

# VOMS CC API Reference Manual

## 1.5.0

Generated by Doxygen 1.3.6

Tue Oct 20 07:26:32 2009



# Contents

<b>1</b>	<b>VOMS CC API Data Structure Index</b>	<b>1</b>
1.1	VOMS CC API Data Structures . . . . .	1
<b>2</b>	<b>VOMS CC API File Index</b>	<b>3</b>
2.1	VOMS CC API File List . . . . .	3
<b>3</b>	<b>VOMS CC API Data Structure Documentation</b>	<b>5</b>
3.1	attribute Struct Reference . . . . .	5
3.2	attributelist Struct Reference . . . . .	6
3.3	contactdata Struct Reference . . . . .	7
3.4	data Struct Reference . . . . .	9
3.5	voms Struct Reference . . . . .	10
3.6	vomsdata Struct Reference . . . . .	14
<b>4</b>	<b>VOMS CC API File Documentation</b>	<b>23</b>
4.1	voms_api.h File Reference . . . . .	23



# Chapter 1

## VOMS CC API Data Structure Index

### 1.1 VOMS CC API Data Structures

Here are the data structures with brief descriptions:

<a href="#">attribute</a>	5
<a href="#">attributelist</a>	6
<a href="#">contactdata</a>	7
<a href="#">data</a> (User's characteristics: can be repeated )	9
<a href="#">voms</a>	10
<a href="#">vomsdata</a>	14



## Chapter 2

# VOMS CC API File Index

### 2.1 VOMS CC API File List

Here is a list of all files with brief descriptions:

<a href="#">voms_api.h</a> . . . . .	23
--------------------------------------	----





## Chapter 3

# VOMS CC API Data Structure Documentation

### 3.1 attribute Struct Reference

```
#include <voms_api.h>
```

#### Data Fields

- std::string [name](#)
- std::string [qualifier](#)
- std::string [value](#)

#### 3.1.1 Field Documentation

##### 3.1.1.1 std::string [attribute::name](#)

attribute's group

Definition at line 47 of file voms\_api.h.

##### 3.1.1.2 std::string [attribute::qualifier](#)

attribute's qualifier

Definition at line 48 of file voms\_api.h.

##### 3.1.1.3 std::string [attribute::value](#)

attribute's value

Definition at line 49 of file voms\_api.h.

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

- [voms\\_api.h](#)

## 3.2 attributelist Struct Reference

```
#include <voms_api.h>
```

### Data Fields

- std::string [grantor](#)
- std::vector< [attribute](#) > [attributes](#)

### 3.2.1 Field Documentation

#### 3.2.1.1 std::vector<[attribute](#)> [attributelist::attributes](#)

The attributes themselves.

Definition at line 54 of file voms\_api.h.

#### 3.2.1.2 std::string [attributelist::grantor](#)

Who granted these attributes.

Definition at line 53 of file voms\_api.h.

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

- [voms\\_api.h](#)

## 3.3 contactdata Struct Reference

```
#include <voms_api.h>
```

### Data Fields

- std::string [nick](#)
- std::string [host](#)
- std::string [contact](#)
- std::string [vo](#)
- int [port](#)
- int [version](#)

### 3.3.1 Field Documentation

#### 3.3.1.1 std::string [contactdata::contact](#)

The subject of the server's certificate  
Definition at line 72 of file voms\_api.h.

#### 3.3.1.2 std::string [contactdata::host](#)

The hostname of the server  
Definition at line 71 of file voms\_api.h.

#### 3.3.1.3 std::string [contactdata::nick](#)

The alias of the server  
Definition at line 70 of file voms\_api.h.

#### 3.3.1.4 int [contactdata::port](#)

The port on which the server is listening  
Definition at line 74 of file voms\_api.h.

#### 3.3.1.5 int [contactdata::version](#)

The version of globus under which the server is running  
Definition at line 76 of file voms\_api.h.

#### 3.3.1.6 std::string [contactdata::vo](#)

The VO served by this server  
Definition at line 73 of file voms\_api.h.

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

- [voms\\_api.h](#)

## 3.4 data Struct Reference

User's characteristics: can be repeated.

```
#include <voms_api.h>
```

### Data Fields

- `std::string` [group](#)
- `std::string` [role](#)
- `std::string` [cap](#)

#### 3.4.1 Detailed Description

User's characteristics: can be repeated.

Definition at line 38 of file `voms_api.h`.

#### 3.4.2 Field Documentation

##### 3.4.2.1 `std::string` [data::cap](#)

user's capability

Definition at line 41 of file `voms_api.h`.

##### 3.4.2.2 `std::string` [data::group](#)

user's group

Definition at line 39 of file `voms_api.h`.

##### 3.4.2.3 `std::string` [data::role](#)

user's role

Definition at line 40 of file `voms_api.h`.

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

- [voms\\_api.h](#)

## 3.5 voms Struct Reference

```
#include <voms_api.h>
```

### Public Member Functions

- [voms](#) (const [voms](#) &)
- [voms](#) ()
- [voms](#) & [operator=](#) (const [voms](#) &)
- [~voms](#) ()
- AC \* [GetAC](#) ()
- std::vector< [attributelist](#) > & [GetAttributes](#) ()
- std::vector< std::string > [GetTargets](#) ()

### Data Fields

- int [version](#)
- int [siglen](#)
- std::string [signature](#)
- std::string [user](#)
- std::string [userca](#)
- std::string [server](#)
- std::string [serverca](#)
- std::string [voname](#)
- std::string [uri](#)
- std::string [date1](#)
- std::string [date2](#)
- [data\\_type](#) type
- std::vector< [data](#) > [std](#)
- std::string [custom](#)
- std::vector< std::string > [fqan](#)
- std::string [serial](#)

### Friends

- class [vomsdata](#)
- int [TranslateVOMS](#) (struct vomsdatar \*vd, std::vector< [voms](#) > &v, int \*error)

### 3.5.1 Constructor & Destructor Documentation

**3.5.1.1** `voms::voms (const voms &)`

**3.5.1.2** `voms::voms ()`

**3.5.1.3** `voms::~~voms ()`

### 3.5.2 Member Function Documentation

**3.5.2.1** `AC* voms::GetAC ()`

**3.5.2.2** `std::vector<attributelist>& voms::GetAttributes ()`

Generic attributes

**3.5.2.3** `std::vector<std::string> voms::GetTargets ()`

**3.5.2.4** `voms& voms::operator= (const voms &)`

### 3.5.3 Friends And Related Function Documentation

**3.5.3.1** `int TranslateVOMS (struct vomsdatar * vd, std::vector< voms > & v, int * error)`  
[friend]

**3.5.3.2** `friend class vomsdata` [friend]

Definition at line 80 of file voms\_api.h.

### 3.5.4 Field Documentation

**3.5.4.1** `std::string voms::custom`

The data returned by an S command

Definition at line 94 of file voms\_api.h.

**3.5.4.2** `std::string voms::date1`

Beginning of validity of the user info

Definition at line 90 of file voms\_api.h.

**3.5.4.3** `std::string voms::date2`

End of validity of the user info

Definition at line 91 of file voms\_api.h.

**3.5.4.4** `std::vector<std::string> voms::fqan`

Keeps the data in the compact format

Definition at line 96 of file voms\_api.h.

**3.5.4.5** `std::string voms::serial`

Serial number. "0" if coming from non-ac

Definition at line 97 of file voms\_api.h.

**3.5.4.6** `std::string voms::server`

The VOMS server DN, as from its certificate

Definition at line 86 of file voms\_api.h.

**3.5.4.7** `std::string voms::serverca`

The CA which signed the VOMS certificate

Definition at line 87 of file voms\_api.h.

**3.5.4.8** `int voms::siglen`

The length of the VOMS server signature

Definition at line 82 of file voms\_api.h.

**3.5.4.9** `std::string voms::signature`

The VOMS server signature

Definition at line 83 of file voms\_api.h.

**3.5.4.10** `std::vector<data> voms::std`

User's characteristics

Definition at line 93 of file voms\_api.h.

**3.5.4.11** `data_type voms::type`

The type of data returned

Definition at line 92 of file voms\_api.h.

**3.5.4.12** `std::string voms::uri`

The URI of the VOMS server

Definition at line 89 of file voms\_api.h.



**3.5.4.13** `std::string voms::user`

The user's DN, as from his certificate

Definition at line 84 of file voms\_api.h.

**3.5.4.14** `std::string voms::userca`

The CA which signed the user's certificate

Definition at line 85 of file voms\_api.h.

**3.5.4.15** `int voms::version`

0 means data didn't originate from an AC

Definition at line 81 of file voms\_api.h.

**3.5.4.16** `std::string voms::voname`

The name of the VO to which the VOMS belongs

Definition at line 88 of file voms\_api.h.

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

- [voms\\_api.h](#)

## 3.6 vomsdata Struct Reference

```
#include <voms_api.h>
```

### Public Member Functions

- [vomsdata](#) (std::string voms\_dir="", std::string cert\_dir="")
- bool [LoadSystemContacts](#) (std::string dir="")
- bool [LoadUserContacts](#) (std::string dir="")
- std::vector< [contactdata](#) > [FindByAlias](#) (std::string alias)
- std::vector< [contactdata](#) > [FindByVO](#) (std::string vo)
- void [Order](#) (std::string att)
- void [ResetOrder](#) (void)
- void [AddTarget](#) (std::string target)
- std::vector< std::string > [ListTargets](#) (void)
- void [ResetTargets](#) (void)
- std::string [ServerErrors](#) (void)
- bool [Retrieve](#) (X509 \*cert, STACK\_OF(X509)\*chain, [recurse\\_type](#) how=RECURSE\_CHAIN)
- bool [Contact](#) (std::string hostname, int port, std::string servsubject, std::string command)
- bool [Contact](#) (std::string hostname, int port, std::string servsubject, std::string command, int timeout)
- bool [ContactRaw](#) (std::string hostname, int port, std::string servsubject, std::string command, std::string &raw, int &version)
- bool [ContactRaw](#) (std::string hostname, int port, std::string servsubject, std::string command, std::string &raw, int &version, int timeout)
- void [SetVerificationType](#) ([verify\\_type](#) how)
- void [SetLifetime](#) (int lifetime)
- bool [Import](#) (std::string buffer)
- bool [Export](#) (std::string &data)
- bool [DefaultData](#) (voms &)
- std::string [ErrorMessage](#) (void)
- bool [RetrieveFromCtx](#) ([gss\\_ctx\\_id\\_t](#) context, [recurse\\_type](#) how)
- bool [RetrieveFromCred](#) ([gss\\_cred\\_id\\_t](#) credential, [recurse\\_type](#) how)
- bool [Retrieve](#) (X509\_EXTENSION \*ext)
- bool [RetrieveFromProxy](#) ([recurse\\_type](#) how)
- bool [Retrieve](#) (FILE \*file, [recurse\\_type](#) how)
- [~vomsdata](#) ()
- [vomsdata](#) (const [vomsdata](#) &)
- void [SetRetryCount](#) (int retryCount)
- void [SetVerificationTime](#) (time\_t)
- bool [LoadCredentials](#) (X509 \*, EVP\_PKEY \*, STACK\_OF(X509)\*)

### Data Fields

- [verror\\_type](#) error
- std::vector< [voms](#) > data
- std::string workvo
- std::string extra\_data

## 3.6.1 Constructor & Destructor Documentation

### 3.6.1.1 vomsdata::vomsdata (std::string *voms\_dir* = "", std::string *cert\_dir* = "")

**Parameters:**

*voms\_dir* The directory which contains the certificate of the VOMS server

*cert\_dir* The directory which contains the certificate of the CA

If *voms\_dir* is empty, the value of the environment variable X509\_VOMS\_DIR is taken.

If *cert\_dir* is empty, the value of the environment variable X509\_CERT\_DIR is taken.

### 3.6.1.2 vomsdata::~~vomsdata ()

### 3.6.1.3 vomsdata::vomsdata (const vomsdata &)

## 3.6.2 Member Function Documentation

### 3.6.2.1 void vomsdata::AddTarget (std::string *target*)

Adds a target to the AC.

**Parameters:**

*target* The target to be added. it should be a FQDN.

### 3.6.2.2 bool vomsdata::Contact (std::string *hostname*, int *port*, std::string *servsubject*, std::string *command*, int *timeout*)

Contacts a VOMS server to get a certificate

It is the equivalent of the voms\_proxy\_init command, but without the --include functionality.

**Parameters:**

*hostname* FQDN of the VOMS server

*port* the port on which the VOMS server is listening

*servsubject* the subject of the server's certificate

*command* the command sent to the server

**Returns:**

failure (F) or success (T)

### 3.6.2.3 bool vomsdata::Contact (std::string *hostname*, int *port*, std::string *servsubject*, std::string *command*)

Contacts a VOMS server to get a certificate

It is the equivalent of the voms\_proxy\_init command, but without the --include functionality.

**Parameters:**

*hostname* FQDN of the VOMS server

*port* the port on which the VOMS server is listening

*servsubject* the subject of the server's certificate

*command* the command sent to the server

**Returns:**

failure (F) or success (T)

**3.6.2.4 bool vomsdata::ContactRaw (std::string *hostname*, int *port*, std::string *servsubject*, std::string *command*, std::string & *raw*, int & *version*, int *timeout*)**

Same as Contact, however it does not start the verification process, and the message received from the server is not parsed.

**Parameters:**

*hostname* FQDN of the VOMS server

*port* the port on which the VOMS server is listening

*servsubject* the subject of the server's certificate

*command* the command sent to the server

*raw* OUTPUT PARAMETER the answer from the server

*version* OUTPUT PARAMETER the version of the answer

**Returns:**

failure (F) or success (T)

**3.6.2.5 bool vomsdata::ContactRaw (std::string *hostname*, int *port*, std::string *servsubject*, std::string *command*, std::string & *raw*, int & *version*)**

Same as Contact, however it does not start the verification process, and the message received from the server is not parsed.

**Parameters:**

*hostname* FQDN of the VOMS server

*port* the port on which the VOMS server is listening

*servsubject* the subject of the server's certificate

*command* the command sent to the server

*raw* OUTPUT PARAMETER the answer from the server

*version* OUTPUT PARAMETER the version of the answer

**Returns:**

failure (F) or success (T)

**3.6.2.6 bool vomsdata::DefaultData (voms &)**

Get the default data extension from those present in the pseudo certificate

### 3.6.2.7 `std::string vomsdata::ErrorMessage (void)`

Gets a textual description of the error.

**Returns:**

A string containing the error message.

### 3.6.2.8 `bool vomsdata::Export (std::string & data)`

Exports data from `vomsdata::data` to the format used for inclusion into a certificate.

The function doesn't verify the data

**Parameters:**

*data* The certificate extension

**Returns:**

Failure (F) or Success (T)

### 3.6.2.9 `std::vector<contactdata> vomsdata::FindByAlias (std::string alias)`

Finds servers which share a common alias.

**Parameters:**

*alias* The alias to look for.

**Returns:**

The servers found. The order in which they are returned is unspecified.

### 3.6.2.10 `std::vector<contactdata> vomsdata::FindByVO (std::string vo)`

Finds servers which serve a common VO

**Parameters:**

*vo* The VO name to look for.

**Returns:**

The servers found. The order in which they are returned is unspecified.

### 3.6.2.11 `bool vomsdata::Import (std::string buffer)`

Converts data from the format used for inclusion into a certificate to the internal format

The function does verify the data.

**Parameters:**

*buffer* contains the data to be converted

**Returns:**

Failure (F) or Success (T)

**3.6.2.12** `std::vector<std::string> vomsdata::ListTargets (void)`

Returns the list of targets.

**3.6.2.13** `bool vomsdata::LoadCredentials (X509 *, EVP_PKEY *, STACK_OF(X509)*)`**3.6.2.14** `bool vomsdata::LoadSystemContacts (std::string dir = "")`

Loads the system wide configuration files.

**Parameters:**

*dir* The directory in which the files are stored.

If *dir* is empty, defaults to /opt/edg/etc/vomses.

**Returns:**

True if all went OK, false otherwise.

**3.6.2.15** `bool vomsdata::LoadUserContacts (std::string dir = "")`

Loads the user-specific configuration files.

**Parameters:**

*dir* The directory in which the files are stored.

If *dir* is empty, defaults to \$VOMS\_USERCONF. If this is empty too, defaults to \$HOME/.edg/vomses, or to ~/.edg/vomses as a last resort.

**Returns:**

True if all went OK, false otherwise.

**3.6.2.16** `void vomsdata::Order (std::string att)`

Sets up the ordering of the results.

Defines the ordering of the data returned by [Contact\(\)](#). Results are ordered in the same order as the calls to this function.

**Parameters:**

*att* The attribute to be ordered.

**3.6.2.17** `void vomsdata::ResetOrder (void)`

Resets the ordering.

**3.6.2.18** `void vomsdata::ResetTargets (void)`

Resets the target list.

**3.6.2.19 bool vomsdata::Retrieve (FILE \* *file*, recurse\_type *how*)**

Gets VOMS information from a proxy saved as a file.

**Parameters:**

*the* file

*how* Recursion type

**Returns:**

failure (F) or success (T)

Note: Does NOT verify that the proxy is valid. Such verification must be obtained through other means.

**3.6.2.20 bool vomsdata::Retrieve (X509\_EXTENSION \* *ext*)**

Gets VOMS information from the given extension

**Parameters:**

*ext* The extension to parse.

**Returns:**

failure (F) or success (T)

**3.6.2.21 bool vomsdata::Retrieve (X509 \* *cert*, STACK\_OF(X509)\* *chain*, recurse\_type *how* = RECURSE\_CHAIN)**

Extracts the VOMS extension from an X.509 certificate. The function doesn't check the validity of the certificates, but it does check the content of the user data.

**Parameters:**

*cert* The certificate with the VOMS extensions

*chain* The chain of the validation certificates (only the intermediate ones)

*how* Recursion type

**Returns:**

failure (F) or success (T)

**3.6.2.22 bool vomsdata::RetrieveFromCred (gss\_cred\_id\_t *credential*, recurse\_type *how*)**

Gets VOMS information from the given globus credential

**Parameters:**

*credential* The credential from which to retrieve the certificate.

*how* Recursion type

**Returns:**

failure (F) or success (T)

### 3.6.2.23 `bool vomsdata::RetrieveFromCtx (gss_ctx_id_t context, recurse_type how)`

Gets VOMS information from the given globus context

**Parameters:**

*context* The context from which to retrieve the certificate.

*how* Recursion type

**Returns:**

failure (F) or success (T)

### 3.6.2.24 `bool vomsdata::RetrieveFromProxy (recurse_type how)`

Gets VOMS information from an existing globus proxy

**Parameters:**

*how* Recursion type

**Returns:**

failure (F) or success (T)

### 3.6.2.25 `std::string vomsdata::ServerErrors (void)`

Gets the error message returned by the server

### 3.6.2.26 `void vomsdata::SetLifetime (int lifetime)`

Set requested lifetime for the [Contact\(\)](#) call.

**Parameters:**

*lifetime* Requested lifetime, in seconds

### 3.6.2.27 `void vomsdata::SetRetryCount (int retryCount)`

### 3.6.2.28 `void vomsdata::SetVerificationTime (time_t)`

### 3.6.2.29 `void vomsdata::SetVerificationType (verify_type how)`

Sets the type of verification done on the data.

**Parameters:**

*how* The type of verification.

## 3.6.3 Field Documentation

### 3.6.3.1 `std::vector<voms> vomsdata::data`

User's info, as in the certificate extension. It may contain data gathered from more than one VOMS server, Definition at line 344 of file voms\_api.h.



### 3.6.3.2 `verror_type vomsdata::error`

Error code

Definition at line 189 of file voms\_api.h.

### 3.6.3.3 `std::string vomsdata::extra_data`

The data specified by the user with the `--include` switch.

Note that this field doesn't contain the result of a request to the VOMS server, but instead data specified by the user.

The reason for the introduction of this extension is to let a user include important data into his proxy certificate, like, for example, a kerberos ticket

Definition at line 348 of file voms\_api.h.

### 3.6.3.4 `std::string vomsdata::workvo`

The value of the `-vo` option of the `voms-proxy-init` command

Definition at line 347 of file voms\_api.h.

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

- [voms\\_api.h](#)



## Chapter 4

# VOMS CC API File Documentation

### 4.1 voms\_api.h File Reference

```
#include <fstream>
#include <string>
#include <vector>
#include <openssl/x509.h>
#include <openssl/bio.h>
#include <sys/types.h>
#include "newformat.h"
```

#### Data Structures

- struct [data](#)  
*User's characteristics: can be repeated.*
- struct [attribute](#)
- struct [attributelist](#)
- struct [contactdata](#)
- struct [voms](#)
- struct [vomsdata](#)
- class [vomsdata::Initializer](#)

#### Typedefs

- typedef void \* [gss\\_cred\\_id\\_t](#)
- typedef void \* [gss\\_ctx\\_id\\_t](#)
- typedef bool(\* [check\\_sig](#))(X509 \*, void \*, [verror\\_type](#) &)

#### Enumerations

- enum [data\\_type](#) { [TYPE\\_NODATA](#), [TYPE\\_STD](#), [TYPE\\_CUSTOM](#) }

*The type of data returned.*

- enum `recurse_type` { `RECURSE_CHAIN`, `RECURSE_NONE`, `RECURSE_DEEP` }
- enum `verify_type` {  
`VERIFY_FULL` = 0xffffffff, `VERIFY_NONE` = 0x00000000, `VERIFY_DATE` = 0x00000001,  
`VERIFY_TARGET` = 0x00000002,  
`VERIFY_KEY` = 0x00000004, `VERIFY_SIGN` = 0x00000008, `VERIFY_ORDER` = 0x00000010,  
`VERIFY_ID` = 0x00000020,  
`VERIFY_CERTLIST` = 0x00000040 }
- enum `verror_type` {  
`VERR_NONE`, `VERR_NOCKET`, `VERR_NOIDENT`, `VERR_COMM`,  
`VERR_PARAM`, `VERR_NOEXT`, `VERR_NOINIT`, `VERR_TIME`,  
`VERR_IDCHECK`, `VERR_EXTRAINFO`, `VERR_FORMAT`, `VERR_NODATA`,  
`VERR_PARSE`, `VERR_DIR`, `VERR_SIGN`, `VERR_SERVER`,  
`VERR_MEM`, `VERR_VERIFY`, `VERR_TYPE`, `VERR_ORDER`,  
`VERR_SERVERCODE`, `VERR_NOTAVAIL`, `VERR_FILE` }

*Error codes.*

## Functions

- int `getMajorVersionNumber` (void)
- int `getMinorVersionNumber` (void)
- int `getPatchVersionNumber` (void)

### 4.1.1 Typedef Documentation

#### 4.1.1.1 typedef bool(\* `check_sig`)(X509 \*, void \*, `verror_type` &)

Definition at line 168 of file `voms_api.h`.

#### 4.1.1.2 typedef void\* `gss_cred_id_t`

Definition at line 26 of file `voms_api.h`.

#### 4.1.1.3 typedef void\* `gss_ctx_id_t`

Definition at line 27 of file `voms_api.h`.

### 4.1.2 Enumeration Type Documentation

#### 4.1.2.1 enum `data_type`

The type of data returned.

Enumeration values:

`TYPE_NODATA` no data

*TYPE\_STD* group, role, capability triplet  
*TYPE\_CUSTOM* result of an S command

Definition at line 60 of file voms\_api.h.

#### 4.1.2.2 enum [recurse\\_type](#)

Enumeration values:

*RECURSE\_CHAIN*  
*RECURSE\_NONE*  
*RECURSE\_DEEP*

Definition at line 121 of file voms\_api.h.

#### 4.1.2.3 enum [verify\\_type](#)

Enumeration values:

*VERIFY\_FULL*  
*VERIFY\_NONE*  
*VERIFY\_DATE*  
*VERIFY\_TARGET*  
*VERIFY\_KEY*  
*VERIFY\_SIGN*  
*VERIFY\_ORDER*  
*VERIFY\_ID*  
*VERIFY\_CERTLIST*

Definition at line 127 of file voms\_api.h.

#### 4.1.2.4 enum [verror\\_type](#)

Error codes.

Enumeration values:

*VERR\_NONE*  
*VERR\_NOSOCKET* Socket problem  
*VERR\_NOIDENT* Cannot identify itself (certificate problem)  
*VERR\_COMM* Server problem  
*VERR\_PARAM* Wrong parameters  
*VERR\_NOEXT* VOMS extension missing  
*VERR\_NOINIT* Initialization error  
*VERR\_TIME* Error in time checking  
*VERR\_IDCHECK* User data in extension different from the real ones  
*VERR\_EXTRAINFO* VO name and URI missing  
*VERR\_FORMAT* Wrong data format

***VERR\_NODATA*** Empty extension  
***VERR\_PARSE*** Parse error  
***VERR\_DIR*** Directory error  
***VERR\_SIGN*** Signature error  
***VERR\_SERVER*** Unidentifiable VOMS server  
***VERR\_MEM*** Memory problems  
***VERR\_VERIFY*** Generic verification error  
***VERR\_TYPE*** Returned data of unknown type  
***VERR\_ORDER*** Ordering different than required  
***VERR\_SERVERCODE*** Error message from the server  
***VERR\_NOTAVAIL*** Method not available  
***VERR\_FILE*** Error reading data from file

Definition at line 141 of file voms\_api.h.

### **4.1.3 Function Documentation**

**4.1.3.1 int getMajorVersionNumber (void)**

**4.1.3.2 int getMinorVersionNumber (void)**

**4.1.3.3 int getPatchVersionNumber (void)**

# Index

- ~voms
  - voms, [11](#)
- ~vomldata
  - vomldata, [15](#)
- AddTarget
  - vomldata, [15](#)
- attribute, [5](#)
  - name, [5](#)
  - qualifier, [5](#)
  - value, [5](#)
- attributelist, [6](#)
  - attributes, [6](#)
  - grantor, [6](#)
- attributes
  - attributelist, [6](#)
- cap
  - data, [9](#)
- check\_sig
  - voms\_api.h, [24](#)
- Contact
  - vomldata, [15](#)
- contact
  - contactdata, [7](#)
- contactdata, [7](#)
  - contact, [7](#)
  - host, [7](#)
  - nick, [7](#)
  - port, [7](#)
  - version, [7](#)
  - vo, [7](#)
- ContactRaw
  - vomldata, [16](#)
- custom
  - voms, [11](#)
- data, [9](#)
  - cap, [9](#)
  - group, [9](#)
  - role, [9](#)
  - vomldata, [20](#)
- data\_type
  - voms\_api.h, [24](#)
- date1
  - voms, [11](#)
- date2
  - voms, [11](#)
- DefaultData
  - vomldata, [16](#)
- error
  - vomldata, [20](#)
- ErrorMessage
  - vomldata, [16](#)
- Export
  - vomldata, [17](#)
- extra\_data
  - vomldata, [21](#)
- FindByAlias
  - vomldata, [17](#)
- FindByVO
  - vomldata, [17](#)
- fqan
  - voms, [11](#)
- GetAC
  - voms, [11](#)
- GetAttributes
  - voms, [11](#)
- getMajorVersionNumber
  - voms\_api.h, [26](#)
- getMinorVersionNumber
  - voms\_api.h, [26](#)
- getPatchVersionNumber
  - voms\_api.h, [26](#)
- GetTargets
  - voms, [11](#)
- grantor
  - attributelist, [6](#)
- group
  - data, [9](#)
- gss\_cred\_id\_t
  - voms\_api.h, [24](#)
- gss\_ctx\_id\_t
  - voms\_api.h, [24](#)
- host
  - contactdata, [7](#)

- Import
  - vomsdata, 17
- ListTargets
  - vomsdata, 17
- LoadCredentials
  - vomsdata, 18
- LoadSystemContacts
  - vomsdata, 18
- LoadUserContacts
  - vomsdata, 18
- name
  - attribute, 5
- nick
  - contactdata, 7
- operator=
  - voms, 11
- Order
  - vomsdata, 18
- port
  - contactdata, 7
- qualifier
  - attribute, 5
- RECURSE\_CHAIN
  - voms\_api.h, 25
- RECURSE\_DEEP
  - voms\_api.h, 25
- RECURSE\_NONE
  - voms\_api.h, 25
- recurse\_type
  - voms\_api.h, 25
- ResetOrder
  - vomsdata, 18
- ResetTargets
  - vomsdata, 18
- Retrieve
  - vomsdata, 18, 19
- RetrieveFromCred
  - vomsdata, 19
- RetrieveFromCtx
  - vomsdata, 19
- RetrieveFromProxy
  - vomsdata, 20
- role
  - data, 9
- serial
  - voms, 12
- server
  - voms, 12
- serverca
  - voms, 12
- ServerErrors
  - vomsdata, 20
- SetLifetime
  - vomsdata, 20
- SetRetryCount
  - vomsdata, 20
- SetVerificationTime
  - vomsdata, 20
- SetVerificationType
  - vomsdata, 20
- siglen
  - voms, 12
- signature
  - voms, 12
- std
  - voms, 12
- TranslateVOMS
  - voms, 11
- type
  - voms, 12
- TYPE\_CUSTOM
  - voms\_api.h, 25
- TYPE\_NODATA
  - voms\_api.h, 24
- TYPE\_STD
  - voms\_api.h, 24
- uri
  - voms, 12
- user
  - voms, 12
- userca
  - voms, 13
- value
  - attribute, 5
- VERIFY\_CERTLIST
  - voms\_api.h, 25
- VERIFY\_DATE
  - voms\_api.h, 25
- VERIFY\_FULL
  - voms\_api.h, 25
- VERIFY\_ID
  - voms\_api.h, 25
- VERIFY\_KEY
  - voms\_api.h, 25
- VERIFY\_NONE
  - voms\_api.h, 25
- VERIFY\_ORDER
  - voms\_api.h, 25
- VERIFY\_SIGN



- voms\_api.h, [25](#)
- VERIFY\_TARGET
  - voms\_api.h, [25](#)
- verify\_type
  - voms\_api.h, [25](#)
- VERR\_COMM
  - voms\_api.h, [25](#)
- VERR\_DIR
  - voms\_api.h, [26](#)
- VERR\_EXTRAINFO
  - voms\_api.h, [25](#)
- VERR\_FILE
  - voms\_api.h, [26](#)
- VERR\_FORMAT
  - voms\_api.h, [25](#)
- VERR\_IDCHECK
  - voms\_api.h, [25](#)
- VERR\_MEM
  - voms\_api.h, [26](#)
- VERR\_NODATA
  - voms\_api.h, [25](#)
- VERR\_NOEXT
  - voms\_api.h, [25](#)
- VERR\_NOIDENT
  - voms\_api.h, [25](#)
- VERR\_NOINIT
  - voms\_api.h, [25](#)
- VERR\_NONE
  - voms\_api.h, [25](#)
- VERR\_NOSOCKET
  - voms\_api.h, [25](#)
- VERR\_NOTAVAIL
  - voms\_api.h, [26](#)
- VERR\_ORDER
  - voms\_api.h, [26](#)
- VERR\_PARAM
  - voms\_api.h, [25](#)
- VERR\_PARSE
  - voms\_api.h, [26](#)
- VERR\_SERVER
  - voms\_api.h, [26](#)
- VERR\_SERVERCODE
  - voms\_api.h, [26](#)
- VERR\_SIGN
  - voms\_api.h, [26](#)
- VERR\_TIME
  - voms\_api.h, [25](#)
- VERR\_TYPE
  - voms\_api.h, [26](#)
- VERR\_VERIFY
  - voms\_api.h, [26](#)
- verror\_type
  - voms\_api.h, [25](#)
- version
  - contactdata, [7](#)
  - voms, [13](#)
- vo
  - contactdata, [7](#)
- voms, [10](#)
  - ~voms, [11](#)
  - custom, [11](#)
  - date1, [11](#)
  - date2, [11](#)
  - fqan, [11](#)
  - GetAC, [11](#)
  - GetAttributes, [11](#)
  - GetTargets, [11](#)
  - operator=, [11](#)
  - serial, [12](#)
  - server, [12](#)
  - serverca, [12](#)
  - siglen, [12](#)
  - signature, [12](#)
  - std, [12](#)
  - TranslateVOMS, [11](#)
  - type, [12](#)
  - uri, [12](#)
  - user, [12](#)
  - userca, [13](#)
  - version, [13](#)
  - voms, [11](#)
  - vomsdata, [11](#)
  - voname, [13](#)
- voms\_api.h
  - RECURSE\_CHAIN, [25](#)
  - RECURSE\_DEEP, [25](#)
  - RECURSE\_NONE, [25](#)
  - TYPE\_CUSTOM, [25](#)
  - TYPE\_NODATA, [24](#)
  - TYPE\_STD, [24](#)
  - VERIFY\_CERTLIST, [25](#)
  - VERIFY\_DATE, [25](#)
  - VERIFY\_FULL, [25](#)
  - VERIFY\_ID, [25](#)
  - VERIFY\_KEY, [25](#)
  - VERIFY\_NONE, [25](#)
  - VERIFY\_ORDER, [25](#)
  - VERIFY\_SIGN, [25](#)
  - VERIFY\_TARGET, [25](#)
  - VERR\_COMM, [25](#)
  - VERR\_DIR, [26](#)
  - VERR\_EXTRAINFO, [25](#)
  - VERR\_FILE, [26](#)
  - VERR\_FORMAT, [25](#)
  - VERR\_IDCHECK, [25](#)
  - VERR\_MEM, [26](#)
  - VERR\_NODATA, [25](#)
  - VERR\_NOEXT, [25](#)

- VERR\_NOIDENT, 25
- VERR\_NOINIT, 25
- VERR\_NONE, 25
- VERR\_NOCKET, 25
- VERR\_NOTAVAIL, 26
- VERR\_ORDER, 26
- VERR\_PARAM, 25
- VERR\_PARSE, 26
- VERR\_SERVER, 26
- VERR\_SERVERCODE, 26
- VERR\_SIGN, 26
- VERR\_TIME, 25
- VERR\_TYPE, 26
- VERR\_VERIFY, 26
- voms\_api.h, 23
  - check\_sig, 24
  - data\_type, 24
  - getMajorVersionNumber, 26
  - getMinorVersionNumber, 26
  - getPatchVersionNumber, 26
  - gss\_cred\_id\_t, 24
  - gss\_ctx\_id\_t, 24
  - recurse\_type, 25
  - verify\_type, 25
  - verror\_type, 25
- vomsdata, 14
  - ~vomsdata, 15
  - AddTarget, 15
  - Contact, 15
  - ContactRaw, 16
  - data, 20
  - DefaultData, 16
  - error, 20
  - ErrorMessage, 16
  - Export, 17
  - extra\_data, 21
  - FindByAlias, 17
  - FindByVO, 17
  - Import, 17
  - ListTargets, 17
  - LoadCredentials, 18
  - LoadSystemContacts, 18
  - LoadUserContacts, 18
  - Order, 18
  - ResetOrder, 18
  - ResetTargets, 18
  - Retrieve, 18, 19
  - RetrieveFromCred, 19
  - RetrieveFromCtx, 19
  - RetrieveFromProxy, 20
  - ServerErrors, 20
  - SetLifetime, 20
  - SetRetryCount, 20
  - SetVerificationTime, 20
  - SetVerificationType, 20
  - voms, 11
  - vomsdata, 15
  - workvo, 21
- voname
  - voms, 13
- workvo
  - vomsdata, 21