VoiSentry Administration via REST

API Guide

Revision 1.3

PROPRIETARY INFORMATION

Document Revision

Rev	Date	By	Detail
1.1	19/03/18	ac	Initial for release
1.2	25/05/18	ebj	Reformatted
1.3	20/09/18	ac/ebj	Removed superfluous commas

CONTENTS

1 INTRODUCTION

Most of the administrative functions that are available via the HTML (web) User Interface are also achievable via Web Services (REST) calls. This enables administrative functions to be integrated into third-party applications The REST interface is accessible at the following URI, where $TARGET is the IP address of the Node, and <REST_function_invocation> represents the function name and query string of the REST function to invoke:

```
https://$TARGET:/ws/<REST_function_invocation>
```

All functions are shown below, using the ubiquitous command-line program ‘curl’, which is available for most OSs. As administrative tasks are notionally being undertaken on behalf of a particular administrator or tenant, along with every REST call you must provide a valid Username and Password that is appropriate to the request being made.

2 SUMMARY AND SERVICE NAME

- ping - Ping the system, get some system information.
- node_reset - Reboot or powerdown Nodes.
- node_ip - Get/Set IP Address information.
- node_decluster - Remove a node from the cluster.
- node_forceout - Force out and unresponsive node from the cluster.
- cluster_create - Create a cluster.
- cluster_join - Join an existing cluster.
- node_data - List the Data of all or specific nodes.
- node_status - List the status of all or a specific node.
- account_list - List all or a specific account.
- account_create - Create an account.
- account_edit - Edit an existing account.
- account_delete - Delete an account.
- dataset_list - List all or specific dataset.
- dataset_create - Create a dataset
- dataset_delete - Delete a dataset
- accesskey_list - List all or a specific accesskey.
- accesskey_create - Create an accesskey.
- accesskey_edit - Edit an existing accesskey.
- accesskey_delete - Delete an accesskey.
- ssh_logins - Enable / disable the ability to login via SSH.

3 API

3.1 ping

Ping the system, get some system information.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.

Usage example:

```
curl -k --user superuser:password https://$TARGET:/ws/ping
```

Returns:

```
{
  'clustername'   : "cluster1",     # A string, the name of the
cluster
  'serialno'      : "1987123491",   # A string, the serial number of
the node
  'nodestatus'    : 'A'             # 'A' active, 'B' blocked (Node
may be going out of service)
}
```

3.2 node_reset

Reboot this node / Reboot that node / Power off this node / Power off that node.

Reboot this node Restarts the node you are logged in to.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** node_reset

Usage example:

```
curl -k --user superuser:password -d ''
https://$TARGET:/ws/node_reset
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result'   : ""       # A String containing a message
}
```

Reboot that node Restarts the specified node.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** node_reset
- **node** Serial number of node you wish to restart.

Usage example:

```
curl -k --user superuser:password -d ''
https://$TARGET:/ws/node_reset?node=serno
```

Returns:

```
{
  'status'   : 200,     # An integer, a status code (See Status
Codes for more Details)
  'result'   : ""       # A String containing a message
}
```

Powerdown this node Shuts down the node you are logged into

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** node_reset

Usage example:

```
curl -k --user superuser:password -d ''
https://$TARGET:/ws/node_reset?poweroff
```

Returns:

```
{
  'status'   : 200,     # An integer, a status code (See Status
Codes for more Details)
  'result'   : ""       # A String containing a message
}
```

Powerdown that node Shuts down the specified node.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** node_reset
- **node** Serial number of node you wish to restart.

Usage example:

```
curl -k --user superuser:password -d ''
https://$TARGET:/ws/node_reset?poweroff\&node=serno
```

Returns:

```
{
  'status'   : 200,     # An integer, a status code (See Status
Codes for more Details)
  'result'   : ""       # A String containing a message
}
```

3.3 node_ip

Get/Set this IP / Get/Set that IP.

Get/Set this IP get/set (set only when not clustered) this node IP address.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** node_ip

Usage example:

```
curl -k --user superuser:password https://$TARGET:/ws/node_ip
```

Returns:

```
{
  'status'      : 200,                                  # An integer
containing a status code (See Status Codes for more Details)
  'result': {
    'domain'      : "aculab.com",                           # A
String, the domain name
    'ip4_gateway' : "10.202.100.254",                       # A
String, the IP4 Gateway
    'ip6'         : "fb9f:ab1c:a25f:0:9c8:f048:8f28:fe24",  # A
String, the IP6 IP Address
    'ip4'         : "10.101.110.200/211.211.0.0",           # A
String, the IP4 IP Address
    'nameservers' : "10.202.167.207 10.202.167.208",        # A
string, a list of nameservers
    'ip6_gateway' : "fb9f:ab1c:a25f:0:9c8:f048:8f28:fe24"   # A
String, the IP6 Gateway
  }
}
```

get/set that IP get/set (set only when not clustered) that node IP address.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** node_ip
- **node** Serial number of node you wish to get/set.

Usage example:

```
curl -k --user superuser:password
https://$TARGET:/ws/node_ip?node=serno
```

Returns:

```
{
  'status'      : 200,                                  # An integer
containing a status code (See Status Codes for more Details)
  'result': {
    'domain'      : "aculab.com",                       # A
String, the domain name
    'ip4_gateway' : "10.202.100.254",                   # A
String, the IP4 Gateway
    'ip6'         : "fb9f:ab1c:a25f:0:9c8:f048:8f28:fe24",  # A
String, the IP6 IP Address
    'ip4'         : "10.101.110.200/211.211.0.0",       # A
String, the IP4 IP Address
    'nameservers' : "10.202.167.207 10.202.167.208",    # A
string, a list of nameservers
    'ip6_gateway' : "fb9f:ab1c:a25f:0:9c8:f048:8f28:fe24"   # A
String, the IP6 Gateway
  }
}
```

3.4 node_decluster

Decluster this node / Decluster that node.

Decluster this node Decluster the node you are logged into.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** node_decluster

Usage example:

```
curl -k --user superuser:password -d ''
https://$TARGET:/ws/node_decluster
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result'   : ""       # A String containing a message
}
```

Decluster that node Decluster the specified node.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** node_decluster
- **node** Serial number of node you wish to decluster.

Usage example:

```
curl -k --user superuser:password -d ''
https://$TARGET:/ws/node_decluster?node=serno
```

Returns:

```
{
  'status'   : 200,     # An integer, a status code (See Status
Codes for more Details)
  'result'   : ""       # A String containing a message
}
```

3.5 node_forceout

Forces an unavailable node out of the cluster.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** node_forceout
- **node** Serial number of node you wish to force-out.

Usage example:

```
curl -k --user superuser:password -d ''
https://$TARGET:/ws/node_forceout?node=serno
```

Returns:

```
{
  'status'   : 200,     # An integer, a status code (See Status
Codes for more Details)
  'result'   : ""       # A String containing a message
}
```

3.6 cluster_create

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** cluster_create
- **name** Name of cluster you wish to create.
- **pass** Password for cluster you wish to create.

Usage example:

```
curl -k --user superuser:password -d ''
https://$TARGET:/ws/cluster_create?name=cluster_name\&pass=cluster_p
ass
```

Returns:

```
{
  'status'   : 200,     # An integer, a status code (See Status
Codes for more Details)
  'result'   : ""       # A String containing a message
}
```

3.7 cluster_join

Joins this node to an established Cluster.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** cluster_join
- **name** Name of cluster you wish to join.
- **pass** Password for cluster you wish to join.
- **IP address** IP address of a node in the Cluster you wish to join.

Usage example:

```
curl -k --user superuser:password -d ''
https://$TARGET:/ws/cluster_join?name=cluster_name\&pass=cluster_pas
s\&ip=<ip_address>
```

Returns:

```
{
  'status'   : 200,     # An integer, a status code (See Status
Codes for more Details)
  'result'   : ""       # A String containing a message
}
```

3.8 cluster_quota

Returns Cluster wide quota All / Tenant.

All Returns All Cluster wide quota.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** cluster_quota

Usage example:

```
curl -k --user superuser:password https://$TARGET:/ws/cluster_quota
```

Returns:

```
{
  'status'      : 200,     # An integer containing a status code
(See Status Codes for more Details)
  'result'      : {
    'cluster'       :  { # Cluster
        'verifs'      : 10,         # An Integer, Number of
verifications
        'maxenrols'   : 10000,      # An Integer, Maximum number of
enrolments
        'enrols'      : 20,         # An Integer, Number of
enrolments
        'maxverifs'   : 2000        # An Integer, Maximum number of
verifications
    },
    'Tenants'       :  { # tenants
        "Tenant1"       :  { # tenant
            'verifs'      : 10,     # An Integer, Number of
verifications
            'maxenrols'   : 10000,  # An Integer, Maximum number of
enrolments
            'enrols'      : 20,     # An Integer, Number of
enrolments
            'maxverifs'   : 2000    # An Integer, Maximum number of
verifications
        },
        "Tenant2"       :  { # Additional tenants
        }
    }
  }
}
```

Tenant Returns Tenant quota only.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** cluster_quota
- **Tenant** Name of tenant whose quota you wish to return.

Usage example:

```
curl -k --user superuser:password
https://$TARGET:/ws/cluster_quota?tenant=tenant       # that tenant
quota only
curl -k --user tenant:password
https://$TARGET:/ws/cluster_quota?tenant=tenant      # that tenant
quota only
```

Returns:

```
{
  'status'      : 200,     # An integer containing a status code
(See Status Codes for more Details)
  'result'      : {
    "Tenant1"       :  { # tenant
        'verifs'      : 10,     # An Integer, Number of
verifications
        'maxenrols'   : 10000,  # An Integer, Maximum number of
enrolments
        'enrols'      : 20,     # An Integer, Number of enrolments
        'maxverifs'   : 2000    # An Integer, Maximum number of
verifications
    }
  }
}
```

3.9 node_data

List Node Data all/available/active / List That Node Data all/available/active.

All Returns Node Data all/available/active.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** node_data

Usage example:

```
curl -k --user superuser:password https://$TARGET:/ws/node_data
# list of all nodes (and node data)
curl -k --user superuser:password https://$TARGET:/ws/node_data?all
# list of all nodes (and node data)
curl -k --user superuser:password
https://$TARGET:/ws/node_data?available     # list of available node
curl -k --user superuser:password
https://$TARGET:/ws/node_data?active        # list of active nodes
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result': {
    "1505726377": { # Node Serial number
        'status'         : "B",
# A String,     'A' Active, 'B' Blocked
        'dbvol'          : 29.0,
# A Float,      database volume size
        'dbfree'         : 20.1,
# A Float,      database volume free space
        'id'             : "1005126101",
# A String,     serial number of node
        'enrol_ave'      : 0.0,
# A Float,      short-term average enrolment time in seconds
        'verif_pm'       : 0,
# An Integer,   verifications per minute
        'enrol_pm'       : 0,
# An Integer,   enrolments per minute
        'available'      : "A",
# A String,     'A' available (visible) to cluster, else 'U'
unavailable
        'ip6'            : "fb9f:ab1c:a25f:0:9c8:f048:8f28:fe24",
# A String,     IP6 IP address of Node
        'ip4'            : "10.101.111.211",
# A String,     IP4 IP address of Node
        'verif_ave'      : 0.0,
# A Float,      short-term average verification time in seconds
        'verif_max'      : 0.0,
# A Float,      short-term maximum verification time
        'enrol_max'      : 0.0,
# A Float,      short-term maximum enrolment time
        'cluster_busy'   : false,
# true/false,   a Node is busy with maintenance activity
        'clustering'     : false
# true/false,   this Node is synchronising data on joining the
cluster
        },
    "1505726378": { # Additional node serial numbers
        }
    }
}
```

That node Returns Node Data all/available/active.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** node_data
- **Node** Serial number of node you wish to list.

Usage example:

```
curl -k --user superuser:password
https://$TARGET:/ws/node_data?node=serno
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result': {
    'status'         : "B",                                    # A
String,     'A' Active, 'B' Blocked
    'dbvol'          : 29,                                     # A
Float,      database volume size in GiB
    'dbfree'         : 20.1,                                   # A
Float,      database volume free space
    'id'             : "1005126101",                           # A
String,     serial number of node
    'enrol_ave'      : 0.0,                                    # A
Float,      short-term average enrolment time in seconds
    'verif_pm'       : 0,                                      # An
Integer,   verifications per minute
    'enrol_pm'       : 0,                                      # An
Integer,   containinig enrolments per minute
    'available'      : "A",                                    # A
string,     'A' available (visible) to cluster, else 'U' unavailable
    'ip6'            : "fb9f:ab1c:a25f:0:9c8:f048:8f28:fe24",    # A
String,     IP6 IP address of Node
    'ip4'            : "10.101.111.211",                       # A
String,     IP4 IP address of Node
    'verif_ave'      : 0.0,                                    # A
Float,      short-term average verification time in seconds
    'verif_max'      : 0.0,                                    # A
Float,      short-term maximum verification time
    'enrol_max'      : 0.0,                                    # A
Float,      short-term maximum enrolment time
    'cluster_busy'   : false,                                  #
true/false,   a Node is busy with maintenance activity
    'clustering'     : false                                   #
true/false,   this Node is synchronising data on joining the cluster
  }
}
```

3.10 node_status

List This Node Status / List That Node Status / Set node to active / Set node to block.

This Node Returns Status of the node you are logged into.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** node_status

Usage example:

```
curl -k --user superuser:password https://$TARGET:/ws/node_status
```

Returns:

```
{
    'status'    : 200,      # An integer containing a status code
(See Status Codes for more Details)
    "result"    : {
        "1000000001": {     # Serial number of node
            'available'     : "A",          # A string, Availability
of node A=available U=unavailable
            'status'        : "B",          # A string, 'A' Active,
'B' Blocked
            'cluster_busy'  : false,        # true/false,   a Node
is busy with maintenance activity
            'clustering'    : false,        # true/false,   this
Node is synchronising data on joining the cluster
            'ip4'           : "1.2.3.4",    # IPv4 address of node
            'ip6'           : ""            # IPv6 address of node
(if one is assigned)
        },
        "1000000002": {     # Additional nodes
        }
    }
}
```

That node Returns Status of a specific node.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** node_status
- **node** Serial number of node you wish to show status.

Usage example:

```
curl -k --user superuser:password
https://$TARGET:/ws/node_status?node=serno
```

Returns:

```
{
    'status'   : 200,   # An integer containing a status code (See
Status Codes for more Details)
    'result'    {
        'available'     : "A"       # A String, Availability of node
A=available U=unavailable
        'status'        : "B"       # A string, 'A' Active, 'B'
Blocked
        'cluster_busy'  : false,    # True/False,   a Node is busy
with maintenance activity
        'clustering'    : false,    # True/False,   this Node is
synchronising data on joining the cluster
        'ip4'           : "1.2.3.4",    # IPv4 address of node
        'ip6'           : ""            # IPv6 address of node (if
```

```
one is assigned)
     }
}
```

node_status?active/blocked Sets this Node to Active or Blocked.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** node_status

Usage example:

```
curl -k --user superuser:password -d ''
https://$TARGET:/ws/node_status?active                # set node
status to active
curl -k --user superuser:password -d ''
https://$TARGET:/ws/node_status?block                 # set node
status to block
```

Returns:

```
{
    'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
    'result'   : ""       # A String containing a message
}
```

active/blocked Sets this Node to Active or Blocked.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** node_status
- **node** Serial number of node you wish to show status.

Usage example:

```
curl -k --user superuser:password -d ''
https://$TARGET:/ws/node_status?node=serno\&active   # set that node
status to active
curl -k --user superuser:password -d ''
https://$TARGET:/ws/node_status?node=serno\&block    # set that
status to block
```

Returns:

```
{
    'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
    'result'   : ""      # A String containing a message
}
```

3.11 account_list

List all accounts / list a specific account / list of current user account.

All Returns a list of All accounts.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** account_list

Usage example:

```
curl -k --user superuser:password https://$TARGET:/ws/account_list
# list of all accounts
curl -k --user tenant:password    https://$TARGET:/ws/account_list
# list of all accounts (including all users) for this tenant
curl -k --user user:password      https://$TARGET:/ws/account_list
# listing of current user account
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result'   : {
        "admin1": {     # A string containing the name if the
account
            'username'              : "admin1",     # A String,
The name of account being listed
            'created'               : "1506007135", # A String,
The date the account was created
            'logins'                : 0,            # An Integer,
Number of times the account has used their login
            'creator'               : "superuser",  # A String,
The account used to create this account
            'quota_enrolments'      : 100,          # An Integer,
Maximum (quota) total number of enrolments
            'active'                : "T",          # A String,
Account is enabled = T or Disabled = F
            'userlevel'             : 2,            # An Integer,
Account level - Superuser=3 Admin=2 tenant=1 user=0
            'accessed'              : "1506007135", # A String,
The date the account was last accessed
            'quota_verifications'   : 0             # An Integer,
Maximum (quota) of verifications
        },
        "admin2": {     # Additional Accounts
        }
    }
}
```

Tenants Returns a list of all tenants accounts.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** account_list
- **Tenant** Name of tenant whose accounts you wish to return.

Usage example:

```
curl -k --user superuser:password
https://$TARGET:/ws/account_list?tenant=account
```

Returns:

```
Returns:
{
   'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
   'result'   : {
         "user1_2": {     # A string containing the name if the
account
             'username'              : "user1_2",    # A string,
The name of account being listed
             'created'               : "1506007135", # A string,
The date the account was created
             'logins'                : 0,            # An Integer,
Number of times the account has used their login
             'creator'               : "tenant1",    # A string,
The account used to create this account
             'quota_enrolments'      : 100,          # An Integer,
Maximum (quota) total number of enrolments
             'active'                : "T",          # A string,
Account is enabled = T or Disabled = F
             'userlevel'             : 0,            # An Integer,
Account level - Superuser=3 Admin=2 tenant=1 user=0
             'accessed'              : "1506007135", # A string,
The date the account was last accessed
             'quota_verifications'   : 0             # An Integer,
Maximum (quota) of verifications
          },
          "User1_1": {     # Additional Accounts
          }
     }
 }
```

Account Returns a list of a specific account.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** account_list
- **account** Name of account you wish to return.

Usage example:

```
curl -k --user superuser:password
https://$TARGET:/ws/account_list?account=account  # listing of this
specific account
curl -k --user tenant:password
https://$TARGET:/ws/account_list?account=account  # listing of this
account, provided it belongs to this tenant
```

Returns:

```
{
   'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
   'result'   : {
         "tenant1": {     # A string containing the name if the
account
             'username'              : "tenant1",    # A string,
The name of account being listed
             'created'               : "1506007135", # A string,
The date the account was created
             'logins'                : 5,            # An Integer,
Number of times the account has used their login
             'creator'               : "tenant1",    # A string,
The account used to create this account
             'quota_enrolments'      : 100,          # An Integer,
Maximum (quota) total number of enrolments
             'active'                : "T",          # A string,
Account is enabled = T or Disabled = F
             'userlevel'             : 1,            # An Integer,
Account level - Superuser=3 Admin=2 tenant=1 user=0
             'accessed'              : "1506007135", # A string,
The date the account was last accessed
             'quota_verifications'   : 0             # An Integer,
Maximum (quota) of verifications
          }
     }
 }
```

3.12 account_create

Create Admin/Tenant/User accounts.

admin Create Admin Accounts.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** account_create
- **account** Name of account you are creating.
- **type** Type of account admin, tenant or user.
- **userpassword** Password for the account being created.

Usage example:

```
curl -k -d '' --user superuser:password
https://$TARGET:/ws/account_create?account=account_name\&type=admin\
&userpassword=password
```

Returns:

```
{
    'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
    'result' {
        'username'            : "Admin1",       # A string,     The
name of account being listed
        'active'              : "T",            # A string,
Account is enabled = T or Disabled = F
        'userlevel'           : 2,              # An Integer,
Account level - superuser=3 admin=2 tenant=1 user=0
        'creator'             : "superuser",    # A string,     The
account used to create this account
        'logins'              : 0,              # An Integer,
Number of times the account has used their login
        'accessed'            : 1106100360,     # An Integer,   The
date the account was last accessed
        'created'             : 1106100360      # An Integer,   The
date the account was created
    }
}
```

tenant Create Tenant Accounts.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** account_create
- **account** Name of account you are creating.
- **type** Type of account admin, tenant or user.
- **userpassword** Password for the account being created.
- **maxenrols** Maximum (quota) total number of enrolments.
- **maxverifs** Password for the account being created.

Usage example:

```
curl -k -d '' --user superuser:password
https://$TARGET:/ws/account_create?account=account_name\&type=tenant
\&userpassword=password\&maxenrols=2000\&maxverifs=20000
curl -k -d '' --user admin:password
https://$TARGET:/ws/account_create?account=account_name\&type=tenant
\&userpassword=password\&maxenrols=2000\&maxverifs=20000
```

Returns:

```
{
    'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
    'result' {
        'username'            : "Tenant1",      # A string,     The
name of account being listed
        'active'              : "T",            # A string,
Account is enabled = T or Disabled = F
        'userlevel'           : 1,              # An Integer,
Account level - superuser=3 admin=2 tenant=1 user=0
        'creator'             : "superuser",    # A string,     The
account used to create this account
        'logins'              : 0,              # An Integer,
Number of times the account has used their login
        'accessed'            : 1106100360,     # An Integer,   The
date the account was last accessed
        'created'             : 1106100360,     # An Integer,   The
date the account was created
        'quota_enrolments'    : 0,              # An Integer,
Maximum (quota) total number of enrolments (0 means 'unlimited')
        'quota_verifications' : 0               # An Integer,
Maximum (quota) of verifications (0 means 'unlimited')
    }
}
```

user Create User Accounts (Only Tenants can create Users)

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** account_create
- **account** Name of account you are creating.
- **type** Type of account admin, tenant or user.
- **userpassword** Password for the account being created.

Usage example:

```
curl -k -d '' --user tenant:password
https://$TARGET:/ws/account_create?account=account_name\&type=user\&
userpassword=password
```

Returns:

```
{
    'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
    'result' {
        'username'            : "user1",        # A string,     The
name of account being listed
        'active'              : "T",            # A string,
Account is enabled = T or Disabled = F
        'userlevel'           : 0,              # An Integer,
Account level - superuser=3 admin=2 tenant=1 user=0
        'creator'             : "tenant1",      # A string,     The
account used to create this account
        'logins'              : 0,              # An Integer,
Number of times the account has used their login
        'accessed'            : 1106100360,     # An Integer,   The
date the account was last accessed
        'created'             : 1106100360      # An Integer,   The
date the account was created
    }
}
```

3.13 account_edit

Enable/Disable Enable Disable Accounts.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** account_edit
- **account** Name of account you are editing.
- **enable** enable account=T Disable Account=F.

Usage example:

```
curl -k -d '' --user superuser:password
https://$TARGET:/ws/account_edit?account=accountname\&enable=T
curl -k -d '' --user tenant:password
https://$TARGET:/ws/account_edit?account=username\&enable=F
```

Returns:

```
{
     'status'   : 200,     # An integer containing a status code
(See Status Codes for more Details)
     'result' {
         "user": { # account type
             'username'            : "user1",        # A string,
The name of account being listed
             'active'              : "T",            # A string,
Account is enabled = T or Disabled = F
             'userlevel'           : 0,              # An Integer,
Account level - superuser=3 admin=2 tenant=1 user=0
             'creator'             : "tenant1",      # A string,
The account used to create this account
             'logins'              : 0,              # An Integer,
Number of times the account has used their login
             'accessed'            : 1106100360,     # An Integer,
The date the account was last accessed
             'created'             : 1106100360,     # An Integer,
The date the account was created
             'quota_enrolments'    : 0,              # An Integer,
Maximum (quota) total number of enrolments
             'quota_verifications' : 0               # An Integer,
Maximum (quota) of verifications
         }
     }
 }
```

Tenant Edit a Tenants Account (must be Superuser or Admin).

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** account_edit
- **account** Name of account you are editing.
- **maxenrols** Maximum (quota) total number of enrolments.
- **maxverifs** Maximum (quota) total number of verifications.
- **enable** enable account=T Disable Account=F.

Usage example:

```
curl -k -d '' --user superuser:password
https://$TARGET:/ws/account_edit?account=tenant\&maxenrols=2000\&max
verifs=20000\&enable=T
curl -k -d '' --user admin:password
https://$TARGET:/ws/account_edit?account=tenant\&maxenrols=1000\&max
verifs=10000\&enable=F
```

Returns:

```
          {
    'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
    'result' {
        "tenant": { # account type
            'username'            : "tenant1",      # A string,
The name of account being listed
            'active'              : "T",            # A string,
Account is enabled = T or Disabled = F
            'userlevel'           : 0,              # An Integer,
Account level - superuser=3 admin=2 tenant=1 user=0
            'creator'             : "admin1",       # A string,
The account used to create this account
            'logins'              : 0,              # An Integer,
Number of times the account has used their login
            'accessed'            : 1106100360,     # An Integer,
The date the account was last accessed
            'created'             : 1106100360,     # An Integer,
The date the account was created
            'quota_enrolments'    : 0,              # An Integer,
Maximum (quota) total number of enrolments
            'quota_verifications' : 0               # An Integer,
Maximum (quota) of verifications
        }
    }
}
```

3.14 account_delete

Delete Account / Force Delete Account.

Delete an Account (Tenants will not normally be deleted if they have access keys or Datasets, these must be deleted first. Use force to override).

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** account_delete
- **account** Name of account you are deleting.

Usage example:

```
curl -k -d '' --user superuser:password
https://$TARGET:/ws/account_delete?account=accountname
curl -k -d '' --user admin:password
https://$TARGET:/ws/account_delete?account=tenantname
curl -k -d '' --user tenant:password
https://$TARGET:/ws/account_delete?account=username
```

Returns:

```
{
  'status'   : 200,     # An integer, a status code (See Status
Codes for more Details),
  'result'   : ""      # A String containing a message,
}
```

Force Delete an Account and any associated access keys or datasets.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** account_delete
- **account** Name of account you are deleting.

Usage example:

```
curl -k -d '' --user superuser:password
https://$TARGET:/ws/account_delete?account=accountname\&force
```

Returns:

```
{
  'status'   : 200,     # An integer, a status code (See Status
Codes for more Details),
  'result'   : ""       # A String containing a message,
}
```

3.15 dataset_list

List all datasets / list of all tenants datasets / list of a dataset.

All Datasets Returns a list of All accounts.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** dataset_list

Usage example:

```
curl -k --user tenant:password    https://$TARGET:/ws/dataset_list
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result'   : {
        "Dataset1" : {    # Dataset Name
            'created'       : 1506007185,   # An integer,   The date
the account was created
            'records'       : 0,            # An integer,   A
snapshot of the record count
            'createdby'     : "user1_1",    # A string,     The
account used to create this dataset
            'tenant'        : "Tenant1"     # A string,     The
tenant that owns dataset
        },
        "Dataset2" : {     # Additional Datasets belonging to the
Tenant
    }
}
```

Tenant datasets Returns a list of all tenants datasets

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** dataset_list

Usage example:

```
curl -k --user superuser:password
https://$TARGET:/ws/dataset_list?tenant=tenant
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result'   : {
    "Tenant1": {
        "Dataset1" : {    # Dataset Name
            'created'       : 1506007185,   # An integer,   The date
the account was created
            'records'       : 0,            # An integer,   A
snapshot of the record count
            'createdby'     : "user1_1",    # A string,     The
account used to create this dataset
            'tenant'        : "Tenant1"     # A string,     The
tenant that owns dataset
        },
        "Dataset2" : {     # Additional Datasets belonging to this
Tenant

        }
    },
    "Tenant2": {

    }
  }
}
```

Dataset Returns a listing of a dataset

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** dataset_list
- **tenant** Name of tenant whose accounts you wish to return.

Usage example:

```
curl -k --user tenant:password
https://$TARGET:/ws/dataset_list?dataset=dataset          # listing
of this dataset, provided it belongs to this tenant
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result'   : {
        "Dataset1" : {    # Dataset Name
            'created'       : 1506007185,   # An integer,   The date
the account was created
            'records'       : 0,            # An integer,   A
snapshot of the record count
            'createdby'     : "user1_1",    # A string,     The
account used to create this dataset
            'tenant'        : "Tenant1"     # A string,     The
tenant that owns dataset
        }
    }
}
```

3.16 dataset_create

Create Dataset

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** dataset_create
- **dataset** The Name of the Dataset you want to create.

Usage example:

```
curl -k -d '' --user tenant:password
https://$TARGET:/ws/dataset_create?dataset=dataset_name
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result'   : {
        "Dataset1" : {    # Dataset Name
            'tenant'        : "Tenant1"     # A string,     The
tenant that owns dataset
            'createdby'     : "user1_1",    # A string,     The
account used to create this dataset
            'created'       : 1506007185    # An Integer,   The date
the account was created
        }
    }
}
```

3.17 dataset_delete

Delete a Dataset / Force Delete a Dataset.

Delete Dataset (Datasets must have no access keys before they can be deleted. Use force to override).

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** dataset_delete
- **tenant** The Tenant that owns dataset.
- **dataset** The Name of the Dataset you want to delete.

Usage example:

```
curl -k -d '' --user tenant:password
https://$TARGET:/ws/dataset_delete?tenant=tenant\&dataset=dataset_na
me
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result'   : ""       # A String containing a message
}
```

Force Delete Dataset (Will delete any access keys associated with the dataset).

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** dataset_delete
- **tenant** The Tenant that owns dataset.
- **dataset** The Name of the Dataset you want to delete.

Usage example:

```
curl -k -d '' --user tenant:password
https://$TARGET:/ws/dataset_delete?tenant=tenant\&dataset=dataset_na
me\&force
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result'   : ""       # A String containing a message
}
```

3.18 accesskey_list

List all access keys / list of a tenants access keys / return tenants accesskey / Returns specified accesskey.

All Returns a list of All access keys.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** accesskey_list

Usage example:

```
curl -k --user superuser:password https://$TARGET:/ws/accesskey_list
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result'   : {
        “tenant2”: {
            "8b510b45-62e7-429e-b8c8-2f74315dc722": { # Accesskey
Name
                'maxenrols' : 1000,             # An Integer,
Maximum number of enrolments
                'maxverifs' : 10000,            # An Integer,
Maximum number of verifications,
                'created'   : 1506007250,       # An Integer,  The
date the accesskey was created
                'enabled'   : "T",              # A string,
Accesskey is enabled = T or Disabled = F
                'dataset'   : "dataset2",       # A string,    The
name of the dataset that owns accesskey
                'createdby' : "tenant2",        # A string,    The
account used to create this dataset
                'notes'     : "A note here",    # A string,    Notes
added to the accesskey
                'tenant'    : "tenant2"         # A string,    The
tenant that owns dataset
            },
            "9b510b45-92e7-929e-98c8-9f74315dc722": { # Additional
Access keys
            }
        },
        “tenantN”: { # Additional tenants with Access keys
        }
    }
}
```

Usage example:

```
curl -k --user tenant2:password
https://$TARGET:/ws/accesskey_list
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result'   : {
        "8b510b45-62e7-429e-b8c8-2f74315dc722": { # Accesskey Name
            'maxenrols' : 1000,             # An Integer,  Maximum
number of enrolments
            'maxverifs' : 10000,            # An Integer,  Maximum
number of verifications
            'created'   : 1506007250,       # An Integer,  The date
the accesskey was created
            'enabled'   : "T",              # A string,    Accesskey
is enabled = T or Disabled = F
            'dataset'   : "dataset2",       # A string,    The name
of the dataset that owns accesskey
            'createdby' : "tenant2",        # A string,    The
account used to create this dataset
            'notes'     : "A note here",    # A string,    Notes
added to the accesskey
            'tenant'    : "tenant2"         # A string,    The
tenant that owns dataset
        },
        "9b510b45-92e7-929e-98c8-9f74315dc722": { # Additional
Access keys
        }
    }
}
```

All tenants access keys Returns a list of a tenants access keys

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** accesskey_list
- **tenant** The name of the tenant that owns the accesskey.

Usage example:

```
curl -k --user superuser:password
https://$TARGET:/ws/accesskey_list?tenant=tenant2              #
list of all access keys this tenant
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result'   : {
        "8b510b45-62e7-429e-b8c8-2f74315dc722": { # Accesskey Name
            'maxenrols' : 1000,             # An Integer,  Maximum
number of enrolments
            'maxverifs' : 10000,            # An Integer,  Maximum
number of verifications
            'created'   : 1506007250,       # An Integer,  The date
the accesskey was created
            'enabled'   : "T",              # A string,    Accesskey
is enabled = T or Disabled = F
            'dataset'   : "dataset2",       # A string,    The name
of the dataset that owns accesskey
            'createdby' : "tenant2",        # A string,    The
account used to create this dataset
            'notes'     : "A note here",    # A string,    Notes
added to the accesskey
            'tenant'    : "tenant2"         # A string,    The
tenant that owns dataset
        },
        "9b510b45-92e7-929e-98c8-9f74315dc722": { # Additional
access keys
        }
    }
}
```

Tenants accesskey Returns a tenants accesskey.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** accesskey_list
- **tenant** The name of the tenant that owns the accesskey.
- **accesskey** The name of the accesskey you wish to return.

Usage example:

```
curl -k --user superuser:password
https://$TARGET:/ws/accesskey_list?tenant=tenant2\&accesskey=whateve
r
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result'   : {
        "8b510b45-62e7-429e-b8c8-2f74315dc722": { # Accesskey Name
            'maxenrols' : 1000,             # An Integer,  Maximum
number of enrolments
            'maxverifs' : 10000,            # An Integer,  Maximum
number of verifications
            'created'   : 1506007250,       # An Integer,  The date
the accesskey was created
            'enabled'   : "T",              # A string,    Accesskey
is enabled = T or Disabled = F
            'dataset'   : "dataset2",       # A string,    The name
of the dataset that owns accesskey
            'createdby' : "tenant2",        # A string,    The
account used to create this dataset
            'notes'     : "A note here",    # A string,    Notes
added to the accesskey
            'tenant'    : "tenant2"         # A string,    The
tenant that owns dataset
        }
    }
}
```

Accesskey Returns the specified accesskey.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** accesskey_list
- **accesskey** The name of the accesskey you wish to return.

Usage example:

```
curl -k --user superuser:password
https://$TARGET:/ws/accesskey_list?accesskey=whatever
curl -k --user tenant:password
https://$TARGET:/ws/accesskey_list?accesskey=whatever
```

Returns:

```
{
  'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
  'result'   : {
        "8b510b45-62e7-429e-b8c8-2f74315dc722": { # Accesskey Name
            'maxenrols' : 1000,            # An Integer,  Maximum
number of enrolments
            'maxverifs' : 10000,           # An Integer,  Maximum
number of verifications
            'created'   : 1506007250,      # An Integer,  The date
the accesskey was created
            'enabled'   : "T",             # A string,    accesskey
is enabled = T or Disabled = F
            'dataset'   : "dataset2",      # A string,    The name
of the dataset that owns accesskey
            'createdby' : "tenant2",       # A string,    The
account used to create this dataset
            'notes'     : "A note here",   # A string,    Notes
added to the accesskey
            'tenant'    : "tenant2"        # A string,    The
tenant that owns dataset
        }
    }
}
```

3.19 accesskey_create

Create an accesskey

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** accesskey_create
- **dataset** The Name of the Dataset you want to create.
- **maxenrols** Maximum (quota) total number of enrolments.
- **maxverifs** Maximum (quota) total number of verifications.
- **note** A Note that can be added to the accesskey.
- **enable** enable account=T Disable Account=F.

Usage example:

```
curl -k -d '' --user tenant:password
https://$TARGET:/ws/accesskey_create?dataset=dataset_name\&maxenrols
=1000\&maxverifs=10000\&note=an%20appropriately%20quoted%20string\&e
nable=F
```

Returns:

```
{
   'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
   'result'   : {
         "8b510b45-62e7-429e-b8c8-2f74315dc722": { # Accesskey Name
             'maxenrols' : 1000,             # An Integer,  Maximum
number of enrolments
             'maxverifs' : 10000,            # An Integer,  Maximum
number of verifications
             'created'   : 1506007250,       # An Integer,  The date
the accesskey was created
             'enabled'   : "T",              # A string,
accesskey is enabled = T or Disabled = F
             'dataset'   : "dataset2",       # A string,    The name
of the dataset that owns accesskey
             'createdby' : "tenant2",        # A string,    The
account used to create this dataset
             'notes'     : "A note here",    # A string,    Notes
added to the accesskey
             'tenant'    : "tenant2"         # A string,    The
tenant that owns dataset
         }
     }
 }
```

3.20 accesskey_edit

Edit an accesskey / Enable or disable an Accesskey.

Edit an accesskey.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** accesskey_edit
- **accesskey** The name of the accesskey you wish to edit.
- **maxenrols** Maximum (quota) total number of enrolments.
- **maxverifs** Maximum (quota) total number of verifications.
- **note** A Note that can be added to the accesskey.
- **enable** enable account=T Disable Account=F.

Usage example:

```
curl -k -d '' --user tenant:password
https://$TARGET:/ws/accesskey_edit?accesskey=whatever\&maxenrols=200
0\&maxverifs=20000\&note=\&enable=T
```

Returns:

```
{
   'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
   'result'   : {
         "8b510b45-62e7-429e-b8c8-2f74315dc722": { # Accesskey Name
             'maxenrols' : 1000,            # An Integer, Maximum
number of enrolments
             'maxverifs' : 10000,           # An Integer, Maximum
number of verifications
             'created'   : 1506007250,      # An Integer, The date
the accesskey was created
             'enabled'   : "T",             # A string,   accesskey
is enabled = T or Disabled = F
             'dataset'   : "dataset2",      # A string,   The name
of the dataset that owns accesskey
             'createdby' : "tenant2",       # A string,   The
account used to create the associated dataset
             'notes'     : "A note here",   # A string,   Notes
added to the accesskey
             'tenant'    : "tenant2"        # A string,   The
tenant that owns the associated dataset
          }
     }
 }
```

Enable/disable Enable or disable an Accesskey.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** accesskey_edit
- **accesskey** The name of the accesskey you wish to edit.
- **enable** enable account=T Disable Account=F.

Usage example:

```
curl -k --user superuser:password
https://$TARGET:/ws/accesskey_edit?accesskey=whatever\&enable=T
```

Returns:

```
{
   'status'   : 200,     # An integer containing a status code (See
Status Codes for more Details)
   'result'   : {
         "8b510b45-62e7-429e-b8c8-2f74315dc722": { # Accesskey Name
             'maxenrols' : 1000,             # An Integer, Maximum
number of enrolments
             'maxverifs' : 10000,            # An Integer, Maximum
number of verifications
             'created'   : 1506007250,       # An Integer, The date
the accesskey was created
             'enabled'   : "T",              # A string,   accesskey
is enabled = T or Disabled = F
             'dataset'   : "dataset2",       # A string,   The name
of the dataset that owns accesskey
             'createdby' : "tenant2",        # A string,   The
account used to create the associated dataset
             'notes'     : "A note here",    # A string,   Notes
added to the accesskey
             'tenant'    : "tenant2"         # A string,   The
tenant that owns the associated dataset
         }
     }
 }
```

3.21 accesskey_delete

Delete an accesskey.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** accesskey_delete
- **accesskey** The name of the accesskey you wish to delete.

Usage example:

```
curl -k --user tenant:password -d ''
https://$TARGET:/ws/accesskey_delete?accesskey=whatever
```

Returns:

```
{
  'status'   : 200,     # An integer, a status code (See Status
Codes for more Details)
  'result'   : ""       # A String containing a message
}
```

3.22 ssh_logins

Enable/Disable SSH Logins.

Required arguments:

- **user:password** The account name and password used to login.
- **$TARGET:** The IP address of the node.
- **command** ssh_logins
- **enable** enable account=T Disable Account=F.

Usage example:

```
curl -k --user superuser:password -d ''
https://$TARGET:/ws/ssh_logins?enable=T
curl -k --user superuser:password -d ''
https://$TARGET:/ws/ssh_logins?enable=F
```

Returns:

```
{
  'status'   : 200,     # An integer, a status code (See
Status Codes for more Details)
  'result'   : ""       # A String containing a message
}
```