OCI Cloud Guardrails - EN

Oracle OCI Cloud Guardrails

Ludovic Dessemon Enterprise Cloud Strategist at Oracle

Description

Prowler is a command line tool for OCI Cloud Guardrails, originally developed by Toni de la Fuente (https://github.com/toniblyx/prowler)

OCI Prowler is an adaptation for Oracle Cloud Infrastructure, developed by Ludovic Dessemon, Enterprise Cloud Strategist (Oracle Canada) - July 2020

Requirements and Installation

This script has been written in bash and Python using the Python SDK for Oracle Cloud Infrastructure, Oracle Identity Cloud Service REST APIs, and OCI CLI.

Oracle Identity Cloud Service REST APIs

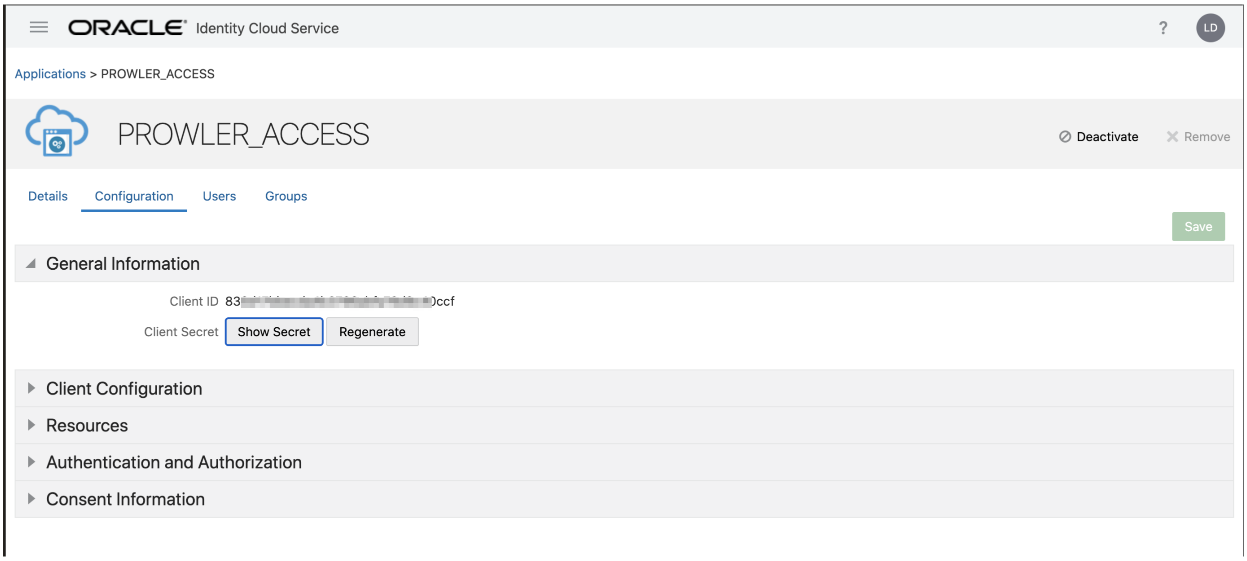
The Oracle Identity Cloud Service REST APIs support SCIM 2.0 compliant endpoints with standard SCIM 2.0 core schemas and Oracle schema extensions to programmatically manage users, groups, applications, and identity functions, such as password management and administrative tasks. To make REST API calls to your Oracle Identity Cloud Service environment, you need an OAuth2 access token to use for authorization. The access token provides a session (with scope and expiration), that your client application can use to perform tasks in Oracle Identity Cloud Service.

The following sections walk you through the steps required to use the OCI Prowler with Oracle Identity Cloud Service REST APIs:

Register a Confidential Application in Oracle Identity Cloud Service

- Access your Oracle Identity Cloud Service instance (for example: https://tenant-base-url/ui/v1/adminconsole) and log in with your Identity Domain Administrator credentials.
- Click **Applications**, and then **Add**
- Select **Confidential Application** as the type of application.
- Enter an **application name** (eg. ProwlerAccess) and a description, and then click **Next**.
- On the Authorization page, define the following items:
  - Select the **Configure this application as a client now** option.
  - Select **Client Credentials** from the **Allowed Grant Type** section.
  - At the bottom of the page select the **Grant the client access to Identity Cloud Service Admin APIs** check box.
  - Click your cursor in the box that appears below the check box, and then select the type of access for the REST APIs. For example, select **Identity Domain Administrator.** Your credentials and all tasks available to the Identity Domain Administrator will be accessible to you.
  - Click **Next**, and then **Finish**.

    - Make note (using your preferred note utility) of the Client ID and the Client Secret that appear in the confirmation window, and then click **Close**.
- Click **Activate** in the upper-right section of the page to activate the application.



Client ID [redacted]

Client Secret

Oracle OCI CLI

Overview

The CLI is a small-footprint tool that you can use on its own or with the Console to complete Oracle Cloud Infrastructure tasks. The CLI provides the same core functionality as the Console, plus additional commands. Some of these, such as the ability to run scripts, extend Console functionality.
This script uses OCI CLI.

Installation

Linux

- Open a terminal.
- To run the installer script, run the following command.

```
bash -c "$(curl -L https://raw.githubusercontent.com/oracle/oci-cli/master/scripts/install/install.sh)"
```

- Respond to the Installation Script Prompts.

If you’re using Oracle Linux 7, you can use yum to install the CLI.

```
sudo yum install python36-oci-cli
```

Mac OS X

You can use Homebrew to install, upgrade, and uninstall the CLI on Mac OS.

To install the CLI on Mac OS X with Homebrew:

```
brew update && brew install oci-cli
```

To upgrade your CLI install on Mac OS X using Homebrew:

```
brew update && brew upgrade oci-cli
```

To uninstall the CLI on Mac OS X using Homebrew:

```
brew uninstall oci-cli
```

Windows

- Open the PowerShell console using the Run as Administrator option.
- The installer enables auto-complete by installing and running a script. To allow this script to run, you must enable the RemoteSigned execution policy.
  To configure the remote execution policy for PowerShell, run the following command.

```
Set-ExecutionPolicy RemoteSigned
```

- Download the installer script:

```
Invoke-WebRequest https://raw.githubusercontent.com/oracle/oci-cli/master/scripts/install/install.ps1 -OutFile install.ps1
```

- Run the installer script with or without prompts :

```
iex ((New-ObjectSystem.Net.WebClient).DownloadString('https://raw.githubusercontent.com/oracle/oci-cli/master/scripts/install/install.ps1'))
```

Installation Script Prompts

The installation script prompts you for the following information.

- If you do not have a compatible version of Python installed:
  - Windows and Linux: You are prompted to provide a location for installing the binaries and executables. The script will install Python for you.

    - MacOS: You are notified that your version of Python is incompatible. You must upgrade before you can proceed with the installation. The script will not install Python for you.
- When prompted to upgrade the CLI to the newest version, respond with Y to overwrite an existing installation.
- When prompted to update your PATH, respond with Y to be able to invoke the CLI without providing the full path to the executable. This will add oci.exe to your PATH.

Setting up the Config File

Before using the CLI, you must create a config file that contains the required credentials for working with Oracle Cloud Infrastructure. You can create this file using a setup dialog or manually using a text editor.

```
vi ~/.oci/config
```

To have the CLI walk you through the first-time setup process, use the oci setup config command. The command prompts you for the information required for the config file and the API public/private keys. The setup dialog generates an API key pair and creates the config file.

If you want to set up the API public/private keys yourself and write your own config file, see SDK and Tool Configuration.

OCI SDK for Python

It is highly recommended that a Python virtual environment be used when installing oci.

Please consult the Installing packages using pip and virtualenv guide from the Python Software Foundation for more information about virtual environments.

See the installation guide for installation troubleshooting and alternative install methods.

Once your virtual environment is active, oci can be installed using pip.

```
pip install oci
```

Usage

1. Run the `prowler` command without options (it will use your environment variable credentials if they exist or will default to using the `~/.oci/config` file and run checks):

   ```
   ./prowler
   ```

   Use `-l` to list all available checks and the groups (sections) that reference them

2. For a single check use option `-c`:

   ```
   ./prowler -c check310
   ```

   or multiple checks separated by comma:

   ```
   ./prowler -c check310,check722
   ```

   or all checks but some of them:

   ```
   ./prowler -E check42,check43
   ```

   or for custom profile:

   ```
   ./prowler -p custom-profile -c check11
   ```

   or for a group of checks use group name:

   ```
   ./prowler -g group1
   ```

   or exclude some checks in the group:

   ```
   ./prowler -g group4 -E check42,check43
   ```

Save your reports

1. If you want to save your report for later analysis thare are different ways, natively (supported text, mono, csv, json, json-asff and junit-xml see note below for more info):

   ```
   ./prowler -M csv
   ```

   or with multiple formats at the same time:

   ```
   ./prowler -M csv,json,json-asff
   ```

   or just a group of checks in multiple formats:

   ```
   ./prowler -g gdpr -M csv,json,json-asff
   ```

   Now `-M` creates a file inside the prowler root directory named `ociprowler-output--YYYYMMDDHHMMSS.format`. You don't have to specify anything else, no pipes, no redirects.

   or just saving the output to a file like below:

   ```
   ./prowler -M mono > prowler-report.txt
   ```

or if you want a coloured HTML report do:

```
pip install ansi2html
./prowler | ansi2html -la > report.html
```

To generate JUnit report files, include the junit-xml format. This can be combined with any other format. Files are written inside a prowler root directory named `junit-reports`:

```
./prowler -M text,junit-xml
```

> Note about output formats to use with `-M`: "text" is the default one with colors, "mono" is like default one but monochrome, "csv" is comma separated values, "json" plain basic json (without comma between lines) .

2. If you want to run Prowler to check multiple OCI accounts in parallel (runs up to 4 simultaneously `-P 4`):

```
grep -E '^\[([0-9A-Aa-z_-]+)\]'  ~/.oci/config | tr -d '][' | shuf | \
xargs -n 1 -L 1 -I @ -r -P 4 ./prowler -p @ -M csv  2> /dev/null  >> all-accounts.csv
```

3. For help use:

```
./prowler -h
```

Advanced Usage

Custom folder for custom checks

Flag `-x /my/own/checks` will include any check in that particular directory.

Show or log only FAILs

In order to remove noise and get only FAIL findings there is a `-q` flag that makes Prowler to show and log only FAILs. It can be combined with any other option.

```
./prowler -q -M csv -b
```

Run Prowler

export OCI_CONFIG=/Users/ludovicdessemon/.oci/config
export PATH=**OCI_CLI_Path**/bin:$PATH

Add Custom Checks

In order to add any new check feel free to create a new extra check in the extras group or other group. To do so, you will need to follow these steps:

1. Follow structure in file `checks/check_sample`
2. Name your check with a number part of an existing group or a new one
3. Save changes and run it as `./prowler -c extraNN`

Add Custom Groups

1. Follow structure in file `groups/groupN_sample`
2. Name your group with a non existing number
3. Save changes and run it as `./prowler -g extraNN`
    - You can also create a group with only the checks that you want to perform in your company, for instance a group named `group9_mycompany` with only the list of checks that you care or your particular compliance applies.

License

Any piece of code is licensed as Apache License 2.0 as specified in each file. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0