Landscape API (Python 3) Documentation

Release 0.8.0

Jiří Altman

Oct 02, 2022

CONTENTS:

1 Landscape API (Python 3) 1
- 1.1 Features . . . 1
- 1.2 Quick start . . . 1
- 1.3 Known issues . . . 2
- 1.4 Credits . . . 2
- 1.5 License . . . 2

2 Installation 3
- 2.1 Stable release . . . 3
- 2.2 From sources . . . 3

3 Usage 5

4 landscape_api 7
- 4.1 landscape_api package . . . 7

5 Contributing 33
- 5.1 Types of Contributions . . . 33
- 5.2 Get Started! . . . 34
- 5.3 Pull Request Guidelines . . . 35
- 5.4 Tips . . . 35
- 5.5 Deploying . . . 35

6 Credits 37
- 6.1 Development Lead . . . 37
- 6.2 Contributors . . . 37

7 History 39
- 7.1 v0.7.1 (2021-01-27) . . . 39
- 7.2 v0.7.0 (2021-01-27) . . . 39
- 7.3 v0.6.1 (2020-06-17) . . . 39
- 7.4 v0.5.0 (2020-06-11) . . . 39
- 7.5 v0.4.2 (2020-06-10) . . . 40
- 7.6 v0.4.1 (2020-06-10) . . . 40
- 7.7 v0.3.5 (2020-06-10) . . . 40
- 7.8 v0.3.4 (2020-06-09) . . . 40
- 7.9 v0.2.0alpha (2020-06-08) - First pre-release of landscape-api package . . . 40
- 7.10 0.1.3 (2020-06-07) . . . 41

8 Indices and tables 43

Python Module Index **45**

Index **47**

CHAPTER

ONE

LANDSCAPE API (PYTHON 3)

Client for the Landscape API (Python 3)

- Free software: MIT license
- Documentation: https://landscape-api-py3.readthedocs.io.

1.1 Features

- easy installation from **PyPI** (you can use **pipenv**, **pip**, **pipex**, **Chocolatey**, ...)
- working on **Windows** (**pipx** create **landscape-api.exe** shim)
- working with **Python>=v3.5** (easily **from landscape_api_py3.base import API**)
- for quick use can be installed with **pipx install landscape_api_py3**

1.2 Quick start

Check if you have installed **Python v3.5** and above.

To install Landscape API (Python 3), run this command in your terminal:

On Linux:

```
$ pip install landscape-api-py3
$ python -m landscape-api --uri https://your-uri-to-ls-api/api --key <your API key> --
↪secret <your API secret> --json get-computers --limit 1
```

On Windows:

```
C:\> pip install landscape-api-py3
C:\> python -m landscape-api --uri https://your-uri-to-ls-api/api --key <your API key> --
↪secret <your API secret> --json get-computers --limit 1
```

or you can use **pipx** (virtual environment will be created automatically):

On Linux:

```
$ pip install --user pipx
$ pipx ensurepath
$ exec $SHELL # Restart your shell to reload PATH
$ pipx install landscape-api-py3
$ landscape-api --uri https://your-uri-to-ls-api/api --key <your API key> --secret <your␣
↪API secret> --json get-computers --limit 1
```

On Windows:

```
C:\> pip install --user pipx
C:\> pipx ensurepath
C:\> REM Restart console window to reload PATH
C:\> pipx install landscape-api-py3
C:\> landscape-api --uri https://your-uri-to-ls-api/api --key <your API key> --secret
↪<your API secret> --json get-computers --limit 1
```

1.3 Known issues

- none (issues with dependencies resolved in v0.3.0)

1.4 Credits

Based on package landscape-api from Canonical Ltd. This package was created with Cookiecutter and the audreyr/cookiecutter-pypackage project template.

1.5 License

CHAPTER

TWO

INSTALLATION

2.1 Stable release

To install Landscape API (Python 3), run this command in your terminal:

```
$ pip install landscape_api_py3
```

This is the preferred method to install Landscape API (Python 3), as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.2 From sources

The sources for Landscape API (Python 3) can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/jurya/landscape_api_py3
```

Or download the tarball:

```
$ curl -OJL https://github.com/jurya/landscape_api_py3/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER

THREE

USAGE

To use Landscape API (Python 3) as CLI:

```
$ landscape-api --json get-computers --limit 1
```

To use Landscape API (Python 3) in a Python project:

```
from landscape_api.base import API, HTTPError
```

CHAPTER

FOUR

LANDSCAPE_API

4.1 landscape_api package

4.1.1 Submodules

4.1.2 landscape_api.base module

Base module for Landscape API (Python 3).

class landscape_api.base.**API**(*uri*, *access_key*, *secret_key*, *ssl_ca_file=None*, *json=False*, *schema=None*)

Bases: *API*

accept_pending_computers(*computer_ids*, *existing_ids={}*, *access_group=None*)

Accept a list of pending computers associated with the account used for authentication.

Parameters

- **computer_ids** (*list (of integer)*) – A list of computer IDs to accept.
- **existing_ids** (*mapping*) – A mapping from pending computer IDs to existing ones.
- **access_group** (*unicode*) – The access group to put the computers into

add_access_groups_to_role(*name*, *access_groups*)

Add the given access groups to a role.

Parameters

- **name** (*unicode*) – The name of the role to modify.
- **access_groups** (*list (of unicode)*) – A list of names of access groups to add to the role.

add_annotation_to_computers(*query*, *key*, *value=None*)

Add annotation key and optional value to a selection of computers.

Parameters

- **query** (*unicode*) – A query string used to select the computers to which to add the annotation.
- **key** (*unicode*) – Annotation key to add to the selected computers.
- **value** (*unicode*) – Annotation value associated with the provided key to add to the selected computers.

add_apt_sources_to_repository_profile(*name*, *apt_sources*)

Add APT sources to a repository profile. An activity will be created to add the given source to the the computers associated with the given profile.

Parameters

- **name** (*unicode*) – Name of the repository profile.
- **apt_sources** (*list (of unicode)*) – The names of the APT sources to add.

add_package_filters_to_pocket(*name*, *series*, *distribution*, *packages*)

Add package filters to a repository pocket. The pocket must be in pull mode and support blacklist/whitelist filtering.

Parameters

- **name** (*unicode*) – The name of the pocket to operate on.
- **series** (*unicode*) – The name of the series containing the pocket.
- **distribution** (*unicode*) – The name of the distribution containing the series.
- **packages** (*list (of unicode)*) – A list of names of packages to be added or removed from the pocket filter.

add_permissions_to_role(*name*, *permissions*)

Add permissions to a role.

Parameters

- **name** (*unicode*) – The name of the role to modify.
- **permissions** (*list (of unicode)*) – A list of permissions to add.

add_persons_to_role(*name*, *persons*)

Add permissions to a role.

Parameters

- **name** (*unicode*) – The name of the role to modify.
- **persons** (*list (of unicode)*) – A list of emails of persons to add.

add_pockets_to_repository_profile(*name*, *pockets*, *series*, *distribution*)

Add repository pockets to a repository profile. An activity will be created to add the given pockets to the APT sources of the computers associated with the given profile.

Parameters

- **name** (*unicode*) – Name of the repository profile.
- **pockets** (*list (of unicode)*) – The names of the pockets to add.
- **series** (*unicode*) – The name of the series the pockets belongs to.
- **distribution** (*unicode*) – The name of the distribution the series belongs to.

add_tags_to_computers(*query*, *tags*)

Add tags to a selection of computers.

Parameters

- **query** (*unicode*) – A query string used to select the computers to add tags to.
- **tags** (*list (of unicode)*) – Tag names to be applied.

add_uploader_gpg_keys_to_pocket(*name*, *series*, *distribution*, *gpg_keys*)

Add GPG keys to a repository pocket in upload mode to validate uploaded packages.

Parameters

- **name** (*unicode*) – The name of the pocket on which to associate keys.
- **series** (*unicode*) – The name of the series containing the pocket.
- **distribution** (*unicode*) – The name of the distribution containing the series.
- **gpg_keys** (*list (of unicode)*) – A list of GPG keys on which to operate.

approve_activities(*query*)

Approve activities associated with the current account.

Parameters

query (*unicode*) – A query string used to select activities on which to operate.

associate_alert(*name*, *tags=[]*, *all_computers=False*)

Associate an alert to computers with specific tags or to all computers. If a tag with a given name doesn't exist, it will be automatically created. An 'all_computers' value of 'true' and a list of tags are mutually exclusive. Only one or the other may be passed.

Parameters

- **name** (*unicode*) – Name of the entity.
- **tags** (*list (of unicode)*) – Tags to change entity association for
- **all_computers** (*boolean*) – If true, change the 'all_computers' flag state for the entity. If the flag is enabled, associated tags will be kept, but they will not be effective until the flag is disabled.

associate_package_profile(*name*, *tags=[]*, *all_computers=False*)

Associate a package profile to computers with specific tags or to all computers. If a tag with a given name doesn't exist, it will be automatically created.

An 'all_computers' value of 'true' and a list of tags are mutually exclusive. Only one or the other may be passed.

Parameters

- **name** (*unicode*) – Name of the entity.
- **tags** (*list (of unicode)*) – Tags to change entity association for
- **all_computers** (*boolean*) – If true, change the 'all_computers' flag state for the entity. If the flag is enabled, associated tags will be kept, but they will not be effective until the flag is disabled.

associate_removal_profile(*name*, *tags=[]*, *all_computers=False*)

Associate a removal profile to computers with the specified tags, or all computers.

tags and all_computers=true are mutually exclusive.

Parameters

- **name** (*unicode*) – Name of the entity.
- **tags** (*list (of unicode)*) – Tags to change entity association for
- **all_computers** (*boolean*) – If true, change the 'all_computers' flag state for the entity. If the flag is enabled, associated tags will be kept, but they will not be effective until the flag is disabled.

associate_repository_profile(*name*, *tags=[]*, *all_computers=False*)

Associate repository profile to computers with specified tags or to all computers.

Parameters

- **name** (*unicode*) – Name of the entity.
- **tags** (*list (of unicode)*) – Tags to change entity association for
- **all_computers** (*boolean*) – If true, change the 'all_computers' flag state for the entity. If the flag is enabled, associated tags will be kept, but they will not be effective until the flag is disabled.

associate_upgrade_profile(*name*, *tags=[]*, *all_computers=False*)

Associate an upgrade profile to computers with the specified tags, or all computers.

tags and all_computers=true are mutually exclusive.

Parameters

- **name** (*unicode*) – Name of the entity.
- **tags** (*list (of unicode)*) – Tags to change entity association for
- **all_computers** (*boolean*) – If true, change the 'all_computers' flag state for the entity. If the flag is enabled, associated tags will be kept, but they will not be effective until the flag is disabled.

call(*method*, ***kwargs*)

Invoke an API method, automatically encoding the arguments as defined in the schema.

call_arbitrary(*method*, *arguments*)

Invoke an API method in a raw form, without encoding any parameters.

Returns

The result as returned by the API method. If the `json` parameter to *API* was passed as `True`, then the raw result will be returned. Otherwise it will be decoded as json and returned as a Python object.

cancel_activities(*query*)

Cancel activities associated with the current account.

Returns a list of activities ids that were cancelled.

Parameters

query (*unicode*) – A query string used to select activities on which to operate.

change_computers_access_group(*query*, *access_group*)

Change the access group for a selection of computers.

Parameters

- **query** (*unicode*) – A query string used to select the computers to change access group for.
- **access_group** (*unicode*) – The name of the access group to assign selected computers to.

copy_package_profile(*name*, *destination_name=None*, *title=None*, *description=None*, *access_group=None*)

Copy an existing package profile to a package profile with a new name and optionally a different title and description.

Parameters

- **name** (*unicode*) – A name of the existing package profile to copy.
- **destination_name** (*unicode*) – The profile name of the copied package profile.
- **title** (*unicode*) – A title for the new profile. If not specified, the title of the source profile is used.
- **description** (*unicode*) – A description for the new profile. If not specified, the title of the source profile is used.
- **access_group** (*unicode*) – Name of the access group to copy the profile to. Defaults to the origin's access group.

copy_role(*name*, *destination_name*, *description=None*)

Copy an existing access role to a role with a new name.

Parameters

- **name** (*unicode*) – The name of the existing role.
- **destination_name** (*unicode*) – The name of the new role.
- **description** (*unicode*) – The description of the new role.

copy_script(*script_id*, *destination_title*, *access_group=None*)

Copy an existing script to a script with a new name.

Parameters

- **script_id** (*integer*) – The identity of the existing script.
- **destination_title** (*unicode*) – The title of the new script.
- **access_group** (*unicode*) – The access group for the new script. It defaults to the same access group as the existing script.

create_access_group(*title*, *parent=None*)

Create a new access group.

Parameters

- **title** (*unicode title*) – The title of the access group.
- **parent** (*unicode*) – The title of the parent access group.

create_apt_source(*name*, *apt_line*, *gpg_key=None*, *access_group='global'*)

Create an APT source in the account used for authentication.

Parameters

- **name** (*unicode*) – Name of the APT source. It must be unique within the account, start with an alphanumeric character and only contain lowercase letters, numbers and - or + signs.
- **apt_line** (*unicode*) – The APT line of the source.
- **gpg_key** (*unicode*) – Name of the GPG key used to sign the repository
- **access_group** (*unicode*) – An optional name of the access group to create the APT source into.

create_cloud_otps(*count=1*)

Create one-time passwords used for registration of cloud instances.

Parameters

count (*integer*) – The number of OTPs to create

create_distribution(*name*, *access_group='global'*)

Create a repository distribution associated with the account.

Parameters

- **name** (*unicode*) – The name of the distribution. It must be unique within the account, start with an alphanumeric character and only contain lowercase letters, numbers and - or + signs.
- **access_group** (*unicode*) – An optional name of the access group to create the distribution into.

create_package_profile(*title*, *description*, *source_computer_id=None*, *material=None*, *constraints=[]*, *access_group='global'*)

Create a package profile.

source_computer_id and material are mutually exclusive.

Parameters

- **title** (*unicode title*) – The title of the package profile to create.
- **description** (*unicode*) – The description of the new profile.
- **source_computer_id** (*integer*) – A computer ID to find a computer which will be used as the basis of the package profile.
- **material** (*unicode*) – Package data in the format of 'dpkg –get-selections' or CSV (as exported by Landscape).
- **constraints** (*list (of unicode)*) – Alternative to material, constraint specifications in the form of "depends packagename" or "conflicts packagename < 1.0".
- **access_group** (*unicode*) – Optional name of the access group to create the profile into

create_pocket(*name*, *series*, *distribution*, *components*, *architectures*, *mode*, *gpg_key*, *include_udeb=False*, *mirror_uri=None*, *mirror_suite=None*, *mirror_gpg_key=None*, *pull_series=None*, *pull_pocket=None*, *filter_type=None*, *upload_allow_unsigned=False*)

Create a pocket associated with a series in the account.

Parameters

- **name** (*unicode*) – The name of the pocket. It must be unique within series, start with an alphanumeric character and only contain lowercase letters, numbers and - or + signs.
- **series** (*unicode*) – The name of the series to create the pocket in.
- **distribution** (*unicode*) – The name of the distribution the series belongs to.
- **components** (*list (of unicode)*) – A list of components the pocket will handle.
- **architectures** (*list (of unicode)*) – A list of architectures the pocket will handle.
- **mode** (*unicode*) – The pocket mode. Can be 'pull', 'mirror' and 'upload'.
- **gpg_key** (*unicode*) – The name of the GPG key to use to sign packages lists for this pocket. The GPG key provided must have a private key associated with it.

- **include_udeb** (*boolean*) – Whether the pocket should include selected components also for .udeb packages (debian-installer). It's 'false' by default.
- **mirror_uri** (*unicode*) – The URI to mirror for pockets in 'mirror' mode.
- **mirror_suite** (*unicode*) – The repository entry under dists/ to mirror for pockets in 'mirror' mode. This parameter is optional and defaults to the same name as local series and pocket. If the suite name ends with a '/', the remote repository is flat (packages are not grouped in components); in this case a single value can be passed for the 'components' parameter. Packages from the remote repository will be mirrored in the specified component.
- **mirror_gpg_key** (*unicode*) – The name of the GPG key to use to verify the mirrored archive signature. If none is given, the stock Ubuntu archive one will be used.
- **pull_series** (*unicode*) – The name of the series pull_pocket belongs to. Must be a series in the same distribution series belongs to. If not specified, it defaults to series.
- **pull_pocket** (*unicode*) – The name of a pocket in current distribution to sync packages from for pockets in 'pull' mode.
- **filter_type** (*unicode*) – If specified, the type of the filter of the pocket. Can be either 'whitelist' or 'blacklist'.
- **upload_allow_unsigned** (*boolean*) – For pockets in upload mode, a boolean indicating whether uploaded packages are required to be signed or not. It's 'false' by default.

create_removal_profile(*title*, *days_without_exchange*, *access_group='global'*)

Create a removal profile.

Parameters

- **title** (*unicode*) – The title of the profile to create.
- **days_without_exchange** (*integer*) – The length of time after which a computer may be removed.
- **access_group** (*unicode*) – An optional name of an access group the profile will apply to.

create_repository_profile(*title*, *description=None*, *access_group='global'*)

Create a repository profile.

Parameters

- **title** (*unicode title*) – Title of the repository profile. It must start with an alphanumeric character and only contain lowercase letters, numbers and - or + signs.
- **description** (*unicode*) – Description of the repository profile.
- **access_group** (*unicode*) – Optional name of the access group to create the profile in.

create_role(*name*, *description=None*)

Create a new access role.

Parameters

- **name** (*unicode*) – The name of the role.
- **description** (*unicode*) – The description of the role.

create_saved_search(*title*, *search*, *name=None*)

Create a new saved search associated with the current account.

Parameters

- **name** (*unicode*) – The "slug" name for this saved search. It must consist of only lowercase ASCII letters, numbers and hyphens. This is the text which must be used when using the "search:name" syntax. If this parameter is not included a name will be generated automatically based on the title.
- **title** (*unicode title*) – The display name for the SavedSearch.
- **search** (*unicode*) – The search string to save.

create_script(*title*, *time_limit*, *code*, *username=None*, *access_group=None*)

Create a new script.

Parameters

- **title** (*unicode title*) – The title of the new script.
- **time_limit** (*integer*) – Amount of time to wait for the process to end.
- **code** (*data*) – The filename holding the script contents.
- **username** (*unicode*) – The user to execute the script as.
- **access_group** (*unicode*) – The access group for the new script.

create_script_attachment(*script_id*, *file*)

Add a script attachment.

Parameters

- **script_id** (*integer*) – The identity of the script to add the attachment to.
- **file** (*file*) – The file to attach

create_series(*name*, *distribution*, *pockets=[]*, *components=[]*, *architectures=[]*, *gpg_key=None*, *mirror_uri=None*, *mirror_series=None*, *mirror_gpg_key=None*, *include_udeb=False*)

Create a series associated with a distribution in the account.

Parameters

- **name** (*unicode*) – The name of the series. It must be unique within series within the distribution, start with an alphanumeric character and only contain lowercase letters, numbers and - or + signs.
- **distribution** (*unicode*) – The name of the distribution to create the series in.
- **pockets** (*list (of unicode)*) – Pockets that will be created in the series, they will be in mirror mode by default.
- **components** (*list (of unicode)*) – List of components for the created pockets. This parameter is **optional** if no pocket is specified.
- **architectures** (*list (of unicode)*) – List of architectures for the created pockets. This parameter is **optional** if no pocket is specified
- **gpg_key** (*unicode*) – The name of the GPG key to use to sign packages lists of the created pockets. This parameter is **optional** if no pocket is specified.
- **mirror_uri** (*unicode*) – The URI to mirror for the created pockets. This parameter is **optional** if no pocket is specified.
- **mirror_series** (*unicode*) – The remote series to mirror. If not specified, it defaults to the name of the series being created. If a pockets parameter also passed, each of the created pockets will mirror the relevant dists/<mirror_series>-<pocket> repository of the remote archive.

- **mirror_gpg_key** (*unicode*) – The name of the GPG key to use to verify the mirrored repositories for created pockets. If none is given, the stock Ubuntu archive one will be used.
- **include_udeb** (*boolean*) – Whether the pocket should include selected components also for .udeb packages (debian-installer). It's 'false' by default.

create_upgrade_profile(*title*, *every*, *at_minute*, *on_days=[]*, *at_hour=None*, *deliver_within=1*, *deliver_delay_window=0*, *security_upgrade=False*, *upgrade_type=None*, *autoremove=False*, *access_group='global'*)

Create an upgrade profile.

Parameters

- **title** (*unicode title*) – A human readable title for this upgrade profile.
- **every** (*unicode*) – The frequency at which you wish this upgrade profile to be executed. Valid choices are "hour" and "week".
- **on_days** (*list (of unicode)*) – A list of days of the week on which the upgrade profile will be run. The day names must be abbreviated to their first two letters, as: "mo", "tu", "we", "th", "fr", "sa", "su". Required when the every parameter is "week" but optional when the every parameter is "hour".
- **at_hour** (*integer*) – The hour, in 24h format, at which the upgrade profile will be run.
- **at_minute** (*integer*) – The minute of the hour (0-59) at which the upgrade profile will be run.
- **deliver_within** (*integer*) – An optional number of hours within which the upgrade task should be delivered to computers. The window will be from the time specified by this API call (on_days, at_hour, at_minute) until the provided number of hours later. Defaults to 1 hour.
- **deliver_delay_window** (*integer*) – Randomise delivery within the given timeframe specified in minutes.
- **security_upgrade** (*boolean*) – (Deprecated) Whether this upgrade is a security upgrade or not.
- **upgrade_type** (*unicode*) – The type of upgrade profile, either "security" or "all".
- **autoremove** (*boolean*) – Whether this upgrade should also autoremove old packages.
- **access_group** (*unicode*) – An optional name of the access group to create the profile into.

derive_series(*name*, *origin*, *distribution*)

Derive a series from another one in the same distribution. The derived series will have pockets with names corresponding to the origin series, each one configured to pull from the pocket in origin series.

Parameters

- **name** (*unicode*) – The name of the derived series. It must be unique within the distribution, start with an alphanumeric character and only contain lowercase letters, numbers and - or + signs.
- **origin** (*unicode*) – The name of the origin series.
- **distribution** (*unicode*) – The name of the distribution to derive the series in.

diff_pull_pocket(*name*, *series*, *distribution*)

Return a list of the changes between a pocket configured in pull mode and its origin one.

Parameters

- **name** (*unicode*) – The name of the pocket.
- **series** (*unicode*) – The name of the series.
- **distribution** (*unicode*) – The name of the distribution.

disable_administrator(*email*)

Disable an administrator of your account.

Parameters

email (*unicode*) – The name of the person to disable.

disassociate_alert(*name*, *tags=[]*, *all_computers=False*)

Disassociate an alert from computers with specific tags or from all computers.

Parameters

- **name** (*unicode*) – Name of the entity.
- **tags** (*list (of unicode)*) – Tags to change entity association for
- **all_computers** (*boolean*) – If true, change the 'all_computers' flag state for the entity. If the flag is enabled, associated tags will be kept, but they will not be effective until the flag is disabled.

disassociate_package_profile(*name*, *tags=[]*, *all_computers=False*)

Disassociate package profile from computers with specified tags or from all computers.

Parameters

- **name** (*unicode*) – Name of the entity.
- **tags** (*list (of unicode)*) – Tags to change entity association for
- **all_computers** (*boolean*) – If true, change the 'all_computers' flag state for the entity. If the flag is enabled, associated tags will be kept, but they will not be effective until the flag is disabled.

disassociate_removal_profile(*name*, *tags=[]*, *all_computers=False*)

Disassociate a removal profile from computers with the specified tags, or from all computers.

tags and all_computers=true are mutually exclusive.

Parameters

- **name** (*unicode*) – Name of the entity.
- **tags** (*list (of unicode)*) – Tags to change entity association for
- **all_computers** (*boolean*) – If true, change the 'all_computers' flag state for the entity. If the flag is enabled, associated tags will be kept, but they will not be effective until the flag is disabled.

disassociate_repository_profile(*name*, *tags=[]*, *all_computers=False*)

Disassociate repository profile from computers with specified tags or from all computers.

Parameters

- **name** (*unicode*) – Name of the entity.
- **tags** (*list (of unicode)*) – Tags to change entity association for

- **all_computers** (*boolean*) – If true, change the 'all_computers' flag state for the entity. If the flag is enabled, associated tags will be kept, but they will not be effective until the flag is disabled.

disassociate_upgrade_profile(*name*, *tags=[]*, *all_computers=False*)

Disassociate an upgrade profile from computers with the specified tags, or from all computers.

tags and all_computers=true are mutually exclusive.

Parameters

- **name** (*unicode*) – Name of the entity.
- **tags** (*list (of unicode)*) – Tags to change entity association for
- **all_computers** (*boolean*) – If true, change the 'all_computers' flag state for the entity. If the flag is enabled, associated tags will be kept, but they will not be effective until the flag is disabled.

edit_package_profile(*name*, *title=None*, *add_constraints=[]*, *remove_constraints=[]*)

Add or remove constraints related to a package profile. Constraints can be dependencies or conflicts.

Parameters

- **name** (*unicode*) – The name of the package profile.
- **title** (*unicode title*) – The new title of the package profile.
- **add_constraints** (*list (of unicode)*) – List of constraints specifications to add in the form of "depends packagename" or "conflicts packagename < 1.0".
- **remove_constraints** (*list (of unicode)*) – List of constraints specifications to remove in the form of "depends packagename" or "conflicts packagename < 1.0".

edit_pocket(*name*, *series*, *distribution*, *components=[]*, *architectures=[]*, *gpg_key=None*, *mirror_uri=None*, *mirror_suite=None*, *mirror_gpg_key=None*, *upload_allow_unsigned=None*, *include_udeb=None*)

Edit configuration for a repository pocket from a series in a distribution.

Parameters

- **name** (*unicode*) – The name of the pocket to edit.
- **series** (*unicode*) – The name of the series containing the pocket.
- **distribution** (*unicode*) – The name of the distribution containing the series.
- **components** (*list (of unicode)*) – A list of components the pocket will handle.
- **architectures** (*list (of unicode)*) – A list of architectures the pocket will handle.
- **gpg_key** (*unicode*) – The name of the GPG key to use to sign packages lists for this pocket. The GPG key provided must have a private key associated with it.
- **mirror_uri** (*unicode*) – The URI to mirror for pockets in 'mirror' mode.
- **mirror_suite** (*unicode*) – The repository entry under dists/ to mirror for pockets in 'mirror' mode.
- **mirror_gpg_key** (*unicode*) – The name of the GPG key to use to verify the mirrored archive signature. If '-' is given, the stock Ubuntu archive one will be used.
- **upload_allow_unsigned** (*boolean*) – For pockets in upload mode, a boolean indicating whether uploaded packages are required to be signed or not.

- **include_udeb** (*boolean*) – Whether the pocket should include selected components also for .udeb packages (debian-installer).

edit_removal_profile(*name*, *title=None*, *days_without_exchange=None*)

Edit an removal profile.

Parameters

- **name** (*unicode*) – The name of the profile to edit.
- **title** (*unicode*) – The new title of the profile.
- **days_without_exchange** (*integer*) – The length of time after which a computer may be removed.

edit_repository_profile(*name*, *title=None*, *description=None*)

Edit a repository profile.

Parameters

- **name** (*unicode*) – Name of the repository profile to edit.
- **title** (*unicode title*) – Title of the repository profile.
- **description** (*unicode*) – Description of the repository profile.

edit_saved_search(*name*, *title=None*, *search=None*)

Edit a saved search associated with the current account.

Parameters

- **name** (*unicode*) – The "slug" name for this saved search, this is the text which must be used when using the "search:name" syntax. A saved search with this name must already exist in the account.
- **title** (*unicode title*) – The new display name for the saved search. If this parameter is not included then the title will not be modified.
- **search** (*unicode*) – The search string to save. If this parameter is not included then the search string will not be modified.

edit_script(*script_id*, *title=None*, *time_limit=None*, *code=None*, *username=None*)

Edit a script.

Parameters

- **script_id** (*integer*) – The identifier of the script you wish to edit.
- **title** (*unicode title*) – The new script title.
- **time_limit** (*integer*) – Amount of time to wait for the process to end.
- **code** (*data*) – The filename holding the script contents.
- **username** (*unicode*) – The user to execute the script as.

edit_upgrade_profile(*name*, *title=None*, *every=None*, *on_days=[]*, *at_hour=None*, *at_minute=None*, *deliver_within=1*, *deliver_delay_window=None*, *security_upgrade=False*, *upgrade_type=None*, *autoremove=None*)

Edit an upgrade profile.

Parameters

- **name** (*unicode*) – The name for this upgrade profile.
- **title** (*unicode title*) – The new title of the upgrade profile.

- **every** (*unicode*) – The frequency at which you wish this upgrade profile to be executed. Valid choices are "hour" and "week".
- **on_days** (*list (of unicode)*) – A list of days of the week on which the upgrade profile will be run. The day names must be abbreviated to their first two letters, as: "mo", "tu", "we", "th", "fr", "sa", "su". Required when the every parameter is "week" but optional when the every parameter is "hour".
- **at_hour** (*integer*) – The hour, in 24h format, at which the upgrade profile will be run.
- **at_minute** (*integer*) – The minute of the hour (0-59) at which the upgrade profile will be run.
- **deliver_within** (*integer*) – An optional number of hours within which the upgrade task should be delivered to computers. The window will be from the time specified by this API call (on_days, at_hour, at_minute) until the provided number of hours later. Defaults to 1 hour.
- **deliver_delay_window** (*integer*) – Randomise delivery within the given timeframe specified in minutes.
- **security_upgrade** (*boolean*) – (Deprecated) Whether this upgrade is a security upgrade or not.
- **upgrade_type** (*unicode*) – The type of upgrade profile, either "security" or "all".
- **autoremove** (*boolean*) – Whether this upgrade should also autoremove old packages.

execute_script(*query*, *script_id*, *username=None*, *deliver_after=None*)

Execute a script on computers.

Parameters

- **query** (*unicode*) – A query string used to select the computers to execute the script on. Multiple occurrences will be joined with a logical AND.
- **script_id** (*integer*) – The identity of the script stored in the server.
- **username** (*unicode*) – The username to execute the script as on the client. Required if the script has no default username.
- **deliver_after** (*unicode*) – A time in the future to deliver the script.

```
extra_actions = [action(name='ssh', method_name='ssh', doc='Try to ssh to a
landscape computer', required_args=[{'name':  'query', 'type':  'unicode', 'doc':
'A query string which should return one computer'}], optional_args=[{'name':
'user', 'type':  'unicode', 'default':  None, 'doc':  'If specified, the user to
pass to the ssh command'}])]
```

get_access_groups(*names=[]*)

List all access groups in the account.

Parameters

names (*list (of unicode)*) – The name of the access group.

get_activities(*query=''*, *limit=1000*, *offset=0*)

Retrieve activities associated with the current account, ordered by creation time.

Some activities requested take an extended period of time to complete. These activities will not have discrete activity_status values. Instead they will report estimated percent complete in the progress field for the activity. The progress field will have one of the following values:

```
0: if activity is not started
-1: if an error occurred
1 to 100: percent complete of ongoing activity
```

Common examples of activities with progress would be syncing a pocket repository mirror or provisioning a new system.

Parameters

- **query** (*unicode*) – A query string with space separated tokens used to filter the returned result objects.
- **limit** (*integer*) – The maximum number of results returned by the method. It defaults to 1000.
- **offset** (*integer*) – The offset inside the list of results.

get_activity_types()

Retrieve a list of possible activity types for use with the *type:* query criteria.

get_administrators()

Retrieve the list of administrators in the account.

get_alert_subscribers(*alert_type*)

Get a a list of people who subscribe to a given alert.

Parameters

alert_type (*unicode*) – The alert type to check the subscription on.

get_alerts()

Get a list of alerts in an account.

get_apt_sources(*names=[]*)

Get a list of apt sources in the account used for authentication.

Parameters

names (*list (of unicode)*) – List of names of the APT source to be returned. Multiple names can be supplied.

get_computers(*query=''*, *limit=1000*, *offset=0*, *with_network=False*, *with_hardware=False*, *with_annotations=False*)

Get a list of computers associated with the account.

Parameters

- **query** (*unicode*) – A query string with space separated tokens used to filter the returned result objects.
- **limit** (*integer*) – The maximum number of results returned by the method. It defaults to 1000.
- **offset** (*integer*) – The offset inside the list of results.
- **with_network** (*boolean*) – If true, include the details of all network devices attached to the computer.
- **with_hardware** (*boolean*) – If true, include the details of all hardware information known.
- **with_annotations** (*boolean*) – If true, include the details of all custom annotation information known.

get_computers_not_upgraded(*query=''*, *limit=1000*, *offset=0*)

Report the ids of computers, within a given selection, that are not covered by an upgrade profile.

Parameters

- **query** (*unicode*) – A query string with space separated tokens used to filter the returned result objects.
- **limit** (*integer*) – The maximum number of results returned by the method. It defaults to 1000.
- **offset** (*integer*) – The offset inside the list of results.

get_csv_compliance_data(*query=''*, *limit=1000*, *offset=0*)

Return a CSV formatted report of compliance data.

Parameters

- **query** (*unicode*) – A query string with space separated tokens used to filter the returned result objects.
- **limit** (*integer*) – The maximum number of results returned by the method. It defaults to 1000.
- **offset** (*integer*) – The offset inside the list of results.

get_distributions(*names=[]*)

Get info about distributions.

Parameters

names (*list (of unicode)*) – A list of distribution names to get info for. If this is not provided, the call will return all distributions for the account.

get_event_log(*days=None*, *limit=1000*, *offset=0*)

Retrieve event log for the account.

Parameters

- **days** (*integer*) – The number of days prior to today from which to fetch log entries. It defaults to 30 days.
- **limit** (*integer*) – The maximum number of results returned by the method. It defaults to 1000.
- **offset** (*integer*) – The offset inside the list of results.

get_gpg_keys(*names=[]*)

Get info about GPG keys.

Parameters

names (*list (of unicode)*) – A list of GPG keys to get info for. If this is not provided, the call will return all keys for the account.

get_juju_environments()

Get the details of all the Juju environments in the account.

get_juju_models()

Get the details of all the Juju models in the account.

get_not_pinging_computers(*since_minutes*, *query=''*, *limit=1000*, *offset=0*)

Report the ids of computers, within a given selection, that have not pinged the server in a given number of minutes.

Parameters

- **query** (*unicode*) – A query string with space separated tokens used to filter the returned result objects.
- **limit** (*integer*) – The maximum number of results returned by the method. It defaults to 1000.
- **offset** (*integer*) – The offset inside the list of results.
- **since_minutes** (*integer*) – The number of minutes elapsed in which no ping from included computers has been seen.

get_package_profiles(*names=[]*)

Get the details of all Package Profiles defined in the account.

Parameters

names (*list (of unicode)*) – A list of package profile names to limit the result.

get_packages(*query*, *search=None*, *names=[]*, *installed=None*, *available=None*, *upgrade=None*, *held=None*, *offset=0*, *limit=1000*)

Get a list of packages associated with the account used for authentication.

A package is considered installed if dpkg reports it as installed on the system.

A package is considered available if it can be fetched from an APT source. Note that this means that it's possible for an installed package to be not available.

A package is considered an upgrade if it's available and if it has a version higher than a non-held installed package with the same name.

Parameters

- **query** (*unicode*) – A query string used to select computers to query packages on.
- **search** (*unicode*) – A string to restrict the search to. All fields are searched, not just those returned. (e.g., description)
- **names** (*list (of unicode)*) – Restrict the search to these package names.
- **installed** (*boolean*) – If true only packages in the installed state will be returned, if false only packages not installed will be returned. If not given both installed and not installed packages will be returned.
- **available** (*boolean*) – If true only packages in the available state will be returned, if false only packages not available will be returned. If not given both available and not available packages will be returned.
- **upgrade** (*boolean*) – If true, only installable packages that are upgrades for an for an installed one are returned. If false, only installable packages that are not upgrades are returned. If not given, packages will be returned regardless of wether they are upgrades or not.
- **held** (*boolean*) – If true, only installed packages that are held on computers are returned. If false, only packages that are not held on computers are returned. If not given, packages will be returned regardless of the held state.
- **offset** (*integer*) – The offset inside the list of results.
- **limit** (*integer*) – The maximum number of results returned by the method. It defaults to 1000.

get_pending_computers()

Get a list of pending computers associated with the account used for authentication.

get_permissions()

List all available permissions.

get_removal_profiles()

List all existing removal profiles.

get_repository_profiles(*names=[]*)

Get a list of repository profiles in the account.

Parameters

names (*list (of unicode)*) – A list of repository profile names to get info for. If this is not provided, the call will return all repository profiles for the account.

get_roles(*names=[]*)

Get all available roles.

Parameters

names (*list (of unicode)*) – A list of role names to limit the result.

get_saved_searches(*offset=0*, *limit=1000*)

Retrieve saved searches associated with the current account.

Parameters

- **offset** (*integer*) – The offset inside the list of results.
- **limit** (*integer*) – The maximum number of results returned by the method. It defaults to 1000.

get_script_code(*script_id*)

Retrieve the code portion of a given script.

Parameters

script_id (*integer*) – The identity of the script you wish to get the code for.

get_scripts(*limit=1000*, *offset=0*)

Retrieve stored scripts associated with the current account.

Parameters

- **limit** (*integer*) – The maximum number of results returned by the method. It defaults to 1000.
- **offset** (*integer*) – The offset inside the list of results.

get_settings()

Get all settings and their value for the current LDS installation.

get_upgrade_profiles(*upgrade_type=None*)

List all previously created upgrade profiles.

Parameters

upgrade_type (*unicode*) – The type of upgrade you wish to list. This can be either "all" or "security", in which case the result will be a list of upgrade profiles with an upgrade type of "all" or "security" respectively. If omitted, the resulting list will contain all upgrade profiles, regardless of their upgrade type.

get_upgraded_computers_by_frequency(*query=''*, *limit=1000*, *offset=0*)

Return a dictionary of computer IDs broken down by their upgrade schedule.

Parameters

- **query** (*unicode*) – A query string with space separated tokens used to filter the returned result objects.
- **limit** (*integer*) – The maximum number of results returned by the method. It defaults to 1000.
- **offset** (*integer*) – The offset inside the list of results.

get_usn_time_to_fix(*query=''*, *limit=1000*, *offset=0*, *fixed_in_days=[]*, *pending_in_days=None*, *in_last=None*)

Return a break down of machines unpatched periods following a USN release.

Parameters

- **query** (*unicode*) – A query string with space separated tokens used to filter the returned result objects.
- **limit** (*integer*) – The maximum number of results returned by the method. It defaults to 1000.
- **offset** (*integer*) – The offset inside the list of results.
- **fixed_in_days** (*list (of integer)*) – A list of periods of days to report on USN fixes being applied in
- **pending_in_days** (*integer*) – The period of days in the past to search for USNs that are pending on a computer. This is independent of the in_last argument.
- **in_last** (*integer*) – The period of days to look into the past to find USN releases to be considered in these statistics.

import_gpg_key(*name*, *material*)

Import a GPG key.

Parameters

- **name** (*unicode*) – Name of the GPG key. It must be unique within the account, start with an alphanumeric character and only contain lowercase letters, numbers and - or + signs.
- **material** (*unicode*) – The text representation of the key.

import_gpg_key_from_file(*name*, *filename*)

Import a GPG key with contents from the given filename.

install_packages(*query*, *packages*, *deliver_after=None*, *deliver_delay_window=0*)

Install packages on selected computers.

Parameters

- **query** (*unicode*) – A qualified criteria to be used in the search.
- **packages** (*list (of unicode)*) – A list of package names on which to operate. Multiple package names can be supplied.
- **deliver_after** (*unicode*) – A time in the future to perform the package operation.
- **deliver_delay_window** (*integer*) – Randomise delivery within the given time frame specified in minutes

invite_administrator(*name*, *email*, *roles=['GlobalAdmin']*)

Invite an administrator to your account.

Parameters

- **name** (*unicode line*) – The name of the person to invite.
- **email** (*unicode line*) – The email address of the administrator, to which the invitation will be send.
- **roles** (*list (of unicode)*) – If specified, the roles that the administrator is going to have in your account. Default to GlobalAdmin

list_pocket(*name*, *series*, *distribution*)

Return a list of the packages in a pocket.

Parameters

- **name** (*unicode*) – The name of the pocket.
- **series** (*unicode*) – The name of the series.
- **distribution** (*unicode*) – The name of the distribution.

modify_package_profile(*name*, *title=None*, *add_constraints=[]*, *remove_constraints=[]*)

THIS FUNCTION IS DEPRECATED. please use edit-package-profile instead. Add or remove constraints related to a package profile. Constraints can be dependencies or conflicts.

Parameters

- **name** (*unicode*) – The name of the package profile.
- **title** (*unicode title*) – The new title of the package profile.
- **add_constraints** (*list (of unicode)*) – List of constraints specifications to add in the form of "depends packagename" or "conflicts packagename < 1.0".
- **remove_constraints** (*list (of unicode)*) – List of constraints specifications to remove in the form of "depends packagename" or "conflicts packagename < 1.0".

```
overridden_apis = {'ImportGPGKey': {'doc': None, 'method':
'import_gpg_key_from_file', 'replace_args': {'material': {'doc': 'The filename of
the GPG file.', 'name': 'filename', 'type': 'unicode'}}}}
```

pull_packages_to_pocket(*name*, *series*, *distribution*)

Import packages to a pocket in pull mode from its parent pocket.

Parameters

- **name** (*unicode*) – The name of the pocket to pull packages to.
- **series** (*unicode*) – The name of the series.
- **distribution** (*unicode*) – The name of the distribution.

reboot_computers(*computer_ids*, *deliver_after=None*)

Reboot a list of computers.

Parameters

- **computer_ids** (*list (of integer)*) – A list of computer ids to reboot.
- **deliver_after** (*unicode*) – A time in the future to deliver the script.

register_juju_environment(*name*, *endpoint*, *username*, *password*)

Register a Juju environment.

Parameters

- **name** (*unicode*) – The name of the environment.
- **endpoint** (*unicode*) – The endpoint address of the Juju API.
- **username** (*unicode*) – The username used to authenticate with the Juju API.
- **password** (*unicode*) – The password used to authenticate with the Juju API.

register_juju_model(*name*, *endpoint*, *username*, *password*, *uuid*)

Register a Juju model.

Parameters

- **name** (*unicode*) – The name of the model.
- **endpoint** (*unicode*) – The endpoint address of the Juju API.
- **username** (*unicode*) – The username used to authenticate with the Juju API.
- **password** (*unicode*) – The password used to authenticate with the Juju API.
- **uuid** (*unicode*) – The UUID of the model to register from the controller.

reject_pending_computers(*computer_ids*)

Reject a list of pending computers associated with the account used for authentication.

Parameters

computer_ids (*list (of integer)*) – A list of computer IDs to reject.

remove_access_group(*name*)

Remove an access group.

Parameters

name (*unicode*) – The name of the access group to remove.

remove_access_groups_from_role(*name*, *access_groups*)

Remove the given access groups to a role.

Parameters

- **name** (*unicode*) – The name of the role to modify.
- **access_groups** (*list (of unicode)*) – A list of names of access groups to remove from the role.

remove_annotation_from_computers(*query*, *key*)

Remove annotation key from a selection of computers.

Parameters

- **query** (*unicode*) – A query string used to select the computers from which to remove annotation.
- **key** (*unicode*) – Annotation key to disassociate.

remove_apt_source(*name*)

Remove apt source from the account.

Parameters

name (*unicode*) – The name of the apt source to be removed.

remove_apt_source_from_repository_profile(*name*, *apt_source*)

Remove APT source from a repository profile. An activity will be created to remove the source from the computers associated with the given profile.

Parameters

- **name** (*unicode*) – Name of the repository profile.
- **apt_source** (*unicode*) – The name of the APT source to remove.

remove_apt_sources(*names*)

Deprecated: use RemoveAPTSource instead.

Parameters

names (*list (of unicode)*) – List of names of the APT sources be removed. Multiple names can be supplied.

remove_apt_sources_from_repository_profile(*name*, *apt_sources*)

Deprecated: use RemoveAPTSourceFromRepositoryProfile instead.

Parameters

- **name** (*unicode*) – Name of the repository profile.
- **apt_sources** (*list (of unicode)*) – The names of the APT sources to remove.

remove_computers(*computer_ids*)

Remove a list of computers by ID.

Parameters

computer_ids (*list (of integer)*) – A list of computer ids to remove.

remove_distribution(*name*)

Remove the specified repository distribution.

Parameters

name (*unicode*) – The name of the distribution to remove.

remove_gpg_key(*name*)

Remove a GPG key.

Parameters

name (*unicode*) – Name of the GPG key to remove.

remove_juju_environment(*name*)

Remove a Juju environment from the account.

Parameters

name (*unicode*) – Name of the environment to remove.

remove_juju_model(*name*)

Remove a Juju model from the account.

Parameters

name (*unicode*) – Name of the model to remove.

remove_package_filters_from_pocket(*name*, *series*, *distribution*, *packages*)

Remove package filters from a repository pocket. The pocket must be in pull mode and support blacklist/whitelist filtering.

Parameters

- **name** (*unicode*) – The name of the pocket to operate on.

- **series** (*unicode*) – The name of the series containing the pocket.
- **distribution** (*unicode*) – The name of the distribution containing the series.
- **packages** (*list (of unicode)*) – A list of names of packages to be added or removed from the pocket filter.

remove_package_profile(*name*)

Remove a package profile, given its name.

Parameters

name (*unicode*) – The name of the package profile to remove.

remove_packages(*query*, *packages*, *deliver_after=None*, *deliver_delay_window=0*)

Remove packages on selected computers.

Parameters

- **query** (*unicode*) – A qualified criteria to be used in the search.
- **packages** (*list (of unicode)*) – A list of package names on which to operate. Multiple package names can be supplied.
- **deliver_after** (*unicode*) – A time in the future to perform the package operation.
- **deliver_delay_window** (*integer*) – Randomise delivery within the given time frame specified in minutes

remove_packages_from_pocket(*name*, *series*, *distribution*, *packages*)

Remove packages from pockets in upload mode.

Parameters

- **name** (*unicode*) – The name of the pocket to remove packages from.
- **series** (*unicode*) – The name of the series containing the pocket.
- **distribution** (*unicode*) – The name of the distribution containing the series.
- **packages** (*list (of unicode)*) – A list of names of packages to be removed from the pockets.

remove_permissions_from_role(*name*, *permissions*)

Remove permissions from a role.

Parameters

- **name** (*unicode*) – The name of the role to modify.
- **permissions** (*list (of unicode)*) – A list of permissions to remove.

remove_persons_from_role(*name*, *persons*)

Remove people from a role.

Parameters

- **name** (*unicode*) – The name of the role to modify.
- **persons** (*list (of unicode)*) – A list of email addresses of people to remove.

remove_pocket(*name*, *series*, *distribution*)

Remove a repository pocket from a series in a distribution.

Parameters

- **name** (*unicode*) – The name of the pocket to remove.

- **series** (*unicode*) – The name of the series containing the pocket.
- **distribution** (*unicode*) – The name of the distribution containing the series.

remove_pockets_from_repository_profile(*name*, *pockets*, *series*, *distribution*)

Remove repository pockets from a repository profile. An activity will be created to remove the pockets from the APT sources of the computers associated with the given profile.

Parameters

- **name** (*unicode*) – Name of the repository profile.
- **pockets** (*list (of unicode)*) – The names of the pockets to remove.
- **series** (*unicode*) – The name of the series the pocket belongs to.
- **distribution** (*unicode*) – The name of the distribution the series belongs to.

remove_removal_profile(*name*)

Remove an existing removal profile by name.

Parameters
name (*unicode*) – The name of the removal profile you wish toremove.

remove_repository_profile(*name*)

Remove repository profile from the account.

Parameters
name (*unicode*) – The name of the repository profile to be removed.

remove_repository_profiles(*names*)

Deprecated: use RemoveRepositoryProfile instead.

Parameters
names (*list (of unicode)*) – Names of the repository profiles to be removed.

remove_role(*name*)

Remove an access role.

Parameters
name (*unicode*) – The name of the role.

remove_saved_search(*name*)

Remove a saved search associated with the current account.

Parameters
name (*unicode*) – The "slug" name for this saved search.

remove_script(*script_id*)

Remove scripts.

Parameters
script_id (*integer*) – The identity of the script to remove.

remove_script_attachment(*script_id*, *filename*)

Remove a script attachment.

Parameters

- **script_id** (*integer*) – The identity of the script to remove.
- **filename** (*unicode*) – The filename of the attachment to remove.

remove_series(*name*, *distribution*)

Remove a repository series from a distribution.

Parameters

- **name** (*unicode*) – The name of the series to remove.
- **distribution** (*unicode*) – The name of the distribution.

remove_tags_from_computers(*query*, *tags*)

Remove tags from a selection of computers.

Parameters

- **query** (*unicode*) – A query string used to select the computers to remove tags from.
- **tags** (*list (of unicode)*) – Tag names to be removed.

remove_upgrade_profile(*name*)

Remove an existing upgrade profile by name.

Parameters

name (*unicode*) – The name of the upgrade profile you wish to cancel.

remove_uploader_gpg_keys_from_pocket(*name*, *series*, *distribution*, *gpg_keys*)

Remove GPG keys for uploaded packages validation from a repository pocket in upload mode.

Parameters

- **name** (*unicode*) – The name of the pocket on which to associate keys.
- **series** (*unicode*) – The name of the series containing the pocket.
- **distribution** (*unicode*) – The name of the distribution containing the series.
- **gpg_keys** (*list (of unicode)*) – A list of GPG keys on which to operate.

rename_computers(*computer_titles*)

Rename a set of computers.

Parameters

computer_titles (*mapping*) – mapping of computer_ids to computer titles

run_query(*action_name*, *arguments*)

Make a low-level query against the Landscape API, using details provided in the *API* constructor.

set_settings(*key_values*)

Set configuration settings for the current LDS installation.

Parameters

key_values (*list (of unicode)*) – Key/value pairs to set, separated by '='. 'true' and 'false' strings will be interpreted as booleans.

shutdown_computers(*computer_ids*, *deliver_after=None*)

Shutdown a list of computers.

Parameters

- **computer_ids** (*list (of integer)*) – A list of computer ids to shutdown.
- **deliver_after** (*unicode*) – A time in the future to deliver the script.

ssh(*query*, *user=None*)

Calls *get_computers*, and then the ssh command with the given result.

subscribe_to_alert(*alert_type*)

Subscribe to an alert.

Parameters
alert_type (*unicode*) – The alert type to add a subscription to.

sync_mirror_pocket(*name*, *series*, *distribution*)

Synchronize a mirror repository pocket.

Parameters

- **name** (*unicode*) – The name of the pocket to synchronize.
- **series** (*unicode*) – The name of the series.
- **distribution** (*unicode*) – The name of the distribution.

unsubscribe_from_alert(*alert_type*)

Unsubscribe from an alert.

Parameters
alert_type (*unicode*) – The alert type to remove a subscription from.

upgrade_packages(*query*, *packages=[]*, *security_only=False*, *deliver_after=None*, *deliver_delay_window=0*)

Request upgrading of all packages identified as being upgradable, on all computers selected by query.

Parameters

- **query** (*unicode*) – A qualified criteria to be used in the search.
- **packages** (*list (of unicode)*) – List of package names on which to perform an upgrade. Multiple package names can be supplied like packages.1=foo and packages.2=bar.
- **security_only** (*boolean*) – If 'true' then only packages with USNs, i.e. security upgrades will be applied.
- **deliver_after** (*unicode*) – A time in the future to perform the package upgrade.
- **deliver_delay_window** (*integer*) – Randomise delivery within the given time frame specified in minutes

version = '2011-08-01'

landscape_api.base.**run_query**(*access_key*, *secret_key*, *action*, *params*, *uri*, *ssl_ca_file=True*, *version='2011-08-01'*)

Make a low-level query against the Landscape API.

Parameters

- **access_key** – The user access key.
- **secret_key** – The user secret key.
- **action** – The type of methods to call. For example, "GetComputers".
- **params** – A dictionary of the parameters to pass to the action.
- **uri** – The root URI of the API service. For example, "https://landscape.canonical.com/".
- **ssl_ca_file** – Path to the server's SSL Certificate Authority certificate. For example, "~/landscape_server_ca.crt".

4.1.3 Module contents

Top-level package for Landscape API (Python 3).

CHAPTER

FIVE

CONTRIBUTING

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at https://github.com/jurya/landscape_api_py3/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

5.1.4 Write Documentation

Landscape API (Python 3) could always use more documentation, whether as part of the official Landscape API (Python 3) docs, in docstrings, or even on the web in blog posts, articles, and such.

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/jurya/landscape_api_py3/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

5.2 Get Started!

Ready to contribute? Here's how to set up *landscape_api_py3* for local development.

1. Fork the *landscape_api_py3* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/landscape_api_py3.git
```

3. Install your local copy into a virtualenv (using pipenv). Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ pipenv install --dev --pre
```

4. Setup pre-commit and pre-push hooks:

```
$ pipenv run pre-commit install -t pre-commit
$ pipenv run pre-commit install -t pre-push
```

5. Activate virtual environment:

```
$ pipenv shell
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 landscape_api_py3 tests
$ pytest
$ tox
```

   flake8, pytest, tox (and other tools) are installed automatically by pipenv (step 3).

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.8, and for PyPy. Check https://travis-ci.com/jurya/landscape_api_py3/pull_requests and make sure that the tests pass for all supported Python versions.

5.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_landscape_api_py3
```

5.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bump2version patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

CHAPTER

SIX

CREDITS

6.1 Development Lead

- Jiří Altman <altman.jiri@gmail.com>

6.2 Contributors

None yet. Why not be the first?

CHAPTER

SEVEN

HISTORY

7.1 v0.7.1 (2021-01-27)

- fixed documentation

7.2 v0.7.0 (2021-01-27)

- fix #245: CreateScript endpoint raises exception (solution suggested and tested)
- fix #246: CreateScriptAttachment endpoint raises exception (solution suggested and tested)
- minor format change
- add pyproject.toml

7.3 v0.6.1 (2020-06-17)

- fix #19
- minor updates

7.4 v0.5.0 (2020-06-11)

- added support for Python 3.5, 3.6, 3.7, 3.8 and above
- fixed minor bugs
- updated documentation (Installation, Usage, Quick start)

7.5 v0.4.2 (2020-06-10)

- fixed documentation import bug
- fixed default CA cert bug

7.6 v0.4.1 (2020-06-10)

- fixed bug with imports - now it's compatible with Canonical **landscape-api**

7.7 v0.3.5 (2020-06-10)

- minor fixes

7.8 v0.3.4 (2020-06-09)

- replaced **pycurl** –> **requests**

7.9 v0.2.0alpha (2020-06-08) - First pre-release of landscape-api package

- first **ALPHA non-production** version release of **landscape-api** ported to **Python v3.8**

7.9.1 Known issues (v0.2.0alpha):

- on **Windows** download CA certificate file from https://curl.haxx.se/ca/cacert.pem and use **–ssl-ca-file** or **LANDSCAPE_API_SSL_CA_FILE** (see Landscape API documentation here)
- on **Linux** depends on **gnutls** and **libssl** (require **pycurl** package for installation)

7.9.2 Before installation of the package (v0.2.0alpha):

- on **Ubuntu 16.04** (Xenial Xerus) use **sudo apt-get install -y libgnutls-dev**
- on **Ubuntu 20.04** (Focal Fossa) use **sudo apt-get install -y libgnutls28-dev libcurl4-openssl-dev libssl-dev**
- on **Windows 10** simply use **pipx install landscape_api_py3**

7.10 0.1.3 (2020-06-07)

- first release on PyPI

CHAPTER

EIGHT

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

l

landscape_api, 32
landscape_api.base, 7

INDEX

A

accept_pending_computers() (*landscape_api.base.API method*), 7
add_access_groups_to_role() (*landscape_api.base.API method*), 7
add_annotation_to_computers() (*landscape_api.base.API method*), 7
add_apt_sources_to_repository_profile() (*landscape_api.base.API method*), 7
add_package_filters_to_pocket() (*landscape_api.base.API method*), 8
add_permissions_to_role() (*landscape_api.base.API method*), 8
add_persons_to_role() (*landscape_api.base.API method*), 8
add_pockets_to_repository_profile() (*landscape_api.base.API method*), 8
add_tags_to_computers() (*landscape_api.base.API method*), 8
add_uploader_gpg_keys_to_pocket() (*landscape_api.base.API method*), 8
API (*class in landscape_api.base*), 7
approve_activities() (*landscape_api.base.API method*), 9
associate_alert() (*landscape_api.base.API method*), 9
associate_package_profile() (*landscape_api.base.API method*), 9
associate_removal_profile() (*landscape_api.base.API method*), 9
associate_repository_profile() (*landscape_api.base.API method*), 9
associate_upgrade_profile() (*landscape_api.base.API method*), 10

C

call() (*landscape_api.base.API method*), 10
call_arbitrary() (*landscape_api.base.API method*), 10
cancel_activities() (*landscape_api.base.API method*), 10
change_computers_access_group() (*landscape_api.base.API method*), 10
copy_package_profile() (*landscape_api.base.API method*), 10
copy_role() (*landscape_api.base.API method*), 11
copy_script() (*landscape_api.base.API method*), 11
create_access_group() (*landscape_api.base.API method*), 11
create_apt_source() (*landscape_api.base.API method*), 11
create_cloud_otps() (*landscape_api.base.API method*), 11
create_distribution() (*landscape_api.base.API method*), 12
create_package_profile() (*landscape_api.base.API method*), 12
create_pocket() (*landscape_api.base.API method*), 12
create_removal_profile() (*landscape_api.base.API method*), 13
create_repository_profile() (*landscape_api.base.API method*), 13
create_role() (*landscape_api.base.API method*), 13
create_saved_search() (*landscape_api.base.API method*), 13
create_script() (*landscape_api.base.API method*), 14
create_script_attachment() (*landscape_api.base.API method*), 14
create_series() (*landscape_api.base.API method*), 14
create_upgrade_profile() (*landscape_api.base.API method*), 15

D

derive_series() (*landscape_api.base.API method*), 15
diff_pull_pocket() (*landscape_api.base.API method*), 15
disable_administrator() (*landscape_api.base.API method*), 16
disassociate_alert() (*landscape_api.base.API*

method), 16
disassociate_package_profile() (*landscape_api.base.API method*), 16
disassociate_removal_profile() (*landscape_api.base.API method*), 16
disassociate_repository_profile() (*landscape_api.base.API method*), 16
disassociate_upgrade_profile() (*landscape_api.base.API method*), 17

E

edit_package_profile() (*landscape_api.base.API method*), 17
edit_pocket() (*landscape_api.base.API method*), 17
edit_removal_profile() (*landscape_api.base.API method*), 18
edit_repository_profile() (*landscape_api.base.API method*), 18
edit_saved_search() (*landscape_api.base.API method*), 18
edit_script() (*landscape_api.base.API method*), 18
edit_upgrade_profile() (*landscape_api.base.API method*), 18
execute_script() (*landscape_api.base.API method*), 19
extra_actions (*landscape_api.base.API attribute*), 19

G

get_access_groups() (*landscape_api.base.API method*), 19
get_activities() (*landscape_api.base.API method*), 19
get_activity_types() (*landscape_api.base.API method*), 20
get_administrators() (*landscape_api.base.API method*), 20
get_alert_subscribers() (*landscape_api.base.API method*), 20
get_alerts() (*landscape_api.base.API method*), 20
get_apt_sources() (*landscape_api.base.API method*), 20
get_computers() (*landscape_api.base.API method*), 20
get_computers_not_upgraded() (*landscape_api.base.API method*), 20
get_csv_compliance_data() (*landscape_api.base.API method*), 21
get_distributions() (*landscape_api.base.API method*), 21
get_event_log() (*landscape_api.base.API method*), 21
get_gpg_keys() (*landscape_api.base.API method*), 21
get_juju_environments() (*landscape_api.base.API method*), 21
get_juju_models() (*landscape_api.base.API method*), 21
get_not_pinging_computers() (*landscape_api.base.API method*), 21
get_package_profiles() (*landscape_api.base.API method*), 22
get_packages() (*landscape_api.base.API method*), 22
get_pending_computers() (*landscape_api.base.API method*), 22
get_permissions() (*landscape_api.base.API method*), 23
get_removal_profiles() (*landscape_api.base.API method*), 23
get_repository_profiles() (*landscape_api.base.API method*), 23
get_roles() (*landscape_api.base.API method*), 23
get_saved_searches() (*landscape_api.base.API method*), 23
get_script_code() (*landscape_api.base.API method*), 23
get_scripts() (*landscape_api.base.API method*), 23
get_settings() (*landscape_api.base.API method*), 23
get_upgrade_profiles() (*landscape_api.base.API method*), 23
get_upgraded_computers_by_frequency() (*landscape_api.base.API method*), 23
get_usn_time_to_fix() (*landscape_api.base.API method*), 24

I

import_gpg_key() (*landscape_api.base.API method*), 24
import_gpg_key_from_file() (*landscape_api.base.API method*), 24
install_packages() (*landscape_api.base.API method*), 24
invite_administrator() (*landscape_api.base.API method*), 24

L

landscape_api
    module, 32
landscape_api.base
    module, 7
list_pocket() (*landscape_api.base.API method*), 25

M

modify_package_profile() (*landscape_api.base.API method*), 25
module
    landscape_api, 32
    landscape_api.base, 7

O

overridden_apis (*landscape_api.base.API attribute*), 25

P

pull_packages_to_pocket() (*landscape_api.base.API method*), 25

R

reboot_computers() (*landscape_api.base.API method*), 25
register_juju_environment() (*landscape_api.base.API method*), 25
register_juju_model() (*landscape_api.base.API method*), 26
reject_pending_computers() (*landscape_api.base.API method*), 26
remove_access_group() (*landscape_api.base.API method*), 26
remove_access_groups_from_role() (*landscape_api.base.API method*), 26
remove_annotation_from_computers() (*landscape_api.base.API method*), 26
remove_apt_source() (*landscape_api.base.API method*), 26
remove_apt_source_from_repository_profile() (*landscape_api.base.API method*), 26
remove_apt_sources() (*landscape_api.base.API method*), 27
remove_apt_sources_from_repository_profile() (*landscape_api.base.API method*), 27
remove_computers() (*landscape_api.base.API method*), 27
remove_distribution() (*landscape_api.base.API method*), 27
remove_gpg_key() (*landscape_api.base.API method*), 27
remove_juju_environment() (*landscape_api.base.API method*), 27
remove_juju_model() (*landscape_api.base.API method*), 27
remove_package_filters_from_pocket() (*landscape_api.base.API method*), 27
remove_package_profile() (*landscape_api.base.API method*), 28
remove_packages() (*landscape_api.base.API method*), 28
remove_packages_from_pocket() (*landscape_api.base.API method*), 28
remove_permissions_from_role() (*landscape_api.base.API method*), 28
remove_persons_from_role() (*landscape_api.base.API method*), 28
remove_pocket() (*landscape_api.base.API method*), 28
remove_pockets_from_repository_profile() (*landscape_api.base.API method*), 29
remove_removal_profile() (*landscape_api.base.API method*), 29
remove_repository_profile() (*landscape_api.base.API method*), 29
remove_repository_profiles() (*landscape_api.base.API method*), 29
remove_role() (*landscape_api.base.API method*), 29
remove_saved_search() (*landscape_api.base.API method*), 29
remove_script() (*landscape_api.base.API method*), 29
remove_script_attachment() (*landscape_api.base.API method*), 29
remove_series() (*landscape_api.base.API method*), 29
remove_tags_from_computers() (*landscape_api.base.API method*), 30
remove_upgrade_profile() (*landscape_api.base.API method*), 30
remove_uploader_gpg_keys_from_pocket() (*landscape_api.base.API method*), 30
rename_computers() (*landscape_api.base.API method*), 30
run_query() (*in module landscape_api.base*), 31
run_query() (*landscape_api.base.API method*), 30

S

set_settings() (*landscape_api.base.API method*), 30
shutdown_computers() (*landscape_api.base.API method*), 30
ssh() (*landscape_api.base.API method*), 30
subscribe_to_alert() (*landscape_api.base.API method*), 30
sync_mirror_pocket() (*landscape_api.base.API method*), 31

U

unsubscribe_from_alert() (*landscape_api.base.API method*), 31
upgrade_packages() (*landscape_api.base.API method*), 31

V

version (*landscape_api.base.API attribute*), 31