

KnowARC Reference Manual

Generated by Doxygen 1.5.1

Fri Jul 4 12:01:54 2008

Contents

1	KnowARC Namespace Index	1
1.1	KnowARC Namespace List	1
2	KnowARC Hierarchical Index	3
2.1	KnowARC Class Hierarchy	3
3	KnowARC Class Index	5
3.1	KnowARC Class List	5
4	KnowARC Namespace Documentation	7
4.1	ArcSec Namespace Reference	7
5	KnowARC Class Documentation	11
5.1	ArcSec::AllowPDP Class Reference	11
5.2	ArcSec::ArcAlgFactory Class Reference	12
5.3	ArcSec::ArcAttributeFactory Class Reference	13
5.4	ArcSec::ArcAuthZ Class Reference	14
5.5	ArcSec::ArcEvaluator Class Reference	15
5.6	ArcSec::ArcFnFactory Class Reference	16
5.7	ArcSec::ArcPDP Class Reference	17
5.8	ArcSec::ArcPDPServiceInvoker Class Reference	18
5.9	ArcSec::ArcPolicy Class Reference	19
5.10	ArcSec::ArcRequestItem Class Reference	20
5.11	ArcSec::ArcRule Class Reference	21
5.12	ArcSec::CountPDP Class Reference	22
5.13	ArcSec::DelegationPDP Class Reference	23
5.14	ArcSec::DenyPDP Class Reference	24
5.15	Arc::LDAPQuery Class Reference	25
5.16	Arc::MCC_HTTP Class Reference	26
5.17	Arc::MCC_HTTP_Client Class Reference	27

5.18 Arc::MCC_HTTP_Service Class Reference	28
5.19 Arc::MCC_SOAP Class Reference	29
5.20 Arc::MCC_SOAP_Service Class Reference	30
5.21 Arc::MCC_TCP Class Reference	31
5.22 Arc::MCC_TCP_Client Class Reference	32
5.23 Arc::MCC_TCP_Service Class Reference	33
5.24 Arc::MCC_TLS Class Reference	34
5.25 Arc::MCC_TLS_Client Class Reference	36
5.26 Arc::MCC_TLS_Service Class Reference	37
5.27 Arc::PayloadHTTP Class Reference	38
5.28 Arc::PayloadTCPsocket Class Reference	42
5.29 Arc::PayloadTLSStream Class Reference	43
5.30 ArcSec::SimpleListPDP Class Reference	45
5.31 ArcSec::UsernameTokenSH Class Reference	46
5.32 ArcSec::XACMLPolicy Class Reference	47
5.33 ArcSec::XACMLRule Class Reference	48
5.34 ArcSec::XACMLTarget Class Reference	49

Chapter 1

KnowARC Namespace Index

1.1 KnowARC Namespace List

Here is a list of all documented namespaces with brief descriptions:

[ArcSec](#) (ArcRequest, Parsing the specified Arc request format) 7

Chapter 2

KnowARC Hierarchical Index

2.1 KnowARC Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

ArcSec::AllowPDP	11
ArcSec::ArcAlgFactory	12
ArcSec::ArcAttributeFactory	13
ArcSec::ArcAuthZ	14
ArcSec::ArcEvaluator	15
ArcSec::ArcFnFactory	16
ArcSec::ArcPDP	17
ArcSec::ArcPDPServiceInvoker	18
ArcSec::ArcPolicy	19
ArcSec::ArcRequestItem	20
ArcSec::ArcRule	21
ArcSec::CountPDP	22
ArcSec::DelegationPDP	23
ArcSec::DenyPDP	24
Arc::LDAPQuery	25
Arc::MCC_HTTP	26
Arc::MCC_HTTP_Client	27
Arc::MCC_HTTP_Service	28
Arc::MCC_SOAP	29
Arc::MCC_SOAP_Service	30
Arc::MCC_TCP	31
Arc::MCC_TCP_Client	32
Arc::MCC_TCP_Service	33
Arc::MCC_TLS	34
Arc::MCC_TLS_Client	36
Arc::MCC_TLS_Service	37
Arc::PayloadHTTP	38
Arc::PayloadTCPSocket	42
Arc::PayloadTLSStream	43
ArcSec::SimpleListPDP	45
ArcSec::UsernameTokenSH	46
ArcSec::XACMLPolicy	47

ArcSec::XACMLRule	48
ArcSec::XACMLTarget	49

Chapter 3

KnowARC Class Index

3.1 KnowARC Class List

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

ArcSec::AllowPDP (This PDP always return true (allow))	11
ArcSec::ArcAlgFactory (Algorithm factory class for Arc)	12
ArcSec::ArcAttributeFactory (Attribute factory class for Arc specified attributes)	13
ArcSec::ArcAuthZ (Tests message against list of PDPs)	14
ArcSec::ArcEvaluator (Execute the policy evaluation, based on the request and policy)	15
ArcSec::ArcFnFactory (Function factory class for Arc specified attributes)	16
ArcSec::ArcPDP (ArcPDP - PDP which can handle the Arc specific request and policy schema)	17
ArcSec::ArcPDPServiceInvoker (ArcPDPServiceInvoker - client which will invoke pdpservice)	18
ArcSec::ArcPolicy (ArcPolicy class to parse and operate Arc specific <Policy> node)	19
ArcSec::ArcRequestItem (Container, <Subjects, Actions, Objects, Contexts> tuple)	20
ArcSec::ArcRule (ArcRule class to parse Arc specific <Rule> node)	21
ArcSec::CountPDP (CountPDP - PDP which can handle the Arc specific request and policy schema)	22
ArcSec::DelegationPDP	23
ArcSec::DenyPDP (This PDP always returns false (deny))	24
Arc::LDAPQuery	25
Arc::MCC_HTTP (A base class for HTTP client and service MCCs)	26
Arc::MCC_HTTP_Client	27
Arc::MCC_HTTP_Service	28
Arc::MCC_SOAP (A base class for SOAP client and service MCCs)	29
Arc::MCC_SOAP_Service	30
Arc::MCC_TCP (A base class for TCP client and service MCCs)	31
Arc::MCC_TCP_Client	32
Arc::MCC_TCP_Service	33
Arc::MCC_TLS (A base class for SOAP client and service MCCs)	34
Arc::MCC_TLS_Client	36
Arc::MCC_TLS_Service	37
Arc::PayloadHTTP	38
Arc::PayloadTCPsocket	42
Arc::PayloadTLSStream	43
ArcSec::SimpleListPDP (Tests X509 subject against list of subjects in file)	45
ArcSec::UsernameTokenSH (Adds WS-Security Username Token into SOAP Header)	46

ArcSec::XACMLPolicy (XACMLPolicy class to parse and operate XACML specific <Policy> node)	47
ArcSec::XACMLRule (XACMLRule class to parse XACML specific <Rule> node)	48
ArcSec::XACMLTarget (XACMLTarget class to parse and operate XACML specific <Target> node)	49

Chapter 4

KnowARC Namespace Documentation

4.1 ArcSec Namespace Reference

ArcRequest, Parsing the specified Arc request format.

Classes

- class **DelegationSecAttr**
- class **DelegationMultiSecAttr**
- class **DelegationSH**
- class [AllowPDP](#)
This PDP always return true (allow).
- class [ArcAuthZ](#)
Tests message against list of PDPs.
- class [ArcAlgFactory](#)
Algorithm factory class for Arc.
- class [ArcAttributeFactory](#)
Attribute factory class for Arc specified attributes.
- class [ArcEvaluator](#)
Execute the policy evaluation, based on the request and policy.
- class [ArcFnFactory](#)
Function factory class for Arc specified attributes.
- class [ArcPDP](#)
[ArcPDP](#) - PDP which can handle the Arc specific request and policy schema.
- class [ArcPolicy](#)
[ArcPolicy](#) class to parse and operate Arc specific <Policy> node.
- class **ArcRequest**

- class [ArcRequestItem](#)
Container, <Subjects, Actions, Objects, Contexts> tuple.
- class [ArcRule](#)
[ArcRule](#) class to parse Arc specific <Rule> node.
- class **AttributeDesignator**
- class **AttributeSelector**
- class [CountPDP](#)
[CountPDP](#) - PDP which can handle the Arc specific request and policy schema.
- class [DelegationPDP](#)
- class [DenyPDP](#)
This PDP always returns false (deny).
- class [ArcPDPServiceInvoker](#)
[ArcPDPServiceInvoker](#) - client which will invoke pdpservice.
- class [SimpleListPDP](#)
Tests X509 subject against list of subjects in file.
- class [UsernameTokenSH](#)
Adds WS-Security Username Token into SOAP Header.
- class [XACMLPolicy](#)
[XACMLPolicy](#) class to parse and operate XACML specific <Policy> node.
- class **XACMLRequest**
- class [XACMLRule](#)
[XACMLRule](#) class to parse XACML specific <Rule> node.
- class **XACMLTargetMatch**
- class **XACMLTargetMatchGroup**
- class **XACMLTargetSection**
- class [XACMLTarget](#)
[XACMLTarget](#) class to parse and operate XACML specific <Target> node.

Typedefs

- typedef std::pair< AttributeValue *, Function * > [Match](#)
- typedef std::list< [Match](#) > [AndList](#)
- typedef std::list< [AndList](#) > [OrList](#)

Enumerations

- enum **Id_MatchResult** { **ID_MATCH** = 0, **ID_PARTIAL_MATCH** = 1, **ID_NO_MATCH** = 2 }

4.1.1 Detailed Description

ArcRequest, Parsing the specified Arc request format.

4.1.2 Typedef Documentation

4.1.2.1 typedef std::list<Match> ArcSec::AndList

AndList - include items inside one <Subject> (or <Resource> <Action> <Condition>).

"Or" relationship meand the request should satisfy any of the items <Subjects>
 <Subject type="X500DN"/>/O=Grid/OU=KnowARC/CN=ABC</Subject> <Subject
 type="VOMSAttribute"/>/vo.knowarc/usergroupA</Subject> <Subject> <SubFraction
 type="X500DN"/>/O=Grid/OU=KnowARC/CN=XYZ</SubFraction> <SubFraction
 type="ShibName"/>urn:mace:shibboleth:examples</SubFraction> </Subject> <GroupIdRef
 location="/subjectgroup.xml">subgrpexample1</GroupIdRef> </Subjects>

4.1.2.2 typedef std::pair<AttributeValue*, Function*> ArcSec::Match

pair Match include the AttributeValue object in <Rule> and the Function which is used to handle the AttributeValue, default function is "Equal", if some other function is used, it should be explicitly specified, e.g. Subject Type="string" Function="Match"/>/vo.knowarc/usergroupA</Subject><Subjects> example inside <Rule>: <Subjects> <Subject type="X500Name"/>/O=Nordu-Grid/OU=UIO/CN=test</Subject> <Subject type="string"/>/vo.knowarc/usergroupA</Subject> <Subject> <SubFraction type="string"/>/O=Grid/OU=KnowARC/CN=XYZ</SubFraction> <SubFraction type="string"/>urn:mace:shibboleth:examples</SubFraction> </Subject> <GroupIdRef location="/subjectgroup.xml">subgrpexample1</GroupIdRef> </Subjects>

4.1.2.3 typedef std::list<AndList> ArcSec::OrList

OrList - include items inside one <Subjects> (or <Resources> <Actions> <Conditions>).

Chapter 5

KnowARC Class Documentation

5.1 ArcSec::AllowPDP Class Reference

This PDP always return true (allow).

```
#include <AllowPDP.h>
```

Public Member Functions

- **AllowPDP** (Arc::Config *cfg)
- virtual bool **isPermitted** (Arc::Message *msg)

Static Public Member Functions

- static PDP * **get_allow_pdp** (Arc::Config *cfg, Arc::ChainContext *ctx)

5.1.1 Detailed Description

This PDP always return true (allow).

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

- AllowPDP.h

5.2 ArcSec::ArcAlgFactory Class Reference

Algorithm factory class for Arc.

```
#include <ArcAlgFactory.h>
```

Public Member Functions

- virtual CombiningAlg * [createAlg](#) (const std::string &type)

5.2.1 Detailed Description

Algorithm factory class for Arc.

5.2.2 Member Function Documentation

5.2.2.1 virtual CombiningAlg* ArcSec::ArcAlgFactory::createAlg (const std::string & *type*)
[virtual]

return a Alg object according to the "CombiningAlg" attribute in the <Policy> node; The [ArcAlgFactory](#) itself will release the Alg objects

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

- ArcAlgFactory.h

5.3 ArcSec::ArcAttributeFactory Class Reference

Attribute factory class for Arc specified attributes.

```
#include <ArcAttributeFactory.h>
```

Public Member Functions

- virtual AttributeValue * [createValue](#) (const Arc::XMLNode &node, const std::string &type)

5.3.1 Detailed Description

Attribute factory class for Arc specified attributes.

5.3.2 Member Function Documentation

5.3.2.1 `virtual AttributeValue* ArcSec::ArcAttributeFactory::createValue (const Arc::XMLNode & node, const std::string & type) [virtual]`

creat a AttributeValue according to the value in the XML node and the type; It should be the caller to release the AttributeValue Object

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

- ArcAttributeFactory.h

5.4 ArcSec::ArcAuthZ Class Reference

Tests message against list of PDPs.

```
#include <ArcAuthZ.h>
```

Public Member Functions

- **ArcAuthZ** (Arc::Config *cfg, Arc::ChainContext *ctx)
- virtual bool **Handle** (Arc::Message *msg)

Static Public Member Functions

- static SecHandler * **get_sechandler** (Arc::Config *cfg, Arc::ChainContext *ctx)

Protected Member Functions

- bool **MakePDPs** (Arc::Config *cfg)

Classes

- class **PDPDesc**

5.4.1 Detailed Description

Tests message against list of PDPs.

This class implements SecHandler interface. It's **Handle()** method runs provided Message instance against all PDPs specified in configuration. If any of PDPs returns positive result **Handle()** return true, otherwise false. This class is the main entry for configuring authorization, and could include different PDP configured inside.

5.4.2 Member Function Documentation

5.4.2.1 bool ArcSec::ArcAuthZ::MakePDPs (Arc::Config * *cfg*) [protected]

Create PDP according to conf info

5.4.2.2 virtual bool ArcSec::ArcAuthZ::Handle (Arc::Message * *msg*) [virtual]

Get authorization decision

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

- ArcAuthZ.h

5.5 ArcSec::ArcEvaluator Class Reference

Execute the policy evaluation, based on the request and policy.

```
#include <ArcEvaluator.h>
```

Public Member Functions

- **ArcEvaluator** (Arc::XMLNode *cfg)
- **ArcEvaluator** (const char *cfgfile)
- virtual Response * **evaluate** (Request *request)
- virtual Response * **evaluate** (const Source &request)
- virtual Response * **evaluate** (Request *request, const Source &policy)
- virtual Response * **evaluate** (const Source &request, const Source &policy)
- virtual Response * **evaluate** (Request *request, Policy *policyobj)
- virtual Response * **evaluate** (const Source &request, Policy *policyobj)
- virtual AttributeFactory * **getAttrFactory** ()
- virtual FnFactory * **getFnFactory** ()
- virtual AlgFactory * **getAlgFactory** ()
- virtual void **addPolicy** (const Source &policy, const std::string &id="")
- virtual void **addPolicy** (Policy *policy, const std::string &id="")
- virtual void **removePolicies** (void)
- virtual void **setCombiningAlg** (EvaluatorCombiningAlg alg)

Protected Member Functions

- virtual Response * **evaluate** (EvaluationCtx *ctx)

Friends

- class **EvaluatorContext**

5.5.1 Detailed Description

Execute the policy evaluation, based on the request and policy.

5.5.2 Member Function Documentation

5.5.2.1 virtual Response* ArcSec::ArcEvaluator::evaluate (Request *request) [virtual]

Evaluate the request based on the policy information inside PolicyStore

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

- ArcEvaluator.h

5.6 ArcSec::ArcFnFactory Class Reference

Function factory class for Arc specified attributes.

```
#include <ArcFnFactory.h>
```

Public Member Functions

- virtual Function * [createFn](#) (const std::string &type)

5.6.1 Detailed Description

Function factory class for Arc specified attributes.

5.6.2 Member Function Documentation

5.6.2.1 virtual Function* ArcSec::ArcFnFactory::createFn (const std::string &type) [virtual]

return a Function object according to the "Function" attribute in the XML node; The [ArcFnFactory](#) itself will release the Function objects

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

- ArcFnFactory.h

5.7 ArcSec::ArcPDP Class Reference

[ArcPDP](#) - PDP which can handle the Arc specific request and policy schema.

```
#include <ArcPDP.h>
```

Public Member Functions

- **ArcPDP** (Arc::Config *cfg)
- virtual bool **isPermitted** (Arc::Message *msg)

Static Public Member Functions

- static PDP * **get_arc_pdp** (Arc::Config *cfg, Arc::ChainContext *ctx)

Static Protected Attributes

- static Arc::Logger **logger**

5.7.1 Detailed Description

[ArcPDP](#) - PDP which can handle the Arc specific request and policy schema.

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

- ArcPDP.h

5.8 ArcSec::ArcPDPServiceInvoker Class Reference

[ArcPDPServiceInvoker](#) - client which will invoke pdpservice.

```
#include <ArcPDPServiceInvoker.h>
```

Public Member Functions

- **ArcPDPServiceInvoker** (Arc::Config *cfg)
- virtual bool **isPermitted** (Arc::Message *msg)

Static Public Member Functions

- static PDP * **get_pdp_service_invoker** (Arc::Config *cfg, Arc::ChainContext *ctx)

Static Protected Attributes

- static Arc::Logger **logger**

5.8.1 Detailed Description

[ArcPDPServiceInvoker](#) - client which will invoke pdpservice.

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

- ArcPDPServiceInvoker.h

5.9 ArcSec::ArcPolicy Class Reference

[ArcPolicy](#) class to parse and operate Arc specific <Policy> node.

```
#include <ArcPolicy.h>
```

Public Member Functions

- [ArcPolicy](#) (Arc::XMLNode *node)
- [ArcPolicy](#) (Arc::XMLNode *node, EvaluatorContext *ctx)
- virtual Result **eval** (EvaluationCtx *ctx)
- virtual void **setEvaluatorContext** (EvaluatorContext *evaluatorcontext)
- virtual void **make_policy** ()
- virtual MatchResult **match** (EvaluationCtx *ctx)
- virtual std::string **getEffect** ()
- virtual EvalResult & **getEvalResult** ()
- virtual void **setEvalResult** (EvalResult &res)

Static Protected Attributes

- static Arc::Logger **logger**

5.9.1 Detailed Description

[ArcPolicy](#) class to parse and operate Arc specific <Policy> node.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 ArcSec::ArcPolicy::ArcPolicy (Arc::XMLNode * *node*)

Constructor

5.9.2.2 ArcSec::ArcPolicy::ArcPolicy (Arc::XMLNode * *node*, EvaluatorContext * *ctx*)

Constructor

5.9.3 Member Function Documentation

5.9.3.1 virtual void ArcSec::ArcPolicy::make_policy () [virtual]

Parse XMLNode, and construct the low-level Rule object

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

- ArcPolicy.h

5.10 ArcSec::ArcRequestItem Class Reference

Container, <Subjects, Actions, Objects, Contexts> tuple.

```
#include <ArcRequestItem.h>
```

Public Member Functions

- **ArcRequestItem** (Arc::XMLNode &node, AttributeFactory *attrfactory)
- virtual SubList **getSubjects** () const
- virtual void **setSubjects** (const SubList &sl)
- virtual ResList **getResources** () const
- virtual void **setResources** (const ResList &rl)
- virtual ActList **getActions** () const
- virtual void **setActions** (const ActList &actions)
- virtual CtxList **getContexts** () const
- virtual void **setContexts** (const CtxList &ctx)

5.10.1 Detailed Description

Container, <Subjects, Actions, Objects, Contexts> tuple.

Specified [ArcRequestItem](#) which can parse Arc request formate

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

- ArcRequestItem.h

5.11 ArcSec::ArcRule Class Reference

[ArcRule](#) class to parse Arc specific <Rule> node.

```
#include <ArcRule.h>
```

Public Member Functions

- **ArcRule** (Arc::XMLNode *node, EvaluatorContext *ctx)
- virtual std::string **getEffect** ()
- virtual Result **eval** (EvaluationCtx *ctx)
- virtual MatchResult **match** (EvaluationCtx *ctx)
- virtual EvalResult & **getEvalResult** ()
- virtual void **setEvalResult** (EvalResult &res)

Static Protected Attributes

- static Arc::Logger **logger**

5.11.1 Detailed Description

[ArcRule](#) class to parse Arc specific <Rule> node.

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

- ArcRule.h

5.12 ArcSec::CountPDP Class Reference

[CountPDP](#) - PDP which can handle the Arc specific request and policy schema.

```
#include <CountPDP.h>
```

Public Member Functions

- **CountPDP** (Arc::Config *cfg)
- virtual bool **isPermitted** (Arc::Message *msg)

Static Public Member Functions

- static PDP * **get_count_pdp** (Arc::Config *cfg, Arc::ChainContext *ctx)

Static Protected Attributes

- static Arc::Logger **logger**

5.12.1 Detailed Description

[CountPDP](#) - PDP which can handle the Arc specific request and policy schema.

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

- CountPDP.h

5.13 ArcSec::DelegationPDP Class Reference

```
#include <DelegationPDP.h>
```

Public Member Functions

- **DelegationPDP** (Arc::Config *cfg)
- virtual bool **isPermitted** (Arc::Message *msg)

Static Public Member Functions

- static PDP * **get_delegation_pdp** (Arc::Config *cfg, Arc::ChainContext *ctx)

Static Protected Attributes

- static Arc::Logger **logger**

5.13.1 Detailed Description

DeleagtionPDP - PDP which can handle the Arc specific request and policy provided as identity delegation policy.

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

- DelegationPDP.h

5.14 ArcSec::DenyPDP Class Reference

This PDP always returns false (deny).

```
#include <DenyPDP.h>
```

Public Member Functions

- **DenyPDP** (Arc::Config *cfg)
- virtual bool **isPermitted** (Arc::Message *msg)

Static Public Member Functions

- static PDP * **get_deny_pdp** (Arc::Config *cfg, Arc::ChainContext *ctx)

5.14.1 Detailed Description

This PDP always returns false (deny).

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

- DenyPDP.h

5.15 Arc::LDAPQuery Class Reference

```
#include <LDAPQuery.h>
```

Public Member Functions

- [LDAPQuery](#) (const std::string &ldaphost, int ldapport, bool anonymous=true, const std::string &usersn="", int timeout=TIMEOUT)
- [~LDAPQuery](#) ()
- bool [Query](#) (const std::string &base, const std::string &filter="(objectclass=*)", const std::list< std::string > &attributes=std::list< std::string >(), URL::Scope scope=URL::subtree)
- bool [Result](#) (ldap_callback callback, void *ref)

Friends

- int [my_sasl_interact](#) (ldap *, unsigned int, void *, void *)

5.15.1 Detailed Description

[LDAPQuery](#) class; querying of LDAP servers.

5.15.2 Constructor & Destructor Documentation

5.15.2.1 Arc::LDAPQuery::LDAPQuery (const std::string & ldaphost, int ldapport, bool anonymous = true, const std::string & usersn = "", int timeout = TIMEOUT)

Constructs a new [LDAPQuery](#) object and sets connection options. The connection is first established when calling [Query](#).

5.15.2.2 Arc::LDAPQuery::~~LDAPQuery ()

Destructor. Will disconnect from the ldapserver if still connected.

5.15.3 Member Function Documentation

5.15.3.1 bool Arc::LDAPQuery::Query (const std::string & base, const std::string & filter = "(objectclass=*)", const std::list< std::string > & attributes = std::list< std::string >(), URL::Scope scope = URL::subtree)

Queries the ldap server.

5.15.3.2 bool Arc::LDAPQuery::Result (ldap_callback callback, void * ref)

Retrieves the result of the query from the ldap-server.

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

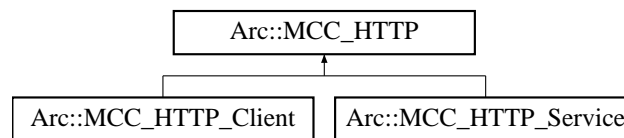
- [LDAPQuery.h](#)

5.16 Arc::MCC_HTTP Class Reference

A base class for HTTP client and service MCCs.

```
#include <MCCHTTP.h>
```

Inheritance diagram for Arc::MCC_HTTP::



Public Member Functions

- `MCC_HTTP` (Arc::Config *cfg)

Static Protected Attributes

- static Arc::Logger **logger**

5.16.1 Detailed Description

A base class for HTTP client and service MCCs.

This is a base class for HTTP client and service MCCs. It provides some common functionality for them, i.e. so far only a logger.

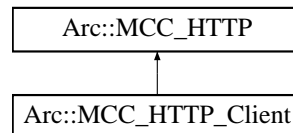
The documentation for this class was generated from the following file:

- `MCCHTTP.h`

5.17 Arc::MCC_HTTP_Client Class Reference

```
#include <MCCHTTP.h>
```

Inheritance diagram for Arc::MCC_HTTP_Client::



Public Member Functions

- **MCC_HTTP_Client** (Arc::Config *cfg)
- virtual MCC_Status **process** (Message &, Message &)

Protected Attributes

- std::string **method_**
- std::string **endpoint_**

5.17.1 Detailed Description

This class is a client part of HTTP MCC. It accepts PayloadRawInterface payload and uses it as body to generate HTTP request. Request is passed to next MCC as PayloadRawInterface type of payload. Returned PayloadStreamInterface payload is parsed into HTTP response and it's body is passed back to calling MCC as PayloadRawInterface. Attributes of request/input message of type HTTP:name are translated into HTTP header with corresponding 'name's. Special attributes HTTP:METHOD and HTTP:ENDPOINT specify method and URL in HTTP request. If not present method and URL are taken from configuration. In output/response message following attributes are present: HTTP:CODE - response code of HTTP HTTP:REASON - reason string of HTTP response HTTP:name - all 'name' attributes of HTTP header.

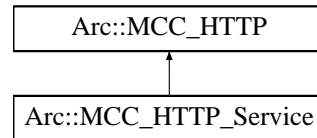
The documentation for this class was generated from the following file:

- MCCHTTP.h

5.18 Arc::MCC_HTTP_Service Class Reference

```
#include <MCCHTTP.h>
```

Inheritance diagram for Arc::MCC_HTTP_Service::



Public Member Functions

- **MCC_HTTP_Service** (Arc::Config *cfg)
- virtual MCC_Status **process** (Message &, Message &)

5.18.1 Detailed Description

This class implements MCC to processes HTTP request. On input payload with PayloadStreamInterface is expected. HTTP message is read from stream and its body is converted into PayloadRaw and passed to next MCC. Returned payload of PayloadRawInterface type is treated as body part of returning [Payload-HTTP](#). Generated HTTP response is sent through stream passed in input payload. During processing of request/input message following attributes are generated: HTTP:METHORD - HTTP method e.g. GET, PUT, POST, etc. HTTP:ENDPOINT - URL taken from HTTP request ENDPOINT - global attribute equal to HTTP:ENDPOINT HTTP:RANGESTART - start of requested byte range HTTP:RANGEEND - end of requested byte range (inclusive) HTTP:name - all 'name' attributes of HTTP header. Attributes of response message of HTTP:name type are translated into HTTP header with corresponding 'name's.

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

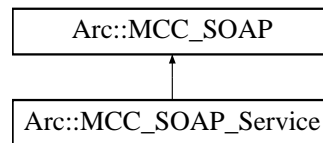
- MCCHTTP.h

5.19 Arc::MCC_SOAP Class Reference

A base class for SOAP client and service MCCs.

```
#include <MCCSOAP.h>
```

Inheritance diagram for Arc::MCC_SOAP::



Public Member Functions

- **MCC_SOAP** (Arc::Config *cfg)

Static Protected Attributes

- static Arc::Logger **logger**

5.19.1 Detailed Description

A base class for SOAP client and service MCCs.

This is a base class for SOAP client and service MCCs. It provides some common functionality for them, i.e. so far only a logger.

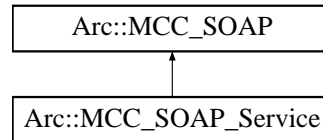
The documentation for this class was generated from the following file:

- MCCSOAP.h

5.20 Arc::MCC_SOAP_Service Class Reference

```
#include <MCCSOAP.h>
```

Inheritance diagram for Arc::MCC_SOAP_Service::



Public Member Functions

- **MCC_SOAP_Service** (Arc::Config *cfg)
- virtual MCC_Status **process** (Message &, Message &)

5.20.1 Detailed Description

This MCC parses SOAP message from input payload. On input payload with PayloadRawInterface is expected. It's converted into PayloadSOAP and passed next MCC. Returned PayloadSOAP is converted into PayloadRaw and returned to calling MCC.

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

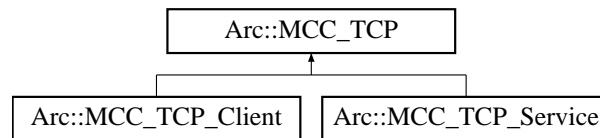
- MCCSOAP.h

5.21 Arc::MCC_TCP Class Reference

A base class for TCP client and service MCCs.

```
#include <MCCTCP.h>
```

Inheritance diagram for Arc::MCC_TCP::



Public Member Functions

- `MCC_TCP` (Arc::Config *cfg)

Static Protected Attributes

- static Arc::Logger **logger**

Friends

- class **PayloadTCPSocket**

5.21.1 Detailed Description

A base class for TCP client and service MCCs.

This is a base class for TCP client and service MCCs. It provides some common functionality for them, i.e. so far only a logger.

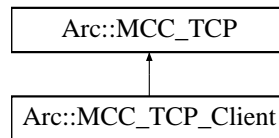
The documentation for this class was generated from the following file:

- MCCTCP.h

5.22 Arc::MCC_TCP_Client Class Reference

```
#include <MCCTCP.h>
```

Inheritance diagram for Arc::MCC_TCP_Client::



Public Member Functions

- **MCC_TCP_Client** (Arc::Config *cfg)
- virtual MCC_Status **process** (Message &, Message &)

5.22.1 Detailed Description

This class is MCC implementing TCP client. Upon creation it connects to specified TCP post at specified host. process() method accepts PayloadRawInterface type of payload. Content of payload is sent over TCP socket. It returns PayloadStreamInterface payload for previous MCC to read response.

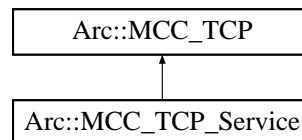
The documentation for this class was generated from the following file:

- MCCTCP.h

5.23 Arc::MCC_TCP_Service Class Reference

```
#include <MCCTCP.h>
```

Inheritance diagram for Arc::MCC_TCP_Service::



Public Member Functions

- [MCC_TCP_Service](#) (Arc::Config *cfg)
- virtual MCC_Status **process** (Message &, Message &)

Friends

- class `mcc_tcp_exec_t`

Classes

- class `mcc_tcp_exec_t`

5.23.1 Detailed Description

This class is MCC implementing TCP server. Upon creation this object binds to specified TCP ports and listens for incoming TCP connections on dedicated thread. Each connection is accepted and dedicated thread is created. Then that thread is used to call process() method of next MCC in chain. That method is passed payload implementing PayloadStreamInterface. On response payload with PayloadRawInterface is expected. Alternatively called MCC may use provided PayloadStreamInterface to send it's response back directly. During processing of request this MCC generates following attributes: TCP:HOST - IP address of interface to which local TCP socket is bound TCP:PORT - port number to which local TCP socket is bound TCP:REMOTEHOST - IP address from which connection is accepted TCP:REMOTEPORT - TCP port from which connection is accepted TCP:ENDPOINT - URL-like representation of remote connection - `://HOST:PORT` ENDPOINT - global attribute equal to TCP:ENDPOINT

5.23.2 Constructor & Destructor Documentation

5.23.2.1 Arc::MCC_TCP_Service::MCC_TCP_Service (Arc::Config * *cfg*)

executing function for connection thread

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

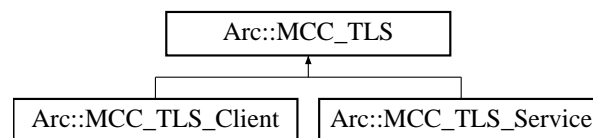
- MCCTCP.h

5.24 Arc::MCC_TLS Class Reference

A base class for SOAP client and service MCCs.

```
#include <MCCTLS.h>
```

Inheritance diagram for Arc::MCC_TLS::



Public Member Functions

- **MCC_TLS** (Arc::Config *cfg)
- const std::string & **CADir** (void) const
- const std::string & **CAFile** (void) const
- const std::string & **ProxyFile** (void) const
- const std::string & **CertFile** (void) const
- const std::string & **KeyFile** (void) const
- bool **GlobusPolicy** (void) const

Protected Member Functions

- bool **tls_load_certificate** (SSL_CTX *sslctx, const std::string &cert_file, const std::string &key_file, const std::string &password, const std::string &random_file)
- bool **do_ssl_init** (void)

Static Protected Member Functions

- static void **ssl_locking_cb** (int mode, int n, const char *file, int line)
- static unsigned long **ssl_id_cb** (void)

Protected Attributes

- std::string **ca_dir_**
- std::string **ca_file_**
- std::string **proxy_file_**
- std::string **cert_file_**
- std::string **key_file_**
- bool **globus_policy_**

Static Protected Attributes

- static bool **ssl_initialized_**
- static Glib::Mutex **lock_**
- static Glib::Mutex * **ssl_locks_**
- static Logger **logger**

5.24.1 Detailed Description

A base class for SOAP client and service MCCs.

This is a base class for SOAP client and service MCCs. It provides some common functionality for them, i.e. so far only a logger.

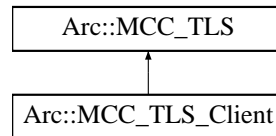
The documentation for this class was generated from the following file:

- MCCTLS.h

5.25 Arc::MCC_TLS_Client Class Reference

```
#include <MCCTLS.h>
```

Inheritance diagram for Arc::MCC_TLS_Client::



Public Member Functions

- **MCC_TLS_Client** (Arc::Config *cfg)
- virtual MCC_Status **process** (Message &, Message &)
- virtual void **Next** (MCCInterface *next, const std::string &label="")

5.25.1 Detailed Description

This class is MCC implementing TLS client.

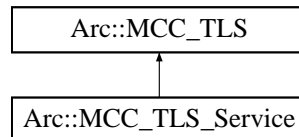
The documentation for this class was generated from the following file:

- MCCTLS.h

5.26 Arc::MCC_TLS_Service Class Reference

```
#include <MCCTLS.h>
```

Inheritance diagram for Arc::MCC_TLS_Service::



Public Member Functions

- **MCC_TLS_Service** (Arc::Config *cfg)
- virtual MCC_Status **process** (Message &, Message &)

5.26.1 Detailed Description

This MCC implements TLS server side functionality. Upon creation this object creates SSL_CTX object and configures SSL_CTX object with some environment information about credential. Because we cannot know the "socket" when the creation of MCC_TLS_Service/MCC_TLS_Client object (not like [MCC_TCP_Client](#), which can creat socket in the constructor method by using information in configuration file), we can only creat "ssl" object which is binded to specified "socket", when [MCC_HTTP_Client](#) calls the process() method of [MCC_TLS_Client](#) object, or [MCC_TCP_Service](#) calls the process() method of [MCC_TLS_Service](#) object. The "ssl" object is embeded in a payload called PayloadTLSSocket.

The process() method of [MCC_TLS_Service](#) is passed payload implementing PayloadStreamInterface and the method returns empty PayloadRaw payload in "outmsg". The ssl object is created and bound to Stream payload when constructing the PayloadTLSSocket in the process() method.

During processing of message this MCC generates attribute TLS:PEERDN which contains Distinguished Name of remote peer.

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

- MCCTLS.h

5.27 Arc::PayloadHTTP Class Reference

```
#include <PayloadHTTP.h>
```

Public Member Functions

- [PayloadHTTP](#) (PayloadStreamInterface &stream)
- [PayloadHTTP](#) (const std::string &method, const std::string &url, PayloadStreamInterface &stream)
- [PayloadHTTP](#) (const std::string &method, const std::string &url)
- [PayloadHTTP](#) (int code, const std::string &reason, PayloadStreamInterface &stream)
- [PayloadHTTP](#) (int code, const std::string &reason)
- virtual **operator bool** (void)
- virtual bool **operator!** (void)
- virtual const std::string & [Attribute](#) (const std::string &name)
- virtual const std::map< std::string, std::string > & [Attributes](#) (void)
- virtual void [Attribute](#) (const std::string &name, const std::string &value)
- virtual bool [Flush](#) (void)
- virtual std::string [Method](#) ()
- virtual std::string [Endpoint](#) ()
- virtual std::string [Reason](#) ()
- virtual int [Code](#) ()
- virtual void [Body](#) (PayloadRawInterface &body, bool ownership=true)
- virtual char **operator[]** (int pos) const
- virtual char * [Content](#) (int pos=-1)
- virtual int [Size](#) (void) const
- virtual char * [Insert](#) (int pos=0, int size=0)
- virtual char * [Insert](#) (const char *s, int pos=0, int size=0)
- virtual char * [Buffer](#) (unsigned int num=0)
- virtual int [BufferSize](#) (unsigned int num=0) const
- virtual int [BufferPos](#) (unsigned int num=0) const
- virtual bool [Truncate](#) (unsigned int size)

Protected Member Functions

- bool [readline](#) (std::string &line)
- bool [read](#) (char *buf, int &size)
- bool [parse_header](#) (void)
- bool [get_body](#) (void)

Protected Attributes

- bool [valid_](#)
- PayloadStreamInterface & [stream_](#)
- PayloadRawInterface * [body_](#)
- bool [body_own_](#)
- std::string [uri_](#)
- int [version_major_](#)
- int [version_minor_](#)
- std::string [method_](#)

- int `code_`
- std::string `reason_`
- int `length_`
- bool `chunked_`
- bool `keep_alive_`
- std::map< std::string, std::string > `attributes_`
- char `tbuf_` [1024]
- int `tbuflen_`

5.27.1 Detailed Description

This class implements parsing and generation of HTTP messages. It implements only subset of HTTP/1.1 and also provides an PayloadRawInterface for including as payload into Message passed through MCC chains.

5.27.2 Constructor & Destructor Documentation

5.27.2.1 Arc::PayloadHTTP::PayloadHTTP (PayloadStreamInterface & *stream*)

Constructor - creates object by parsing HTTP request or response from stream. Supplied stream is associated with object for later use.

5.27.2.2 Arc::PayloadHTTP::PayloadHTTP (const std::string & *method*, const std::string & *url*, PayloadStreamInterface & *stream*)

Constructor - creates HTTP request to be sent through stream. HTTP message is not sent yet.

5.27.2.3 Arc::PayloadHTTP::PayloadHTTP (const std::string & *method*, const std::string & *url*)

Constructor - creates HTTP request to be rendered through Raw interface.

5.27.2.4 Arc::PayloadHTTP::PayloadHTTP (int *code*, const std::string & *reason*, PayloadStreamInterface & *stream*)

Constructor - creates HTTP response to be sent through stream. HTTP message is not sent yet.

5.27.2.5 Arc::PayloadHTTP::PayloadHTTP (int *code*, const std::string & *reason*)

Constructor - creates HTTP response to be rendered through Raw interface.

5.27.3 Member Function Documentation

5.27.3.1 bool Arc::PayloadHTTP::readline (std::string & *line*) [protected]

Read from stream till

5.27.3.2 **bool Arc::PayloadHTTP::read (char * *buf*, int & *size*)** [protected]

Read up to 'size' bytes from stream_

5.27.3.3 **bool Arc::PayloadHTTP::parse_header (void)** [protected]

Read HTTP header and fill internal variables

5.27.3.4 **bool Arc::PayloadHTTP::get_body (void)** [protected]

Read Body of HTTP message and attach it to inherited PayloadRaw object

5.27.3.5 **virtual const std::string& Arc::PayloadHTTP::Attribute (const std::string & *name*)** [virtual]

Returns HTTP header attribute with specified name. Empty string if no such attribute.

5.27.3.6 **virtual const std::map<std::string,std::string>& Arc::PayloadHTTP::Attributes (void)** [virtual]

Returns all HTTP header attributes.

5.27.3.7 **virtual void Arc::PayloadHTTP::Attribute (const std::string & *name*, const std::string & *value*)** [virtual]

Sets HTTP header attribute 'name' to 'value'

5.27.3.8 **virtual bool Arc::PayloadHTTP::Flush (void)** [virtual]

Send created object through associated stream. If there is no stream associated then HTTP specific data is inserted into Raw buffers of this object. In last case this operation should not be repeated till content of buffer is completely rewritten.

5.27.3.9 **virtual void Arc::PayloadHTTP::Body (PayloadRawInterface & *body*, bool *ownership* = true)** [virtual]

Assign HTTP body. Assigned object is not copied. Instead it is remembered and made available through Raw interface. If 'ownership' is true then passed object is treated as being owned by this instance and destroyed in destructor.

5.27.4 Member Data Documentation

5.27.4.1 **PayloadRawInterface* [Arc::PayloadHTTP::body_](#)** [protected]

stream used to communicate to outside

5.27.4.2 `bool Arc::PayloadHTTP::body_own_` [protected]

associated HTTP Body if any

5.27.4.3 `std::string Arc::PayloadHTTP::uri_` [protected]

if true body_ is owned by this

5.27.4.4 `int Arc::PayloadHTTP::version_major_` [protected]

URI being contacted

5.27.4.5 `int Arc::PayloadHTTP::version_minor_` [protected]

major number of HTTP version - must be 1

5.27.4.6 `std::string Arc::PayloadHTTP::method_` [protected]

minor number of HTTP version - must be 0 or 1

5.27.4.7 `int Arc::PayloadHTTP::code_` [protected]

HTTP method being used or requested

5.27.4.8 `std::string Arc::PayloadHTTP::reason_` [protected]

HTTP code being sent or supplied

5.27.4.9 `int Arc::PayloadHTTP::length_` [protected]

HTTP reason being sent or supplied

5.27.4.10 `bool Arc::PayloadHTTP::chunked_` [protected]

Content-length of HTTP message

5.27.4.11 `bool Arc::PayloadHTTP::keep_alive_` [protected]

true if content is chunked

5.27.4.12 `std::map<std::string, std::string> Arc::PayloadHTTP::attributes_` [protected]

true if conection should not be closed after response

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

- PayloadHTTP.h

5.28 Arc::PayloadTCPSocket Class Reference

```
#include <PayloadTCPSocket.h>
```

Public Member Functions

- [PayloadTCPSocket](#) (const char *hostname, int port, Logger &logger)
- [PayloadTCPSocket](#) (const std::string endpoint, Logger &logger)
- [PayloadTCPSocket](#) (int s, Logger &logger)
- [PayloadTCPSocket](#) ([PayloadTCPSocket](#) &s, Logger &logger)
- [PayloadTCPSocket](#) (PayloadStream &s, Logger &logger)
- virtual bool **Get** (char *buf, int &size)
- virtual bool **Put** (const char *buf, int size)

5.28.1 Detailed Description

This class extends PayloadStream with TCP socket specific features

5.28.2 Constructor & Destructor Documentation

5.28.2.1 Arc::PayloadTCPSocket::PayloadTCPSocket (const char * *hostname*, int *port*, Logger & *logger*)

Constructor - connects to TCP server at specified hostname:port

5.28.2.2 Arc::PayloadTCPSocket::PayloadTCPSocket (const std::string *endpoint*, Logger & *logger*)

Constructor - connects to TCP server at specified endpoint - hostname:port

5.28.2.3 Arc::PayloadTCPSocket::PayloadTCPSocket (int *s*, Logger & *logger*) [inline]

Constructor - creates object of already connected socket. Socket is NOT closed in destructor.

5.28.2.4 Arc::PayloadTCPSocket::PayloadTCPSocket ([PayloadTCPSocket](#) & *s*, Logger & *logger*) [inline]

Copy constructor - inherits socket of copied object. Socket is NOT closed in destructor.

5.28.2.5 Arc::PayloadTCPSocket::PayloadTCPSocket (PayloadStream & *s*, Logger & *logger*) [inline]

Copy constructor - inherits handle of copied object. Handle is NOT closed in destructor.

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

- PayloadTCPSocket.h

5.29 Arc::PayloadTLSStream Class Reference

```
#include <PayloadTLSStream.h>
```

Public Member Functions

- [PayloadTLSStream](#) (Logger &logger, SSL *ssl=NULL)
- **PayloadTLSStream** ([PayloadTLSStream](#) &stream)
- virtual [~PayloadTLSStream](#) (void)
- void **HandleError** (int code=SSL_ERROR_NONE)
- virtual bool **Get** (char *buf, int &size)
- virtual bool **Get** (std::string &buf)
- virtual std::string **Get** (void)
- virtual bool **Put** (const char *buf, int size)
- virtual bool **Put** (const std::string &buf)
- virtual bool **Put** (const char *buf)
- virtual **operator bool** (void)
- virtual bool **operator!** (void)
- virtual int **Timeout** (void) const
- virtual void **Timeout** (int to)
- X509 * [GetPeerCert](#) (void)
- [STACK_OF](#) (X509)*GetPeerChain(void)
- X509 * [GetCert](#) (void)

Static Public Member Functions

- static void **HandleError** (Logger &logger, int code=SSL_ERROR_NONE)
- static void **ClearError** (void)

Protected Attributes

- int **timeout_**
- SSL * **ssl_**
- Logger & **logger_**

5.29.1 Detailed Description

Implemetation of PayloadStreamInterface for SSL handle.

5.29.2 Constructor & Destructor Documentation

5.29.2.1 Arc::PayloadTLSStream::PayloadTLSStream (Logger & *logger*, SSL * *ssl* = NULL)

Constructor. Attaches to already open handle. Handle is not managed by this class and must be closed by external code.

5.29.2.2 **virtual Arc::PayloadTLSStream::~~PayloadTLSStream (void)** [virtual]

Destructor.

5.29.3 Member Function Documentation

5.29.3.1 **X509* Arc::PayloadTLSStream::GetPeerCert (void)**

Get peer certificate from the established ssl. Obtained X509 object is owned by this instance and becomes invalid after destruction. Still obtained has to be freed at end of usage.

5.29.3.2 **Arc::PayloadTLSStream::STACK_OF (X509)**

Get chain of peer certificates from the established ssl. Obtained X509 object is owned by this instance and becomes invalid after destruction.

5.29.3.3 **X509* Arc::PayloadTLSStream::GetCert (void)**

Get local certificate from associated ssl. Obtained X509 object is owned by this instance and becomes invalid after destruction.

5.29.4 Member Data Documentation

5.29.4.1 **SSL* Arc::PayloadTLSStream::ssl_** [protected]

Timeout for read/write operations

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

- PayloadTLSStream.h

5.30 ArcSec::SimpleListPDP Class Reference

Tests X509 subject against list of subjects in file.

```
#include <SimpleListPDP.h>
```

Public Member Functions

- **SimpleListPDP** (Arc::Config *cfg)
- virtual bool **isPermitted** (Arc::Message *msg)

Static Public Member Functions

- static PDP * **get_simplelist_pdp** (Arc::Config *cfg, Arc::ChainContext *ctx)

Static Protected Attributes

- static Arc::Logger **logger**

5.30.1 Detailed Description

Tests X509 subject against list of subjects in file.

This class implements PDP interface. It's isPermitted() method compares X590 subject of requestor obtained from TLS layer (TLS:PEERDN) to list of subjects (ne per line) in external file. Locations of file is defined by 'location' attribute of PDP caonfiguration. Returns true if subject is present in list, otherwise false.

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

- SimpleListPDP.h

5.31 ArcSec::UsernameTokenSH Class Reference

Adds WS-Security Username Token into SOAP Header.

```
#include <UsernameTokenSH.h>
```

Public Member Functions

- **UsernameTokenSH** (Arc::Config *cfg, Arc::ChainContext *ctx)
- virtual bool **Handle** (Arc::Message *msg)

Static Public Member Functions

- static SecHandler * **get_sechandler** (Arc::Config *cfg, Arc::ChainContext *ctx)

5.31.1 Detailed Description

Adds WS-Security Username Token into SOAP Header.

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

- UsernameTokenSH.h

5.32 ArcSec::XACMLPolicy Class Reference

[XACMLPolicy](#) class to parse and operate XACML specific <Policy> node.

```
#include <XACMLPolicy.h>
```

Public Member Functions

- [XACMLPolicy](#) (Arc::XMLNode &node, EvaluatorContext *ctx)
- virtual Result **eval** (EvaluationCtx *ctx)
- virtual MatchResult **match** (EvaluationCtx *ctx)
- virtual std::string **getEffect** ()
- virtual EvalResult & **getEvalResult** ()

Static Protected Attributes

- static Arc::Logger **logger**

5.32.1 Detailed Description

[XACMLPolicy](#) class to parse and operate XACML specific <Policy> node.

5.32.2 Constructor & Destructor Documentation

5.32.2.1 ArcSec::XACMLPolicy::XACMLPolicy (Arc::XMLNode & *node*, EvaluatorContext * *ctx*)

Constructor -

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

- XACMLPolicy.h

5.33 ArcSec::XACMLRule Class Reference

[XACMLRule](#) class to parse XACML specific <Rule> node.

```
#include <XACMLRule.h>
```

Public Member Functions

- **XACMLRule** (Arc::XMLNode &node, EvaluatorContext *ctx)
- virtual std::string **getEffect** ()
- virtual Result **eval** (EvaluationCtx *ctx)
- virtual MatchResult **match** (EvaluationCtx *ctx)
- virtual EvalResult & **getEvalResult** ()

Static Protected Attributes

- static Arc::Logger **logger**

5.33.1 Detailed Description

[XACMLRule](#) class to parse XACML specific <Rule> node.

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

- XACMLRule.h

5.34 ArcSec::XACMLTarget Class Reference

[XACMLTarget](#) class to parse and operate XACML specific <Target> node.

```
#include <XACMLTarget.h>
```

Public Member Functions

- [XACMLTarget](#) (Arc::XMLNode &node, EvaluatorContext *ctx)
- virtual MatchResult **match** (EvaluationCtx *ctx)

5.34.1 Detailed Description

[XACMLTarget](#) class to parse and operate XACML specific <Target> node.

5.34.2 Constructor & Destructor Documentation

5.34.2.1 ArcSec::XACMLTarget::XACMLTarget (Arc::XMLNode & *node*, EvaluatorContext * *ctx*)

Constructor -

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

- XACMLTarget.h

Index

- ~LDAPQuery
 - Arc::LDAPQuery, [25](#)
- ~PayloadTLSStream
 - Arc::PayloadTLSStream, [43](#)
- AndList
 - ArcSec, [9](#)
- Arc::LDAPQuery, [25](#)
 - ~LDAPQuery, [25](#)
 - LDAPQuery, [25](#)
 - Query, [25](#)
 - Result, [25](#)
- Arc::MCC_HTTP, [26](#)
- Arc::MCC_HTTP_Client, [27](#)
- Arc::MCC_HTTP_Service, [28](#)
- Arc::MCC_SOAP, [29](#)
- Arc::MCC_SOAP_Service, [30](#)
- Arc::MCC_TCP, [31](#)
- Arc::MCC_TCP_Client, [32](#)
- Arc::MCC_TCP_Service, [33](#)
 - MCC_TCP_Service, [33](#)
- Arc::MCC_TLS, [34](#)
- Arc::MCC_TLS_Client, [36](#)
- Arc::MCC_TLS_Service, [37](#)
- Arc::PayloadHTTP, [38](#)
- Arc::PayloadHTTP
 - Attribute, [40](#)
 - Attributes, [40](#)
 - attributes_, [41](#)
 - Body, [40](#)
 - body_, [40](#)
 - body_own_, [40](#)
 - chunked_, [41](#)
 - code_, [41](#)
 - Flush, [40](#)
 - get_body, [40](#)
 - keep_alive_, [41](#)
 - length_, [41](#)
 - method_, [41](#)
 - parse_header, [40](#)
 - PayloadHTTP, [39](#)
 - read, [39](#)
 - readline, [39](#)
 - reason_, [41](#)
 - uri_, [41](#)
 - version_major_, [41](#)
 - version_minor_, [41](#)
- Arc::PayloadTCPSocket, [42](#)
 - Arc::PayloadTCPSocket
 - PayloadTCPSocket, [42](#)
- Arc::PayloadTLSStream, [43](#)
 - Arc::PayloadTLSStream
 - ~PayloadTLSStream, [43](#)
 - GetCert, [44](#)
 - GetPeerCert, [44](#)
 - PayloadTLSStream, [43](#)
 - ssl_, [44](#)
 - STACK_OF, [44](#)
- ArcPolicy
 - ArcSec::ArcPolicy, [19](#)
- ArcSec, [7](#)
- ArcSec
 - AndList, [9](#)
 - Match, [9](#)
 - OrList, [9](#)
- ArcSec::AllowPDP, [11](#)
- ArcSec::ArcAlgFactory, [12](#)
- ArcSec::ArcAlgFactory
 - createAlg, [12](#)
- ArcSec::ArcAttributeFactory, [13](#)
- ArcSec::ArcAttributeFactory
 - createValue, [13](#)
- ArcSec::ArcAuthZ, [14](#)
- ArcSec::ArcAuthZ
 - Handle, [14](#)
 - MakePDPs, [14](#)
- ArcSec::ArcEvaluator, [15](#)
- ArcSec::ArcEvaluator
 - evaluate, [15](#)
- ArcSec::ArcFnFactory, [16](#)
- ArcSec::ArcFnFactory
 - createFn, [16](#)
- ArcSec::ArcPDP, [17](#)
- ArcSec::ArcPDPServiceInvoker, [18](#)
- ArcSec::ArcPolicy, [19](#)
- ArcSec::ArcPolicy
 - ArcPolicy, [19](#)
 - make_policy, [19](#)
- ArcSec::ArcRequestItem, [20](#)
- ArcSec::ArcRule, [21](#)

- ArcSec::CountPDP, 22
- ArcSec::DelegationPDP, 23
- ArcSec::DenyPDP, 24
- ArcSec::SimpleListPDP, 45
- ArcSec::UsernameTokenSH, 46
- ArcSec::XACMLPolicy, 47
 - ArcSec::XACMLPolicy
 - XACMLPolicy, 47
- ArcSec::XACMLRule, 48
- ArcSec::XACMLTarget, 49
- ArcSec::XACMLTarget
 - XACMLTarget, 49
- Attribute
 - Arc::PayloadHTTP, 40
- Attributes
 - Arc::PayloadHTTP, 40
- attributes_
 - Arc::PayloadHTTP, 41
- Body
 - Arc::PayloadHTTP, 40
- body_
 - Arc::PayloadHTTP, 40
- body_own_
 - Arc::PayloadHTTP, 40
- chunked_
 - Arc::PayloadHTTP, 41
- code_
 - Arc::PayloadHTTP, 41
- createAlg
 - ArcSec::ArcAlgFactory, 12
- createFn
 - ArcSec::ArcFnFactory, 16
- createValue
 - ArcSec::ArcAttributeFactory, 13
- evaluate
 - ArcSec::ArcEvaluator, 15
- Flush
 - Arc::PayloadHTTP, 40
- get_body
 - Arc::PayloadHTTP, 40
- GetCert
 - Arc::PayloadTLSStream, 44
- GetPeerCert
 - Arc::PayloadTLSStream, 44
- Handle
 - ArcSec::ArcAuthZ, 14
- keep_alive_
 - Arc::PayloadHTTP, 41
- LDAPQuery
 - Arc::LDAPQuery, 25
- length_
 - Arc::PayloadHTTP, 41
- make_policy
 - ArcSec::ArcPolicy, 19
- MakePDPs
 - ArcSec::ArcAuthZ, 14
- Match
 - ArcSec, 9
- MCC_TCP_Service
 - Arc::MCC_TCP_Service, 33
- method_
 - Arc::PayloadHTTP, 41
- OrList
 - ArcSec, 9
- parse_header
 - Arc::PayloadHTTP, 40
- PayloadHTTP
 - Arc::PayloadHTTP, 39
- PayloadTCPSocket
 - Arc::PayloadTCPSocket, 42
- PayloadTLSStream
 - Arc::PayloadTLSStream, 43
- Query
 - Arc::LDAPQuery, 25
- read
 - Arc::PayloadHTTP, 39
- readline
 - Arc::PayloadHTTP, 39
- reason_
 - Arc::PayloadHTTP, 41
- Result
 - Arc::LDAPQuery, 25
- ssl_
 - Arc::PayloadTLSStream, 44
- STACK_OF
 - Arc::PayloadTLSStream, 44
- uri_
 - Arc::PayloadHTTP, 41
- version_major_
 - Arc::PayloadHTTP, 41
- version_minor_
 - Arc::PayloadHTTP, 41
- XACMLPolicy
 - ArcSec::XACMLPolicy, 47
- XACMLTarget
 - ArcSec::XACMLTarget, 49