PlanetLab Central API Documentation

PlanetLab Central API Documentation

Table of Contents

Chapter 1. Introduction

The PlanetLab Central API (PLCAPI) is the interface through which the PlanetLab Central database should be accessed and maintained. The API is used by the website, by nodes, by automated scripts, and by users to access and update information about users, nodes, sites, slices, and other entities maintained by the database.

Authentication

The API should be accessed via XML-RPC over HTTPS. The API supports the standard introspection calls system.listMethods, system.methodSignature, and system.methodHelp, and the standard batching call system.multicall. With the exception of these calls, all PLCAPI calls take an authentication structure as their first argument. All authentication structures require the specification of `AuthMethod`. If the documentation for a call does not further specify the authentication structure, then any of (but only) the following authentication structures may be used:

- Session authentication. User sessions are typically valid for 24 hours. Node sessions are valid until the next reboot. Obtain a session key with GetSession using another form of authentication, such as password or GnuPG authentication.

AuthMethod	`session`
session	Session key

- Password authentication.

AuthMethod	`password`
Username	Username, typically an e-mail address
AuthString	Authentication string, typically a password

- GnuPG authentication. Users may upload a GPG public key using AddPersonKey. Peer GPG keys should be added with AddPeer or UpdatePeer.

AuthMethod	`gpg`
name	Peer or user name
signature	GnuPG signature of the canonicalized[1] XML-RPC[1] representation of the rest of the arguments to the call.

- Anonymous authentication.

AuthMethod	`anonymous`

Roles

Some functions may only be called by users with certain roles (see GetRoles), and others may return different information to different callers depending on the role(s) of the caller.

The `node` and `anonymous` roles are pseudo-roles. A function that allows the `node` role may be called by automated scripts running on a node, such as the Boot and Node Managers. A function that allows the `anonymous` role may be called by anyone; an API authentication structure must still be specified (see the Section called *Authentication*).

Filters

Most of the `Get` methods take a filter argument. Filters may be arrays of integer (and sometimes string) identifiers, or a struct representing a filter on the attributes of the entities being queried. For example,

```
>>> GetNodes([1,2,3])
>>> GetNodes({'node_id': [1,2,3]})
```

Would be equivalent queries. Attributes that are themselves arrays (such as `interface_ids` and `slice_ids` for nodes) cannot be used in filters.

Filters support a few extra features illustrated in the following examples.

Pattern Matching

`*` can be used in a text value and have the usual meaning, so all nodes in the *fr* can be obtained with:

```
GetNodes ( { 'hostname' : '*.fr' } )
```

Negation

Fields starting with a `~` are negated, so non-local nodes can be fetched with:

```
GetNodes( { '~peer_id' : None } )
```

Numeric comparisons

Strictly greater/smaller operations are achieved by prepending the field name like in:

```
GetEvents( { '>time' : 1178531418 } )
```

Greater/smaller or equal:

```
GetEvents( { ']event_id' : 2305 } )
```

Filtering on a sequence field

A field starting with '&' or '|' should refer to a sequence type; the semantics is then that the object's value (expected to be a list) should contain all (&) or any (|) value specified in the corresponding filter value.

```
GetPersons ( { '|role_ids' : [ 20, 40 ] } )

GetPersons ( { '|roles' : ['tech', 'pi'] } )

GetPersons ( { '&roles' : ['admin', 'tech'] } )

GetPersons ( { '&roles' : 'tech' } )
```

Sorting and Clipping

The following 3 special fields can be used to extract only a subset of the results for pagination:

```
GetNodes( { '-SORT' : 'hostname' , '-OFFSET' : 30 , '-LIMIT' : 25 }
```

All criteria / Any criteria

The default in the vast majority of the code is to select objects that match ALL the criteria specified in the struct. It is possible to search for objects that match ANY of these by adding the special '-OR' key (the value is then ignored), as in:

```
GetPersons ( { '-OR' : 'anything', 'site_id':2, '&roles':['admin'] } )
```

Tags

The PLC API comes with a feature called *tags*, that basically aims at supporting an extensible data model. A few classes (as of this writing, Nodes, Interfaces, Sites, Persons and Slices) are eligible for being dynamically extended beyond the basic set of fields that are built into the database schema.

Historically, this is a generalization of the concept of *SliceAttribute* , and the more recent concept of *InterfaceSetting* , that with release 5.0 have been renamed into *SliceTag* and *InterfaceTag* , respectively.

Low level

The low level interface to tags relies on the following items:

- A *TagType* object basically models a new column that needs to be added to other objects. In much the same way as nodes are named through a *hostname* , tagtypes are named with a *tagname*, plus additional information (*category*, *description*).
- *description* is mostly informative, it is used by the web interface to provide more details on the meaning of that tag.
- *category* is used in a variety of ways, in the web interface again. Over time this has become a means to attach various information to a tag type, so it is used as some sort of a poorman's tag tag system :).

- The convention is to set in category a set of slash-separated fields, like the following real examples demonstrate.

```
>>> tagnames=['arch','fcdistro','hrn','hmac','exempt_node_until']
>>> for tt in GetTagTypes(tagnames,['tagname','category']):
>>> ... print "tagname=%-18s category=%s"%(tt['tagname'], tt['category'])
tagname=hrn               category=node/sfa
tagname=hmac              category=slice/auth
tagname=exempt_node_until  category=node/myops
tagname=fcdistro          category=node/slice/config/ui/header=f/rank=w
tagname=arch              category=node/slice/config/ui/header=A/rank=x
```

- *roles* may also be attached to a given tag_type (use AddRoleToTagType or DeleteRoleFromTagType). This is an evolution over the former system based on so-called 'min_role_id', and now any set of roles may be mentioned. More importantly, each type (Node, Person, ...) implements its own policy to let or not non-admin callers change their tags. For example in the current implementation, non-admin users can only change their own person tags. See PLC/AuthorizeHelpers.py for that code.
- The low-level method for managaing tags is then, once the TagType is known to the system, to attach a value to, say, a Node, by calling *AddNodeTag* , and then as usual change this value with *UpdateNodeTag* , or delete it with *DeleteNodeTag* .

Accessors

A rather more convenient way to use tags is through Accessors. This convenience is located in *PLC/Accessors* , and allows you to easily define Get or Set methods dedicated to a given tag. This is for instance how the *GetNodeArch* and *SetNodeArch* methods are implemented. These methods greatly simplify tags manipulation as they take care of

- Creating and enforcing *TagTypes* ; each time you restart your plc, the tag_types mentioned in accessor definitions are created and checked (in terms of the category, description and roles defined in the various calls to define_accessors).
- Create or update the, say, *NodeTag* object, as needed.
- In addition, an accessor definition mentions *get_roles* (defaults to all_roles), and *set_roles* . These values are used as follows. *get_roles* is attached to the Get accessor, so callers that do not have this role cannot run the Get accessor. *set_roles* is attached to the Set accessor, as well as to the corresponding TagType, which in turn is used for checking write access to the tag type.

Site-specific accessors can be defined in */usr/share/plc_api/PLC/Accessors/Accessors_site.py* and will be preserved across updates of the *plcapi* rpm.

The accessors mechanism does not currently support setting slice tags that apply only on a given node or nodegroup.

Through regular Add/Get/Update methods

Finally, tags may also get manipulated through the *AddNode*, *GetNodes*, and *UpdateNode* methods:

- The `define_accessors` function in the Accessors factory has an optional argument named `expose_in_api` . When this is set, the corresponding tag becomes visible from the Add/Get/Update methods almost as if it was a native tag.

- So for instance the following code would be legal and do as expected:

```
# create a x86_64 node
>>> AddNode({'hostname':'pl1.foo.com','arch':'x86_64'})
# get details for pl1.foo.com including tag 'arch' tag
>>> GetNodes(['pl1.foo.com'],['boot_state','node_type','arch'])
# set the 'deployment' tag
>>> UpdateNode('pl1.foo.com',{'deployment':'beta'})
# get all alpha and beta nodes
>>> GetNodes({'deployment':'*a'},['hostname','deployment'])
```

- The current limitations about tags, as opposed to native fields, is that for performance, tags won't get returned when using the implicit set of columns. So for instance:

```
# get all details for 'pl1.foo.com'
>>> node=GetNodes(['pl1.foo.com'])[0]
# this did not return the 'arch' tag
>>> 'arch' in node
False
```

- For a similar reason, any tag used in the filter argument will *have to* be mentioned in the list of returned columns as well. For example:

```
# if 'hrn' is not part of the result, this does not work
>>> ns=GetNodes({'hrn':'ple.*'},['hostname'])
Database error b59e068c-589a-4ad5-9dd8-63cc38f2a2eb:
column "hrn" does not exist
LINE 1: ...M view_nodes WHERE deleted IS False AND (True AND hrn ILIKE ...
... abridged ...
# this can be worked around by just returning 'hrn' as well
>>> ns=GetNodes({'hrn':'ple.*'},['hrn','hostname'])
```

Nodegroups

In earlier versions up to v4.2, *NodeGroups* used to be defined extensively. So you would, basically, create an empty nodegroup instance, and then use *AddNodeToNodeGroup* or *DeleteNodeFromNodeGroup* to manage the nodegroup's contents.

The new model has been redefined as follows. You now define a nodegroup as the set of nodes for which a given *Tag* has a given value, which are defined once and for good when creating the *NodeGroup* object.

So for instance for managing the set of nodes that are running various levels of software code, PLC has defined two *NodeGroups* named `alpha` and `beta`. With the new model, we would now do something like the following, using the built-in `deployment` tag that is created for that purpose:

```
### creating node groups
>>> AddNodeGroup('alphanodes','deployment','alpha')
21
>>> AddNodeGroup('betanodes','deployment','beta')
22
### checking contents (no node has 'deployment' set to either 'alpha' or 'beta' yet)
>>> for ng in GetNodeGroups(['alphanodes','betanodes'],['groupname','node_ids']): pr
{'groupname': u'alphanodes', 'node_ids': []}
{'groupname': u'betanodes', 'node_ids': []}

### displaying node ids
>>> for n in GetNodes({'hostname':'*.inria.fr'},['hostname','node_id']): print n
{'hostname': u'vnode01.inria.fr', 'node_id': 1}
{'hostname': u'vnode02.inria.fr', 'node_id': 2}

### setting 'deployment' for these two nodes
>>> SetNodeDeployment('vnode01.inria.fr','alpha')
>>> for ng in GetNodeGroups(['alphanodes','betanodes'],['groupname','node_ids']): pr
{'groupname': u'alphanodes', 'node_ids': [1]}
{'groupname': u'betanodes', 'node_ids': []}
```

```
>>> SetNodeDeployment('vnode02.inria.fr','beta')

### checking contents again
>>> for ng in GetNodeGroups(['alphanodes','betanodes'],['groupname','node_ids']): pr
{'groupname': u'alphanodes', 'node_ids': [1]}
{'groupname': u'betanodes', 'node_ids': [2]}
```

PlanetLab shell

A command-line program called **plcsh** simplifies authentication structure handling, and is useful for scripting. This program is distributed as a Linux RPM called PLCAPI and requires Python $\geq$2.4.

```
usage: plcsh [options]

options:
  -f CONFIG, --config=CONFIG
                        PLC configuration file
  -h URL, --url=URL     API URL
  -c CACERT, --cacert=CACERT
                        API SSL certificate
  -k INSECURE, --insecure=INSECURE
                        Do not check SSL certificate
  -m METHOD, --method=METHOD
                        API authentication method
  -s SESSION, --session=SESSION
                        API session key
  -u USER, --user=USER  API user name
  -p PASSWORD, --password=PASSWORD
                        API password
  -r ROLE, --role=ROLE  API role
  -x, --xmlrpc          Use XML-RPC interface
  --help                show this help message and exit
```

Specify at least the API URL and your user name:

```
plcsh --url https://www.planet-lab.org/PLCAPI/ -u user@site.edu
```

You will be presented with a prompt. From here, you can invoke API calls and omit the authentication structure, as it will be filled in automatically.

```
user@site.edu connected using password authentication
Type "system.listMethods()" or "help(method)" for more information.
[user@site.edu]>>> AuthCheck()
1
[user@site.edu]>>> GetNodes([121], ['node_id', 'hostname'])
[{'node_id': 121, 'hostname': 'planetlab-1.cs.princeton.edu'}]
```

As this program is actually a Python interpreter, you may create variables, execute for loops, import other packages, etc., directly on the command line as you would using the regular Python shell.

To use **plcsh** programmatically, import the `PLC.Shell` module:

```
#!/usr/bin/python

import sys

# Default location that the PLCAPI RPM installs the PLC class
sys.path.append('/usr/share/plc_api')

# Initialize shell environment. Shell() will define all PLCAPI methods
# in the specified namespace (specifying globals() will define them
# globally).
```

```
from PLC.Shell import Shell
plc = Shell(globals(),
            url = "https://www.planet-lab.org/PLCAPI/",
            user = "user@site.edu",
            password = "password")

# Both are equivalent
nodes = GetNodes([121], ['node_id', 'hostname'])
nodes = plc.GetNodes([121], ['node_id', 'hostname'])
```

Using regular python

It is also possible to write simple regular-python scripts, as illustrated in the example below. The only difference with the examples above is that all API calls need to be passed a first argument for authentication. This example would write in a file the name of all the hosts attached to a given slice.

```
#!/usr/bin/env python

import xmlrpclib

plc_host='www.planet-lab.eu'

slice_name='inria_heartbeat'

auth = { 'AuthMethod' : 'password',
         'Username' : 'thierry.parmentelat@inria.fr',
         'AuthString' : 'xxxxxx',
}

api_url="https://%s:443/PLCAPI/"%plc_host

plc_api = xmlrpclib.ServerProxy(api_url,allow_none=True)

# the slice's node ids
node_ids = plc_api.GetSlices(auth,slice_name,['node_ids'])[0]['node_ids']

# get hostname for these nodes
slice_nodes = plc_api.GetNodes(auth,node_ids,['hostname'])

# store in a file
f=open('mynodes.txt','w')
for node in slice_nodes:
    print >>f,node['hostname']
f.close()
```

Chapter 2. PlanetLab API Methods

AddAddressType

Prototype:

AddAddressType (auth, address_type_fields)

Description:

Adds a new address type. Fields specified in address_type_fields are used.
Returns the new address_type_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *address_type_fields* : struct
  - *name* : string, Address type
  - *description* : string, Address type description

Returns:

- int, New address_type_id (> 0) if successful

AddAddressTypeToAddress

Prototype:

AddAddressTypeToAddress (auth, address_type_id_or_name, address_id)

Description:

Adds an address type to the specified address.
PIs may only update addresses of their own sites.
Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *address_type_id_or_name* : int or string
  - int, Address type identifier
  - string, Address type

- *address_id* : int, Address identifier

Returns:

- int, 1 if successful

AddBootState

Prototype:

AddBootState (auth, name)

Description:

Adds a new node boot state.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `name` : string, Boot state

Returns:

- int, 1 if successful

AddConfFile

Prototype:

AddConfFile (auth, conf_file_fields)

Description:

Adds a new node configuration file. Any fields specified in conf_file_fields are used, otherwise defaults are used.

Returns the new conf_file_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `conf_file_fields` : struct
  - `file_owner` : string, chown(1) owner
  - `postinstall_cmd` : string, Shell command to execute after installing
  - `error_cmd` : string, Shell command to execute if any error occurs
  - `preinstall_cmd` : string, Shell command to execute prior to installing
  - `dest` : string, Absolute path where file should be installed
  - `ignore_cmd_errors` : boolean, Install file anyway even if an error occurs

- `enabled` : boolean, Configuration file is active
- `file_permissions` : string, chmod(1) permissions
- `source` : string, Relative path on the boot server where file can be downloaded
- `always_update` : boolean, Always attempt to install file even if unchanged
- `file_group` : string, chgrp(1) owner

Returns:

- int, New conf_file_id (> 0) if successful

AddConfFileToNode

Prototype:

AddConfFileToNode (auth, conf_file_id, node_id_or_hostname)

Description:

Adds a configuration file to the specified node. If the node is already linked to the configuration file, no errors are returned.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `conf_file_id` : int, Configuration file identifier
- `node_id_or_hostname` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- int, 1 if successful

AddConfFileToNodeGroup

Prototype:

AddConfFileToNodeGroup (auth, conf_file_id, nodegroup_id_or_name)

Description:

Adds a configuration file to the specified node group. If the node group is already linked to the configuration file, no errors are returned.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *conf_file_id* : int, Configuration file identifier
- *nodegroup_id_or_name* : int or string
  - int, Node group identifier
  - string, Node group name

Returns:

- int, 1 if successful

AddIlink

Prototype:

AddIlink (auth, src_if_id, dst_if_id, tag_type_id_or_name, value)

Description:

Create a link between two interfaces The link has a tag type, that needs be created beforehand and an optional value.

Returns the new ilink_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin, pi, tech, user

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *src_if_id* : int, source interface identifier
- *dst_if_id* : int, destination interface identifier
- *tag_type_id_or_name* : int or string
  - int, Node tag type identifier
  - string, Node tag type name

- *value* : string, optional ilink value

Returns:

- int, New ilink_id (> 0) if successful

AddInitScript

Prototype:

AddInitScript (auth, initscript_fields)

Description:

Adds a new initscript. Any fields specified in initscript_fields are used, otherwise defaults are used.

Returns the new initscript_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *initscript_fields* : struct
  - *enabled* : boolean, Initscript is active
  - *name* : string, Initscript name
  - *script* : string, Initscript

Returns:

- int, New initscript_id (> 0) if successful

AddInterface

Prototype:

AddInterface (auth, node_id_or_hostname, interface_fields)

Description:

Adds a new network for a node. Any values specified in interface_fields are used, otherwise defaults are used.

If type is static, then ip, gateway, network, broadcast, netmask, and dns1 must all be specified in interface_fields. If type is dhcp, these parameters, even if specified, are ignored.

PIs and techs may only add interfaces to their own nodes. Admins may add interfaces to any node.

Returns the new interface_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin, pi, tech

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *node_id_or_hostname* : int or string
  - int, Node identifier
  - string, Fully qualified hostname

- *interface_fields* : struct
  - *last_updated* : int, Date and time when node entry was created
  - *network* : string, Subnet address
  - *is_primary* : boolean, Is the primary interface for this node
  - *dns1* : string, IP address of primary DNS server

- *hostname* : string, (Optional) Hostname
- *mac* : string, MAC address
- *interface_tag_ids* : array, List of interface settings
  - int

- *bwlimit* : int, Bandwidth limit
- *broadcast* : string, Network broadcast address
- *method* : string, Addressing method (e.g., 'static' or 'dhcp')
- *netmask* : string, Subnet mask
- *dns2* : string, IP address of secondary DNS server
- *ip* : string, IP address
- *ifname* : string, accessor
- *type* : string, Address type (e.g., 'ipv4')
- *gateway* : string, IP address of primary gateway

Returns:

- int, New interface_id (> 0) if successful

AddInterfaceTag

Prototype:

AddInterfaceTag (auth, interface_id, tag_type_id_or_name, value)

Description:

Sets the specified setting for the specified interface to the specified value.

Admins have full access. Non-admins need to (1) have at least one of the roles attached to the tagtype, and (2) belong in the same site as the tagged subject.

Returns the new interface_tag_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin, pi, tech, user

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *interface_id* : int, Node interface identifier
- *tag_type_id_or_name* : int or string
  - int, Node tag type identifier
  - string, Node tag type name

- *value* : string, Interface setting value

Returns:

- int, New interface_tag_id (> 0) if successful

AddKeyType

Prototype:

AddKeyType (auth, name)

Description:

Adds a new key type.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `name` : string, Key type

Returns:

- int, 1 if successful

AddLeases

Prototype:

AddLeases (auth, node_id_or_hostname_s, slice_id_or_name, t_from, t_until)

Description:

Adds a new lease. Mandatory arguments are node(s), slice, t_from and t_until times can be either integers, datetime's, or human readable (see Timestamp)

PIs may only add leases associated with their own sites (i.e., to a slice that belongs to their site). Users may only add leases associated with their own slices.

Returns the new lease_ids if successful, faults otherwise.

Allowed Roles:

admin, pi, user

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `node_id_or_hostname_s` : int or array of int or string or array of string
  - int, Node identifier
  - array of int, Node identifier
  - string, Fully qualified hostname
  - array of string, Fully qualified hostname
- `slice_id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

- *t_from* : int or string
  - int, timeslot start (unix timestamp)
  - string, timeslot start (formatted as %Y-%m-%d %H:%M:%S)

- *t_until* : int or string
  - int, timeslot end (unix timestamp)
  - string, timeslot end (formatted as %Y-%m-%d %H:%M:%S)

Returns:

- struct, 'new_ids' is the list of newly created ids, 'errors' is a list of error strings

AddMessage

Prototype:

AddMessage (auth, message_fields)

Description:

Adds a new message template. Any values specified in message_fields are used, otherwise defaults are used.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *message_fields* : struct
  - *enabled* : boolean, Message is enabled
  - *message_id* : string, Message identifier
  - *template* : string, Message template
  - *subject* : string, Message summary

Returns:

- int, 1 if successful

AddNetworkMethod

Prototype:

AddNetworkMethod (auth, name)

Description:

Adds a new network method.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `name` : string, Network method

Returns:

- int, 1 if successful

AddNetworkType

Prototype:

AddNetworkType (auth, name)

Description:

Adds a new network type.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `name` : string, Network type

Returns:

- int, 1 if successful

AddNode

Prototype:

AddNode (auth, site_id_or_login_base, node_fields)

Description:

Adds a new node. Any values specified in node_fields are used, otherwise defaults are used.

PIs and techs may only add nodes to their own sites. Admins may add nodes to any site.

Returns the new node_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- *site_id_or_login_base* : int or string
  - int, Site identifier
  - string, Site slice prefix

- *node_fields* : struct
  - *hrn* : string, accessor
  - *fcdistro* : string, accessor
  - *boot_state* : string, Boot state
  - *virt* : string, accessor
  - *hostname* : string, Fully qualified hostname
  - *node_type* : string, Node type
  - *version* : string, Apparent Boot CD version
  - *extensions* : string, accessor
  - *pldistro* : string, accessor
  - *deployment* : string, accessor
  - *model* : string, Make and model of the actual machine
  - *arch* : string, accessor

Returns:

- int, New node_id (> 0) if successful

AddNodeGroup

Prototype:

AddNodeGroup (auth, groupname, tag_type_id_or_tagname, value)

Description:

Adds a new node group. Any values specified in nodegroup_fields are used, otherwise defaults are used.

Returns the new nodegroup_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *groupname* : string, Node group name
- *tag_type_id_or_tagname* : int or string
  - int, Node tag type identifier
  - string, Node tag type name

- *value* : string, Node tag value

Returns:

- int, New nodegroup_id (> 0) if successful

AddNodeTag

Prototype:

AddNodeTag (auth, node_id, tag_type_id_or_name, value)

Description:

Sets the specified tag for the specified node to the specified value.

Admins have full access. Non-admins need to (1) have at least one of the roles attached to the tagtype, and (2) belong in the same site as the tagged subject.

Returns the new node_tag_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin, pi, tech, user

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *node_id* : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- *tag_type_id_or_name* : int or string
  - int, Node tag type identifier
  - string, Node tag type name
- *value* : string, Node tag value

Returns:

- int, New node_tag_id (> 0) if successful

AddNodeToPCU

Prototype:

AddNodeToPCU (auth, node_id_or_hostname, pcu_id, port)

Description:

Adds a node to a port on a PCU. Faults if the node has already been added to the PCU or if the port is already in use.

Non-admins may only update PCUs at their sites.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech

Parameters:

- *auth* : struct, API authentication structure

- `AuthMethod` : string, Authentication method to use

- `node_id_or_hostname` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

- `pcu_id` : int, PCU identifier
- `port` : int, PCU port number

Returns:

- int, 1 if successful

AddNodeType

Prototype:

AddNodeType (auth, name)

Description:

Adds a new node node type.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `name` : string, Node type

Returns:

- int, 1 if successful

AddPCU

Prototype:

AddPCU (auth, site_id_or_login_base, pcu_fields)

Description:

Adds a new power control unit (PCU) to the specified site. Any fields specified in pcu_fields are used, otherwise defaults are used.

PIs and technical contacts may only add PCUs to their own sites.

Returns the new pcu_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure

  - `AuthMethod` : string, Authentication method to use

- `site_id_or_login_base` : int or string
  - int, Site identifier
  - string, Site slice prefix

- `pcu_fields` : struct
  - `username` : string, PCU username
  - `protocol` : string, PCU protocol, e.g. ssh, https, telnet
  - `ip` : string, PCU IP address
  - `notes` : string, Miscellaneous notes
  - `hostname` : string, PCU hostname
  - `model` : string, PCU model string
  - `password` : string, PCU username

Returns:

- int, New pcu_id (> 0) if successful

AddPCUProtocolType

Prototype:

AddPCUProtocolType (auth, pcu_type_id_or_model, protocol_type_fields)

Description:

Adds a new pcu protocol type.
Returns the new pcu_protocol_type_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `pcu_type_id_or_model` : int or string
  - int, PCU Type Identifier
  - string, PCU model

- `protocol_type_fields` : struct
  - `supported` : boolean, Is the port/protocol supported by PLC
  - `protocol` : string, Protocol
  - `port` : int, PCU port
  - `pcu_type_id` : int, PCU type identifier

Returns:

- int, New pcu_protocol_type_id (> 0) if successful

AddPCUType

Prototype:

AddPCUType (auth, pcu_type_fields)

Description:

Adds a new pcu type.
Returns the new pcu_type_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *pcu_type_fields* : struct
  - *model* : string, PCU model
  - *name* : string, PCU full name

Returns:

- int, New pcu_type_id (> 0) if successful

AddPeer

Prototype:

AddPeer (auth, peer_fields)

Description:

Adds a new peer.
Returns the new peer_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *peer_fields* : struct
  - *peername* : string, Peer name
  - *peer_url* : string, Peer API URL
  - *key* : string, Peer GPG public key
  - *hrn_root* : string, Root of this peer in a hierarchical naming space
  - *cacert* : string, Peer SSL public certificate

- *shortname* : string, Peer short name

Returns:

- int, New peer_id (> 0) if successful

AddPerson

Prototype:

AddPerson (auth, person_fields)

Description:

Adds a new account. Any fields specified in person_fields are used, otherwise defaults are used.

Accounts are disabled by default. To enable an account, use UpdatePerson().

Returns the new person_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin, pi

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *person_fields* : struct
  - *bio* : string, Biography
  - *first_name* : string, Given name
  - *last_name* : string, Surname
  - *title* : string, Title
  - *url* : string, Home page
  - *phone* : string, Telephone number
  - *hrn* : string, accessor
  - *sfa_created* : string, accessor
  - *showconf* : string, accessor
  - *columnconf* : string, accessor
  - *password* : string, Account password in crypt() form
  - *email* : string, Primary e-mail address
  - *advanced* : string, accessor

Returns:

- int, New person_id (> 0) if successful

AddPersonKey

Prototype:

AddPersonKey (auth, person_id_or_email, key_fields)

Description:

Adds a new key to the specified account.
Non-admins can only modify their own keys.
Returns the new key_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin, pi, tech, user

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `person_id_or_email` : int or string
  - int, User identifier
  - string, Primary e-mail address
- `key_fields` : struct
  - `key_type` : string, Key type
  - `key` : string, Key value

Returns:

- int, New key_id (> 0) if successful

AddPersonTag

Prototype:

AddPersonTag (auth, person_id, tag_type_id_or_name, value)

Description:

Sets the specified setting for the specified person to the specified value.
Admins have full access. Non-admins can change their own tags.
Returns the new person_tag_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin, pi, tech, user

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `person_id` : int, User identifier
- `tag_type_id_or_name` : int or string
  - int, Node tag type identifier

  - string, Node tag type name

- *value* : string, Person setting value

Returns:

- int, New person_tag_id (> 0) if successful

AddPersonToSite

Prototype:

AddPersonToSite (auth, person_id_or_email, site_id_or_login_base)

Description:

Adds the specified person to the specified site. If the person is already a member of the site, no errors are returned. Does not change the person's primary site.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *person_id_or_email* : int or string
  - int, User identifier
  - string, Primary e-mail address

- *site_id_or_login_base* : int or string
  - int, Site identifier
  - string, Site slice prefix

Returns:

- int, 1 if successful

AddPersonToSlice

Prototype:

AddPersonToSlice (auth, person_id_or_email, slice_id_or_name)

Description:

Adds the specified person to the specified slice. If the person is already a member of the slice, no errors are returned.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *person_id_or_email* : int or string
  - int, User identifier
  - string, Primary e-mail address

- *slice_id_or_name* : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- int, 1 if successful

AddRole

Prototype:

AddRole (auth, role_id, name)

Description:

Adds a new role.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *role_id* : int, Role identifier
- *name* : string, Role

Returns:

- int, 1 if successful

AddRoleToPerson

Prototype:

AddRoleToPerson (auth, role_id_or_name, person_id_or_email)

Description:

Grants the specified role to the person.

PIs can only grant the tech and user roles to users and techs at their sites. Admins can grant any role to any user.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `role_id_or_name` : int or string
  - int, Role identifier
  - string, Role
- `person_id_or_email` : int or string
  - int, User identifier
  - string, Primary e-mail address

Returns:

- int, 1 if successful

AddRoleToTagType

Prototype:

AddRoleToTagType (auth, role_id_or_name, tag_type_id_or_tagname)

Description:

Add the specified role to the tagtype so that users with that role can tweak the tag.
Only admins can call this method
Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `role_id_or_name` : int or string
  - int, Role identifier
  - string, Role
- `tag_type_id_or_tagname` : int or string
  - int, Node tag type identifier
  - string, Node tag type name

Returns:

- int, 1 if successful

AddSession

Prototype:

AddSession (auth, person_id_or_email)

Description:

Creates and returns a new session key for the specified user. (Used for website 'user sudo')

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `person_id_or_email` : int or string
  - int, User identifier
  - string, Primary e-mail address

Returns:

- string, Session key

AddSite

Prototype:

AddSite (auth, site_fields)

Description:

Adds a new site, and creates a node group for that site. Any fields specified in site_fields are used, otherwise defaults are used.

Returns the new site_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `site_fields` : struct
  - `name` : string, Full site name
  - `url` : string, URL of a page that describes the site
  - `enabled` : boolean, Has been enabled
  - `longitude` : double, Decimal longitude of the site
  - `max_slivers` : int, Maximum number of slivers that the site is able to create
  - `max_slices` : int, Maximum number of slices that the site is able to create
  - `login_base` : string, Site slice prefix
  - `ext_consortium_id` : int, external consortium id

- *latitude* : double, Decimal latitude of the site
- *is_public* : boolean, Publicly viewable site
- *abbreviated_name* : string, Abbreviated site name

Returns:

- int, New site_id (> 0) if successful

AddSiteAddress

Prototype:

AddSiteAddress (auth, site_id_or_login_base, address_fields)

Description:

Adds a new address to a site. Fields specified in address_fields are used; some are not optional.

PIs may only add addresses to their own sites.

Returns the new address_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin, pi

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *site_id_or_login_base* : int or string
  - int, Site identifier
  - string, Site slice prefix
- *address_fields* : struct
  - *city* : string, City
  - *country* : string, Country
  - *line3* : string, Address line 3
  - *line2* : string, Address line 2
  - *line1* : string, Address line 1
  - *state* : string, State or province
  - *postalcode* : string, Postal code

Returns:

- int, New address_id (> 0) if successful

AddSiteTag

Prototype:

AddSiteTag (auth, site_id, tag_type_id_or_name, value)

Description:

Sets the specified setting for the specified site to the specified value.

Admins have full access. Non-admins need to (1) have at least one of the roles attached to the tagtype, and (2) belong in the same site as the tagged subject.

Returns the new site_tag_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin, pi, tech, user

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *site_id* : int, Site identifier
- *tag_type_id_or_name* : int or string
  - int, Node tag type identifier
  - string, Node tag type name
- *value* : string, Site setting value

Returns:

- int, New site_tag_id (> 0) if successful

AddSlice

Prototype:

AddSlice (auth, slice_fields)

Description:

Adds a new slice. Any fields specified in slice_fields are used, otherwise defaults are used.

Valid slice names are lowercase and begin with the login_base (slice prefix) of a valid site, followed by a single underscore. Thereafter, only letters, numbers, or additional underscores may be used.

PIs may only add slices associated with their own sites (i.e., slice prefixes must always be the login_base of one of their sites).

Returns the new slice_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin, pi

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *slice_fields* : struct

- *enable_hmac* : string, accessor
- *description* : string, Slice description
- *initscript* : string, accessor
- *ipv6_address* : string, accessor
- *pldistro* : string, accessor
- *arch* : string, accessor
- *vref* : string, accessor
- *hrn* : string, accessor
- *instantiation* : string, Slice instantiation state
- *fcdistro* : string, accessor
- *name* : string, Slice name
- *url* : string, URL further describing this slice
- *max_nodes* : int, Maximum number of nodes that can be assigned to this slice
- *initscript_code* : string, accessor
- *omf_control* : string, accessor
- *sfa_created* : string, accessor

Returns:

- int, New slice_id (> 0) if successful

AddSliceInstantiation

Prototype:

AddSliceInstantiation (auth, name)

Description:

Adds a new slice instantiation state.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *name* : string, Slice instantiation state

Returns:

- int, 1 if successful

AddSliceTag

Prototype:

AddSliceTag (auth, slice_id_or_name, tag_type_id_or_name, value, node_id_or_hostname, nodegroup_id_or_name)

Description:

Sets the specified tag of the slice to the specified value. If nodegroup is specified, this applies to all slivers of that group. If node is specified, this only applies to a sliver.

Admins have full access, including on nodegroups.

Non-admins need to have at least one of the roles attached to the tagtype. In addition: (*) Users may only set tags of slices or slivers of which they are members. (*) PIs may only set tags of slices in their site (*) techs cannot use this method

Returns the new slice_tag_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin, pi, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `slice_id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `tag_type_id_or_name` : int or string
  - int, Node tag type identifier
  - string, Node tag type name
- `value` : string or string
  - string, Slice attribute value
  - string, Initscript name
- `node_id_or_hostname` : int or string or nil
  - int, Node identifier
  - string, Fully qualified hostname
  - nil
- `nodegroup_id_or_name` : int or string
  - int, Node group identifier
  - string, Node group name

Returns:

- int, New slice_tag_id (> 0) if successful

AddSliceToNodes

Prototype:

AddSliceToNodes (auth, slice_id_or_name, node_id_or_hostname_list)

Description:

Adds the specified slice to the specified nodes. Nodes may be either local or foreign nodes.

If the slice is already associated with a node, no errors are returned.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, user

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `slice_id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `node_id_or_hostname_list` : array of int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- int, 1 if successful

AddSliceToNodesWhitelist

Prototype:

AddSliceToNodesWhitelist (auth, slice_id_or_name, node_id_or_hostname_list)

Description:

Adds the specified slice to the whitelist on the specified nodes. Nodes may be either local or foreign nodes.

If the slice is already associated with a node, no errors are returned.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `slice_id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `node_id_or_hostname_list` : array of int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- int, 1 if successful

AddTagType

Prototype:

AddTagType (auth, tag_type_fields)

Description:

Adds a new type of node tag. Any fields specified are used, otherwise defaults are used.

Returns the new node_tag_id (> 0) if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *tag_type_fields* : struct
  - *category* : string, Node tag category
  - *description* : string, Node tag type description
  - *tagname* : string, Node tag type name

Returns:

- int, New node_tag_id (> 0) if successful

AuthCheck

Prototype:

AuthCheck (auth)

Description:

Returns 1 if the user or node authenticated successfully, faults otherwise.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

Returns:

- int, 1 if successful

BindObjectToPeer

Prototype:

BindObjectToPeer (auth, object_type, object_id, shortname, remote_object_id)

Description:

This method is a hopefully temporary hack to let the sfa correctly attach the objects it creates to a remote peer object. This is needed so that the sfa federation link can work in parallel with RefreshPeer, as RefreshPeer depends on remote objects being correctly marked.

BindRemoteObjectToPeer is allowed to admins only.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *object_type* : string, Object type, among 'site','person','slice','node','key'
- *object_id* : int, object_id
- *shortname* : string, peer shortname
- *remote_object_id* : int, remote object_id, set to 0 if unknown

Returns:

- int, 1 if successful

BlacklistKey

Prototype:

BlacklistKey (auth, key_id)

Description:

Blacklists a key, disassociating it and all others identical to it from all accounts and preventing it from ever being added again.

WARNING: Identical keys associated with other accounts with also be blacklisted.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *key_id* : int, Key identifier

Returns:

- int, 1 if successful

BootGetNodeDetails

Prototype:

BootGetNodeDetails (auth)

Description:

Returns a set of details about the calling node, including a new node session value.

Allowed Roles:

node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use, always 'hmac'
  - `value` : string, HMAC of node key and method call
  - `node_id` : int, Node identifier

Returns:

- struct
  - `boot_state` : string, Boot state
  - `model` : string, Make and model of the actual machine
  - `hostname` : string, Fully qualified hostname
  - `networks` : array of struct
    - `is_primary` : boolean, Is the primary interface for this node
    - `last_updated` : int, Date and time when node entry was created
    - `network` : string, Subnet address
    - `ip` : string, IP address
    - `dns1` : string, IP address of primary DNS server
    - `hostname` : string, (Optional) Hostname
    - `netmask` : string, Subnet mask
    - `interface_tag_ids` : array, List of interface settings
      - int

    - `interface_id` : int, Node interface identifier
    - `broadcast` : string, Network broadcast address
    - `mac` : string, MAC address
    - `node_id` : int, Node associated with this interface
    - `gateway` : string, IP address of primary gateway
    - `dns2` : string, IP address of secondary DNS server
    - `bwlimit` : int, Bandwidth limit
    - `type` : string, Address type (e.g., 'ipv4')
    - `method` : string, Addressing method (e.g., 'static' or 'dhcp')

  - `session` : string, Session key

BootNotifyOwners

Prototype:

BootNotifyOwners (auth, message_id, include_pis, include_techs, include_support)

Description:

Notify the owners of the node, and/or support about an event that happened on the machine.

Returns 1 if successful.

Allowed Roles:

node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `message_id` : string, Message identifier
- `include_pis` : int, Notify PIs
- `include_techs` : int, Notify technical contacts
- `include_support` : int, Notify support

Returns:

- int, 1 if successful

BootUpdateNode

Prototype:

BootUpdateNode (auth, node_fields)

Description:

Allows the calling node to update its own record. Only the primary network can be updated, and the node IP cannot be changed.

Returns 1 if updated successfully.

Allowed Roles:

node

Parameters:

- `auth` : struct or struct
  - struct, API authentication structure
    - `AuthMethod` : string, Authentication method to use, always 'hmac'
    - `value` : string, HMAC of node key and method call
    - `node_id` : int, Node identifier
  - struct, API authentication structure
    - `session` : string, Session key
    - `AuthMethod` : string, Authentication method to use, always 'session'

- *node_fields* : struct
  - *boot_state* : string, Boot state
  - *ssh_rsa_key* : string, Last known SSH host key
  - *primary_network* : struct
    - *network* : string, Subnet address
    - *dns2* : string, IP address of secondary DNS server
    - *dns1* : string, IP address of primary DNS server
    - *netmask* : string, Subnet mask
    - *method* : string, Addressing method (e.g., 'static' or 'dhcp')
    - *broadcast* : string, Network broadcast address
    - *mac* : string, MAC address
    - *gateway* : string, IP address of primary gateway
  - *ssh_host_key* : string, Last known SSH host key

Returns:

- int, 1 if successful

DeleteAddress

Prototype:

DeleteAddress (auth, address_id)

Description:

Deletes an address.
PIs may only delete addresses from their own sites.
Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *address_id* : int, Address identifier

Returns:

- int, 1 if successful

DeleteAddressType

Prototype:

DeleteAddressType (auth, address_type_id_or_name)

Description:

Deletes an address type.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `address_type_id_or_name` : int or string
  - int, Address type identifier
  - string, Address type

Returns:

- int, 1 if successful

DeleteAddressTypeFromAddress

Prototype:

DeleteAddressTypeFromAddress (auth, address_type_id_or_name, address_id)

Description:

Deletes an address type from the specified address.
PIs may only update addresses of their own sites.
Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `address_type_id_or_name` : int or string
  - int, Address type identifier
  - string, Address type
- `address_id` : int, Address identifier

Returns:

- int, 1 if successful

DeleteBootState

Prototype:

DeleteBootState (auth, name)

Description:

Deletes a node boot state.

WARNING: This will cause the deletion of all nodes in this boot state.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `name` : string, Boot state

Returns:

- int, 1 if successful

DeleteConfFile

Prototype:

DeleteConfFile (auth, conf_file_id)

Description:

Returns an array of structs containing details about node configuration files. If conf_file_ids is specified, only the specified configuration files will be queried.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `conf_file_id` : int, Configuration file identifier

Returns:

- int, 1 if successful

DeleteConfFileFromNode

Prototype:

DeleteConfFileFromNode (auth, conf_file_id, node_id_or_hostname)

Description:

Deletes a configuration file from the specified node. If the node is not linked to the configuration file, no errors are returned.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *conf_file_id* : int, Configuration file identifier
- *node_id_or_hostname* : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- int, 1 if successful

DeleteConfFileFromNodeGroup

Prototype:

DeleteConfFileFromNodeGroup (auth, conf_file_id, nodegroup_id_or_name)

Description:

Deletes a configuration file from the specified nodegroup. If the nodegroup is not linked to the configuration file, no errors are returned.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *conf_file_id* : int, Configuration file identifier
- *nodegroup_id_or_name* : int or string
  - int, Node group identifier
  - string, Node group name

Returns:

- int, 1 if successful

DeleteIlink

Prototype:

DeleteIlink (auth, ilink_id)

Description:

Deletes the specified ilink

Attributes may require the caller to have a particular role in order to be deleted, depending on the related tag type. Admins may delete attributes of any slice or sliver.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, user

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `ilink_id` : int, ilink identifier

Returns:

- int, 1 if successful

DeleteInitScript

Prototype:

DeleteInitScript (auth, initscript_id_or_name)

Description:

Deletes an existing initscript.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `initscript_id_or_name` : int or string
  - int, Initscript identifier
  - string, Initscript name

Returns:

- int, 1 if successful

DeleteInterface

Prototype:

DeleteInterface (auth, interface_id)

Description:

Deletes an existing interface.

Admins may delete any interface. PIs and techs may only delete interface interfaces associated with nodes at their sites.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *interface_id* : int, Node interface identifier

Returns:

- int, 1 if successful

DeleteInterfaceTag

Prototype:

DeleteInterfaceTag (auth, interface_tag_id)

Description:

Deletes the specified interface setting

Admins have full access. Non-admins need to (1) have at least one of the roles attached to the tagtype, and (2) belong in the same site as the tagged subject.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, user, tech

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *interface_tag_id* : int, Interface setting identifier

Returns:

- int, 1 if successful

DeleteKey

Prototype:

DeleteKey (auth, key_id)

Description:

Deletes a key.

Non-admins may only delete their own keys.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech, user

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *key_id* : int, Key identifier

Returns:

- int, 1 if successful

DeleteKeyType

Prototype:

DeleteKeyType (auth, name)

Description:

Deletes a key type.

WARNING: This will cause the deletion of all keys of this type.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `name` : string, Key type

Returns:

- int, 1 if successful

DeleteLeases

Prototype:

DeleteLeases (auth, lease_ids)

Description:

Deletes a lease.

Users may only delete leases attached to their slices. PIs may delete any of the leases for slices at their sites, or any slices of which they are members. Admins may delete any lease.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech, user

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `lease_ids` : int or array of int
  - int, Lease identifier
  - array of int, Lease identifier

Returns:

- int, 1 if successful

DeleteMessage

Prototype:

DeleteMessage (auth, message_id)

Description:

Deletes a message template.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `message_id` : string, Message identifier

Returns:

- int, 1 if successful

DeleteNetworkMethod

Prototype:

DeleteNetworkMethod (auth, name)

Description:

Deletes a network method.

WARNING: This will cause the deletion of all network interfaces that use this method.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `name` : string, Network method

Returns:

- int, 1 if successful

DeleteNetworkType

Prototype:

DeleteNetworkType (auth, name)

Description:

Deletes a network type.

WARNING: This will cause the deletion of all network interfaces that use this type.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `name` : string, Network type

Returns:

- int, 1 if successful

DeleteNode

Prototype:

DeleteNode (auth, node_id_or_hostname)

Description:

Mark an existing node as deleted.

PIs and techs may only delete nodes at their own sites. ins may delete nodes at any site.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `node_id_or_hostname` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- int, 1 if successful

DeleteNodeFromPCU

Prototype:

DeleteNodeFromPCU (auth, node_id_or_hostname, pcu_id)

Description:

Deletes a node from a PCU.
Non-admins may only update PCUs at their sites.
Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `node_id_or_hostname` : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- `pcu_id` : int, PCU identifier

Returns:

- int, 1 if successful

DeleteNodeGroup

Prototype:

DeleteNodeGroup (auth, node_group_id_or_name)

Description:

Delete an existing Node Group.
ins may delete any node group
Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `node_group_id_or_name` : int or string
  - int, Node group identifier
  - string, Node group name

Returns:

- int, 1 if successful

DeleteNodeTag

Prototype:

DeleteNodeTag (auth, node_tag_id)

Description:

Deletes the specified node tag

Admins have full access. Non-admins need to (1) have at least one of the roles attached to the tagtype, and (2) belong in the same site as the tagged subject.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, user, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `node_tag_id` : int, Node tag identifier

Returns:

- int, 1 if successful

DeleteNodeType

Prototype:

DeleteNodeType (auth, name)

Description:

Deletes a node node type.

WARNING: This will cause the deletion of all nodes in this boot state.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `name` : string, Node type

Returns:

- int, 1 if successful

DeletePCU

Prototype:

DeletePCU (auth, pcu_id)

Description:

Deletes a PCU.
Non-admins may only delete PCUs at their sites.
Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `pcu_id` : int, PCU identifier

Returns:

- int, 1 if successful

DeletePCUProtocolType

Prototype:

DeletePCUProtocolType (auth, protocol_type_id)

Description:

Deletes a PCU protocol type.
Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `protocol_type_id` : int, PCU protocol type identifier

Returns:

- int, 1 if successful

DeletePCUType

Prototype:

DeletePCUType (auth, pcu_type_id)

Description:

Deletes a PCU type.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `pcu_type_id` : int, PCU Type Identifier

Returns:

- int, 1 if successful

DeletePeer

Prototype:

DeletePeer (auth, peer_id_or_name)

Description:

Mark an existing peer as deleted. All entities (e.g., slices, keys, nodes, etc.) for which this peer is authoritative will also be deleted or marked as deleted.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `peer_id_or_name` : int or string
  - int, Peer identifier
  - string, Peer name

Returns:

- int, 1 if successful

DeletePerson

Prototype:

DeletePerson (auth, person_id_or_email)

Description:

Mark an existing account as deleted.

Users and techs can only delete themselves. PIs can only delete themselves and other non-PIs at their sites. ins can delete anyone.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, user, tech

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *person_id_or_email* : int or string
  - int, User identifier
  - string, Primary e-mail address

Returns:

- int, 1 if successful

DeletePersonFromSite

Prototype:

DeletePersonFromSite (auth, person_id_or_email, site_id_or_login_base)

Description:

Removes the specified person from the specified site. If the person is not a member of the specified site, no error is returned.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *person_id_or_email* : int or string
  - int, User identifier
  - string, Primary e-mail address

- *site_id_or_login_base* : int or string
  - int, Site identifier
  - string, Site slice prefix

Returns:

- int, 1 if successful

DeletePersonFromSlice

Prototype:

DeletePersonFromSlice (auth, person_id_or_email, slice_id_or_name)

Description:

Deletes the specified person from the specified slice. If the person is not a member of the slice, no errors are returned.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `person_id_or_email` : int or string
  - int, User identifier
  - string, Primary e-mail address
- `slice_id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- int, 1 if successful

DeletePersonTag

Prototype:

DeletePersonTag (auth, person_tag_id)

Description:

Deletes the specified person setting

Admins have full access. Non-admins can change their own tags.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, user

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `person_tag_id` : int, Person setting identifier

Returns:

- int, 1 if successful

DeleteRole

Prototype:

DeleteRole (auth, role_id_or_name)

Description:

Deletes a role.

WARNING: This will remove the specified role from all accounts that possess it, and from all node and slice attributes that refer to it.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `role_id_or_name` : int or string
  - int, Role identifier
  - string, Role

Returns:

- int, 1 if successful

DeleteRoleFromPerson

Prototype:

DeleteRoleFromPerson (auth, role_id_or_name, person_id_or_email)

Description:

Deletes the specified role from the person.

PIs can only revoke the tech and user roles from users and techs at their sites. ins can revoke any role from any user.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `role_id_or_name` : int or string
  - int, Role identifier
  - string, Role
- `person_id_or_email` : int or string
  - int, User identifier

- string, Primary e-mail address

Returns:

- int, 1 if successful

DeleteRoleFromTagType

Prototype:

DeleteRoleFromTagType (auth, role_id_or_name, tag_type_id_or_tagname)

Description:

Delete the specified role from the tagtype so that users with that role can no longer tweak the tag.

Only admins can call this method

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `role_id_or_name` : int or string
  - int, Role identifier
  - string, Role

- `tag_type_id_or_tagname` : int or string
  - int, Node tag type identifier
  - string, Node tag type name

Returns:

- int, 1 if successful

DeleteSession

Prototype:

DeleteSession (auth)

Description:

Invalidates the current session.

Returns 1 if successful.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- *auth* : struct, API authentication structure
  - *session* : string, Session key
  - *AuthMethod* : string, Authentication method to use, always 'session'

Returns:

- int, 1 if successful

DeleteSite

Prototype:

DeleteSite (auth, site_id_or_login_base)

Description:

Mark an existing site as deleted. The accounts of people who are not members of at least one other non-deleted site will also be marked as deleted. Nodes, PCUs, and slices associated with the site will be deleted.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *site_id_or_login_base* : int or string
  - int, Site identifier
  - string, Site slice prefix

Returns:

- int, 1 if successful

DeleteSiteTag

Prototype:

DeleteSiteTag (auth, site_tag_id)

Description:

Deletes the specified site setting

Admins have full access. Non-admins need to (1) have at least one of the roles attached to the tagtype, and (2) belong in the same site as the tagged subject.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, user

Parameters:

- *auth* : struct, API authentication structure

- `AuthMethod` : string, Authentication method to use

- `site_tag_id` : int, Site setting identifier

Returns:

- int, 1 if successful

DeleteSlice

Prototype:

DeleteSlice (auth, slice_id_or_name)

Description:

Deletes the specified slice.

Users may only delete slices of which they are members. PIs may delete any of the slices at their sites, or any slices of which they are members. Admins may delete any slice.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, user

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `slice_id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- int, 1 if successful

DeleteSliceFromNodes

Prototype:

DeleteSliceFromNodes (auth, slice_id_or_name, node_id_or_hostname_list)

Description:

Deletes the specified slice from the specified nodes. If the slice is not associated with a node, no errors are returned.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, user

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- *slice_id_or_name* : int or string
  - int, Slice identifier
  - string, Slice name

- *node_id_or_hostname_list* : array of int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- int, 1 if successful

DeleteSliceFromNodesWhitelist

Prototype:

DeleteSliceFromNodesWhitelist (auth, slice_id_or_name, node_id_or_hostname_list)

Description:

Deletes the specified slice from the whitelist on the specified nodes. Nodes may be either local or foreign nodes.

If the slice is already associated with a node, no errors are returned.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *slice_id_or_name* : int or string
  - int, Slice identifier
  - string, Slice name

- *node_id_or_hostname_list* : array of int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- int, 1 if successful

DeleteSliceInstantiation

Prototype:

DeleteSliceInstantiation (auth, instantiation)

Description:

Deletes a slice instantiation state.

WARNING: This will cause the deletion of all slices of this instantiation.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `instantiation` : string, Slice instantiation state

Returns:

- int, 1 if successful

DeleteSliceTag

Prototype:

DeleteSliceTag (auth, slice_tag_id)

Description:

Deletes the specified slice or sliver attribute.

Attributes may require the caller to have a particular role in order to be deleted. Users may only delete attributes of slices or slivers of which they are members. PIs may only delete attributes of slices or slivers at their sites, or of which they are members. Admins may delete attributes of any slice or sliver.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `slice_tag_id` : int, Slice tag identifier

Returns:

- int, 1 if successful

DeleteTagType

Prototype:

DeleteTagType (auth, tag_type_id_or_name)

Description:

Deletes the specified node tag type.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *tag_type_id_or_name* : int or string
  - int, Node tag type identifier
  - string, Node tag type name

Returns:

- int, 1 if successful

GenerateNodeConfFile

Prototype:

GenerateNodeConfFile (auth, node_id_or_hostname, regenerate_node_key)

Description:

Creates a new node configuration file if all network settings are present. This function will generate a new node key for the specified node, effectively invalidating any old configuration files.

Non-admins can only generate files for nodes at their sites.

Returns the contents of the file if successful, faults otherwise.

Allowed Roles:

admin, pi, tech

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *node_id_or_hostname* : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- *regenerate_node_key* : boolean, True if you want to regenerate node key

Returns:

- string, Node configuration file

GetAddressTypes

Prototype:

GetAddressTypes (auth, address_type_filter, return_fields)

Description:

Returns an array of structs containing details about address types. If address_type_filter is specified and is an array of address type identifiers, or a struct of address type attributes, only address types matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `address_type_filter` : array of int or string or struct
  - array of int or string
    - int, Address type identifier
    - string, Address type

  - struct, Attribute filter
    - `name` : string or array of string
      - string, Address type
      - array of string, Address type

    - `address_type_id` : int or array of int
      - int, Address type identifier
      - array of int, Address type identifier

    - `description` : string or array of string
      - string, Address type description
      - array of string, Address type description

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `name` : string, Address type
  - `address_type_id` : int, Address type identifier

- `description` : string, Address type description

GetAddresses

Prototype:

GetAddresses (auth, address_filter, return_fields)

Description:

Returns an array of structs containing details about addresses. If address_filter is specified and is an array of address identifiers, or a struct of address attributes, only addresses matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `address_filter` : array of int or struct
  - array of int, Address identifier
  - struct, Attribute filter
    - `city` : string or array of string
      - string, City
      - array of string, City

    - `address_id` : int or array of int
      - int, Address identifier
      - array of int, Address identifier

    - `country` : string or array of string
      - string, Country
      - array of string, Country

    - `line3` : string or array of string
      - string, Address line 3
      - array of string, Address line 3

    - `line2` : string or array of string
      - string, Address line 2
      - array of string, Address line 2

    - `line1` : string or array of string
      - string, Address line 1
      - array of string, Address line 1

- *address_type_ids* : array or array of array
  - array, Address type identifiers
    - int
  - array of array, Address type identifiers
    - int
- *state* : string or array of string
  - string, State or province
  - array of string, State or province
- *postalcode* : string or array of string
  - string, Postal code
  - array of string, Postal code
- *address_types* : array or array of array
  - array, Address types
    - string
  - array of array, Address types
    - string
- *return_fields* : array, List of fields to return
  - string

Returns:

- array of struct
  - *address_type_ids* : array, Address type identifiers
    - int
  - *city* : string, City
  - *address_id* : int, Address identifier
  - *state* : string, State or province
  - *postalcode* : string, Postal code
  - *country* : string, Country
  - *address_types* : array, Address types
    - string
  - *line3* : string, Address line 3

- *line2* : string, Address line 2
- *line1* : string, Address line 1

GetBootMedium

Prototype:

GetBootMedium (auth, node_id_or_hostname, action, filename, options)

Description:

This method is a redesign based on former, supposedly dedicated, AdmGenerateNodeConfFile

As compared with its ancestor, this method provides a much more detailed interface, that allows to (*) either just preview the node config file -- in which case the node key is NOT recomputed, and NOT provided in the output (*) or regenerate the node config file for storage on a floppy that is, exactly what the ancestor method used todo, including renewing the node's key (*) or regenerate the config file and bundle it inside an ISO or USB image (*) or just provide the generic ISO or USB boot images in which case of course the node_id_or_hostname parameter is not used

action is expected among the following string constants according the node type value:

for a 'regular' node: (*) node-preview (*) node-floppy (*) node-iso (*) node-usb (*) generic-iso (*) generic-usb

Apart for the preview mode, this method generates a new node key for the specified node, effectively invalidating any old boot medium. Note that 'reservable' nodes do not support 'node-floppy', 'generic-iso' nor 'generic-usb'.

In addition, two return mechanisms are supported. (*) The default behaviour is that the file's content is returned as a base64-encoded string. This is how the ancestor method used to work. To use this method, pass an empty string as the file parameter.

(*) Or, for efficiency -- this makes sense only when the API is used by the web pages that run on the same host -- the caller may provide a filename, in which case the resulting file is stored in that location instead. The filename argument can use the following markers, that are expanded within the method - %d : default root dir (some builtin dedicated area under /var/tmp/) Using this is recommended, and enforced for non-admin users - %n : the node's name when this makes sense, or a mktemp-like name when generic media is requested - %s : a file suffix appropriate in the context (.txt, .iso or the like) - %v : the bootcd version string (e.g. 4.0) - %p : the PLC name - %f : the nodefamily - %a : arch With the file-based return mechanism, the method returns the full pathname of the result file; ** WARNING ** It is the caller's responsability to remove this file after use.

Options: an optional array of keywords. options are not supported for generic images Currently supported are - 'partition' - for USB actions only - 'cramfs' - 'serial' or 'serial:<console_spec>' console_spec (or 'default') is passed as-is to bootcd/build.sh it is expected to be a colon separated string denoting tty - baudrate - parity - bits e.g. ttyS0:115200:n:8 - 'variant:<variantname>' passed to build.sh as -V <variant> variants are used to run a different kernel on the bootCD see kvariant.sh for how to create a variant - 'no-hangcheck' - disable hangcheck - 'systemd-debug' - turn on systemd debug in bootcd

Tags: the following tags are taken into account when attached to the node: 'serial', 'cramfs', 'kvariant', 'kargs', 'no-hangcheck', 'systemd-debug'

Security: - Non-admins can only generate files for nodes at their sites. - Non-admins, when they provide a filename, *must* specify it in the %d area

Housekeeping: Whenever needed, the method stores intermediate files in a private area, typically not located under the web server's accessible area, and are cleaned up by the method.

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `node_id_or_hostname` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

- `action` : string, Action mode, expected value depends of the type of node
- `filename` : string, Empty string for verbatim result, resulting file full path otherwise
- `options` : array, Options
  - string

Returns:

- string, Node boot medium, either inlined, or filename, depending on the filename parameter

GetBootStates

Prototype:

GetBootStates (auth)

Description:

Returns an array of all valid node boot states.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

Returns:

- array of string, Boot state

GetConfFiles

Prototype:

GetConfFiles (auth, conf_file_filter, return_fields)

Description:

Returns an array of structs containing details about configuration files. If conf_file_filter is specified and is an array of configuration file identifiers, or a struct of configuration file attributes, only configuration files matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin, node

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *conf_file_filter* : array of int or struct
  - array of int, Configuration file identifier
  - struct, Attribute filter
    - *file_owner* : string or array of string
      - string, chown(1) owner
      - array of string, chown(1) owner

    - *postinstall_cmd* : string or array of string
      - string, Shell command to execute after installing
      - array of string, Shell command to execute after installing

    - *error_cmd* : string or array of string
      - string, Shell command to execute if any error occurs
      - array of string, Shell command to execute if any error occurs

    - *preinstall_cmd* : string or array of string
      - string, Shell command to execute prior to installing
      - array of string, Shell command to execute prior to installing

    - *node_ids* : int or array of int
      - int, List of nodes linked to this file
      - array of int, List of nodes linked to this file

    - *dest* : string or array of string
      - string, Absolute path where file should be installed
      - array of string, Absolute path where file should be installed

    - *ignore_cmd_errors* : boolean or array of boolean
      - boolean, Install file anyway even if an error occurs
      - array of boolean, Install file anyway even if an error occurs

    - *enabled* : boolean or array of boolean
      - boolean, Configuration file is active
      - array of boolean, Configuration file is active

    - *conf_file_id* : int or array of int
      - int, Configuration file identifier
      - array of int, Configuration file identifier

    - *file_permissions* : string or array of string

    - string, chmod(1) permissions
    - array of string, chmod(1) permissions

  - *source* : string or array of string
    - string, Relative path on the boot server where file can be downloaded
    - array of string, Relative path on the boot server where file can be downloaded

  - *nodegroup_ids* : int or array of int
    - int, List of node groups linked to this file
    - array of int, List of node groups linked to this file

  - *always_update* : boolean or array of boolean
    - boolean, Always attempt to install file even if unchanged
    - array of boolean, Always attempt to install file even if unchanged

  - *file_group* : string or array of string
    - string, chgrp(1) owner
    - array of string, chgrp(1) owner

- *return_fields* : array, List of fields to return
  - string

Returns:

- array of struct
  - *postinstall_cmd* : string, Shell command to execute after installing
  - *preinstall_cmd* : string, Shell command to execute prior to installing
  - *node_ids* : int, List of nodes linked to this file
  - *dest* : string, Absolute path where file should be installed
  - *ignore_cmd_errors* : boolean, Install file anyway even if an error occurs
  - *file_permissions* : string, chmod(1) permissions
  - *always_update* : boolean, Always attempt to install file even if unchanged
  - *file_group* : string, chgrp(1) owner
  - *file_owner* : string, chown(1) owner
  - *error_cmd* : string, Shell command to execute if any error occurs
  - *nodegroup_ids* : int, List of node groups linked to this file
  - *enabled* : boolean, Configuration file is active
  - *conf_file_id* : int, Configuration file identifier
  - *source* : string, Relative path on the boot server where file can be downloaded

GetEventObjects

Prototype:

GetEventObjects (auth, event_filter, return_fields)

Description:

Returns an array of structs containing details about events and faults. If event_filter is specified and is an array of event identifiers, or a struct of event attributes, only events matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `event_filter` : struct, Attribute filter
  - `fault_code` : int or array of int
    - int, Event fault code
    - array of int, Event fault code

  - `event_id` : int or array of int
    - int, Event identifier
    - array of int, Event identifier

  - `object_type` : string or array of string
    - string, What type of object is this event affecting
    - array of string, What type of object is this event affecting

  - `object_id` : int or array of int
    - int, ID of objects affected by this event
    - array of int, ID of objects affected by this event

  - `node_id` : int or array of int
    - int, Identifier of node responsible for event, if any
    - array of int, Identifier of node responsible for event, if any

  - `call` : string or array of string
    - string, Call responsible for this event, including paramters
    - array of string, Call responsible for this event, including paramters

  - `time` : int or array of int
    - int, Date and time that the event took place, in seconds since UNIX epoch
    - array of int, Date and time that the event took place, in seconds since UNIX epoch

- `person_id` : int or array of int
  - int, Identifier of person responsible for event, if any
  - array of int, Identifier of person responsible for event, if any

- `message` : string or array of string
  - string, High level description of this event
  - array of string, High level description of this event

- `runtime` : double or array of double
  - double, Runtime of event
  - array of double, Runtime of event

- `call_name` : string or array of string
  - string, Call responsible for this event
  - array of string, Call responsible for this event

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `fault_code` : int, Event fault code
  - `object_type` : string, What type of object is this event affecting
  - `node_id` : int, Identifier of node responsible for event, if any
  - `message` : string, High level description of this event
  - `event_id` : int, Event identifier
  - `object_id` : int, ID of objects affected by this event
  - `call_name` : string, Call responsible for this event
  - `call` : string, Call responsible for this event, including paramters
  - `time` : int, Date and time that the event took place, in seconds since UNIX epoch
  - `person_id` : int, Identifier of person responsible for event, if any
  - `runtime` : double, Runtime of event

GetEvents

Prototype:

GetEvents (auth, event_filter, return_fields)

Description:

Returns an array of structs containing details about events and faults. If event_filter is specified and is an array of event identifiers, or a struct of event attributes, only events matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `event_filter` : array of int or struct
  - array of int, Event identifier
  - struct, Attribute filter
    - `auth_type` : int or array of int
      - int, Type of auth used. i.e. AuthMethod
      - array of int, Type of auth used. i.e. AuthMethod

    - `fault_code` : int or array of int
      - int, Event fault code
      - array of int, Event fault code

    - `event_id` : int or array of int
      - int, Event identifier
      - array of int, Event identifier

    - `object_ids` : array or array of array
      - array, IDs of objects affected by this event
        - int

      - array of array, IDs of objects affected by this event
        - int

    - `node_id` : int or array of int
      - int, Identifier of node responsible for event, if any
      - array of int, Identifier of node responsible for event, if any

    - `call` : string or array of string
      - string, Call responsible for this event, including paramters
      - array of string, Call responsible for this event, including paramters

    - `time` : int or array of int
      - int, Date and time that the event took place, in seconds since UNIX epoch
      - array of int, Date and time that the event took place, in seconds since UNIX epoch

    - `person_id` : int or array of int

    - int, Identifier of person responsible for event, if any
    - array of int, Identifier of person responsible for event, if any

  - `message` : string or array of string
    - string, High level description of this event
    - array of string, High level description of this event

  - `runtime` : double or array of double
    - double, Runtime of event
    - array of double, Runtime of event

  - `call_name` : string or array of string
    - string, Call responsible for this event
    - array of string, Call responsible for this event

  - `object_types` : array or array of array
    - array, What type of object were affected by this event
      - string

    - array of array, What type of object were affected by this event
      - string

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `auth_type` : int, Type of auth used. i.e. AuthMethod
  - `fault_code` : int, Event fault code
  - `object_ids` : array, IDs of objects affected by this event
    - int

  - `node_id` : int, Identifier of node responsible for event, if any
  - `message` : string, High level description of this event
  - `event_id` : int, Event identifier
  - `call_name` : string, Call responsible for this event
  - `call` : string, Call responsible for this event, including paramters
  - `time` : int, Date and time that the event took place, in seconds since UNIX epoch
  - `person_id` : int, Identifier of person responsible for event, if any
  - `runtime` : double, Runtime of event

- *object_types* : array, What type of object were affected by this event
  - string

GetIlinks

Prototype:

GetIlinks (auth, ilink_filter, return_fields)

Description:

Returns an array of structs containing details about nodes and related tags.

If ilink_filter is specified and is an array of ilink identifiers, only ilinks matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin, pi, user, node

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *ilink_filter* : array of int or int or struct
  - array of int, ilink identifier
  - int, ilink id
  - struct, Attribute filter
    - *tag_type_id* : int or array of int
      - int, Node tag type identifier
      - array of int, Node tag type identifier

    - *ilink_id* : int or array of int
      - int, ilink identifier
      - array of int, ilink identifier

    - *src_interface_id* : int or array of int
      - int, source interface identifier
      - array of int, source interface identifier

    - *value* : string or array of string
      - string, optional ilink value
      - array of string, optional ilink value

    - *dst_interface_id* : int or array of int
      - int, destination interface identifier
      - array of int, destination interface identifier

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `tag_type_id` : int, Node tag type identifier
  - `dst_interface_id` : int, destination interface identifier
  - `value` : string, optional ilink value
  - `src_interface_id` : int, source interface identifier
  - `ilink_id` : int, ilink identifier

GetInitScripts

Prototype:

GetInitScripts (auth, initscript_filter, return_fields)

Description:

Returns an array of structs containing details about initscripts. If initscript_filter is specified and is an array of initscript identifiers, or a struct of initscript attributes, only initscripts matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `initscript_filter` : array of int or string or struct
  - array of int or string
    - int, Initscript identifier
    - string, Initscript name

  - struct, Attribute filter
    - `initscript_id` : int or array of int
      - int, Initscript identifier
      - array of int, Initscript identifier

    - `enabled` : boolean or array of boolean
      - boolean, Initscript is active
      - array of boolean, Initscript is active

    - `name` : string or array of string

    - string, Initscript name
    - array of string, Initscript name

  - `script` : string or array of string
    - string, Initscript
    - array of string, Initscript

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `initscript_id` : int, Initscript identifier
  - `enabled` : boolean, Initscript is active
  - `name` : string, Initscript name
  - `script` : string, Initscript

GetInterfaceAlias

Prototype:

GetInterfaceAlias (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag alias

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceBackdoor

Prototype:

GetInterfaceBackdoor (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag backdoor

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceChannel

Prototype:

GetInterfaceChannel (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag channel

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,

- nil,

GetInterfaceDriver

Prototype:

GetInterfaceDriver (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag driver

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceEssid

Prototype:

GetInterfaceEssid (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag essid

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceFreq

Prototype:

GetInterfaceFreq (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag freq

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceIfname

Prototype:

GetInterfaceIfname (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag ifname

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string

- int, Node interface identifier
- string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceIwconfig

Prototype:

GetInterfaceIwconfig (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag iwconfig

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceIwpriv

Prototype:

GetInterfaceIwpriv (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag iwpriv

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure

- `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceKey

Prototype:

GetInterfaceKey (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag key

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceKey1

Prototype:

GetInterfaceKey1 (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag key1

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceKey2

Prototype:

GetInterfaceKey2 (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag key2

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceKey3

Prototype:

GetInterfaceKey3 (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag key3

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceKey4

Prototype:

GetInterfaceKey4 (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag key4

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,

- nil,

GetInterfaceMode

Prototype:

GetInterfaceMode (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag mode

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceNw

Prototype:

GetInterfaceNw (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag nw

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceRate

Prototype:

GetInterfaceRate (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag rate

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceSecurityMode

Prototype:

GetInterfaceSecurityMode (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag securitymode

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string

- int, Node interface identifier
- string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceSens

Prototype:

GetInterfaceSens (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag sens

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceSliversIPv6Prefix

Prototype:

GetInterfaceSliversIPv6Prefix (auth, id_or_name)

Description:

Accessor 'get' method designed for Interface objects using tag sliversipv6prefix

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure

- `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

Returns:

- string or nil
  - string,
  - nil,

GetInterfaceTags

Prototype:

GetInterfaceTags (auth, interface_tag_filter, return_fields)

Description:

Returns an array of structs containing details about interfaces and related settings.

If interface_tag_filter is specified and is an array of interface setting identifiers, only interface settings matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `interface_tag_filter` : array of int or int or struct
  - array of int, Interface setting identifier
  - int, Interface setting id
  - struct, Attribute filter
    - `category` : string or array of string
      - string, Node tag category
      - array of string, Node tag category

    - `description` : string or array of string
      - string, Node tag type description
      - array of string, Node tag type description

    - `ip` : string or array of string
      - string, IP address
      - array of string, IP address

- *value* : string or array of string
  - string, Interface setting value
  - array of string, Interface setting value

- *interface_id* : int or array of int
  - int, Node interface identifier
  - array of int, Node interface identifier

- *interface_tag_id* : int or array of int
  - int, Interface setting identifier
  - array of int, Interface setting identifier

- *tagname* : string or array of string
  - string, Node tag type name
  - array of string, Node tag type name

- *tag_type_id* : int or array of int
  - int, Node tag type identifier
  - array of int, Node tag type identifier

- *return_fields* : array, List of fields to return
  - string

Returns:

- array of struct
  - *category* : string, Node tag category
  - *interface_tag_id* : int, Interface setting identifier
  - *description* : string, Node tag type description
  - *tagname* : string, Node tag type name
  - *ip* : string, IP address
  - *tag_type_id* : int, Node tag type identifier
  - *value* : string, Interface setting value
  - *interface_id* : int, Node interface identifier

GetInterfaces

Prototype:

GetInterfaces (auth, interface_filter, return_fields)

Description:

Returns an array of structs containing details about network interfaces. If interfaces_filter is specified and is an array of interface identifiers, or a struct of interface fields and values, only interfaces matching the filter will be returned.

If return_fields is given, only the specified details will be returned.

Allowed Roles:

admin, pi, user, tech, node, anonymous

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `interface_filter` : array of int or string or int or string or struct
  - array of int or string
    - int, Node interface identifier
    - string, IP address

  - int, interface id
  - string, ip address
  - struct, Attribute filter
    - `last_updated` : int or array of int
      - int, Date and time when node entry was created
      - array of int, Date and time when node entry was created

    - `network` : string or array of string
      - string, Subnet address
      - array of string, Subnet address

    - `is_primary` : boolean or array of boolean
      - boolean, Is the primary interface for this node
      - array of boolean, Is the primary interface for this node

    - `dns1` : string or array of string
      - string, IP address of primary DNS server
      - array of string, IP address of primary DNS server

    - `hostname` : string or array of string
      - string, (Optional) Hostname
      - array of string, (Optional) Hostname

    - `mac` : string or array of string
      - string, MAC address
      - array of string, MAC address

    - `interface_tag_ids` : array or array of array
      - array, List of interface settings

    - int

  - array of array, List of interface settings
    - int

- `interface_id` : int or array of int
  - int, Node interface identifier
  - array of int, Node interface identifier

- `broadcast` : string or array of string
  - string, Network broadcast address
  - array of string, Network broadcast address

- `method` : string or array of string
  - string, Addressing method (e.g., 'static' or 'dhcp')
  - array of string, Addressing method (e.g., 'static' or 'dhcp')

- `netmask` : string or array of string
  - string, Subnet mask
  - array of string, Subnet mask

- `node_id` : int or array of int
  - int, Node associated with this interface
  - array of int, Node associated with this interface

- `dns2` : string or array of string
  - string, IP address of secondary DNS server
  - array of string, IP address of secondary DNS server

- `ip` : string or array of string
  - string, IP address
  - array of string, IP address

- `bwlimit` : int or array of int
  - int, Bandwidth limit
  - array of int, Bandwidth limit

- `type` : string or array of string
  - string, Address type (e.g., 'ipv4')
  - array of string, Address type (e.g., 'ipv4')

- `gateway` : string or array of string
  - string, IP address of primary gateway

    - array of string, IP address of primary gateway

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `is_primary` : boolean, Is the primary interface for this node
  - `last_updated` : int, Date and time when node entry was created
  - `network` : string, Subnet address
  - `ip` : string, IP address
  - `dns1` : string, IP address of primary DNS server
  - `hostname` : string, (Optional) Hostname
  - `netmask` : string, Subnet mask
  - `interface_tag_ids` : array, List of interface settings
    - int

  - `interface_id` : int, Node interface identifier
  - `broadcast` : string, Network broadcast address
  - `mac` : string, MAC address
  - `node_id` : int, Node associated with this interface
  - `gateway` : string, IP address of primary gateway
  - `dns2` : string, IP address of secondary DNS server
  - `bwlimit` : int, Bandwidth limit
  - `type` : string, Address type (e.g., 'ipv4')
  - `method` : string, Addressing method (e.g., 'static' or 'dhcp')

GetKeyTypes

Prototype:

GetKeyTypes (auth)

Description:

Returns an array of all valid key types.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

Returns:

- array of string, Key type

GetKeys

Prototype:

GetKeys (auth, key_filter, return_fields)

Description:

Returns an array of structs containing details about keys. If key_filter is specified and is an array of key identifiers, or a struct of key attributes, only keys matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Admin may query all keys. Non-admins may only query their own keys.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `key_filter` : array of int or struct
  - array of int
    - int, Key identifier

  - struct, Attribute filter
    - `peer_key_id` : int or array of int
      - int, Foreign key identifier at peer
      - array of int, Foreign key identifier at peer

    - `key_type` : string or array of string
      - string, Key type
      - array of string, Key type

    - `key` : string or array of string
      - string, Key value
      - array of string, Key value

    - `person_id` : int or array of int
      - int, User to which this key belongs
      - array of int, User to which this key belongs

    - `key_id` : int or array of int
      - int, Key identifier
      - array of int, Key identifier

    - `peer_id` : int or array of int
      - int, Peer to which this key belongs
      - array of int, Peer to which this key belongs

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `peer_id` : int, Peer to which this key belongs
  - `key_type` : string, Key type
  - `key` : string, Key value
  - `person_id` : int, User to which this key belongs
  - `key_id` : int, Key identifier
  - `peer_key_id` : int, Foreign key identifier at peer

GetLeaseGranularity

Prototype:

GetLeaseGranularity (auth)

Description:

Returns the granularity in seconds for the reservation system

Allowed Roles:

admin, pi, user, tech, node, anonymous

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

Returns:

- int, the granularity in seconds for the reservation system

GetLeases

Prototype:

GetLeases (auth, lease_filter, return_fields)

Description:

Returns an array of structs containing details about leases. If lease_filter is specified and is an array of lease identifiers or lease names, or a struct of lease attributes, only leases matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

All leases are exposed to all users.

In addition to the usual filter capabilities, the following are supported: * GetLeases ({ 'alive' : '2010-02-20 20:00' , <regular_filter_fields...> }) returns the leases that are active at that point in time * GetLeases ({ 'alive' : ('2010-02-20 20:00' , '2010-02-20 21:00') , ... }) ditto for a time range

This is implemented in the LeaseFilter class; negation actually is supported through the usual '~alive' form, although maybe not really useful.

Allowed Roles:

admin, pi, user, node

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *lease_filter* : int or array of int or struct
  - int, Lease identifier
  - array of int, Lease identifier
  - struct, Lease filter -- adds the 'alive' and 'clip' capabilities for filtering on leases
    - *lease_id* : int or array of int
      - int, Lease identifier
      - array of int, Lease identifier

    - *name* : string or array of string
      - string, Slice name
      - array of string, Slice name

    - *slice_id* : int or array of int
      - int, Slice identifier
      - array of int, Slice identifier

    - *hostname* : string or array of string
      - string, Fully qualified hostname
      - array of string, Fully qualified hostname

    - *site_id* : int or array of int
      - int, Identifier of the site to which this slice belongs
      - array of int, Identifier of the site to which this slice belongs

    - *alive* : int or string or array
      - int, int_timestamp: leases alive at that time
      - string, str_timestamp: leases alive at that time
      - array, timeslot: the leases alive during this timeslot

    - *node_type* : string or array of string
      - string, Node type

    - array of string, Node type

- `node_id` : int or array of int
  - int, Node identifier
  - array of int, Node identifier

- `clip` : int or string or array
  - int, int_timestamp: leases alive after that time
  - string, str_timestamp: leases alive after at that time
  - array, timeslot: the leases alive during this timeslot

- `duration` : int or array of int
  - int, duration in seconds
  - array of int, duration in seconds

- `expired` : boolean or array of boolean
  - boolean, time slot is over
  - array of boolean, time slot is over

- `t_from` : int or string or array of int or string
  - int or string
    - int, timeslot start (unix timestamp)
    - string, timeslot start (formatted as %Y-%m-%d %H:%M:%S)

  - array of int or string
    - int, timeslot start (unix timestamp)
    - string, timeslot start (formatted as %Y-%m-%d %H:%M:%S)

- `t_until` : int or string or array of int or string
  - int or string
    - int, timeslot end (unix timestamp)
    - string, timeslot end (formatted as %Y-%m-%d %H:%M:%S)

  - array of int or string
    - int, timeslot end (unix timestamp)
    - string, timeslot end (formatted as %Y-%m-%d %H:%M:%S)

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - *lease_id* : int, Lease identifier
  - *site_id* : int, Identifier of the site to which this slice belongs
  - *node_type* : string, Node type
  - *node_id* : int, Node identifier
  - *duration* : int, duration in seconds
  - *expired* : boolean, time slot is over
  - *t_from* : int or string
    - int, timeslot start (unix timestamp)
    - string, timeslot start (formatted as %Y-%m-%d %H:%M:%S)
  - *name* : string, Slice name
  - *slice_id* : int, Slice identifier
  - *hostname* : string, Fully qualified hostname
  - *t_until* : int or string
    - int, timeslot end (unix timestamp)
    - string, timeslot end (formatted as %Y-%m-%d %H:%M:%S)

GetMessages

Prototype:

GetMessages (auth, message_filter, return_fields)

Description:

Returns an array of structs containing details about message templates. If message template_filter is specified and is an array of message template identifiers, or a struct of message template attributes, only message templates matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin, node

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *message_filter* : array of string or struct
  - array of string, Message identifier
  - struct, Attribute filter
    - *enabled* : boolean or array of boolean
      - boolean, Message is enabled
      - array of boolean, Message is enabled
    - *message_id* : string or array of string

    - string, Message identifier
    - array of string, Message identifier

  - `template` : string or array of string
    - string, Message template
    - array of string, Message template

  - `subject` : string or array of string
    - string, Message summary
    - array of string, Message summary

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `enabled` : boolean, Message is enabled
  - `message_id` : string, Message identifier
  - `template` : string, Message template
  - `subject` : string, Message summary

GetNetworkMethods

Prototype:

GetNetworkMethods (auth)

Description:

Returns a list of all valid network methods.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

Returns:

- array of string, Network method

GetNetworkTypes

Prototype:

GetNetworkTypes (auth)

Description:

Returns a list of all valid network types.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

Returns:

- array of string, Network type

GetNodeArch

Prototype:

GetNodeArch (auth, id_or_name)

Description:

Accessor 'get' method designed for Node objects using tag arch

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- string or nil
  - string,
  - nil,

GetNodeCramfs

Prototype:

GetNodeCramfs (auth, id_or_name)

Description:

Accessor 'get' method designed for Node objects using tag cramfs

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- string or nil
  - string,
  - nil,

GetNodeDeployment

Prototype:

GetNodeDeployment (auth, id_or_name)

Description:

Accessor 'get' method designed for Node objects using tag deployment

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- string or nil
  - string,

- nil,

GetNodeExtensions

Prototype:

GetNodeExtensions (auth, id_or_name)

Description:

Accessor 'get' method designed for Node objects using tag extensions

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- string or nil
  - string,
  - nil,

GetNodeFcdistro

Prototype:

GetNodeFcdistro (auth, id_or_name)

Description:

Accessor 'get' method designed for Node objects using tag fcdistro

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- string or nil
  - string,
  - nil,

GetNodeFlavour

Prototype:

GetNodeFlavour (auth, node_id_or_name)

Description:

Returns detailed information on a given node's flavour, i.e. its base installation.

This depends on the global PLC settings in the PLC_FLAVOUR area, optionnally overridden by any of the following tags if set on that node:

'arch', 'pldistro', 'fcdistro', 'deployment', 'extensions', 'virt',

Allowed Roles:

admin, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `node_id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- struct
  - `extensions` : array of string, extensions to add to the base install
  - `fcdistro` : string, the fcdistro this node should be based upon
  - `nodefamily` : string, the nodefamily this node should be based upon
  - `plain` : boolean, use plain bootstrapfs image if set (for tests)

GetNodeGroups

Prototype:

GetNodeGroups (auth, nodegroup_filter, return_fields)

Description:

Returns an array of structs containing details about node groups. If nodegroup_filter is specified and is an array of node group identifiers or names, or a struct of node group attributes, only node groups matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin, pi, user, tech, node, anonymous

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `nodegroup_filter` : array of int or string or struct
  - array of int or string
    - int, Node group identifier
    - string, Node group name

  - struct, Attribute filter
    - `node_ids` : array or array of array
      - array, List of node_ids that belong to this nodegroup
        - int

      - array of array, List of node_ids that belong to this nodegroup
        - int

    - `value` : string or array of string
      - string, value that the nodegroup definition is based upon
      - array of string, value that the nodegroup definition is based upon

    - `groupname` : string or array of string
      - string, Node group name
      - array of string, Node group name

    - `nodegroup_id` : int or array of int
      - int, Node group identifier
      - array of int, Node group identifier

    - `tagname` : string or array of string
      - string, Tag name that the nodegroup definition is based upon
      - array of string, Tag name that the nodegroup definition is based upon

    - `tag_type_id` : int or array of int
      - int, Node tag type id
      - array of int, Node tag type id

    - `conf_file_ids` : array or array of array
      - array, List of configuration files specific to this node group
        - int

      - array of array, List of configuration files specific to this node group

    - int

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `node_ids` : array, List of node_ids that belong to this nodegroup
    - int

  - `value` : string, value that the nodegroup definition is based upon
  - `groupname` : string, Node group name
  - `nodegroup_id` : int, Node group identifier
  - `tagname` : string, Tag name that the nodegroup definition is based upon
  - `tag_type_id` : int, Node tag type id
  - `conf_file_ids` : array, List of configuration files specific to this node group
    - int

GetNodeHrn

Prototype:

GetNodeHrn (auth, id_or_name)

Description:

Accessor 'get' method designed for Node objects using tag hrn

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- string or nil
  - string,

- nil,

GetNodeKargs

Prototype:

GetNodeKargs (auth, id_or_name)

Description:

Accessor 'get' method designed for Node objects using tag kargs

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- string or nil
  - string,
  - nil,

GetNodeKvariant

Prototype:

GetNodeKvariant (auth, id_or_name)

Description:

Accessor 'get' method designed for Node objects using tag kvariant

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- string or nil
  - string,
  - nil,

GetNodeNoHangcheck

Prototype:

GetNodeNoHangcheck (auth, id_or_name)

Description:

Accessor 'get' method designed for Node objects using tag no-hangcheck

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- string or nil
  - string,
  - nil,

GetNodePlainBootstrapfs

Prototype:

GetNodePlainBootstrapfs (auth, id_or_name)

Description:

Accessor 'get' method designed for Node objects using tag plain-bootstrapfs

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string

- int, Node identifier
- string, Fully qualified hostname

Returns:

- string or nil
  - string,
  - nil,

GetNodePldistro

Prototype:

GetNodePldistro (auth, id_or_name)

Description:

Accessor 'get' method designed for Node objects using tag pldistro

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- string or nil
  - string,
  - nil,

GetNodeSerial

Prototype:

GetNodeSerial (auth, id_or_name)

Description:

Accessor 'get' method designed for Node objects using tag serial

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure

- `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- string or nil
  - string,
  - nil,

GetNodeTags

Prototype:

GetNodeTags (auth, node_tag_filter, return_fields)

Description:

Returns an array of structs containing details about nodes and related tags.

If node_tag_filter is specified and is an array of node tag identifiers, only node tags matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `node_tag_filter` : array of int or int or struct
  - array of int, Node tag identifier
  - int, Node tag id
  - struct, Attribute filter
    - `category` : string or array of string
      - string, Node tag category
      - array of string, Node tag category

    - `description` : string or array of string
      - string, Node tag type description
      - array of string, Node tag type description

    - `hostname` : string or array of string
      - string, Fully qualified hostname
      - array of string, Fully qualified hostname

- *value* : string or array of string
  - string, Node tag value
  - array of string, Node tag value

- *node_id* : int or array of int
  - int, Node identifier
  - array of int, Node identifier

- *node_tag_id* : int or array of int
  - int, Node tag identifier
  - array of int, Node tag identifier

- *tagname* : string or array of string
  - string, Node tag type name
  - array of string, Node tag type name

- *tag_type_id* : int or array of int
  - int, Node tag type identifier
  - array of int, Node tag type identifier

- *return_fields* : array, List of fields to return
  - string

Returns:

- array of struct
  - *category* : string, Node tag category
  - *node_id* : int, Node identifier
  - *node_tag_id* : int, Node tag identifier
  - *description* : string, Node tag type description
  - *tagname* : string, Node tag type name
  - *tag_type_id* : int, Node tag type identifier
  - *hostname* : string, Fully qualified hostname
  - *value* : string, Node tag value

GetNodeTypes

Prototype:

GetNodeTypes (auth)

Description:

Returns an array of all valid node node types.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

Returns:

- array of string, Node type

GetNodeVirt

Prototype:

GetNodeVirt (auth, id_or_name)

Description:

Accessor 'get' method designed for Node objects using tag virt

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- string or nil
  - string,
  - nil,

GetNodes

Prototype:

GetNodes (auth, node_filter, return_fields)

Description:

Returns an array of structs containing details about nodes. If node_filter is specified and is an array of node identifiers or hostnames, or a struct of node attributes, only nodes matching the filter will be returned.

If return_fields is specified, only the specified details will be returned. NOTE that if return_fields is unspecified, the complete set of native fields are returned, which DOES NOT include tags at this time.

Some fields may only be viewed by admins.

Allowed Roles:

admin, pi, user, tech, node, anonymous

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `node_filter` : array of int or string or string or int or struct
  - array of int or string
    - int, Node identifier
    - string, Fully qualified hostname

  - string, hostname
  - int, node_id
  - struct, Attribute filter
    - `last_updated` : int or array of int
      - int, Date and time when node entry was created
      - array of int, Date and time when node entry was created

    - `key` : string or array of string
      - string, (Admin only) Node key
      - array of string, (Admin only) Node key

    - `boot_state` : string or array of string
      - string, Boot state
      - array of string, Boot state

    - `site_id` : int or array of int
      - int, Site at which this node is located
      - array of int, Site at which this node is located

    - `pcu_ids` : array or array of array
      - array, List of PCUs that control this node
        - int

      - array of array, List of PCUs that control this node
        - int

    - `node_type` : string or array of string
      - string, Node type
      - array of string, Node type

- `session` : string or array of string
  - string, (Admin only) Node session value
  - array of string, (Admin only) Node session value

- `ssh_rsa_key` : string or array of string
  - string, Last known SSH host key
  - array of string, Last known SSH host key

- `last_pcu_reboot` : int or array of int
  - int, Date and time when PCU reboot was attempted
  - array of int, Date and time when PCU reboot was attempted

- `node_tag_ids` : array or array of array
  - array, List of tags attached to this node
    - int

  - array of array, List of tags attached to this node
    - int

- `verified` : boolean or array of boolean
  - boolean, Whether the node configuration is verified correct
  - array of boolean, Whether the node configuration is verified correct

- `last_contact` : int or array of int
  - int, Date and time when node last contacted plc
  - array of int, Date and time when node last contacted plc

- `peer_node_id` : int or array of int
  - int, Foreign node identifier at peer
  - array of int, Foreign node identifier at peer

- `hostname` : string or array of string
  - string, Fully qualified hostname
  - array of string, Fully qualified hostname

- `run_level` : string or array of string
  - string, Run level
  - array of string, Run level

- `slice_ids` : array or array of array
  - array, List of slices on this node
    - int

  - array of array, List of slices on this node
    - int

- `last_time_spent_offline` : int or array of int
  - int, Length of time the node was last offline after failure and before reboot
  - array of int, Length of time the node was last offline after failure and before reboot

- `version` : string or array of string
  - string, Apparent Boot CD version
  - array of string, Apparent Boot CD version

- `peer_id` : int or array of int
  - int, Peer to which this node belongs
  - array of int, Peer to which this node belongs

- `node_id` : int or array of int
  - int, Node identifier
  - array of int, Node identifier

- `last_boot` : int or array of int
  - int, Date and time when node last booted
  - array of int, Date and time when node last booted

- `interface_ids` : array or array of array
  - array, List of network interfaces that this node has
    - int

  - array of array, List of network interfaces that this node has
    - int

- `conf_file_ids` : array or array of array
  - array, List of configuration files specific to this node
    - int

  - array of array, List of configuration files specific to this node
    - int

- `last_pcu_confirmation` : int or array of int
  - int, Date and time when PCU reboot was confirmed
  - array of int, Date and time when PCU reboot was confirmed

- `nodegroup_ids` : array or array of array
  - array, List of node groups that this node is in
    - int
  - array of array, List of node groups that this node is in
    - int
- `slice_ids_whitelist` : array or array of array
  - array, List of slices allowed on this node
    - int
  - array of array, List of slices allowed on this node
    - int
- `last_time_spent_online` : int or array of int
  - int, Length of time the node was last online before shutdown/failure
  - array of int, Length of time the node was last online before shutdown/failure
- `boot_nonce` : string or array of string
  - string, (Admin only) Random value generated by the node at last boot
  - array of string, (Admin only) Random value generated by the node at last boot
- `last_download` : int or array of int
  - int, Date and time when node boot image was created
  - array of int, Date and time when node boot image was created
- `date_created` : int or array of int
  - int, Date and time when node entry was created
  - array of int, Date and time when node entry was created
- `model` : string or array of string
  - string, Make and model of the actual machine
  - array of string, Make and model of the actual machine
- `ports` : array or array of array
  - array, List of PCU ports that this node is connected to
    - int
  - array of array, List of PCU ports that this node is connected to
    - int

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `last_updated` : int, Date and time when node entry was created
  - `key` : string, (Admin only) Node key
  - `session` : string, (Admin only) Node session value
  - `boot_state` : string, Boot state
  - `site_id` : int, Site at which this node is located
  - `pcu_ids` : array, List of PCUs that control this node
    - int
  - `node_type` : string, Node type
  - `node_id` : int, Node identifier
  - `last_boot` : int, Date and time when node last booted
  - `interface_ids` : array, List of network interfaces that this node has
    - int
  - `slice_ids_whitelist` : array, List of slices allowed on this node
    - int
  - `run_level` : string, Run level
  - `ssh_rsa_key` : string, Last known SSH host key
  - `last_pcu_reboot` : int, Date and time when PCU reboot was attempted
  - `node_tag_ids` : array, List of tags attached to this node
    - int
  - `nodegroup_ids` : array, List of node groups that this node is in
    - int
  - `verified` : boolean, Whether the node configuration is verified correct
  - `last_contact` : int, Date and time when node last contacted plc
  - `peer_node_id` : int, Foreign node identifier at peer
  - `hostname` : string, Fully qualified hostname
  - `last_time_spent_offline` : int, Length of time the node was last offline after failure and before reboot
  - `conf_file_ids` : array, List of configuration files specific to this node
    - int

- `last_time_spent_online` : int, Length of time the node was last online before shutdown/failure
- `slice_ids` : array, List of slices on this node
  - int

- `boot_nonce` : string, (Admin only) Random value generated by the node at last boot
- `version` : string, Apparent Boot CD version
- `last_pcu_confirmation` : int, Date and time when PCU reboot was confirmed
- `last_download` : int, Date and time when node boot image was created
- `date_created` : int, Date and time when node entry was created
- `model` : string, Make and model of the actual machine
- `peer_id` : int, Peer to which this node belongs
- `ports` : array, List of PCU ports that this node is connected to
  - int

GetPCUProtocolTypes

Prototype:

GetPCUProtocolTypes (auth, protocol_type_filter, return_fields)

Description:

Returns an array of PCU Types.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `protocol_type_filter` : array of int or struct
  - array of int, PCU type identifier
  - struct, Attribute filter
    - `pcu_protocol_type_id` : int or array of int
      - int, PCU protocol type identifier
      - array of int, PCU protocol type identifier

    - `supported` : boolean or array of boolean
      - boolean, Is the port/protocol supported by PLC
      - array of boolean, Is the port/protocol supported by PLC

    - `protocol` : string or array of string
      - string, Protocol

    - array of string, Protocol

  - *port* : int or array of int
    - int, PCU port
    - array of int, PCU port

  - *pcu_type_id* : int or array of int
    - int, PCU type identifier
    - array of int, PCU type identifier

- *return_fields* : array, List of fields to return
  - string

Returns:

- array of struct
  - *pcu_protocol_type_id* : int, PCU protocol type identifier
  - *supported* : boolean, Is the port/protocol supported by PLC
  - *protocol* : string, Protocol
  - *port* : int, PCU port
  - *pcu_type_id* : int, PCU type identifier

GetPCUTypes

Prototype:

GetPCUTypes (auth, pcu_type_filter, return_fields)

Description:

Returns an array of PCU Types.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *pcu_type_filter* : array of int or string or string or int or struct
  - array of int or string
    - int, PCU Type Identifier
    - string, PCU model

  - string, model

- int, node_id
- struct, Attribute filter
  - `model` : string or array of string
    - string, PCU model
    - array of string, PCU model
  - `pcu_protocol_types` : array or array of array
    - array, PCU Protocol Type List
      - struct
    - array of array, PCU Protocol Type List
      - struct
  - `pcu_protocol_type_ids` : array or array of array
    - array, PCU Protocol Type Identifiers
      - int
    - array of array, PCU Protocol Type Identifiers
      - int
  - `name` : string or array of string
    - string, PCU full name
    - array of string, PCU full name
  - `pcu_type_id` : int or array of int
    - int, PCU Type Identifier
    - array of int, PCU Type Identifier
- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `model` : string, PCU model
  - `pcu_protocol_types` : array, PCU Protocol Type List
    - struct
  - `pcu_protocol_type_ids` : array, PCU Protocol Type Identifiers
    - int

- `name` : string, PCU full name
- `pcu_type_id` : int, PCU Type Identifier

GetPCUs

Prototype:

GetPCUs (auth, pcu_filter, return_fields)

Description:

Returns an array of structs containing details about power control units (PCUs). If pcu_filter is specified and is an array of PCU identifiers, or a struct of PCU attributes, only PCUs matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Admin may query all PCUs. Non-admins may only query the PCUs at their sites.

Allowed Roles:

admin, pi, tech, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `pcu_filter` : array of int or struct
  - array of int, PCU identifier
  - struct, Attribute filter
    - `username` : string or array of string
      - string, PCU username
      - array of string, PCU username

    - `protocol` : string or array of string
      - string, PCU protocol, e.g. ssh, https, telnet
      - array of string, PCU protocol, e.g. ssh, https, telnet

    - `node_ids` : array or array of array
      - array, List of nodes that this PCU controls
        - int

      - array of array, List of nodes that this PCU controls
        - int

    - `ip` : string or array of string
      - string, PCU IP address
      - array of string, PCU IP address

    - `pcu_id` : int or array of int

    - int, PCU identifier
    - array of int, PCU identifier

  - `hostname` : string or array of string
    - string, PCU hostname
    - array of string, PCU hostname

  - `site_id` : int or array of int
    - int, Identifier of site where PCU is located
    - array of int, Identifier of site where PCU is located

  - `ports` : array or array of array
    - array, List of the port numbers that each node is connected to
      - int

    - array of array, List of the port numbers that each node is connected to
      - int

  - `last_updated` : int or array of int
    - int, Date and time when node entry was created
    - array of int, Date and time when node entry was created

  - `model` : string or array of string
    - string, PCU model string
    - array of string, PCU model string

  - `password` : string or array of string
    - string, PCU username
    - array of string, PCU username

  - `notes` : string or array of string
    - string, Miscellaneous notes
    - array of string, Miscellaneous notes

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `username` : string, PCU username
  - `protocol` : string, PCU protocol, e.g. ssh, https, telnet

- `node_ids` : array, List of nodes that this PCU controls
  - int

- `ip` : string, PCU IP address
- `pcu_id` : int, PCU identifier
- `site_id` : int, Identifier of site where PCU is located
- `last_updated` : int, Date and time when node entry was created
- `password` : string, PCU username
- `notes` : string, Miscellaneous notes
- `hostname` : string, PCU hostname
- `model` : string, PCU model string
- `ports` : array, List of the port numbers that each node is connected to
  - int

GetPeerData

Prototype:

GetPeerData (auth)

Description:

Returns lists of local objects that a peer should cache in its database as foreign objects. Also returns the list of foreign nodes in this database, for which the calling peer is authoritative, to assist in synchronization of slivers.

See the implementation of RefreshPeer for how this data is used.

Allowed Roles:

admin, peer

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

Returns:

- struct
  - `Slices` : array, List of local slices
    - struct

  - `Keys` : array, List of local keys
    - struct

  - `Sites` : array, List of local sites
    - struct

  - `Persons` : array, List of local users

- struct

- `Nodes` : array, List of local nodes
  - struct

- `db_time` : double, (Debug) Database fetch time

GetPeerName

Prototype:

GetPeerName (auth)

Description:

Returns this peer's name, as defined in the config as PLC_NAME

Allowed Roles:

admin, peer, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

Returns:

- string, Peer name

GetPeers

Prototype:

GetPeers (auth, peer_filter, return_fields)

Description:

Returns an array of structs containing details about peers. If person_filter is specified and is an array of peer identifiers or peer names, or a struct of peer attributes, only peers matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin, node, pi, user

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `peer_filter` : array of int or string or struct
  - array of int or string
    - int, Peer identifier
    - string, Peer name

- struct, Attribute filter
  - `node_ids` : array or array of array
    - array, List of nodes for which this peer is authoritative
      - int
    - array of array, List of nodes for which this peer is authoritative
      - int
  - `key_ids` : array or array of array
    - array, List of keys for which this peer is authoritative
      - int
    - array of array, List of keys for which this peer is authoritative
      - int
  - `person_ids` : array or array of array
    - array, List of users for which this peer is authoritative
      - int
    - array of array, List of users for which this peer is authoritative
      - int
  - `peername` : string or array of string
    - string, Peer name
    - array of string, Peer name
  - `peer_url` : string or array of string
    - string, Peer API URL
    - array of string, Peer API URL
  - `slice_ids` : array or array of array
    - array, List of slices for which this peer is authoritative
      - int
    - array of array, List of slices for which this peer is authoritative
      - int
  - `key` : string or array of string
    - string, Peer GPG public key

    - array of string, Peer GPG public key

  - `hrn_root` : string or array of string
    - string, Root of this peer in a hierarchical naming space
    - array of string, Root of this peer in a hierarchical naming space

  - `cacert` : string or array of string
    - string, Peer SSL public certificate
    - array of string, Peer SSL public certificate

  - `site_ids` : array or array of array
    - array, List of sites for which this peer is authoritative
      - int

    - array of array, List of sites for which this peer is authoritative
      - int

  - `peer_id` : int or array of int
    - int, Peer identifier
    - array of int, Peer identifier

  - `shortname` : string or array of string
    - string, Peer short name
    - array of string, Peer short name

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `node_ids` : array, List of nodes for which this peer is authoritative
    - int

  - `key_ids` : array, List of keys for which this peer is authoritative
    - int

  - `peer_url` : string, Peer API URL
  - `key` : string, Peer GPG public key
  - `hrn_root` : string, Root of this peer in a hierarchical naming space
  - `cacert` : string, Peer SSL public certificate
  - `site_ids` : array, List of sites for which this peer is authoritative

  - int

- `person_ids` : array, List of users for which this peer is authoritative
  - int

- `peername` : string, Peer name
- `slice_ids` : array, List of slices for which this peer is authoritative
  - int

- `shortname` : string, Peer short name
- `peer_id` : int, Peer identifier

GetPersonAdvanced

Prototype:

GetPersonAdvanced (auth, id_or_name)

Description:

Accessor 'get' method designed for Person objects using tag advanced

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, User identifier
  - string, Primary e-mail address

Returns:

- string or nil
  - string,
  - nil,

GetPersonColumnconf

Prototype:

GetPersonColumnconf (auth, id_or_name)

Description:

Accessor 'get' method designed for Person objects using tag columnconf

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, User identifier
  - string, Primary e-mail address

Returns:

- string or nil
  - string,
  - nil,

GetPersonHrn

Prototype:

GetPersonHrn (auth, id_or_name)

Description:

Accessor 'get' method designed for Person objects using tag hrn

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, User identifier
  - string, Primary e-mail address

Returns:

- string or nil
  - string,
  - nil,

GetPersonSfaCreated

Prototype:

GetPersonSfaCreated (auth, id_or_name)

Description:

Accessor 'get' method designed for Person objects using tag sfa_created

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, User identifier
  - string, Primary e-mail address

Returns:

- string or nil
  - string,
  - nil,

GetPersonShowconf

Prototype:

GetPersonShowconf (auth, id_or_name)

Description:

Accessor 'get' method designed for Person objects using tag showconf

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, User identifier
  - string, Primary e-mail address

Returns:

- string or nil
  - string,

- nil,

GetPersonTags

Prototype:

GetPersonTags (auth, person_tag_filter, return_fields)

Description:

Returns an array of structs containing details about persons and related settings.

If person_tag_filter is specified and is an array of person setting identifiers, only person settings matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin, pi, user, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `person_tag_filter` : array of int or int or struct
  - array of int, Person setting identifier
  - int, Person setting id
  - struct, Attribute filter
    - `category` : string or array of string
      - string, Node tag category
      - array of string, Node tag category

    - `description` : string or array of string
      - string, Node tag type description
      - array of string, Node tag type description

    - `value` : string or array of string
      - string, Person setting value
      - array of string, Person setting value

    - `tagname` : string or array of string
      - string, Node tag type name
      - array of string, Node tag type name

    - `person_tag_id` : int or array of int
      - int, Person setting identifier
      - array of int, Person setting identifier

    - `person_id` : int or array of int
      - int, User identifier

    - array of int, User identifier

  - *tag_type_id* : int or array of int
    - int, Node tag type identifier
    - array of int, Node tag type identifier

  - *email* : string or array of string
    - string, Primary e-mail address
    - array of string, Primary e-mail address

- *return_fields* : array, List of fields to return
  - string

Returns:

- array of struct
  - *category* : string, Node tag category
  - *description* : string, Node tag type description
  - *tagname* : string, Node tag type name
  - *person_tag_id* : int, Person setting identifier
  - *person_id* : int, User identifier
  - *tag_type_id* : int, Node tag type identifier
  - *email* : string, Primary e-mail address
  - *value* : string, Person setting value

GetPersons

Prototype:

GetPersons (auth, person_filter, return_fields)

Description:

Returns an array of structs containing details about users. If person_filter is specified and is an array of user identifiers or usernames, or a struct of user attributes, only users matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Users and techs may only retrieve details about themselves. PIs may retrieve details about themselves and others at their sites. Admins and nodes may retrieve details about all accounts.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- `person_filter` : array of int or string or string or int or struct
  - array of int or string
    - int, User identifier
    - string, Primary e-mail address
  - string, email
  - int, person_id
  - struct, Attribute filter
    - `role_ids` : array or array of array
      - array, List of role identifiers
        - int
      - array of array, List of role identifiers
        - int
    - `last_updated` : int or array of int
      - int, Date and time of last update
      - array of int, Date and time of last update
    - `last_name` : string or array of string
      - string, Surname
      - array of string, Surname
    - `site_ids` : array or array of array
      - array, List of site identifiers
        - int
      - array of array, List of site identifiers
        - int
    - `first_name` : string or array of string
      - string, Given name
      - array of string, Given name
    - `title` : string or array of string
      - string, Title
      - array of string, Title
    - `slice_ids` : array or array of array
      - array, List of slice identifiers
        - int

  - array of array, List of slice identifiers
    - int

- `person_id` : int or array of int
  - int, User identifier
  - array of int, User identifier

- `peer_id` : int or array of int
  - int, Peer to which this user belongs
  - array of int, Peer to which this user belongs

- `email` : string or array of string
  - string, Primary e-mail address
  - array of string, Primary e-mail address

- `bio` : string or array of string
  - string, Biography
  - array of string, Biography

- `key_ids` : array or array of array
  - array, List of key identifiers
    - int

  - array of array, List of key identifiers
    - int

- `phone` : string or array of string
  - string, Telephone number
  - array of string, Telephone number

- `peer_person_id` : int or array of int
  - int, Foreign user identifier at peer
  - array of int, Foreign user identifier at peer

- `person_tag_ids` : array or array of array
  - array, List of tags attached to this person
    - int

  - array of array, List of tags attached to this person
    - int

- `password` : string or array of string
  - string, Account password in crypt() form
  - array of string, Account password in crypt() form

- `roles` : array or array of array
  - array, List of roles
    - string

  - array of array, List of roles
    - string

- `url` : string or array of string
  - string, Home page
  - array of string, Home page

- `verification_key` : string or array of string
  - string, Reset password key
  - array of string, Reset password key

- `enabled` : boolean or array of boolean
  - boolean, Has been enabled
  - array of boolean, Has been enabled

- `date_created` : int or array of int
  - int, Date and time when account was created
  - array of int, Date and time when account was created

- `verification_expires` : int or array of int
  - int, Date and time when verification_key expires
  - array of int, Date and time when verification_key expires

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `bio` : string, Biography
  - `first_name` : string, Given name
  - `role_ids` : array, List of role identifiers
    - int

- `last_updated` : int, Date and time of last update
- `roles` : array, List of roles
  - string

- `title` : string, Title
- `url` : string, Home page
- `key_ids` : array, List of key identifiers
  - int

- `enabled` : boolean, Has been enabled
- `slice_ids` : array, List of slice identifiers
  - int

- `phone` : string, Telephone number
- `peer_person_id` : int, Foreign user identifier at peer
- `last_name` : string, Surname
- `person_id` : int, User identifier
- `date_created` : int, Date and time when account was created
- `site_ids` : array, List of site identifiers
  - int

- `peer_id` : int, Peer to which this user belongs
- `email` : string, Primary e-mail address
- `person_tag_ids` : array, List of tags attached to this person
  - int

GetPlcRelease

Prototype:

GetPlcRelease (auth)

Description:

Returns various information about the current myplc installation.

Allowed Roles:

admin, pi, user, tech, node, anonymous

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

Returns:

- struct
  - *rpms* : string
  - *build* : string
  - *tags* : string

GetRoles

Prototype:

GetRoles (auth)

Description:

Get an array of structs containing details about all roles.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

Returns:

- array of struct
  - *name* : string, Role
  - *role_id* : int, Role identifier

GetSession

Prototype:

GetSession (auth, expires)

Description:

Returns a new session key if a user or node authenticated successfully, faults otherwise.

Default value for 'expires' is 24 hours. Otherwise, the returned session 'expires' in the given number of seconds.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *expires* : int, expires

Returns:

- string, Session key

GetSessions

Prototype:

GetSessions (auth, session_filter)

Description:

Returns an array of structs containing details about users sessions. If session_filter is specified and is an array of user identifiers or session_keys, or a struct of session attributes, only sessions matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *session_filter* : array of int or string or struct
  - array of int or string
    - int, Account identifier, if applicable
    - string, Session key

  - struct, Attribute filter
    - *person_id* : int or array of int
      - int, Account identifier, if applicable
      - array of int, Account identifier, if applicable

    - *node_id* : int or array of int
      - int, Node identifier, if applicable
      - array of int, Node identifier, if applicable

    - *session_id* : string or array of string
      - string, Session key
      - array of string, Session key

    - *expires* : int or array of int
      - int, Date and time when session expires, in seconds since UNIX epoch
      - array of int, Date and time when session expires, in seconds since UNIX epoch

Returns:

- array of struct
  - *person_id* : int, Account identifier, if applicable
  - *node_id* : int, Node identifier, if applicable

- `expires` : int, Date and time when session expires, in seconds since UNIX epoch
- `session_id` : string, Session key

GetSiteHrn

Prototype:

GetSiteHrn (auth, id_or_name)

Description:

Accessor 'get' method designed for Site objects using tag hrn

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Site identifier
  - string, Site slice prefix

Returns:

- string or nil
  - string,
  - nil,

GetSiteSfaCreated

Prototype:

GetSiteSfaCreated (auth, id_or_name)

Description:

Accessor 'get' method designed for Site objects using tag sfa_created

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Site identifier
  - string, Site slice prefix

Returns:

- string or nil
  - string,
  - nil,

GetSiteTags

Prototype:

GetSiteTags (auth, site_tag_filter, return_fields)

Description:

Returns an array of structs containing details about sites and related settings.

If site_tag_filter is specified and is an array of site setting identifiers, only site settings matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin, pi, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `site_tag_filter` : array of int or int or struct
  - array of int, Site setting identifier
  - int, Site setting id
  - struct, Attribute filter
    - `category` : string or array of string
      - string, Node tag category
      - array of string, Node tag category

    - `description` : string or array of string
      - string, Node tag type description
      - array of string, Node tag type description

    - `site_id` : int or array of int
      - int, Site identifier
      - array of int, Site identifier

    - `value` : string or array of string
      - string, Site setting value
      - array of string, Site setting value

    - `login_base` : string or array of string

    - string, Site slice prefix
    - array of string, Site slice prefix

- *tagname* : string or array of string
    - string, Node tag type name
    - array of string, Node tag type name

- *site_tag_id* : int or array of int
    - int, Site setting identifier
    - array of int, Site setting identifier

- *tag_type_id* : int or array of int
    - int, Node tag type identifier
    - array of int, Node tag type identifier

- *return_fields* : array, List of fields to return
    - string

Returns:

- array of struct
    - *category* : string, Node tag category
    - *login_base* : string, Site slice prefix
    - *description* : string, Node tag type description
    - *tagname* : string, Node tag type name
    - *site_tag_id* : int, Site setting identifier
    - *tag_type_id* : int, Node tag type identifier
    - *site_id* : int, Site identifier
    - *value* : string, Site setting value

GetSites

Prototype:

GetSites (auth, site_filter, return_fields)

Description:

Returns an array of structs containing details about sites. If site_filter is specified and is an array of site identifiers or hostnames, or a struct of site attributes, only sites matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Allowed Roles:

admin, pi, user, tech, node, anonymous

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `site_filter` : array of int or string or string or int or struct
  - array of int or string
    - int, Site identifier
    - string, Site slice prefix

  - string, login_base
  - int, site_id
  - struct, Attribute filter
    - `last_updated` : int or array of int
      - int, Date and time when site entry was last updated, in seconds since UNIX epoch
      - array of int, Date and time when site entry was last updated, in seconds since UNIX epoch

    - `node_ids` : array or array of array
      - array, List of site node identifiers
        - int

      - array of array, List of site node identifiers
        - int

    - `site_id` : int or array of int
      - int, Site identifier
      - array of int, Site identifier

    - `pcu_ids` : array or array of array
      - array, List of PCU identifiers
        - int

      - array of array, List of PCU identifiers
        - int

    - `max_slices` : int or array of int
      - int, Maximum number of slices that the site is able to create
      - array of int, Maximum number of slices that the site is able to create

    - `ext_consortium_id` : int or array of int
      - int, external consortium id
      - array of int, external consortium id

- `peer_site_id` : int or array of int
  - int, Foreign site identifier at peer
  - array of int, Foreign site identifier at peer

- `abbreviated_name` : string or array of string
  - string, Abbreviated site name
  - array of string, Abbreviated site name

- `person_ids` : array or array of array
  - array, List of account identifiers
    - int

  - array of array, List of account identifiers
    - int

- `slice_ids` : array or array of array
  - array, List of slice identifiers
    - int

  - array of array, List of slice identifiers
    - int

- `latitude` : double or array of double
  - double, Decimal latitude of the site
  - array of double, Decimal latitude of the site

- `peer_id` : int or array of int
  - int, Peer to which this site belongs
  - array of int, Peer to which this site belongs

- `max_slivers` : int or array of int
  - int, Maximum number of slivers that the site is able to create
  - array of int, Maximum number of slivers that the site is able to create

- `is_public` : boolean or array of boolean
  - boolean, Publicly viewable site
  - array of boolean, Publicly viewable site

- `address_ids` : array or array of array
  - array, List of address identifiers
    - int

    - array of array, List of address identifiers
      - int

- `name` : string or array of string
  - string, Full site name
  - array of string, Full site name

- `url` : string or array of string
  - string, URL of a page that describes the site
  - array of string, URL of a page that describes the site

- `site_tag_ids` : array or array of array
  - array, List of tags attached to this site
    - int

  - array of array, List of tags attached to this site
    - int

- `enabled` : boolean or array of boolean
  - boolean, Has been enabled
  - array of boolean, Has been enabled

- `longitude` : double or array of double
  - double, Decimal longitude of the site
  - array of double, Decimal longitude of the site

- `login_base` : string or array of string
  - string, Site slice prefix
  - array of string, Site slice prefix

- `date_created` : int or array of int
  - int, Date and time when site entry was created, in seconds since UNIX epoch
  - array of int, Date and time when site entry was created, in seconds since UNIX epoch

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `last_updated` : int, Date and time when site entry was last updated, in seconds since UNIX epoch
  - `node_ids` : array, List of site node identifiers
    - int
  - `site_id` : int, Site identifier
  - `pcu_ids` : array, List of PCU identifiers
    - int
  - `max_slices` : int, Maximum number of slices that the site is able to create
  - `ext_consortium_id` : int, external consortium id
  - `peer_site_id` : int, Foreign site identifier at peer
  - `abbreviated_name` : string, Abbreviated site name
  - `person_ids` : array, List of account identifiers
    - int
  - `slice_ids` : array, List of slice identifiers
    - int
  - `latitude` : double, Decimal latitude of the site
  - `peer_id` : int, Peer to which this site belongs
  - `max_slivers` : int, Maximum number of slivers that the site is able to create
  - `is_public` : boolean, Publicly viewable site
  - `address_ids` : array, List of address identifiers
    - int
  - `name` : string, Full site name
  - `url` : string, URL of a page that describes the site
  - `site_tag_ids` : array, List of tags attached to this site
    - int
  - `enabled` : boolean, Has been enabled
  - `longitude` : double, Decimal longitude of the site
  - `login_base` : string, Site slice prefix
  - `date_created` : int, Date and time when site entry was created, in seconds since UNIX epoch

GetSliceArch

Prototype:

GetSliceArch (auth, id_or_name)

Description:

Accessor 'get' method designed for Slice objects using tag arch

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- string or nil
  - string,
  - nil,

GetSliceFamily

Prototype:

GetSliceFamily (auth, slice_id_or_name)

Description:

Returns the slice vserver reference image that a given slice should be based on. This depends on the global PLC settings in the PLC_FLAVOUR area, optionnally overridden by any of the 'vref', 'arch', 'pldistro', 'fcdistro' tag if set on the slice.

Allowed Roles:

admin, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `slice_id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- string, the slicefamily this slice should be based upon

GetSliceFcdistro

Prototype:

GetSliceFcdistro (auth, id_or_name)

Description:

Accessor 'get' method designed for Slice objects using tag fcdistro

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- string or nil
  - string,
  - nil,

GetSliceHmac

Prototype:

GetSliceHmac (auth, id_or_name)

Description:

Accessor 'get' method designed for Slice objects using tag hmac

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- string or nil
  - string,

- nil,

GetSliceHrn

Prototype:

GetSliceHrn (auth, id_or_name)

Description:

Accessor 'get' method designed for Slice objects using tag hrn

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- string or nil
  - string,
  - nil,

GetSliceIPv6Address

Prototype:

GetSliceIPv6Address (auth, id_or_name)

Description:

Accessor 'get' method designed for Slice objects using tag ipv6_address

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- string or nil
  - string,
  - nil,

GetSliceInitscript

Prototype:

GetSliceInitscript (auth, id_or_name)

Description:

Accessor 'get' method designed for Slice objects using tag initscript

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- string or nil
  - string,
  - nil,

GetSliceInitscriptCode

Prototype:

GetSliceInitscriptCode (auth, id_or_name)

Description:

Accessor 'get' method designed for Slice objects using tag initscript_code

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string

- int, Slice identifier
- string, Slice name

Returns:

- string or nil
  - string,
  - nil,

GetSliceInstantiations

Prototype:

GetSliceInstantiations (auth)

Description:

Returns an array of all valid slice instantiation states.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

Returns:

- array of string, Slice instantiation state

GetSliceKeys

Prototype:

GetSliceKeys (auth, slice_filter, return_fields)

Description:

Returns an array of structs containing public key info for users in the specified slices. If slice_filter is specified and is an array of slice identifiers or slice names, or a struct of slice attributes, only slices matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Users may only query slices of which they are members. PIs may query any of the slices at their sites. Admins and nodes may query any slice. If a slice that cannot be queried is specified in slice_filter, details about that slice will not be returned.

Allowed Roles:

admin, pi, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `slice_filter` : array of int or string or struct

- array of int or string
  - int, Slice identifier
  - string, Slice name

- struct, Attribute filter
  - `creator_person_id` : int or array of int
    - int, Identifier of the account that created this slice
    - array of int, Identifier of the account that created this slice

  - `instantiation` : string or array of string
    - string, Slice instantiation state
    - array of string, Slice instantiation state

  - `description` : string or array of string
    - string, Slice description
    - array of string, Slice description

  - `slice_id` : int or array of int
    - int, Slice identifier
    - array of int, Slice identifier

  - `node_ids` : array or array of array
    - array, List of nodes in this slice
      - int

    - array of array, List of nodes in this slice
      - int

  - `url` : string or array of string
    - string, URL further describing this slice
    - array of string, URL further describing this slice

  - `max_nodes` : int or array of int
    - int, Maximum number of nodes that can be assigned to this slice
    - array of int, Maximum number of nodes that can be assigned to this slice

  - `person_ids` : array or array of array
    - array, List of accounts that can use this slice
      - int

    - array of array, List of accounts that can use this slice
      - int

- `expires` : int or array of int
  - int, Date and time when slice expires, in seconds since UNIX epoch
  - array of int, Date and time when slice expires, in seconds since UNIX epoch

- `site_id` : int or array of int
  - int, Identifier of the site to which this slice belongs
  - array of int, Identifier of the site to which this slice belongs

- `created` : int or array of int
  - int, Date and time when slice was created, in seconds since UNIX epoch
  - array of int, Date and time when slice was created, in seconds since UNIX epoch

- `peer_slice_id` : int or array of int
  - int, Foreign slice identifier at peer
  - array of int, Foreign slice identifier at peer

- `slice_tag_ids` : array or array of array
  - array, List of slice attributes
    - int

  - array of array, List of slice attributes
    - int

- `peer_id` : int or array of int
  - int, Peer to which this slice belongs
  - array of int, Peer to which this slice belongs

- `name` : string or array of string
  - string, Slice name
  - array of string, Slice name

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `person_id` : int, User identifier
  - `key` : string, Key value

- `name` : string, Slice name
- `slice_id` : int, Slice identifier
- `email` : string, Primary e-mail address

GetSliceOmfControl

Prototype:

GetSliceOmfControl (auth, id_or_name)

Description:

Accessor 'get' method designed for Slice objects using tag omf_control

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- string or nil
  - string,
  - nil,

GetSlicePldistro

Prototype:

GetSlicePldistro (auth, id_or_name)

Description:

Accessor 'get' method designed for Slice objects using tag pldistro

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier

- string, Slice name

Returns:

- string or nil
  - string,
  - nil,

GetSliceSfaCreated

Prototype:

GetSliceSfaCreated (auth, id_or_name)

Description:

Accessor 'get' method designed for Slice objects using tag sfa_created

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- string or nil
  - string,
  - nil,

GetSliceSliverHMAC

Prototype:

GetSliceSliverHMAC (auth, id_or_name)

Description:

Accessor 'get' method designed for Slice objects using tag enable_hmac

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- string or nil
  - string,
  - nil,

GetSliceSshKey

Prototype:

GetSliceSshKey (auth, id_or_name)

Description:

Accessor 'get' method designed for Slice objects using tag ssh_key

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- string or nil
  - string,
  - nil,

GetSliceTags

Prototype:

GetSliceTags (auth, slice_tag_filter, return_fields)

Description:

Returns an array of structs containing details about slice and sliver attributes. An attribute is a sliver attribute if the node_id field is set. If slice_tag_filter is specified and is an array of slice attribute identifiers, or a struct of slice attribute attributes, only slice attributes matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Users may only query attributes of slices or slivers of which they are members. PIs may only query attributes of slices or slivers at their sites, or of which they are members. Admins may query attributes of any slice or sliver.

Allowed Roles:

admin, pi, user, node

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *slice_tag_filter* : array of int or struct
  - array of int, Slice tag identifier
  - struct, Attribute filter
    - *category* : string or array of string
      - string, Node tag category
      - array of string, Node tag category

    - *description* : string or array of string
      - string, Node tag type description
      - array of string, Node tag type description

    - *slice_id* : int or array of int
      - int, Slice identifier
      - array of int, Slice identifier

    - *value* : string or array of string
      - string, Slice attribute value
      - array of string, Slice attribute value

    - *nodegroup_id* : int or array of int
      - int, Node group identifier
      - array of int, Node group identifier

    - *slice_tag_id* : int or array of int
      - int, Slice tag identifier
      - array of int, Slice tag identifier

    - *tagname* : string or array of string
      - string, Node tag type name
      - array of string, Node tag type name

    - *tag_type_id* : int or array of int
      - int, Node tag type identifier
      - array of int, Node tag type identifier

- *node_id* : int or array of int
  - int, Node identifier
  - array of int, Node identifier

- *name* : string or array of string
  - string, Slice name
  - array of string, Slice name

- *return_fields* : array, List of fields to return
  - string

Returns:

- array of struct
  - *nodegroup_id* : int, Node group identifier
  - *category* : string, Node tag category
  - *node_id* : int, Node identifier
  - *slice_tag_id* : int, Slice tag identifier
  - *slice_id* : int, Slice identifier
  - *tag_type_id* : int, Node tag type identifier
  - *description* : string, Node tag type description
  - *tagname* : string, Node tag type name
  - *value* : string, Slice attribute value
  - *name* : string, Slice name

GetSliceTicket

Prototype:

GetSliceTicket (auth, slice_id_or_name)

Description:

Returns a ticket for, or signed representation of, the specified slice. Slice tickets may be used to manually instantiate or update a slice on a node. Present this ticket to the local Node Manager interface to redeem it.

If the slice has not been added to a node with AddSliceToNodes, and the ticket is redeemed on that node, it will be deleted the next time the Node Manager contacts the API.

Users may only obtain tickets for slices of which they are members. PIs may obtain tickets for any of the slices at their sites, or any slices of which they are members. Admins may obtain tickets for any slice.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, user, peer

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `slice_id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- string, Signed slice ticket

GetSliceVref

Prototype:

GetSliceVref (auth, id_or_name)

Description:

Accessor 'get' method designed for Slice objects using tag vref

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- string or nil
  - string,
  - nil,

GetSlices

Prototype:

GetSlices (auth, slice_filter, return_fields)

Description:

Returns an array of structs containing details about slices. If slice_filter is specified and is an array of slice identifiers or slice names, or a struct of slice attributes, only slices matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Users may only query slices of which they are members. PIs may query any of the slices at their sites. Admins and nodes may query any slice. If a slice that cannot be queried is specified in slice_filter, details about that slice will not be returned.

Allowed Roles:

admin, pi, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `slice_filter` : array of int or string or string or int or struct
  - array of int or string
    - int, Slice identifier
    - string, Slice name

  - string, name
  - int, slice_id
  - struct, Attribute filter
    - `creator_person_id` : int or array of int
      - int, Identifier of the account that created this slice
      - array of int, Identifier of the account that created this slice

    - `instantiation` : string or array of string
      - string, Slice instantiation state
      - array of string, Slice instantiation state

    - `description` : string or array of string
      - string, Slice description
      - array of string, Slice description

    - `slice_id` : int or array of int
      - int, Slice identifier
      - array of int, Slice identifier

    - `node_ids` : array or array of array
      - array, List of nodes in this slice
        - int

      - array of array, List of nodes in this slice
        - int

    - `url` : string or array of string
      - string, URL further describing this slice
      - array of string, URL further describing this slice

    - `max_nodes` : int or array of int

  - int, Maximum number of nodes that can be assigned to this slice
  - array of int, Maximum number of nodes that can be assigned to this slice

- `person_ids` : array or array of array
  - array, List of accounts that can use this slice
    - int
  - array of array, List of accounts that can use this slice
    - int

- `expires` : int or array of int
  - int, Date and time when slice expires, in seconds since UNIX epoch
  - array of int, Date and time when slice expires, in seconds since UNIX epoch

- `site_id` : int or array of int
  - int, Identifier of the site to which this slice belongs
  - array of int, Identifier of the site to which this slice belongs

- `created` : int or array of int
  - int, Date and time when slice was created, in seconds since UNIX epoch
  - array of int, Date and time when slice was created, in seconds since UNIX epoch

- `peer_slice_id` : int or array of int
  - int, Foreign slice identifier at peer
  - array of int, Foreign slice identifier at peer

- `slice_tag_ids` : array or array of array
  - array, List of slice attributes
    - int
  - array of array, List of slice attributes
    - int

- `peer_id` : int or array of int
  - int, Peer to which this slice belongs
  - array of int, Peer to which this slice belongs

- `name` : string or array of string
  - string, Slice name
  - array of string, Slice name

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `description` : string, Slice description
  - `node_ids` : array, List of nodes in this slice
    - int
  - `expires` : int, Date and time when slice expires, in seconds since UNIX epoch
  - `site_id` : int, Identifier of the site to which this slice belongs
  - `creator_person_id` : int, Identifier of the account that created this slice
  - `instantiation` : string, Slice instantiation state
  - `name` : string, Slice name
  - `slice_id` : int, Slice identifier
  - `created` : int, Date and time when slice was created, in seconds since UNIX epoch
  - `url` : string, URL further describing this slice
  - `max_nodes` : int, Maximum number of nodes that can be assigned to this slice
  - `person_ids` : array, List of accounts that can use this slice
    - int
  - `peer_slice_id` : int, Foreign slice identifier at peer
  - `slice_tag_ids` : array, List of slice attributes
    - int
  - `peer_id` : int, Peer to which this slice belongs

GetSlivers

Prototype:

GetSlivers (auth, node_id_or_hostname)

Description:

Returns a struct containing information about the specified node (or calling node, if called by a node and node_id_or_hostname is not specified), including the current set of slivers bound to the node.

All of the information returned by this call can be gathered from other calls, e.g. GetNodes, GetInterfaces, GetSlices, etc. This function exists almost solely for the benefit of Node Manager.

Allowed Roles:

admin, node

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *node_id_or_hostname* : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- struct
  - *xmpp* : struct
    - *password* : string, username for the XMPP server
    - *user* : string, username for the XMPP server
    - *server* : string, hostname for the XMPP server

  - *interfaces* : array of struct
    - *is_primary* : boolean, Is the primary interface for this node
    - *last_updated* : int, Date and time when node entry was created
    - *network* : string, Subnet address
    - *ip* : string, IP address
    - *dns1* : string, IP address of primary DNS server
    - *hostname* : string, (Optional) Hostname
    - *netmask* : string, Subnet mask
    - *interface_tag_ids* : array, List of interface settings
      - int

    - *interface_id* : int, Node interface identifier
    - *broadcast* : string, Network broadcast address
    - *mac* : string, MAC address
    - *node_id* : int, Node associated with this interface
    - *gateway* : string, IP address of primary gateway
    - *dns2* : string, IP address of secondary DNS server
    - *bwlimit* : int, Bandwidth limit
    - *type* : string, Address type (e.g., 'ipv4')
    - *method* : string, Addressing method (e.g., 'static' or 'dhcp')

  - *conf_files* : array of struct
    - *postinstall_cmd* : string, Shell command to execute after installing
    - *preinstall_cmd* : string, Shell command to execute prior to installing
    - *node_ids* : int, List of nodes linked to this file
    - *dest* : string, Absolute path where file should be installed
    - *ignore_cmd_errors* : boolean, Install file anyway even if an error occurs
    - *file_permissions* : string, chmod(1) permissions
    - *always_update* : boolean, Always attempt to install file even if unchanged
    - *file_group* : string, chgrp(1) owner

- *file_owner* : string, chown(1) owner
- *error_cmd* : string, Shell command to execute if any error occurs
- *nodegroup_ids* : int, List of node groups linked to this file
- *enabled* : boolean, Configuration file is active
- *conf_file_id* : int, Configuration file identifier
- *source* : string, Relative path on the boot server where file can be downloaded

- *node_id* : int, Node identifier
- *accounts* : array of struct
  - *keys* : array of struct
    - *key_type* : string, Key type
    - *key* : string, Key value
  - *name* : string, unix style account name
- *groups* : array of string, Node group name
- *leases* : array of struct
  - *t_from* : int or string
    - int, timeslot start (unix timestamp)
    - string, timeslot start (formatted as %Y-%m-%d %H:%M:%S)
  - *slice_id* : int, Slice identifier
  - *t_until* : int or string
    - int, timeslot end (unix timestamp)
    - string, timeslot end (formatted as %Y-%m-%d %H:%M:%S)
- *reservation_policy* : string, one among none, lease_or_idle, lease_or_shared
- *slivers* : array of struct
  - *instantiation* : string, Slice instantiation state
  - *name* : string, Slice name
  - *slice_id* : int, Slice identifier
  - *keys* : array of struct
    - *key_type* : string, Key type
    - *key* : string, Key value
  - *expires* : int, Date and time when slice expires, in seconds since UNIX epoch
  - *attributes* : array of struct
    - *value* : string, Slice attribute value
    - *tagname* : string, Node tag type name

- *hostname* : string, Fully qualified hostname
- *initscripts* : array of struct
  - *initscript_id* : int, Initscript identifier
  - *enabled* : boolean, Initscript is active
  - *name* : string, Initscript name
  - *script* : string, Initscript

- *timestamp* : int, Timestamp of this call, in seconds since UNIX epoch

GetTagTypes

Prototype:

GetTagTypes (auth, tag_type_filter, return_fields)

Description:

Returns an array of structs containing details about node tag types.

The usual filtering scheme applies on this method.

Allowed Roles:

admin, pi, user, tech, node

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *tag_type_filter* : array of int or string or int or string or struct
  - array of int or string
    - int, Node tag type identifier
    - string, Node tag type name

  - int or string
    - int, Node tag type identifier
    - string, Node tag type name

  - struct, Attribute filter
    - *category* : string or array of string
      - string, Node tag category
      - array of string, Node tag category

    - *role_ids* : array or array of array
      - array, List of role identifiers
        - int

      - array of array, List of role identifiers
        - int

- `description` : string or array of string
  - string, Node tag type description
  - array of string, Node tag type description

- `roles` : array or array of array
  - array, List of roles
    - string

  - array of array, List of roles
    - string

- `tagname` : string or array of string
  - string, Node tag type name
  - array of string, Node tag type name

- `tag_type_id` : int or array of int
  - int, Node tag type identifier
  - array of int, Node tag type identifier

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `category` : string, Node tag category
  - `role_ids` : array, List of role identifiers
    - int

  - `description` : string, Node tag type description
  - `roles` : array, List of roles
    - string

  - `tagname` : string, Node tag type name
  - `tag_type_id` : int, Node tag type identifier

GetWhitelist

Prototype:

GetWhitelist (auth, node_filter, return_fields)

Description:

Returns an array of structs containing details about the specified nodes whitelists. If node_filter is specified and is an array of node identifiers or hostnames, or a struct of node attributes, only nodes matching the filter will be returned. If return_fields is specified, only the specified details will be returned.

Some fields may only be viewed by admins.

Allowed Roles:

admin, pi, user, tech, node, anonymous

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *node_filter* : array of int or string or struct
  - array of int or string
    - int, Node identifier
    - string, Fully qualified hostname

  - struct, Attribute filter
    - *last_updated* : int or array of int
      - int, Date and time when node entry was created
      - array of int, Date and time when node entry was created

    - *key* : string or array of string
      - string, (Admin only) Node key
      - array of string, (Admin only) Node key

    - *boot_state* : string or array of string
      - string, Boot state
      - array of string, Boot state

    - *site_id* : int or array of int
      - int, Site at which this node is located
      - array of int, Site at which this node is located

    - *pcu_ids* : array or array of array
      - array, List of PCUs that control this node
        - int

      - array of array, List of PCUs that control this node
        - int

- `node_type` : string or array of string
  - string, Node type
  - array of string, Node type

- `session` : string or array of string
  - string, (Admin only) Node session value
  - array of string, (Admin only) Node session value

- `ssh_rsa_key` : string or array of string
  - string, Last known SSH host key
  - array of string, Last known SSH host key

- `last_pcu_reboot` : int or array of int
  - int, Date and time when PCU reboot was attempted
  - array of int, Date and time when PCU reboot was attempted

- `node_tag_ids` : array or array of array
  - array, List of tags attached to this node
    - int

  - array of array, List of tags attached to this node
    - int

- `verified` : boolean or array of boolean
  - boolean, Whether the node configuration is verified correct
  - array of boolean, Whether the node configuration is verified correct

- `last_contact` : int or array of int
  - int, Date and time when node last contacted plc
  - array of int, Date and time when node last contacted plc

- `peer_node_id` : int or array of int
  - int, Foreign node identifier at peer
  - array of int, Foreign node identifier at peer

- `hostname` : string or array of string
  - string, Fully qualified hostname
  - array of string, Fully qualified hostname

- `run_level` : string or array of string
  - string, Run level
  - array of string, Run level

- `slice_ids` : array or array of array
  - array, List of slices on this node
    - int
  - array of array, List of slices on this node
    - int
- `last_time_spent_offline` : int or array of int
  - int, Length of time the node was last offline after failure and before reboot
  - array of int, Length of time the node was last offline after failure and before reboot
- `version` : string or array of string
  - string, Apparent Boot CD version
  - array of string, Apparent Boot CD version
- `peer_id` : int or array of int
  - int, Peer to which this node belongs
  - array of int, Peer to which this node belongs
- `node_id` : int or array of int
  - int, Node identifier
  - array of int, Node identifier
- `last_boot` : int or array of int
  - int, Date and time when node last booted
  - array of int, Date and time when node last booted
- `interface_ids` : array or array of array
  - array, List of network interfaces that this node has
    - int
  - array of array, List of network interfaces that this node has
    - int
- `conf_file_ids` : array or array of array
  - array, List of configuration files specific to this node
    - int
  - array of array, List of configuration files specific to this node
    - int

- `last_pcu_confirmation` : int or array of int
  - int, Date and time when PCU reboot was confirmed
  - array of int, Date and time when PCU reboot was confirmed

- `nodegroup_ids` : array or array of array
  - array, List of node groups that this node is in
    - int

  - array of array, List of node groups that this node is in
    - int

- `slice_ids_whitelist` : array or array of array
  - array, List of slices allowed on this node
    - int

  - array of array, List of slices allowed on this node
    - int

- `last_time_spent_online` : int or array of int
  - int, Length of time the node was last online before shutdown/failure
  - array of int, Length of time the node was last online before shutdown/failure

- `boot_nonce` : string or array of string
  - string, (Admin only) Random value generated by the node at last boot
  - array of string, (Admin only) Random value generated by the node at last boot

- `last_download` : int or array of int
  - int, Date and time when node boot image was created
  - array of int, Date and time when node boot image was created

- `date_created` : int or array of int
  - int, Date and time when node entry was created
  - array of int, Date and time when node entry was created

- `model` : string or array of string
  - string, Make and model of the actual machine
  - array of string, Make and model of the actual machine

- `ports` : array or array of array

    - array, List of PCU ports that this node is connected to
      - int

    - array of array, List of PCU ports that this node is connected to
      - int

- `return_fields` : array, List of fields to return
  - string

Returns:

- array of struct
  - `last_updated` : int, Date and time when node entry was created
  - `key` : string, (Admin only) Node key
  - `session` : string, (Admin only) Node session value
  - `boot_state` : string, Boot state
  - `site_id` : int, Site at which this node is located
  - `pcu_ids` : array, List of PCUs that control this node
    - int

  - `node_type` : string, Node type
  - `node_id` : int, Node identifier
  - `last_boot` : int, Date and time when node last booted
  - `interface_ids` : array, List of network interfaces that this node has
    - int

  - `slice_ids_whitelist` : array, List of slices allowed on this node
    - int

  - `run_level` : string, Run level
  - `ssh_rsa_key` : string, Last known SSH host key
  - `last_pcu_reboot` : int, Date and time when PCU reboot was attempted
  - `node_tag_ids` : array, List of tags attached to this node
    - int

  - `nodegroup_ids` : array, List of node groups that this node is in
    - int

  - `verified` : boolean, Whether the node configuration is verified correct
  - `last_contact` : int, Date and time when node last contacted plc
  - `peer_node_id` : int, Foreign node identifier at peer
  - `hostname` : string, Fully qualified hostname

- `last_time_spent_offline` : int, Length of time the node was last offline after failure and before reboot
- `conf_file_ids` : array, List of configuration files specific to this node
  - int
- `last_time_spent_online` : int, Length of time the node was last online before shutdown/failure
- `slice_ids` : array, List of slices on this node
  - int
- `boot_nonce` : string, (Admin only) Random value generated by the node at last boot
- `version` : string, Apparent Boot CD version
- `last_pcu_confirmation` : int, Date and time when PCU reboot was confirmed
- `last_download` : int, Date and time when node boot image was created
- `date_created` : int, Date and time when node entry was created
- `model` : string, Make and model of the actual machine
- `peer_id` : int, Peer to which this node belongs
- `ports` : array, List of PCU ports that this node is connected to
  - int

NotifyPersons

Prototype:

NotifyPersons (auth, person_filter, subject, body)

Description:

Sends an e-mail message to the specified users. If person_filter is specified and is an array of user identifiers or usernames, or a struct of user attributes, only users matching the filter will receive the message.

Returns 1 if successful.

Allowed Roles:

admin, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `person_filter` : array of int or string or struct
  - array of int or string
    - int, User identifier
    - string, Primary e-mail address
  - struct, Attribute filter

- `role_ids` : array or array of array
  - array, List of role identifiers
    - int
  - array of array, List of role identifiers
    - int
- `last_updated` : int or array of int
  - int, Date and time of last update
  - array of int, Date and time of last update
- `last_name` : string or array of string
  - string, Surname
  - array of string, Surname
- `site_ids` : array or array of array
  - array, List of site identifiers
    - int
  - array of array, List of site identifiers
    - int
- `first_name` : string or array of string
  - string, Given name
  - array of string, Given name
- `title` : string or array of string
  - string, Title
  - array of string, Title
- `slice_ids` : array or array of array
  - array, List of slice identifiers
    - int
  - array of array, List of slice identifiers
    - int
- `person_id` : int or array of int
  - int, User identifier
  - array of int, User identifier

- `peer_id` : int or array of int
  - int, Peer to which this user belongs
  - array of int, Peer to which this user belongs

- `email` : string or array of string
  - string, Primary e-mail address
  - array of string, Primary e-mail address

- `bio` : string or array of string
  - string, Biography
  - array of string, Biography

- `key_ids` : array or array of array
  - array, List of key identifiers
    - int

  - array of array, List of key identifiers
    - int

- `phone` : string or array of string
  - string, Telephone number
  - array of string, Telephone number

- `peer_person_id` : int or array of int
  - int, Foreign user identifier at peer
  - array of int, Foreign user identifier at peer

- `person_tag_ids` : array or array of array
  - array, List of tags attached to this person
    - int

  - array of array, List of tags attached to this person
    - int

- `password` : string or array of string
  - string, Account password in crypt() form
  - array of string, Account password in crypt() form

- `roles` : array or array of array
  - array, List of roles
    - string

    - array of array, List of roles
      - string

- `url` : string or array of string
  - string, Home page
  - array of string, Home page

- `verification_key` : string or array of string
  - string, Reset password key
  - array of string, Reset password key

- `enabled` : boolean or array of boolean
  - boolean, Has been enabled
  - array of boolean, Has been enabled

- `date_created` : int or array of int
  - int, Date and time when account was created
  - array of int, Date and time when account was created

- `verification_expires` : int or array of int
  - int, Date and time when verification_key expires
  - array of int, Date and time when verification_key expires

- `subject` : string, E-mail subject
- `body` : string, E-mail body

Returns:

- int, 1 if successful

NotifySupport

Prototype:

NotifySupport (auth, subject, body)

Description:

Sends an e-mail message to the configured support address.
Returns 1 if successful.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `subject` : string, E-mail subject
- `body` : string, E-mail body

Returns:

- int, 1 if successful

RebootNode

Prototype:

RebootNode (auth, node_id_or_hostname)

Description:

Sends the specified node a specially formatted UDP packet which should cause it to reboot immediately.

Admins can reboot any node. Techs and PIs can only reboot nodes at their site.

Returns 1 if the packet was successfully sent (which only whether the packet was sent, not whether the reboot was successful).

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `node_id_or_hostname` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- int, 1 if successful

RebootNodeWithPCU

Prototype:

RebootNodeWithPCU (auth, node_id_or_hostname, testrun)

Description:

Uses the associated PCU to attempt to reboot the given Node.

Admins can reboot any node. Techs and PIs can only reboot nodes at their site.

Returns 1 if the reboot proceeded without error (Note: this does not guarantee that the reboot is successful). Returns -1 if external dependencies for this call are not available. Returns "error string" if the reboot failed with a specific message.

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure

  - `AuthMethod` : string, Authentication method to use

- `node_id_or_hostname` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

- `testrun` : boolean, Run as a test, or as a real reboot

Returns:

- int, 1 if successful

RefreshPeer

Prototype:

RefreshPeer (auth, peer_id_or_peername)

Description:

Fetches site, node, slice, person and key data from the specified peer and caches it locally; also deletes stale entries. Upon successful completion, returns a dict reporting various timers. Faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `peer_id_or_peername` : int or string
  - int, Peer identifier
  - string, Peer name

Returns:

- int, 1 if successful

ReportRunlevel

Prototype:

ReportRunlevel (auth, report_fields, node_id_or_hostname)

Description:

report runlevel

Allowed Roles:

node, admin

Parameters:

- `auth` : struct or struct or struct

- struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use, always 'hmac'
  - `value` : string, HMAC of node key and method call
  - `node_id` : int, Node identifier

- struct, API authentication structure
  - `session` : string, Session key
  - `AuthMethod` : string, Authentication method to use, always 'session'

- struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `report_fields` : struct
  - `run_level` : string, Run level

- `node_id_or_hostname` : int or string
  - int, Node identifier
  - string, Fully qualified hostname

Returns:

- int, 1 if successful

ResetPassword

Prototype:

ResetPassword (auth, person_id_or_email, verification_key, verification_expires)

Description:

If verification_key is not specified, then a new verification_key will be generated and stored with the user's account. The key will be e-mailed to the user in the form of a link to a web page.

The web page should verify the key by calling this function again and specifying verification_key. If the key matches what has been stored in the user's account, a new random password will be e-mailed to the user.

Returns 1 if verification_key was not specified, or was specified and is valid, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `person_id_or_email` : int or string
  - int, User identifier
  - string, Primary e-mail address

- *verification_key* : string, Reset password key
- *verification_expires* : int, Date and time when verification_key expires

Returns:

- int, 1 if verification_key is valid

ResolveSlices

Prototype:

ResolveSlices (auth, slice_filter)

Description:

This method is similar to GetSlices, except that (1) the returned columns are restricted to 'name', 'slice_id' and 'expires', and (2) it returns expired slices too. This method is designed to help third-party software solve slice names from their slice_id (e.g. PlanetFlow Central). For this reason it is accessible with anonymous authentication (among others).

Allowed Roles:

admin, pi, user, tech, anonymous

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *slice_filter* : array of int or string or string or int or struct
  - array of int or string
    - int, Slice identifier
    - string, Slice name

  - string, name
  - int, slice_id
  - struct, Attribute filter
    - *expires* : int or array of int
      - int, Date and time when slice expires, in seconds since UNIX epoch
      - array of int, Date and time when slice expires, in seconds since UNIX epoch

    - *name* : string or array of string
      - string, Slice name
      - array of string, Slice name

    - *slice_id* : int or array of int
      - int, Slice identifier
      - array of int, Slice identifier

Returns:

- array of struct
  - `expires` : int, Date and time when slice expires, in seconds since UNIX epoch
  - `name` : string, Slice name
  - `slice_id` : int, Slice identifier

RetrieveSlicePersonKeys

Prototype:

RetrieveSlicePersonKeys (auth, slice_id_or_name, person_filter)

Description:

This method exposes the public ssh keys for people in a slice It expects a slice name or id, and returns a dictionary on emails. This method is designed to help third-party software authenticate users (e.g. the OMF Experiment Controller). For this reason it is accessible with anonymous authentication.

Allowed Roles:

admin, pi, user, tech, anonymous

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `slice_id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

- `person_filter` : struct, Attribute filter
  - `role_ids` : array or array of array
    - array, List of role identifiers
      - int

    - array of array, List of role identifiers
      - int

  - `last_updated` : int or array of int
    - int, Date and time of last update
    - array of int, Date and time of last update

  - `last_name` : string or array of string
    - string, Surname
    - array of string, Surname

- `site_ids` : array or array of array
  - array, List of site identifiers
    - int
  - array of array, List of site identifiers
    - int

- `first_name` : string or array of string
  - string, Given name
  - array of string, Given name

- `title` : string or array of string
  - string, Title
  - array of string, Title

- `slice_ids` : array or array of array
  - array, List of slice identifiers
    - int
  - array of array, List of slice identifiers
    - int

- `person_id` : int or array of int
  - int, User identifier
  - array of int, User identifier

- `peer_id` : int or array of int
  - int, Peer to which this user belongs
  - array of int, Peer to which this user belongs

- `email` : string or array of string
  - string, Primary e-mail address
  - array of string, Primary e-mail address

- `bio` : string or array of string
  - string, Biography
  - array of string, Biography

- `key_ids` : array or array of array
  - array, List of key identifiers
    - int

    - array of array, List of key identifiers
      - int

- `phone` : string or array of string
  - string, Telephone number
  - array of string, Telephone number

- `peer_person_id` : int or array of int
  - int, Foreign user identifier at peer
  - array of int, Foreign user identifier at peer

- `person_tag_ids` : array or array of array
  - array, List of tags attached to this person
    - int

  - array of array, List of tags attached to this person
    - int

- `password` : string or array of string
  - string, Account password in crypt() form
  - array of string, Account password in crypt() form

- `roles` : array or array of array
  - array, List of roles
    - string

  - array of array, List of roles
    - string

- `url` : string or array of string
  - string, Home page
  - array of string, Home page

- `verification_key` : string or array of string
  - string, Reset password key
  - array of string, Reset password key

- `enabled` : boolean or array of boolean
  - boolean, Has been enabled
  - array of boolean, Has been enabled

- `date_created` : int or array of int
  - int, Date and time when account was created
  - array of int, Date and time when account was created

- `verification_expires` : int or array of int
  - int, Date and time when verification_key expires
  - array of int, Date and time when verification_key expires

Returns:

- struct, ssh keys hashed on emails

RetrieveSliceSliverKeys

Prototype:

RetrieveSliceSliverKeys (auth, slice_id_or_name, node_filter)

Description:

This method exposes the public ssh keys for a slice's slivers. It expects a slice name or id, and returns a dictionary on hostnames. This method is designed to help third-party software authenticate slivers (e.g. the OMF Experiment Controller). For this reason it is accessible with anonymous authentication.

Allowed Roles:

admin, pi, user, tech, anonymous

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `slice_id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name

- `node_filter` : struct, Attribute filter
  - `last_updated` : int or array of int
    - int, Date and time when node entry was created
    - array of int, Date and time when node entry was created

  - `key` : string or array of string
    - string, (Admin only) Node key
    - array of string, (Admin only) Node key

  - `boot_state` : string or array of string
    - string, Boot state
    - array of string, Boot state

- `site_id` : int or array of int
  - int, Site at which this node is located
  - array of int, Site at which this node is located

- `pcu_ids` : array or array of array
  - array, List of PCUs that control this node
    - int

  - array of array, List of PCUs that control this node
    - int

- `node_type` : string or array of string
  - string, Node type
  - array of string, Node type

- `session` : string or array of string
  - string, (Admin only) Node session value
  - array of string, (Admin only) Node session value

- `ssh_rsa_key` : string or array of string
  - string, Last known SSH host key
  - array of string, Last known SSH host key

- `last_pcu_reboot` : int or array of int
  - int, Date and time when PCU reboot was attempted
  - array of int, Date and time when PCU reboot was attempted

- `node_tag_ids` : array or array of array
  - array, List of tags attached to this node
    - int

  - array of array, List of tags attached to this node
    - int

- `verified` : boolean or array of boolean
  - boolean, Whether the node configuration is verified correct
  - array of boolean, Whether the node configuration is verified correct

- `last_contact` : int or array of int
  - int, Date and time when node last contacted plc
  - array of int, Date and time when node last contacted plc

- *peer_node_id* : int or array of int
  - int, Foreign node identifier at peer
  - array of int, Foreign node identifier at peer

- *hostname* : string or array of string
  - string, Fully qualified hostname
  - array of string, Fully qualified hostname

- *run_level* : string or array of string
  - string, Run level
  - array of string, Run level

- *slice_ids* : array or array of array
  - array, List of slices on this node
    - int

  - array of array, List of slices on this node
    - int

- *last_time_spent_offline* : int or array of int
  - int, Length of time the node was last offline after failure and before reboot
  - array of int, Length of time the node was last offline after failure and before reboot

- *version* : string or array of string
  - string, Apparent Boot CD version
  - array of string, Apparent Boot CD version

- *peer_id* : int or array of int
  - int, Peer to which this node belongs
  - array of int, Peer to which this node belongs

- *node_id* : int or array of int
  - int, Node identifier
  - array of int, Node identifier

- *last_boot* : int or array of int
  - int, Date and time when node last booted
  - array of int, Date and time when node last booted

- *interface_ids* : array or array of array
  - array, List of network interfaces that this node has
    - int

  - array of array, List of network interfaces that this node has
    - int

- `conf_file_ids` : array or array of array
  - array, List of configuration files specific to this node
    - int
  - array of array, List of configuration files specific to this node
    - int

- `last_pcu_confirmation` : int or array of int
  - int, Date and time when PCU reboot was confirmed
  - array of int, Date and time when PCU reboot was confirmed

- `nodegroup_ids` : array or array of array
  - array, List of node groups that this node is in
    - int
  - array of array, List of node groups that this node is in
    - int

- `slice_ids_whitelist` : array or array of array
  - array, List of slices allowed on this node
    - int
  - array of array, List of slices allowed on this node
    - int

- `last_time_spent_online` : int or array of int
  - int, Length of time the node was last online before shutdown/failure
  - array of int, Length of time the node was last online before shutdown/failure

- `boot_nonce` : string or array of string
  - string, (Admin only) Random value generated by the node at last boot
  - array of string, (Admin only) Random value generated by the node at last boot

- `last_download` : int or array of int
  - int, Date and time when node boot image was created
  - array of int, Date and time when node boot image was created

- `date_created` : int or array of int
  - int, Date and time when node entry was created
  - array of int, Date and time when node entry was created

- `model` : string or array of string
  - string, Make and model of the actual machine
  - array of string, Make and model of the actual machine

- `ports` : array or array of array
  - array, List of PCU ports that this node is connected to
    - int

  - array of array, List of PCU ports that this node is connected to
    - int

Returns:

- struct, ssh keys hashed on hostnames

SetInterfaceAlias

Prototype:

SetInterfaceAlias (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag alias

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address

- `value` : string, New tag value

Returns:

- nil,

SetInterfaceBackdoor

Prototype:

SetInterfaceBackdoor (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag backdoor

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceChannel

Prototype:

SetInterfaceChannel (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag channel

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceDriver

Prototype:

SetInterfaceDriver (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag driver

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceEssid

Prototype:

SetInterfaceEssid (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag essid

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceFreq

Prototype:

SetInterfaceFreq (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag freq

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceIfname

Prototype:

SetInterfaceIfname (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag ifname

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceIwconfig

Prototype:

SetInterfaceIwconfig (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag iwconfig

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceIwpriv

Prototype:

SetInterfaceIwpriv (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag iwpriv

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceKey

Prototype:

SetInterfaceKey (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag key

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceKey1

Prototype:

SetInterfaceKey1 (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag key1

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceKey2

Prototype:

SetInterfaceKey2 (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag key2

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceKey3

Prototype:

SetInterfaceKey3 (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag key3

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceKey4

Prototype:

SetInterfaceKey4 (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag key4

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceMode

Prototype:

SetInterfaceMode (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag mode

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceNw

Prototype:

SetInterfaceNw (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag nw

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceRate

Prototype:

SetInterfaceRate (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag rate

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceSecurityMode

Prototype:

SetInterfaceSecurityMode (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag securitymode

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceSens

Prototype:

SetInterfaceSens (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag sens

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetInterfaceSliversIPv6Prefix

Prototype:

SetInterfaceSliversIPv6Prefix (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Interface objects using tag sliversipv6prefix

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node interface identifier
  - string, IP address
- `value` : string, New tag value

Returns:

- nil,

SetNodeArch

Prototype:

SetNodeArch (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Node objects using tag arch

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- `value` : string, New tag value

Returns:

- nil,

SetNodeCramfs

Prototype:

SetNodeCramfs (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Node objects using tag cramfs

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- `value` : string, New tag value

Returns:

- nil,

SetNodeDeployment

Prototype:

SetNodeDeployment (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Node objects using tag deployment

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- `value` : string, New tag value

Returns:

- nil,

SetNodeExtensions

Prototype:

SetNodeExtensions (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Node objects using tag extensions

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- `value` : string, New tag value

Returns:

- nil,

SetNodeFcdistro

Prototype:

SetNodeFcdistro (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Node objects using tag fcdistro

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- `value` : string, New tag value

Returns:

- nil,

SetNodeHrn

Prototype:

SetNodeHrn (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Node objects using tag hrn

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- `value` : string, New tag value

Returns:

- nil,

SetNodeKargs

Prototype:

SetNodeKargs (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Node objects using tag kargs

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- `value` : string, New tag value

Returns:

- nil,

SetNodeKvariant

Prototype:

SetNodeKvariant (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Node objects using tag kvariant

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- `value` : string, New tag value

Returns:

- nil,

SetNodeNoHangcheck

Prototype:

SetNodeNoHangcheck (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Node objects using tag no-hangcheck

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- `value` : string, New tag value

Returns:

- nil,

SetNodePlainBootstrapfs

Prototype:

SetNodePlainBootstrapfs (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Node objects using tag plain-bootstrapfs

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- `value` : string, New tag value

Returns:

- nil,

SetNodePldistro

Prototype:

SetNodePldistro (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Node objects using tag pldistro

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- `value` : string, New tag value

Returns:

- nil,

SetNodeSerial

Prototype:

SetNodeSerial (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Node objects using tag serial

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- `value` : string, New tag value

Returns:

- nil,

SetNodeVirt

Prototype:

SetNodeVirt (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Node objects using tag virt

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Node identifier
  - string, Fully qualified hostname
- `value` : string, New tag value

Returns:

- nil,

SetPersonAdvanced

Prototype:

SetPersonAdvanced (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Person objects using tag advanced

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, User identifier
  - string, Primary e-mail address
- `value` : string, New tag value

Returns:

- nil,

SetPersonColumnconf

Prototype:

SetPersonColumnconf (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Person objects using tag columnconf

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, User identifier
  - string, Primary e-mail address
- `value` : string, New tag value

Returns:

- nil,

SetPersonHrn

Prototype:

SetPersonHrn (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Person objects using tag hrn

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, User identifier
  - string, Primary e-mail address
- `value` : string, New tag value

Returns:

- nil,

SetPersonPrimarySite

Prototype:

SetPersonPrimarySite (auth, person_id_or_email, site_id_or_login_base)

Description:

Makes the specified site the person's primary site. The person must already be a member of the site.

Admins may update anyone. All others may only update themselves.

Allowed Roles:

admin, pi, user, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `person_id_or_email` : int or string
  - int, User identifier
  - string, Primary e-mail address
- `site_id_or_login_base` : int or string
  - int, Site identifier
  - string, Site slice prefix

Returns:

- int, 1 if successful

SetPersonSfaCreated

Prototype:

SetPersonSfaCreated (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Person objects using tag sfa_created

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, User identifier
  - string, Primary e-mail address
- `value` : string, New tag value

Returns:

- nil,

SetPersonShowconf

Prototype:

SetPersonShowconf (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Person objects using tag showconf

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, User identifier
  - string, Primary e-mail address
- `value` : string, New tag value

Returns:

- nil,

SetSiteHrn

Prototype:

SetSiteHrn (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Site objects using tag hrn

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Site identifier
  - string, Site slice prefix
- `value` : string, New tag value

Returns:

- nil,

SetSiteSfaCreated

Prototype:

SetSiteSfaCreated (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Site objects using tag sfa_created

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Site identifier
  - string, Site slice prefix
- `value` : string, New tag value

Returns:

- nil,

SetSliceArch

Prototype:

SetSliceArch (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Slice objects using tag arch

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `value` : string, New tag value

Returns:

- nil,

SetSliceFcdistro

Prototype:

SetSliceFcdistro (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Slice objects using tag fcdistro

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `value` : string, New tag value

Returns:

- nil,

SetSliceHmac

Prototype:

SetSliceHmac (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Slice objects using tag hmac

Allowed Roles:

admin, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `value` : string, New tag value

Returns:

- nil,

SetSliceHrn

Prototype:

SetSliceHrn (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Slice objects using tag hrn

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `value` : string, New tag value

Returns:

- nil,

SetSliceIPv6Address

Prototype:

SetSliceIPv6Address (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Slice objects using tag ipv6_address

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `value` : string, New tag value

Returns:

- nil,

SetSliceInitscript

Prototype:

SetSliceInitscript (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Slice objects using tag initscript

Allowed Roles:

admin, pi, user

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `value` : string, New tag value

Returns:

- nil,

SetSliceInitscriptCode

Prototype:

SetSliceInitscriptCode (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Slice objects using tag initscript_code

Allowed Roles:

admin, pi, user

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `value` : string, New tag value

Returns:

- nil,

SetSliceOmfControl

Prototype:

SetSliceOmfControl (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Slice objects using tag omf_control

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `value` : string, New tag value

Returns:

- nil,

SetSlicePldistro

Prototype:

SetSlicePldistro (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Slice objects using tag pldistro

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `value` : string, New tag value

Returns:

- nil,

SetSliceSfaCreated

Prototype:

SetSliceSfaCreated (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Slice objects using tag sfa_created

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `value` : string, New tag value

Returns:

- nil,

SetSliceSliverHMAC

Prototype:

SetSliceSliverHMAC (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Slice objects using tag enable_hmac

Allowed Roles:

admin, pi, tech, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `value` : string, New tag value

Returns:

- nil,

SetSliceSshKey

Prototype:

SetSliceSshKey (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Slice objects using tag ssh_key

Allowed Roles:

admin, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `value` : string, New tag value

Returns:

- nil,

SetSliceVref

Prototype:

SetSliceVref (auth, id_or_name, value)

Description:

Accessor 'set' method designed for Slice objects using tag vref

Allowed Roles:

admin, pi, user, node

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `id_or_name` : int or string
  - int, Slice identifier
  - string, Slice name
- `value` : string, New tag value

Returns:

- nil,

UnBindObjectFromPeer

Prototype:

UnBindObjectFromPeer (auth, object_type, object_id, shortname)

Description:

This method is a hopefully temporary hack to let the sfa correctly detach the objects it creates from a remote peer object. This is needed so that the sfa federation link can work in parallel with RefreshPeer, as RefreshPeer depends on remote objects being correctly marked.

UnBindObjectFromPeer is allowed to admins only.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `object_type` : string, Object type, among 'site','person','slice','node','key'
- `object_id` : int, object_id
- `shortname` : string, peer shortname

Returns:

- int, 1 if successful

UpdateAddress

Prototype:

UpdateAddress (auth, address_id, address_fields)

Description:

Updates the parameters of an existing address with the values in address_fields.

PIs may only update addresses of their own sites.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *address_id* : int, Address identifier
- *address_fields* : struct
  - *city* : string, City
  - *country* : string, Country
  - *line3* : string, Address line 3
  - *line2* : string, Address line 2
  - *line1* : string, Address line 1
  - *state* : string, State or province
  - *postalcode* : string, Postal code

Returns:

- int, 1 if successful

UpdateAddressType

Prototype:

UpdateAddressType (auth, address_type_id_or_name, address_type_fields)

Description:

Updates the parameters of an existing address type with the values in address_type_fields.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *address_type_id_or_name* : int or string

- int, Address type identifier
- string, Address type

- *address_type_fields* : struct
  - *name* : string, Address type
  - *description* : string, Address type description

Returns:

- int, 1 if successful

UpdateConfFile

Prototype:

UpdateConfFile (auth, conf_file_id, conf_file_fields)

Description:

Updates a node configuration file. Only the fields specified in conf_file_fields are updated, all other fields are left untouched.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *conf_file_id* : int, Configuration file identifier
- *conf_file_fields* : struct
  - *file_owner* : string, chown(1) owner
  - *postinstall_cmd* : string, Shell command to execute after installing
  - *error_cmd* : string, Shell command to execute if any error occurs
  - *preinstall_cmd* : string, Shell command to execute prior to installing
  - *dest* : string, Absolute path where file should be installed
  - *ignore_cmd_errors* : boolean, Install file anyway even if an error occurs
  - *enabled* : boolean, Configuration file is active
  - *file_permissions* : string, chmod(1) permissions
  - *source* : string, Relative path on the boot server where file can be downloaded
  - *always_update* : boolean, Always attempt to install file even if unchanged
  - *file_group* : string, chgrp(1) owner

Returns:

- int, 1 if successful

UpdateIlink

Prototype:

UpdateIlink (auth, ilink_id, value)

Description:

Updates the value of an existing ilink

Access rights depend on the tag type.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech, user

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `ilink_id` : int, ilink identifier
- `value` : string, optional ilink value

Returns:

- int, 1 if successful

UpdateInitScript

Prototype:

UpdateInitScript (auth, initscript_id, initscript_fields)

Description:

Updates an initscript. Only the fields specified in initscript_fields are updated, all other fields are left untouched.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use
- `initscript_id` : int, Initscript identifier
- `initscript_fields` : struct
  - `enabled` : boolean, Initscript is active
  - `name` : string, Initscript name
  - `script` : string, Initscript

Returns:

- int, 1 if successful

UpdateInterface

Prototype:

UpdateInterface (auth, interface_id, interface_fields)

Description:

Updates an existing interface network. Any values specified in interface_fields are used, otherwise defaults are used. Acceptable values for method are dhcp and static. If type is static, then ip, gateway, network, broadcast, netmask, and dns1 must all be specified in interface_fields. If type is dhcp, these parameters, even if specified, are ignored.

PIs and techs may only update interfaces associated with their own nodes. Admins may update any interface network.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *interface_id* : int, Node interface identifier
- *interface_fields* : struct
  - *last_updated* : int, Date and time when node entry was created
  - *network* : string, Subnet address
  - *is_primary* : boolean, Is the primary interface for this node
  - *dns1* : string, IP address of primary DNS server
  - *hostname* : string, (Optional) Hostname
  - *mac* : string, MAC address
  - *interface_tag_ids* : array, List of interface settings
    - int

  - *bwlimit* : int, Bandwidth limit
  - *broadcast* : string, Network broadcast address
  - *method* : string, Addressing method (e.g., 'static' or 'dhcp')
  - *netmask* : string, Subnet mask
  - *dns2* : string, IP address of secondary DNS server
  - *ip* : string, IP address
  - *ifname* : string, accessor
  - *type* : string, Address type (e.g., 'ipv4')
  - *gateway* : string, IP address of primary gateway

Returns:

- int, 1 if successful

UpdateInterfaceTag

Prototype:

UpdateInterfaceTag (auth, interface_tag_id, value)

Description:

Updates the value of an existing interface setting

Admins have full access. Non-admins need to (1) have at least one of the roles attached to the tagtype, and (2) belong in the same site as the tagged subject.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech, user

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *interface_tag_id* : int, Interface setting identifier
- *value* : string, Interface setting value

Returns:

- int, 1 if successful

UpdateKey

Prototype:

UpdateKey (auth, key_id, key_fields)

Description:

Updates the parameters of an existing key with the values in key_fields.

Non-admins may only update their own keys.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech, user

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *key_id* : int, Key identifier
- *key_fields* : struct
  - *key_type* : string, Key type
  - *key* : string, Key value

Returns:

- int, 1 if successful

UpdateLeases

Prototype:

UpdateLeases (auth, lease_ids, input_fields)

Description:

Updates the parameters of a (set of) existing lease(s) with the values in lease_fields; specifically this applies to the timeslot definition. As a convenience you may, in addition to the t_from and t_until fields, you can also set the 'duration' field.

Users may only update leases attached to their slices. PIs may update any of the leases for slices at their sites, or any slices of which they are members. Admins may update any lease.

Returns a dict of successfully updated lease_ids and error messages.

Allowed Roles:

admin, pi, tech, user

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *lease_ids* : int or array of int
  - int, Lease identifier
  - array of int, Lease identifier

- *input_fields* : struct
  - *duration* : int, duration in seconds
  - *t_from* : int or string
    - int, timeslot start (unix timestamp)
    - string, timeslot start (formatted as %Y-%m-%d %H:%M:%S)

  - *t_until* : int or string
    - int, timeslot end (unix timestamp)
    - string, timeslot end (formatted as %Y-%m-%d %H:%M:%S)

Returns:

- struct, 'updated_ids' is the list ids updated, 'errors' is a list of error strings

UpdateMessage

Prototype:

UpdateMessage (auth, message_id, message_fields)

Description:

Updates the parameters of an existing message template with the values in message_fields.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *message_id* : string, Message identifier
- *message_fields* : struct
  - *enabled* : boolean, Message is enabled
  - *template* : string, Message template

Returns:

- int, 1 if successful

UpdateNode

Prototype:

UpdateNode (auth, node_id_or_hostname, node_fields)

Description:

Updates a node. Only the fields specified in node_fields are updated, all other fields are left untouched.

PIs and techs can update only the nodes at their sites. Only admins can update the key, session, and boot_nonce fields.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *node_id_or_hostname* : int or string
  - int, Node identifier
  - string, Fully qualified hostname

- *node_fields* : struct
  - *hrn* : string, accessor
  - *fcdistro* : string, accessor
  - *version* : string, Apparent Boot CD version
  - *slices* : array of int or string
    - int, Slice identifier
    - string, Slice name

  - *boot_state* : string, Boot state
  - *conf_files* : array of int, ConfFile identifier

- `interfaces` : array of int or struct
  - int, Interface identifier
  - struct, Attribute filter
    - `last_updated` : int or array of int
      - int, Date and time when node entry was created
      - array of int, Date and time when node entry was created
    - `network` : string or array of string
      - string, Subnet address
      - array of string, Subnet address
    - `is_primary` : boolean or array of boolean
      - boolean, Is the primary interface for this node
      - array of boolean, Is the primary interface for this node
    - `dns1` : string or array of string
      - string, IP address of primary DNS server
      - array of string, IP address of primary DNS server
    - `hostname` : string or array of string
      - string, (Optional) Hostname
      - array of string, (Optional) Hostname
    - `mac` : string or array of string
      - string, MAC address
      - array of string, MAC address
    - `interface_tag_ids` : array or array of array
      - array, List of interface settings
        - int
      - array of array, List of interface settings
        - int
    - `interface_id` : int or array of int
      - int, Node interface identifier
      - array of int, Node interface identifier
    - `broadcast` : string or array of string
      - string, Network broadcast address
      - array of string, Network broadcast address
    - `method` : string or array of string

    - string, Addressing method (e.g., 'static' or 'dhcp')
    - array of string, Addressing method (e.g., 'static' or 'dhcp')

- `netmask` : string or array of string
    - string, Subnet mask
    - array of string, Subnet mask

- `node_id` : int or array of int
    - int, Node associated with this interface
    - array of int, Node associated with this interface

- `dns2` : string or array of string
    - string, IP address of secondary DNS server
    - array of string, IP address of secondary DNS server

- `ip` : string or array of string
    - string, IP address
    - array of string, IP address

- `bwlimit` : int or array of int
    - int, Bandwidth limit
    - array of int, Bandwidth limit

- `type` : string or array of string
    - string, Address type (e.g., 'ipv4')
    - array of string, Address type (e.g., 'ipv4')

- `gateway` : string or array of string
    - string, IP address of primary gateway
    - array of string, IP address of primary gateway

- `hostname` : string, Fully qualified hostname
- `site_id` : int, Site at which this node is located
- `boot_nonce` : string, (Admin only) Random value generated by the node at last boot
- `node_type` : string, Node type
- `session` : string, (Admin only) Node session value
- `extensions` : string, accessor
- `pldistro` : string, accessor
- `key` : string, (Admin only) Node key
- `virt` : string, accessor
- `model` : string, Make and model of the actual machine
- `arch` : string, accessor

- *deployment* : string, accessor
- *slices_whitelist* : array of int or string
  - int, Slice identifier
  - string, Slice name

Returns:

- int, 1 if successful

UpdateNodeGroup

Prototype:

UpdateNodeGroup (auth, nodegroup_id_or_name, nodegroup_fields)

Description:

Updates a custom node group.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *nodegroup_id_or_name* : int or string
  - int, Node group identifier
  - string, Node group name
- *nodegroup_fields* : struct
  - *groupname* : string, Node group name
  - *value* : string, value that the nodegroup definition is based upon

Returns:

- int, 1 if successful

UpdateNodeTag

Prototype:

UpdateNodeTag (auth, node_tag_id, value)

Description:

Updates the value of an existing node tag

Admins have full access. Non-admins need to (1) have at least one of the roles attached to the tagtype, and (2) belong in the same site as the tagged subject.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech, user

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `node_tag_id` : int, Node tag identifier
- `value` : string, Node tag value

Returns:

- int, 1 if successful

UpdatePCU

Prototype:

UpdatePCU (auth, pcu_id, pcu_fields)

Description:

Updates the parameters of an existing PCU with the values in pcu_fields.
Non-admins may only update PCUs at their sites.
Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `pcu_id` : int, PCU identifier
- `pcu_fields` : struct
  - `username` : string, PCU username
  - `last_updated` : int, Date and time when node entry was created
  - `node_ids` : array, List of nodes that this PCU controls
    - int

  - `ip` : string, PCU IP address
  - `notes` : string, Miscellaneous notes
  - `hostname` : string, PCU hostname
  - `protocol` : string, PCU protocol, e.g. ssh, https, telnet
  - `model` : string, PCU model string
  - `password` : string, PCU username
  - `ports` : array, List of the port numbers that each node is connected to
    - int

Returns:

- int, 1 if successful

UpdatePCUProtocolType

Prototype:

UpdatePCUProtocolType (auth, protocol_type_id, protocol_type_fields)

Description:

Updates a pcu protocol type. Only the fields specified in port_typee_fields are updated, all other fields are left untouched.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *protocol_type_id* : int, PCU protocol type identifier
- *protocol_type_fields* : struct
  - *supported* : boolean, Is the port/protocol supported by PLC
  - *protocol* : string, Protocol
  - *port* : int, PCU port
  - *pcu_type_id* : int, PCU type identifier

Returns:

- int, 1 if successful

UpdatePCUType

Prototype:

UpdatePCUType (auth, pcu_type_id, pcu_type_fields)

Description:

Updates a PCU type. Only the fields specified in pcu_typee_fields are updated, all other fields are left untouched.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *pcu_type_id* : int, PCU Type Identifier

- *pcu_type_fields* : struct
  - *model* : string, PCU model
  - *name* : string, PCU full name

Returns:

- int, 1 if successful

UpdatePeer

Prototype:

UpdatePeer (auth, peer_id_or_name, peer_fields)

Description:

Updates a peer. Only the fields specified in peer_fields are updated, all other fields are left untouched.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *peer_id_or_name* : int or string
  - int, Peer identifier
  - string, Peer name

- *peer_fields* : struct
  - *peername* : string, Peer name
  - *peer_url* : string, Peer API URL
  - *key* : string, Peer GPG public key
  - *hrn_root* : string, Root of this peer in a hierarchical naming space
  - *cacert* : string, Peer SSL public certificate
  - *shortname* : string, Peer short name

Returns:

- int, 1 if successful

UpdatePerson

Prototype:

UpdatePerson (auth, person_id_or_email, person_fields)

Description:

Updates a person. Only the fields specified in person_fields are updated, all other fields are left untouched.

Users and techs can only update themselves. PIs can only update themselves and other non-PIs at their sites.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, user, tech

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `person_id_or_email` : int or string
  - int, User identifier
  - string, Primary e-mail address

- `person_fields` : struct
  - `bio` : string, Biography
  - `first_name` : string, Given name
  - `last_name` : string, Surname
  - `keys` : array of int or struct
    - int, Key identifier
    - struct, Attribute filter
      - `peer_key_id` : int or array of int
        - int, Foreign key identifier at peer
        - array of int, Foreign key identifier at peer

      - `key_type` : string or array of string
        - string, Key type
        - array of string, Key type

      - `key` : string or array of string
        - string, Key value
        - array of string, Key value

      - `person_id` : int or array of int
        - int, User to which this key belongs
        - array of int, User to which this key belongs

      - `key_id` : int or array of int
        - int, Key identifier
        - array of int, Key identifier

      - `peer_id` : int or array of int
        - int, Peer to which this key belongs

- array of int, Peer to which this key belongs

- `roles` : array of int or string
  - int, Role identifier
  - string, Role name

- `title` : string, Title
- `url` : string, Home page
- `slices` : array of int or string
  - int, Slice identifier
  - string, Slice name

- `enabled` : boolean, Has been enabled
- `sites` : array of int or string
  - int, Site identifier
  - string, Site name

- `phone` : string, Telephone number
- `hrn` : string, accessor
- `sfa_created` : string, accessor
- `showconf` : string, accessor
- `columnconf` : string, accessor
- `password` : string, Account password in crypt() form
- `email` : string, Primary e-mail address
- `advanced` : string, accessor

Returns:

- int, 1 if successful

UpdatePersonTag

Prototype:

UpdatePersonTag (auth, person_tag_id, value)

Description:

Updates the value of an existing person setting
Admins have full access. Non-admins can change their own tags.
Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech, user

Parameters:

- `auth` : struct, API authentication structure

- `AuthMethod` : string, Authentication method to use

- `person_tag_id` : int, Person setting identifier
- `value` : string, Person setting value

Returns:

- int, 1 if successful

UpdateSite

Prototype:

UpdateSite (auth, site_id_or_login_base, site_fields)

Description:

Updates a site. Only the fields specified in update_fields are updated, all other fields are left untouched.

PIs can only update sites they are a member of. Only admins can update max_slices, max_slivers, and login_base.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi

Parameters:

- `auth` : struct, API authentication structure
  - `AuthMethod` : string, Authentication method to use

- `site_id_or_login_base` : int or string
  - int, Site identifier
  - string, Site slice prefix

- `site_fields` : struct
  - `name` : string, Full site name
  - `persons` : array of int or string
    - int, Person identifier
    - string, Email address

  - `url` : string, URL of a page that describes the site
  - `enabled` : boolean, Has been enabled
  - `longitude` : double, Decimal longitude of the site
  - `max_slivers` : int, Maximum number of slivers that the site is able to create
  - `max_slices` : int, Maximum number of slices that the site is able to create
  - `login_base` : string, Site slice prefix
  - `ext_consortium_id` : int, external consortium id
  - `latitude` : double, Decimal latitude of the site
  - `is_public` : boolean, Publicly viewable site
  - `abbreviated_name` : string, Abbreviated site name

- `addresses` : array of int or struct
  - int, Address identifer
  - struct, Attribute filter
    - `city` : string or array of string
      - string, City
      - array of string, City
    - `address_id` : int or array of int
      - int, Address identifier
      - array of int, Address identifier
    - `country` : string or array of string
      - string, Country
      - array of string, Country
    - `line3` : string or array of string
      - string, Address line 3
      - array of string, Address line 3
    - `line2` : string or array of string
      - string, Address line 2
      - array of string, Address line 2
    - `line1` : string or array of string
      - string, Address line 1
      - array of string, Address line 1
    - `address_type_ids` : array or array of array
      - array, Address type identifiers
        - int
      - array of array, Address type identifiers
        - int
    - `state` : string or array of string
      - string, State or province
      - array of string, State or province
    - `postalcode` : string or array of string
      - string, Postal code
      - array of string, Postal code
    - `address_types` : array or array of array

- array, Address types
  - string

- array of array, Address types
  - string

Returns:

- int, 1 if successful

UpdateSiteTag

Prototype:

UpdateSiteTag (auth, site_tag_id, value)

Description:

Updates the value of an existing site setting

Admins have full access. Non-admins need to (1) have at least one of the roles attached to the tagtype, and (2) belong in the same site as the tagged subject.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, tech, user

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *site_tag_id* : int, Site setting identifier
- *value* : string, Site setting value

Returns:

- int, 1 if successful

UpdateSlice

Prototype:

UpdateSlice (auth, slice_id_or_name, slice_fields)

Description:

Updates the parameters of an existing slice with the values in slice_fields.

Users may only update slices of which they are members. PIs may update any of the slices at their sites, or any slices of which they are members. Admins may update any slice.

Only PIs and admins may update max_nodes. Slices cannot be renewed (by updating the expires parameter) more than 8 weeks into the future.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, user

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use

- *slice_id_or_name* : int or string
  - int, Slice identifier
  - string, Slice name

- *slice_fields* : struct
  - *enable_hmac* : string, accessor
  - *description* : string, Slice description
  - *initscript* : string, accessor
  - *expires* : int, Date and time when slice expires, in seconds since UNIX epoch
  - *ipv6_address* : string, accessor
  - *persons* : array of int or string
    - int, Person identifier
    - string, Email address

  - *pldistro* : string, accessor
  - *arch* : string, accessor
  - *vref* : string, accessor
  - *hrn* : string, accessor
  - *instantiation* : string, Slice instantiation state
  - *fcdistro* : string, accessor
  - *url* : string, URL further describing this slice
  - *max_nodes* : int, Maximum number of nodes that can be assigned to this slice
  - *initscript_code* : string, accessor
  - *omf_control* : string, accessor
  - *nodes* : array of int or string
    - int, Node identifier
    - string, Fully qualified hostname

  - *sfa_created* : string, accessor

Returns:

- int, 1 if successful

UpdateSliceTag

Prototype:

UpdateSliceTag (auth, slice_tag_id, value)

Description:

Updates the value of an existing slice or sliver attribute.

Users may only update attributes of slices or slivers of which they are members. PIs may only update attributes of slices or slivers at their sites, or of which they are members. Admins may update attributes of any slice or sliver.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin, pi, user, node

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *slice_tag_id* : int, Slice tag identifier
- *value* : string or string
  - string, Slice attribute value
  - string, Initscript name

Returns:

- int, 1 if successful

UpdateTagType

Prototype:

UpdateTagType (auth, tag_type_id_or_name, tag_type_fields)

Description:

Updates the parameters of an existing tag type with the values in tag_type_fields.

Returns 1 if successful, faults otherwise.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *tag_type_id_or_name* : int or string
  - int, Node tag type identifier
  - string, Node tag type name
- *tag_type_fields* : struct
  - *category* : string, Node tag category

- *description* : string, Node tag type description
- *tagname* : string, Node tag type name

Returns:

- int, 1 if successful

VerifyPerson

Prototype:

VerifyPerson (auth, person_id_or_email, verification_key, verification_expires)

Description:

Verify a new (must be disabled) user's e-mail address and registration.

If verification_key is not specified, then a new verification_key will be generated and stored with the user's account. The key will be e-mailed to the user in the form of a link to a web page.

The web page should verify the key by calling this function again and specifying verification_key. If the key matches what has been stored in the user's account, then an e-mail will be sent to the user's PI (and support if the user is requesting a PI role), asking the PI (or support) to enable the account.

Returns 1 if the verification key if valid.

Allowed Roles:

admin

Parameters:

- *auth* : struct, API authentication structure
  - *AuthMethod* : string, Authentication method to use
- *person_id_or_email* : int or string
  - int, User identifier
  - string, Primary e-mail address
- *verification_key* : string, Reset password key
- *verification_expires* : int, Date and time when verification_key expires

Returns:

- int, 1 if verification_key is valid

system.listMethods

Prototype:

system.listMethods ()

Description:

This method lists all the methods that the XML-RPC server knows how to dispatch.

Allowed Roles:

Parameters:

- None

Returns:

- array, List of methods

system.methodHelp

Prototype:

system.methodHelp (method)

Description:

Returns help text if defined for the method passed, otherwise returns an empty string.

Allowed Roles:

Parameters:

- `method` : string, Method name

Returns:

- string, Method help

system.methodSignature

Prototype:

system.methodSignature (method)

Description:

Returns an array of known signatures (an array of arrays) for the method name passed. If no signatures are known, returns a none-array (test for type != array to detect missing signature).

Allowed Roles:

Parameters:

- `method` : string, Method name

Returns:

- array of array, Method signature
  - string

system.multicall

Prototype:

system.multicall (calls)

Description:

Process an array of calls, and return an array of results. Calls should be structs of the form

{'methodName': string, 'params': array}

Each result will either be a single-item array containg the result value, or a struct of the form

{'faultCode': int, 'faultString': string}

This is useful when you need to make lots of small calls without lots of round trips.

Allowed Roles:

Parameters:

- `calls` : array of struct
  - `params` : array, Method arguments
  - `methodName` : string, Method name

Returns:

- array of mixed or struct
  - array of mixed
  - struct
    - `faultCode` : int, XML-RPC fault code
    - `faultString` : int, XML-RPC fault detail