- **context** – The injection context for settings and services
- **message\_body** – The body of the message

**Returns** A tuple of the parsed message and a message receipt instance

**Raises** `MessageParseError` – If the message can't be parsed

**class** `aries_cloudagent.transport.wire_format.JsonWireFormat`  
 Bases: `aries_cloudagent.transport.wire_format.BaseWireFormat`

Unencrypted wire format.

**encode\_message** (*context*: `aries_cloudagent.config.injection_context.InjectionContext`, *mes-*  
*sage\_json*: `Union[str, bytes]`, *recipient\_keys*: `Sequence[str]`, *routing\_keys*:  
`Sequence[str]`, *sender\_key*: `str`) → `Union[str, bytes]`  
 Encode an outgoing message for transport.

#### Parameters

- **context** – The injection context for settings and services
- **message\_json** – The message body to serialize
- **recipient\_keys** – A sequence of recipient verkeys
- **routing\_keys** – A sequence of routing verkeys
- **sender\_key** – The verification key of the sending agent

**Returns** The encoded message

**Raises** `MessageEncodeError` – If the message could not be encoded

**parse\_message** (*context*: `aries_cloudagent.config.injection_context.InjectionContext`,  
*message\_body*: `Union[str, bytes]`) → `Tuple[dict,`  
`aries_cloudagent.transport.inbound.receipt.MessageReceipt]`  
 Deserialize an incoming message and further populate the request context.

#### Parameters

- **context** – The injection context for settings and services
- **message\_body** – The body of the message

**Returns** A tuple of the parsed message and a message receipt instance

**Raises** `MessageParseError` – If the JSON parsing failed

## 1.1.12 aries\_cloudagent.utils package

### Submodules

#### aries\_cloudagent.utils.classloader module

The classloader provides utilities to dynamically load classes and modules.

**class** aries\_cloudagent.utils.classloader.**ClassLoader**

Bases: `object`

Class used to load classes from modules dynamically.

**classmethod** `load_class` (*class\_name: str, default\_module: str = None, package: str = None*)

Resolve a complete class path (ie. `typing.Dict`) to the class itself.

##### Parameters

- **class\_name** – the class name
- **default\_module** – the default module to load, if not part of in the class name
- **package** – the parent package to search for the module

**Returns** The resolved class

##### Raises

- `ClassNotFoundError` – If the class could not be resolved at path
- `ModuleLoadError` – If there was an error loading the module

**classmethod** `load_module` (*mod\_path: str, package: str = None*) → module

Load a module by its absolute path.

##### Parameters

- **mod\_path** – the absolute or relative module path
- **package** – the parent package to search for the module

**Returns** The resolved module or *None* if the module cannot be found

**Raises** `ModuleLoadError` – If there was an error loading the module

**classmethod** `load_subclass_of` (*base\_class: Type[CT\_co], mod\_path: str, package: str = None*)

Resolve an implementation of a base path within a module.

##### Parameters

- **base\_class** – the base class being implemented
- **mod\_path** – the absolute module path
- **package** – the parent package to search for the module

**Returns** The resolved class

##### Raises

- `ClassNotFoundError` – If the module or class implementation could not be found
- `ModuleLoadError` – If there was an error loading the module

**classmethod** `scan_subpackages` (*package: str*) → Sequence[str]

Return a list of sub-packages defined under a named package.

```
exception aries_cloudagent.utils.classloader.ClassNotFoundError(*args, error_code:
                                                                str = None,
                                                                **kwargs)
```

Bases: *aries\_cloudagent.core.error.BaseError*

Class not found error.

```
exception aries_cloudagent.utils.classloader.ModuleLoadError(*args, error_code:
                                                                str = None,
                                                                **kwargs)
```

Bases: *aries\_cloudagent.core.error.BaseError*

Module load error.

## aries\_cloudagent.utils.http module

HTTP utility methods.

```
exception aries_cloudagent.utils.http.FetchError(*args, error_code: str = None,
                                                    **kwargs)
```

Bases: *aries\_cloudagent.core.error.BaseError*

Error raised when an HTTP fetch fails.

```
aries_cloudagent.utils.http.fetch(url: str, *, headers: dict = None, retry: bool = True,
                                   max_attempts: int = 5, interval: float = 1.0, back-
                                   off: float = 0.25, request_timeout: float = 10.0, con-
                                   nector: <sphinx.ext.autodoc.importer._MockObject
                                   object at 0x7f7951716a90> = None, session:
                                   <sphinx.ext.autodoc.importer._MockObject object at
                                   0x7f7951716ba8> = None, json: bool = False)
```

Fetch from an HTTP server with automatic retries and timeouts.

### Parameters

- **url** – the address to fetch
- **headers** – an optional dict of headers to send
- **retry** – flag to retry the fetch
- **max\_attempts** – the maximum number of attempts to make
- **interval** – the interval between retries, in seconds
- **backoff** – the backoff interval, in seconds
- **request\_timeout** – the HTTP request timeout, in seconds
- **connector** – an optional existing BaseConnector
- **session** – a shared ClientSession
- **json** – flag to parse the result as JSON

```
aries_cloudagent.utils.http.fetch_stream(url: str, *, headers: dict = None, retry:
                                         bool = True, max_attempts: int = 5, in-
                                         terval: float = 1.0, backoff: float = 0.25,
                                         request_timeout: float = 10.0, connector:
                                         <sphinx.ext.autodoc.importer._MockObject
                                         object at 0x7f7951716a90> = None, session:
                                         <sphinx.ext.autodoc.importer._MockObject
                                         object at 0x7f7951716ba8> = None)
```

Fetch from an HTTP server with automatic retries and timeouts.

#### Parameters

- **url** – the address to fetch
- **headers** – an optional dict of headers to send
- **retry** – flag to retry the fetch
- **max\_attempts** – the maximum number of attempts to make
- **interval** – the interval between retries, in seconds
- **backoff** – the backoff interval, in seconds
- **request\_timeout** – the HTTP request timeout, in seconds
- **connector** – an optional existing BaseConnector
- **session** – a shared ClientSession
- **json** – flag to parse the result as JSON

### aries\_cloudagent.utils.repeat module

Utils for repeating tasks.

```
class aries_cloudagent.utils.repeat.RepeatAttempt (seq:
                                                    aries_cloudagent.utils.repeat.RepeatSequence,
                                                    index: int = 1)
```

Bases: `object`

Represents the current iteration in a repeat sequence.

#### **final**

Check if this is the last instance in the sequence.

**next** () → `aries_cloudagent.utils.repeat.RepeatAttempt`

Get the next attempt instance.

#### **next\_interval**

Calculate the interval before the next attempt.

**timeout** (interval: float = None)

Create a context manager for timing out an attempt.

```
class aries_cloudagent.utils.repeat.RepeatSequence (limit: int = 0, interval: float = 0.0,
                                                    backoff: float = 0.0)
```

Bases: `object`

Represents a repetition sequence.

**next\_interval** (index: int) → float

Calculate the time before the next attempt.

**start** () → aries\_cloudagent.utils.repeat.RepeatAttempt  
Get the first attempt in the sequence.

## aries\_cloudagent.utils.stats module

Classes for tracking performance and timing.

**class** aries\_cloudagent.utils.stats.**Collector** (\*, enabled: bool = True, log\_path: str = None)

Bases: `object`

Collector for a set of statistics.

**enabled**

Accessor for the collector's enabled property.

**extract** (groups: Sequence[str] = None) → dict

Extract statistics for a specific set of groups.

**log** (name: str, duration: float, start: float = None)

Log an entry in the statistics if the collector is enabled.

**mark** (\*names)

Make a custom decorator function for adding to the set of groups.

**reset** ()

Reset the collector's statistics.

**results**

Accessor for the current set of collected statistics.

**timer** (\*groups)

Create a new timer attached to this collector.

**wrap** (obj, prop\_name: Union[str, Sequence[str]], groups: Sequence[str] = None, \*, ignore\_missing: bool = False)

Wrap a method on a class or class instance.

**wrap\_coro** (fn, groups: Sequence[str])

Wrap a coroutine instance to collect timing statistics on execution.

**wrap\_fn** (fn, groups: Sequence[str])

Wrap a function instance to collect timing statistics on execution.

**class** aries\_cloudagent.utils.stats.**Stats**

Bases: `object`

A collection of statistics.

**extract** (names: Sequence[str] = None) → dict

Summarize the stats in a dictionary.

**log** (name: str, duration: float)

Log an entry in the stats.

**class** aries\_cloudagent.utils.stats.**Timer** (collector: aries\_cloudagent.utils.stats.Collector, groups: Sequence[str])

Bases: `object`

Timer instance for a running task.

**classmethod** **now** ()

Fetch a standard timer value.

**start ()** → aries\_cloudagent.utils.stats.Timer  
Start the timer.

**stop ()**  
Stop the timer.

### aries\_cloudagent.utils.task\_queue module

Classes for managing a set of async tasks.

**class** aries\_cloudagent.utils.task\_queue.**CompletedTask** (*task: \_asyncio.Task, exc\_info: Tuple, ident: str = None, timing: dict = None*)

Bases: `object`

Represent the result of a queued task.

**class** aries\_cloudagent.utils.task\_queue.**PendingTask** (*coro: Coroutine[T\_co, T\_contra, V\_co], complete\_hook: Callable = None, ident: str = None, task\_future: \_asyncio.Future = None, queued\_time: float = None*)

Bases: `object`

Represent a task in the queue.

**cancel ()**  
Cancel the pending task.

**cancelled**  
Accessor for the cancelled property.

**task**  
Accessor for the task.

**class** aries\_cloudagent.utils.task\_queue.**TaskQueue** (*max\_active: int = 0, timed: bool = False, trace\_fn: Callable = None*)

Bases: `object`

A class for managing a set of async tasks.

**add\_active** (*task: \_asyncio.Task, task\_complete: Callable = None, ident: str = None, timing: dict = None*) → `_asyncio.Task`  
Register an active async task with an optional completion callback.

#### Parameters

- **task** – The `asyncio` task instance
- **task\_complete** – An optional callback to run on completion
- **ident** – A string identifier for the task
- **timing** – An optional dictionary of timing information

**add\_pending** (*pending: aries\_cloudagent.utils.task\_queue.PendingTask*)  
Add a task to the pending queue.

**Parameters pending** – The `PendingTask` to add to the task queue

**cancel ()**  
Cancel any pending or active tasks in the queue.

**cancel\_pending()**

Cancel any pending tasks in the queue.

**cancelled**

Accessor for the cancelled property of the queue.

**complete** (*timeout: float = None, cleanup: bool = True*)

Cancel any pending tasks and wait for, or cancel active tasks.

**completed\_task** (*task: \_asyncio.Task, task\_complete: Callable, ident: str, timing: dict = None*)

Clean up after a task has completed and run callbacks.

**current\_active**

Accessor for the current number of active tasks in the queue.

**current\_pending**

Accessor for the current number of pending tasks in the queue.

**current\_size**

Accessor for the total number of tasks in the queue.

**drain()** → *\_asyncio.Task*

Start the process to run queued tasks.

**flush()**

Wait for any active or pending tasks to be completed.

**max\_active**

Accessor for the maximum number of active tasks in the queue.

**put** (*coro: Coroutine[T\_co, T\_contra, V\_co], task\_complete: Callable = None, ident: str = None*) →

*aries\_cloudagent.utils.task\_queue.PendingTask*

Add a new task to the queue, delaying execution if busy.

**Parameters**

- **coro** – The coroutine to run
- **task\_complete** – A callback to run on completion
- **ident** – A string identifier for the task

Returns: a future resolving to the asyncio task instance once queued

**ready**

Accessor for the ready property of the queue.

**run** (*coro: Coroutine[T\_co, T\_contra, V\_co], task\_complete: Callable = None, ident: str = None, timing:*

*dict = None*) → *\_asyncio.Task*

Start executing a coroutine as an async task, bypassing the pending queue.

**Parameters**

- **coro** – The coroutine to run
- **task\_complete** – An optional callback to run on completion
- **ident** – A string identifier for the task
- **timing** – An optional dictionary of timing information

Returns: the new asyncio task instance

**wait\_for** (*timeout: float*)

Wait for all queued tasks to complete with a timeout.

`aries_cloudagent.utils.task_queue.coro_ident` (*coro: Coroutine[T\_co, T\_contra, V\_co]*)  
Extract an identifier for a coroutine.

`aries_cloudagent.utils.task_queue.coro_timed` (*coro: Coroutine[T\_co, T\_contra, V\_co]*,  
*timing: dict*)  
Capture timing for a coroutine.

`aries_cloudagent.utils.task_queue.task_exc_info` (*task: \_asyncio.Task*)  
Extract exception info from an asyncio task.

### 1.1.13 aries\_cloudagent.verifier package

#### Submodules

#### aries\_cloudagent.verifier.base module

Base Verifier class.

**class** `aries_cloudagent.verifier.base.BaseVerifier`  
Bases: `abc.ABC`

Base class for verifier.

**verify\_presentation** (*presentation\_request, presentation, schemas, credential\_definitions,*  
*rev\_reg\_defs, rev\_reg\_entries*)  
Verify a presentation.

#### Parameters

- **presentation\_request** – Presentation request data
- **presentation** – Presentation data
- **schemas** – Schema data
- **credential\_definitions** – credential definition data
- **rev\_reg\_defs** – revocation registry definitions
- **rev\_reg\_entries** – revocation registry entries

#### aries\_cloudagent.verifier.indy module

Indy verifier implementation.

**class** `aries_cloudagent.verifier.indy.IndyVerifier` (*wallet*)  
Bases: `aries_cloudagent.verifier.base.BaseVerifier`

Indy verifier class.

**static pre\_verify** (*pres\_req: dict, pres: dict*) -> (`<enum 'PreVerifyResult'>`, `<class 'str'>`)  
Check for essential components and tampering in presentation.

Visit encoded attribute values against raw, and predicate bounds, in presentation, cross-reference against presentation request.

#### Parameters

- **pres\_req** – presentation request
- **pres** – corresponding presentation



**Returns** An instance of *PreVerifyResult* representing the validation result

**verify\_presentation** (*presentation\_request*, *presentation*, *schemas*, *credential\_definitions*,  
*rev\_reg\_defs*, *rev\_reg\_entries*) → bool

Verify a presentation.

**Parameters**

- **presentation\_request** – Presentation request data
- **presentation** – Presentation data
- **schemas** – Schema data
- **credential\_definitions** – credential definition data
- **rev\_reg\_defs** – revocation registry definitions
- **rev\_reg\_entries** – revocation registry entries

**class** aries\_cloudagent.verifier.indy.PreVerifyResult

Bases: `enum.Enum`

Represent the result of IndyVerifier.pre\_verify.

**ENCODING\_MISMATCH** = 'demonstrates tampering with raw values'

**INCOMPLETE** = 'missing essential components'

**OK** = 'ok'

## 1.1.14 aries\_cloudagent.wallet package

Abstract and Indy wallet handling.

### Submodules

#### aries\_cloudagent.wallet.base module

Wallet base class.

**class** aries\_cloudagent.wallet.base.BaseWallet (*config: dict*)

Bases: `abc.ABC`

Abstract wallet interface.

**close** ()

Close previously-opened wallet, removing it if so configured.

**create\_local\_did** (*seed: str = None*, *did: str = None*, *metadata: dict = None*) →  
aries\_cloudagent.wallet.base.DIDInfo

Create and store a new local DID.

**Parameters**

- **seed** – Optional seed to use for did
- **did** – The DID to use
- **metadata** – Metadata to store with DID

**Returns** The created *DIDInfo*

**create\_public\_did** (*seed: str = None, did: str = None, metadata: dict = {}*) → *aries\_cloudagent.wallet.base.DIDInfo*  
 Create and store a new public DID.

Implicitly flags all other dids as not public.

**Parameters**

- **seed** – Optional seed to use for did
- **did** – The DID to use
- **metadata** – Metadata to store with DID

**Returns** The created *DIDInfo*

**create\_signing\_key** (*seed: str = None, metadata: dict = None*) → *aries\_cloudagent.wallet.base.KeyInfo*  
 Create a new public/private signing keypair.

**Parameters**

- **seed** – Optional seed allowing deterministic key creation
- **metadata** – Optional metadata to store with the keypair

**Returns** A *KeyInfo* representing the new record

**created**

Check whether the wallet was created on the last open call.

**get\_local\_did** (*did: str*) → *aries\_cloudagent.wallet.base.DIDInfo*  
 Find info for a local DID.

**Parameters** **did** – The DID to get info for

**Returns** A *DIDInfo* instance for the DID

**get\_local\_did\_for\_verkey** (*verkey: str*) → *aries\_cloudagent.wallet.base.DIDInfo*  
 Resolve a local DID from a verkey.

**Parameters** **verkey** – Verkey to get DID info for

**Returns** A *DIDInfo* instance for the DID

**get\_local\_dids** () → *Sequence[aries\_cloudagent.wallet.base.DIDInfo]*  
 Get list of defined local DIDs.

**Returns** A list of *DIDInfo* instances

**get\_public\_did** () → *aries\_cloudagent.wallet.base.DIDInfo*  
 Retrieve the public did.

**Returns** The created *DIDInfo*

**get\_signing\_key** (*verkey: str*) → *aries\_cloudagent.wallet.base.KeyInfo*  
 Fetch info for a signing keypair.

**Parameters** **verkey** – The verification key of the keypair

**Returns** A *KeyInfo* representing the keypair

**handle**

Get internal wallet reference.

**Returns** Defaults to None

**name**

Accessor for the wallet name.

**open()**

Open wallet, removing and/or creating it if so configured.

**opened**

Check whether wallet is currently open.

**pack\_message** (*message: str, to\_verkeys: Sequence[str], from\_verkey: str = None*) → bytes

Pack a message for one or more recipients.

**Parameters**

- **message** – The message to pack
- **to\_verkeys** – The verkeys to pack the message for
- **from\_verkey** – The sender verkey

**Returns** The packed message

**replace\_local\_did\_metadata** (*did: str, metadata: dict*)

Replace the metadata associated with a local DID.

**Parameters**

- **did** – DID to replace metadata for
- **metadata** – The new metadata

**replace\_signing\_key\_metadata** (*verkey: str, metadata: dict*)

Replace the metadata associated with a signing keypair.

**Parameters**

- **verkey** – The verification key of the keypair
- **metadata** – The new metadata to store

**set\_public\_did** (*did: str*) → aries\_cloudagent.wallet.base.DIDInfo

Assign the public did.

**Returns** The created *DIDInfo*

**sign\_message** (*message: bytes, from\_verkey: str*) → bytes

Sign a message using the private key associated with a given verkey.

**Parameters**

- **message** – The message to sign
- **from\_verkey** – Sign using the private key related to this verkey

**Returns** The signature

**type**

Accessor for the wallet type.

**unpack\_message** (*enc\_message: bytes*) -> (<class 'str'>, <class 'str'>, <class 'str'>)

Unpack a message.

**Parameters** **enc\_message** – The encrypted message

**Returns** (message, from\_verkey, to\_verkey)

**Return type** A tuple

**verify\_message** (*message: bytes, signature: bytes, from\_verkey: str*) → bool  
 Verify a signature against the public key of the signer.

**Parameters**

- **message** – The message to verify
- **signature** – The signature to verify
- **from\_verkey** – Verkey to use in verification

**Returns** True if verified, else False

**class** aries\_cloudagent.wallet.base.**DIDInfo** (*did, verkey, metadata*)  
 Bases: `tuple`

**did**  
 Alias for field number 0

**metadata**  
 Alias for field number 2

**verkey**  
 Alias for field number 1

**class** aries\_cloudagent.wallet.base.**KeyInfo** (*verkey, metadata*)  
 Bases: `tuple`

**metadata**  
 Alias for field number 1

**verkey**  
 Alias for field number 0

## aries\_cloudagent.wallet.basic module

In-memory implementation of BaseWallet interface.

**class** aries\_cloudagent.wallet.basic.**BasicWallet** (*config: dict = None*)  
 Bases: `aries_cloudagent.wallet.base.BaseWallet`

In-memory wallet implementation.

**WALLET\_TYPE** = 'basic'

**close**()  
 Not applicable to in-memory wallet.

**create\_local\_did** (*seed: str = None, did: str = None, metadata: dict = None*) → `aries_cloudagent.wallet.base.DIDInfo`  
 Create and store a new local DID.

**Parameters**

- **seed** – Optional seed to use for did
- **did** – The DID to use
- **metadata** – Metadata to store with DID

**Returns** A *DIDInfo* instance representing the created DID

**Raises** `WalletDuplicateError` – If the DID already exists in the wallet

**create\_signing\_key** (*seed: str = None, metadata: dict = None*) → *aries\_cloudagent.wallet.base.KeyInfo*  
 Create a new public/private signing keypair.

**Parameters**

- **seed** – Seed to use for signing key
- **metadata** – Optional metadata to store with the keypair

**Returns** A *KeyInfo* representing the new record

**Raises** *WalletDuplicateError* – If the resulting verkey already exists in the wallet

**created**

Check whether the wallet was created on the last open call.

**get\_local\_did** (*did: str*) → *aries\_cloudagent.wallet.base.DIDInfo*  
 Find info for a local DID.

**Parameters** **did** – The DID to get info for

**Returns** A *DIDInfo* instance representing the found DID

**Raises** *WalletNotFoundError* – If the DID is not found

**get\_local\_did\_for\_verkey** (*verkey: str*) → *aries\_cloudagent.wallet.base.DIDInfo*  
 Resolve a local DID from a verkey.

**Parameters** **verkey** – The verkey to get the local DID for

**Returns** A *DIDInfo* instance representing the found DID

**Raises** *WalletNotFoundError* – If the verkey is not found

**get\_local\_dids** () → *Sequence[aries\_cloudagent.wallet.base.DIDInfo]*  
 Get list of defined local DIDs.

**Returns** A list of locally stored DIDs as *DIDInfo* instances

**get\_signing\_key** (*verkey: str*) → *aries\_cloudagent.wallet.base.KeyInfo*  
 Fetch info for a signing keypair.

**Parameters** **verkey** – The verification key of the keypair

**Returns** A *KeyInfo* representing the keypair

**Raises** *WalletNotFoundError* – if no keypair is associated with the verification key

**name**

Accessor for the wallet name.

**open** ()

Not applicable to in-memory wallet.

**opened**

Check whether wallet is currently open.

**Returns** True

**pack\_message** (*message: str, to\_verkeys: Sequence[str], from\_verkey: str = None*) → bytes  
 Pack a message for one or more recipients.

**Parameters**

- **message** – The message to pack
- **to\_verkeys** – List of verkeys to pack for

- **from\_verkey** – Sender verkey to pack from

**Returns** The resulting packed message bytes

**Raises** `WalletError` – If the message is not provided

**replace\_local\_did\_metadata** (*did: str, metadata: dict*)

Replace metadata for a local DID.

**Parameters**

- **did** – The DID to replace metadata for
- **metadata** – The new metadata

**Raises** `WalletNotFoundError` – If the DID doesn't exist

**replace\_signing\_key\_metadata** (*verkey: str, metadata: dict*)

Replace the metadata associated with a signing keypair.

**Parameters**

- **verkey** – The verification key of the keypair
- **metadata** – The new metadata to store

**Raises** `WalletNotFoundError` – if no keypair is associated with the verification key

**sign\_message** (*message: bytes, from\_verkey: str*) → bytes

Sign a message using the private key associated with a given verkey.

**Parameters**

- **message** – Message bytes to sign
- **from\_verkey** – The verkey to use to sign

**Returns** A signature

**Raises**

- `WalletError` – If the message is not provided
- `WalletError` – If the verkey is not provided

**type**

Accessor for the wallet type.

**unpack\_message** (*enc\_message: bytes*) -> (<class 'str'>, <class 'str'>, <class 'str'>)

Unpack a message.

**Parameters** **enc\_message** – The packed message bytes

**Returns** (message, from\_verkey, to\_verkey)

**Return type** A tuple

**Raises**

- `WalletError` – If the message is not provided
- `WalletError` – If there is a problem unpacking the message

**verify\_message** (*message: bytes, signature: bytes, from\_verkey: str*) → bool

Verify a signature against the public key of the signer.

**Parameters**

- **message** – Message to verify

- **signature** – Signature to verify
- **from\_verkey** – Verkey to use in verification

**Returns** True if verified, else False

**Raises**

- `WalletError` – If the verkey is not provided
- `WalletError` – If the signature is not provided
- `WalletError` – If the message is not provided

## aries\_cloudagent.wallet.crypto module

Cryptography functions used by BasicWallet.

```
class aries_cloudagent.wallet.crypto.PackMessageSchema (*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)
```

Bases: `marshmallow.schema.Schema`

Packed message schema.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.wallet.crypto.PackRecipientHeaderSchema (*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)
```

Bases: `marshmallow.schema.Schema`

Packed recipient header schema.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.wallet.crypto.PackRecipientSchema (*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)
```

Bases: `marshmallow.schema.Schema`

Packed recipient schema.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.wallet.crypto.PackRecipientsSchema (*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)
```

Bases: `marshmallow.schema.Schema`

Packed recipients schema.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
aries_cloudagent.wallet.crypto.create_keypair (seed: bytes = None) → Tuple[bytes, bytes]
```

Create a public and private signing keypair from a seed value.

**Parameters** **seed** – Seed for keypair

**Returns** A tuple of (public key, secret key)

```
aries_cloudagent.wallet.crypto.decode_pack_message (enc_message: bytes, find_key: Callable) → Tuple[str, Optional[str], str]
```

Decode a packed message.

Disassemble and unencrypt a packed message, returning the message content, verification key of the sender (if available), and verification key of the recipient.

**Parameters**

- **enc\_message** – The encrypted message
- **find\_key** – Function to retrieve private key

**Returns** A tuple of (message, sender\_vk, recip\_vk)

**Raises**

- `ValueError` – If the packed message is invalid
- `ValueError` – If the packed message recipients are invalid
- `ValueError` – If the pack algorithm is unsupported
- `ValueError` – If the sender's public key was not provided

```
aries_cloudagent.wallet.crypto.decode_pack_message_outer (enc_message: bytes) → Tuple[dict, dict, bool]
```

Decode the outer wrapper of a packed message and extract the recipients.

**Parameters** **enc\_message** – The encrypted message

Returns: a tuple of the decoded wrapper, recipients, and authcrypt flag

```
aries_cloudagent.wallet.crypto.decode_pack_message_payload (wrapper: dict, payload_key: bytes) → str
```

Decode the payload of a packed message once the CEK is known.

**Parameters**

- **wrapper** – The decoded message wrapper
- **payload\_key** – The decrypted payload key



`aries_cloudagent.wallet.crypto.decrypt_plaintext` (*ciphertext: bytes, recips\_bin: bytes, nonce: bytes, key: bytes*) → str

Decrypt the payload of a packed message.

**Parameters**

- **ciphertext** –
- **recips\_bin** –
- **nonce** –
- **key** –

**Returns** The decrypted string

`aries_cloudagent.wallet.crypto.encode_pack_message` (*message: str, to\_verkeys: Sequence[bytes], from\_secret: bytes = None*) → bytes

Assemble a packed message for a set of recipients, optionally including the sender.

**Parameters**

- **message** – The message to pack
- **to\_verkeys** – The verkeys to pack the message for
- **from\_secret** – The sender secret

**Returns** The encoded message

`aries_cloudagent.wallet.crypto.encrypt_plaintext` (*message: str, add\_data: bytes, key: bytes*) → Tuple[bytes, bytes, bytes]

Encrypt the payload of a packed message.

**Parameters**

- **message** – Message to encrypt
- **add\_data** –
- **key** – Key used for encryption

**Returns** A tuple of (ciphertext, nonce, tag)

`aries_cloudagent.wallet.crypto.extract_pack_recipients` (*recipients: Sequence[dict]*) → dict

Extract the pack message recipients into a dict indexed by verkey.

**Parameters** **recipients** – Recipients to locate

**Raises** `ValueError` – If the recipients block is mal-formatted

`aries_cloudagent.wallet.crypto.extract_payload_key` (*sender\_cek: dict, recip\_secret: bytes*) → Tuple[bytes, str]

Extract the payload key from pack recipient details.

**Returns:** A tuple of the CEK and sender verkey

`aries_cloudagent.wallet.crypto.prepare_pack_recipient_keys` (*to\_verkeys: Sequence[bytes], from\_secret: bytes = None*) → Tuple[str, bytes]

Assemble the recipients block of a packed message.

**Parameters**

- **to\_verkeys** – Verkeys of recipients
- **from\_secret** – Secret to use for signing keys

**Returns** A tuple of (json result, key)

`aries_cloudagent.wallet.crypto.random_seed()` → bytes  
Generate a random seed value.

**Returns** A new random seed

`aries_cloudagent.wallet.crypto.seed_to_did(seed: str)` → str  
Derive a DID from a seed value.

**Parameters** **seed** – The seed to derive

**Returns** The DID derived from the seed

`aries_cloudagent.wallet.crypto.sign_message(message: bytes, secret: bytes)` → bytes  
Sign a message using a private signing key.

**Parameters**

- **message** – The message to sign
- **secret** – The private signing key

**Returns** The signature

`aries_cloudagent.wallet.crypto.sign_pk_from_sk(secret: bytes)` → bytes  
Extract the verkey from a secret signing key.

`aries_cloudagent.wallet.crypto.validate_seed(seed: (<class 'str'>, <class 'bytes'>))` → bytes  
Convert a seed parameter to standard format and check length.

**Parameters** **seed** – The seed to validate

**Returns** The validated and encoded seed

`aries_cloudagent.wallet.crypto.verify_signed_message(signed: bytes, verkey: bytes)` → bool  
Verify a signed message according to a public verification key.

**Parameters**

- **signed** – The signed message
- **verkey** – The verkey to use in verification

**Returns** True if verified, else False

## aries\_cloudagent.wallet.error module

Wallet-related exceptions.

**exception** `aries_cloudagent.wallet.error.WalletDuplicateError(*args, error_code: str = None, **kwargs)`

Bases: `aries_cloudagent.wallet.error.WalletError`

Duplicate record exception.

**exception** `aries_cloudagent.wallet.error.WalletError(*args, error_code: str = None, **kwargs)`

Bases: `aries_cloudagent.core.error.BaseError`

General wallet exception.

```
exception aries_cloudagent.wallet.error.WalletNotFoundError(*args, error_code:
                                                         str = None,
                                                         **kwargs)
```

Bases: *aries\_cloudagent.wallet.error.WalletError*

Record not found exception.

## aries\_cloudagent.wallet.indy module

Indy implementation of BaseWallet interface.

```
class aries_cloudagent.wallet.indy.IndyWallet(config: dict = None)
    Bases: aries_cloudagent.wallet.base.BaseWallet
```

Indy wallet implementation.

**DEFAULT\_FRESHNESS** = 0

**DEFAULT\_KEY** = ''

**DEFAULT\_KEY\_DERIVATION** = 'ARGON2I\_MOD'

**DEFAULT\_NAME** = 'default'

**DEFAULT\_STORAGE\_TYPE** = None

**KEY\_DERIVATION\_ARGON2I\_INT** = 'ARGON2I\_INT'

**KEY\_DERIVATION\_ARGON2I\_MOD** = 'ARGON2I\_MOD'

**KEY\_DERIVATION\_RAW** = 'RAW'

**WALLET\_TYPE** = 'indy'

**close()**

Close previously-opened wallet, removing it if so configured.

**create(replace: bool = False)**

Create a new wallet.

**Parameters** **replace** – Removes the old wallet if True

**Raises**

- **WalletError** – If there was a problem removing the wallet
- **WalletError** – IF there was a libindy error

**create\_local\_did(seed: str = None, did: str = None, metadata: dict = None) →**  
                                   *aries\_cloudagent.wallet.base.DIDInfo*

Create and store a new local DID.

**Parameters**

- **seed** – Optional seed to use for did
- **did** – The DID to use
- **metadata** – Metadata to store with DID

**Returns** A *DIDInfo* instance representing the created DID

**Raises**

- **WalletDuplicateError** – If the DID already exists in the wallet

- `WalletError` – If there is a libindy error

**create\_signing\_key** (*seed: str = None, metadata: dict = None*) → `aries_cloudagent.wallet.base.KeyInfo`

Create a new public/private signing keypair.

**Parameters**

- **seed** – Seed for key
- **metadata** – Optional metadata to store with the keypair

**Returns** A *KeyInfo* representing the new record

**Raises**

- `WalletDuplicateError` – If the resulting verkey already exists in the wallet
- `WalletError` – If there is a libindy error

**created**

Check whether the wallet was created on the last open call.

**classmethod generate\_wallet\_key** (*seed: str = None*) → `str`

Generate a raw Indy wallet key.

**get\_credential\_definition\_tag\_policy** (*credential\_definition\_id: str*)

Return the tag policy for a given credential definition ID.

**get\_local\_did** (*did: str*) → `aries_cloudagent.wallet.base.DIDInfo`

Find info for a local DID.

**Parameters** **did** – The DID to get info for

**Returns** A *DIDInfo* instance representing the found DID

**Raises**

- `WalletNotFoundError` – If the DID is not found
- `WalletError` – If there is a libindy error

**get\_local\_did\_for\_verkey** (*verkey: str*) → `aries_cloudagent.wallet.base.DIDInfo`

Resolve a local DID from a verkey.

**Parameters** **verkey** – The verkey to get the local DID for

**Returns** A *DIDInfo* instance representing the found DID

**Raises** `WalletNotFoundError` – If the verkey is not found

**get\_local\_dids** () → `Sequence[aries_cloudagent.wallet.base.DIDInfo]`

Get list of defined local DIDs.

**Returns** A list of locally stored DIDs as *DIDInfo* instances

**get\_signing\_key** (*verkey: str*) → `aries_cloudagent.wallet.base.KeyInfo`

Fetch info for a signing keypair.

**Parameters** **verkey** – The verification key of the keypair

**Returns** A *KeyInfo* representing the keypair

**Raises**

- `WalletNotFoundError` – If no keypair is associated with the verification key
- `WalletError` – If there is a libindy error

**handle**

Get internal wallet reference.

**Returns** A handle to the wallet

**master\_secret\_id**

Accessor for the master secret id.

**Returns** The master secret id

**name**

Accessor for the wallet name.

**Returns** The wallet name

**open()**

Open wallet, removing and/or creating it if so configured.

**Raises**

- `WalletError` – If wallet not found after creation
- `WalletNotFoundError` – If the wallet is not found
- `WalletError` – If the wallet is already open
- `WalletError` – If there is a libindy error

**opened**

Check whether wallet is currently open.

**Returns** True if open, else False

**pack\_message** (*message: str, to\_verkeys: Sequence[str], from\_verkey: str = None*) → bytes

Pack a message for one or more recipients.

**Parameters**

- **message** – The message to pack
- **to\_verkeys** – List of verkeys to pack for
- **from\_verkey** – Sender verkey to pack from

**Returns** The resulting packed message bytes

**Raises**

- `WalletError` – If no message is provided
- `WalletError` – If a libindy error occurs

**remove()**

Remove an existing wallet.

**Raises**

- `WalletNotFoundError` – If the wallet could not be found
- `WalletError` – If there was an libindy error

**replace\_local\_did\_metadata** (*did: str, metadata: dict*)

Replace metadata for a local DID.

**Parameters**

- **did** – The DID to replace metadata for
- **metadata** – The new metadata

**replace\_signing\_key\_metadata** (*verkey: str, metadata: dict*)

Replace the metadata associated with a signing keypair.

**Parameters**

- **verkey** – The verification key of the keypair
- **metadata** – The new metadata to store

**Raises** `WalletNotFoundError` – if no keypair is associated with the verification key

**set\_credential\_definition\_tag\_policy** (*credential\_definition\_id: str, taggables: Sequence[str] = None, retroactive: bool = True*)

Set the tag policy for a given credential definition ID.

**Parameters**

- **credential\_definition\_id** – The ID of the credential definition
- **taggables** – A sequence of string values representing attribute names; empty array for none, None for all
- **retroactive** – Whether to apply the policy to previously-stored credentials

**sign\_message** (*message: bytes, from\_verkey: str*) → bytes

Sign a message using the private key associated with a given verkey.

**Parameters**

- **message** – Message bytes to sign
- **from\_verkey** – The verkey to use to sign

**Returns** A signature

**Raises**

- `WalletError` – If the message is not provided
- `WalletError` – If the verkey is not provided
- `WalletError` – If a libindy error occurs

**type**

Accessor for the wallet type.

**unpack\_message** (*enc\_message: bytes*) -> (<class 'str'>, <class 'str'>, <class 'str'>)

Unpack a message.

**Parameters** **enc\_message** – The packed message bytes

**Returns** (message, from\_verkey, to\_verkey)

**Return type** A tuple

**Raises**

- `WalletError` – If the message is not provided
- `WalletError` – If a libindy error occurs

**verify\_message** (*message: bytes, signature: bytes, from\_verkey: str*) → bool

Verify a signature against the public key of the signer.

**Parameters**

- **message** – Message to verify
- **signature** – Signature to verify

- **from\_verkey** – Verkey to use in verification

**Returns** True if verified, else False

**Raises**

- `WalletError` – If the verkey is not provided
- `WalletError` – If the signature is not provided
- `WalletError` – If the message is not provided
- `WalletError` – If a libindy error occurs

## aries\_cloudagent.wallet.plugin module

Utility for loading Postgres wallet plug-in.

`aries_cloudagent.wallet.plugin.file_ext()`

Determine file extension based on platform.

`aries_cloudagent.wallet.plugin.load_postgres_plugin(storage_config, storage_creds, raise_exc=False)`

Load postgres dll and configure postgres wallet.

## aries\_cloudagent.wallet.provider module

Default wallet provider classes.

**class** `aries_cloudagent.wallet.provider.WalletProvider`

Bases: `aries_cloudagent.config.base.BaseProvider`

Provider for the default configurable wallet classes.

**WALLET\_TYPES** = {'basic': 'aries\_cloudagent.wallet.basic.BasicWallet', 'indy': 'aries

**provide** (*settings:* `aries_cloudagent.config.base.BaseSettings`, *injector:* `aries_cloudagent.config.base.BaseInjector`)

Create and open the wallet instance.

## aries\_cloudagent.wallet.routes module

Wallet admin routes.

**class** `aries_cloudagent.wallet.routes.DIDListSchema` (\*, *only=None*, *exclude=()*, *many=False*, *context=None*, *load\_only=()*, *dump\_only=()*, *partial=False*, *unknown=None*)

Bases: `marshmallow.schema.Schema`

Result schema for connection list.

**opts** = `<marshmallow.schema.SchemaOpts object>`

**class** `aries_cloudagent.wallet.routes.DIDResultSchema` (\*, *only=None*, *exclude=()*, *many=False*, *context=None*, *load\_only=()*, *dump\_only=()*, *partial=False*, *unknown=None*)

Bases: `marshmallow.schema.Schema`

Result schema for a DID.

**opts** = <marshmallow.schema.SchemaOpts object>

```
class aries_cloudagent.wallet.routes.DIDSchema (*, only=None, exclude=(), many=False,
                                              context=None, load_only=(),
                                              dump_only=(), partial=False, un-
                                              known=None)
```

Bases: marshmallow.schema.Schema

Result schema for a DID.

**opts** = <marshmallow.schema.SchemaOpts object>

```
class aries_cloudagent.wallet.routes.GetTagPolicyResultSchema (*, only=None,
                                                             exclude=(),
                                                             many=False,
                                                             context=None,
                                                             load_only=(),
                                                             dump_only=(),
                                                             partial=False,
                                                             unknown=None)
```

Bases: marshmallow.schema.Schema

Result schema for tagging policy get request.

**opts** = <marshmallow.schema.SchemaOpts object>

```
class aries_cloudagent.wallet.routes.SetTagPolicyRequestSchema (*, only=None,
                                                                exclude=(),
                                                                many=False,
                                                                context=None,
                                                                load_only=(),
                                                                dump_only=(),
                                                                par-
                                                                tial=False, un-
                                                                known=None)
```

Bases: marshmallow.schema.Schema

Request schema for tagging policy set request.

**opts** = <marshmallow.schema.SchemaOpts object>

**aries\_cloudagent.wallet.routes.format\_did\_info** (info: aries\_cloudagent.wallet.base.DIDInfo)  
Serialize a DIDInfo object.

**aries\_cloudagent.wallet.routes.register** (app: <sphinx.ext.autodoc.importer.\_MockObject  
object at 0x7f7951333198>)

Register routes.

**aries\_cloudagent.wallet.routes.wallet\_create\_did** (request:  
<sphinx.ext.autodoc.importer.\_MockObject  
object at 0x7f7951333198>)

Request handler for creating a new wallet DID.

**Parameters** **request** – aiohttp request object

**Returns** The DID info

**aries\_cloudagent.wallet.routes.wallet\_did\_list** (request:  
<sphinx.ext.autodoc.importer.\_MockObject  
object at 0x7f7951333198>)

Request handler for searching wallet DIDs.



**Parameters** **request** – aiohttp request object

**Returns** The DID list response

```
aries_cloudagent.wallet.routes.wallet_get_public_did(request:
                                                    <sphinx.ext.autodoc.importer._MockObject
                                                    object at 0x7f7951333198>)
```

Request handler for fetching the current public DID.

**Parameters** **request** – aiohttp request object

**Returns** The DID info

```
aries_cloudagent.wallet.routes.wallet_get_tagging_policy(request:
                                                         <sphinx.ext.autodoc.importer._MockObject
                                                         object at
                                                         0x7f7951333198>)
```

Request handler for getting the tag policy associated with a cred def.

**Parameters** **request** – aiohttp request object

**Returns** A JSON object containing the tagging policy

```
aries_cloudagent.wallet.routes.wallet_set_public_did(request:
                                                         <sphinx.ext.autodoc.importer._MockObject
                                                         object at 0x7f7951333198>)
```

Request handler for setting the current public DID.

**Parameters** **request** – aiohttp request object

**Returns** The updated DID info

```
aries_cloudagent.wallet.routes.wallet_set_tagging_policy(request:
                                                         <sphinx.ext.autodoc.importer._MockObject
                                                         object at
                                                         0x7f7951333198>)
```

Request handler for setting the tag policy associated with a cred def.

**Parameters** **request** – aiohttp request object

**Returns** An empty JSON response

## aries\_cloudagent.wallet.util module

Wallet utility functions.

```
aries_cloudagent.wallet.util.b58_to_bytes(val: str) → bytes
    Convert a base 58 string to bytes.
```

```
aries_cloudagent.wallet.util.b64_to_bytes(val: str, urlsafe=False) → bytes
    Convert a base 64 string to bytes.
```

```
aries_cloudagent.wallet.util.b64_to_str(val: str, urlsafe=False, encoding=None) → str
    Convert a base 64 string to string on input encoding (default utf-8).
```

```
aries_cloudagent.wallet.util.bytes_to_b58(val: bytes) → str
    Convert a byte string to base 58.
```

```
aries_cloudagent.wallet.util.bytes_to_b64(val: bytes, urlsafe=False, pad=True) → str
    Convert a byte string to base 64.
```

```
aries_cloudagent.wallet.util.pad(val: str) → str
    Pad base64 values if need be: JWT calls to omit trailing padding.
```

`aries_cloudagent.wallet.util.set_urlsafe_b64` (*val: str, urlsafe: bool = True*) → str  
Set URL safety in base64 encoding.

`aries_cloudagent.wallet.util.str_to_b64` (*val: str, urlsafe=False, encoding=None, pad=True*) → str  
Convert a string to base64 string on input encoding (default utf-8).

`aries_cloudagent.wallet.util.unpad` (*val: str*) → str  
Remove padding from base64 values if need be.

## 1.2 Submodules

### 1.3 `aries_cloudagent.version` module

Library version information.

---

## aries\_cloudagent.connections package

---

### 2.1 Subpackages

#### 2.1.1 aries\_cloudagent.connections.models package

##### Subpackages

##### aries\_cloudagent.connections.models.diddoc package

DID Document model support.

Copyright 2017-2019 Government of Canada Public Services and Procurement Canada - buyandsell.gc.ca

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

**class** aries\_cloudagent.connections.models.diddoc.DIDDoc (did: str = None)

Bases: object

DID document, grouping a DID with verification keys and services.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

**CONTEXT** = 'https://w3id.org/did/v1'

**add\_service\_pubkeys** (service: dict, tags: Union[Sequence[str], str]) →  
List[aries\_cloudagent.connections.models.diddoc.publickey.PublicKey]

Add public keys specified in service. Return public keys so discovered.

**Parameters**

- **service** – service from DID document
- **tags** – potential tags marking public keys of type of interest (the standard is still coalescing)

**Raises** `ValueError` – for public key reference not present in DID document.

Returns: list of public keys from the document service specification

#### **authnkey**

Accessor for public keys marked as authentication keys, by identifier.

**classmethod** **deserialize** (*did\_doc: dict*) → `aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc`  
Construct DIDDoc object from dict representation.

**Parameters** **did\_doc** – DIDDoc dict representation

**Raises** `ValueError` – for bad DID or missing mandatory item.

Returns: DIDDoc from input json

#### **did**

Accessor for DID.

**classmethod** **from\_json** (*did\_doc\_json: str*) → `aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc`  
Construct DIDDoc object from json representation.

**Parameters** **did\_doc\_json** – DIDDoc json representation

Returns: DIDDoc from input json

#### **pubkey**

Accessor for public keys by identifier.

**serialize** () → `str`  
Dump current object to a JSON-compatible dictionary.

**Returns** dict representation of current DIDDoc

#### **service**

Accessor for services by identifier.

**set** (*item: Union[aries\_cloudagent.connections.models.diddoc.service.Service, aries\_cloudagent.connections.models.diddoc.publickey.PublicKey]*) → `aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc`  
Add or replace service or public key; return current DIDDoc.

**Raises** `ValueError` – if input item is neither service nor public key.

**Parameters** **item** – service or public key to set

Returns: the current DIDDoc

**to\_json** () → `str`  
Dump current object as json (JSON-LD).

**Returns** json representation of current DIDDoc

**class** `aries_cloudagent.connections.models.diddoc.LinkedDataKeySpec` (*ver\_type, au-  
thn\_type, specifier*)

Bases: `tuple`

#### **authn\_type**

Alias for field number 1

**specifier**

Alias for field number 2

**ver\_type**

Alias for field number 0

```

class aries_cloudagent.connections.models.diddoc.PublicKey (did: str, ident: str,
                                                            value: str, pk_type:
                                                            aries_cloudagent.connections.models.diddoc.p
                                                            = None, controller:
                                                            str = None, authn:
                                                            bool = False)

```

Bases: `object`

Public key specification to embed in DID document.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

**authn**

Accessor for the authentication marker.

Returns: whether public key is marked as having DID authentication privilege

**controller**

Accessor for the controller DID.

**did**

Accessor for the DID.

**id**

Accessor for the public key identifier.

**to\_dict** () → dict

Return dict representation of public key to embed in DID document.

**type**

Accessor for the public key type.

**value**

Accessor for the public key value.

```

class aries_cloudagent.connections.models.diddoc.PublicKeyType

```

Bases: `enum.Enum`

Class encapsulating public key types.

**ED25519\_SIG\_2018** = `LinkedDataKeySpec(ver_type='Ed25519VerificationKey2018', authn_type=`**EDDSA\_SA\_SIG\_SECP256K1** = `LinkedDataKeySpec(ver_type='Secp256k1VerificationKey2018', au`**RSA\_SIG\_2018** = `LinkedDataKeySpec(ver_type='RsaVerificationKey2018', authn_type='RsaSig`**authn\_type**

Accessor for the authentication type identifier.

**get** = `<function PublicKeyType.get>`**specification** (val: str) → str

Return specifier and input value for use in public key specification.

**Parameters** **val** – value of public key

Returns: dict mapping applicable specifier to input value

**specifier**

Accessor for the value specifier.

**ver\_type**

Accessor for the verification type identifier.

```
class aries_cloudagent.connections.models.diddoc.Service(did: str, ident: str,  
                                                    typ: str, recip_keys:  
                                                    Union[Sequence[T_co],  
                                                    aries_cloudagent.connections.models.diddoc.publ  
                                                    routing_keys:  
                                                    Union[Sequence[T_co],  
                                                    aries_cloudagent.connections.models.diddoc.publ  
                                                    endpoint: str, priority:  
                                                    int = 0)
```

Bases: `object`

Service specification to embed in DID document.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

**did**

Accessor for the DID value.

**endpoint**

Accessor for the endpoint value.

**id**

Accessor for the service identifier.

**priority**

Accessor for the priority value.

**recip\_keys**

Accessor for the recipient keys.

**routing\_keys**

Accessor for the routing keys.

**to\_dict** () → dict

Return dict representation of service to embed in DID document.

**type**

Accessor for the service type.

## Submodules

### aries\_cloudagent.connections.models.diddoc.diddoc module

DID Document classes.

Copyright 2017-2019 Government of Canada Public Services and Procurement Canada - buyandsell.gc.ca

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

```
class aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc(did: str = None)
```

Bases: `object`

DID document, grouping a DID with verification keys and services.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

**CONTEXT** = 'https://w3id.org/did/v1'

```
add_service_pubkeys(service: dict, tags: Union[Sequence[str], str]) → List[aries_cloudagent.connections.models.diddoc.publickey.PublicKey]
```

Add public keys specified in service. Return public keys so discovered.

#### Parameters

- **service** – service from DID document
- **tags** – potential tags marking public keys of type of interest (the standard is still coalescing)

**Raises** `ValueError` – for public key reference not present in DID document.

Returns: list of public keys from the document service specification

#### authnkey

Accessor for public keys marked as authentication keys, by identifier.

```
classmethod deserialize(did_doc: dict) → aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc
```

Construct DIDDoc object from dict representation.

**Parameters** *did\_doc* – DIDDoc dict representation

**Raises** `ValueError` – for bad DID or missing mandatory item.

Returns: DIDDoc from input json

#### did

Accessor for DID.

```
classmethod from_json(did_doc_json: str) → aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc
```

Construct DIDDoc object from json representation.

**Parameters** *did\_doc\_json* – DIDDoc json representation

Returns: DIDDoc from input json

#### pubkey

Accessor for public keys by identifier.

```
serialize() → str
```

Dump current object to a JSON-compatible dictionary.

**Returns** dict representation of current DIDDoc

#### service

Accessor for services by identifier.

```
set (item: Union[aries_cloudagent.connections.models.diddoc.service.Service,
    aries_cloudagent.connections.models.diddoc.publickey.PublicKey]) →
    aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc
    Add or replace service or public key; return current DIDDoc.
```

**Raises** `ValueError` – if input item is neither service nor public key.

**Parameters** `item` – service or public key to set

Returns: the current DIDDoc

```
to_json () → str
    Dump current object as json (JSON-LD).
```

**Returns** json representation of current DIDDoc

## aries\_cloudagent.connections.models.diddoc.publickey module

DID Document Public Key classes.

Copyright 2017-2019 Government of Canada Public Services and Procurement Canada - buyandsell.gc.ca

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

```
class aries_cloudagent.connections.models.diddoc.publickey.LinkedDataKeySpec (ver_type,
    au-
    thn_type,
    spec-
    i-
    fier)
```

Bases: `tuple`

**authn\_type**

Alias for field number 1

**specifier**

Alias for field number 2

**ver\_type**

Alias for field number 0



```

class aries_cloudagent.connections.models.diddoc.publickey.PublicKey (did:
                                                                    str,
                                                                    ident:
                                                                    str,
                                                                    value:
                                                                    str,
                                                                    pk_type:
                                                                    aries_cloudagent.connections.m
                                                                    =
                                                                    None,
                                                                    con-
                                                                    troller:
                                                                    str =
                                                                    None,
                                                                    authn:
                                                                    bool =
                                                                    False)

```

Bases: `object`

Public key specification to embed in DID document.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

**authn**

Accessor for the authentication marker.

Returns: whether public key is marked as having DID authentication privilege

**controller**

Accessor for the controller DID.

**did**

Accessor for the DID.

**id**

Accessor for the public key identifier.

**to\_dict** () → dict

Return dict representation of public key to embed in DID document.

**type**

Accessor for the public key type.

**value**

Accessor for the public key value.

```

class aries_cloudagent.connections.models.diddoc.publickey.PublicKeyType

```

Bases: `enum.Enum`

Class encapsulating public key types.

```

ED25519_SIG_2018 = LinkedDataKeySpec(ver_type='Ed25519VerificationKey2018', authn_type=

```

```

EDDSA_SA_SIG_SECP256K1 = LinkedDataKeySpec(ver_type='Secp256k1VerificationKey2018', au

```

```

RSA_SIG_2018 = LinkedDataKeySpec(ver_type='RsaVerificationKey2018', authn_type='RsaSig

```

**authn\_type**

Accessor for the authentication type identifier.

```

get = <function PublicKeyType.get>

```

**specification** (*val: str*) → str

Return specifier and input value for use in public key specification.

**Parameters** **val** – value of public key

Returns: dict mapping applicable specifier to input value

**specifier**

Accessor for the value specifier.

**ver\_type**

Accessor for the verification type identifier.

### **aries\_cloudagent.connections.models.diddoc.service module**

DID Document Service classes.

Copyright 2017-2019 Government of Canada Public Services and Procurement Canada - buyandsell.gc.ca

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

```
class aries_cloudagent.connections.models.diddoc.service.Service (did: str,  
ident: str,  
typ: str,  
recip_keys:  
Union[Sequence[T_co],  
aries_cloudagent.connections.models.  
routing_keys:  
Union[Sequence[T_co],  
aries_cloudagent.connections.models.  
endpoint: str,  
priority: int  
= 0)
```

Bases: `object`

Service specification to embed in DID document.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

**did**

Accessor for the DID value.

**endpoint**

Accessor for the endpoint value.

**id**

Accessor for the service identifier.

**priority**

Accessor for the priority value.

**recip\_keys**

Accessor for the recipient keys.

**routing\_keys**

Accessor for the routing keys.

**to\_dict()** → dict

Return dict representation of service to embed in DID document.

**type**

Accessor for the service type.

**aries\_cloudagent.connections.models.diddoc.util module**

DIDDoc utility methods.

Copyright 2017-2019 Government of Canada Public Services and Procurement Canada - buyandsell.gc.ca

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

`aries_cloudagent.connections.models.diddoc.util.canon_did(uri: str) → str`

Convert a URI into a DID if need be, left-stripping ‘did:sov:’ if present.

**Parameters** `uri` – input URI or DID

**Raises** `ValueError` – for invalid input.

`aries_cloudagent.connections.models.diddoc.util.canon_ref(did: str, ref: str, delimiter: str = None)`

Given a reference in a DID document, return it in its canonical form of a URI.

**Parameters**

- **did** – DID acting as the identifier of the DID document
- **ref** – reference to canonicalize, either a DID or a fragment pointing to a location in the DID doc
- **delimiter** – delimiter character marking fragment (default ‘#’) or introducing identifier (‘;’) against DID resource

`aries_cloudagent.connections.models.diddoc.util.ok_did(token: str) → bool`

Whether input token looks like a valid decentralized identifier.

**Parameters** `token` – candidate string

Returns: whether input token looks like a valid schema identifier

`aries_cloudagent.connections.models.diddoc.util.resource(ref: str, delimiter: str = None) → str`

Extract the resource for an identifier.

Given a (URI) reference, return up to its delimiter (exclusively), or all of it if there is none.

**Parameters**

- **ref** – reference
- **delimiter** – delimiter character (default None maps to ‘#’, or ‘;’ introduces identifiers)

### Submodules

#### `aries_cloudagent.connections.models.connection_record` module

Handle connection information interface with non-secrets storage.

```
class aries_cloudagent.connections.models.connection_record.ConnectionRecord(*,
    connection_id: str
    = None,
    my_did: str
    = None,
    their_did: str
    = None,
    their_label: str
    = None,
    their_role: str
    = None,
    initiator: str
    = None,
    invitation_key: str
    = None,
    request_id: str
    = None,
    state: str
    = None,
    inbound_connection_id: str
    = None,
    error_msg: str
    = None,
    routing_state: str
    = None,
```

Bases: `aries_cloudagent.messaging.models.base_record.BaseRecord`

Represents a single pairwise connection.

```
ACCEPT_AUTO = 'auto'
ACCEPT_MANUAL = 'manual'
CACHE_ENABLED = True
DIRECTION_RECEIVED = 'received'
DIRECTION_SENT = 'sent'
INITIATOR_EXTERNAL = 'external'
INITIATOR_MULTIUSE = 'multiuse'
INITIATOR_SELF = 'self'
INVITATION_MODE_MULTI = 'multi'
INVITATION_MODE_ONCE = 'once'
INVITATION_MODE_STATIC = 'static'
LOG_STATE_FLAG = 'debug.connections'

class Meta
    Bases: object
    ConnectionRecord metadata.
    schema_class = 'ConnectionRecordSchema'

RECORD_ID_NAME = 'connection_id'
RECORD_TYPE = 'connection'
RECORD_TYPE_INVITATION = 'connection_invitation'
RECORD_TYPE_REQUEST = 'connection_request'
ROUTING_STATE_ACTIVE = 'active'
ROUTING_STATE_ERROR = 'error'
ROUTING_STATE_NONE = 'none'
ROUTING_STATE_REQUEST = 'request'
STATE_ACTIVE = 'active'
STATE_ERROR = 'error'
STATE_INACTIVE = 'inactive'
STATE_INIT = 'init'
STATE_INVITATION = 'invitation'
STATE_REQUEST = 'request'
STATE_RESPONSE = 'response'
TAG_NAMES = {'invitation_key', 'my_did', 'request_id', 'their_did'}
WEBHOOK_TOPIC = 'connections'
```

**attach\_invitation** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, invitation: aries\_cloudagent.protocols.connections.v1\_0.messages.connection\_invitation.ConnectionInvitation*)  
Persist the related connection invitation to storage.

#### Parameters

- **context** – The injection context to use
- **invitation** – The invitation to relate to this connection record

**attach\_request** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, request: aries\_cloudagent.protocols.connections.v1\_0.messages.connection\_request.ConnectionRequest*)  
Persist the related connection request to storage.

#### Parameters

- **context** – The injection context to use
- **request** – The request to relate to this connection record

**connection\_id**  
Accessor for the ID associated with this connection.

**is\_multiuse\_invitation**  
Accessor for multi use invitation mode.

**is\_ready**  
Accessor for connection readiness.

**post\_save** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, \*args, \*\*kwargs*)  
Perform post-save actions.

**Parameters context** – The injection context to use

**record\_value**  
Accessor to for the JSON record value properties for this connection.

**classmethod retrieve\_by\_did** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, their\_did: str = None, my\_did: str = None, initiator: str = None*) → *aries\_cloudagent.connections.models.connection\_record.ConnectionRecord*  
Retrieve a connection record by target DID.

#### Parameters

- **context** – The injection context to use
- **their\_did** – The target DID to filter by
- **my\_did** – One of our DIDs to filter by
- **initiator** – Filter connections by the initiator value

**classmethod retrieve\_by\_invitation\_key** (*context: aries\_cloudagent.config.injection\_context.InjectionContext, invitation\_key: str, initiator: str = None*) → *aries\_cloudagent.connections.models.connection\_record.ConnectionRecord*  
Retrieve a connection record by invitation key.

#### Parameters

- **context** – The injection context to use
- **invitation\_key** – The key on the originating invitation
- **initiator** – Filter by the initiator value

```
classmethod retrieve_by_request_id (context: aries_cloudagent.config.injection_context.InjectionContext,  
                                     request_id: str) →  
                                     aries_cloudagent.connections.models.connection_record.ConnectionRecord
```

Retrieve a connection record from our previous request ID.

**Parameters**

- **context** – The injection context to use
- **request\_id** – The ID of the originating connection request

```
retrieve_invitation (context: aries_cloudagent.config.injection_context.InjectionContext) →  
                     aries_cloudagent.protocols.connections.v1_0.messages.connection_invitation.ConnectionInvitation
```

Retrieve the related connection invitation.

**Parameters context** – The injection context to use

```
retrieve_request (context: aries_cloudagent.config.injection_context.InjectionContext) →  
                  aries_cloudagent.protocols.connections.v1_0.messages.connection_request.ConnectionRequest
```

Retrieve the related connection invitation.

**Parameters context** – The injection context to use

```
class aries_cloudagent.connections.models.connection_record.ConnectionRecordSchema (*args,  
                                                                                       **kwargs)
```

Bases: *aries\_cloudagent.messaging.models.base\_record.BaseRecordSchema*

Schema to allow serialization/deserialization of connection records.

```
class Meta
```

Bases: *object*

ConnectionRecordSchema metadata.

```
model_class
```

alias of *ConnectionRecord*

```
accept = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<OneOf  
alias = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, r  
connection_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=  
error_msg = <fields.String(default=<marshmallow.missing>, attribute=None, validate=Non  
inbound_connection_id = <fields.String(default=<marshmallow.missing>, attribute=None, v  
initiator = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<On  
invitation_key = <fields.String(default=<marshmallow.missing>, attribute=None, validat  
invitation_mode = <fields.String(default=<marshmallow.missing>, attribute=None, valid  
my_did = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<aries  
request_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=No  
routing_state = <fields.String(default=<marshmallow.missing>, attribute=None, validate=  
their_did = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<ar  
their_label = <fields.String(default=<marshmallow.missing>, attribute=None, validate=N  
their_role = <fields.String(default=<marshmallow.missing>, attribute=None, validate=No
```



**aries\_cloudagent.connections.models.connection\_target module**

Record used to handle routing of messages to another agent.

```
class aries_cloudagent.connections.models.connection_target.ConnectionTarget (*,
    did:
    str
    =
    None,
    end-
    point:
    str
    =
    None,
    la-
    bel:
    str
    =
    None,
    re-
    cip-
    i-
    ent_keys:
    Se-
    quence[str]
    =
    None,
    rout-
    ing_keys:
    Se-
    quence[str]
    =
    None,
    sender_key:
    str
    =
    None)
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModel*

Record used to handle routing of messages to another agent.

```
class Meta
```

Bases: *object*

ConnectionTarget metadata.

```
    schema_class = 'ConnectionTargetSchema'
```

```
class aries_cloudagent.connections.models.connection_target.ConnectionTargetSchema (*args,
    **kwargs)
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModelSchema*

ConnectionTarget schema.

```
class Meta
```

Bases: *object*

ConnectionTargetSchema metadata.

**model\_class**  
alias of *ConnectionTarget*

**did**  
Used by autodoc\_mock\_imports.

**endpoint**  
Used by autodoc\_mock\_imports.

**label**  
Used by autodoc\_mock\_imports.

**recipient\_keys**  
Used by autodoc\_mock\_imports.

**routing\_keys**  
Used by autodoc\_mock\_imports.

**sender\_key**  
Used by autodoc\_mock\_imports.

---

aries\_cloudagent.protocols package

---

### 3.1 Subpackages

#### 3.1.1 aries\_cloudagent.protocols.actionmenu package

##### Subpackages

aries\_cloudagent.protocols.actionmenu.handlers package

##### Submodules

aries\_cloudagent.protocols.actionmenu.handlers.menu\_handler module

aries\_cloudagent.protocols.actionmenu.handlers.menu\_request\_handler module

aries\_cloudagent.protocols.actionmenu.handlers.perform\_handler module

aries\_cloudagent.protocols.actionmenu.messages package

##### Submodules

aries\_cloudagent.protocols.actionmenu.messages.menu module

aries\_cloudagent.protocols.actionmenu.messages.menu\_request module

aries\_cloudagent.protocols.actionmenu.messages.perform module

`aries_cloudagent.protocols.actionmenu.models` package

#### Submodules

`aries_cloudagent.protocols.actionmenu.models.menu_form` module

`aries_cloudagent.protocols.actionmenu.models.menu_form_param` module

`aries_cloudagent.protocols.actionmenu.models.menu_option` module

#### Submodules

`aries_cloudagent.protocols.actionmenu.base_service` module

`aries_cloudagent.protocols.actionmenu.controller` module

`aries_cloudagent.protocols.actionmenu.driver_service` module

`aries_cloudagent.protocols.actionmenu.message_types` module

`aries_cloudagent.protocols.actionmenu.routes` module

`aries_cloudagent.protocols.actionmenu.util` module

### 3.1.2 `aries_cloudagent.protocols.basicmessage` package

#### Subpackages

`aries_cloudagent.protocols.basicmessage.handlers` package

#### Submodules

`aries_cloudagent.protocols.basicmessage.handlers.basicmessage_handler` module

`aries_cloudagent.protocols.basicmessage.messages` package

#### Submodules

`aries_cloudagent.protocols.basicmessage.messages.basicmessage` module

#### Submodules

`aries_cloudagent.protocols.basicmessage.message_types` module

`aries_cloudagent.protocols.basicmessage.routes` module

### 3.1.3 `aries_cloudagent.protocols.connections` package

### Subpackages

`aries_cloudagent.protocols.connections.handlers` package

### Submodules

`aries_cloudagent.protocols.connections.handlers.connection_invitation_handler` module

`aries_cloudagent.protocols.connections.handlers.connection_request_handler` module

`aries_cloudagent.protocols.connections.handlers.connection_response_handler` module

`aries_cloudagent.protocols.connections.messages` package

### Submodules

`aries_cloudagent.protocols.connections.messages.connection_invitation` module

`aries_cloudagent.protocols.connections.messages.connection_request` module

`aries_cloudagent.protocols.connections.messages.connection_response` module

`aries_cloudagent.protocols.connections.messages.problem_report` module

`aries_cloudagent.protocols.connections.models` package

### Submodules

`aries_cloudagent.protocols.connections.models.connection_detail` module

### Submodules

`aries_cloudagent.protocols.connections.manager` module

`aries_cloudagent.protocols.connections.message_types` module

`aries_cloudagent.protocols.connections.routes` module

## 3.1.4 `aries_cloudagent.protocols.credentials` package

### Subpackages

`aries_cloudagent.protocols.credentials.handlers` package

### Submodules

`aries_cloudagent.protocols.credentials.handlers.credential_issue_handler` module

`aries_cloudagent.protocols.credentials.handlers.credential_offer_handler` module

`aries_cloudagent.protocols.credentials.handlers.credential_request_handler` module

`aries_cloudagent.protocols.credentials.handlers.credential_stored_handler` module

`aries_cloudagent.protocols.credentials.messages` package

#### Submodules

`aries_cloudagent.protocols.credentials.messages.credential_issue` module

`aries_cloudagent.protocols.credentials.messages.credential_offer` module

`aries_cloudagent.protocols.credentials.messages.credential_request` module

`aries_cloudagent.protocols.credentials.messages.credential_stored` module

`aries_cloudagent.protocols.credentials.models` package

#### Submodules

`aries_cloudagent.protocols.credentials.models.credential_exchange` module

#### Submodules

`aries_cloudagent.protocols.credentials.manager` module

`aries_cloudagent.protocols.credentials.message_types` module

`aries_cloudagent.protocols.credentials.routes` module

### 3.1.5 `aries_cloudagent.protocols.discovery` package

#### Subpackages

`aries_cloudagent.protocols.discovery.handlers` package

#### Submodules

`aries_cloudagent.protocols.discovery.handlers.disclose_handler` module

`aries_cloudagent.protocols.discovery.handlers.query_handler` module

`aries_cloudagent.protocols.discovery.messages` package

#### Submodules

`aries_cloudagent.protocols.discovery.messages.disclose` module

`aries_cloudagent.protocols.discovery.messages.query` module

#### Submodules

`aries_cloudagent.protocols.discovery.message_types` module

`aries_cloudagent.protocols.discovery.routes` module

### 3.1.6 `aries_cloudagent.protocols.introduction` package

#### Subpackages

`aries_cloudagent.protocols.introduction.handlers` package

#### Submodules

`aries_cloudagent.protocols.introduction.handlers.forward_invitation_handler` module

`aries_cloudagent.protocols.introduction.handlers.invitation_handler` module

`aries_cloudagent.protocols.introduction.handlers.invitation_request_handler` module

`aries_cloudagent.protocols.introduction.messages` package

#### Submodules

`aries_cloudagent.protocols.introduction.messages.forward_invitation` module

`aries_cloudagent.protocols.introduction.messages.invitation` module

`aries_cloudagent.protocols.introduction.messages.invitation_request` module

#### Submodules

`aries_cloudagent.protocols.introduction.base_service` module

`aries_cloudagent.protocols.introduction.demo_service` module

`aries_cloudagent.protocols.introduction.message_types` module

`aries_cloudagent.protocols.introduction.routes` module

### 3.1.7 `aries_cloudagent.protocols.issue_credential` package

#### Subpackages

### aries\_cloudagent.protocols.issue\_credential.v1\_0 package

#### Subpackages

### aries\_cloudagent.protocols.issue\_credential.v1\_0.handlers package

#### Submodules

### aries\_cloudagent.protocols.issue\_credential.v1\_0.handlers.credential\_ack\_handler module

Credential ack message handler.

```
class aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_ack_handler.CredentialAckHandler
    Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Message handler class for credential acks.

```
handle (context:          aries_cloudagent.messaging.request_context.RequestContext,      responder:
        aries_cloudagent.messaging.responder.BaseResponder)
    Message handler logic for credential acks.
```

#### Parameters

- **context** – request context
- **responder** – responder callback

### aries\_cloudagent.protocols.issue\_credential.v1\_0.handlers.credential\_issue\_handler module

Credential issue message handler.

```
class aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_issue_handler.CredentialIssueHandler
    Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Message handler class for credential offers.

```
handle (context:          aries_cloudagent.messaging.request_context.RequestContext,      responder:
        aries_cloudagent.messaging.responder.BaseResponder)
    Message handler logic for credential offers.
```

#### Parameters

- **context** – request context
- **responder** – responder callback

### aries\_cloudagent.protocols.issue\_credential.v1\_0.handlers.credential\_offer\_handler module

Credential offer message handler.

```
class aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_offer_handler.CredentialOfferHandler
    Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Message handler class for credential offers.

```
handle (context:          aries_cloudagent.messaging.request_context.RequestContext,      responder:
        aries_cloudagent.messaging.responder.BaseResponder)
    Message handler logic for credential offers.
```



### Parameters

- **context** – request context
- **responder** – responder callback

## `aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_proposal_handler` module

Credential proposal message handler.

```
class aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_proposal_handler
    Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Message handler class for credential proposals.

```
handle (context:          aries_cloudagent.messaging.request_context.RequestContext,      responder:
        aries_cloudagent.messaging.responder.BaseResponder)
    Message handler logic for credential proposals.
```

### Parameters

- **context** – proposal context
- **responder** – responder callback

## `aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_request_handler` module

Credential request message handler.

```
class aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_request_handler
    Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Message handler class for credential requests.

```
handle (context:          aries_cloudagent.messaging.request_context.RequestContext,      responder:
        aries_cloudagent.messaging.responder.BaseResponder)
    Message handler logic for credential requests.
```

### Parameters

- **context** – request context
- **responder** – responder callback

## `aries_cloudagent.protocols.issue_credential.v1_0.messages` package

### Subpackages

## `aries_cloudagent.protocols.issue_credential.v1_0.messages.inner` package

### Submodules

## `aries_cloudagent.protocols.issue_credential.v1_0.messages.inner.credential_preview` module

A credential preview inner object.

```
class aries_cloudagent.protocols.issue_credential.v1_0.messages.inner.credential_preview.C
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModel*

Class representing a preview of an attribute.

```
class Meta
```

Bases: *object*

Attribute preview metadata.

```
schema_class = 'CredAttrSpecSchema'
```

```
b64_decoded_value() → str
```

Value, base64-decoded if applicable.

```
static list_plain(plain: dict)
```

Return a list of *CredAttrSpec* without MIME types from names/values.

**Parameters** *plain* – dict mapping names to values

**Returns** List of *CredAttrSpec*s with no MIME types

```
class aries_cloudagent.protocols.issue_credential.v1_0.messages.inner.credential_preview.C
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModelSchema*

Attribute preview schema.

```
class Meta
```

Bases: *object*

Attribute preview schema metadata.

```
model_class
```

alias of *CredAttrSpec*

```
mime_type = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
```

```
name = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, re
```

```
value = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, r
```

```
class aries_cloudagent.protocols.issue_credential.v1_0.messages.inner.credential_preview.C
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModel*

Class representing a credential preview inner object.

**class Meta**

Bases: *object*

Credential preview metadata.

**message\_type** = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/issue-credential/1.0/credential'

**schema\_class** = 'CredentialPreviewSchema'

**attr\_dict** (*decode: bool = False*)

Return name:value pair per attribute.

**Parameters** **decode** – whether first to decode attributes with MIME type

**mime\_types** ()

Return per-attribute mapping from name to MIME type.

Return empty dict if no attribute has MIME type.

**class** *aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.inner.credential\_preview.CredentialPreview*

Bases: *aries\_cloudagent.messaging.models.base.BaseModelSchema*

Credential preview schema.

**class Meta**

Bases: *object*

Credential preview schema metadata.

**model\_class**

alias of *CredentialPreview*

**attributes** = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None)

## Submodules

### *aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_ack* module

A credential ack message.

**class** *aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_ack.CredentialAck*

Bases: *aries\_cloudagent.messaging.ack.message.Ack*

Class representing a credential ack message.

**class Meta**

Bases: *object*

Credential metadata.

**handler\_class** = 'aries\_cloudagent.protocols.issue\_credential.v1\_0.handlers.credential\_ack'

**message\_type** = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/issue-credential/1.0/ack'

**schema\_class** = 'CredentialAckSchema'

**class** *aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_ack.CredentialAckSchema*

Bases: *aries\_cloudagent.messaging.ack.message.AckSchema*

Credential ack schema.

```
class Meta
    Bases: object
    Schema metadata.
    model_class
        alias of CredentialAck
```

## aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_issue module

A credential content message.

```
class aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue.Credentialia
```

Bases: *aries\_cloudagent.messaging.agent\_message.AgentMessage*

Class representing a credential.

```
class Meta
    Bases: object
    Credential metadata.
    handler_class = 'aries_cloudagent.protocols.issue_credential.v1_0.handlers.credenti
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/issue-credential/1.0/issue-cred
    schema_class = 'CredentialIssueSchema'
```

```
indy_credential (index: int = 0)
```

Retrieve and decode indy credential from attachment.

**Parameters** **index** – ordinal in attachment list to decode and return (typically, list has length 1)

```
classmethod wrap_indy_credential (indy_cred: dict) →
    aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator
    Convert an indy credential offer to an attachment decorator.
```

```
class aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue.Credentialia
```

Bases: *aries\_cloudagent.messaging.agent\_message.AgentMessageSchema*

Credential schema.

```
class Meta
```

```
    Bases: object
```

```
    Credential schema metadata.
```

```
    model_class
```

```
        alias of CredentialIssue
```

```
    comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
```

```
    credentials_attach = <fields.Nested(default=<marshmallow.missing>, attribute=None, val
```

### `aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer` module

A credential offer content message.

```
class aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer.Credentialia
```

Bases: *aries\_cloudagent.messaging.agent\_message.AgentMessage*

Class representing a credential offer.

```
class Meta
```

```
    Bases: object
```

```
    CredentialOffer metadata.
```

```
    handler_class = 'aries_cloudagent.protocols.issue_credential.v1_0.handlers.credenti
```

```
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/issue-credential/1.0/offer-cred
```

```
    schema_class = 'CredentialOfferSchema'
```

```
    indy_offer (index: int = 0) → dict
```

```
        Retrieve and decode indy offer from attachment.
```

**Parameters** `index` – ordinal in attachment list to decode and return (typically, list has length 1)

**classmethod** `wrap_indy_offer` (*indy\_offer: dict*) → `aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator`  
Convert an indy credential offer to an attachment decorator.

**class** `aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer.CredentialOffer`

Bases: `aries_cloudagent.messaging.agent_message.AgentMessageSchema`

Credential offer schema.

**class** `Meta`

Bases: `object`

Credential offer schema metadata.

**model\_class**

alias of `CredentialOffer`

`comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,`

`credential_preview = <fields.Nested(default=<marshmallow.missing>, attribute=None, val`

`offers_attach = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate`

### `aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal` module

A credential proposal content message.

```
class aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal.CredentialProposal
```

Bases: *aries\_cloudagent.messaging.agent\_message.AgentMessage*

Class representing a credential proposal.

```
class Meta
```

Bases: *object*

CredentialProposal metadata.

```
handler_class = 'aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_proposal_handler'
```

```
message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/issue-credential/1.0/propose-credential'
```

```
schema_class = 'CredentialProposalSchema'
```

**class** aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_proposal.**CredentialProposal**

Bases: *aries\_cloudagent.messaging.agent\_message.AgentMessageSchema*

Credential proposal schema.

**class Meta**

Bases: *object*

Credential proposal schema metadata.

**model\_class**

alias of *CredentialProposal*

```
comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
cred_def_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<
credential_proposal = <fields.Nested(default=<marshmallow.missing>, attribute=None, va
issuer_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<a
schema_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<ar
schema_issuer_id = <fields.String(default=<marshmallow.missing>, attribute=None, vali
schema_name = <fields.String(default=<marshmallow.missing>, attribute=None, validate=N
schema_version = <fields.String(default=<marshmallow.missing>, attribute=None, validate=
```

## aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_request module

A credential request content message.

**class** aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_request.**CredentialRequest**

Bases: *aries\_cloudagent.messaging.agent\_message.AgentMessage*

Class representing a credential request.

**class Meta**

Bases: *object*

CredentialRequest metadata.

**handler\_class** = 'aries\_cloudagent.protocols.issue\_credential.v1\_0.handlers.credenti



```

    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/issue-credential/1.0/request-credential'
    schema_class = 'CredentialRequestSchema'

indy_cred_req (index: int = 0)
    Retrieve and decode indy credential request from attachment.

    Parameters index – ordinal in attachment list to decode and return (typically, list has length 1)

classmethod wrap_indy_cred_req (indy_cred_req: dict) →
    aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator
    Convert an indy credential request to an attachment decorator.

class aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_request.CredentialRequest
    Bases: aries_cloudagent.messaging.agent_message.AgentMessageSchema
    Credential request schema.

    class Meta
        Bases: object
        Credential request schema metadata.

    model_class
        alias of CredentialRequest

    comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, if_not_none=' ')>
    requests_attach = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None, if_not_none=' ')>

```

## aries\_cloudagent.protocols.issue\_credential.v1\_0.models package

### Submodules

#### aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange module

Aries#0036 v1.0 credential exchange information with non-secrets storage.

```
class aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10Creden
```

Bases: *aries\_cloudagent.messaging.models.base\_record.BaseExchangeRecord*

Represents an Aries#0036 credential exchange.

**INITIATOR\_EXTERNAL** = 'external'

**INITIATOR\_SELF** = 'self'

**class Meta**

Bases: *object*

CredentialExchange metadata.

**schema\_class** = 'V10CredentialExchangeSchema'

**RECORD\_ID\_NAME** = 'credential\_exchange\_id'

**RECORD\_TYPE** = 'credential\_exchange\_v10'

**ROLE HOLDER** = 'holder'

**ROLE\_ISSUER** = 'issuer'

**STATE\_ACKED** = 'credential\_acked'

**STATE\_CREDENTIAL\_RECEIVED** = 'credential\_received'

**STATE\_ISSUED** = 'credential\_issued'

**STATE\_OFFER\_RECEIVED** = 'offer\_received'

**STATE\_OFFER\_SENT** = 'offer\_sent'

**STATE\_PROPOSAL\_RECEIVED** = 'proposal\_received'

**STATE\_PROPOSAL\_SENT** = 'proposal\_sent'

**STATE\_REQUEST\_RECEIVED** = 'request\_received'

**STATE\_REQUEST\_SENT** = 'request\_sent'

**TAG\_NAMES** = {'thread\_id'}

**WEBHOOK\_TOPIC** = 'issue\_credential'

**credential\_exchange\_id**

Accessor for the ID associated with this exchange.

**record\_value**

Accessor for the JSON record value generated for this credential exchange.

**classmethod retrieve\_by\_connection\_and\_thread** (*context*:

*aries\_cloudagent.config.injection\_context.InjectionContext,*

*connection\_id:* *str,*

*thread\_id:* *str*) *→*

*aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange*

Retrieve a credential exchange record by connection and thread ID.

**class** *aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange*

Bases: *aries\_cloudagent.messaging.models.base\_record.BaseExchangeSchema*

Schema to allow serialization/deserialization of credential exchange records.

**class Meta**

Bases: *object*

V10CredentialExchangeSchema metadata.

```
model_class
    alias of V10CredentialExchange

auto_issue = <fields.Boolean(default=<marshmallow.missing>, attribute=None, validate=None)
auto_offer = <fields.Boolean(default=<marshmallow.missing>, attribute=None, validate=None)
auto_remove = <fields.Boolean(default=True, attribute=None, validate=None, required=False)
connection_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
credential = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
credential_definition_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
credential_exchange_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
credential_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
credential_offer = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
credential_proposal_dict = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
credential_request = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
credential_request_metadata = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
error_msg = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
initiator = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
parent_thread_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
raw_credential = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
revoc_reg_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
revocation_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
role = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
schema_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
state = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
thread_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
```

## Submodules

### `aries_cloudagent.protocols.issue_credential.v1_0.manager` module

Classes to manage credentials.

```
class aries_cloudagent.protocols.issue_credential.v1_0.manager.CredentialManager(context: aries_cloudagent
```

Bases: `object`

Class for managing credentials.

**context**

Accessor for the current request context.

**Returns** The request context for this connection

**create\_offer** (*credential\_exchange\_record: aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange*, *comment: str = None*) → Tuple[aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange, aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_offer.CredentialOffer]  
Create a credential offer, update credential exchange record.

#### Parameters

- **credential\_exchange\_record** – Credential exchange to create offer for
- **comment** – optional human-readable comment to set in offer message

**Returns** A tuple (credential exchange record, credential offer message)

**create\_proposal** (*connection\_id: str, \*, auto\_offer: bool = None, auto\_remove: bool = None, comment: str = None, credential\_preview: aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.inner.credential\_preview.CredentialPreview = None, schema\_id: str = None, schema\_issuer\_did: str = None, schema\_name: str = None, schema\_version: str = None, cred\_def\_id: str = None, issuer\_did: str = None, trace: bool = False*) → aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange  
Create a credential proposal.

#### Parameters

- **connection\_id** – Connection to create proposal for
- **auto\_offer** – Should this proposal request automatically be handled to offer a credential
- **auto\_remove** – Should the record be automatically removed on completion
- **comment** – Optional human-readable comment to include in proposal
- **credential\_preview** – The credential preview to use to create the credential proposal
- **schema\_id** – Schema id for credential proposal
- **schema\_issuer\_did** – Schema issuer DID for credential proposal
- **schema\_name** – Schema name for credential proposal
- **schema\_version** – Schema version for credential proposal
- **cred\_def\_id** – Credential definition id for credential proposal
- **issuer\_did** – Issuer DID for credential proposal

**Returns** Resulting credential exchange record including credential proposal

**create\_request** (*credential\_exchange\_record: aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange*, *holder\_did: str*) → Tuple[aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange, aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_request.CredentialRequest]  
Create a credential request.

#### Parameters

- **credential\_exchange\_record** – Credential exchange record for which to create request
- **holder\_did** – holder DID

**Returns** A tuple (credential exchange record, credential request message)

**issue\_credential** (*credential\_exchange\_record: aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.CredentialExchange, comment: str = None, credential\_values: dict*) → Tuple[aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange, aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_issue.CredentialIssue]

Issue a credential.

#### Parameters

- **credential\_exchange\_record** – The credential exchange record for which to issue a credential
- **comment** – optional human-readable comment pertaining to credential issue
- **credential\_values** – dict of credential attribute {name: value} pairs

**Returns** (Updated credential exchange record, credential message)

**Return type** Tuple

**prepare\_send** (*connection\_id: str, credential\_proposal: aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_proposal.CredentialProposal, auto\_remove: bool = None*) → Tuple[aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange, aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_offer.CredentialOffer]

Set up a new credential exchange for an automated send.

#### Parameters

- **connection\_id** – Connection to create offer for
- **credential\_proposal** – The credential proposal with preview on attribute values to use if auto\_issue is enabled
- **auto\_remove** – Flag to automatically remove the record on completion

**Returns** A tuple of the new credential exchange record and credential offer message

**publish\_pending\_revocations** () → Mapping[str, Sequence[str]]

Publish pending revocations to the ledger.

Returns: mapping from each revocation registry id to its cred rev ids published.

**receive\_credential** () → aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange

Receive a credential from an issuer.

Hold in storage potentially to be processed by controller before storing.

**Returns** Credential exchange record, retrieved and updated

**receive\_credential\_ack** () → aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange

Receive credential ack from holder.

**Returns** credential exchange record, retrieved and updated

**receive\_offer** () → aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange

Receive a credential offer.

**Returns** The credential exchange record, updated

**receive\_proposal** () → aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange

Receive a credential proposal from message in context on manager creation.

**Returns** The resulting credential exchange record, created

**receive\_request** ()

Receive a credential request.

**Parameters** **credential\_request\_message** – Credential request to receive

**Returns** credential exchange record, retrieved and updated

**revoke\_credential** (*rev\_reg\_id: str, cred\_rev\_id: str, publish: bool = False*)

Revoke a previously-issued credential.

Optionally, publish the corresponding revocation registry delta to the ledger.

**Parameters**

- **rev\_reg\_id** – revocation registry id
- **cred\_rev\_id** – credential revocation id
- **publish** – whether to publish the resulting revocation registry delta

**store\_credential** (*credential\_exchange\_record: aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange\_record.CredentialExchangeRecord, credential\_id: str = None*) → Tuple[aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange\_record.CredentialExchangeRecord, aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_ack.CredentialAck]

Store a credential in holder wallet; send ack to issuer.

**Parameters**

- **credential\_exchange\_record** – credential exchange record with credential to store and ack
- **credential\_id** – optional credential identifier to override default on storage

**Returns** (Updated credential exchange record, credential ack message)

**Return type** Tuple

**exception** aries\_cloudagent.protocols.issue\_credential.v1\_0.manager.CredentialManagerError (

Bases: *aries\_cloudagent.core.error.BaseError*

Credential error.

## aries\_cloudagent.protocols.issue\_credential.v1\_0.message\_types module

Message and inner object type identifiers for Connections.

## aries\_cloudagent.protocols.issue\_credential.v1\_0.routes module

Credential exchange admin routes.

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10AttributeMimeTypesResultsS
```

Bases: `marshmallow.schema.Schema`

Result schema for credential attribute MIME types by credential definition.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialExchangeListResu
```

Bases: `marshmallow.schema.Schema`

Result schema for Aries#0036 v1.0 credential exchange query.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialIssueRequestSch
```

Bases: `marshmallow.schema.Schema`

Request schema for sending credential issue admin message.

**opts** = `<marshmallow.schema.SchemaOpts object>`



```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialOfferRequestSch
```

Bases: `aries_cloudagent.utils.tracing.AdminAPIMessageTracingSchema`

Request schema for sending credential offer admin message.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialProblemReportRe
```

Bases: `marshmallow.schema.Schema`

Request schema for sending problem report.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialProposalRequest
```

Bases: `aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialProposalRequestSchemaBase`

Request schema for sending credential proposal on mandatory proposal preview.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialProposalRequestSchema
```

Bases: `aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialProposalRequestSchemaBase`

Request schema for sending credential proposal on optional proposal preview.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialProposalRequestSchema
```

Bases: `aries_cloudagent.utils.tracing.AdminAPIMessageTracingSchema`

Base class for request schema for sending credential proposal admin message.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialStoreRequestSchema
```

Bases: `marshmallow.schema.Schema`

Request schema for sending a credential store admin message.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10PublishRevocationsResultsS
```

Bases: `marshmallow.schema.Schema`

Result schema for revocation publication API call.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
aries_cloudagent.protocols.issue_credential.v1_0.routes.attribute_mime_types_get (request:
<sphinx.ext.
ob-
ject
at
0x7f79512d48
```

Request handler for getting credential attribute MIME types.

**Parameters** **request** – aiohttp request object

**Returns** The MIME types response

```
aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_issue (request:
<sphinx.ext.
ob-
ject
at
0x7f79512d4
```

Request handler for sending credential.

**Parameters** **request** – aiohttp request object

**Returns** The credential exchange record

```
aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_list (request:
<sphinx.ext.
ob-
ject
at
0x7f79512d48
```

Request handler for searching connection records.

**Parameters** **request** – aiohttp request object

**Returns** The connection list response

```
aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_problem_report
```

Request handler for sending problem report.

**Parameters** `request` – aiohttp request object

`aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_publish_revocation`

Request handler for publishing pending revocations to the ledger.

**Parameters** `request` – aiohttp request object

**Returns** Credential revocation ids published as revoked by revocation registry id.

`aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_remove` (`request: <sphinx.ext.autodoc.Object at 0x7f79512d4810`

Request handler for removing a credential exchange record.

**Parameters** `request` – aiohttp request object

`aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_retrieve` (`request: <sphinx.ext.autodoc.Object at 0x7f79512d4810`

Request handler for fetching single connection record.

**Parameters** `request` – aiohttp request object

**Returns** The credential exchange record

`aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_revoke` (`request: <sphinx.ext.autodoc.Object at 0x7f79512d4810`

Request handler for storing a credential request.

**Parameters** `request` – aiohttp request object

**Returns** The credential request details.

`aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_send` (`request: <sphinx.ext.autodoc.Object at 0x7f79512d4810`

Request handler for sending credential from issuer to holder from attr values.

If both issuer and holder are configured for automatic responses, the operation ultimately results in credential issue; otherwise, the result waits on the first response not automated; the credential exchange record retains state regardless.

**Parameters** `request` – aiohttp request object

**Returns** The credential exchange record

`aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_send_bound_offer`

Request handler for sending bound credential offer.

A holder initiates this sequence with a credential proposal; this message responds with an offer bound to the proposal.

**Parameters** `request` – aiohttp request object

**Returns** The credential exchange record

`aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_send_free_offer`

Request handler for sending free credential offer.

An issuer initiates a such a credential offer, free from any holder-initiated corresponding credential proposal with preview.

**Parameters** `request` – aiohttp request object

**Returns** The credential exchange record

`aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_send_proposal` (`request`)

Request handler for sending credential proposal.

**Parameters** `request` – aiohttp request object

**Returns** The credential exchange record

`aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_send_request` (`request`)

Request handler for sending credential request.

**Parameters** `request` – aiohttp request object

**Returns** The credential exchange record

`aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_store` (*request:* `<sphinx.ext.ob-`  
`ject`  
`at`  
`0x7f79512d4`

Request handler for storing credential.

**Parameters** `request` – aiohttp request object

**Returns** The credential exchange record

`aries_cloudagent.protocols.issue_credential.v1_0.routes.register` (*app:* `<sphinx.ext.autodoc.importer._MockC`  
`object` `at`  
`0x7f79512d48d0>`)

Register routes.

### Submodules

`aries_cloudagent.protocols.issue_credential.message_types` module

`aries_cloudagent.protocols.issue_credential.routes` module

### 3.1.8 aries\_cloudagent.protocols.present\_proof package

#### Subpackages

`aries_cloudagent.protocols.present_proof.v1_0` package

#### Subpackages

`aries_cloudagent.protocols.present_proof.v1_0.handlers` package

#### Submodules

`aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_ack_handler` module

Presentation ack message handler.

**class** `aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_ack_handler.PresentationAckHandler`  
Bases: `aries_cloudagent.messaging.base_handler.BaseHandler`

Message handler class for presentation acks.

**handle** (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:* `aries_cloudagent.messaging.responder.BaseResponder`)  
Message handler logic for presentation acks.

#### Parameters

- **context** – request context
- **responder** – responder callback

**aries\_cloudagent.protocols.present\_proof.v1\_0.handlers.presentation\_handler module**

Presentation message handler.

**class** aries\_cloudagent.protocols.present\_proof.v1\_0.handlers.presentation\_handler.**PresentationHandler**  
 Bases: *aries\_cloudagent.messaging.base\_handler.BaseHandler*

Message handler class for presentations.

**handle** (*context:* aries\_cloudagent.messaging.request\_context.RequestContext, *responder:* aries\_cloudagent.messaging.responder.BaseResponder)  
 Message handler logic for presentations.

**Parameters**

- **context** – request context
- **responder** – responder callback

**aries\_cloudagent.protocols.present\_proof.v1\_0.handlers.presentation\_proposal\_handler module**

Presentation proposal message handler.

**class** aries\_cloudagent.protocols.present\_proof.v1\_0.handlers.presentation\_proposal\_handler.**PresentationProposalHandler**  
 Bases: *aries\_cloudagent.messaging.base\_handler.BaseHandler*

Message handler class for presentation proposals.

**handle** (*context:* aries\_cloudagent.messaging.request\_context.RequestContext, *responder:* aries\_cloudagent.messaging.responder.BaseResponder)  
 Message handler logic for presentation proposals.

**Parameters**

- **context** – proposal context
- **responder** – responder callback

**aries\_cloudagent.protocols.present\_proof.v1\_0.handlers.presentation\_request\_handler module**

Presentation request message handler.

**class** aries\_cloudagent.protocols.present\_proof.v1\_0.handlers.presentation\_request\_handler.**PresentationRequestHandler**  
 Bases: *aries\_cloudagent.messaging.base\_handler.BaseHandler*

Message handler class for Aries#0037 v1.0 presentation requests.

**handle** (*context:* aries\_cloudagent.messaging.request\_context.RequestContext, *responder:* aries\_cloudagent.messaging.responder.BaseResponder)  
 Message handler logic for Aries#0037 v1.0 presentation requests.

**Parameters**

- **context** – request context
- **responder** – responder callback

`aries_cloudagent.protocols.present_proof.v1_0.messages` package

### Subpackages

`aries_cloudagent.protocols.present_proof.v1_0.messages.inner` package

### Submodules

`aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_preview` module

A presentation preview inner object.

**class** `aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_preview.PresentationPreview`

Bases: `aries_cloudagent.messaging.models.base.BaseModel`

Class representing an attribute specification within a presentation preview.

**class** `Meta`

Bases: `object`

Attr spec metadata.

**schema\_class** = `'PresAttrSpecSchema'`

**class** `Posture`

Bases: `enum.Enum`

Attribute posture: self-attested, revealed claim or unrevealed claim.

**REVEALED\_CLAIM** = 1

**SELF\_ATTESTED** = 0

**UNREVEALED\_CLAIM** = 2

**b64\_decoded\_value**() → str

Value, base64-decoded if applicable.



**static list\_plain** (*plain: dict, cred\_def\_id: str, referent: str = None*)

Return a list of *PresAttrSpec* on input cred def id.

**Parameters**

- **plain** – dict mapping names to values
- **cred\_def\_id** – credential definition identifier to specify
- **referent** – single referent to use, omit for none

**Returns** List of *PresAttrSpec* on input cred def id with no MIME types

**posture**

self-attested, revealed claim, or unrevealed claim.

**Type** Attribute posture

**satisfies** (*pred\_spec: aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.presentation\_preview.PresPredSpec*)

Whether current specified attribute satisfied input specified predicate.

**class** *aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.presentation\_preview.PresAttrSpec*

Bases: *aries\_cloudagent.messaging.models.base.BaseModelSchema*

Attribute specification schema.

**class Meta**

Bases: *object*

Attribute specification schema metadata.

**model\_class**

alias of *PresAttrSpec*

**cred\_def\_id** = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<

**mime\_type** = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None

**name** = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, re

**referent** = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None

**value** = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, r

**class** *aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.presentation\_preview.PresPredSpec*

Bases: *aries\_cloudagent.messaging.models.base.BaseModel*

Class representing a predicate specification within a presentation preview.

**class Meta**

Bases: *object*

Pred spec metadata.

```
schema_class = 'PresPredSpecSchema'
```

```
class aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_preview.PresPredSpec
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModelSchema*

Predicate specification schema.

```
class Meta
```

Bases: *object*

Predicate specification schema metadata.

```
model_class
```

alias of *PresPredSpec*

```
cred_def_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<
```

```
name = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, re
```

```
predicate = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<ar
```

```
threshold = <fields.Integer(default=<marshmallow.missing>, attribute=None, validate=No
```

```
class aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_preview.PresPredSpec
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModel*

Class representing presentation preview.

```
class Meta
```

Bases: *object*

Presentation preview metadata.

```
message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/present-proof/1.0/presentation-
```

```
schema_class = 'PresentationPreviewSchema'
```

```
has_attr_spec(cred_def_id: str, name: str, value: str) → bool
```

Return whether preview contains given attribute specification.

**Parameters**

- **cred\_def\_id** – credential definition identifier

- **name** – attribute name
- **value** – attribute value

**Returns** Whether preview contains matching attribute specification.

```
indy_proof_request (name: str = None, version: str = None, nonce: str
                    = None, ledger: aries_cloudagent.ledger.indy.IndyLedger
                    = None, non_revoc_intervals: Mapping[str,
                    aries_cloudagent.revocation.models.indy.NonRevocationInterval] = None)
                    → dict
```

Return indy proof request corresponding to presentation preview.

Typically the verifier turns the proof preview into a proof request.

#### Parameters

- **name** – for proof request
- **version** – version for proof request
- **nonce** – nonce for proof request
- **ledger** – ledger with credential definitions, to check for revocation support
- **non\_revoc\_intervals** – non-revocation interval to use per cred def id where applicable (default from and to the current time if applicable)

**Returns** Indy proof request dict.

```
class aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_preview.Pr
```

Bases: *aries\_cloudagent.messaging.models.base.BaseModelSchema*

Presentation preview schema.

```
class Meta
```

Bases: *object*

Presentation preview schema metadata.

```
model_class
```

alias of *PresentationPreview*

```
attributes = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=No
```

```
predicates = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=No
```

## Submodules

**aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation module**

A (proof) presentation content message.

```

class aries_cloudagent.protocols.present_proof.v1_0.messages.presentation.Presentation(_id:
    str
    =
    None
    *,
    com-
    ment
    str
    =
    None
    pre-
    sen-
    ta-
    tions
    Se-
    quen
    =
    None
    **kw

```

Bases: *aries\_cloudagent.messaging.agent\_message.AgentMessage*

Class representing a (proof) presentation.

**class Meta**

Bases: *object*

Presentation metadata.

**handler\_class** = 'aries\_cloudagent.protocols.present\_proof.v1\_0.handlers.presentation'

**message\_type** = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/present-proof/1.0/presentation'

**schema\_class** = 'PresentationSchema'

**indy\_proof** (index: int = 0)

Retrieve and decode indy proof from attachment.

**Parameters index** – ordinal in attachment list to decode and return (typically, list has length 1)

```

class aries_cloudagent.protocols.present_proof.v1_0.messages.presentation.PresentationSchema

```

Bases: *aries\_cloudagent.messaging.agent\_message.AgentMessageSchema*

(Proof) presentation schema.

**class Meta**

Bases: *object*

Presentation schema metadata.

**model\_class**

alias of *Presentation*

**comment** = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,

**presentations\_attach** = <fields.Nested(default=<marshmallow.missing>, attribute=None, v

## aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation\_ack module

Represents an explicit RFC 15 ack message, adopted into present-proof protocol.

**class** `aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_ack.PresentationAck`

Bases: `aries_cloudagent.messaging.ack.message.Ack`

Base class representing an explicit ack message for present-proof protocol.

**class** `Meta`

Bases: `object`

PresentationAck metadata.

**handler\_class** = `'aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_ack.Handler'`

**message\_type** = `'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/present-proof/1.0/ack'`

**schema\_class** = `'PresentationAckSchema'`

**class** `aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_ack.PresentationAckSchema`

Bases: `aries_cloudagent.messaging.ack.message.AckSchema`

Schema for PresentationAck class.

**class** `Meta`

Bases: `object`

PresentationAck schema metadata.

**model\_class**

alias of `PresentationAck`

### `aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_proposal` module

A presentation proposal content message.

**class** `aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_proposal.PresentationProposal`

Bases: `aries_cloudagent.messaging.agent_message.AgentMessage`

Class representing a presentation proposal.

```

class Meta
    Bases: object
    PresentationProposal metadata.
    handler_class = 'aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_proposal'
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/present-proof/1.0/propose-presentation'
    schema_class = 'PresentationProposalSchema'

class aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_proposal.PresentationProposal
    Bases: aries_cloudagent.messaging.agent_message.AgentMessageSchema
    Presentation proposal schema.

class Meta
    Bases: object
    Presentation proposal schema metadata.
    model_class
        alias of PresentationProposal
    comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, if_exists=True, if_missing=None)>
    presentation_proposal = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None, if_exists=True, if_missing=None)>

```

### aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation\_request module

A presentation request content message.

```

class aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_request.PresentationRequest

```

Bases: *aries\_cloudagent.messaging.agent\_message.AgentMessage*

Class representing a presentation request.

```

class Meta
    Bases: object
    PresentationRequest metadata.
    handler_class = 'aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_request'
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/present-proof/1.0/request-presentation'

```

```
    schema_class = 'PresentationRequestSchema'

    indy_proof_request (index: int = 0)
        Retrieve and decode indy proof request from attachment.

        Parameters index – ordinal in attachment list to decode and return (typically, list has length
            1)

class aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_request.PresentationRequestSchema:
    Bases: aries_cloudagent.messaging.agent_message.AgentMessageSchema
    Presentation request schema.

    class Meta:
        Bases: object
        Presentation request schema metadata.

    model_class
        alias of PresentationRequest

    comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, if_not_possible=None)>
    request_presentations_attach = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None, if_not_possible=None)>
```

**aries\_cloudagent.protocols.present\_proof.v1\_0.models package**

### Submodules

**aries\_cloudagent.protocols.present\_proof.v1\_0.models.presentation\_exchange module**

Aries#0037 v1.0 presentation exchange information with non-secrets storage.

```
class aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange.V10Present
```



Bases: *aries\_cloudagent.messaging.models.base\_record.BaseExchangeRecord*

Represents an Aries#0037 v1.0 presentation exchange.

```
INITIATOR_EXTERNAL = 'external'
INITIATOR_SELF = 'self'

class Meta
    Bases: object
    V10PresentationExchange metadata.
    schema_class = 'V10PresentationExchangeSchema'
RECORD_ID_NAME = 'presentation_exchange_id'
RECORD_TYPE = 'presentation_exchange_v10'
ROLE_PROVER = 'prover'
ROLE_VERIFIER = 'verifier'
STATE_PRESENTATION_ACKED = 'presentation_acked'
STATE_PRESENTATION_RECEIVED = 'presentation_received'
STATE_PRESENTATION_SENT = 'presentation_sent'
STATE_PROPOSAL_RECEIVED = 'proposal_received'
STATE_PROPOSAL_SENT = 'proposal_sent'
STATE_REQUEST_RECEIVED = 'request_received'
STATE_REQUEST_SENT = 'request_sent'
STATE_VERIFIED = 'verified'
TAG_NAMES = {'thread_id'}
WEBHOOK_TOPIC = 'present_proof'

presentation_exchange_id
    Accessor for the ID associated with this exchange.

record_value
    Accessor for JSON record value generated for this presentation exchange.
```

```
class aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange.V10Present
```

Bases: *aries\_cloudagent.messaging.models.base\_record.BaseExchangeSchema*

Schema for de/serialization of v1.0 presentation exchange records.

```
class Meta
    Bases: object
    V10PresentationExchangeSchema metadata.
    model_class
        alias of V10PresentationExchange

auto_present = <fields.Boolean(default=<marshmallow.missing>, attribute=None, validate=
connection_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=
error_msg = <fields.String(default=<marshmallow.missing>, attribute=None, validate=Non
initiator = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<On
```

```

presentation = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
presentation_exchange_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
presentation_proposal_dict = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
presentation_request = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
role = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<OneOf(choices=('holder', 'issuer', 'verifier'), error_message='Invalid role'))>
state = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, if_present=True)
thread_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, if_present=True)
verified = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<OneOf(choices=('true', 'false'), error_message='Invalid verified status'))>

```

## aries\_cloudagent.protocols.present\_proof.v1\_0.util package

### Submodules

#### aries\_cloudagent.protocols.present\_proof.v1\_0.util.indy module

Utilities for dealing with indy conventions.

```
aries_cloudagent.protocols.present_proof.v1_0.util.indy.indy_proof_req_preview2indy_request
```

Build indy requested-credentials structure.

Given input proof request and presentation preview, use credentials in holder's wallet to build indy requested credentials structure for input to proof creation.

#### Parameters

- **indy\_proof\_request** – indy proof request
- **pres\_preview** – preview from presentation proposal, if applicable
- **holder** – holder injected into current context

### Submodules

#### aries\_cloudagent.protocols.present\_proof.v1\_0.manager module

Classes to manage presentations.

```
class aries_cloudagent.protocols.present_proof.v1_0.manager.PresentationManager(context: Context, wallet: Wallet,
                                                                              aries_cloudagent.protocols.present_proof.v1_0.util.indy: indy_util)
    Bases: object
```

Class for managing presentations.

**context**

Accessor for the current request context.

**Returns** The injection context for this presentation manager

**create\_bound\_request** (*presentation\_exchange\_record: aries\_cloudagent.protocols.present\_proof.v1\_0.models.presentation\_exchange\_record, name: str = None, version: str = None, nonce: str = None, comment: str = None*)

Create a presentation request bound to a proposal.

**Parameters**

- **presentation\_exchange\_record** – Presentation exchange record for which to create presentation request
- **name** – name to use in presentation request (None for default)
- **version** – version to use in presentation request (None for default)
- **nonce** – nonce to use in presentation request (None to generate)
- **comment** – Optional human-readable comment pertaining to request creation

**Returns** A tuple (updated presentation exchange record, presentation request message)

**create\_exchange\_for\_proposal** (*connection\_id: str, presentation\_proposal\_message: aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation\_proposal.PresentationProposal, auto\_present: bool = None*)

Create a presentation exchange record for input presentation proposal.

**Parameters**

- **connection\_id** – connection identifier
- **presentation\_proposal\_message** – presentation proposal to serialize to exchange record
- **auto\_present** – whether to present proof upon receiving proof request (default to configuration setting)

**Returns** Presentation exchange record, created

**create\_exchange\_for\_request** (*connection\_id: str, presentation\_request\_message: aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation\_request.PresentationRequest*)

Create a presentation exchange record for input presentation request.

**Parameters**

- **connection\_id** – connection identifier
- **presentation\_request\_message** – presentation request to use in creating exchange record, extracting indy proof request and thread id

**Returns** Presentation exchange record, updated

**create\_presentation** (*presentation\_exchange\_record: aries\_cloudagent.protocols.present\_proof.v1\_0.models.presentation\_exchange\_record, requested\_credentials: dict, comment: str = None*)

Create a presentation.

**Parameters**

- **presentation\_exchange\_record** – Record to update
- **requested\_credentials** – Indy formatted requested\_credentials
- **comment** – optional human-readable comment

Example *requested\_credentials* format:

```
{
  "self_attested_attributes": {
    "j233ffbc-bd35-49b1-934f-51e083106f6d": "value"
  },
  "requested_attributes": {
    "6253ffbb-bd35-49b3-934f-46e083106f6c": {
      "cred_id": "5bfa40b7-062b-4ae0-a251-a86c87922c0e",
      "revealed": true
    }
  },
  "requested_predicates": {
    "bfc8a97d-60d3-4f21-b998-85eeabe5c8c0": {
      "cred_id": "5bfa40b7-062b-4ae0-a251-a86c87922c0e"
    }
  }
}
```

**Returns** A tuple (updated presentation exchange record, presentation message)

**receive\_presentation()**

Receive a presentation, from message in context on manager creation.

**Returns** presentation exchange record, retrieved and updated

**receive\_presentation\_ack()**

Receive a presentation ack, from message in context on manager creation.

**Returns** presentation exchange record, retrieved and updated

**receive\_proposal()**

Receive a presentation proposal from message in context on manager creation.

**Returns** Presentation exchange record, created

**receive\_request** (*presentation\_exchange\_record: aries\_cloudagent.protocols.present\_proof.v1\_0.models.presentation\_exch*)

Receive a presentation request.

**Parameters** **presentation\_exchange\_record** – presentation exchange record with request to receive

**Returns** The presentation\_exchange\_record, updated

**send\_presentation\_ack** (*presentation\_exchange\_record: aries\_cloudagent.protocols.present\_proof.v1\_0.models.presentation\_exch*)

Send acknowledgement of presentation receipt.

**Parameters** **presentation\_exchange\_record** – presentation exchange record with thread id

**verify\_presentation** (*presentation\_exchange\_record: aries\_cloudagent.protocols.present\_proof.v1\_0.models.presentation\_exch*)

Verify a presentation.

**Parameters** **presentation\_exchange\_record** – presentation exchange record with presentation request and presentation to verify

**Returns** presentation record, updated

```
exception aries_cloudagent.protocols.present_proof.v1_0.manager.PresentationManagerError (*a
er
ro
st
=
Ne
**
```

Bases: `aries_cloudagent.core.error.BaseError`

Presentation error.

## aries\_cloudagent.protocols.present\_proof.v1\_0.message\_types module

Message and inner object type identifiers for Connections.

## aries\_cloudagent.protocols.present\_proof.v1\_0.routes module

Admin routes for presentations.

```
class aries_cloudagent.protocols.present_proof.v1_0.routes.IndyProofReqAttrSpecSchema (*,
only=I
ex-
clude=
many=
con-
text=N
load_o
dump_
par-
tial=F
un-
known
```

Bases: `marshmallow.schema.Schema`

Schema for attribute specification in indy proof request.

`opts = <marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.present_proof.v1_0.routes.IndyProofReqNonRevoked (*,
only=None,
ex-
clude=(),
many=False
con-
text=None,
load_only=(
dump_only=
par-
tial=False,
un-
known=Non
```

Bases: `marshmallow.schema.Schema`

Non-revocation times specification in indy proof request.

`opts = <marshmallow.schema.SchemaOpts object>`

```

class aries_cloudagent.protocols.present_proof.v1_0.routes.IndyProofReqPredSpecSchema(*,
    only=None,
    exclude=(),
    include=(),
    many=False,
    context=None,
    load_only=(),
    dump_only=(),
    partial=False,
    unknown=None)

    Bases: marshmallow.schema.Schema
    Schema for predicate specification in indy proof request.
    opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.protocols.present_proof.v1_0.routes.IndyProofReqSpecRestrictionsSchema(*,
    only=None,
    exclude=(),
    include=(),
    many=False,
    context=None,
    load_only=(),
    dump_only=(),
    partial=False,
    unknown=None)

    Bases: marshmallow.schema.Schema
    Schema for restrictions in attr or pred specifier indy proof request.
    opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.protocols.present_proof.v1_0.routes.IndyProofRequestSchema(*,
    only=None,
    exclude=(),
    include=(),
    many=False,
    context=None,
    load_only=(),
    dump_only=(),
    partial=False,
    unknown=None)

    Bases: marshmallow.schema.Schema
    Schema for indy proof request.
    opts = <marshmallow.schema.SchemaOpts object>

```

```
class aries_cloudagent.protocols.present_proof.v1_0.routes.IndyRequestedCredsRequestedAttr:
```

Bases: `marshmallow.schema.Schema`

Schema for requested attributes within indy requested credentials structure.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.present_proof.v1_0.routes.IndyRequestedCredsRequestedPred:
```

Bases: `marshmallow.schema.Schema`

Schema for requested predicates within indy requested credentials structure.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.present_proof.v1_0.routes.V10PresentationExchangeListScher
```

Bases: `marshmallow.schema.Schema`

Result schema for an Aries#0037 v1.0 presentation exchange query.

**opts** = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.present_proof.v1_0.routes.V10PresentationProposalRequestS
```

Bases: aries\_cloudagent.utils.tracing.AdminAPIMessageTracingSchema

Request schema for sending a presentation proposal admin message.

**opts** = <marshmallow.schema.SchemaOpts object>

```
class aries_cloudagent.protocols.present_proof.v1_0.routes.V10PresentationRequestRequestSch
```

Bases: aries\_cloudagent.utils.tracing.AdminAPIMessageTracingSchema

Request schema for sending a proof request.

**opts** = <marshmallow.schema.SchemaOpts object>

```
class aries_cloudagent.protocols.present_proof.v1_0.routes.V10PresentationRequestSchema (*,
```

Bases: aries\_cloudagent.utils.tracing.AdminAPIMessageTracingSchema

Request schema for sending a presentation.

**opts** = <marshmallow.schema.SchemaOpts object>



```
aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_create_request(request: aiohttp.Request) -> Dict[str, Any]
```

Request handler for creating a free presentation request.

The presentation request will not be bound to any proposal or existing connection.

**Parameters** `request` – aiohttp request object

**Returns** The presentation exchange details

```
aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_credentials_list(request: aiohttp.Request) -> Dict[str, Any]
```

Request handler for searching applicable credential records.

**Parameters** `request` – aiohttp request object

**Returns** The credential list response

```
aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_list(request: aiohttp.Request) -> Dict[str, Any]
```

Request handler for searching presentation exchange records.

**Parameters** `request` – aiohttp request object

**Returns** The presentation exchange list response

```
aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_remove(request: aiohttp.Request) -> Dict[str, Any]
```

Request handler for removing a presentation exchange record.

**Parameters** `request` – aiohttp request object

```
aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_retrieve(request: aiohttp.Request) -> Dict[str, Any]
```

Request handler for fetching a single presentation exchange record.

**Parameters** `request` – aiohttp request object

**Returns** The presentation exchange record response

`aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_send_bound_request`

Request handler for sending a presentation request free from any proposal.

**Parameters** `request` – aiohttp request object

**Returns** The presentation exchange details

`aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_send_free_request`

Request handler for sending a presentation request free from any proposal.

**Parameters** `request` – aiohttp request object

**Returns** The presentation exchange details

`aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_send_presentation`

Request handler for sending a presentation.

**Parameters** `request` – aiohttp request object

**Returns** The presentation exchange details

`aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_send_proposal` (re

Request handler for sending a presentation proposal.

**Parameters** `request` – aiohttp request object

**Returns** The presentation exchange details

`aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_verify_presentation`

Request handler for verifying a presentation request.

**Parameters** `request` – aiohttp request object

**Returns** The presentation exchange details

```
aries_cloudagent.protocols.present_proof.v1_0.routes.register(app:
                                                                <sphinx.ext.autodoc.importer._MockObject
                                                                object          at
                                                                0x7f79508e04e0>)
```

Register routes.

## Submodules

`aries_cloudagent.protocols.present_proof.message_types` module

`aries_cloudagent.protocols.present_proof.routes` module

## 3.1.9 aries\_cloudagent.protocols.presentations package

### Subpackages

`aries_cloudagent.protocols.presentations.handlers` package

### Submodules

`aries_cloudagent.protocols.presentations.handlers.credential_presentation_handler` module

`aries_cloudagent.protocols.presentations.handlers.presentation_request_handler` module

`aries_cloudagent.protocols.presentations.messages` package

### Submodules

`aries_cloudagent.protocols.presentations.messages.credential_presentation` module

`aries_cloudagent.protocols.presentations.messages.presentation_request` module

`aries_cloudagent.protocols.presentations.models` package

### Submodules

`aries_cloudagent.protocols.presentations.models.presentation_exchange` module

### Submodules

`aries_cloudagent.protocols.presentations.manager` module

`aries_cloudagent.protocols.presentations.message_types` module

`aries_cloudagent.protocols.presentations.routes` module

## 3.1.10 aries\_cloudagent.protocols.problem\_report package

## Submodules

`aries_cloudagent.protocols.problem_report.handler` module

`aries_cloudagent.protocols.problem_report.message` module

`aries_cloudagent.protocols.problem_report.message_types` module

### 3.1.11 `aries_cloudagent.protocols.routing` package

#### Subpackages

`aries_cloudagent.protocols.routing.handlers` package

#### Submodules

`aries_cloudagent.protocols.routing.handlers.forward_handler` module

`aries_cloudagent.protocols.routing.handlers.route_query_request_handler` module

`aries_cloudagent.protocols.routing.handlers.route_query_response_handler` module

`aries_cloudagent.protocols.routing.handlers.route_update_request_handler` module

`aries_cloudagent.protocols.routing.handlers.route_update_response_handler` module

`aries_cloudagent.protocols.routing.messages` package

#### Submodules

`aries_cloudagent.protocols.routing.messages.forward` module

`aries_cloudagent.protocols.routing.messages.route_query_request` module

`aries_cloudagent.protocols.routing.messages.route_query_response` module

`aries_cloudagent.protocols.routing.messages.route_update_request` module

`aries_cloudagent.protocols.routing.messages.route_update_response` module

`aries_cloudagent.protocols.routing.models` package

#### Submodules

`aries_cloudagent.protocols.routing.models.paginate` module

`aries_cloudagent.protocols.routing.models.paginated` module

`aries_cloudagent.protocols.routing.models.route_query_result` module

`aries_cloudagent.protocols.routing.models.route_record` module

`aries_cloudagent.protocols.routing.models.route_update` module

`aries_cloudagent.protocols.routing.models.route_updated` module

#### Submodules

`aries_cloudagent.protocols.routing.manager` module

`aries_cloudagent.protocols.routing.message_types` module

### 3.1.12 `aries_cloudagent.protocols.trustping` package

#### Subpackages

`aries_cloudagent.protocols.trustping.handlers` package

#### Submodules

`aries_cloudagent.protocols.trustping.handlers.ping_handler` module

`aries_cloudagent.protocols.trustping.handlers.ping_response_handler` module

`aries_cloudagent.protocols.trustping.messages` package

#### Submodules

`aries_cloudagent.protocols.trustping.messages.ping` module

`aries_cloudagent.protocols.trustping.messages.ping_response` module

#### Submodules

`aries_cloudagent.protocols.trustping.message_types` module

`aries_cloudagent.protocols.trustping.routes` module



## CHAPTER 4

---

### Indices and tables

---

- `genindex`





---

## Python Module Index

---

### a

aries\_cloudagent, 3  
aries\_cloudagent.admin, 3  
aries\_cloudagent.admin.base\_server, 3  
aries\_cloudagent.admin.error, 4  
aries\_cloudagent.admin.server, 4  
aries\_cloudagent.cache, 6  
aries\_cloudagent.cache.base, 6  
aries\_cloudagent.cache.basic, 7  
aries\_cloudagent.commands, 7  
aries\_cloudagent.commands.help, 8  
aries\_cloudagent.commands.provision, 8  
aries\_cloudagent.commands.start, 8  
aries\_cloudagent.config, 9  
aries\_cloudagent.config.argparse, 9  
aries\_cloudagent.config.base, 11  
aries\_cloudagent.config.base\_context, 12  
aries\_cloudagent.config.default\_context, 13  
aries\_cloudagent.config.error, 13  
aries\_cloudagent.config.injection\_context, 13  
aries\_cloudagent.config.injector, 15  
aries\_cloudagent.config.ledger, 15  
aries\_cloudagent.config.logging, 15  
aries\_cloudagent.config.provider, 16  
aries\_cloudagent.config.settings, 17  
aries\_cloudagent.config.util, 18  
aries\_cloudagent.config.wallet, 18  
aries\_cloudagent.connections, 127  
aries\_cloudagent.connections.models, 127  
aries\_cloudagent.connections.models.connection\_record, 136  
aries\_cloudagent.connections.models.connection\_target, 141  
aries\_cloudagent.connections.models.diddoc, 127  
aries\_cloudagent.connections.models.diddoc.didoc, 40  
aries\_cloudagent.connections.models.diddoc.publickey, 132  
aries\_cloudagent.connections.models.diddoc.service, 134  
aries\_cloudagent.connections.models.diddoc.util, 135  
aries\_cloudagent.core, 18  
aries\_cloudagent.core.conductor, 18  
aries\_cloudagent.core.dispatcher, 20  
aries\_cloudagent.core.error, 21  
aries\_cloudagent.core.plugin\_registry, 22  
aries\_cloudagent.core.protocol\_registry, 23  
aries\_cloudagent.holder, 24  
aries\_cloudagent.holder.base, 24  
aries\_cloudagent.holder.indy, 25  
aries\_cloudagent.issuer, 27  
aries\_cloudagent.issuer.base, 27  
aries\_cloudagent.issuer.indy, 29  
aries\_cloudagent.ledger, 31  
aries\_cloudagent.ledger.base, 32  
aries\_cloudagent.ledger.error, 33  
aries\_cloudagent.ledger.indy, 34  
aries\_cloudagent.ledger.provider, 37  
aries\_cloudagent.ledger.routes, 37  
aries\_cloudagent.ledger.util, 39  
aries\_cloudagent.messaging, 39  
aries\_cloudagent.messaging.ack, 39  
aries\_cloudagent.messaging.ack.message, 39  
aries\_cloudagent.messaging.agent\_message, 66  
aries\_cloudagent.messaging.base\_handler, 69  
aries\_cloudagent.messaging.credential\_definitions, 40  
aries\_cloudagent.messaging.credential\_definitions.1

aries_cloudagent.messaging.credential_definitions, 42	aries_cloudagent.protocols.issue_credential.v1_0.handler, 148
aries_cloudagent.messaging.decorators, 42	aries_cloudagent.protocols.issue_credential.v1_0.handler, 148
aries_cloudagent.messaging.decorators.attach_decorator, 42	aries_cloudagent.protocols.issue_credential.v1_0.handler, 148
aries_cloudagent.messaging.decorators.base, 48	aries_cloudagent.protocols.issue_credential.v1_0.handler, 148
aries_cloudagent.messaging.decorators.definition, 49	aries_cloudagent.protocols.issue_credential.v1_0.handler, 149
aries_cloudagent.messaging.decorators.localization_decorator, 49	aries_cloudagent.protocols.issue_credential.v1_0.handler, 149
aries_cloudagent.messaging.decorators.plan, 50	aries_cloudagent.protocols.issue_credential.v1_0.handler, 160
aries_cloudagent.messaging.decorators.signature_decorator, 51	aries_cloudagent.protocols.issue_credential.v1_0.handler, 163
aries_cloudagent.messaging.decorators.thread_decorator, 53	aries_cloudagent.protocols.issue_credential.v1_0.handler, 149
aries_cloudagent.messaging.decorators.timing_decorator, 54	aries_cloudagent.protocols.issue_credential.v1_0.handler, 151
aries_cloudagent.messaging.decorators.transport_decorator, 56	aries_cloudagent.protocols.issue_credential.v1_0.handler, 152
aries_cloudagent.messaging.error, 69	aries_cloudagent.protocols.issue_credential.v1_0.handler, 153
aries_cloudagent.messaging.models, 57	aries_cloudagent.protocols.issue_credential.v1_0.handler, 154
aries_cloudagent.messaging.models.base, 59	aries_cloudagent.protocols.issue_credential.v1_0.handler, 156
aries_cloudagent.messaging.models.base_record, 59	aries_cloudagent.protocols.issue_credential.v1_0.handler, 149
aries_cloudagent.messaging.request_context, 69	aries_cloudagent.protocols.issue_credential.v1_0.handler, 149
aries_cloudagent.messaging.responder, 70	aries_cloudagent.protocols.issue_credential.v1_0.handler, 149
aries_cloudagent.messaging.schemas, 63	aries_cloudagent.protocols.issue_credential.v1_0.handler, 157
aries_cloudagent.messaging.schemas.routes, 63	aries_cloudagent.protocols.issue_credential.v1_0.handler, 157
aries_cloudagent.messaging.schemas.util, 65	aries_cloudagent.protocols.issue_credential.v1_0.handler, 163
aries_cloudagent.messaging.util, 72	aries_cloudagent.protocols.issue_credential.v1_0.handler, 170
aries_cloudagent.messaging.valid, 73	aries_cloudagent.protocols.issue_credential.v1_0.handler, 170
aries_cloudagent.protocols, 143	aries_cloudagent.protocols.issue_credential.v1_0.handler, 170
aries_cloudagent.protocols.actionmenu, 143	aries_cloudagent.protocols.issue_credential.v1_0.handler, 170
aries_cloudagent.protocols.basicmessage, 144	aries_cloudagent.protocols.issue_credential.v1_0.handler, 170
aries_cloudagent.protocols.connections, 144	aries_cloudagent.protocols.issue_credential.v1_0.handler, 170
aries_cloudagent.protocols.discovery, 146	aries_cloudagent.protocols.issue_credential.v1_0.handler, 171
aries_cloudagent.protocols.introduction, 147	aries_cloudagent.protocols.issue_credential.v1_0.handler, 171
aries_cloudagent.protocols.issue_credential, 147	aries_cloudagent.protocols.issue_credential.v1_0.handler, 171
aries_cloudagent.protocols.issue_credential, 148	aries_cloudagent.protocols.issue_credential.v1_0.handler, 182

[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.messages, 185](#)  
[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.messages, 172](#)  
[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.messages, 172](#)  
[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.messages, 172](#)  
[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.messages, 172](#)  
[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.messages, 175](#)  
[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.messages, 176](#)  
[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.messages, 177](#)  
[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.messages, 178](#)  
[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.messages, 179](#)  
[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.messages, 179](#)  
[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.messages, 185](#)  
[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.messages, 182](#)  
[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.messages, 182](#)  
[aries\\_cloudagent.protocols.problem\\_report, 191](#)  
[aries\\_cloudagent.protocols.routing, 192](#)  
[aries\\_cloudagent.protocols.trustping, 193](#)  
[aries\\_cloudagent.storage, 75](#)  
[aries\\_cloudagent.storage.base, 75](#)  
[aries\\_cloudagent.storage.basic, 77](#)  
[aries\\_cloudagent.storage.error, 79](#)  
[aries\\_cloudagent.storage.indy, 80](#)  
[aries\\_cloudagent.storage.provider, 82](#)  
[aries\\_cloudagent.storage.record, 82](#)  
[aries\\_cloudagent.transport, 83](#)  
[aries\\_cloudagent.transport.error, 98](#)  
[aries\\_cloudagent.transport.inbound, 83](#)  
[aries\\_cloudagent.transport.inbound.base, 83](#)  
[aries\\_cloudagent.transport.inbound.deliveries, 84](#)  
[aries\\_cloudagent.transport.inbound.http, 85](#)  
[aries\\_cloudagent.transport.inbound.manager, 86](#)  
[aries\\_cloudagent.transport.inbound.message, 87](#)  
[aries\\_cloudagent.transport.inbound.receipt, 88](#)  
[aries\\_cloudagent.transport.inbound.session,](#)



## A

accept (*aries\_cloudagent.connections.models.connection\_record.ConnectionRecordSchema*  
     *attribute*), 140  
 ACCEPT\_AUTO (*aries\_cloudagent.connections.models.connection\_record.ConnectionRecord*  
     *attribute*), 138  
 ACCEPT\_MANUAL (*aries\_cloudagent.connections.models.connection\_record.ConnectionRecord*  
     *attribute*), 138  
 accept\_response ()  
     (*aries\_cloudagent.transport.inbound.session.InboundSession*  
     *method*), 90  
 accept\_taa () (in *module*  
     *aries\_cloudagent.config.ledger*), 15  
 accept\_txn\_author\_agreement ()  
     (*aries\_cloudagent.ledger.base.BaseLedger*  
     *method*), 32  
 accept\_txn\_author\_agreement ()  
     (*aries\_cloudagent.ledger.indy.IndyLedger*  
     *method*), 34  
 AcceptResult (class in *aries\_cloudagent.transport.inbound.session*),  
     89  
 Ack (class in *aries\_cloudagent.messaging.ack.message*),  
     39  
 Ack.Meta (class in *aries\_cloudagent.messaging.ack.message*),  
     39  
 AckSchema (class in *aries\_cloudagent.messaging.ack.message*),  
     39  
 AckSchema.Meta (class in *aries\_cloudagent.messaging.ack.message*),  
     39  
 acquire () (*aries\_cloudagent.cache.base.BaseCache*  
     *method*), 6  
 add\_active () (*aries\_cloudagent.utils.task\_queue.TaskQueue*  
     *method*), 106  
 add\_arguments () (*aries\_cloudagent.config.argparse.AdminGroup*  
     *method*), 9  
 add\_arguments () (*aries\_cloudagent.config.argparse.ArgumentGroup*  
     *method*), 9  
     *add\_arguments () (aries\_cloudagent.config.argparse.DebugGroup*  
         *method*), 9  
     *add\_arguments () (aries\_cloudagent.config.argparse.GeneralGroup*  
         *method*), 9  
     *add\_arguments () (aries\_cloudagent.config.argparse.LedgerGroup*  
         *method*), 10  
     *add\_arguments () (aries\_cloudagent.config.argparse.LoggingGroup*  
         *method*), 10  
     *add\_arguments () (aries\_cloudagent.config.argparse.ProtocolGroup*  
         *method*), 10  
     *add\_arguments () (aries\_cloudagent.config.argparse.TransportGroup*  
         *method*), 10  
     *add\_arguments () (aries\_cloudagent.config.argparse.WalletGroup*  
         *method*), 11  
     *add\_message () (aries\_cloudagent.transport.inbound.delivery\_queue.DeliveryQueue*  
         *method*), 84  
     *add\_model () (aries\_cloudagent.messaging.decorators.base.BaseDecorator*  
         *method*), 48  
     *add\_pending () (aries\_cloudagent.utils.task\_queue.TaskQueue*  
         *method*), 106  
     *add\_record () (aries\_cloudagent.storage.base.BaseStorage*  
         *method*), 75  
     *add\_record () (aries\_cloudagent.storage.basic.BasicStorage*  
         *method*), 77  
     *add\_record () (aries\_cloudagent.storage.indy.IndyStorage*  
         *method*), 80  
     *add\_reply\_thread\_ids ()*  
         (*aries\_cloudagent.transport.inbound.session.InboundSession*  
         *method*), 90  
     *add\_reply\_verkeys ()*  
         (*aries\_cloudagent.transport.inbound.session.InboundSession*  
         *method*), 90  
     *add\_service\_pubkeys ()*  
         (*aries\_cloudagent.connections.models.diddoc.DIDDoc*  
         *method*), 127  
     *add\_service\_pubkeys ()*  
         (*aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc*  
         *method*), 131  
     *add\_trace\_decorator ()*  
         (*aries\_cloudagent.messaging.agent\_message.AgentMessage*

`method`), 66  
`add_trace_report()`  
     (`aries_cloudagent.messaging.agent_message.AgentMessage`  
     `method`), 66  
`add_webhook_target()`  
     (`aries_cloudagent.admin.base_server.BaseAdminServer`  
     `method`), 3  
`add_webhook_target()`  
     (`aries_cloudagent.admin.server.AdminServer`  
     `method`), 4  
`AdminError`, 4  
`AdminGroup` (class  
     `aries_cloudagent.config.argparse`), 9  
`AdminModulesSchema` (class  
     `aries_cloudagent.admin.server`), 4  
`AdminResponder` (class  
     `aries_cloudagent.admin.server`), 4  
`AdminServer` (class  
     `aries_cloudagent.admin.server`), 4  
`AdminSetupError`, 4  
`AdminStatusSchema` (class  
     `aries_cloudagent.admin.server`), 5  
`AgentMessage` (class  
     `aries_cloudagent.messaging.agent_message`),  
     66  
`AgentMessage.Meta` (class  
     `aries_cloudagent.messaging.agent_message`),  
     66  
`AgentMessageError`, 67  
`AgentMessageSchema` (class  
     `aries_cloudagent.messaging.agent_message`),  
     68  
`AgentMessageSchema.Meta` (class  
     `aries_cloudagent.messaging.agent_message`),  
     68  
`alias` (`aries_cloudagent.connections.models.connection_record.ConnectionRecordSchema`  
     `attribute`), 140  
`AMLRecordSchema` (class  
     `aries_cloudagent.ledger.routes`), 37  
`ArgsParseError`, 13  
`ArgumentGroup` (class  
     `aries_cloudagent.config.argparse`), 9  
`aries_cloudagent` (module), 3  
`aries_cloudagent.admin` (module), 3  
`aries_cloudagent.admin.base_server` (mod-  
     ule), 3  
`aries_cloudagent.admin.error` (module), 4  
`aries_cloudagent.admin.server` (module), 4  
`aries_cloudagent.cache` (module), 6  
`aries_cloudagent.cache.base` (module), 6  
`aries_cloudagent.cache.basic` (module), 7  
`aries_cloudagent.commands` (module), 7  
`aries_cloudagent.commands.help` (module), 8  
`aries_cloudagent.commands.provision`  
     (module), 8  
`aries_cloudagent.commands.start` (module),  
     8  
`aries_cloudagent.config` (module), 9  
`aries_cloudagent.config.argparse` (mod-  
     ule), 9  
`aries_cloudagent.config.base` (module), 11  
`aries_cloudagent.config.base_context`  
     (module), 12  
`aries_cloudagent.config.default_context`  
     (module), 13  
`aries_cloudagent.config.error` (module), 13  
`aries_cloudagent.config.injection_context`  
     (module), 13  
`aries_cloudagent.config.injector` (mod-  
     ule), 15  
`aries_cloudagent.config.ledger` (module),  
     15  
`aries_cloudagent.config.logging` (module),  
     15  
`aries_cloudagent.config.provider` (mod-  
     ule), 16  
`aries_cloudagent.config.settings` (mod-  
     ule), 17  
`aries_cloudagent.config.util` (module), 18  
`aries_cloudagent.config.wallet` (module),  
     18  
`aries_cloudagent.connections` (module), 127  
`aries_cloudagent.connections.models`  
     (module), 127  
`aries_cloudagent.connections.models.connection_record`  
     (module), 136  
`aries_cloudagent.connections.models.connection_target`  
     (module), 141  
`aries_cloudagent.connections.models.diddoc`  
     (module), 130  
`aries_cloudagent.connections.models.diddoc.diddoc`  
     (module), 130  
`aries_cloudagent.connections.models.diddoc.publickey`  
     (module), 132  
`aries_cloudagent.connections.models.diddoc.service`  
     (module), 134  
`aries_cloudagent.connections.models.diddoc.util`  
     (module), 135  
`aries_cloudagent.core` (module), 18  
`aries_cloudagent.core.conductor` (module),  
     18  
`aries_cloudagent.core.dispatcher` (mod-  
     ule), 20  
`aries_cloudagent.core.error` (module), 21  
`aries_cloudagent.core.plugin_registry`  
     (module), 22  
`aries_cloudagent.core.protocol_registry`  
     (module), 23

<a href="#">aries_cloudagent.holder (module), 24</a>	<a href="#">aries_cloudagent.messaging.models.base (module), 57</a>
<a href="#">aries_cloudagent.holder.base (module), 24</a>	<a href="#">aries_cloudagent.messaging.models.base_record (module), 59</a>
<a href="#">aries_cloudagent.holder.indy (module), 25</a>	<a href="#">aries_cloudagent.messaging.request_context (module), 69</a>
<a href="#">aries_cloudagent.issuer (module), 27</a>	<a href="#">aries_cloudagent.messaging.responder (module), 70</a>
<a href="#">aries_cloudagent.issuer.base (module), 27</a>	<a href="#">aries_cloudagent.messaging.schemas (module), 63</a>
<a href="#">aries_cloudagent.issuer.indy (module), 29</a>	<a href="#">aries_cloudagent.messaging.schemas.routes (module), 63</a>
<a href="#">aries_cloudagent.ledger (module), 31</a>	<a href="#">aries_cloudagent.messaging.schemas.util (module), 65</a>
<a href="#">aries_cloudagent.ledger.base (module), 32</a>	<a href="#">aries_cloudagent.messaging.util (module), 72</a>
<a href="#">aries_cloudagent.ledger.error (module), 33</a>	<a href="#">aries_cloudagent.messaging.valid (module), 73</a>
<a href="#">aries_cloudagent.ledger.indy (module), 34</a>	<a href="#">aries_cloudagent.protocols (module), 143</a>
<a href="#">aries_cloudagent.ledger.provider (module), 37</a>	<a href="#">aries_cloudagent.protocols.actionmenu (module), 143</a>
<a href="#">aries_cloudagent.ledger.routes (module), 37</a>	<a href="#">aries_cloudagent.protocols.basicmessage (module), 144</a>
<a href="#">aries_cloudagent.ledger.util (module), 39</a>	<a href="#">aries_cloudagent.protocols.connections (module), 144</a>
<a href="#">aries_cloudagent.messaging (module), 39</a>	<a href="#">aries_cloudagent.protocols.discovery (module), 146</a>
<a href="#">aries_cloudagent.messaging.ack (module), 39</a>	<a href="#">aries_cloudagent.protocols.introduction (module), 147</a>
<a href="#">aries_cloudagent.messaging.ack.message (module), 39</a>	<a href="#">aries_cloudagent.protocols.issue_credential (module), 147</a>
<a href="#">aries_cloudagent.messaging.agent_message (module), 66</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0 (module), 148</a>
<a href="#">aries_cloudagent.messaging.base_handler (module), 69</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handler (module), 148</a>
<a href="#">aries_cloudagent.messaging.credential_definition (module), 40</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handlers (module), 148</a>
<a href="#">aries_cloudagent.messaging.credential_definition_util (module), 40</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handlers.factory (module), 148</a>
<a href="#">aries_cloudagent.messaging.credential_definition_util.util (module), 42</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handlers.impl (module), 149</a>
<a href="#">aries_cloudagent.messaging.decorators (module), 42</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handlers.impl.factory (module), 149</a>
<a href="#">aries_cloudagent.messaging.decorators.attach_decorator (module), 42</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handlers.impl.impl (module), 149</a>
<a href="#">aries_cloudagent.messaging.decorators.base (module), 48</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handlers.impl.impl.factory (module), 149</a>
<a href="#">aries_cloudagent.messaging.decorators.default (module), 49</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handlers.impl.impl.impl (module), 149</a>
<a href="#">aries_cloudagent.messaging.decorators.localization_decorator (module), 49</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handlers.impl.impl.impl.factory (module), 149</a>
<a href="#">aries_cloudagent.messaging.decorators.please_attach_decorator (module), 50</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handlers.impl.impl.impl.impl (module), 149</a>
<a href="#">aries_cloudagent.messaging.decorators.signature_decorator (module), 51</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handlers.impl.impl.impl.impl.factory (module), 149</a>
<a href="#">aries_cloudagent.messaging.decorators.thread_decorator (module), 53</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handlers.impl.impl.impl.impl.impl (module), 149</a>
<a href="#">aries_cloudagent.messaging.decorators.timing_decorator (module), 54</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handlers.impl.impl.impl.impl.impl.factory (module), 149</a>
<a href="#">aries_cloudagent.messaging.decorators.transport_decorator (module), 56</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handlers.impl.impl.impl.impl.impl.impl (module), 149</a>
<a href="#">aries_cloudagent.messaging.error (module), 69</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handlers.impl.impl.impl.impl.impl.impl.factory (module), 149</a>
<a href="#">aries_cloudagent.messaging.models (module), 57</a>	<a href="#">aries_cloudagent.protocols.issue_credential.v1_0.handlers.impl.impl.impl.impl.impl.impl.factory (module), 149</a>







(module), 96  
 aries\_cloudagent.transport.pack\_format  
 (module), 99  
 aries\_cloudagent.transport.queue (mod-  
 ule), 97  
 aries\_cloudagent.transport.queue.base  
 (module), 97  
 aries\_cloudagent.transport.queue.basic  
 (module), 98  
 aries\_cloudagent.transport.stats (mod-  
 ule), 100  
 aries\_cloudagent.transport.wire\_format  
 (module), 100  
 aries\_cloudagent.utils (module), 102  
 aries\_cloudagent.utils.classloader (mod-  
 ule), 102  
 aries\_cloudagent.utils.http (module), 103  
 aries\_cloudagent.utils.repeat (module),  
 104  
 aries\_cloudagent.utils.stats (module), 105  
 aries\_cloudagent.utils.task\_queue (mod-  
 ule), 106  
 aries\_cloudagent.verifier (module), 108  
 aries\_cloudagent.verifier.base (module),  
 108  
 aries\_cloudagent.verifier.indy (module),  
 108  
 aries\_cloudagent.version (module), 126  
 aries\_cloudagent.wallet (module), 109  
 aries\_cloudagent.wallet.base (module), 109  
 aries\_cloudagent.wallet.basic (module),  
 112  
 aries\_cloudagent.wallet.crypto (module),  
 115  
 aries\_cloudagent.wallet.error (module),  
 118  
 aries\_cloudagent.wallet.indy (module), 119  
 aries\_cloudagent.wallet.plugin (module),  
 123  
 aries\_cloudagent.wallet.provider (mod-  
 ule), 123  
 aries\_cloudagent.wallet.routes (module),  
 123  
 aries\_cloudagent.wallet.util (module), 125  
 assign\_thread\_from()  
 (aries\_cloudagent.messaging.agent\_message.AgentMessage46  
 method), 66  
 assign\_thread\_id()  
 (aries\_cloudagent.messaging.agent\_message.AgentMessage46  
 method), 66  
 assign\_trace\_decorator()  
 (aries\_cloudagent.messaging.agent\_message.AgentMessage47  
 method), 66  
 assign\_trace\_from()

(aries\_cloudagent.messaging.agent\_message.AgentMessage  
 method), 66  
 attach\_invitation()  
 (aries\_cloudagent.connections.models.connection\_record.Connec  
 method), 138  
 attach\_request() (aries\_cloudagent.connections.models.connection\_  
 method), 139  
 AttachDecorator (class in  
 aries\_cloudagent.messaging.decorators.attach\_decorator),  
 42  
 AttachDecorator.Meta (class in  
 aries\_cloudagent.messaging.decorators.attach\_decorator),  
 43  
 AttachDecoratorData (class in  
 aries\_cloudagent.messaging.decorators.attach\_decorator),  
 44  
 AttachDecoratorData.Meta (class in  
 aries\_cloudagent.messaging.decorators.attach\_decorator),  
 44  
 AttachDecoratorData1JWS (class in  
 aries\_cloudagent.messaging.decorators.attach\_decorator),  
 45  
 AttachDecoratorData1JWS.Meta (class in  
 aries\_cloudagent.messaging.decorators.attach\_decorator),  
 45  
 AttachDecoratorData1JWSSchema (class in  
 aries\_cloudagent.messaging.decorators.attach\_decorator),  
 45  
 AttachDecoratorData1JWSSchema.Meta (class  
 in aries\_cloudagent.messaging.decorators.attach\_decorator),  
 45  
 AttachDecoratorData1JWS (class in  
 aries\_cloudagent.messaging.decorators.attach\_decorator),  
 46  
 AttachDecoratorData1JWS.Meta (class in  
 aries\_cloudagent.messaging.decorators.attach\_decorator),  
 46  
 AttachDecoratorData1JWSHeader (class in  
 aries\_cloudagent.messaging.decorators.attach\_decorator),  
 46  
 AttachDecoratorData1JWSHeader.Meta (class  
 in aries\_cloudagent.messaging.decorators.attach\_decorator),  
 46  
 AttachDecoratorData1JWSHeaderSchema (class  
 in aries\_cloudagent.messaging.decorators.attach\_decorator),  
 46  
 AttachDecoratorData1JWSHeaderSchema.Meta  
 (class in aries\_cloudagent.messaging.decorators.attach\_decorator),  
 46  
 AttachDecoratorData1JWSSchema (class in  
 aries\_cloudagent.messaging.decorators.attach\_decorator),  
 46  
 AttachDecoratorData1JWSSchema.Meta (class  
 in aries\_cloudagent.messaging.decorators.attach\_decorator),  
 46

47 method), 150  
AttachDecoratorDataSchema (class in b64\_decoded\_value() aries\_cloudagent.messaging.decorators.attach\_decorator), (aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.p  
47 method), 172  
AttachDecoratorDataSchema.Meta (class in b64\_to\_bytes() (in module aries\_cloudagent.messaging.decorators.attach\_decorator), aries\_cloudagent.wallet.util), 125  
47 b64\_to\_str() (in module aries\_cloudagent.wallet.util), 125  
AttachDecoratorSchema (class in aries\_cloudagent.messaging.decorators.attach\_decorator), 33  
47 Base58SHA256Hash (class in aries\_cloudagent.messaging.valid), 73  
AttachDecoratorSchema.Meta (class in aries\_cloudagent.messaging.decorators.attach\_decorator), (aries\_cloudagent.messaging.decorators.attach\_decorator.Attachbl  
47 attribute), 44  
attr\_dict() (aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.inner.issue\_credential\_preview.CredentialPreview  
method), 151  
base64\_ (aries\_cloudagent.messaging.decorators.attach\_decorator.Attachbl  
attribute), 47  
aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.URL (class in aries\_cloudagent.messaging.valid), 73  
167 aries\_cloudagent.messaging.valid), 73  
attributes (aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.inner.credential\_preview.CredentialPreviewSchema  
attribute), 151  
aries\_cloudagent.messaging.valid), 73  
attributes (aries\_cloudagent.protocols.present\_proof.v1\_0.messages.server.presentation\_preview.PresentationPreviewSchema  
attribute), 175  
aries\_cloudagent.admin.base\_server), 3  
authn (aries\_cloudagent.connections.models.diddoc.PublicKeyCache (class in aries\_cloudagent.cache.base), 6  
attribute), 129  
BaseDecoratorSet (class in aries\_cloudagent.messaging.decorators.base),  
attribute), 133  
48  
authn\_type (aries\_cloudagent.connections.models.diddoc.LinkedDataKeySpec  
attribute), 128  
BaseExchangeRecord (class in aries\_cloudagent.messaging.models.base\_record),  
attribute), 132  
59  
authn\_type (aries\_cloudagent.connections.models.diddoc.PublicKeyPublicKeyType (class in aries\_cloudagent.messaging.models.base\_record),  
attribute), 133  
aries\_cloudagent.messaging.models.base\_record),  
authn\_type (aries\_cloudagent.connections.models.diddoc.PublicKeyType  
attribute), 129  
BaseExchangeSchema.Meta (class in aries\_cloudagent.messaging.models.base\_record),  
attribute), 128  
59  
authnkey (aries\_cloudagent.connections.models.diddoc.DIDDocBaseDIDDoc (class in aries\_cloudagent.messaging.base\_handler), 69  
attribute), 131  
aries\_cloudagent.messaging.base\_handler), 69  
auto\_issue (aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_exchange.V10CredentialExchangeSchema  
attribute), 160  
24  
auto\_offer (aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_exchange.V10CredentialExchangeSchema  
attribute), 160  
aries\_cloudagent.transport.inbound.base),  
auto\_present (aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation\_exchange.V10PresentationExchangeSchema  
attribute), 181  
BaseInjector (class in aries\_cloudagent.messaging.models.base\_handler), 69  
auto\_remove (aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_exchange.V10CredentialExchangeSchema  
attribute), 160  
BaseIssuer (class in aries\_cloudagent.issuer.base),  
available\_commands() (in module aries\_cloudagent.commands), 7  
BaseLedger (class in aries\_cloudagent.ledger.base),  
32  
BaseMessageQueue (class in aries\_cloudagent.transport.queue.base),  
97  
BaseModel (class in aries\_cloudagent.messaging.models.base\_handler), 69  
(aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_exchange.V10CredentialExchangeSchema),

## B

b58\_to\_bytes() (in module aries\_cloudagent.wallet.util), 125  
b64\_decoded\_value() (aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_exchange.V10CredentialExchangeSchema),

BaseModel.Meta	(class in aries_cloudagent.messaging.models.base),	57	BasicWallet (class in aries_cloudagent.wallet.basic),	112
BaseModelError		57	bind_instance() (aries_cloudagent.config.injector.Injector method),	15
BaseModelSchema	(class in aries_cloudagent.messaging.models.base),	58	bind_provider() (aries_cloudagent.config.injector.Injector method),	15
BaseModelSchema.Meta	(class in aries_cloudagent.messaging.models.base),	58	bind_providers() (aries_cloudagent.config.default_context.DefaultContextBuilder method),	13
BaseOutboundTransport	(class in aries_cloudagent.transport.outbound.base),	92	build() (aries_cloudagent.config.base_context.ContextBuilder method),	13
BaseProvider	(class in aries_cloudagent.config.base),	11	build() (aries_cloudagent.config.default_context.DefaultContextBuilder method),	13
BaseRecord	(class in aries_cloudagent.messaging.models.base_record),	59	byte_count (aries_cloudagent.messaging.decorators.attach_decorator.Attribute),	48
BaseRecord.Meta	(class in aries_cloudagent.messaging.models.base_record)	60	bytes_to_b58() (in module aries_cloudagent.wallet.util),	125
BaseRecordSchema	(class in aries_cloudagent.messaging.models.base_record)	62	bytes_to_b64() (in module aries_cloudagent.wallet.util),	125
BaseRecordSchema.Meta	(class in aries_cloudagent.messaging.models.base_record),	62	ByteSize (class in aries_cloudagent.config.util),	18
BaseResponder	(class in aries_cloudagent.messaging.responder),	70	CACHE_ENABLED (aries_cloudagent.connections.models.connection_record.Attribute),	138
BaseSettings	(class in aries_cloudagent.config.base),	11	CACHE_ENABLED (aries_cloudagent.messaging.models.base_record.BaseRecord.Attribute),	59
BaseStorage	(class in aries_cloudagent.storage.base),	75	cache_key() (aries_cloudagent.messaging.models.base_record.BaseRecord class method),	60
BaseStorageRecordSearch	(class in aries_cloudagent.storage.base),	76	CACHE_TTL (aries_cloudagent.messaging.models.base_record.BaseRecord.Attribute),	59
BaseVerifier	(class in aries_cloudagent.verifier.base),	108	CachedProvider (class in aries_cloudagent.config.provider),	16
BaseWallet	(class in aries_cloudagent.wallet.base),	109	CacheError,	6
BaseWireFormat	(class in aries_cloudagent.transport.wire_format),	100	CacheKeyLock (class in aries_cloudagent.cache.base),	6
basic_tag_query_match()	(in module aries_cloudagent.storage.basic),	79	can_respond (aries_cloudagent.transport.inbound.session.InboundSession.Attribute),	90
basic_tag_value_match()	(in module aries_cloudagent.storage.basic),	79	cancel() (aries_cloudagent.utils.task_queue.PendingTask method),	106
BasicCache	(class in aries_cloudagent.cache.basic),	7	cancel() (aries_cloudagent.utils.task_queue.TaskQueue method),	106
BasicMessageQueue	(class in aries_cloudagent.transport.queue.basic),	98	cancel_pending() (aries_cloudagent.utils.task_queue.TaskQueue method),	106
BasicStorage	(class in aries_cloudagent.storage.basic),	77	cancelled (aries_cloudagent.utils.task_queue.PendingTask.Attribute),	106
BasicStorageRecordSearch	(class in aries_cloudagent.connections.models.diddoc.util),		cancelled (aries_cloudagent.utils.task_queue.TaskQueue.Attribute),	107
			canon() (in module aries_cloudagent.messaging.util),	72
			canon_did() (in module aries_cloudagent.connections.models.diddoc.util),	135
			canon_ref() (in module aries_cloudagent.connections.models.diddoc.util),	

135	close()	(aries_cloudagent.storage.basic.BaseStorageRecordSearch
catalogs (aries_cloudagent.messaging.decorators.localization_decorator	close()	(aries_cloudagent.storage.indy.IndyStorageRecordSearch
attribute), 50	close()	(aries_cloudagent.transport.inbound.session.InboundSession
CATEGORIES (aries_cloudagent.config.argparse.AdminGroup	close()	(aries_cloudagent.wallet.base.BaseWallet
attribute), 9	close()	(aries_cloudagent.wallet.basic.BasicWallet
CATEGORIES (aries_cloudagent.config.argparse.DebugGroup	close()	(aries_cloudagent.wallet.indy.IndyWallet
attribute), 9	close()	(aries_cloudagent.wallet.indy.IndyWallet
CATEGORIES (aries_cloudagent.config.argparse.GeneralGroup	close()	(aries_cloudagent.wallet.indy.IndyWallet
attribute), 9	close()	(aries_cloudagent.wallet.indy.IndyWallet
CATEGORIES (aries_cloudagent.config.argparse.LedgerGroup	close()	(aries_cloudagent.wallet.indy.IndyWallet
attribute), 10	close()	(aries_cloudagent.wallet.indy.IndyWallet
CATEGORIES (aries_cloudagent.config.argparse.LoggingGroup	close()	(aries_cloudagent.wallet.indy.IndyWallet
attribute), 10	close()	(aries_cloudagent.wallet.indy.IndyWallet
CATEGORIES (aries_cloudagent.config.argparse.ProtocolGroup	close()	(aries_cloudagent.wallet.indy.IndyWallet
attribute), 10	close()	(aries_cloudagent.wallet.indy.IndyWallet
CATEGORIES (aries_cloudagent.config.argparse.TransportGroup	close()	(aries_cloudagent.wallet.indy.IndyWallet
attribute), 10	close()	(aries_cloudagent.wallet.indy.IndyWallet
CATEGORIES (aries_cloudagent.config.argparse.WalletGroup	close()	(aries_cloudagent.wallet.indy.IndyWallet
attribute), 10	close()	(aries_cloudagent.wallet.indy.IndyWallet
check_dump_decorators()	close()	(aries_cloudagent.wallet.indy.IndyWallet
(aries_cloudagent.messaging.agent_message.AgentMessageSchema	close()	(aries_cloudagent.wallet.indy.IndyWallet
method), 68	close()	(aries_cloudagent.wallet.indy.IndyWallet
check_existing_schema()	close()	(aries_cloudagent.wallet.indy.IndyWallet
(aries_cloudagent.ledger.indy.IndyLedger	close()	(aries_cloudagent.wallet.indy.IndyWallet
method), 34	close()	(aries_cloudagent.wallet.indy.IndyWallet
check_pool_config()	close()	(aries_cloudagent.wallet.indy.IndyWallet
(aries_cloudagent.ledger.indy.IndyLedger	close()	(aries_cloudagent.wallet.indy.IndyWallet
method), 34	close()	(aries_cloudagent.wallet.indy.IndyWallet
ClassLoader	close()	(aries_cloudagent.wallet.indy.IndyWallet
(class in aries_cloudagent.utils.classloader), 102	close()	(aries_cloudagent.wallet.indy.IndyWallet
ClassNotFoundError, 102	close()	(aries_cloudagent.wallet.indy.IndyWallet
ClassProvider	close()	(aries_cloudagent.wallet.indy.IndyWallet
(class in aries_cloudagent.config.provider), 16	close()	(aries_cloudagent.wallet.indy.IndyWallet
ClassProvider.Inject	close()	(aries_cloudagent.wallet.indy.IndyWallet
(class in aries_cloudagent.config.provider), 16	close()	(aries_cloudagent.wallet.indy.IndyWallet
clear()	close()	(aries_cloudagent.wallet.indy.IndyWallet
(aries_cloudagent.cache.base.BaseCache	close()	(aries_cloudagent.wallet.indy.IndyWallet
method), 6	close()	(aries_cloudagent.wallet.indy.IndyWallet
clear()	close()	(aries_cloudagent.wallet.indy.IndyWallet
(aries_cloudagent.cache.basic.BasicCache	close()	(aries_cloudagent.wallet.indy.IndyWallet
method), 7	close()	(aries_cloudagent.wallet.indy.IndyWallet
clear_binding()	close()	(aries_cloudagent.wallet.indy.IndyWallet
(aries_cloudagent.config.injector.Injector	close()	(aries_cloudagent.wallet.indy.IndyWallet
method), 15	close()	(aries_cloudagent.wallet.indy.IndyWallet
clear_cached()	close()	(aries_cloudagent.wallet.indy.IndyWallet
(aries_cloudagent.messaging.models.base_record.BaseRecord	close()	(aries_cloudagent.wallet.indy.IndyWallet
method), 60	close()	(aries_cloudagent.wallet.indy.IndyWallet
clear_cached_key()	close()	(aries_cloudagent.wallet.indy.IndyWallet
(aries_cloudagent.messaging.models.base_record.BaseRecord	close()	(aries_cloudagent.wallet.indy.IndyWallet
class method), 60	close()	(aries_cloudagent.wallet.indy.IndyWallet
clear_response()	close()	(aries_cloudagent.wallet.indy.IndyWallet
(aries_cloudagent.transport.inbound.session.InboundSession	close()	(aries_cloudagent.wallet.indy.IndyWallet
method), 90	close()	(aries_cloudagent.wallet.indy.IndyWallet
clear_value()	close()	(aries_cloudagent.wallet.indy.IndyWallet
(aries_cloudagent.config.settings.Settings	close()	(aries_cloudagent.wallet.indy.IndyWallet
method), 17	close()	(aries_cloudagent.wallet.indy.IndyWallet
close()	close()	(aries_cloudagent.wallet.indy.IndyWallet
(aries_cloudagent.ledger.indy.IndyLedger	close()	(aries_cloudagent.wallet.indy.IndyWallet
method), 34	close()	(aries_cloudagent.wallet.indy.IndyWallet
close()	close()	(aries_cloudagent.wallet.indy.IndyWallet
(aries_cloudagent.storage.base.BaseStorageRecordSearch	close()	(aries_cloudagent.wallet.indy.IndyWallet
method), 76	close()	(aries_cloudagent.wallet.indy.IndyWallet

[attribute](#)), 160  
[connection\\_id\(aries\\_cloudagent.protocols.present\\_proof.v1\\_0.manager.CredentialManager\)](#) (class in [aries\\_cloudagent.config.base.BaseInjector](#) module), 181  
[connection\\_id\(aries\\_cloudagent.transport.inbound.receive\\_message.ReceiveMessage\)](#) (class in [aries\\_cloudagent.config.base.BaseInjector](#) module), 88  
[connection\\_queued\\_end\(\)](#) (aries\_cloudagent.transport.stats.StatsTracer method), 100  
[connection\\_queued\\_start\(\)](#) (aries\_cloudagent.transport.stats.StatsTracer method), 100  
[connection\\_ready\(aries\\_cloudagent.messaging.request\\_context.RequestContext\)](#) (class in [aries\\_cloudagent.config.injection\\_context.InjectionContext](#) module), 70  
[connection\\_ready\(\)](#) (aries\_cloudagent.transport.stats.StatsTracer method), 100  
[connection\\_record](#) (aries\_cloudagent.messaging.request\_context.RequestContext) (class in [aries\\_cloudagent.config.injection\\_context.InjectionContext](#) module), 70  
[ConnectionRecord](#) (class in [aries\\_cloudagent.config.injection\\_context.InjectionContext](#) module), 136  
[ConnectionRecord.Meta](#) (class in [aries\\_cloudagent.config.injection\\_context.InjectionContext](#) module), 138  
[ConnectionRecordSchema](#) (class in [aries\\_cloudagent.config.injection\\_context.InjectionContext](#) module), 140  
[ConnectionRecordSchema.Meta](#) (class in [aries\\_cloudagent.config.injection\\_context.InjectionContext](#) module), 140  
[ConnectionTarget](#) (class in [aries\\_cloudagent.config.injection\\_context.InjectionContext](#) module), 141  
[ConnectionTarget.Meta](#) (class in [aries\\_cloudagent.config.injection\\_context.InjectionContext](#) module), 141  
[ConnectionTargetSchema](#) (class in [aries\\_cloudagent.config.injection\\_context.InjectionContext](#) module), 141  
[ConnectionTargetSchema.Meta](#) (class in [aries\\_cloudagent.config.injection\\_context.InjectionContext](#) module), 141  
[CONTEXT\(aries\\_cloudagent.connections.models.diddoc.DIDDoc\)](#) (aries\_cloudagent.connections.models.diddoc.DIDDoc attribute), 127  
[CONTEXT\(aries\\_cloudagent.connections.models.diddoc.diddoc.DIDDoc\)](#) (aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc attribute), 131  
[context\(aries\\_cloudagent.protocols.issue\\_credential.v1\\_0.manager.CredentialManager\)](#) (aries\_cloudagent.protocols.issue\_credential.v1\_0.manager.CredentialManager attribute), 160  
[context\(aries\\_cloudagent.protocols.present\\_proof.v1\\_0.manager.PresentationManager\)](#) (aries\_cloudagent.protocols.present\_proof.v1\_0.manager.PresentationManager attribute), 182  
[ContextBuilder](#) (class in [aries\\_cloudagent.config.injection\\_context.InjectionContext](#) module), 12  
[controller\(aries\\_cloudagent.connections.models.diddoc.PublicKey\)](#) (aries\_cloudagent.connections.models.diddoc.PublicKey attribute), 28



<code>create_and_store_schema()</code> ( <i>aries_cloudagent.issuer.indy.IndyIssuer</i> method), 30	<code>method</code> ), 24
<code>create_bound_request()</code> ( <i>aries_cloudagent.protocols.present_proof.v1_0.manager.PresentationManager</i> method), 183	<code>create_presentation()</code> ( <i>aries_cloudagent.holder.indy.IndyHolder</i> method), 26
<code>create_credential()</code> ( <i>aries_cloudagent.issuer.base.BaseIssuer</i> method), 28	<code>create_presentation()</code> ( <i>aries_cloudagent.protocols.present_proof.v1_0.manager.PresentationManager</i> method), 183
<code>create_credential()</code> ( <i>aries_cloudagent.issuer.indy.IndyIssuer</i> method), 30	<code>create_proposal()</code> ( <i>aries_cloudagent.protocols.issue_credential.v1_0.manager.CredentialManager</i> method), 161
<code>create_credential_offer()</code> ( <i>aries_cloudagent.issuer.base.BaseIssuer</i> method), 28	<code>create_public_did()</code> ( <i>aries_cloudagent.wallet.base.BaseWallet</i> method), 109
<code>create_credential_offer()</code> ( <i>aries_cloudagent.issuer.indy.IndyIssuer</i> method), 31	<code>create_request()</code> ( <i>aries_cloudagent.protocols.issue_credential.v1_0.manager.CredentialManager</i> method), 161
<code>create_credential_request()</code> ( <i>aries_cloudagent.holder.base.BaseHolder</i> method), 24	<code>create_revocation_state()</code> ( <i>aries_cloudagent.holder.base.BaseHolder</i> method), 24
<code>create_credential_request()</code> ( <i>aries_cloudagent.holder.indy.IndyHolder</i> method), 25	<code>create_revocation_state()</code> ( <i>aries_cloudagent.holder.indy.IndyHolder</i> method), 26
<code>create_exchange_for_proposal()</code> ( <i>aries_cloudagent.protocols.present_proof.v1_0.manager.PresentationManager</i> method), 183	<code>create_session()</code> ( <i>aries_cloudagent.transport.inbound.base.BaseInboundTransport</i> method), 83
<code>create_exchange_for_request()</code> ( <i>aries_cloudagent.protocols.present_proof.v1_0.manager.PresentationManager</i> method), 183	<code>create_session()</code> ( <i>aries_cloudagent.transport.inbound.manager.InboundTransportManager</i> method), 86
<code>create_keypair()</code> (in module <i>aries_cloudagent.wallet.crypto</i> ), 116	<code>create_session()</code> ( <i>aries_cloudagent.wallet.base.BaseWallet</i> method), 110
<code>create_local_did()</code> ( <i>aries_cloudagent.wallet.base.BaseWallet</i> method), 109	<code>create_session()</code> ( <i>aries_cloudagent.wallet.basic.BasicWallet</i> method), 112
<code>create_local_did()</code> ( <i>aries_cloudagent.wallet.basic.BasicWallet</i> method), 112	<code>create_signing_key()</code> ( <i>aries_cloudagent.wallet.indy.IndyWallet</i> method), 120
<code>create_local_did()</code> ( <i>aries_cloudagent.wallet.indy.IndyWallet</i> method), 119	<code>created</code> ( <i>aries_cloudagent.wallet.base.BaseWallet</i> attribute), 110
<code>create_offer()</code> ( <i>aries_cloudagent.protocols.issue_credential.v1_0.manager.CredentialManager</i> method), 160	<code>created</code> ( <i>aries_cloudagent.wallet.basic.BasicWallet</i> attribute), 113
<code>create_outbound()</code> ( <i>aries_cloudagent.core.dispatcher.DispatcherResponder</i> method), 21	<code>created</code> ( <i>aries_cloudagent.wallet.indy.IndyWallet</i> attribute), 120
<code>create_outbound()</code> ( <i>aries_cloudagent.messaging.responder.BaseResponder</i> method), 70	<code>created_at</code> ( <i>aries_cloudagent.messaging.models.base_record.BaseRecord</i> attribute), 66
<code>create_pool_config()</code> ( <i>aries_cloudagent.ledger.indy.IndyLedger</i> method), 35	<code>cred_def_id</code> ( <i>aries_cloudagent.protocols.issue_credential.v1_0.messages.CredDefId</i> attribute), 156
<code>create_presentation()</code> ( <i>aries_cloudagent.holder.base.BaseHolder</i> method), 24	<code>cred_def_id</code> ( <i>aries_cloudagent.protocols.present_proof.v1_0.messages.CredDefId</i> attribute), 173
	<code>cred_def_id</code> ( <i>aries_cloudagent.protocols.present_proof.v1_0.messages.CredDefId</i> attribute), 174
	<code>CredAttrSpec</code> (class in <i>aries_cloudagent.protocols.issue_credential.v1_0.messages.inner_messages</i> ), 149
	<code>CredAttrSpec.Meta</code> (class in <i>aries_cloudagent.protocols.issue_credential.v1_0.messages.inner_messages</i> ), 150

File	Line	Code
CredAttrSpecSchema		(class in aries_cloudagent.protocols.issue_credential.v1_0.routes),
aries_cloudagent.protocols.issue_credential.v1_0.messages.IssuerCredentialPreview	150	credential_exchange_send() (in module
CredAttrSpecSchema.Meta		(class in aries_cloudagent.protocols.issue_credential.v1_0.routes),
aries_cloudagent.protocols.issue_credential.v1_0.messages.IssuerCredentialPreview	150	credential_exchange_send_bound_offer()
credential(aries_cloudagent.protocols.issue_credential.v1_0.models.CredentialExchangeV10CredentialExchangeSchema		attribute), 160
credential_definition_id		credential_exchange_send_free_offer()
(aries_cloudagent.protocols.issue_credential.v1_0.models.CredentialExchangeV10CredentialExchangeSchema		attribute), 160
credential_definition_id2schema_id()		credential_exchange_send_proposal() (in
(aries_cloudagent.ledger.indy.IndyLedger		method), 35
credential_definition_in_wallet()		credential_exchange_send_request() (in
(aries_cloudagent.issuer.base.BaseIssuer		method), 29
credential_definition_in_wallet()		credential_exchange_store() (in module
(aries_cloudagent.issuer.indy.IndyIssuer		method), 31
credential_definitions_created() (in module aries_cloudagent.messaging.credential_definitions.routes)	41	credential_id(aries_cloudagent.protocols.issue_credential.v1_0.models.CredentialExchangeV10CredentialExchange
credential_definitions_get_credential_definition_id()	42	(in module aries_cloudagent.messaging.credential_definitions.routes)
credential_definitions_send_credential_definition_id()	42	(in module aries_cloudagent.messaging.credential_definitions.routes)
credential_exchange_id		(aries_cloudagent.protocols.issue_credential.v1_0.models.CredentialExchangeV10CredentialExchange
(aries_cloudagent.protocols.issue_credential.v1_0.models.CredentialExchangeV10CredentialExchange		attribute), 159
credential_exchange_id		(aries_cloudagent.protocols.issue_credential.v1_0.models.CredentialExchangeV10CredentialExchange
(aries_cloudagent.protocols.issue_credential.v1_0.models.CredentialExchangeV10CredentialExchange		attribute), 160
credential_exchange_issue() (in module aries_cloudagent.protocols.issue_credential.v1_0.messages)	167	credential_request_metadata
(aries_cloudagent.protocols.issue_credential.v1_0.messages)	167	(aries_cloudagent.protocols.issue_credential.v1_0.models.CredentialExchangeV10CredentialExchange
credential_exchange_list() (in module aries_cloudagent.protocols.issue_credential.v1_0.messages)	167	CredentialAck (class in
credential_exchange_problem_report() (in module aries_cloudagent.protocols.issue_credential.v1_0.messages)	167	CredentialAck.Meta (class in
(in module aries_cloudagent.protocols.issue_credential.v1_0.messages)	168	CredentialHandler (class in
credential_exchange_remove() (in module aries_cloudagent.protocols.issue_credential.v1_0.messages)	168	CredentialAckSchema (class in
(aries_cloudagent.protocols.issue_credential.v1_0.messages)	168	CredentialAckSchema.Meta (class in
credential_exchange_revoke() (in module aries_cloudagent.protocols.issue_credential.v1_0.messages)	152	

CredentialDefinitionGetResultsSchema	151	
(class in aries_cloudagent.messaging.credential_definitions.rules), reviewSchema (class in		
40	aries_cloudagent.protocols.issue_credential.v1_0.messages.inner	
CredentialDefinitionSchema (class in	151	
aries_cloudagent.messaging.credential_definitions.rules), CredentialPreviewSchema.Meta (class in		
40	aries_cloudagent.protocols.issue_credential.v1_0.messages.inner	
CredentialDefinitionsCreatedResultsSchema	151	
(class in aries_cloudagent.messaging.credential_definitions.rules), proposal (class in		
41	aries_cloudagent.protocols.issue_credential.v1_0.messages.crede	
CredentialDefinitionSendRequestSchema	154	
(class in aries_cloudagent.messaging.credential_definitions.rules), proposal.Meta (class in		
40	aries_cloudagent.protocols.issue_credential.v1_0.messages.crede	
CredentialDefinitionSendResultsSchema	155	
(class in aries_cloudagent.messaging.credential_definitions.rules), proposalHandler (class in		
41	aries_cloudagent.protocols.issue_credential.v1_0.handlers.creden	
CredentialIssue (class in	149	
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue), schema (class in		
152	aries_cloudagent.protocols.issue_credential.v1_0.messages.crede	
CredentialIssue.Meta (class in	155	
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue), schema.Meta (class in		
152	aries_cloudagent.protocols.issue_credential.v1_0.messages.crede	
CredentialIssueHandler (class in	156	
aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_issue_handler), (class in		
148	aries_cloudagent.protocols.issue_credential.v1_0.messages.crede	
CredentialIssueSchema (class in	156	
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue), meta (class in		
152	aries_cloudagent.protocols.issue_credential.v1_0.messages.crede	
CredentialIssueSchema.Meta (class in	156	
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue), handler (class in		
153	aries_cloudagent.protocols.issue_credential.v1_0.handlers.creden	
CredentialManager (class in	149	
aries_cloudagent.protocols.issue_credential.v1_0.manager), credentialRequestSchema (class in		
160	aries_cloudagent.protocols.issue_credential.v1_0.messages.crede	
CredentialManagerError, 163	157	
CredentialOffer (class in CredentialRequestSchema.Meta (class in		
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer), protocols.issue_credential.v1_0.messages.crede		
153	157	
CredentialOffer.Meta (class in credentials_attach		
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer), protocols.issue_credential.v1_0.messages.cred		
153	attribute), 153	
CredentialOfferHandler (class in current_active(aries_cloudagent.utils.task_queue.TaskQueue		
aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_offer_handler),		
148	current_pending(aries_cloudagent.utils.task_queue.TaskQueue	
CredentialOfferSchema (class in	attribute), 107	
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer), aries_cloudagent.utils.task_queue.TaskQueue		
154	attribute), 107	
CredentialOfferSchema.Meta (class in		
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer),		
154	data(aries_cloudagent.messaging.decorators.attach_decorator.AttachDeco	
CredentialPreview (class in	attribute), 48	
aries_cloudagent.protocols.issue_credential.v1_0.messages.inner_credential_preview), in		
150	aries_cloudagent.messaging.util), 72	module
CredentialPreview.Meta (class in datetime_to_str() (in		
aries_cloudagent.protocols.issue_credential.v1_0.messages.inner_credential_preview), aries_cloudagent.messaging.util), 72		module



DebugGroup (class in method), 75  
 aries\_cloudagent.config.argparse), 9  
 decode() (aries\_cloudagent.messaging.decorators.signature\_decorator), 52  
 decode\_pack\_message() (in module aries\_cloudagent.wallet.crypto), 116  
 decode\_pack\_message\_outer() (in module aries\_cloudagent.wallet.crypto), 116  
 decode\_pack\_message\_payload() (in module aries\_cloudagent.wallet.crypto), 116  
 DecoratorError, 49  
 DecoratorSet (class in aries\_cloudagent.messaging.decorators.default), 49  
 decrypt\_plaintext() (in module aries\_cloudagent.wallet.crypto), 116  
 default\_endpoint (aries\_cloudagent.messaging.request\_context.RequestContext attribute), 70  
 DEFAULT\_FRESHNESS (aries\_cloudagent.wallet.indy.IndyWallet attribute), 119  
 DEFAULT\_KEY (aries\_cloudagent.wallet.indy.IndyWallet attribute), 119  
 DEFAULT\_KEY\_DERIVATION (aries\_cloudagent.wallet.indy.IndyWallet attribute), 119  
 default\_label (aries\_cloudagent.messaging.request\_context.RequestContext attribute), 70  
 DEFAULT\_NAME (aries\_cloudagent.wallet.indy.IndyWallet attribute), 119  
 DEFAULT\_STORAGE\_TYPE (aries\_cloudagent.wallet.indy.IndyWallet attribute), 119  
 DefaultContextBuilder (class in aries\_cloudagent.config.default\_context), 13  
 delay\_milli (aries\_cloudagent.messaging.decorators.timing\_decorator), 56  
 delete\_credential() (aries\_cloudagent.holder.base.BaseHolder method), 24  
 delete\_credential() (aries\_cloudagent.holder.indy.IndyHolder method), 26  
 delete\_record() (aries\_cloudagent.messaging.models.base.BaseRecord method), 60  
 delete\_record() (aries\_cloudagent.storage.base.BaseStorage method), 75  
 delete\_record() (aries\_cloudagent.storage.basic.BasicStorage method), 78  
 delete\_record() (aries\_cloudagent.storage.indy.IndyStorage method), 80  
 delete\_record\_tags() (aries\_cloudagent.storage.base.BaseStorage method), 75  
 delete\_record\_tags() (aries\_cloudagent.storage.indy.IndyStorage method), 80  
 deliver\_queued\_message() (aries\_cloudagent.transport.outbound.manager.OutboundTransport method), 94  
 DeliveryQueue (class in aries\_cloudagent.transport.inbound.delivery\_queue), 84  
 dequeue() (aries\_cloudagent.transport.queue.base.BaseMessageQueue method), 97  
 dequeue() (aries\_cloudagent.transport.queue.basic.BasicMessageQueue method), 97  
 description (aries\_cloudagent.messaging.decorators.attach\_decorator attribute), 48  
 deserialize() (aries\_cloudagent.connections.models.diddoc.DIDDoc class method), 128  
 deserialize() (aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc class method), 131  
 deserialize() (aries\_cloudagent.messaging.models.base.BaseModel class method), 57  
 did (aries\_cloudagent.connections.models.connection\_target.ConnectionTarget attribute), 142  
 did (aries\_cloudagent.connections.models.diddoc.DIDDoc attribute), 128  
 did (aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc attribute), 131  
 did (aries\_cloudagent.connections.models.diddoc.PublicKey attribute), 129  
 did (aries\_cloudagent.connections.models.diddoc.publickey.PublicKey attribute), 133  
 did (aries\_cloudagent.connections.models.diddoc.Service attribute), 134  
 did (aries\_cloudagent.connections.models.diddoc.service.Service attribute), 134  
 did (aries\_cloudagent.wallet.base.DIDInfo attribute), 112  
 did\_key() (in module aries\_cloudagent.messaging.decorators.attach\_decorator), 48  
 did\_key() (aries\_cloudagent.ledger.base.BaseLedger method), 32  
 DIDDoc (class in aries\_cloudagent.connections.models.diddoc), 127  
 DIDDoc (class in aries\_cloudagent.connections.models.diddoc.diddoc), 131  
 DIDInfo (class in aries\_cloudagent.wallet.base), 112  
 DIDKey (class in aries\_cloudagent.messaging.valid), 73  
 DIDListSchema (class in aries\_cloudagent.wallet.routes), 123

**DIDResultSchema** (class in *aries\_cloudagent.wallet.routes*), 123  
**DIDSchema** (class in *aries\_cloudagent.wallet.routes*), 124  
**direct\_response\_mode** (*aries\_cloudagent.transport.inbound.receipt.MessageReceipt* attribute), 88  
**direct\_response\_requested** (*aries\_cloudagent.transport.inbound.receipt.MessageReceipt* attribute), 89  
**DIRECTION\_RECEIVED** (*aries\_cloudagent.connections.models.connection\_record.ConnectionRecord* attribute), 138  
**DIRECTION\_SENT** (*aries\_cloudagent.connections.models.connection\_record.ConnectionRecord* attribute), 138  
**dispatch\_complete()** (*aries\_cloudagent.core.conductor.Conductor* method), 18  
**dispatch\_complete()** (*aries\_cloudagent.transport.inbound.manager.InboundTransportManager* method), 86  
**Dispatcher** (class in *aries\_cloudagent.core.dispatcher*), 20  
**DispatcherResponder** (class in *aries\_cloudagent.core.dispatcher*), 21  
**dns\_resolvehost\_end()** (*aries\_cloudagent.transport.stats.StatsTracer* method), 100  
**dns\_resolvehost\_start()** (*aries\_cloudagent.transport.stats.StatsTracer* method), 100  
**done** (*aries\_cloudagent.cache.base.CacheKeyLock* attribute), 6  
**drain()** (*aries\_cloudagent.utils.task\_queue.TaskQueue* method), 107  
**dump\_decorators()** (*aries\_cloudagent.messaging.agent\_message.AgentMessageSchema* method), 68  
**E**  
**ED25519\_SIG\_2018** (*aries\_cloudagent.connections.models.diddoc.PublicKey.PublicKeyType* attribute), 133  
**ED25519\_SIG\_2018** (*aries\_cloudagent.connections.models.diddoc.PublicKey.PublicKeyType* attribute), 129  
**EDDSA\_SA\_SIG\_SECP256K1** (*aries\_cloudagent.connections.models.diddoc.PublicKey.PublicKeyType* attribute), 133  
**EDDSA\_SA\_SIG\_SECP256K1** (*aries\_cloudagent.connections.models.diddoc.PublicKey.PublicKeyType* attribute), 129  
**enabled** (*aries\_cloudagent.utils.stats.Collector* attribute), 105  
**encode()** (in module *aries\_cloudagent.messaging.util*), 72  
**encode\_message()** (*aries\_cloudagent.transport.pack\_format.PackWire* method), 99  
**encode\_message()** (*aries\_cloudagent.transport.wire\_format.BaseWire* method), 100  
**encode\_message()** (*aries\_cloudagent.transport.wire\_format.JsonWire* method), 101  
**encode\_outbound()** (*aries\_cloudagent.transport.inbound.session.InboundSession* method), 90  
**encode\_pack\_message()** (in module *aries\_cloudagent.wallet.crypto*), 117  
**encode\_record\_message()** (*aries\_cloudagent.transport.outbound.manager.OutboundTransportManager* method), 90  
**ENCODING\_MISMATCH** (*aries\_cloudagent.verifier.indy.PreVerifyResult* attribute), 109  
**encrypt\_plaintext()** (in module *aries\_cloudagent.wallet.crypto*), 117  
**endpoint** (*aries\_cloudagent.connections.models.diddoc.Service* attribute), 130  
**endpoint** (*aries\_cloudagent.connections.models.diddoc.service.Service* attribute), 134  
**enqueue()** (*aries\_cloudagent.transport.queue.base.BaseMessageQueue* method), 97  
**enqueue()** (*aries\_cloudagent.transport.queue.basic.BasicMessageQueue* method), 98  
**enqueue\_message()** (*aries\_cloudagent.transport.outbound.manager.OutboundTransportManager* method), 94  
**enqueue\_webhook()** (*aries\_cloudagent.transport.outbound.manager.OutboundTransportManager* method), 94  
**epoch\_to\_str()** (in module *aries\_cloudagent.messaging.util*), 72  
**error\_code** (*aries\_cloudagent.core.error.BaseError* attribute), 21  
**error\_code** (*aries\_cloudagent.messaging.error.MessageParseError* attribute), 69  
**error\_code** (*aries\_cloudagent.messaging.error.MessagePrepareError* attribute), 68  
**error\_code** (*aries\_cloudagent.transport.error.MessageEncodeError* attribute), 98  
**error\_code** (*aries\_cloudagent.transport.error.MessageParseError* attribute), 99  
**error\_msg** (*aries\_cloudagent.connections.models.connection\_record.ConnectionRecord* attribute), 140  
**error\_msg** (*aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential* attribute), 160  
**error\_msg** (*aries\_cloudagent.protocols.present\_proof.v1\_0.models.present\_proof* attribute), 181  
**EXAMPLE** (*aries\_cloudagent.messaging.valid.Base58SHA256Hash* attribute), 181

[attribute](#)), 73  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.Base64 attribute](#)), 73  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.Base64URL attribute](#)), 73  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.Base64URLNoPad attribute](#)), 73  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.DIDKey attribute](#)), 73  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.IndyCredDefId attribute](#)), 73  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.IndyDID attribute](#)), 73  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.IndyISO8601DateTime attribute](#)), 74  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.IndyPredicate attribute](#)), 74  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.IndyRawPublicKey attribute](#)), 74  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.IndyRevRegId attribute](#)), 74  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.IndySchemaId attribute](#)), 74  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.IndyVersion attribute](#)), 74  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.IntEpoch attribute](#)), 74  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.JSONWebToken attribute](#)), 74  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.JWSHeaderKid attribute](#)), 75  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.SHA256Hash attribute](#)), 75  
 EXAMPLE ([aries\\_cloudagent.messaging.valid.UUIDFour attribute](#)), 75  
 execute() (in module [aries\\_cloudagent.commands.help](#)), 8  
 execute() (in module [aries\\_cloudagent.commands.provision](#)), 8  
 execute() (in module [aries\\_cloudagent.commands.start](#)), 8  
 expire\_messages() ([aries\\_cloudagent.transport.inbound.delivery\\_queue.DeliveryQueue](#) method), 84  
 expires\_time([aries\\_cloudagent.messaging.decorators.timing\\_decorator.TimingDecoratorSchema](#) attribute), 56  
 extend() ([aries\\_cloudagent.config.base.BaseSettings](#) method), 11  
 extend() ([aries\\_cloudagent.config.settings.Settings](#) method), 17  
 extract() ([aries\\_cloudagent.utils.stats.Collector](#) method), 105  
 extract() ([aries\\_cloudagent.utils.stats.Stats](#) method), 105  
 extract\_decorators() ([aries\\_cloudagent.messaging.agent\\_message.AgentMessageSchema](#) method), 68  
 extract\_decorators() ([aries\\_cloudagent.messaging.decorators.base.BaseDecoratorSet](#) method), 48  
 extract\_pack\_recipients() (in module [aries\\_cloudagent.wallet.crypto](#)), 117  
 extract\_payload\_key() (in module [aries\\_cloudagent.wallet.crypto](#)), 117  
 F  
 fetch() ([aries\\_cloudagent.storage.base.BaseStorageRecordSearch](#) method), 76  
 fetch() ([aries\\_cloudagent.storage.basic.BasicStorageRecordSearch](#) method), 79  
 fetch() ([aries\\_cloudagent.storage.indy.IndyStorageRecordSearch](#) method), 82  
 fetch() (in module [aries\\_cloudagent.utils.http](#)), 103  
 fetch\_all() ([aries\\_cloudagent.storage.base.BaseStorageRecordSearch](#) method), 76  
 fetch\_credential\_definition() ([aries\\_cloudagent.ledger.indy.IndyLedger](#) method), 35  
 fetch\_genesis\_transactions() (in module [aries\\_cloudagent.config.ledger](#)), 15  
 fetch\_schema\_by\_id() ([aries\\_cloudagent.ledger.indy.IndyLedger](#) method), 35  
 fetch\_schema\_by\_seq\_no() ([aries\\_cloudagent.ledger.indy.IndyLedger](#) method), 35  
 fetch\_single() ([aries\\_cloudagent.storage.base.BaseStorageRecordSearch](#) method), 77  
 fetch\_stream() (in module [aries\\_cloudagent.utils.http](#)), 103  
 fetch\_txn\_author\_agreement() ([aries\\_cloudagent.ledger.base.BaseLedger](#) method), 32  
 fetch\_txn\_author\_agreement() ([aries\\_cloudagent.ledger.indy.IndyLedger](#) method), 35  
 field() ([aries\\_cloudagent.messaging.decorators.base.BaseDecoratorSet](#) method), 48  
 fields([aries\\_cloudagent.messaging.decorators.base.BaseDecoratorSet](#) attribute), 48  
 file\_ext() (in module [aries\\_cloudagent.wallet.plugin](#)), 123  
 filename([aries\\_cloudagent.messaging.decorators.attach\\_decorator.AttachDecorator](#) attribute), 48  
 final([aries\\_cloudagent.utils.repeat.RepeatAttempt](#) attribute), 104  
 finished\_deliver()

`(aries_cloudagent.transport.outbound.manager.OutboundTransportManager.ledger.indy.IndyLedger method), 94`  
`finished_encode()` `(aries_cloudagent.transport.outbound.manager.OutboundTransportManager.wallet.indy.IndyWallet method), 94`  
`flush()` `(aries_cloudagent.cache.base.BaseCache method), 6`  
`flush()` `(aries_cloudagent.cache.basic.BasicCache method), 7`  
`flush()` `(aries_cloudagent.transport.outbound.manager.OutboundTransportManager.wallet.indy.IndyWallet method), 94`  
`flush()` `(aries_cloudagent.utils.task_queue.TaskQueue method), 107`  
`format_did_info()` `(in module aries_cloudagent.wallet.routes), 124`  
`from_indy_dict()` `(aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator class method), 43`  
`from_json()` `(aries_cloudagent.connections.models.diddoc.DIDDoc class method), 128`  
`from_json()` `(aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc class method), 131`  
`from_json()` `(aries_cloudagent.messaging.models.base.BaseModel class method), 57`  
`from_storage()` `(aries_cloudagent.messaging.models.base_record.BaseRecord class method), 60`  
`future` `(aries_cloudagent.cache.base.CacheKeyLock attribute), 7`  
**G**  
`GeneralGroup` `(class in aries_cloudagent.config.argparse), 9`  
`generate_wallet_key()` `(aries_cloudagent.wallet.indy.IndyWallet class method), 120`  
`get` `(aries_cloudagent.connections.models.diddoc.publickey.PublicKeyType attribute), 133`  
`get` `(aries_cloudagent.connections.models.diddoc.PublicKeyType attribute), 129`  
`get()` `(aries_cloudagent.cache.base.BaseCache method), 6`  
`get()` `(aries_cloudagent.cache.basic.BasicCache method), 7`  
`get_bool()` `(aries_cloudagent.config.base.BaseSettings method), 12`  
`get_cached_key()` `(aries_cloudagent.messaging.models.base_record.BaseRecord class method), 60`  
`get_credential()` `(aries_cloudagent.holder.base.BaseHolder method), 24`  
`get_credential()` `(aries_cloudagent.holder.indy.IndyHolder method), 26`  
`get_credential_definition()` `(aries_cloudagent.ledger.base.BaseLedger method), 32`  
`get_credential_definition()` `(aries_cloudagent.ledger.indy.IndyLedger method), 35`  
`get_credential_definition_tag_policy()` `(aries_cloudagent.wallet.indy.IndyWallet method), 120`  
`get_credentials()` `(aries_cloudagent.holder.indy.IndyHolder method), 26`  
`get_credentials_for_presentation_request_by_referent()` `(aries_cloudagent.holder.indy.IndyHolder method), 26`  
`get_did_endpoint()` `(in module aries_cloudagent.ledger.routes), 38`  
`get_did_verkey()` `(in module aries_cloudagent.ledger.routes), 38`  
`get_endpoint_for_did()` `(aries_cloudagent.ledger.indy.IndyLedger method), 36`  
`get_int()` `(aries_cloudagent.config.base.BaseSettings method), 12`  
`get_key_for_did()` `(aries_cloudagent.ledger.base.BaseLedger method), 32`  
`get_key_for_did()` `(aries_cloudagent.ledger.indy.IndyLedger method), 36`  
`get_latest_txn_author_acceptance()` `(aries_cloudagent.ledger.base.BaseLedger method), 33`  
`get_latest_txn_author_acceptance()` `(aries_cloudagent.ledger.indy.IndyLedger method), 36`  
`get_local_did()` `(aries_cloudagent.wallet.base.BaseWallet method), 110`  
`get_local_did()` `(aries_cloudagent.wallet.basic.BasicWallet method), 113`  
`get_local_did()` `(aries_cloudagent.wallet.indy.IndyWallet method), 120`  
`get_local_did_for_verkey()` `(aries_cloudagent.wallet.base.BaseWallet method), 110`  
`get_local_did_for_verkey()` `(aries_cloudagent.wallet.basic.BasicWallet method), 113`  
`get_local_did_for_verkey()` `(aries_cloudagent.wallet.indy.IndyWallet method), 120`  
`get_local_dids()` `(aries_cloudagent.wallet.base.BaseWallet method), 110`



<code>method</code> ), 110	<code>method</code> ), 36
<code>get_local_dids()</code> ( <i>aries_cloudagent.wallet.basic.BaseWallet</i> <code>method</code> ), 113	<code>get_settings()</code> ( <i>aries_cloudagent.config.argparse.AdminGroup</i> <code>method</code> ), 9
<code>get_local_dids()</code> ( <i>aries_cloudagent.wallet.indy.IndyWallet</i> <code>method</code> ), 120	<code>get_settings()</code> ( <i>aries_cloudagent.config.argparse.ArgumentGroup</i> <code>method</code> ), 9
<code>get_mime_type()</code> ( <i>aries_cloudagent.holder.base.BaseHolder</i> <code>method</code> ), 25	<code>get_settings()</code> ( <i>aries_cloudagent.config.argparse.DebugGroup</i> <code>method</code> ), 9
<code>get_mime_type()</code> ( <i>aries_cloudagent.holder.indy.IndyHolder</i> <code>method</code> ), 27	<code>get_settings()</code> ( <i>aries_cloudagent.config.argparse.GeneralGroup</i> <code>method</code> ), 9
<code>get_one_message_for_key()</code> ( <i>aries_cloudagent.transport.inbound.delivery_queue.DeliveryQueue</i> <code>method</code> ), 85	<code>get_settings()</code> ( <i>aries_cloudagent.config.argparse.LedgerGroup</i> <code>method</code> ), 10
<code>get_provider()</code> ( <i>aries_cloudagent.config.injector.Injector</i> <code>method</code> ), 15	<code>get_settings()</code> ( <i>aries_cloudagent.config.argparse.LoggingGroup</i> <code>method</code> ), 10
<code>get_public_did()</code> ( <i>aries_cloudagent.wallet.base.BaseWallet</i> <code>method</code> ), 110	<code>get_settings()</code> ( <i>aries_cloudagent.config.argparse.ProtocolGroup</i> <code>method</code> ), 10
<code>get_record()</code> ( <i>aries_cloudagent.storage.base.BaseStorage</i> <code>method</code> ), 75	<code>get_settings()</code> ( <i>aries_cloudagent.config.argparse.TransportGroup</i> <code>method</code> ), 10
<code>get_record()</code> ( <i>aries_cloudagent.storage.basic.BasicStorage</i> <code>method</code> ), 78	<code>get_settings()</code> ( <i>aries_cloudagent.config.argparse.WalletGroup</i> <code>method</code> ), 11
<code>get_record()</code> ( <i>aries_cloudagent.storage.indy.IndyStorage</i> <code>method</code> ), 80	<code>get_signature()</code> ( <i>aries_cloudagent.messaging.agent_message.AgentMessage</i> <code>method</code> ), 67
<code>get_registered()</code> ( <i>aries_cloudagent.config.argparse.group</i> <code>class method</code> ), 11	<code>get_signing_key()</code> ( <i>aries_cloudagent.wallet.base.BaseWallet</i> <code>method</code> ), 110
<code>get_registered_transport_for_scheme()</code> ( <i>aries_cloudagent.transport.outbound.manager.OutboundTransportManager</i> <code>method</code> ), 94	<code>get_signing_key()</code> ( <i>aries_cloudagent.wallet.basic.BaseWallet</i> <code>method</code> ), 113
<code>get_revoc_reg_def()</code> ( <i>aries_cloudagent.ledger.base.BaseLedger</i> <code>method</code> ), 33	<code>get_signing_key()</code> ( <i>aries_cloudagent.wallet.indy.IndyWallet</i> <code>method</code> ), 120
<code>get_revoc_reg_def()</code> ( <i>aries_cloudagent.ledger.indy.IndyLedger</i> <code>method</code> ), 36	<code>get_stats()</code> ( <i>aries_cloudagent.core.conductor.Conductor</i> <code>method</code> ), 18
<code>get_revoc_reg_delta()</code> ( <i>aries_cloudagent.ledger.base.BaseLedger</i> <code>method</code> ), 33	<code>get_str()</code> ( <i>aries_cloudagent.config.base.BaseSettings</i> <code>method</code> ), 12
<code>get_revoc_reg_delta()</code> ( <i>aries_cloudagent.ledger.indy.IndyLedger</i> <code>method</code> ), 36	<code>get_tag_map()</code> ( <i>aries_cloudagent.messaging.models.base_record.BaseRecord</i> <code>class method</code> ), 60
<code>get_revoc_reg_entry()</code> ( <i>aries_cloudagent.ledger.base.BaseLedger</i> <code>method</code> ), 33	<code>get_transport_instance()</code> ( <i>aries_cloudagent.transport.inbound.manager.InboundTransportManager</i> <code>method</code> ), 87
<code>get_revoc_reg_entry()</code> ( <i>aries_cloudagent.ledger.indy.IndyLedger</i> <code>method</code> ), 36	<code>get_transport_instance()</code> ( <i>aries_cloudagent.transport.outbound.manager.OutboundTransportManager</i> <code>method</code> ), 94
<code>get_running_transport_for_endpoint()</code> ( <i>aries_cloudagent.transport.outbound.manager.OutboundTransportManager</i> <code>method</code> ), 94	<code>get_txn_author_agreement()</code> ( <i>aries_cloudagent.ledger.base.BaseLedger</i> <code>method</code> ), 33
<code>get_running_transport_for_scheme()</code> ( <i>aries_cloudagent.transport.outbound.manager.OutboundTransportManager</i> <code>method</code> ), 94	<code>get_txn_author_agreement()</code> ( <i>aries_cloudagent.ledger.indy.IndyLedger</i> <code>method</code> ), 36
<code>get_schema()</code> ( <i>aries_cloudagent.ledger.base.BaseLedger</i> <code>method</code> ), 33	<code>get_value()</code> ( <i>aries_cloudagent.config.base.BaseSettings</i> <code>method</code> ), 17
<code>get_schema()</code> ( <i>aries_cloudagent.ledger.indy.IndyLedger</i> <code>method</code> ), 36	<code>get_value()</code> ( <i>aries_cloudagent.config.settings.Settings</i> <code>method</code> ), 17
	<code>GetTagPolicyResultSchema</code> ( <code>class</code> in <i>aries_cloudagent.wallet.routes</i> ), 124



Index 219





[jws \(aries\\_cloudagent.messaging.decorators.attach\\_decorator.AttachDecorator attribute\), 44](#)  
[jws\\_ \(aries\\_cloudagent.messaging.decorators.attach\\_decorator.AttachDecorator attribute\), 47](#)  
[JWSHeaderKid \(class in aries\\_cloudagent.messaging.valid\), 74](#)  
**K**  
[KEY\\_DERIVATION\\_ARGON2I\\_INT \(aries\\_cloudagent.wallet.indy.IndyWallet attribute\), 119](#)  
[KEY\\_DERIVATION\\_ARGON2I\\_MOD \(aries\\_cloudagent.wallet.indy.IndyWallet attribute\), 119](#)  
[KEY\\_DERIVATION\\_RAW \(aries\\_cloudagent.wallet.indy.IndyWallet attribute\), 119](#)  
[KeyInfo \(class in aries\\_cloudagent.wallet.base\), 112](#)  
[kid \(aries\\_cloudagent.messaging.decorators.attach\\_decorator.AttachDecorator attribute\), 47](#)  
**L**  
[label \(aries\\_cloudagent.connections.models.connection\\_target.ConnectionTarget attribute\), 142](#)  
[lastmod\\_time \(aries\\_cloudagent.messaging.decorators.attach\\_decorator.AttachDecorator attribute\), 48](#)  
[ledger\\_accept\\_taa\(\) \(in module aries\\_cloudagent.ledger.routes\), 38](#)  
[LEDGER\\_CLASSES \(aries\\_cloudagent.ledger.provider.LedgerProvider attribute\), 37](#)  
[ledger\\_config\(\) \(in module aries\\_cloudagent.config.ledger\), 15](#)  
[ledger\\_get\\_taa\(\) \(in module aries\\_cloudagent.ledger.routes\), 39](#)  
[LEDGER\\_TYPE \(aries\\_cloudagent.ledger.base.BaseLedger attribute\), 32](#)  
[LEDGER\\_TYPE \(aries\\_cloudagent.ledger.indy.IndyLedger attribute\), 34](#)  
[LedgerConfigError, 34](#)  
[LedgerError, 34](#)  
[LedgerGroup \(class in aries\\_cloudagent.config.argparse\), 9](#)  
[LedgerProvider \(class in aries\\_cloudagent.ledger.provider\), 37](#)  
[LedgerTransactionError, 34](#)  
[LinkedDataKeySpec \(class in aries\\_cloudagent.connections.models.diddoc\), 128](#)  
[LinkedDataKeySpec \(class in aries\\_cloudagent.connections.models.diddoc.publickey\), 132](#)  
[links \(aries\\_cloudagent.messaging.decorators.attach\\_decorator.AttachDecorator attribute\), 44](#)  
[list\\_plain\(\) \(aries\\_cloudagent.protocols.present\\_proof.v1\\_0.message\\_static method\), 172](#)  
[load\\_argument\\_groups\(\) \(in module aries\\_cloudagent.config.argparse\), 11](#)  
[load\\_class\(\) \(aries\\_cloudagent.utils.classloader.ClassLoader class method\), 102](#)  
[load\\_command\(\) \(in module aries\\_cloudagent.commands\), 7](#)  
[load\\_decorator\(\) \(aries\\_cloudagent.messaging.decorators.base.BaseDecorator method\), 48](#)  
[load\\_module\(\) \(aries\\_cloudagent.utils.classloader.ClassLoader class method\), 102](#)  
[load\\_plugins\(\) \(aries\\_cloudagent.config.default\\_context.DefaultContext method\), 13](#)  
[load\\_plugins\(\) \(in module aries\\_cloudagent.wallet.plugin\), 123](#)  
[load\\_protocol\\_version\(\) \(aries\\_cloudagent.core.plugin\\_registry.PluginRegistry method\), 22](#)  
[load\\_protocols\(\) \(aries\\_cloudagent.core.plugin\\_registry.PluginRegistry method\), 22](#)  
[load\\_resource\(\) \(in module aries\\_cloudagent.config.logging\), 16](#)  
[load\\_subclass\\_of\(\) \(aries\\_cloudagent.utils.classloader.ClassLoader class method\), 102](#)  
[locale \(aries\\_cloudagent.messaging.decorators.localization\\_decorator.LocalizationDecorator attribute\), 50](#)  
[localizable \(aries\\_cloudagent.messaging.decorators.localization\\_decorator.LocalizationDecorator attribute\), 50](#)  
[LocalizationDecorator \(class in aries\\_cloudagent.messaging.decorators.localization\\_decorator\), 49](#)  
[LocalizationDecorator.Meta \(class in aries\\_cloudagent.messaging.decorators.localization\\_decorator\), 50](#)  
[LocalizationDecoratorSchema \(class in aries\\_cloudagent.messaging.decorators.localization\\_decorator\), 50](#)  
[LocalizationDecoratorSchema.Meta \(class in aries\\_cloudagent.messaging.decorators.localization\\_decorator\), 50](#)  
[log\(\) \(aries\\_cloudagent.utils.stats.Collector method\), 105](#)  
[log\(\) \(aries\\_cloudagent.utils.stats.Stats method\), 105](#)  
[log\\_state\(\) \(aries\\_cloudagent.messaging.models.base\\_record.BaseRecord class method\), 60](#)  
[LOG\\_STATE\\_FLAG \(aries\\_cloudagent.connections.models.connection\\_rec\\_attribute\), 138](#)  
[LOG\\_STATE\\_FLAG \(aries\\_cloudagent.messaging.models.base\\_record.BaseRecord attribute\), 138](#)

*attribute*), 60  
 log\_task() (*aries\_cloudagent.core.dispatcher.Dispatcher*  
*method*), 20  
 LoggingConfigurator (class in  
*aries\_cloudagent.config.logging*), 15  
 LoggingGroup (class in  
*aries\_cloudagent.config.argparse*), 10

## M

make\_application()  
     (*aries\_cloudagent.admin.server.AdminServer*  
     *method*), 5  
 make\_application()  
     (*aries\_cloudagent.transport.inbound.http.HttpTransport*  
     *method*), 86  
 make\_application()  
     (*aries\_cloudagent.transport.inbound.ws.WsTransport*  
     *method*), 91  
 make\_credential\_definition\_id()  
     (*aries\_cloudagent.issuer.base.BaseIssuer*  
     *method*), 29  
 make\_credential\_definition\_id()  
     (*aries\_cloudagent.issuer.indy.IndyIssuer*  
     *method*), 31  
 make\_message() (*aries\_cloudagent.core.dispatcher.Dispatcher*  
     *method*), 20  
 make\_model() (*aries\_cloudagent.messaging.models.base.BaseModelSchema*  
     *method*), 58  
 make\_queue() (*aries\_cloudagent.transport.queue.basic.BasicMessageQueue*  
     *method*), 98  
 make\_schema\_id() (*aries\_cloudagent.issuer.base.BaseIssuer*  
     *method*), 29  
 make\_schema\_id() (*aries\_cloudagent.issuer.indy.IndyIssuer*  
     *method*), 31  
 mark() (*aries\_cloudagent.utils.stats.Collector* *method*),  
 105  
 master\_secret\_id (*aries\_cloudagent.wallet.indy.IndyWallet*  
     *attribute*), 121  
 match\_post\_filter() (in module  
     *aries\_cloudagent.messaging.models.base\_record*),  
 63  
 max\_active (*aries\_cloudagent.utils.task\_queue.TaskQueue*  
     *attribute*), 107  
 max\_message\_size (*aries\_cloudagent.transport.inbound.base.BaseInboundTransport*  
     *attribute*), 83  
 merge\_revocation\_registry\_deltas()  
     (*aries\_cloudagent.issuer.base.BaseIssuer*  
     *method*), 29  
 merge\_revocation\_registry\_deltas()  
     (*aries\_cloudagent.issuer.indy.IndyIssuer*  
     *method*), 31  
 message (*aries\_cloudagent.core.error.BaseError*  
     *attribute*), 21  
 message (*aries\_cloudagent.messaging.request\_context.RequestContext*  
     *attribute*), 70  
 message\_count\_for\_key()  
     (*aries\_cloudagent.transport.inbound.delivery\_queue.DeliveryQueue*  
     *method*), 85  
 message\_id (*aries\_cloudagent.messaging.decorators.please\_ack\_decorator*  
     *attribute*), 51  
 message\_receipt (*aries\_cloudagent.messaging.request\_context.RequestContext*  
     *attribute*), 70  
 message\_type (*aries\_cloudagent.messaging.agent\_message.AgentMessage*  
     *attribute*), 66  
 message\_type (*aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.issue\_credential\_v1\_0*  
     *attribute*), 151  
 message\_type (*aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.issue\_credential\_v1\_0*  
     *attribute*), 152  
 message\_type (*aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.issue\_credential\_v1\_0*  
     *attribute*), 153  
 message\_type (*aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.issue\_credential\_v1\_0*  
     *attribute*), 155  
 message\_type (*aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.issue\_credential\_v1\_0*  
     *attribute*), 156  
 message\_type (*aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.issue\_credential\_v1\_0*  
     *attribute*), 151  
 message\_type (*aries\_cloudagent.protocols.present\_proof.v1\_0.messages.present\_proof\_v1\_0*  
     *attribute*), 174  
 message\_type (*aries\_cloudagent.protocols.present\_proof.v1\_0.messages.present\_proof\_v1\_0*  
     *attribute*), 176  
 message\_type (*aries\_cloudagent.protocols.present\_proof.v1\_0.messages.present\_proof\_v1\_0*  
     *attribute*), 177  
 message\_type (*aries\_cloudagent.protocols.present\_proof.v1\_0.messages.present\_proof\_v1\_0*  
     *attribute*), 178  
 message\_type (*aries\_cloudagent.protocols.present\_proof.v1\_0.messages.present\_proof\_v1\_0*  
     *attribute*), 178  
 message\_types (*aries\_cloudagent.core.protocol\_registry.ProtocolRegistry*  
     *attribute*), 23  
 MessageEncodeError, 98  
 MessageParseError, 69, 98  
 MessagePrepareError, 69  
 MessageReceipt (class in  
     *aries\_cloudagent.transport.inbound.receipt*),  
 88  
 metadata (*aries\_cloudagent.wallet.base.DIDInfo* *attribute*), 112  
 metadata (*aries\_cloudagent.wallet.base.KeyInfo* *attribute*), 112  
 mime\_type (*aries\_cloudagent.messaging.decorators.attach\_decorator.AttachDecorator*  
     *attribute*), 48  
 mime\_type (*aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.issue\_credential\_v1\_0*  
     *attribute*), 150  
 mime\_type (*aries\_cloudagent.protocols.present\_proof.v1\_0.messages.present\_proof\_v1\_0*  
     *attribute*), 173  
 mime\_types() (*aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.issue\_credential\_v1\_0*  
     *method*), 151  
 MockResponder (class in



	<code>attribute)</code> , 154	<code>opts (aries_cloudagent.messaging.credential_definitions.routes.Credential</code>
OK	<code>(aries_cloudagent.verifier.indy.PreVerifyResult</code> <code>attribute)</code> , 109	<code>attribute)</code> , 40
<code>ok_did()</code>	(in <code>module</code> <code>aries_cloudagent.connections.models.diddoc.util</code> )	<code>opts (aries_cloudagent.messaging.credential_definitions.routes.Credential</code> <code>attribute)</code> , 40
	135	<code>attribute)</code> , 41
<code>older_than()</code>	<code>(aries_cloudagent.transport.inbound.delivery_service.QuickMessage</code> <code>method)</code> , 85	<code>opts (aries_cloudagent.messaging.credential_definitions.routes.Credential</code> <code>attribute)</code> , 41
<code>on (aries_cloudagent.messaging.decorators.please_ack_decorator</code> <code>attribute)</code> , 51	<code>opts (aries_cloudagent.messaging.credential_definitions.routes.Credential</code> <code>attribute)</code> , 41	
<code>on_startup()</code>	<code>(aries_cloudagent.admin.server.AdminServer</code> <code>method)</code> , 5	<code>opts (aries_cloudagent.messaging.schemas.routes.SchemaGetResultsSche</code> <code>attribute)</code> , 63
<code>open()</code>	<code>(aries_cloudagent.ledger.indy.IndyLedger</code> <code>method)</code> , 36	<code>opts (aries_cloudagent.messaging.schemas.routes.SchemaSchema</code> <code>attribute)</code> , 64
<code>open()</code>	<code>(aries_cloudagent.storage.base.BaseStorageRecordSearch</code> <code>method)</code> , 77	<code>opts (aries_cloudagent.messaging.schemas.routes.SchemasCreatedResults</code> <code>attribute)</code> , 65
<code>open()</code>	<code>(aries_cloudagent.storage.basic.BasicStorageRecordSearch</code> <code>method)</code> , 79	<code>opts (aries_cloudagent.messaging.schemas.routes.SchemaSendRequestSch</code> <code>attribute)</code> , 64
<code>open()</code>	<code>(aries_cloudagent.storage.indy.IndyStorageRecordSearch</code> <code>method)</code> , 82	<code>opts (aries_cloudagent.messaging.schemas.routes.SchemaSendResultsSch</code> <code>attribute)</code> , 64
<code>open()</code>	<code>(aries_cloudagent.wallet.base.BaseWallet</code> <code>method)</code> , 111	<code>opts (aries_cloudagent.protocols.issue_credential.v1_0.routes.V10Attribu</code> <code>attribute)</code> , 164
<code>open()</code>	<code>(aries_cloudagent.wallet.basic.BasicWallet</code> <code>method)</code> , 113	<code>opts (aries_cloudagent.protocols.issue_credential.v1_0.routes.V10Creden</code> <code>attribute)</code> , 164
<code>open()</code>	<code>(aries_cloudagent.wallet.indy.IndyWallet</code> <code>method)</code> , 121	<code>opts (aries_cloudagent.protocols.issue_credential.v1_0.routes.V10Creden</code> <code>attribute)</code> , 164
<code>opened (aries_cloudagent.storage.base.BaseStorageRecordSearch</code> <code>attribute)</code> , 77	<code>opts (aries_cloudagent.protocols.issue_credential.v1_0.routes.V10Creden</code> <code>attribute)</code> , 165	
<code>opened (aries_cloudagent.storage.basic.BasicStorageRecordSearch</code> <code>attribute)</code> , 79	<code>opts (aries_cloudagent.protocols.issue_credential.v1_0.routes.V10Creden</code> <code>attribute)</code> , 165	
<code>opened (aries_cloudagent.storage.indy.IndyStorageRecordSearch</code> <code>attribute)</code> , 82	<code>opts (aries_cloudagent.protocols.issue_credential.v1_0.routes.V10Creden</code> <code>attribute)</code> , 165	
<code>opened (aries_cloudagent.wallet.base.BaseWallet</code> <code>at-</code> <code>tribute)</code> , 111	<code>opts (aries_cloudagent.protocols.issue_credential.v1_0.routes.V10Creden</code> <code>attribute)</code> , 166	
<code>opened (aries_cloudagent.wallet.basic.BasicWallet</code> <code>at-</code> <code>tribute)</code> , 113	<code>opts (aries_cloudagent.protocols.issue_credential.v1_0.routes.V10Creden</code> <code>attribute)</code> , 166	
<code>opened (aries_cloudagent.wallet.indy.IndyWallet</code> <code>attribute)</code> , 121	<code>opts (aries_cloudagent.protocols.issue_credential.v1_0.routes.V10Creden</code> <code>attribute)</code> , 166	
<code>option()</code>	<code>(aries_cloudagent.storage.base.BaseStorageRecordSearch</code> <code>method)</code> , 77	<code>opts (aries_cloudagent.protocols.issue_credential.v1_0.routes.V10Publish</code> <code>attribute)</code> , 167
<code>options (aries_cloudagent.storage.base.BaseStorageRecordSearch</code> <code>attribute)</code> , 77	<code>opts (aries_cloudagent.protocols.present_proof.v1_0.routes.IndyProofReq</code> <code>attribute)</code> , 185	
<code>opts (aries_cloudagent.ledger.routes.AMLRecordSchema</code> <code>attribute)</code> , 37	<code>opts (aries_cloudagent.protocols.present_proof.v1_0.routes.IndyProofReq</code> <code>attribute)</code> , 185	
<code>opts (aries_cloudagent.ledger.routes.TAAAcceptanceSchema</code> <code>attribute)</code> , 38	<code>opts (aries_cloudagent.protocols.present_proof.v1_0.routes.IndyProofReq</code> <code>attribute)</code> , 186	
<code>opts (aries_cloudagent.ledger.routes.TAAAcceptSchema</code> <code>attribute)</code> , 37	<code>opts (aries_cloudagent.protocols.present_proof.v1_0.routes.IndyProofReq</code> <code>attribute)</code> , 186	
<code>opts (aries_cloudagent.ledger.routes.TAAInfoSchema</code> <code>attribute)</code> , 38	<code>opts (aries_cloudagent.protocols.present_proof.v1_0.routes.IndyProofReq</code> <code>attribute)</code> , 186	
<code>opts (aries_cloudagent.ledger.routes.TAARecordSchema</code> <code>attribute)</code> , 38	<code>opts (aries_cloudagent.protocols.present_proof.v1_0.routes.IndyRequeste</code> <code>attribute)</code> , 187	
<code>opts (aries_cloudagent.ledger.routes.TAAResultSchema</code> <code>attribute)</code> , 38	<code>opts (aries_cloudagent.protocols.present_proof.v1_0.routes.IndyRequeste</code> <code>attribute)</code> , 187	



[opts \(aries\\_cloudagent.protocols.present\\_proof.v1\\_0.routes.V1\\_0PresentationExchangeListSchema \(class in aries\\_cloudagent.wallet.crypto\), 187](#)  
[opts \(aries\\_cloudagent.protocols.present\\_proof.v1\\_0.routes.V1\\_0PresentationProposalRequestSchema \(class in aries\\_cloudagent.wallet.crypto\), 188](#)  
[opts \(aries\\_cloudagent.protocols.present\\_proof.v1\\_0.routes.V1\\_0PresentationRequestRequestSchema \(class in aries\\_cloudagent.wallet.crypto\), 188](#)  
[opts \(aries\\_cloudagent.protocols.present\\_proof.v1\\_0.routes.V1\\_0PresentationRequestSchema \(class in aries\\_cloudagent.transport.pack\\_format\), 188](#)  
[opts \(aries\\_cloudagent.wallet.crypto.PackMessageSchema \(class in aries\\_cloudagent.wallet.crypto\), 115](#)  
[opts \(aries\\_cloudagent.wallet.crypto.PackRecipientHeaderSchema \(class in aries\\_cloudagent.wallet.crypto\), 115](#)  
[opts \(aries\\_cloudagent.wallet.crypto.PackRecipientSchema \(class in aries\\_cloudagent.cache.base.CacheKeyLock \(class in aries\\_cloudagent.cache.base\), 115](#)  
[opts \(aries\\_cloudagent.wallet.crypto.PackRecipientsSchema \(class in aries\\_cloudagent.wallet.crypto\), 116](#)  
[opts \(aries\\_cloudagent.wallet.routes.DIDListSchema \(class in aries\\_cloudagent.wallet.routes\), 123](#)  
[opts \(aries\\_cloudagent.wallet.routes.DIDResultSchema \(class in aries\\_cloudagent.wallet.routes\), 124](#)  
[opts \(aries\\_cloudagent.wallet.routes.DIDSchema \(class in aries\\_cloudagent.wallet.routes\), 124](#)  
[opts \(aries\\_cloudagent.wallet.routes.GetTagPolicyResultSchema \(class in aries\\_cloudagent.wallet.routes\), 124](#)  
[opts \(aries\\_cloudagent.wallet.routes.SetTagPolicyRequestSchema \(class in aries\\_cloudagent.wallet.routes\), 124](#)  
[ordered \(aries\\_cloudagent.messaging.models.base.BaseModelSchema \(class in aries\\_cloudagent.messaging.models\), 58](#)  
[out\\_time \(aries\\_cloudagent.messaging.decorators.timing\\_decorator \(class in aries\\_cloudagent.messaging.decorators\), 56](#)  
[outbound\\_message\\_router \(\) \(aries\\_cloudagent.core.conductor.Conductor \(class in aries\\_cloudagent.core\), 19](#)  
[OutboundDeliveryError, 92](#)  
[OutboundMessage \(class in aries\\_cloudagent.transport.outbound.message\), 96](#)  
[OutboundTransportError, 92](#)  
[OutboundTransportManager \(class in aries\\_cloudagent.transport.outbound.manager\), 93](#)  
[OutboundTransportRegistrationError, 93](#)  
**P**  
[pack \(\) \(aries\\_cloudagent.transport.pack\\_format.PackWireFormat \(class in aries\\_cloudagent.transport.pack\\_format\), 99](#)  
[pack\\_message \(\) \(aries\\_cloudagent.wallet.base.BaseWallet \(class in aries\\_cloudagent.wallet\), 111](#)  
[pack\\_message \(\) \(aries\\_cloudagent.wallet.basic.BasicWallet \(class in aries\\_cloudagent.wallet\), 113](#)  
[pack\\_message \(\) \(aries\\_cloudagent.wallet.indy.IndyWallet \(class in aries\\_cloudagent.wallet\), 121](#)  
[PackMessageSchema \(class in aries\\_cloudagent.wallet.crypto\), 115](#)

attribute), 75  
 PATTERN (aries\_cloudagent.messaging.valid.SHA256Hash attribute), 75  
 PATTERN (aries\_cloudagent.messaging.valid.UUIDFour attribute), 75  
 PendingTask (class in aries\_cloudagent.utils.task\_queue), 106  
 perform\_encode() (aries\_cloudagent.transport.outbound.manager.OutboundTransportManager method), 95  
 PleaseAckDecorator (class in aries\_cloudagent.messaging.decorators.please\_ack\_decorator), 50  
 PleaseAckDecorator.Meta (class in aries\_cloudagent.messaging.decorators.please\_ack\_decorator), 51  
 PleaseAckDecoratorSchema (class in aries\_cloudagent.messaging.decorators.please\_ack\_decorator), 51  
 PleaseAckDecoratorSchema.Meta (class in aries\_cloudagent.messaging.decorators.please\_ack\_decorator), 51  
 plugin\_names (aries\_cloudagent.core.plugin\_registry.PluginRegistry attribute), 22  
 PluginRegistry (class in aries\_cloudagent.core.plugin\_registry), 22  
 plugins (aries\_cloudagent.core.plugin\_registry.PluginRegistry attribute), 22  
 plugins\_handler() (aries\_cloudagent.admin.server.AdminServer method), 5  
 populate\_decorators() (aries\_cloudagent.messaging.agent\_message.AgentMessageSchema method), 68  
 port (aries\_cloudagent.transport.inbound.base.InboundTransportConfiguration attribute), 84  
 post\_save() (aries\_cloudagent.connections.models.connection\_record.ConnectionRecord method), 139  
 post\_save() (aries\_cloudagent.messaging.models.base\_record.BaseRecord method), 61  
 posture (aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.presentation\_preview.PresAttrSpec attribute), 173  
 pre\_verify() (aries\_cloudagent.verifier.indy.IndyVerifier static method), 108  
 predicate (aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.presentation\_preview.PresAttrSpec attribute), 174  
 predicates (aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.presentation\_preview.PresentationPreviewSchema attribute), 175  
 prefix (aries\_cloudagent.messaging.decorators.base.BaseDecorator attribute), 48  
 prefix\_tag\_filter() (aries\_cloudagent.messaging.models.base\_record.BaseRecord class method), 61  
 prepare\_disclosed() (aries\_cloudagent.core.protocol\_registry.ProtocolRegistry method), 23  
 prepare\_pack\_recipient\_keys() (in module aries\_cloudagent.wallet.crypto), 117  
 prepare\_send() (aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.issue\_credential\_request.PrepareSend attribute), 162  
 PresAttrSpec (class in aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.presentation\_preview.PresAttrSpec), 173  
 PresAttrSpec.Meta (class in aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.presentation\_preview.PresAttrSpec), 173  
 PresAttrSpec.Posture (class in aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.presentation\_preview.PresAttrSpec), 173  
 PresAttrSpecSchema (class in aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.presentation\_preview.PresAttrSpec), 173  
 PresAttrSpecSchema.Meta (class in aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.presentation\_preview.PresAttrSpec), 173  
 presentation (aries\_cloudagent.protocols.present\_proof.v1\_0.models.presentation.Presentation attribute), 182  
 Presentation (class in aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation.Presentation), 175  
 Presentation.Meta (class in aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation.Presentation), 176  
 presentation\_exchange\_create\_request() (in module aries\_cloudagent.protocols.present\_proof.v1\_0.routes), 188  
 presentation\_exchange\_credentials\_list() (in module aries\_cloudagent.protocols.present\_proof.v1\_0.routes), 189  
 presentation\_exchange\_id (aries\_cloudagent.protocols.present\_proof.v1\_0.models.presentation.Presentation attribute), 181  
 presentation\_exchange\_id (aries\_cloudagent.protocols.present\_proof.v1\_0.models.presentation.Presentation attribute), 181  
 presentation\_exchange\_list() (in module aries\_cloudagent.protocols.present\_proof.v1\_0.routes), 189  
 presentation\_exchange\_retrieve() (in module aries\_cloudagent.protocols.present\_proof.v1\_0.routes), 189  
 presentation\_exchange\_send\_bound\_request() (in module aries\_cloudagent.protocols.present\_proof.v1\_0.routes), 189  
 presentation\_exchange\_send\_free\_request() (in module aries\_cloudagent.protocols.present\_proof.v1\_0.routes), 189

190  
presentation\_exchange\_send\_presentation() 177  
(in module aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation 177  
190  
presentation\_exchange\_send\_proposal() 177  
(in module aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation 177  
190  
presentation\_exchange\_verify\_presentation() 171  
(in module aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation 171  
190  
presentation\_proposal 178  
(aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation.proposal.PresentationProposalSchema  
attribute), 178  
presentation\_proposal\_dict 178  
(aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation.proposal.PresentationProposalSchema  
attribute), 182  
presentation\_request 178  
(aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation.request.PresentationRequestSchema  
attribute), 182  
PresentationAck (class in 178  
aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation.ack.PresentationAck  
176  
PresentationAck.Meta (class in 171  
aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation.ack.PresentationAck  
177  
PresentationAckHandler (class in 179  
aries\_cloudagent.protocols.present\_proof.v1\_0.handlers.presentation.ack.PresentationAckHandler  
170  
PresentationAckSchema (class in 179  
aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation.ack.PresentationAck  
177  
PresentationAckSchema.Meta (class in 176  
aries\_cloudagent.protocols.present\_proof.v1\_0.messages.presentation.ack.PresentationAck  
177  
PresentationHandler (class in 176  
aries\_cloudagent.protocols.present\_proof.v1\_0.handlers.presentation.PresentationHandler  
171  
PresentationManager (class in 176  
aries\_cloudagent.protocols.present\_proof.v1\_0.manager.PresentationManager  
182  
PresentationManagerError, 184 173  
PresentationPreview (class in PresPredSpec.Meta (class in  
aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.presentation.preview.PresentationPreview  
174  
PresentationPreview.Meta (class in PresPredSpecSchema (class in  
aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.presentation.preview.PresentationPreview  
174  
PresentationPreviewSchema (class in PresPredSpecSchema.Meta (class in  
aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.presentation.preview.PresentationPreview  
175  
PresentationPreviewSchema.Meta (class in PreVerifyResult (class in  
aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inner.presentation.preview.PresentationPreview  
175  
PresentationProposal (class in class method), 16

[priority \(aries\\_cloudagent.connections.models.diddoc.Service attribute\), 130](#)  
[priority \(aries\\_cloudagent.connections.models.diddoc.Service attribute\), 134](#)  
[process\\_inbound\(\) \(aries\\_cloudagent.transport.inbound.session.InboundSession method\), 91](#)  
[process\\_queued\(\) \(aries\\_cloudagent.transport.outbound.manager.OutboundTransportManager method\), 95](#)  
[process\\_undelivered\(\) \(aries\\_cloudagent.transport.inbound.manager.InboundTransportManager method\), 87](#)  
[protected \(aries\\_cloudagent.messaging.decorators.attach\\_decorator.AttachDecoratorDataTWSSubject attribute\), 45](#)  
[protected \(aries\\_cloudagent.messaging.decorators.attach\\_decorator.AttachDecoratorDataTWSSchema attribute\), 47](#)  
[ProtocolDefinitionValidationError, 22](#)  
[ProtocolGroup \(class in put\(\) \(aries\\_cloudagent.config.argparse\), 10](#)  
[ProtocolMinorVersionNotSupported, 22](#)  
[ProtocolRegistry \(class in put\\_task\(\) \(aries\\_cloudagent.core.protocol\\_registry\), 23](#)  
**Q**  
[protocols \(aries\\_cloudagent.core.protocol\\_registry.ProtocolRegistry attribute\), 23](#)  
[protocols\\_matching\\_query\(\) \(aries\\_cloudagent.core.protocol\\_registry.ProtocolRegistry method\), 23](#)  
[provide\(\) \(aries\\_cloudagent.config.base.BaseProvider method\), 11](#)  
[provide\(\) \(aries\\_cloudagent.config.provider.CachedProvider method\), 16](#)  
[provide\(\) \(aries\\_cloudagent.config.provider.ClassProvider method\), 17](#)  
[provide\(\) \(aries\\_cloudagent.config.provider.InstanceProvider method\), 17](#)  
[provide\(\) \(aries\\_cloudagent.config.provider.StatsProvider method\), 17](#)  
[provide\(\) \(aries\\_cloudagent.ledger.provider.LedgerProvider method\), 37](#)  
**R**  
[provide\(\) \(aries\\_cloudagent.storage.provider.StorageProvider method\), 82](#)  
[provide\(\) \(aries\\_cloudagent.wallet.provider.WalletProvider method\), 123](#)  
[ProviderError, 12](#)  
[provision\(\) \(in module aries\\_cloudagent.commands.provision\), 8](#)  
[ProvisionError, 8](#)  
[pthid \(aries\\_cloudagent.messaging.decorators.thread\\_decorator.ThreadDecorator attribute\), 54](#)  
[pthid \(aries\\_cloudagent.messaging.decorators.thread\\_decorator.ThreadDecoratorSchema attribute\), 54](#)  
[pubkey \(aries\\_cloudagent.connections.models.diddoc.DIDDoc attribute\), 128](#)  
[pubkey \(aries\\_cloudagent.connections.models.diddoc.DIDDoc attribute\), 131](#)  
[PublicService \(class in aries\\_cloudagent.connections.models.diddoc\), 129](#)  
[PublicKey \(class in aries\\_cloudagent.connections.models.diddoc.publickey\), 129](#)  
[PublicKeyType \(class in aries\\_cloudagent.connections.models.diddoc\), 129](#)  
[PublicKeyType \(class in aries\\_cloudagent.connections.models.diddoc\), 133](#)  
[public\\_key \(aries\\_cloudagent.protocols.issue\\_credential.v1\\_0.manager.CredentialManager method\), 162](#)  
[put\(\) \(aries\\_cloudagent.utils.task\\_queue.TaskQueue method\), 107](#)  
[put\\_task\(\) \(aries\\_cloudagent.core.dispatcher.Dispatcher method\), 20](#)  
**Q**  
[queue\(\) \(aries\\_cloudagent.messaging.models.base\\_record.BaseRecord class method\), 61](#)  
[queue\\_message\(\) \(aries\\_cloudagent.core.dispatcher.Dispatcher method\), 20](#)  
[queue\\_outbound\(\) \(aries\\_cloudagent.core.conductor.Conductor method\), 19](#)  
[queued\\_message\\_count \(aries\\_cloudagent.messaging.decorators.transport\\_decorator.TransportDecorator attribute\), 57](#)  
[QueuedMessage \(class in aries\\_cloudagent.transport.inbound.delivery\\_queue\), 85](#)  
[QueuedOutboundMessage \(class in aries\\_cloudagent.transport.outbound.manager\), 95](#)  
**R**  
[random\\_seed\(\) \(in module aries\\_cloudagent.wallet.crypto\), 118](#)  
[raw\\_credential \(aries\\_cloudagent.protocols.issue\\_credential.v1\\_0.manager.CredentialManager attribute\), 160](#)  
[raw\\_key\(\) \(in module aries\\_cloudagent.messaging.decorators.attach\\_decorator\), 48](#)  
[raw\\_message \(aries\\_cloudagent.transport.inbound.receipt.MessageReceipt attribute\), 89](#)  
[ready \(aries\\_cloudagent.utils.task\\_queue.TaskQueue attribute\), 107](#)  
[receive\(\) \(aries\\_cloudagent.transport.inbound.session.InboundSession method\), 91](#)



[receive\\_credential\(\)](#) (aries\_cloudagent.protocols.issue\_credential.v1\_0.manager.CredentialManager method), 162  
[receive\\_credential\\_ack\(\)](#) (aries\_cloudagent.protocols.issue\_credential.v1\_0.manager.CredentialManager method), 162  
[receive\\_inbound\(\)](#) (aries\_cloudagent.transport.inbound.session.InboundSession attribute), 91  
[receive\\_offer\(\)](#) (aries\_cloudagent.protocols.issue\_credential.v1\_0.manager.CredentialManager method), 162  
[receive\\_presentation\(\)](#) (aries\_cloudagent.protocols.present\_proof.v1\_0.manager.PresentationManager method), 184  
[receive\\_presentation\\_ack\(\)](#) (aries\_cloudagent.protocols.present\_proof.v1\_0.manager.PresentationManager method), 184  
[receive\\_proposal\(\)](#) (aries\_cloudagent.protocols.issue\_credential.v1\_0.manager.CredentialManager method), 162  
[receive\\_proposal\(\)](#) (aries\_cloudagent.protocols.present\_proof.v1\_0.manager.PresentationManager method), 184  
[receive\\_request\(\)](#) (aries\_cloudagent.protocols.issue\_credential.v1\_0.manager.CredentialManager method), 162  
[receive\\_request\(\)](#) (aries\_cloudagent.protocols.present\_proof.v1\_0.manager.PresentationManager method), 184  
[received\\_orders](#) (aries\_cloudagent.messaging.decorators.thread\_decorator.ThreadDecorator attribute), 54  
[received\\_orders](#) (aries\_cloudagent.messaging.decorators.thread\_decorator.ThreadDecoratorSchema attribute), 54  
[recip\\_keys](#) (aries\_cloudagent.connections.models.diddoc.Service attribute), 87  
[recip\\_keys](#) (aries\_cloudagent.connections.models.diddoc.service.Service attribute), 95  
[recipient\\_id](#) (aries\_cloudagent.transport.inbound.receipt.MessageReceipt attribute), 89  
[recipient\\_id\\_public](#) (aries\_cloudagent.transport.inbound.receipt.MessageReceipt attribute), 89  
[recipient\\_keys](#) (aries\_cloudagent.connections.models.connection\_record.ConnectionRecord attribute), 142  
[recipient\\_verkey](#) (aries\_cloudagent.transport.inbound.receipt.MessageReceipt attribute), 89  
[RECORD\\_ID\\_NAME](#) (aries\_cloudagent.connections.models.connection\_record.ConnectionRecord attribute), 138  
[RECORD\\_ID\\_NAME](#) (aries\_cloudagent.messaging.models.base\_record.BaseRecord attribute), 60  
[RECORD\\_ID\\_NAME](#) (aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange attribute), 159  
[RECORD\\_ID\\_NAME](#) (aries\_cloudagent.protocols.present\_proof.v1\_0.models.presentation\_exchange.V10PresentationExchange attribute), 181  
[record\\_tags](#) (aries\_cloudagent.messaging.models.base\_record.BaseRecord attribute), 138  
[RECORD\\_TYPE](#) (aries\_cloudagent.connections.models.connection\_record.ConnectionRecord attribute), 60  
[RECORD\\_TYPE](#) (aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange attribute), 159  
[RECORD\\_TYPE](#) (aries\_cloudagent.protocols.present\_proof.v1\_0.models.presentation\_exchange.V10PresentationExchange attribute), 181  
[RECORD\\_TYPE\\_INVITATION](#) (aries\_cloudagent.connections.models.connection\_record.ConnectionRecord attribute), 60  
[RECORD\\_TYPE\\_MIME\\_TYPES](#) (aries\_cloudagent.holder.indy.IndyHolder attribute), 181  
[RECORD\\_TYPE\\_REQUEST](#) (aries\_cloudagent.connections.models.connection\_record.ConnectionRecord attribute), 139  
[record\\_value](#) (aries\_cloudagent.connections.models.connection\_record.ConnectionRecord attribute), 139  
[record\\_value](#) (aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange attribute), 159  
[record\\_value](#) (aries\_cloudagent.protocols.present\_proof.v1\_0.models.presentation\_exchange.V10PresentationExchange attribute), 181  
[register\(\)](#) (aries\_cloudagent.admin.server.AdminServer method), 184  
[register\(\)](#) (aries\_cloudagent.transport.inbound.manager.InboundTransport attribute), 87  
[register\(\)](#) (aries\_cloudagent.transport.outbound.manager.OutboundTransport attribute), 95  
[register\(\)](#) (in module aries\_cloudagent.ledger.routes), 39  
[register\(\)](#) (in module aries\_cloudagent.messaging.credential\_definitions.routes), 42  
[register\(\)](#) (in module aries\_cloudagent.connections.targets.schema.routes), 65  
[register\(\)](#) (in module aries\_cloudagent.protocols.issue\_credential.v1\_0.routes), 170  
[register\(\)](#) (in module aries\_cloudagent.protocols.present\_proof.v1\_0.routes), 190  
[register\(\)](#) (in module aries\_cloudagent.core.plugin\_registry.PluginRegistry), 124



Index 231







`method`), 131  
`set_cached_key()` (`aries_cloudagent.messaging.models.base.BaseRecord` attribute), 62  
`set_credential_definition_tag_policy()` (`aries_cloudagent.wallet.indy.IndyWallet` attribute), 122  
`set_default()` (`aries_cloudagent.config.settings.Settings` attribute), 17  
`set_public_did()` (`aries_cloudagent.wallet.base.BaseWallet` attribute), 111  
`set_response()` (`aries_cloudagent.transport.inbound.session.InboundSession` attribute), 91  
`set_result()` (`aries_cloudagent.cache.base.CacheKeyLock` attribute), 7  
`set_signature()` (`aries_cloudagent.messaging.agent_message.AgentMessage` attribute), 67  
`set_urlsaf_b64()` (in module `aries_cloudagent.wallet.util`), 125  
`set_value()` (`aries_cloudagent.config.settings.Settings` attribute), 18  
`SetTagPolicyRequestSchema` (class in `aries_cloudagent.wallet.routes`), 124  
`settings` (`aries_cloudagent.config.injection_context.InjectionContext` attribute), 14  
`settings` (`aries_cloudagent.config.injector.Injector` attribute), 15  
`Settings` (class in `aries_cloudagent.config.settings`), 17  
`SettingsError`, 12  
`setup()` (`aries_cloudagent.core.conductor.Conductor` attribute), 19  
`setup()` (`aries_cloudagent.core.dispatcher.Dispatcher` attribute), 21  
`setup()` (`aries_cloudagent.transport.inbound.manager.InboundTransportManager` attribute), 87  
`setup()` (`aries_cloudagent.transport.outbound.manager.OutboundTransportManager` attribute), 95  
`sha256` (`aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator` attribute), 44  
`sha256_` (`aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator` attribute), 47  
`SHA256Hash` (class in `aries_cloudagent.messaging.valid`), 75  
`shutdown_app()` (in module `aries_cloudagent.commands.start`), 8  
`sig_data` (`aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator` attribute), 53  
`sign()` (`aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator` attribute), 45  
`sign_field()` (`aries_cloudagent.messaging.agent_message.AgentMessage` attribute), 67  
`sign_message()` (`aries_cloudagent.wallet.base.BaseWallet` attribute), 111  
`sign_message()` (`aries_cloudagent.wallet.basic.BasicWallet` attribute), 134  
`method`), 114  
`sign_message()` (in module `aries_cloudagent.wallet.crypto`), 118  
`sign_pk_from_sk()` (in module `aries_cloudagent.wallet.crypto`), 118  
`signature` (`aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator` attribute), 45  
`signature` (`aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator` attribute), 47  
`signature` (`aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator` attribute), 53  
`signature_type` (`aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator` attribute), 51  
`SignatureDecorator` (class in `aries_cloudagent.messaging.decorators.signature_decorator`), 51  
`SignatureDecorator.Meta` (class in `aries_cloudagent.messaging.decorators.signature_decorator`), 52  
`SignatureDecoratorSchema` (class in `aries_cloudagent.messaging.decorators.signature_decorator`), 52  
`SignatureDecoratorSchema.Meta` (class in `aries_cloudagent.messaging.decorators.signature_decorator`), 53  
`signatures` (`aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator` attribute), 45  
`signatures` (`aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator` attribute), 47  
`signed` (`aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator` attribute), 45  
`signed` (`aries_cloudagent.messaging.agent_message.AgentMessage` attribute), 68  
`signature_decorator` (`aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator` attribute), 53  
`signature_decorator` (`aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator` attribute), 58  
`signature_decorator` (`aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator` attribute), 58  
`socket_connect_start()` (`aries_cloudagent.transport.stats.StatsTracer` attribute), 100  
`specification()` (`aries_cloudagent.connections.models.diddoc.PublicKey` attribute), 128  
`specification()` (`aries_cloudagent.connections.models.diddoc.PublicKey` attribute), 128  
`specifier` (`aries_cloudagent.connections.models.diddoc.LinkedDataKey` attribute), 128  
`specifier` (`aries_cloudagent.connections.models.diddoc.publickey.PublicKey` attribute), 132  
`specifier` (`aries_cloudagent.connections.models.diddoc.publickey.PublicKey` attribute), 134



STATE\_REQUEST\_SENT (aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange attribute), 159  
STATE\_REQUEST\_SENT (aries\_cloudagent.protocols.present\_proof.v1\_0.models.presentation\_exchange.V10PresentationExchange attribute), 181  
STATE\_RESPONSE (aries\_cloudagent.connections.models.connection\_record.aries\_cloudagent.connections.models.connection\_record.StorageRecord attribute), 138  
STATE\_RETRY (aries\_cloudagent.transport.outbound.manager.QueueOutboundMessage.aries\_cloudagent.transport.outbound.manager.QueueOutboundMessage attribute), 96  
STATE\_VERIFIED (aries\_cloudagent.protocols.present\_proof.v1\_0.models.presentation\_exchange.V10PresentationExchange attribute), 181  
Stats (class in aries\_cloudagent.utils.stats), 105  
StatsProvider (class in aries\_cloudagent.config.provider), 17  
StatsTracer (class in aries\_cloudagent.transport.stats), 100  
status (aries\_cloudagent.messaging.ack.message.AckSchema attribute), 40  
status\_handler() (aries\_cloudagent.admin.server.AdminServer.aries\_cloudagent.admin.server.AdminServer method), 5  
status\_reset\_handler() (aries\_cloudagent.admin.server.AdminServer.aries\_cloudagent.admin.server.AdminServer method), 5  
stop() (aries\_cloudagent.admin.base\_server.BaseAdminServer.aries\_cloudagent.admin.base\_server.BaseAdminServer method), 3  
stop() (aries\_cloudagent.admin.server.AdminServer.aries\_cloudagent.admin.server.AdminServer method), 5  
stop() (aries\_cloudagent.core.conductor.Conductor.aries\_cloudagent.core.conductor.Conductor method), 19  
stop() (aries\_cloudagent.transport.inbound.base.BaseInboundTransport.aries\_cloudagent.transport.inbound.base.BaseInboundTransport method), 83  
stop() (aries\_cloudagent.transport.inbound.http.HttpTransport.aries\_cloudagent.transport.inbound.http.HttpTransport method), 86  
stop() (aries\_cloudagent.transport.inbound.manager.InboundTransportManager.aries\_cloudagent.transport.inbound.manager.InboundTransportManager method), 87  
stop() (aries\_cloudagent.transport.inbound.ws.WsTransport.aries\_cloudagent.transport.inbound.ws.WsTransport method), 92  
stop() (aries\_cloudagent.transport.outbound.base.BaseOutboundTransport.aries\_cloudagent.transport.outbound.base.BaseOutboundTransport method), 92  
stop() (aries\_cloudagent.transport.outbound.http.HttpTransport.aries\_cloudagent.transport.outbound.http.HttpTransport method), 93  
stop() (aries\_cloudagent.transport.outbound.manager.OutboundTransportManager.aries\_cloudagent.transport.outbound.manager.OutboundTransportManager method), 95  
stop() (aries\_cloudagent.transport.outbound.ws.WsTransport.aries\_cloudagent.transport.outbound.ws.WsTransport method), 97  
stop() (aries\_cloudagent.transport.queue.base.BaseMessageQueue.aries\_cloudagent.transport.queue.base.BaseMessageQueue method), 97  
stop() (aries\_cloudagent.transport.queue.basic.BasicMessageQueue.aries\_cloudagent.transport.queue.basic.BasicMessageQueue method), 98  
stop() (aries\_cloudagent.utils.stats.Timer.aries\_cloudagent.utils.stats.Timer method), 106  
storage\_record (aries\_cloudagent.messaging.models.base\_record.BaseRecord attribute), 62  
STORAGE\_TYPES (aries\_cloudagent.storage.provider.StorageProvider.aries\_cloudagent.storage.provider.StorageProvider attribute), 82  
StorageDuplicateError, 79  
StorageError, 80  
StorageExchange (aries\_cloudagent.storage.provider.StorageProvider.aries\_cloudagent.storage.provider.StorageProvider attribute), 82  
StorageProvider (class in aries\_cloudagent.storage.provider.StorageProvider.aries\_cloudagent.storage.provider.StorageProvider attribute), 82  
StorageRecord (class in aries\_cloudagent.storage.provider.StorageProvider.aries\_cloudagent.storage.provider.StorageProvider attribute), 82  
StorageSearchError, 80  
StorageSearchError (aries\_cloudagent.storage.provider.StorageProvider.aries\_cloudagent.storage.provider.StorageProvider attribute), 77  
store\_credential() (aries\_cloudagent.holder.base.BaseHolder.aries\_cloudagent.holder.base.BaseHolder method), 25  
store\_credential() (aries\_cloudagent.holder.indy.IndyHolder.aries\_cloudagent.holder.indy.IndyHolder method), 27  
store\_credential() (aries\_cloudagent.holder.indy.IndyHolder.aries\_cloudagent.holder.indy.IndyHolder method), 27  
str\_to\_b64() (in aries\_cloudagent.wallet.util), 126  
str\_to\_datetime() (in aries\_cloudagent.wallet.util), 126  
str\_to\_epoch() (in aries\_cloudagent.wallet.util), 126  
strip\_tag\_prefix() (aries\_cloudagent.messaging.models.base\_record.BaseRecord class method), 62  
TAAAcceptanceSchema (class in aries\_cloudagent.ledger.routes), 37  
TAAAcceptSchema (class in aries\_cloudagent.ledger.routes), 37  
TAAInfoSchema (class in aries\_cloudagent.ledger.routes), 38  
TAARecordSchema (class in aries\_cloudagent.ledger.routes), 38  
TAAResultSchema (class in aries\_cloudagent.ledger.routes), 38  
TAG\_NAMES (aries\_cloudagent.connections.models.connection\_record.aries\_cloudagent.connections.models.connection\_record attribute), 138  
TAG\_NAMES (aries\_cloudagent.messaging.models.base\_record.BaseRecord attribute), 62



TAG\_NAMES (aries\_cloudagent.protocols.issue\_credential.v1\_0.models.issue\_credential\_exchange.V10CredentialExchange  
attribute), 159 TimingDecorator (class in  
TAG\_NAMES (aries\_cloudagent.protocols.present\_proof.v1\_0.models.present\_proof\_exchange.V10PresentationExchange  
attribute), 181 54  
tag\_query (aries\_cloudagent.storage.base.BaseStorageRecordSchema.Meta (class in  
attribute), 77 aries\_cloudagent.messaging.decorators.timing\_decorator),  
tags (aries\_cloudagent.messaging.models.base\_record.BaseRecord 55  
attribute), 62 TimingDecoratorSchema (class in  
task (aries\_cloudagent.utils.task\_queue.PendingTask aries\_cloudagent.messaging.decorators.timing\_decorator),  
attribute), 106 55  
task\_done () (aries\_cloudagent.transport.queue.base.BaseMessageQueueDecoratorSchema.Meta (class in  
method), 98 aries\_cloudagent.messaging.decorators.timing\_decorator),  
task\_done () (aries\_cloudagent.transport.queue.basic.BasicMessageQueue 55  
method), 98 to\_dict () (aries\_cloudagent.connections.models.diddoc.PublicKey  
task\_exc\_info () (in module method), 129  
aries\_cloudagent.utils.task\_queue), 108 to\_dict () (aries\_cloudagent.connections.models.diddoc.publickey.PublicKey  
TaskQueue (class in method), 133  
aries\_cloudagent.utils.task\_queue), 106 to\_dict () (aries\_cloudagent.connections.models.diddoc.Service  
their\_did (aries\_cloudagent.connections.models.connection\_record.ConnectionRecordSchema  
attribute), 140 to\_dict () (aries\_cloudagent.connections.models.diddoc.service.Service  
their\_label (aries\_cloudagent.connections.models.connection\_record.ConnectionRecordSchema  
attribute), 140 to\_dict () (aries\_cloudagent.messaging.decorators.base.BaseDecorator  
their\_role (aries\_cloudagent.connections.models.connection\_record.ConnectionRecordSchema  
attribute), 140 to\_json () (aries\_cloudagent.connections.models.diddoc.DIDDoc  
thid (aries\_cloudagent.messaging.decorators.thread\_decorator.ThreadDecorator 128  
attribute), 54 to\_json () (aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc  
thid (aries\_cloudagent.messaging.decorators.thread\_decorator.ThreadDecorator 62  
attribute), 54 to\_json () (aries\_cloudagent.messaging.models.base.BaseModel  
thread\_id (aries\_cloudagent.protocols.issue\_credential.v1\_0.models.issue\_credential\_exchange.V10CredentialExchangeSchema  
attribute), 160 topic\_filter (aries\_cloudagent.admin.server.WebhookTarget  
thread\_id (aries\_cloudagent.protocols.present\_proof.v1\_0.models.present\_proof\_exchange.V10PresentationExchangeSchema  
attribute), 182 trace (aries\_cloudagent.messaging.models.base\_record.BaseExchangeSchema  
thread\_id (aries\_cloudagent.transport.inbound.receipt.MessageReceipt 59  
attribute), 89 TransportDecorator (class in  
ThreadDecorator (class in aries\_cloudagent.messaging.decorators.transport\_decorator),  
aries\_cloudagent.messaging.decorators.thread\_decorator), 56  
53 TransportDecorator.Meta (class in  
ThreadDecorator.Meta (class in aries\_cloudagent.messaging.decorators.transport\_decorator),  
aries\_cloudagent.messaging.decorators.thread\_decorator), 56  
53 TransportDecoratorSchema (class in  
ThreadDecoratorSchema (class in aries\_cloudagent.messaging.decorators.transport\_decorator),  
aries\_cloudagent.messaging.decorators.thread\_decorator), 56  
54 TransportDecoratorSchema.Meta (class in  
ThreadDecoratorSchema.Meta (class in aries\_cloudagent.messaging.decorators.transport\_decorator),  
aries\_cloudagent.messaging.decorators.thread\_decorator), 56  
54 TransportError, 99  
threshold (aries\_cloudagent.protocols.present\_proof.v1\_0.messages.inbound.presentation\_previdusPresPredSpecSchema  
attribute), 174 aries\_cloudagent.config argparse), 10  
time\_now () (in module type (aries\_cloudagent.connections.models.diddoc.PublicKey  
aries\_cloudagent.messaging.util), 72 attribute), 129  
timeout () (aries\_cloudagent.utils.repeat.RepeatAttempt type (aries\_cloudagent.connections.models.diddoc.publickey.PublicKey  
method), 104 attribute), 133  
Timer (class in aries\_cloudagent.utils.stats), 105 type (aries\_cloudagent.connections.models.diddoc.Service  
timer () (aries\_cloudagent.utils.stats.Collector attribute), 130



[in aries\\_cloudagent.protocols.present\\_proof.v1\\_0.routes\), method\), 52](#)  
[187](#)  
[verify\\_message\(\) \(aries\\_cloudagent.wallet.base.BaseWallet](#)  
[V10PresentationExchangeSchema \(class in method\), 111](#)  
[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.models.presentation\\_exchange\),](#)  
[181](#)  
[method\), 114](#)  
[V10PresentationExchangeSchema.Meta \(class verify\\_message\(\) \(aries\\_cloudagent.wallet.indy.IndyWallet](#)  
[in aries\\_cloudagent.protocols.present\\_proof.v1\\_0.models.presentation\\_exchange\),](#)  
[181](#)  
[verify\\_presentation\(\)](#)  
[V10PresentationProposalRequestSchema \(aries\\_cloudagent.protocols.present\\_proof.v1\\_0.manager.Presentation](#)  
[\(class in aries\\_cloudagent.protocols.present\\_proof.v1\\_0.routes\) method\), 184](#)  
[187](#)  
[verify\\_presentation\(\)](#)  
[V10PresentationRequestRequestSchema \(aries\\_cloudagent.verifier.base.BaseVerifier](#)  
[\(class in aries\\_cloudagent.protocols.present\\_proof.v1\\_0.routes\) method\), 108](#)  
[188](#)  
[verify\\_presentation\(\)](#)  
[V10PresentationRequestSchema \(class in \(aries\\_cloudagent.verifier.indy.IndyVerifier](#)  
[aries\\_cloudagent.protocols.present\\_proof.v1\\_0.routes\), method\), 109](#)  
[188](#)  
[verify\\_signatures\(\)](#)  
[V10PublishRevocationsResultSchema \(class \(aries\\_cloudagent.messaging.agent\\_message.AgentMessage](#)  
[in aries\\_cloudagent.protocols.issue\\_credential.v1\\_0.routes\) method\), 67](#)  
[166](#)  
[verify\\_signed\\_field\(\)](#)  
[validate\\_data\\_spec\(\) \(aries\\_cloudagent.messaging.agent\\_message.AgentMessage](#)  
[\(aries\\_cloudagent.messaging.decorators.attach\\_decorator.AttachDecoratorDataSchema](#)  
[method\), 47](#)  
[verify\\_signed\\_message\(\) \(in module](#)  
[validate\\_seed\(\) \(in module aries\\_cloudagent.wallet.crypto\), 118](#)  
[aries\\_cloudagent.wallet.crypto\), 118](#)  
[verkey \(aries\\_cloudagent.wallet.base.DIDInfo at-](#)  
[tribute\), 112](#)  
[validate\\_single\\_xor\\_multi\\_sig\(\) \(aries\\_cloudagent.messaging.decorators.attach\\_decorator.AttachDecoratorDataSchema.KeyInfo at-](#)  
[method\), 47](#)  
[tribute\), 112](#)  
[validate\\_version\(\) \(aries\\_cloudagent.core.plugin\\_registry.PluginRegistry](#)  
[method\), 23](#)  
[wait\\_for\(\) \(aries\\_cloudagent.utils.task\\_queue.TaskQueue](#)  
[value \(aries\\_cloudagent.connections.models.diddoc.PublicKey method\), 107](#)  
[attribute\), 129](#)  
[wait\\_response\(\) \(aries\\_cloudagent.transport.inbound.session.Inbound](#)  
[value \(aries\\_cloudagent.connections.models.diddoc.publickey.PublicKey method\), 91](#)  
[attribute\), 133](#)  
[wait\\_until\\_time\(aries\\_cloudagent.messaging.decorators.timing\\_dec-](#)  
[value \(aries\\_cloudagent.messaging.models.base\\_record.BaseRecord attribute\), 56](#)  
[attribute\), 62](#)  
[wallet \(aries\\_cloudagent.storage.indy.IndyStorage at-](#)  
[value \(aries\\_cloudagent.protocols.issue\\_credential.v1\\_0.messages.inner\\_credential\\_preview.CredAttrSpecSchema](#)  
[attribute\), 150](#)  
[wallet\\_config\(\) \(in module](#)  
[value \(aries\\_cloudagent.protocols.present\\_proof.v1\\_0.messages.inner\\_presentation\\_preview.PresentationPreviewSpecSchema](#)  
[attribute\), 173](#)  
[aries\\_cloudagent.wallet.routes\), 124](#)  
[wallet\\_create\\_did\(\) \(in module](#)  
[ver\\_type\(aries\\_cloudagent.connections.models.diddoc.LinkedDataKeySpec](#)  
[attribute\), 129](#)  
[aries\\_cloudagent.wallet.routes\), 124](#)  
[wallet\\_did\\_list\(\) \(in module](#)  
[ver\\_type\(aries\\_cloudagent.connections.models.diddoc.publickey.LinkedDataKeySpec](#)  
[attribute\), 132](#)  
[aries\\_cloudagent.wallet.routes\), 125](#)  
[wallet\\_get\\_public\\_did\(\) \(in module](#)  
[ver\\_type\(aries\\_cloudagent.connections.models.diddoc.publickey.PublicKeyType](#)  
[attribute\), 134](#)  
[aries\\_cloudagent.wallet.routes\), 125](#)  
[wallet\\_get\\_tagging\\_policy\(\) \(in module](#)  
[ver\\_type\(aries\\_cloudagent.connections.models.diddoc.PublicKeyType](#)  
[attribute\), 130](#)  
[aries\\_cloudagent.wallet.routes\), 125](#)  
[wallet\\_set\\_public\\_did\(\) \(in module](#)  
[verified\(aries\\_cloudagent.protocols.present\\_proof.v1\\_0.models.presentation\\_exchange.V10PresentationExchangeSchema](#)  
[attribute\), 182](#)  
[wallet\\_set\\_tagging\\_policy\(\) \(in module](#)  
[verify\(\) \(aries\\_cloudagent.messaging.decorators.attach\\_decorator.AttachDecoratorData](#)  
[method\), 45](#)  
[aries\\_cloudagent.wallet.routes\), 125](#)  
[WALLET\\_TYPE\(aries\\_cloudagent.wallet.basic.BasicWallet](#)  
[verify\(\) \(aries\\_cloudagent.messaging.decorators.signature\\_decorator.SignatureDecorator](#)  
[attribute\), 112](#)

WALLET\_TYPE (*aries\_cloudagent.wallet.indy.IndyWallet*  
*attribute*), 119  
WALLET\_TYPES (*aries\_cloudagent.wallet.provider.WalletProvider*  
*attribute*), 123  
WalletDuplicateError, 118  
WalletError, 118  
WalletGroup (class in  
*aries\_cloudagent.config.argparse*), 10  
WalletNotFoundError, 119  
WalletProvider (class in  
*aries\_cloudagent.wallet.provider*), 123  
webhook\_payload (*aries\_cloudagent.messaging.models.base\_record.BaseRecord*  
*attribute*), 62  
webhook\_router () (*aries\_cloudagent.core.conductor.Conductor*  
*method*), 19  
WEBHOOK\_TOPIC (*aries\_cloudagent.connections.models.connection\_record.ConnectionRecord*  
*attribute*), 138  
WEBHOOK\_TOPIC (*aries\_cloudagent.messaging.models.base\_record.BaseRecord*  
*attribute*), 60  
webhook\_topic (*aries\_cloudagent.messaging.models.base\_record.BaseRecord*  
*attribute*), 62  
WEBHOOK\_TOPIC (*aries\_cloudagent.protocols.issue\_credential.v1\_0.models.credential\_exchange.V10CredentialExchange*  
*attribute*), 159  
WEBHOOK\_TOPIC (*aries\_cloudagent.protocols.present\_proof.v1\_0.models.presentation\_exchange.V10PresentationExchange*  
*attribute*), 181  
WebhookTarget (class in  
*aries\_cloudagent.admin.server*), 5  
websocket\_handler ()  
(*aries\_cloudagent.admin.server.AdminServer*  
*method*), 5  
wire\_format (*aries\_cloudagent.transport.outbound.base.BaseOutboundTransport*  
*attribute*), 92  
WireFormatError, 99  
wrap () (*aries\_cloudagent.utils.stats.Collector* *method*),  
105  
wrap\_coro () (*aries\_cloudagent.utils.stats.Collector*  
*method*), 105  
wrap\_fn () (*aries\_cloudagent.utils.stats.Collector*  
*method*), 105  
wrap\_indy\_cred\_req ()  
(*aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_request.CredentialRequest*  
*class method*), 157  
wrap\_indy\_credential ()  
(*aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_issue.CredentialIssue*  
*class method*), 152  
wrap\_indy\_offer ()  
(*aries\_cloudagent.protocols.issue\_credential.v1\_0.messages.credential\_offer.CredentialOffer*  
*class method*), 154  
WsTransport (class in  
*aries\_cloudagent.transport.inbound.ws*),  
91  
WsTransport (class in  
*aries\_cloudagent.transport.outbound.ws*),  
96