Ubuntu Pro for WSL

Canonical Ltd.

Oct 29, 2024

CONTENTS

Note

This documentation describes a future release of UP4W. UP4W is not yet generally available in the Microsoft Store.

Ubuntu Pro for WSL (UP4W) is a powerful automation tool for managing WSL on Windows. If you are responsible for a fleet of Windows devices, UP4W will enable you to monitor, customise and secure Ubuntu WSL instances at scale.

UP4W is designed to achieve close integration with applications for customising images, enforcing standards and validating security compliance. WSL instances can be created, removed and monitored with Landscape. Microsoft Defender is WSL-aware, making it easy to confirm if instances are compliant. Cloud-init support is built-in, allowing efficient customisation of standard images.

Once you have an Ubuntu Pro subscription, adding your Pro token to UP4W on a Windows host will add that token to all connected WSL instances with the Ubuntu Pro client installed. When the Landscape client is installed on the host, any connected WSL instances will be auto-enrolled in Landscape. WSL instances can then be remotely created, provisioned and managed from the Windows host.

WSL is preferred by many organisations as a solution to run a fully-functioning Linux environment on a Windows machine. UP4W empowers system administrators and corporate security teams to manage large numbers of WSL instances effectively.

Read our *getting started tutorial* to begin.

CHAPTER

ONE

IN THIS DOCUMENTATION

Tutorials	**Start here** with hands-on tutorials for new users, guiding you through your first-steps
How-to guides	**Follow step-by-step** instructions for key operations and common tasks
Reference	**Read technical descriptions** of important factual information relating to UP4W
UP4W Dev	**Review guides and reference material** aimed at contributors

CHAPTER

TWO

PROJECT AND COMMUNITY

UP4W is a member of the Ubuntu family. It's an open-source project that warmly welcomes community contributions, suggestions, fixes and constructive feedback. Check out our contribution page on GitHub in order to bring ideas, report bugs, participate in discussions and much more!

Thinking about using UP4W for your next project? Get in touch!

2.1 Tutorials

These tutorials will help you learn to get the best out of UP4W's features.

2.1.1 Get started with UP4W

Windows Subsystem for Linux (WSL) makes it possible to run Ubuntu on a Windows machine. Ubuntu Pro for WSL (UP4W) ensures that each new Ubuntu WSL instance that you create will automatically attach to your Ubuntu Pro subscription.

In this tutorial you will learn how to install UP4W on Windows and verify that Ubuntu WSL instances are Pro-attaching. You should then be ready for more advanced usage scenarios.

What you will do

- Install UP4W from the Microsoft Store
- Configure UP4W with a Pro token
- Test automatic Pro-attachment of WSL instances

Warning

If you already have Ubuntu WSL pre-installed:

We recommend that any Ubuntu WSL installed is exported then deleted. You can then install it as described in this tutorial. At the end of the tutorial you can import and restore your data.

Read our *how-to guide on backup and restore*.

What you will need

- A Windows 10 or 11 machine with a minimum of 16GB RAM and 8-core processor
- Some familiarity with commands for the Linux shell and PowerShell

Note

WSL enables using a Linux shell and Windows PowerShell side-by-side on the same machine. In this tutorial, commands will be prefixed by a prompt that indicates the shell being used, for example:

- `PS C:\Users\me\tutorial>` is a PowerShell prompt where the current working directory is `C:\Users\me\tutorial`.
- `u@mib:~/tutorial$` indicates a Linux shell prompt login as user "u" where the current working directory is `/home/ubuntu/tutorial/`

Output logs are included in this tutorial when instructive but are sometimes omitted to save space.

Set up Ubuntu WSL

Install WSL

WSL can be installed directly from the Microsoft Store.

If you already have WSL installed, with `~\.wslconfig` on your system, you are advised to backup the file then remove it before continuing the tutorial.

To check if the file exists run:

```
PS C:\Users\me\tutorial> Test-Path -Path "~\.wslconfig"
```

If this returns `True` then the file exists and can be removed with:

```
PS C:\Users\me\tutorial> Remove-Item ~\.wslconfig
```

Install Ubuntu

Ubuntu 24.04 LTS is recommended for this tutorial and can be installed from the Microsoft Store:

Install Ubuntu 24.04 LTS from the Microsoft Store

For other installation options refer to our install Ubuntu on WSL2 guide.

At this point, running `ubuntu2404.exe` in PowerShell will launch an Ubuntu WSL instance and log in to its shell.

To manually associate that Ubuntu instance with a Pro subscription you could run the `sudo pro attach` command from within the Ubuntu instance.

This, however, would need to be repeated manually for each new instance. UP4W solves this scalability problem by automating Pro-attachment. Next, let's take a look at how that works in practice.

Set up Ubuntu Pro for WSL

Get an Ubuntu Pro token

An active Ubuntu Pro subscription provides you with a token that can be added to the Ubuntu Pro client on WSL instances.

Your subscription token can be retrieved from the Ubuntu Pro Dashboard.

Visit the Ubuntu Pro page if you need a new subscription. The `Myself` option for a personal subscription is free for up to 5 machines.

Once you have a token you are ready to install UP4W.

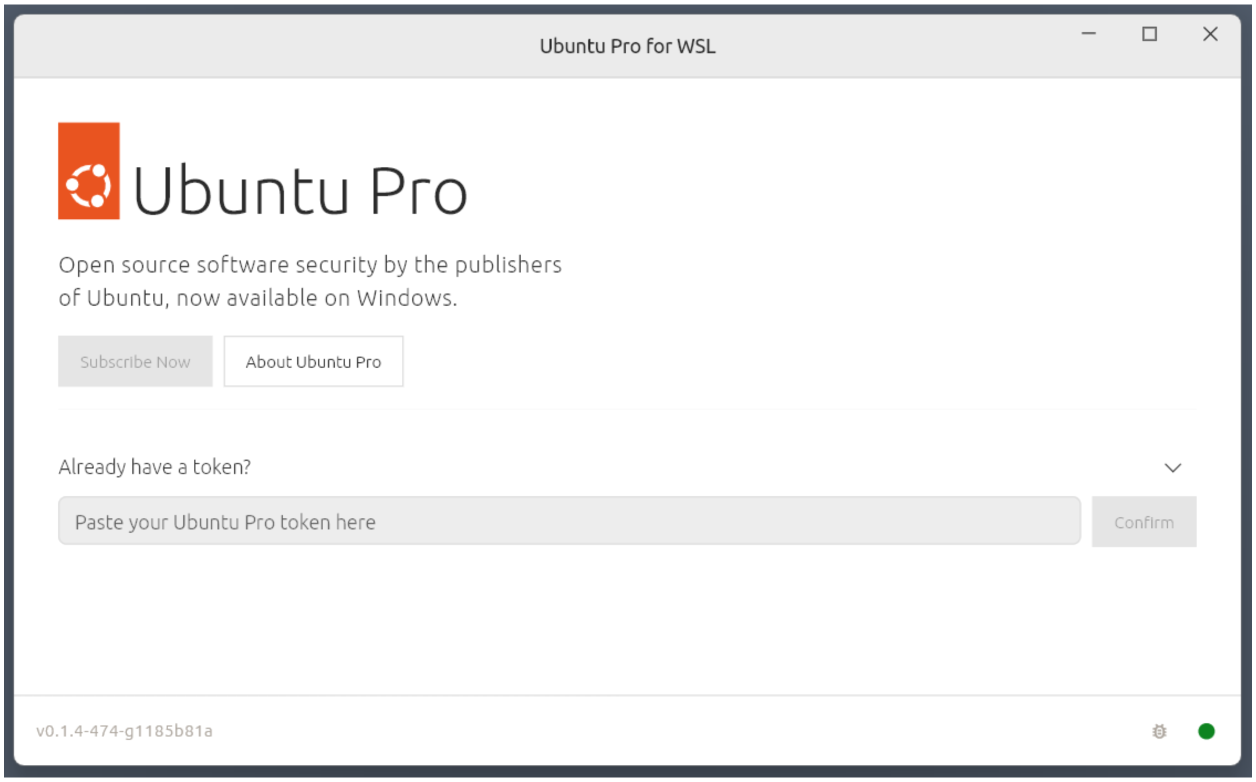
Install and configure UP4W

Warning

The install link below will work only if you're logged in to the Microsoft Store with an account for which access to the app has been enabled.

UP4W can be installed from this link to the Microsoft Store.

Open the application and paste the token you copied from the Ubuntu Pro dashboard:



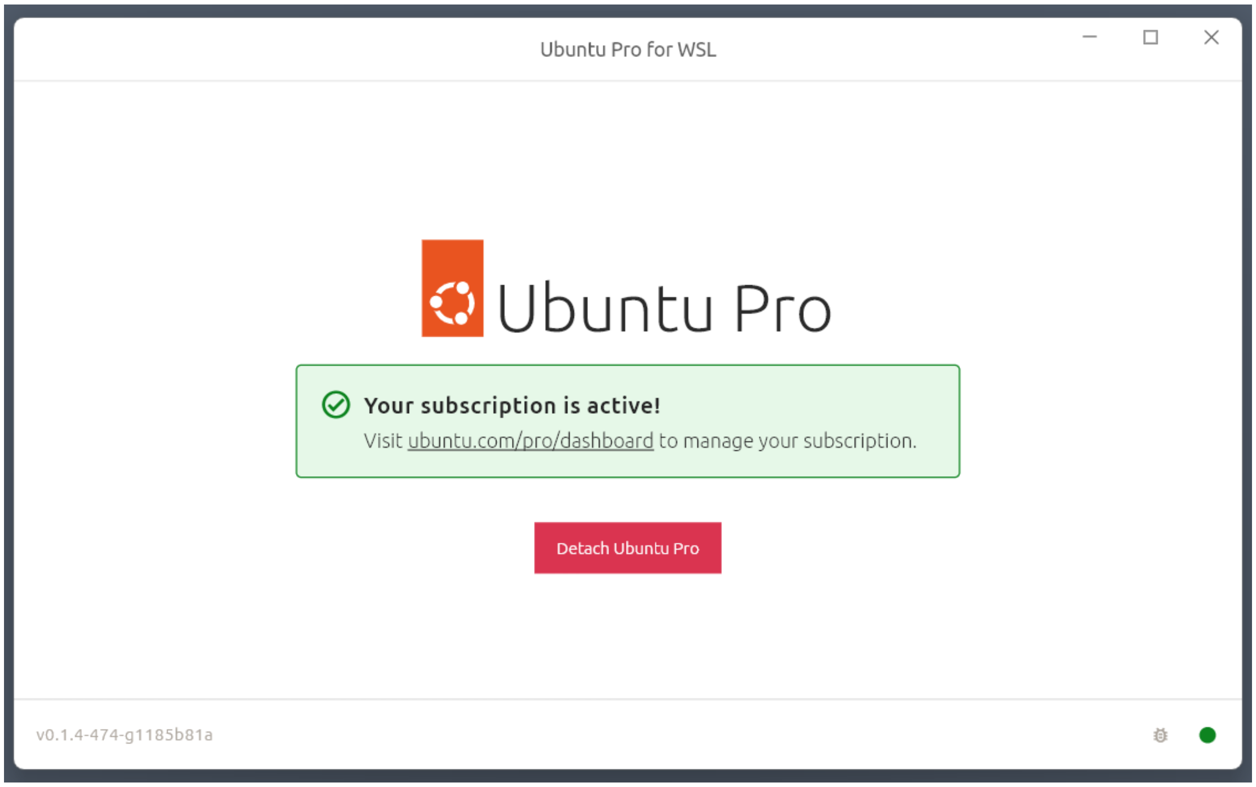
Open source software security by the publishers of Ubuntu, now available on Windows.

Already have a token?

Paste your Ubuntu Pro token here

v0.1.4-474-g1185b81a

After you confirm, a status screen will appear showing that configuration is complete:


Ubuntu Pro for WSL
Ubuntu Pro
Your subscription is active!
Visit ubuntu.com/pro/dashboard to manage your subscription.
Detach Ubuntu Pro
v0.1.4-474-g1185b81a

Done! You can close the UP4W window before continuing. If at any time you want to detach your Pro subscription just open the UP4W application and select **Detach Ubuntu Pro**.

Your Ubuntu Pro subscription is now attached to UP4W on the Windows host. UP4W will automatically forward the subscription to the Ubuntu Pro client on your Ubuntu WSL instances.

Verify Pro-attachment

All Ubuntu WSL instances will now be automatically added to your Ubuntu Pro subscription.

Open Windows PowerShell and run the following command to create a new Ubuntu 24.04 instance, entering a user and password when prompted. For quick testing, set both to u:

```
PS C:\Users\me\tutorial> ubuntu2404.exe
```

You will now be logged in to the new instance shell and can check that UP4W has Pro-attached this instance with:

```
u@mib:~$ pro status
```

The output should indicate that services like ESM are enabled, with account and subscription information also shown:

```
SERVICE          ENTITLED  STATUS       DESCRIPTION
esm-apps         yes       enabled      Expanded Security Maintenance for Applications
esm-infra        yes       enabled      Expanded Security Maintenance for Infrastructure

NOTICES
Operation in progress: pro attach

For a list of all Ubuntu Pro services, run 'pro status --all'
Enable services with: pro enable <service>
```

(continues on next page)

(continued from previous page)

```
     Account: me@ubuntu.com
Subscription: Ubuntu Pro - free personal subscription
```

Packages can also be accessed from all the enabled services. Running `sudo apt update` will produce output like the following:

```
Hit:1 http://archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ppa.launchpad.net/ubuntu-wsl-dev/ppa/ubuntu noble InRelease
Hit:3 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:4 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:5 http://ppa.launchpad.net/landscape/self-hosted-beta/ubuntu noble InRelease
Hit:6 https://esm.ubuntu.com/apps/ubuntu noble-apps-security InRelease
Hit:7 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:8 http://ppa.launchpad.net/cloud-init-dev/proposed/ubuntu noble InRelease
Hit:9 https://esm.ubuntu.com/infra/ubuntu noble-infra-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
```

Now let's check that another Ubuntu instance will also Pro-attach.

Install Ubuntu 22.04 LTS directly from PowerShell:

```
PS C:\Users\me\tutorial> wsl --install Ubuntu-22.04
```

Once you are in the instance shell, enter a username and password then run `pro status`. You should again get confirmation of successful Pro-attachment for the new instance.

> If you want to uninstall UP4W after this tutorial refer to *our how-to guide*.

Next steps

This is only the start of what you can do with UP4W.

If you need to create and manage large numbers of Ubuntu WSL instances you will probably want to use the Windows registry. By using the Windows registry you can associate a Pro token with each new WSL instance using your organisation's own deployment solution.

> For detailed step-by-step instructions on using the Windows registry read our short guide on how to *install and configure UP4W*.

Landscape support is also built-in to UP4W. With a single configuration file, you can create and manage multiple WSL instances that will automatically be registered with your Landscape server:

> For more information, please refer to our tutorial on how to *deploy WSL instances with UP4W and Landscape*.

Our documentation includes several other *how-to guides* for completing specific tasks, *reference* material describing key information relating to UP4W and dedicated *documentation for developers*.

2.1.2 Deployment with UP4W and Landscape

With Ubuntu Pro for WSL (UP4W) an Ubuntu Pro subscription empowers you to manage Ubuntu WSL instances at scale.

In this tutorial you will develop an understanding of how UP4W can help you deploy and manage Ubuntu WSL instance using Landscape.

What you will do

- Deploy an Ubuntu WSL instance locally
- Deploy an Ubuntu WSL instance remotely
- Test automatic configuration of WSL instances by UP4W

What you need

- A Windows 10 or 11 machine with a minimum of 16GB RAM and 8-core processor
- The latest version of Landscape Server set up and configured on a physical or virtual machine
- WSL and Ubuntu 24.04 installed on Windows
- An UP4W installation configured with a Pro token

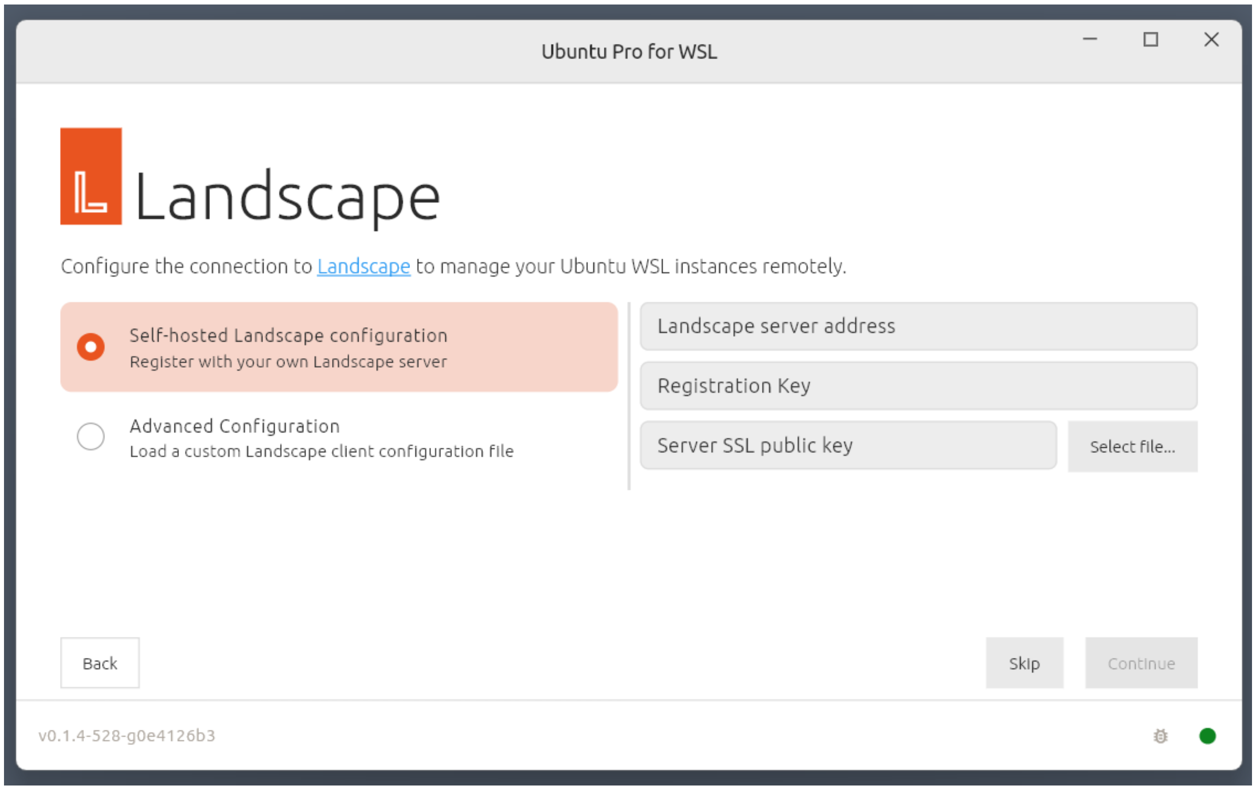
Before following this tutorial it is recommended that you complete the *getting started* tutorial to familiarise yourself with UP4W installation and configuration.

Set things up

To complete this tutorial you will need to have a Landscape server set up and you should be able access your Landscape dashboard in a browser. Please refer to the Landscape documentation for setup and configuration instructions.

Configure Landscape in the UP4W app

In the UP4W app, after entering your Pro token, navigate to the Landscape configuration screen:



Ubuntu Pro for WSL

Landscape

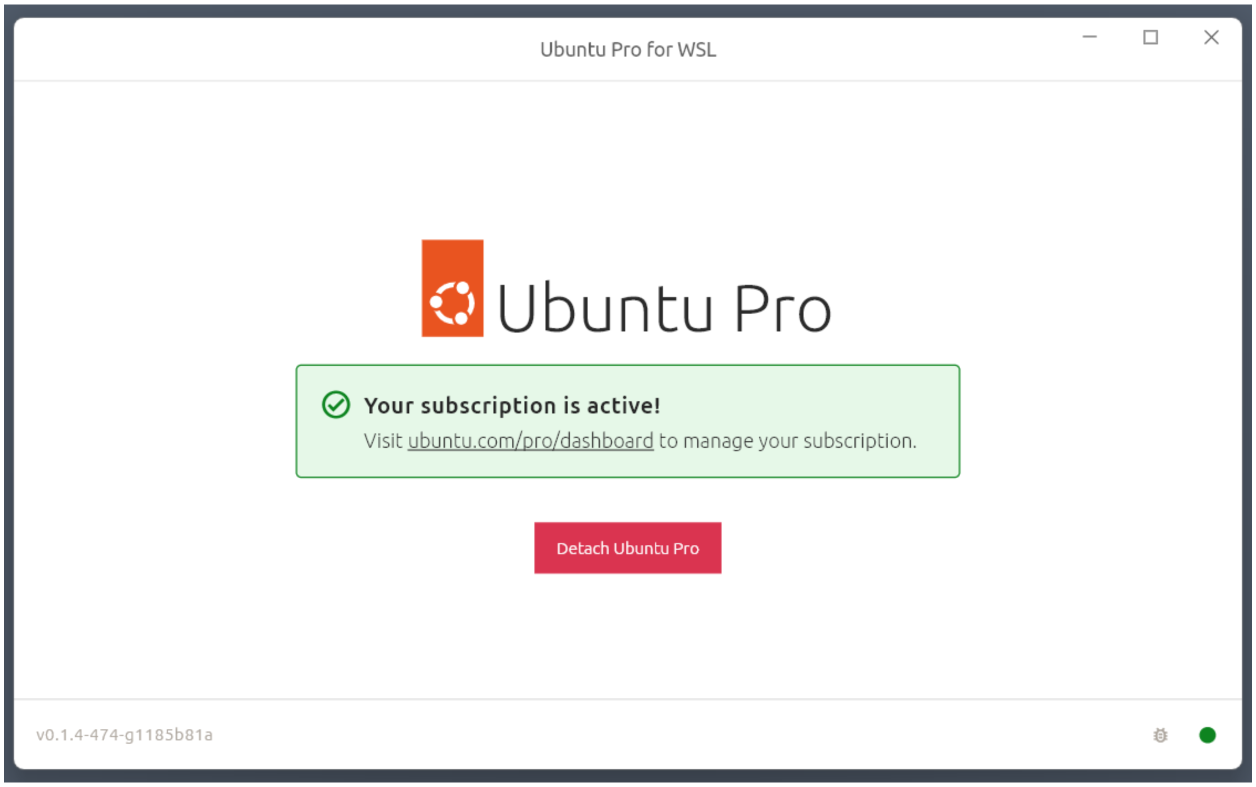
Configure the connection to Landscape to manage your Ubuntu WSL instances remotely.

[✓ Self-hosted Landscape configuration — Register with your own Landscape server]

Advanced Configuration — Load a custom Landscape client configuration file

Choose your preferred configuration option and enter the required details.

When you continue a status screen will appear confirming that configuration is complete:



Ubuntu Pro for WSL

Ubuntu Pro

Your subscription is active!
Visit ubuntu.com/pro/dashboard to manage your subscription.

As well as your Ubuntu Pro subscription being attached to UP4W on the Windows host, this has also configured the

Landscape client built into your UP4W Windows agent to know about your Landscape server. UP4W will forward this configuration to the Landscape client on your Ubuntu WSL instances as well, and all systems where the Landscape client has been configured this way are automatically registered with Landscape.

A dedicated how-to guide on configuring Landscape with UP4W can be found *here*.

Create an Ubuntu WSL instance locally

Open Windows PowerShell and run the following command to create a new Ubuntu 24.04 instance, creating a user and password when prompted. For quick testing, set both to `u`:

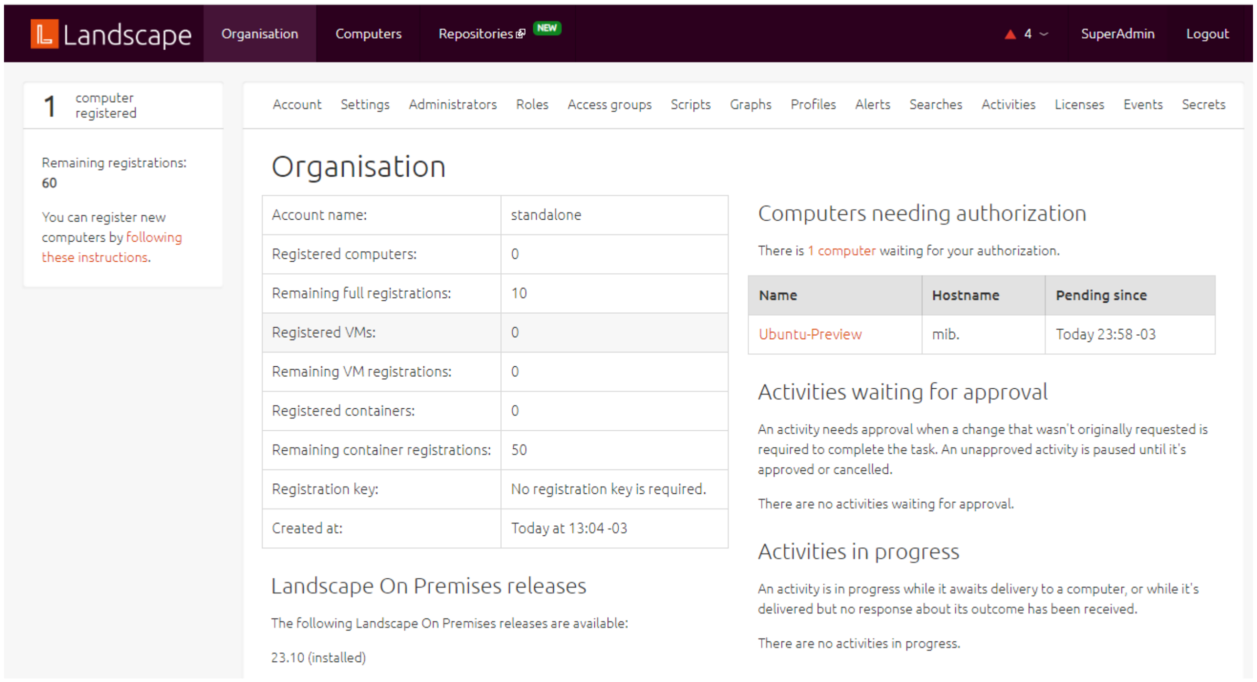
```
PS C:\Users\me\tutorial> ubuntu2404.exe
```

Verify Pro-attachment with:

```
u@mib:~$ pro status
```

UP4W should have also Landscape-registered this instance.

To verify, refresh the Landscape server web page and the instance should be listed under “Computers needing authorisation”.



1 computer registered

Remaining registrations: **60**

You can register new computers by following these instructions.

Organisation

Account name:	standalone
Registered computers:	0
Remaining full registrations:	10
Registered VMs:	0
Remaining VM registrations:	0
Registered containers:	0
Remaining container registrations:	50
Registration key:	No registration key is required.
Created at:	Today at 13:04 -03

Landscape On Premises releases

The following Landscape On Premises releases are available:

23.10 (installed)

Computers needing authorization

There is 1 computer waiting for your authorization.

Name	Hostname	Pending since
Ubuntu-Preview	mib.	Today 23:58 -03

Activities waiting for approval

An activity needs approval when a change that wasn't originally requested is required to complete the task. An unapproved activity is paused until it's approved or cancelled.

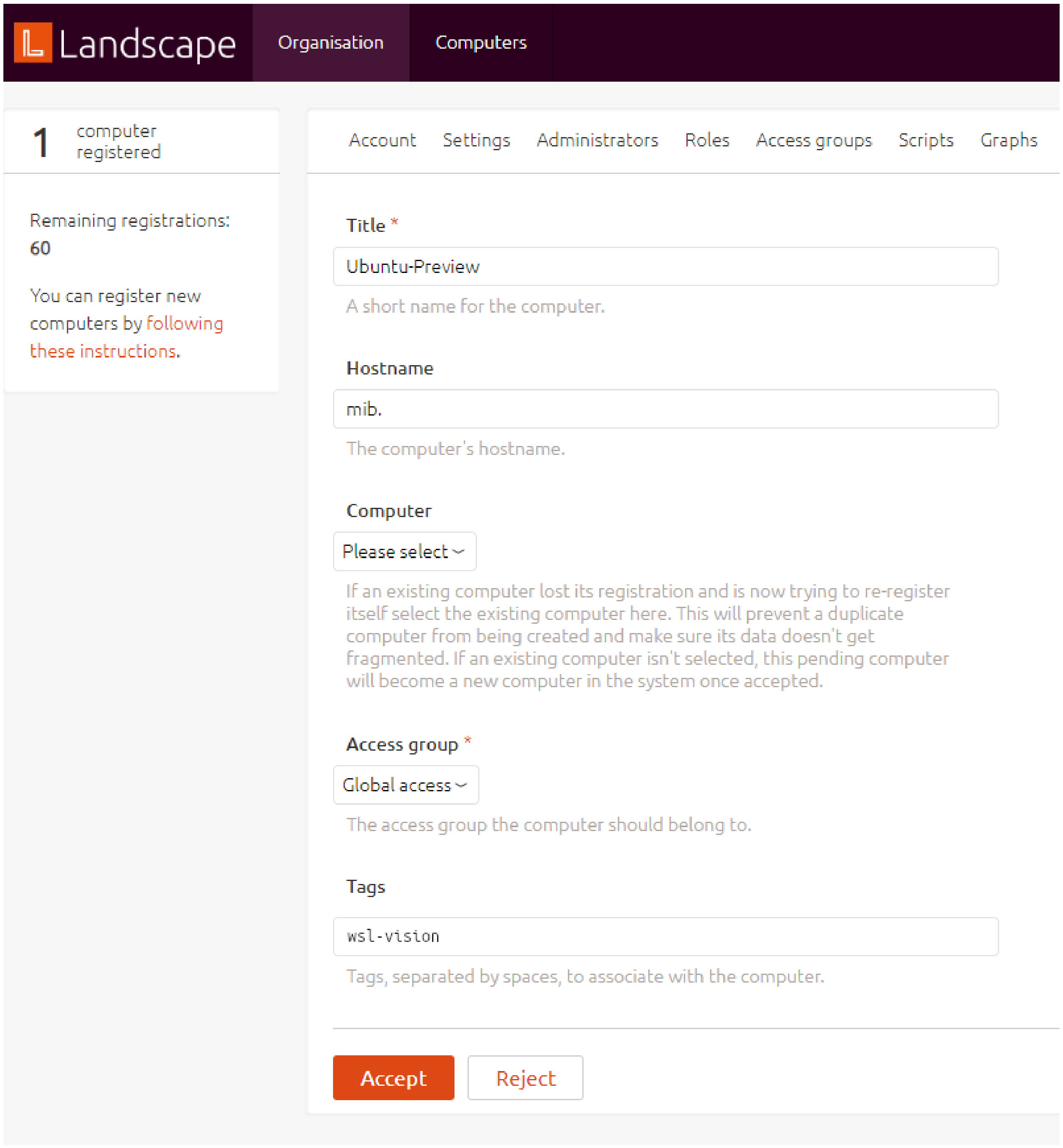
There are no activities waiting for approval.

Activities in progress

An activity is in progress while it awaits delivery to a computer, or while it's delivered but no response about its outcome has been received.

There are no activities in progress.

To accept the registration click on the instance name, set “Tags” to `wsl-vision` in the pop-up then click **Accept**. The `wsl-vision` tag will be used for all the instances accepted into Landscape.



1 computer registered

Remaining registrations: 60

You can register new computers by following these instructions.

Title *

Ubuntu-Preview

A short name for the computer.

Hostname

mib.

The computer's hostname.

Computer

Please select

If an existing computer lost its registration and is now trying to re-register itself select the existing computer here. This will prevent a duplicate computer from being created and make sure its data doesn't get fragmented. If an existing computer isn't selected, this pending computer will become a new computer in the system once accepted.

Access group *

Global access

The access group the computer should belong to.

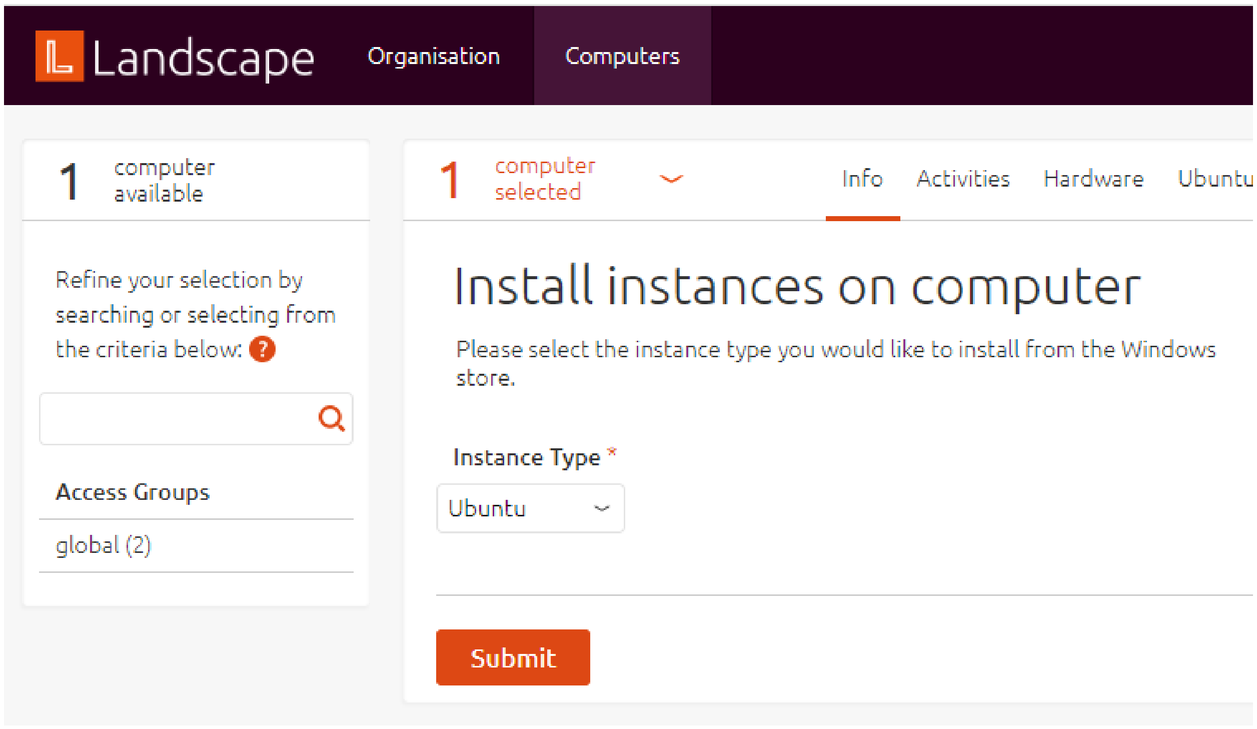
Tags

wsl-vision

Tags, separated by spaces, to associate with the computer.

Create an Ubuntu WSL instance remotely

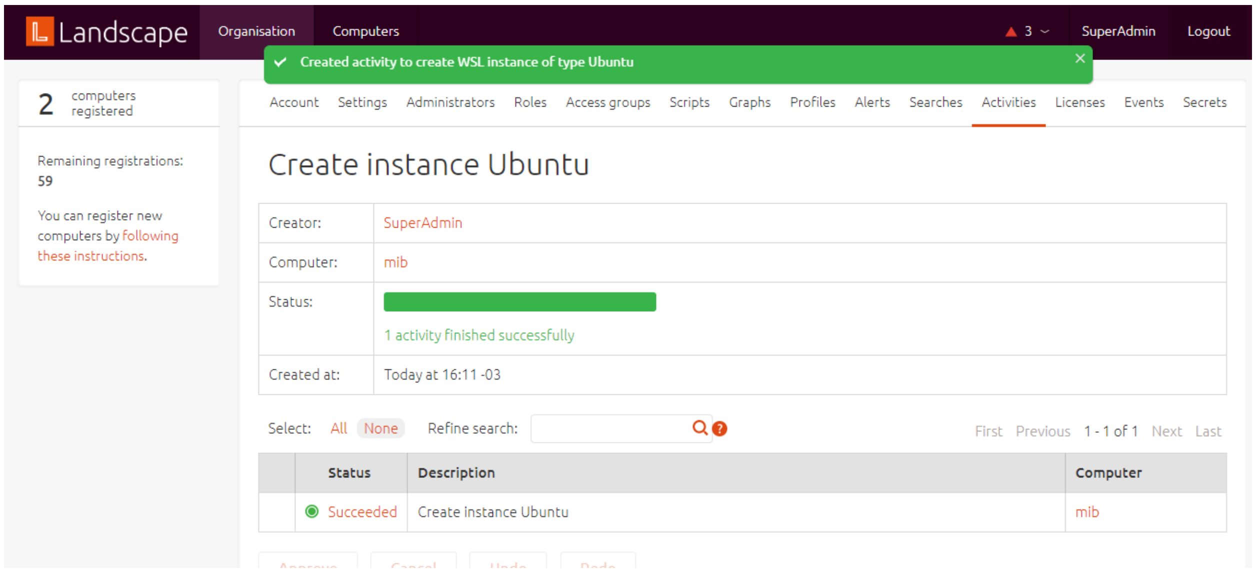
Back on the Landscape page in your web browser, navigate to "Computers" and click on the Windows machine (below: `mib`). You will find "WSL Instances" on the right side of the page. Click on the "Install new" link then set "Instance Type" to "Ubuntu" and click **Submit**. A status page will appear showing the progress of the new instance creation.



Install instances on computer

Please select the instance type you would like to install from the Windows store.

Instance Type *: Ubuntu



Created activity to create WSL instance of type Ubuntu

Remaining registrations: 59

You can register new computers by following these instructions.

Create instance Ubuntu

Creator:	SuperAdmin
Computer:	mib
Status:	1 activity finished successfully
Created at:	Today at 16:11 -03

	Status	Description	Computer
	● Succeeded	Create instance Ubuntu	mib

The Landscape server will talk to the Landscape client built into your UP4W. UP4W will then install the `Ubuntu` application and create an Ubuntu WSL instance automatically. In PowerShell, run `ubuntu.exe` to log in to the new instance.

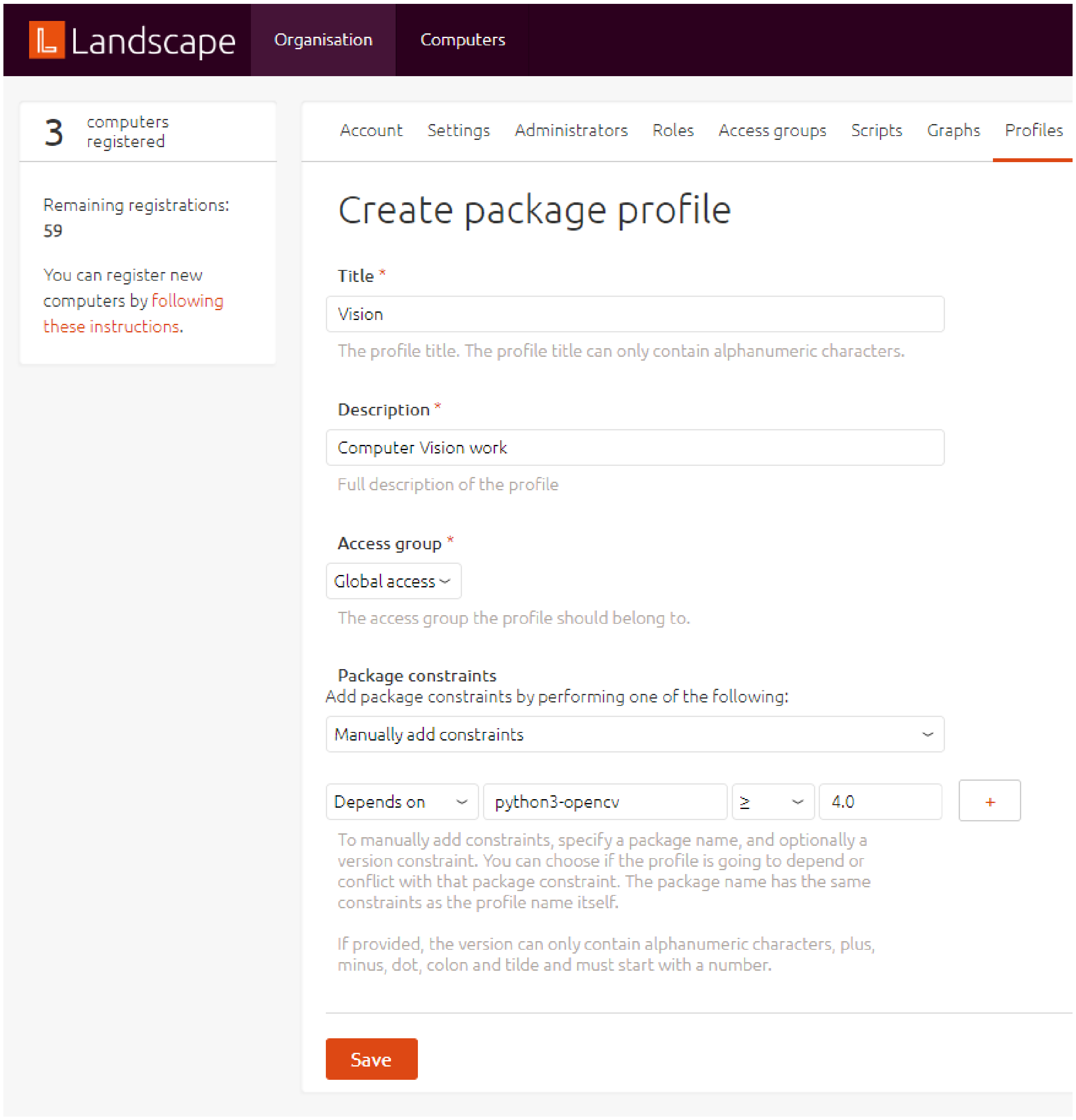
```
PS C:\Users\me\tutorial> ubuntu.exe

u@mib:~$
```

You can run `pro status` to verify pro-attachment and refresh your Landscape server page to verify and accept the registration. As before, apply the `wsl-vision` tag and click `Accept`.

Deploy packages to all Ubuntu WSL instances

On your Landscape server page, navigate to `Organization > Profiles`, click on `Package Profiles` then `Add package profile`. Fill in the form with the following values and click "Save".

Field	Value
Title	Vision
Description	Computer Vision work
Access group	Global
Package constraints	Manually add constraints
	Depends on `python3-opencv` >= `4.0`



3 computers registered

Remaining registrations: 59

You can register new computers by following these instructions.

Create package profile

Title *

Vision

The profile title. The profile title can only contain alphanumeric characters.

Description *

Computer Vision work

Full description of the profile

Access group *

Global access

The access group the profile should belong to.

Package constraints

Add package constraints by performing one of the following:

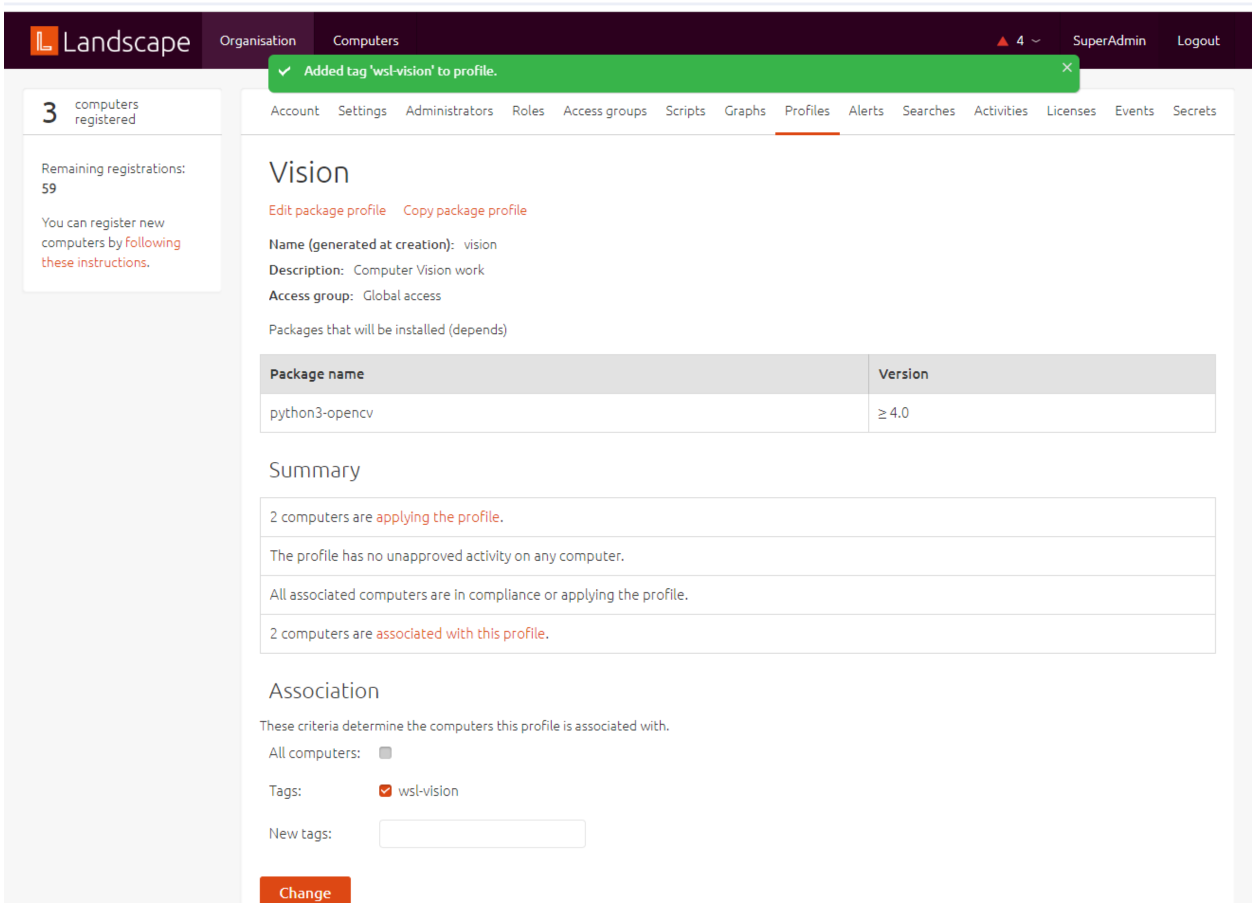
Manually add constraints

Depends on | python3-opencv | ≥ | 4.0

To manually add constraints, specify a package name, and optionally a version constraint. You can choose if the profile is going to depend or conflict with that package constraint. The package name has the same constraints as the profile name itself.

If provided, the version can only contain alphanumeric characters, plus, minus, dot, colon and tilde and must start with a number.

On the bottom of the "Vision" profile page, in the "Association" section, set the "New tags" field to `wsl-vision` and click **Change**.



Landscape | Organisation | Computers | ▲ 4 | SuperAdmin | Logout

✓ Added tag 'wsl-vision' to profile.

3 computers registered

Remaining registrations: 59

You can register new computers by following these instructions.

Account | Settings | Administrators | Roles | Access groups | Scripts | Graphs | Profiles | Alerts | Searches | Activities | Licenses | Events | Secrets

Vision

Edit package profile Copy package profile

Name (generated at creation): vision

Description: Computer Vision work

Access group: Global access

Packages that will be installed (depends)

Package name	Version
python3-opencv	≥ 4.0

Summary

2 computers are applying the profile.

The profile has no unapproved activity on any computer.

All associated computers are in compliance or applying the profile.

2 computers are associated with this profile.

Association

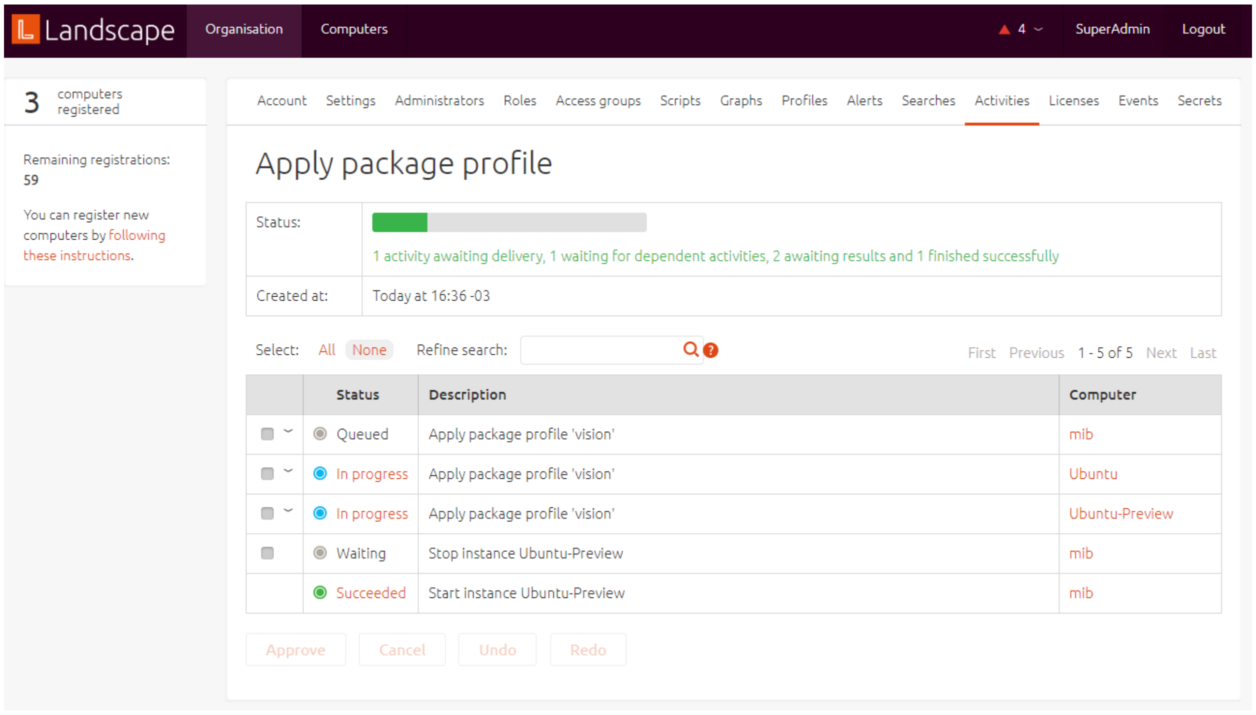
These criteria determine the computers this profile is associated with.

All computers: ☐

Tags: ✓ wsl-vision

New tags:

In the "Summary" section in the middle of the page you will see a status message showing that two computers are `applying the profile`. Click on the `applying the profile` link and then, in the "Activities" list, click on **Apply package profile** to see the progress of the package deployment.



Apply package profile

Status: 1 activity awaiting delivery, 1 waiting for dependent activities, 2 awaiting results and 1 finished successfully

Created at: Today at 16:36 -03

Select: All None Refine search:

First Previous 1 - 5 of 5 Next Last

Status	Description	Computer
Queued	Apply package profile 'vision'	mib
In progress	Apply package profile 'vision'	Ubuntu
In progress	Apply package profile 'vision'	Ubuntu-Preview
Waiting	Stop instance Ubuntu-Preview	mib
Succeeded	Start instance Ubuntu-Preview	mib

When this process has completed, use one of your instance shells to verify that the `python3-opencv` package has been installed. For example, in the `Ubuntu` instance the first three packages returned are:

```
u@mib:~$ apt list --installed | grep -m 3 opencv

libopencv-calib3d4.5d/jammy,now 4.5.4+dfsg-9ubuntu4 amd64 [installed,automatic]
libopencv-contrib4.5d/jammy,now 4.5.4+dfsg-9ubuntu4 amd64 [installed,automatic]
libopencv-core4.5d/jammy,now 4.5.4+dfsg-9ubuntu4 amd64 [installed,automatic]
```

You know how to leverage UP4W and Landscape to efficiently manage your Ubuntu WSL instances at scale.

Next steps

In the rest of the documentation you can find *how-to guides* for completing specific tasks, *reference* material describing key information relating to UP4W and dedicated *documentation for developers*.

2.2 How-to guides

These how-to guides cover key operations and processes in UP4W.

2.2.1 Install UP4W and add a Pro token

To install and configure UP4W you will need:

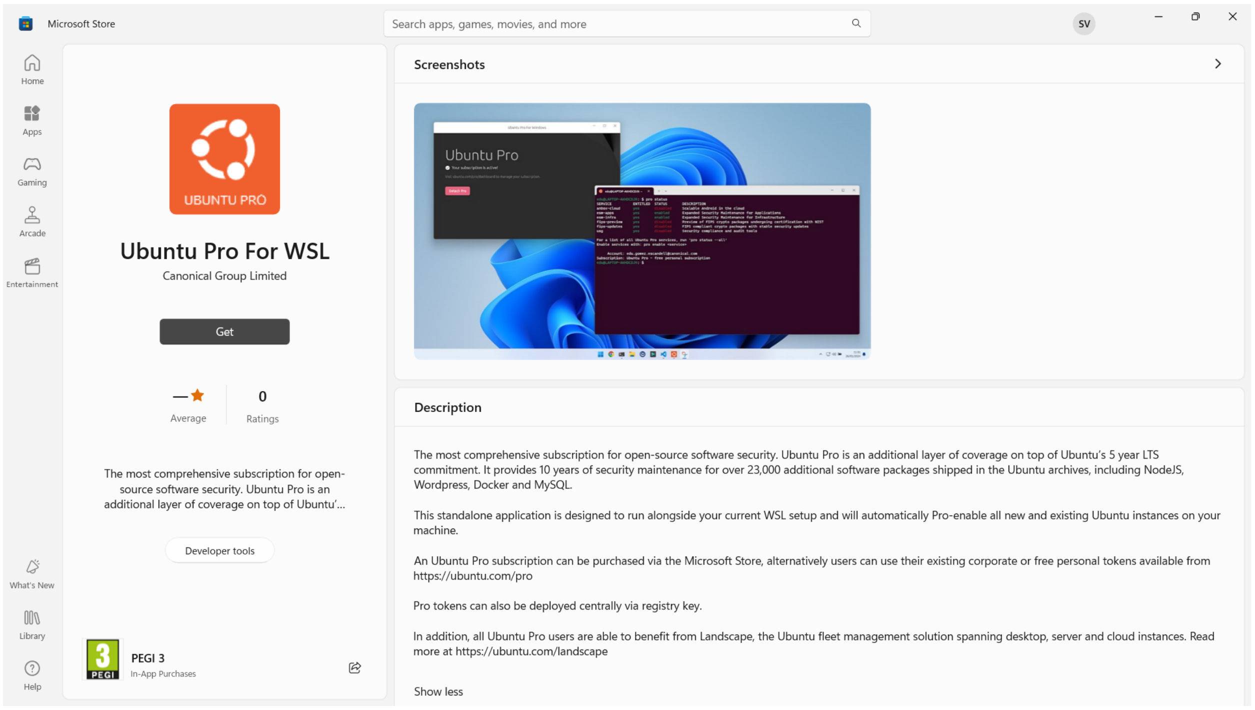
- A Windows 10 or 11 machine
- An Ubuntu Pro token

You should also verify that the *firewall rules are correctly set up*.

Install UP4W

> **Warning**
>
> The install link below will work only if you're logged in to the Microsoft Store with an account for which access to the app has been enabled.

You can install UP4W on this page of the Microsoft Store:



Ubuntu Pro For WSL

Canonical Group Limited

The most comprehensive subscription for open-source software security. Ubuntu Pro is an additional layer of coverage on top of Ubuntu'...

The most comprehensive subscription for open-source software security. Ubuntu Pro is an additional layer of coverage on top of Ubuntu's 5 year LTS commitment. It provides 10 years of security maintenance for over 23,000 additional software packages shipped in the Ubuntu archives, including NodeJS, Wordpress, Docker and MySQL.

This standalone application is designed to run alongside your current WSL setup and will automatically Pro-enable all new and existing Ubuntu instances on your machine.

An Ubuntu Pro subscription can be purchased via the Microsoft Store, alternatively users can use their existing corporate or free personal tokens available from https://ubuntu.com/pro

Pro tokens can also be deployed centrally via registry key.

In addition, all Ubuntu Pro users are able to benefit from Landscape, the Ubuntu fleet management solution spanning desktop, server and cloud instances. Read more at https://ubuntu.com/landscape

Choose a configuration method

After installation has finished you can start configuring UP4W in two ways:

- Windows registry: easier to automate and deploy at scale
- Graphical Windows application: convenient option for individual users

Click the appropriate tab to read more.

Windows registry

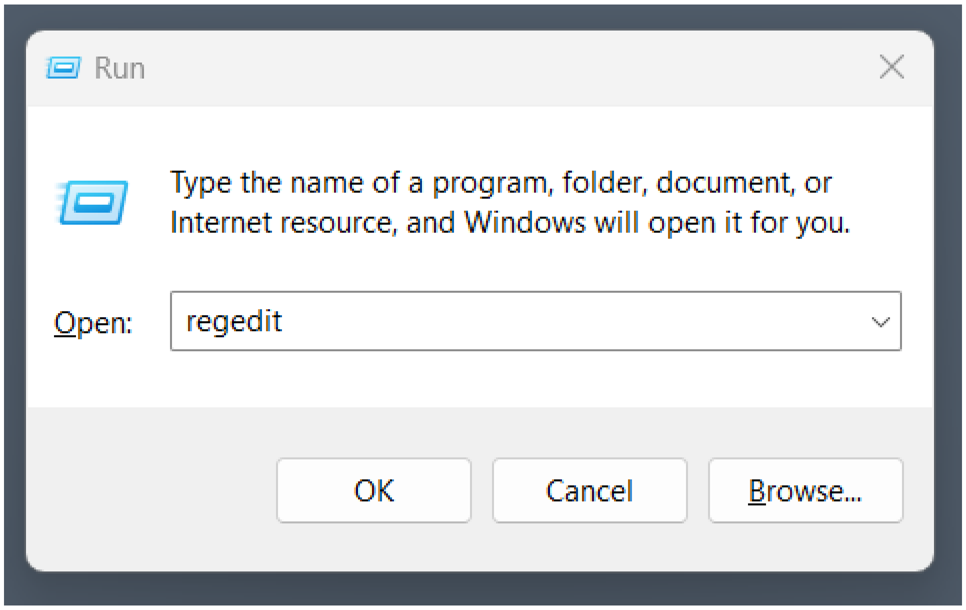
Graphical Windows application

Access the registry

First, ensure that UP4W has run at least once after installation. This ensures that the key and values necessary for configuration will be set up in the registry.

Advanced users of the registry can find relevant information in the Microsoft documentation about alternative methods for modifying the registry data.

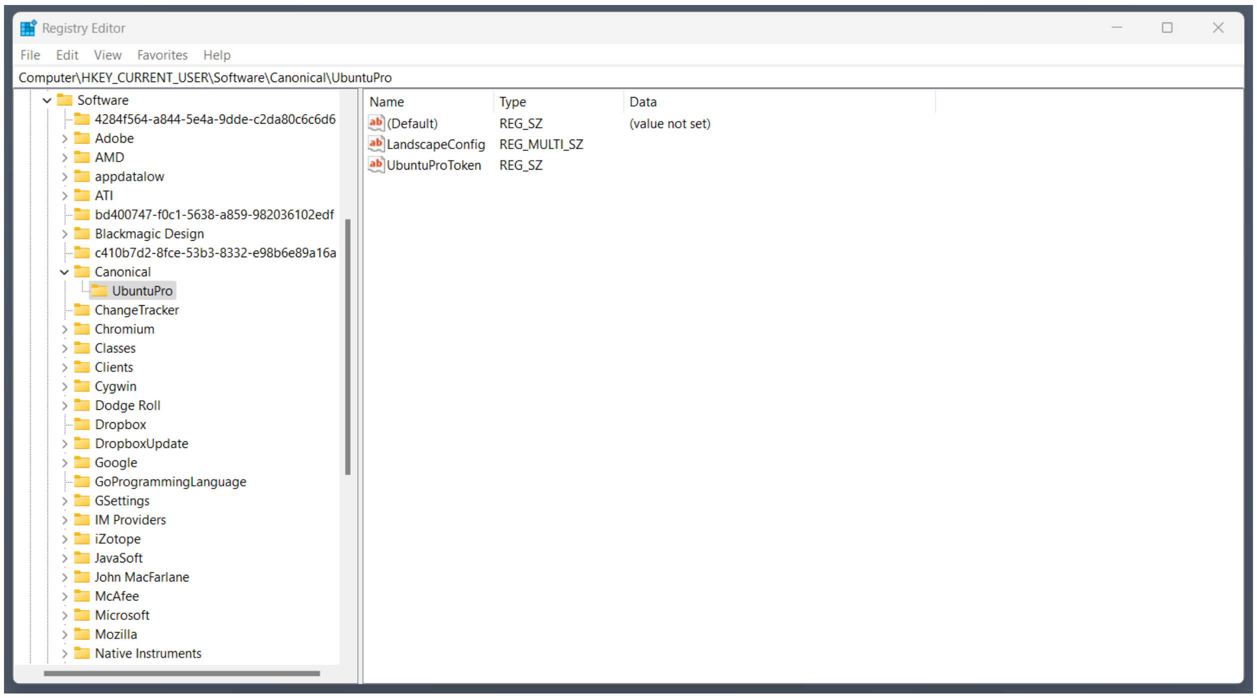
To open the registry type `Win+R` and enter `regedit`:


Run
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.
Open:
OK
Cancel
Browse...

regedit

Add Pro token in the registry

Navigate to `HKEY_CURRENT_USER\Software\Canonical\UbuntuPro`:

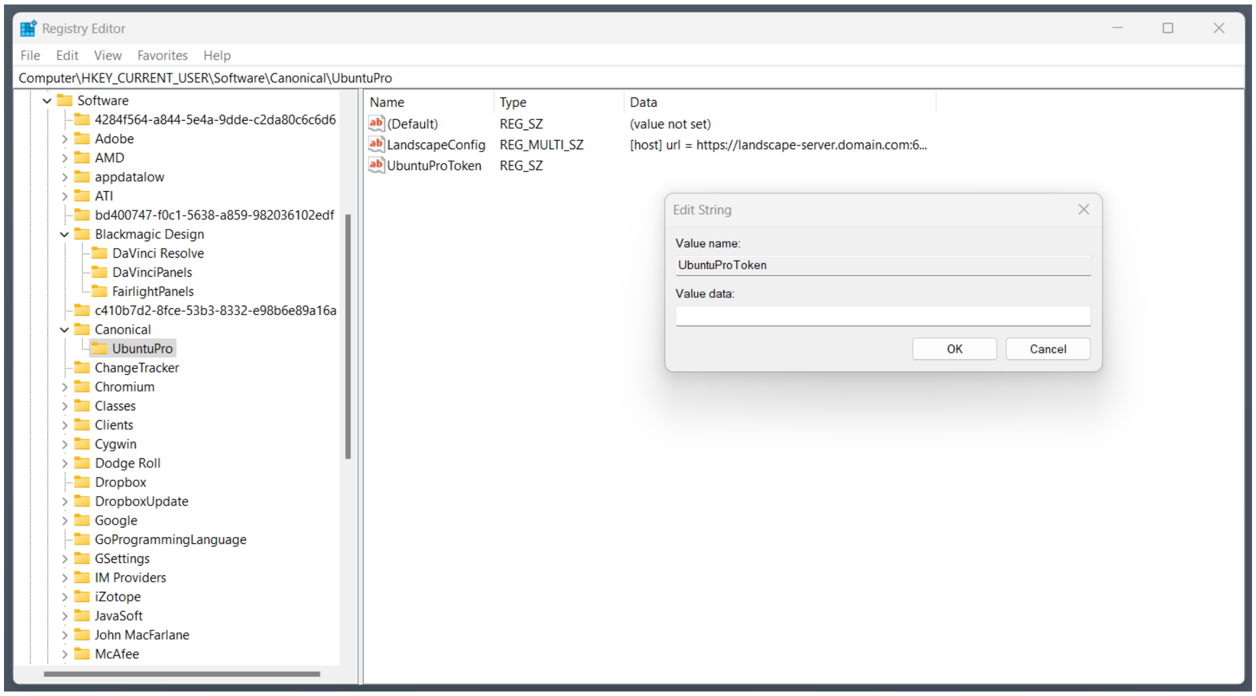


Computer\HKEY_CURRENT_USER\Software\Canonical\UbuntuPro

```
Software
  4284f564-a844-5e4a-9dde-c2da80c6c6d6
  Adobe
  AMD
  appdatalow
  ATI
  bd400747-f0c1-5638-a859-982036102edf
  Blackmagic Design
  c410b7d2-8fce-53b3-8332-e98b6e89a16a
  Canonical
    UbuntuPro
  ChangeTracker
  Chromium
  Classes
  Clients
  Cygwin
  Dodge Roll
  Dropbox
  DropboxUpdate
  Google
  GoProgrammingLanguage
  GSettings
  IM Providers
  iZotope
  JavaSoft
  John MacFarlane
  McAfee
  Microsoft
  Mozilla
  Native Instruments
```

Name	Type	Data
(Default)	REG_SZ	(value not set)
LandscapeConfig	REG_MULTI_SZ	
UbuntuProToken	REG_SZ	

Input your Ubuntu Pro token in `UbuntuProToken > Modify > Write the Ubuntu Pro token`:



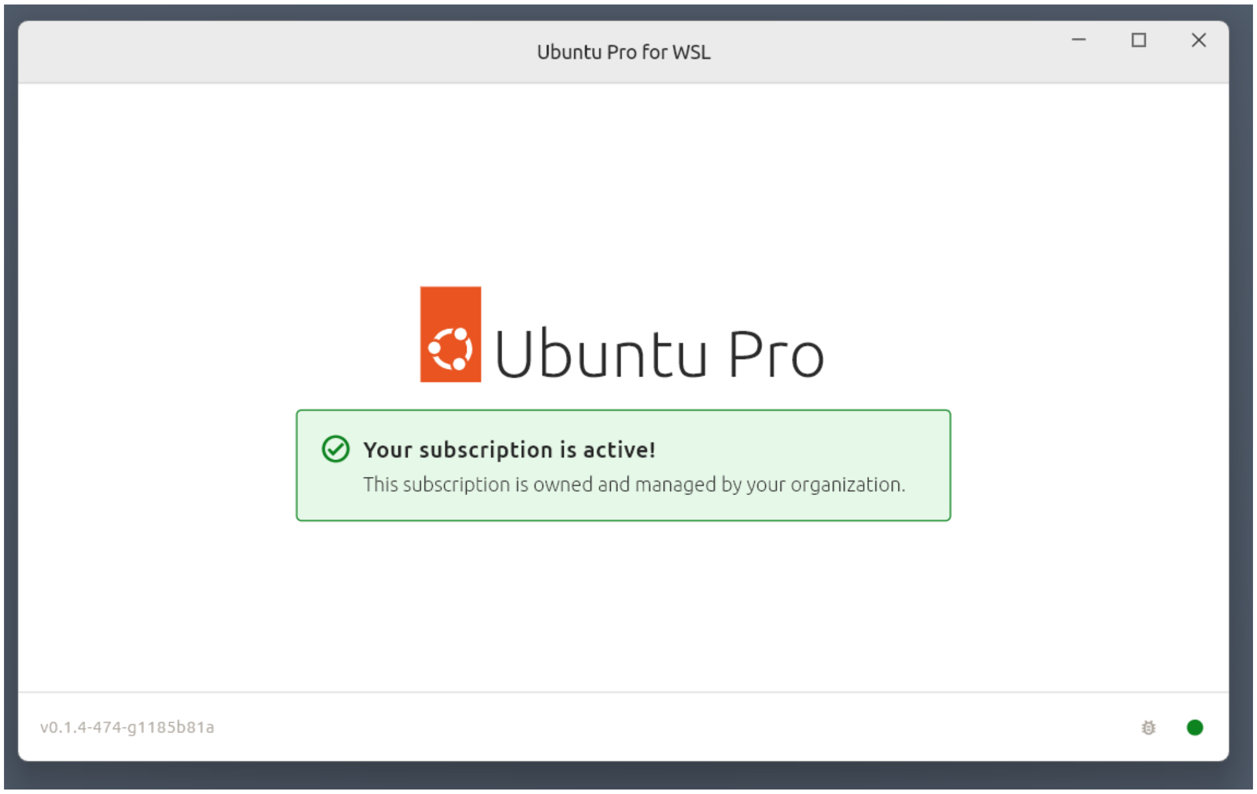
Computer\HKEY_CURRENT_USER\Software\Canonical\UbuntuPro

Name	Type	Data
(Default)	REG_SZ	(value not set)
LandscapeConfig	REG_MULTI_SZ	[host] url = https://landscape-server.domain.com:6...
UbuntuProToken	REG_SZ	

Value name: UbuntuProToken

Value data:

After configuration using the Windows Registry the status in the UP4W Windows application will show that the Pro subscription is active and managed by the user's organisation. Unlike installation through the graphical Windows application, there is no option to detach the Pro subscription in the application interface when the registry is used:



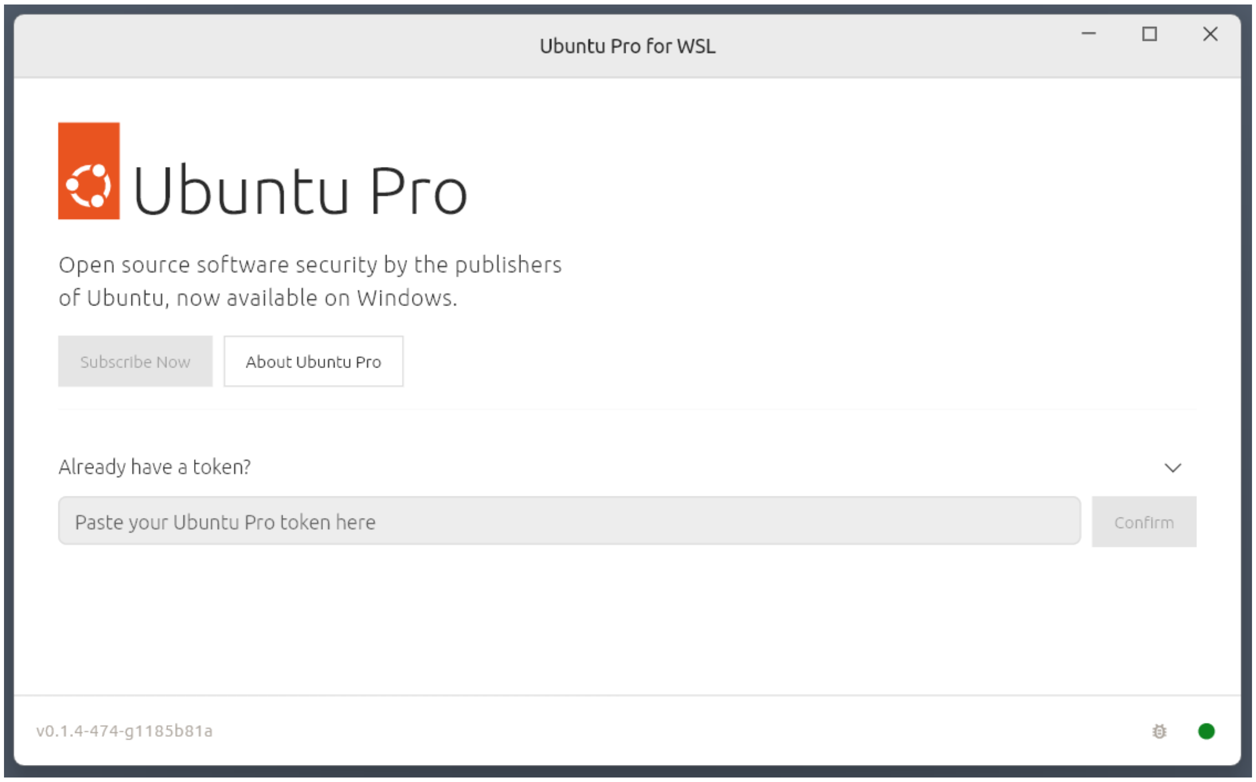
Your subscription is active!

This subscription is owned and managed by your organization.

v0.1.4-474-g1185b81a

Enter your Pro token

Enter your Ubuntu Pro token in the space provided:



Open source software security by the publishers of Ubuntu, now available on Windows.

Already have a token?

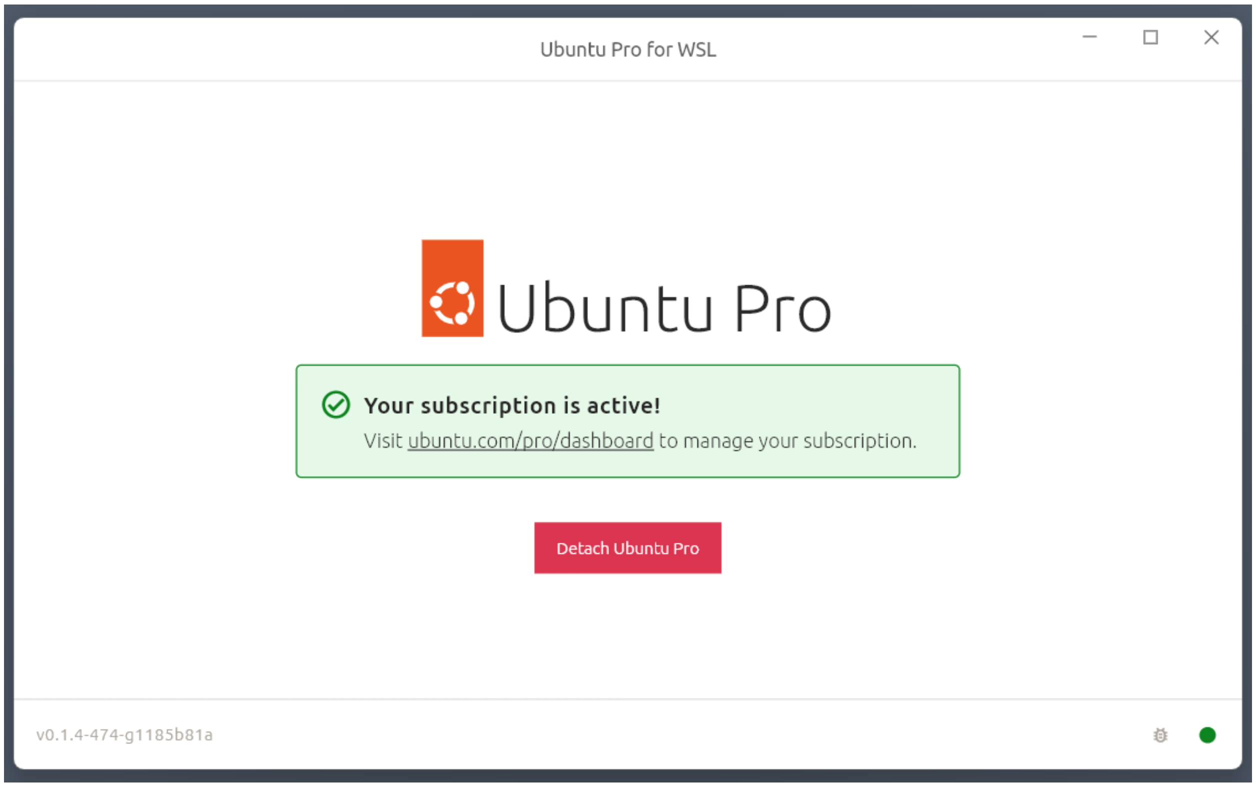
Paste your Ubuntu Pro token here

v0.1.4-474-g1185b81a

Continue to the confirmation screen.

Confirm subscription is active

You should now see that your Pro subscription is active:



Your subscription is active!
Visit ubuntu.com/pro/dashboard to manage your subscription.

Opening the application again at any point will show this screen, confirm the subscription is active and enable detaching of the subscription.

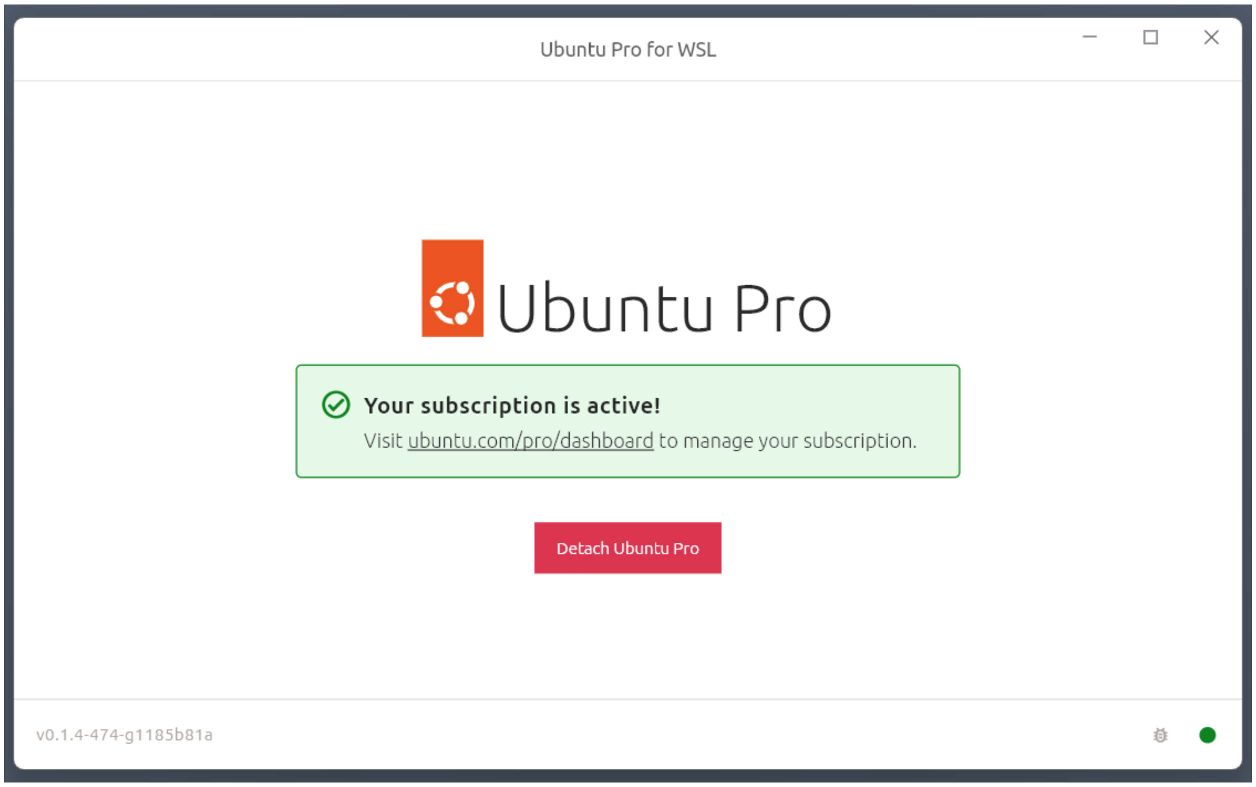
For additional verification steps refer to *our guide*.

2.2.2 Verify active Pro subscription and Pro attachment

If you have just installed and configured UP4W and a verification step is failing, wait for a few seconds and try again. The process should not take longer than a minute.

Pro subscription

After installing UP4W on a Windows machine and entering your token you should see a confirmation that your Pro subscription is active:



Your subscription is active!
Visit ubuntu.com/pro/dashboard to manage your subscription.

Find and run *Ubuntu Pro for WSL* from the Windows start menu at any time and the app will confirm whether you are subscribed.

Pro-attachment

Note

To verify Pro-attachment WSL should be installed on the Windows machine along with an Ubuntu distro — Ubuntu 24.04 LTS will be used in this example.

To verify Pro-attachment a new Ubuntu instance needs to be created. Running the following command in PowerShell will create a new Ubuntu-24.04 instance and prompt you to create a username and password for the machine:

```
PS C:\Users\me\up4wInstall> ubuntu2404.exe
```

You will now be logged in to the new instance shell and can check that UP4W has Pro-attached this instance with:

```
u@mib:~$ pro status
```

The output will confirm the following:

- Services like ESM are enabled
- Account and subscription information for Ubuntu Pro
- Verification of Pro-attachment

```
SERVICE          ENTITLED  STATUS       DESCRIPTION
esm-apps         yes       enabled      Expanded Security Maintenance for Applications
```

(continues on next page)

(continued from previous page)

```
esm-infra        yes       enabled      Expanded Security Maintenance for Infrastructure

NOTICES
Operation in progress: pro attach

For a list of all Ubuntu Pro services, run 'pro status --all'
Enable services with: pro enable <service>

     Account: me@ubuntu.com
Subscription: Ubuntu Pro - free personal subscription
```

Each new Ubuntu WSL instance that is created should automatically now be Pro-attached.

To find other useful Ubuntu pro commands run:

```
u@mib:~$ pro status
```

Landscape

For verification and troubleshooting of Landscape server and client configuration please refer to Landscape | View WSL host machines and child computers.

2.2.3 Back up, restore and duplicate Ubuntu WSL instances

Motivation

You may need to backup one of your Ubuntu WSL instances, if you want to:

- Perform a clean installation without losing data
- Create a snapshot before experimenting with your instance
- Share a pre-configured instance between machines
- Duplicate an instance so it can be run and configured independently

Backing up

Note

For simplicity, PowerShell commands in this section will all be run from the home directory of the user `me`.

To backup an Ubuntu-24.04 instance first make a `backup` folder in your home directory:

```
PS C:\Users\me> mkdir backup
```

You then need to create a compressed version of the Ubuntu instance in that backup directory:

```
PS C:\Users\me> wsl --export Ubuntu-24.04 .\backup\Ubuntu-24.04.tar.gz
```

Removal and deletion

Once you have created a backup of your Ubuntu distro it is safe to remove it from WSL and delete all associated data.

This can be achieved with the following command:

```
PS C:\Users\me> wsl --unregister Ubuntu-24.04
```

Restoring

If you want to restore the Ubuntu-24.04 instance that you have previously backed up run:

```
PS C:\Users\me> wsl --import Ubuntu-24.04 .\backup\Ubuntu2404\ .\backup\Ubuntu-24.04.tar.
↪gz
```

This will import your previous data and if you run `ubuntu2404.exe` an Ubuntu WSL instance should be restored with your previous configuration intact.

To login as a user k, created with the original instance, run:

```
PS C:\Users\me> wsl -d Ubuntu-24.04 -u k
```

Alternatively, add the following to `/etc/wsl.conf` in the instance:

```
[user]
default=k
```

Without specifying a user you will be logged in as the root user.

Duplication

It is also possible to create multiple instances from a base instance. Below the restore process is repeated but the new instances are assigned different names than the original backup:

```
PS C:\Users\me> wsl --import ubuntu2404b .\backup\Ubuntu2404b\ .\backup\Ubuntu-24.04.tar.
↪gz
PS C:\Users\me> wsl --import ubuntu2404c .\backup\Ubuntu2404c\ .\backup\Ubuntu-24.04.tar.
↪gz
```

This will create two additional instances of Ubuntu 24.04 that can be launched and configured independently.

In PowerShell, running `wsl -l -v` will output the new instances in your list of installed distributions:

```
NAME            STATE          VERSION
Ubuntu-24.04    Stopped        2
ubuntu2404b     Stopped        2
ubuntu2404c     Stopped        2
```

To launch the first derived instance and login as the user k run:

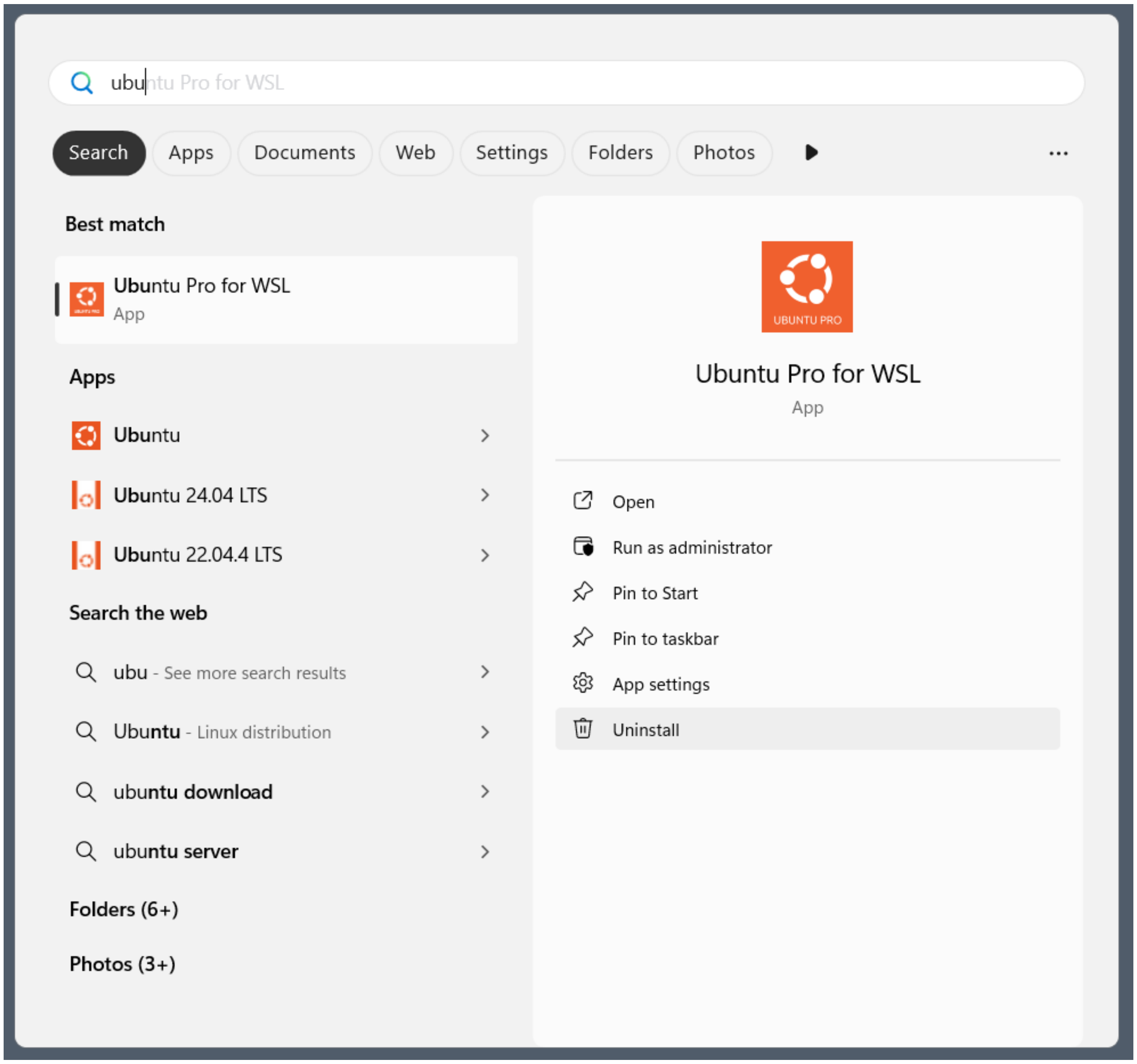
```
PS C:\Users\me> wsl -d ubuntu2404b -u k
```

2.2.4 Uninstall UP4W, Ubuntu WSL and WSL

Uninstalling UP4W, Ubuntu WSL apps and WSL generally only requires finding the relevant application in the Windows Start Menu and clicking **Uninstall**, although in some cases a few additional steps are required.

UP4W

In the Windows Start Menu, locate the “Ubuntu Pro for WSL” application and right-click on it, then click **Uninstall**.



You should also remove the `.ubuntupro` directory from your Windows user profile directory.

```
PS C:\Users\me> Remove-Item -Recurse -Force C:\Users\me\.ubuntupro
```

Ubuntu WSL apps

In PowerShell run the following command to stop WSL:

```
PS C:\Users\me> wsl --shutdown
```

Then, in the Windows Start Menu, locate the “Ubuntu 24.04 LTS” application, right-click on it, and select “Uninstall”.

The instances will be removed automatically.

WSL app

Only do this if you no longer need WSL on your Windows machine.

In the Windows Start Menu locate the “WSL” application, right-click on it then select “Uninstall”.

2.2.5 Configure the Landscape client with UP4W

Choose a configuration method

The Landscape client can be configured in two ways:

- Windows registry: easier to automate and deploy at scale
- Graphical Windows application: convenient option for individual users

Click the appropriate tab to read more.

Windows registry

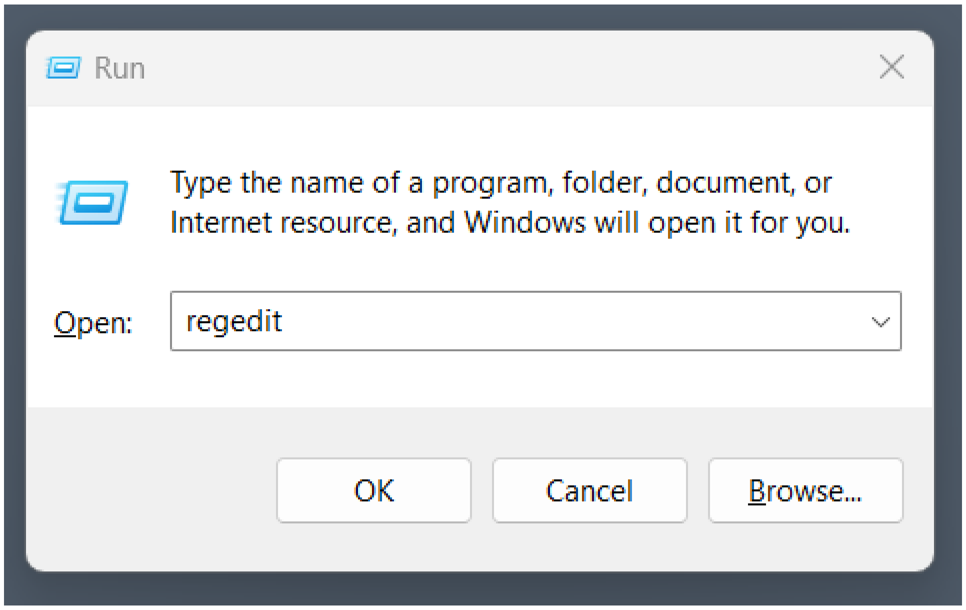
Graphical Windows application

Access the registry

First, ensure that UP4W has run at least once after installation. This ensures that the key and values necessary for configuration will be set up in the registry.

Advanced users of the registry can find relevant information in the Microsoft documentation about alternative methods for modifying the registry data.

To open the registry type `Win+R` and enter `regedit`:

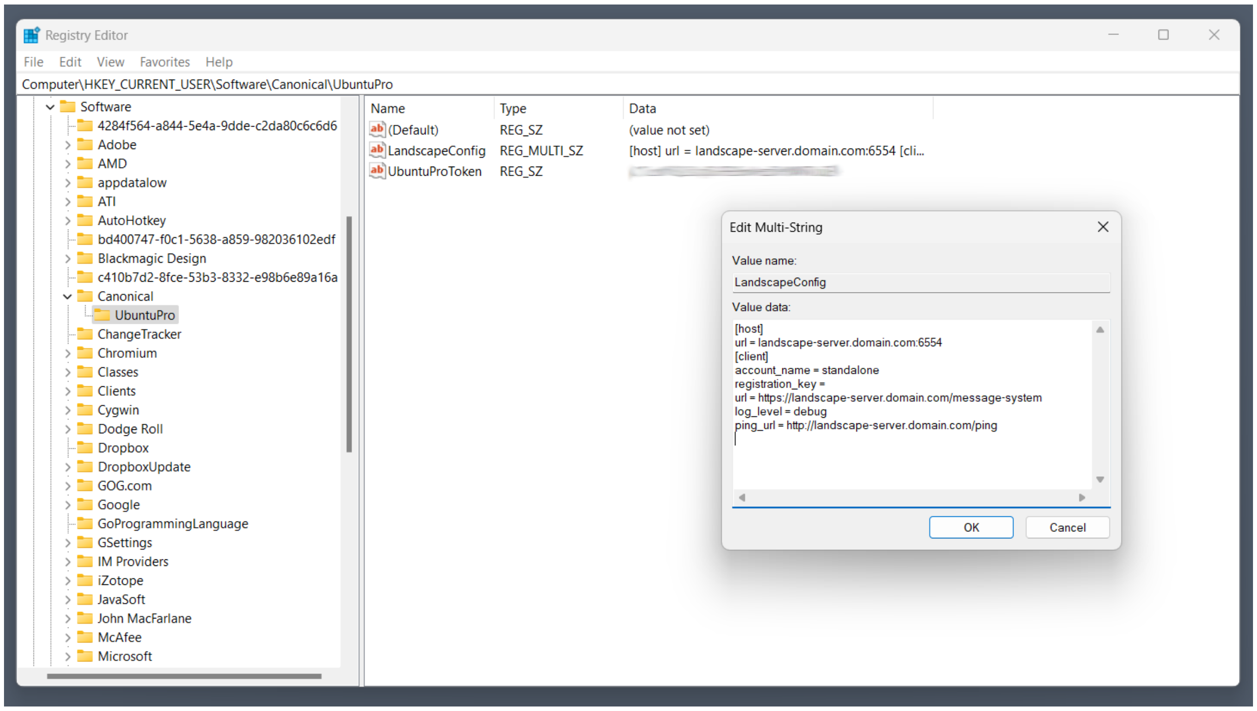


Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: regedit

Configure Landscape in the registry

If you are using Landscape you can input your configuration in `LandscapeConfig > Modify > Write the Landscape config`:



Computer\HKEY_CURRENT_USER\Software\Canonical\UbuntuPro

Name	Type	Data
(Default)	REG_SZ	(value not set)
LandscapeConfig	REG_MULTI_SZ	[host] url = landscape-server.domain.com:6554 [cli...
UbuntuProToken	REG_SZ	[redacted]

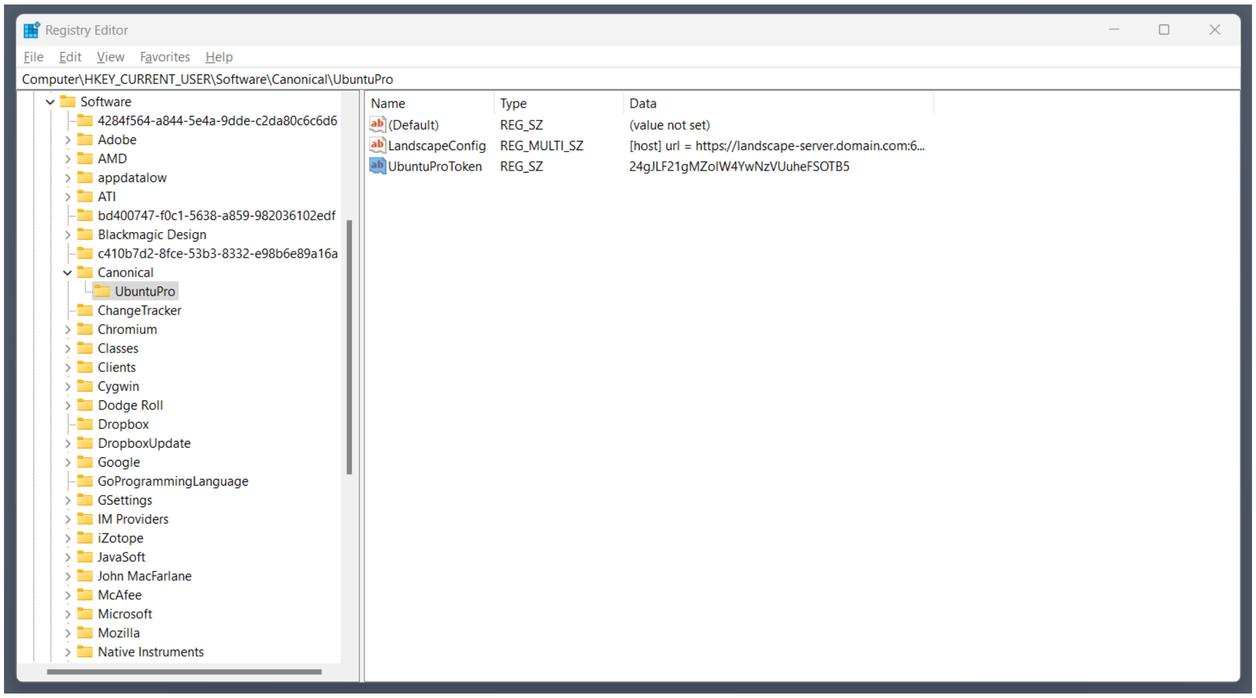
Value name:
LandscapeConfig

Value data:

```
[host]
url = landscape-server.domain.com:6554
[client]
account_name = standalone
registration_key =
url = https://landscape-server.domain.com/message-system
log_level = debug
ping_url = http://landscape-server.domain.com/ping
```

Refer to the section on *Landscape client configuration* for an example.

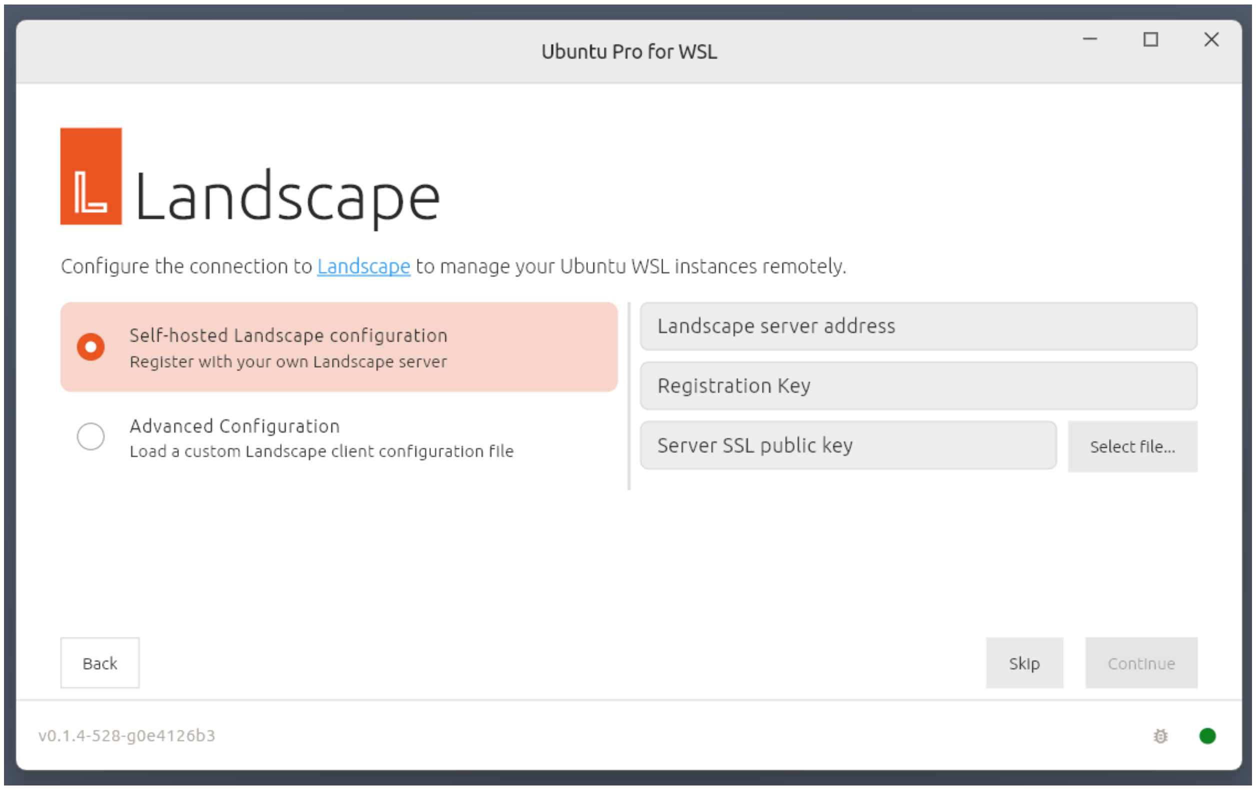
After you have populated the configuration with data you should be ready to create and manage automatically Pro-attaching WSL instances through Landscape:



Computer\HKEY_CURRENT_USER\Software\Canonical\UbuntuPro

Name	Type	Data
(Default)	REG_SZ	(value not set)
LandscapeConfig	REG_MULTI_SZ	[host] url = https://landscape-server.domain.com:6...
UbuntuProToken	REG_SZ	24gJLF21gMZolW4YwNzVUuheFSOTB5

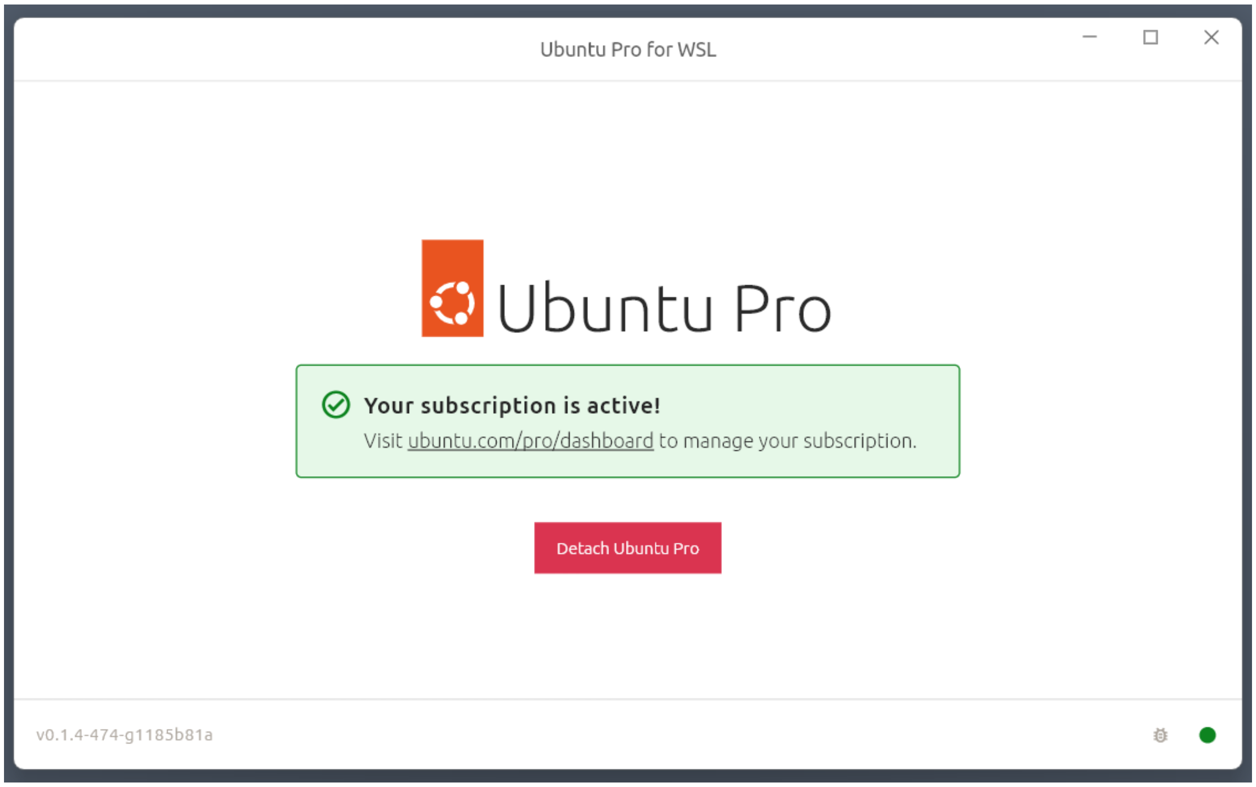
In the UP4W app navigate to the Landscape configuration screen:


Ubuntu Pro for WSL
Landscape
Configure the connection to Landscape to manage your Ubuntu WSL instances remotely.
Self-hosted Landscape configuration
Register with your own Landscape server
Advanced Configuration
Load a custom Landscape client configuration file
Landscape server address
Registration Key
Server SSL public key
Select file...
Back
Skip
Continue
v0.1.4-528-g0e4126b3

Choose your preferred configuration option and enter the required details.

The "Advanced Configuration" option requires you to specify a `landscape.conf`. Refer to the section on *Landscape client configuration* for an example.

When you continue a status screen will appear confirming that configuration is complete:


Ubuntu Pro for WSL
Ubuntu Pro
Your subscription is active!
Visit ubuntu.com/pro/dashboard to manage your subscription.
Detach Ubuntu Pro
v0.1.4-474-g1185b81a

Note on SSL public key

If the machine running the server is not trusted on your network you may need to explicitly reference a path to the SSL public key on a Windows host machine.

For example, if you followed the Landscape Quickstart installation, the auto-generated self-signed certificate can be found at `/etc/ssl/certs/landscape_server.pem`.

This can be copied to a Windows machine:

C:\Users\<YOUR_WINDOWS_USER_NAME>\landscape_server.pem

The path can then be referenced during Landscape configuration in the UP4W Windows app. If necessary, the SSL public key can be added after the Windows host has first been registered in Landscape.

Configuring the landscape client

Both the `LandscapeConfig` data in the Windows registry and the Advanced Configuration option in the graphical Windows application can be configured as follows:

```
[host]
url = landscape-server.domain.com:6554

[client]
url = https://landscape-server.domain.com/message-system
ping_url  = http://landscape-server.domain.com/ping
account_name = standalone
log_level = debug
ssl_public_key = C:\Users\user\Downloads\landscape_server.pem
```

Warning

The `ping_url` must be a `http` address. A `https` address will not work.

A more comprehensive example of the configuration options is provided here.

2.2.6 Install Landscape server in a WSL instance

Motivation

While a Landscape server typically runs on external computers, it can also be set up on a WSL instance on a Windows machine.

This is especially useful if you want to test UP4W on a single Windows device. For example, the *deployment tutorial* can be completed with a Landscape server running in an Ubuntu WSL instance. The Landscape server can then be used to manage other WSL instances running UP4W and the Landscape client.

Guide

In PowerShell, `shutdown` WSL then install the Ubuntu 24.04 LTS instance with the `--root` option.

```
PS C:\Users\me\tutorial> wsl --shutdown

PS C:\Users\me\tutorial> ubuntu2404.exe install --root
```

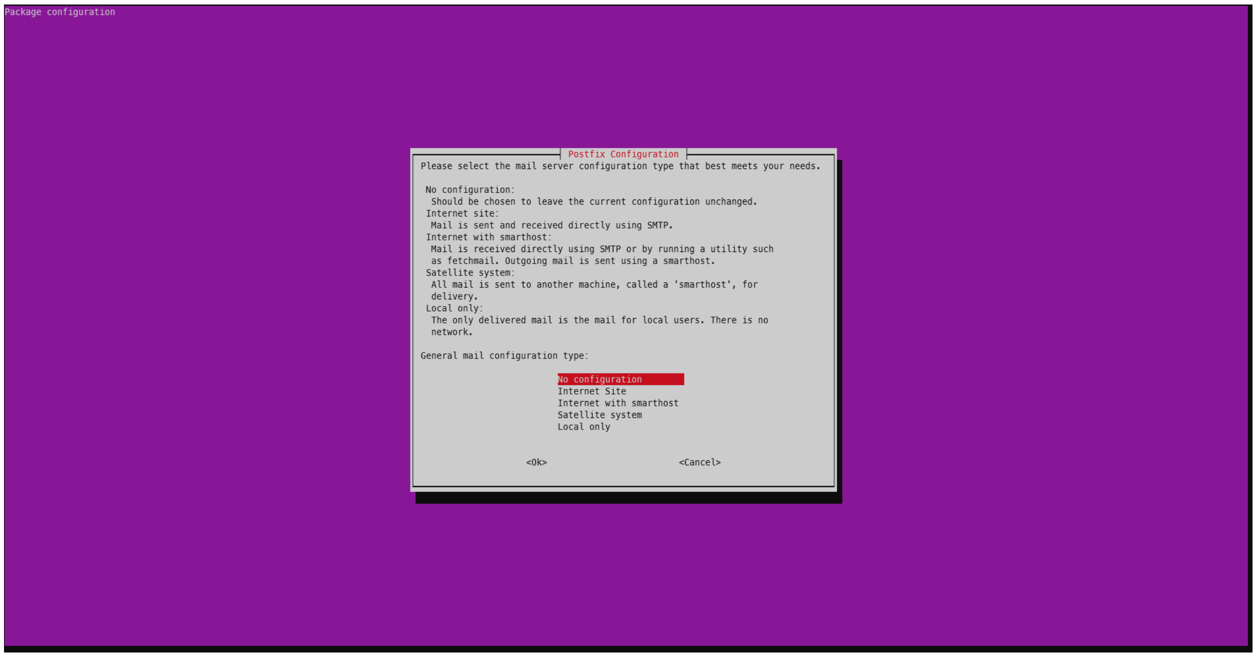
After successful installation log in to the new instance and add the landscape apt repository:

```
PS C:\Users\me\tutorial> ubuntu2204.exe

root@mib:~$ add-apt-repository ppa:landscape/self-hosted-beta -y
```

Update packages and then install the `landscape-server-quickstart` package.

```
root@mib:~$ apt update

root@mib:~$ apt install landscape-server-quickstart -y
```

A dialog will appear for 'Postfix configuration'. For 'General mail configuration type' select **No configuration**. Hit **Tab** to highlight the **Ok** button, press **Enter** and you will be returned to the shell prompt.



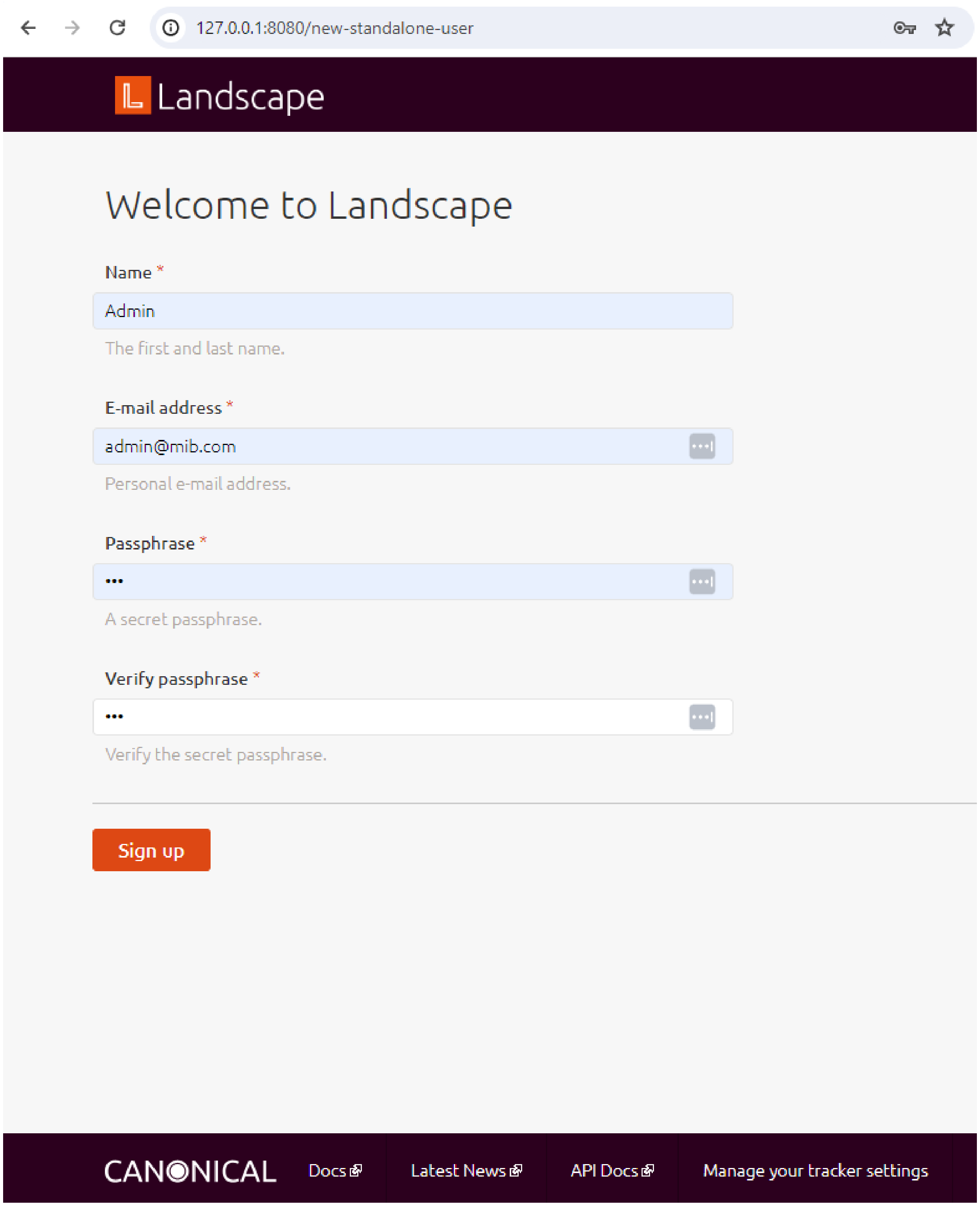
```
Package configuration

Postfix Configuration
Please select the mail server configuration type that best meets your needs.

 No configuration:
  Should be chosen to leave the current configuration unchanged.
 Internet site:
  Mail is sent and received directly using SMTP.
 Internet with smarthost:
  Mail is received directly using SMTP or by running a utility such
  as fetchmail. Outgoing mail is sent using a smarthost.
 Satellite system:
  All mail is sent to another machine, called a 'smarthost', for
  delivery.
 Local only:
  The only delivered mail is the mail for local users. There is no
  network.

General mail configuration type:

                         No configuration
                         Internet Site
                         Internet with smarthost
                         Satellite system
                         Local only


              <Ok>                          <Cancel>
```

If Landscape has installed successfully, the log will indicate that Landscape systemd units are active. An example log is shown below for the first three units:

```
root@mib:~$ systemctl --state=running --no-legend --no-pager | grep -m 3 landscape
  landscape-api.service                   loaded active running LSB: Enable Landscape API
  landscape-appserver.service             loaded active running LSB: Enable Landscape␣
↪frontend UI
  landscape-async-frontend.service        loaded active running LSB: Enable Landscape␣
↪async frontend
```

Once installed Landscape will be served on `localhost` port 8080. Open your favourite browser on Windows and navigate to `http://127.0.0.1:8080` to create the Landscape global admin account. Enter the following credentials and click the **Sign Up** button:

Field	Value
Name	Admin
E-mail address	`admin@mib.com`
Passphrase	123
Verify passphrase	123



127.0.0.1:8080/new-standalone-user

Landscape

Welcome to Landscape

Name *	Admin
	The first and last name.
E-mail address *	admin@mib.com
	Personal e-mail address.
Passphrase *	•••
	A secret passphrase.
Verify passphrase *	•••
	Verify the secret passphrase.

The Landscape client inside of any WSL instance will need the Landscape server certificate to connect to the server.

To achieve this copy the Landscape server certificate into your Windows user profile directory:

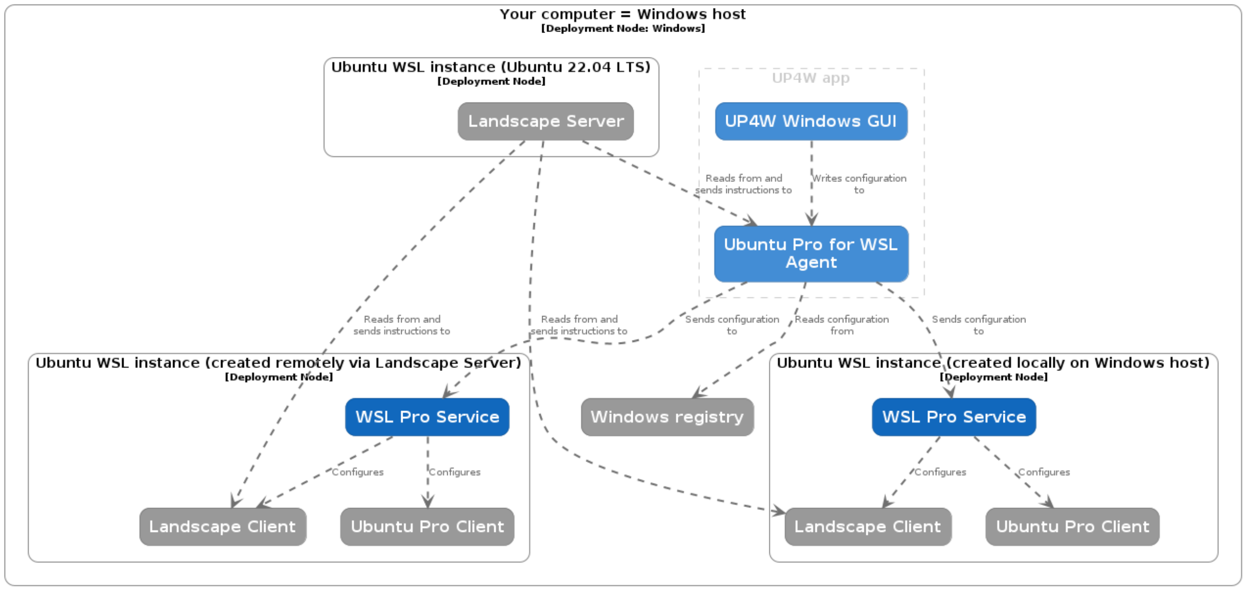
```
root@mib:~$ cp /etc/ssl/certs/landscape_server.pem /mnt/c/users/me/
```

Done – your self-hosted Landscape server is now up and running!

Now if you configure the Landscape client on any Ubuntu WSL instances to detect this server, they will also be registered with the Landscape service included in your Ubuntu Pro subscription.

The server will stay running until you close the terminal. If you do close the terminal running `ubuntu2404.exe` in a new terminal window will start the Landscape server automatically.

Using this server setup when following the *deployment tutorial* would result in the following architecture:



2.2.7 How-to deploy a custom rootfs across multiple Windows machines with the Landscape API

This guide shows how to use the Landscape API to automate the deployment of a custom rootfs across multiple Windows machines. Scaled deployment is enabled by Ubuntu Pro for WSL, which ensures that Ubuntu WSL instances on Windows machines are automatically registered with Landscape. Cloud-init is used for initialisation and final configuration of the instances. To follow the steps outlined in this guide you can use either:

- Bash scripting on Linux, or
- PowerShell scripting on Windows

Prerequisites

- A running self-hosted Landscape server version `24.10~beta.5` or later.
- Multiple Windows machines *already registered with Landscape* via Ubuntu Pro for WSL.
- Make sure you have installed `curl` and `jq`, if you're following this guide using Bash.
- Familiarity with Bash and/or PowerShell.

Prepare the environment

For convenience when writing subsequent commands, first export the following environment variables, modifying the values that are assigned as needed:

Bash

PowerShell

```
# Credentials to authenticate the API requests
export LANDSCAPE_USER_EMAIL=admin@mib.com
export LANDSCAPE_USER_PASSWORD=mib
export LANDSCAPE_URL=https://landscape.mib.com

# The URL of the custom rootfs to be deployed
export ROOTFS_URL="http://landscape.mib.com:9009/ubuntu-24.04-custom.tar.gz"

# The list of IDs of the different Windows machines on which we are going to deploy WSL␣
↪instances
export PARENT_COMPUTER_IDS=(26 30 31)

# The name of the WSL instance to be created
export COMPUTER_NAME=Carbonizer

# Path to the cloud-config file whose contents will be used to initialize the WSL␣
↪instances
export CLOUD_INIT_FILE="~/Downloads/init.yaml"
```

```
# Credentials to authenticate the API requests
$LANDSCAPE_USER_EMAIL="admin@mib.com"
$LANDSCAPE_USER_PASSWORD="mib"
$LANDSCAPE_URL="https://landscape.mib.com"

# The URL of the custom rootfs to be deployed
$ROOTFS_URL="http://landscape.mib.com:9009/ubuntu-24.04-custom.tar.gz"

# The list of IDs of the different Windows machines on which we are going to deploy WSL␣
↪instances
$PARENT_COMPUTER_IDS=@(26, 30, 31)

# The name of the WSL instance to be created
$COMPUTER_NAME="Carbonizer"

# Path to the cloud-config file whose contents will be used to initialize the WSL␣
↪instances
$CLOUD_INIT_FILE="~\Downloads\init.yaml"
```

Generate a Base64-encoded string with the cloud-config data:

Bash

PowerShell

```
BASE64_ENCODED_CLOUD_INIT=$(cat $CLOUD_INIT_FILE | base64 --wrap=0)
```

```
$content = Get-Content -Path $CLOUD_INIT_FILE -Raw
$bytes = [System.Text.Encoding]::UTF8.GetBytes($content)
$BASE64_ENCODED_CLOUD_INIT = [System.Convert]::ToBase64String($bytes)
```

Authenticate against the Landscape API

Build the authentication payload of the form: `{"email": "admin@mib.com", "password": "mib"}` using the values exported in prior steps:

Bash

PowerShell

```
LOGIN_JSON=$( jq -n \
    --arg em "$LANDSCAPE_USER_EMAIL" \
    --arg pwd "$LANDSCAPE_USER_PASSWORD" \
    '{email: $em, password: $pwd}' )
```

```
$LOGIN_JSON = @{
 email = "$LANDSCAPE_USER_EMAIL"
 password = "$LANDSCAPE_USER_PASSWORD"
} | ConvertTo-Json
```

Issue an authenticate request and retrieve the JSON web token (JWT) to be used in the subsequent API requests.

Bash

PowerShell

```
LOGIN_RESPONSE=$( curl -s -X POST "$LANDSCAPE_URL/api/v2/login" \
    --data "$LOGIN_JSON"                                       \
    --header "Content-Type: application/json"                  \
    --header "Accept: application/json" )

JWT=$( echo $LOGIN_RESPONSE | jq .token | tr -d '"')
```

```
$LOGIN_RESPONSE = Invoke-WebRequest -Method POST `
    -URI "$LANDSCAPE_URL/api/v2/login"           `
    -Body "$LOGIN_JSON" -ContentType "application/json"

$JWT = ConvertTo-SecureString -AsPlainText -Force $( $LOGIN_RESPONSE.Content |␣
↪ConvertFrom-Json).token
```

Send the Install request

Build the payload with information about the WSL instance to be deployed. In this case it would look like:

```
{"rootfs_url": "http://landscape.mib.com:9009/ubuntu-24.04-custom.tar.gz", "computer_name
↪": "Carbonizer", "cloud_init": "<base64 encoded material>"}
```

Bash

PowerShell

```
WSL_JSON=$( jq -n                              \
    --arg rf "$ROOTFS_URL"                     \
    --arg cn "$COMPUTER_NAME"                  \
    --arg b64 "$BASE64_ENCODED_CLOUD_INIT"     \
    '{rootfs_url: $rf, computer_name: $cn, cloud_init: $b64}' )
```

```
$WSL_JSON = @{
 rootfs_url = "$ROOTFS_URL"
 computer_name = "$COMPUTER_NAME"
 cloud_init = "$BASE64_ENCODED_CLOUD_INIT"
} | ConvertTo-Json
```

At the moment of this writing there is no specific API endpoint to trigger installation of WSL instances on multiple Windows machines at once. Instead we send one request per target machine.

Bash

PowerShell

```
for COMPUTER_ID in "${PARENT_COMPUTER_IDS[@]}"; do
    API_RESPONSE=$( curl -s -X POST                                   \
        "$LANDSCAPE_URL/api/v2/computers/$COMPUTER_ID/children" \
        --data "$WSL_JSON"                                      \
        --header "Authorization:Bearer $JWT"                    \
        --header "Content-Type: application/json"               \
        --header "Accept: application/json" )

    # show the response
    echo $API_RESPONSE
    echo
done
```

```
foreach ($COMPUTER_ID in $PARENT_COMPUTER_IDS) {
    $API_RESPONSE = Invoke-WebRequest -Method POST -Body "$WSL_JSON" `
        -Uri "$LANDSCAPE_URL/api/v2/computers/$COMPUTER_ID/children" `
        -Authentication Bearer -Token $JWT -ContentType "application/json"

    # show the response
    Write-Output $API_RESPONSE
}
```

When that completes, you'll be able to find activities in the Landscape dashboard about the installation of a new WSL instance for each of the Windows machines listed.

Summarising the steps in a single script

The steps above can be made into a single script:

Bash

PowerShell

```
#!/usr/bin/env bash

# Base64-encoding the cloud-config file contents
BASE64_ENCODED_CLOUD_INIT=$(cat $CLOUD_INIT_FILE | base64 --wrap=0)

# Build the auth payload
LOGIN_JSON=$( jq -n                        \
    --arg em "$LANDSCAPE_USER_EMAIL"       \
    --arg pwd "$LANDSCAPE_USER_PASSWORD" \
```

(continues on next page)

(continued from previous page)

```
    '{email: $em, password: $pwd}' )

# Issue an auth request and retrieve the JWT
LOGIN_RESPONSE=$( curl -s -X POST "$LANDSCAPE_URL/api/v2/login" \
    --data "$LOGIN_JSON"                                       \
    --header "Content-Type: application/json"                  \
    --header "Accept: application/json" )

JWT=$( echo $LOGIN_RESPONSE | jq .token | tr -d '"')

# Build the installation payload
WSL_JSON=$( jq -n                               \
    --arg rf "$ROOTFS_URL"                      \
    --arg cn "$COMPUTER_NAME"                   \
    --arg b64 "$BASE64_ENCODED_CLOUD_INIT"      \
    '{rootfs_url: $rf, computer_name: $cn, cloud_init: $b64}' )

# Issue the command for each Windows machine
for COMPUTER_ID in "${PARENT_COMPUTER_IDS[@]}"; do
    API_RESPONSE=$( curl -s -X POST                                \
        "$LANDSCAPE_URL/api/v2/computers/$COMPUTER_ID/children" \
        --data "$WSL_JSON"                                     \
        --header "Authorization:Bearer $JWT"                   \
        --header "Content-Type: application/json"              \
        --header "Accept: application/json" )

    # show the response
    echo $API_RESPONSE
    echo
done
```

```
# Base64-encoding the cloud-config file contents
$content = Get-Content -Path $CLOUD_INIT_FILE -Raw
$bytes = [System.Text.Encoding]::UTF8.GetBytes($content)
$BASE64_ENCODED_CLOUD_INIT = [System.Convert]::ToBase64String($bytes)

# Build the auth payload
$LOGIN_JSON = @{
 email = "$LANDSCAPE_USER_EMAIL"
 password = "$LANDSCAPE_USER_PASSWORD"
} | ConvertTo-Json

# Issue an auth request and retrieve the JWT
$LOGIN_RESPONSE = Invoke-WebRequest -Method POST `
    -URI "$LANDSCAPE_URL/api/v2/login"            `
    -Body "$LOGIN_JSON" -ContentType "application/json"

$JWT = ConvertTo-SecureString -AsPlainText -Force $( $LOGIN_RESPONSE.Content |␣
↪ConvertFrom-Json).token

# Build the installation payload
$WSL_JSON = @{
```

(continues on next page)

(continued from previous page)

```
 rootfs_url = "$ROOTFS_URL"
 computer_name = "$COMPUTER_NAME"
 cloud_init = "$BASE64_ENCODED_CLOUD_INIT"
} | ConvertTo-Json

# Issue the command for each Windows machine
foreach ($COMPUTER_ID in $PARENT_COMPUTER_IDS) {
    $API_RESPONSE = Invoke-WebRequest -Method POST -Body "$WSL_JSON" `
        -Uri "$LANDSCAPE_URL/api/v2/computers/$COMPUTER_ID/children" `
        -Authentication Bearer -Token $JWT -ContentType "application/json"

    # show the response
    Write-Output $API_RESPONSE
}
```

Further reading

- Visit the Landscape API documentation to learn more about it.
- Landscape documentation about WSL integration contains more information about this and other methods of creating WSL instances on Windows machines registered with Landscape via its REST API.

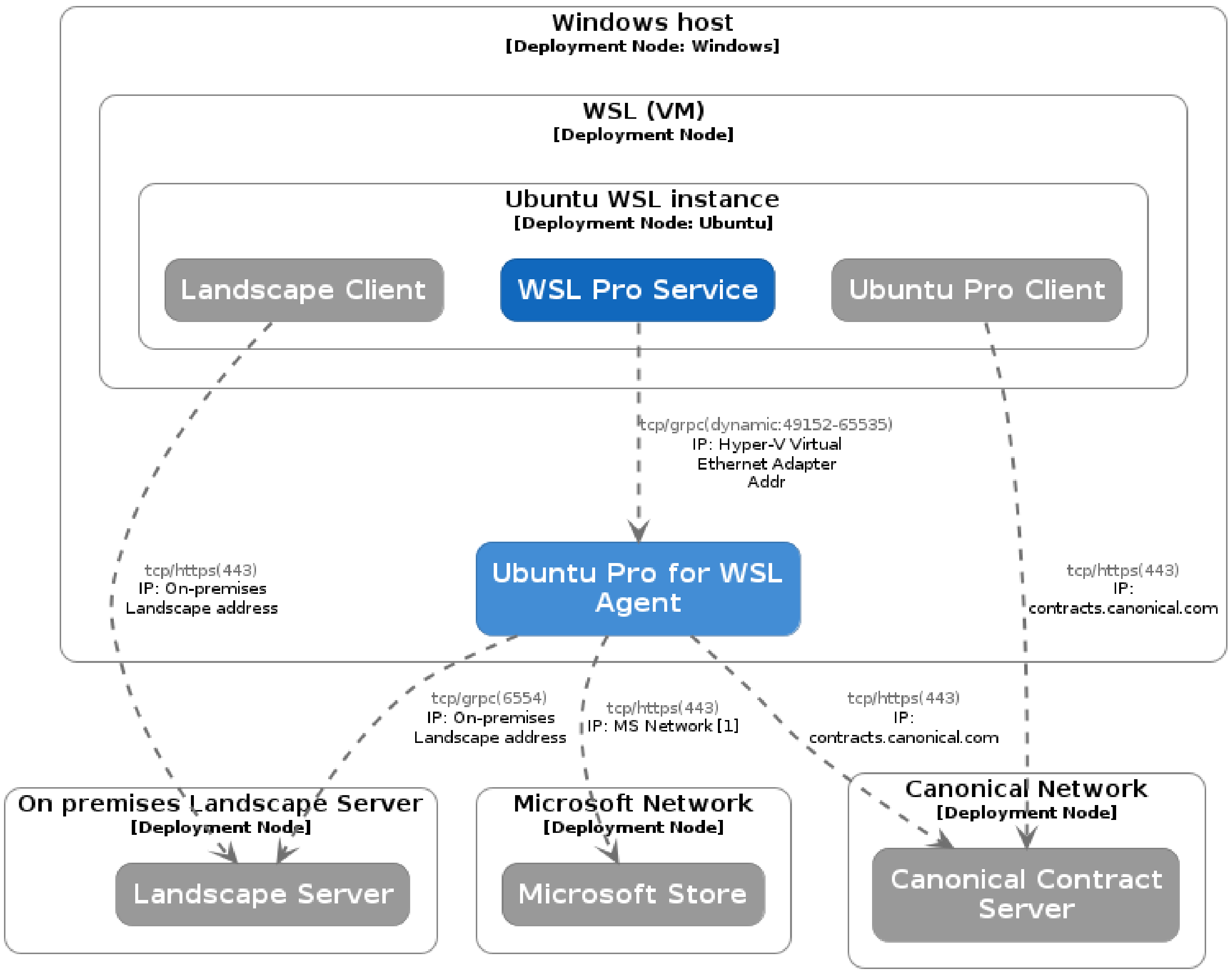
2.3 Reference

Our reference section contains more detailed information about the several pieces that make up the Ubuntu Pro for WSL tool and the value it provides.

2.3.1 Firewall requirements

Firewall rules must be configured for Ubuntu Pro for WSL to operate fully.

The following figure shows the possible connections between the different components and their default ports and protocols:



The following table lists the default ports and protocols used by Ubuntu Pro for WSL:

Description	Client System	Server System	Protocol	Default Port	Target address
Required for online installation of WSL instances[1].	Windows Host / Pro Agent	MS Store	tcp	https (443)	See Microsoft documentation for a list of addresses to allow.
Ubuntu Pro enablement[2]	Windows Host / Pro Agent	Canonical Contract Server	tcp	https (443)	`contracts.canonical.com`
Landscape management[2]	Windows Host / Pro Agent	Landscape Server	tcp	grpc (6554)	On-premise Landscape address
WSL instance management on the Windows host. Firewall rules set up at installation time of the WSL Pro agent.	WSL Instance / wsl-pro-service	Windows Host / Pro Agent	tcp	grpc (dynamic:49 65535)	Hyper-V Virtual Ethernet Adapter IP
Ubuntu Pro[2][3].	WSL Instance / Ubuntu Pro client	Canonical Contract Server	tcp	https (443)	`contracts.canonical.com`
Landscape[2].	WSL Instance / Ubuntu Pro client	Landscape Server	tcp	https (443)	On-premise Landscape address

If the client system is behind a proxy, ensure that the proxy is configured to allow the required connections.

2.3.2 Landscape

Landscape is a systems management tool designed to help you manage and monitor your Ubuntu systems from a unified platform.

See more: Landscape | Documentation

In UP4W, Landscape consists of a remote server and two clients:

1. the usual Ubuntu-side client, in this case a *Landscape client* that comes automatically with any Ubuntu WSL instance, and
2. a Windows-side client, a Landscape client that is built into the *UP4W Windows Agent*.

The latter offers advantages unique to Ubuntu WSL – the ability to create new instances through Landscape and the ability to configure all your instances at scale (when you configure the Windows-side client, the UP4W agent forwards the configuration to the client on each instance).

[1] Access to the Microsoft Store is required for the online installation of WSL instances. Without it Ubuntu Pro for WSL will still be functional but it will only be possible to install WSL instances centrally from Landscape from custom tarballs, not using the official Ubuntu releases.
[2] Access to the contract server and Landscape server is required for proper operation of Ubuntu Pro for WSL.
[3] For air-gapped installation refer to the Ubuntu Pro documentation.

Landscape configuration schema

Both Landscape clients are configured via a single, plain text configuration file (e.g., `landscape.conf` or `landscape.ini`). This file is provided to the Windows host.

The schema for this file is the same as Landscape for Ubuntu desktop or server, with a few additional keys specific to the WSL settings, which can be grouped into keys that affect just the Windows-side client and keys that affect both the Windows-side client and the Ubuntu WSL-side client(s). These additions are documented below.

> See more: Landscape | Configure Ubuntu Pro for WSL for Landscape

Here is an example of what the configuration looks like:

```
[host]
url = landscape-server.domain.com:6554

[client]
url = https://landscape-server.domain.com/message-system
ping_url  = http://landscape-server.domain.com/ping
account_name = standalone
log_level = debug
ssl_public_key = C:\Users\user\Downloads\landscape_server.pem
```

Host

This section contains settings unique to the Windows-side client. Currently these consist of a single key:

- `url`: The URL of your Landscape account followed by a colon (`:`) and the port number. Port 6554 is the default for Landscape Quickstart installations.

Client

This section contains settings used by both clients. Most keys in this section behave the same way they would on a traditional Landscape setup. Only the following keys behave differently:

- `ssl_public_key`: This key must be a Windows path. The WSL instances will have this path translated automatically.
- `computer_title`: This key will be ignored. Instead, each WSL instance will use its Distro name as computer title.
- `hostagent_uid`: This key will be ignored.

  See more: GitHub | Landscape client configuration schema

2.3.3 Landscape (client)

The Landscape client is a `systemd` unit running on *Landscape*-managed Ubuntu machines. It sends information about the system to the Landscape server. The server, in turn, sends instructions that the client executes.

In WSL, there is one Landscape client inside every Ubuntu WSL distro. The Landscape client comes pre-installed in your distro as part of the package `landscape-client`, but it must be configured before it can start running.

> See more: Ubuntu manuals | Landscape client

UP4W will configure all Ubuntu WSL distros for you, so you don't need to configure each WSL instance separately; you specify the configuration once and UP4W will distribute it to every distro.

You can see the status of the Landscape client in any particular Ubuntu WSL instance by starting a shell in that instance and running:

```
systemctl status landscape-client.service
```

2.3.4 Ubuntu Pro

Ubuntu Pro is a subscription service offered by Canonical on top of the Long Term Support (LTS) releases of Ubuntu. It provides access to the following offerings:

- Landscape
- Center for Internet Security (CIS) compliance
- Expanded Security Maintenance (ESM)
- Canonical Support line

  See more: Ubuntu | Ubuntu Pro

Ubuntu Pro services are managed by the *Ubuntu Pro client*.

Ubuntu Pro token

An Ubuntu pro token is a secret string of numbers and letters that acts as proof of purchase for your Ubuntu Pro subscription. Services provided by the Ubuntu Pro subscription will require a token to run.

See more: *Pro-attach*

You can find out what your Ubuntu Pro token is by visiting your Ubuntu Pro dashboard and logging in.

See more: Ubuntu | Ubuntu Pro dashboard

2.3.5 Ubuntu Pro (client)

The Ubuntu Pro client is a command-line utility (a CLI) that manages the different offerings of your Ubuntu Pro subscription. In UP4W, this executable is used within each of the managed WSL distros to enable *Ubuntu Pro* services within that distro.

This executable is provided as part of the `ubuntu-advantage-tools` package, which comes pre-installed in your Ubuntu WSL distros.

See more: Ubuntu manuals | Ubuntu advantage tools

Pro-attach

Pro-attaching a machine (e.g. a desktop computer, a WSL distro, a server, a virtual machine, etc.) means to provide your *Ubuntu Pro token* to the Ubuntu Pro client, so that it can enable Ubuntu Pro services.

2.3.6 Ubuntu WSL

Ubuntu WSL refers to the set of Ubuntu releases that target *WSL*. Each of these releases is distributed as a separate Microsoft Store application. Once installed, each of these releases will run as a separate WSL instance.

See more: Ubuntu WSL | Documentation

2.3.7 Ubuntu Pro for WSL (UP4W)

Ubuntu Pro for WSL (UP4W) is an automation tool running on Windows hosts to manage Ubuntu WSL instances, providing them with compliance by attaching them to your Ubuntu Pro subscription and enrolling them into Landscape.

Some Ubuntu Pro for WSL components run on the Windows host:

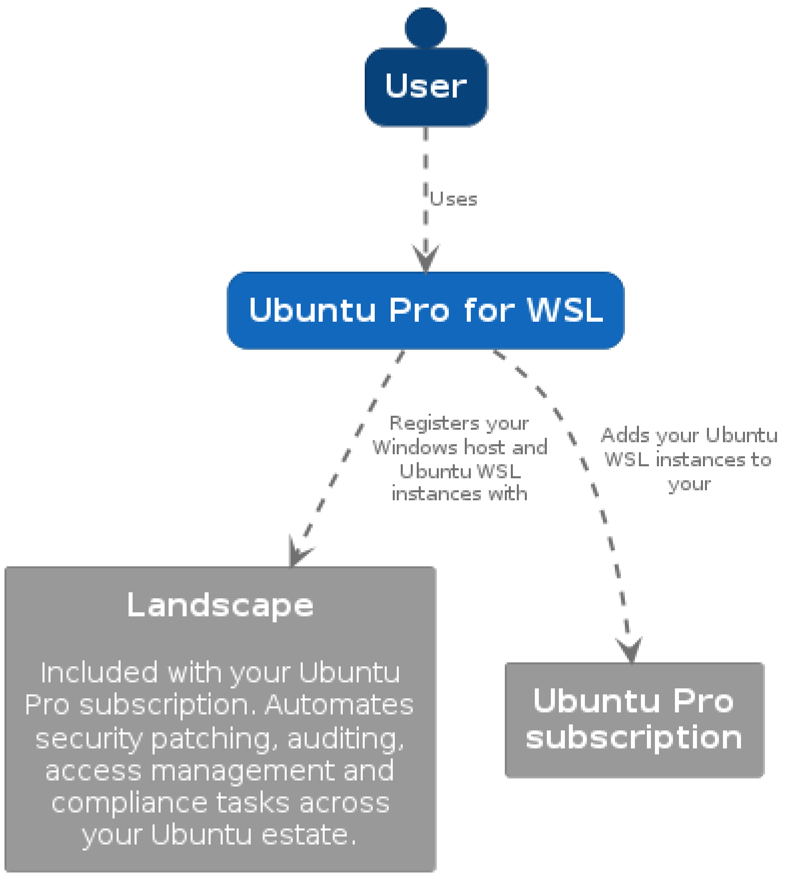
- the *UP4W Windows Agent* providing automation services.

- the *UP4W GUI* for end users to manage their Ubuntu Pro subscription and Landscape configuration.

Additionally, Ubuntu Pro for WSL requires a component running inside each of the WSL distros:

- the *WSL Pro Service* communicates with the Windows Agent to provide automation services.

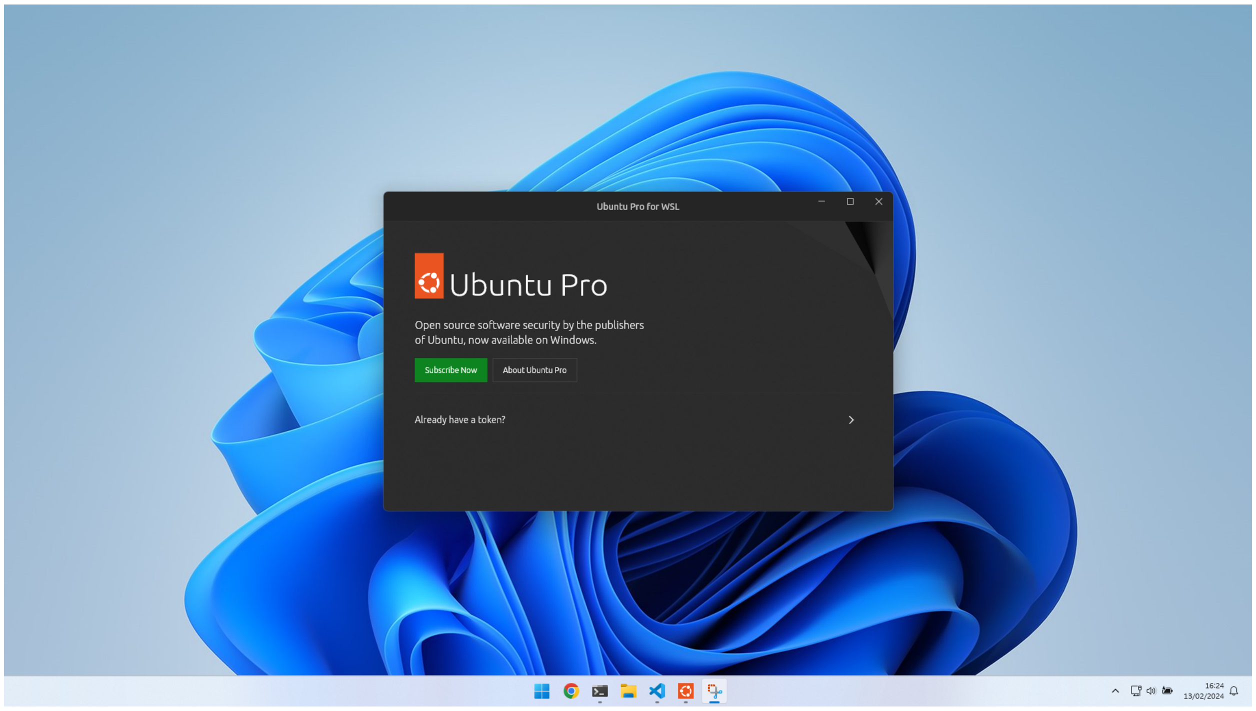
A top-level summary of the architecture is shown below:


User
Uses
Ubuntu Pro for WSL
Registers your Windows host and Ubuntu WSL instances with
Adds your Ubuntu WSL instances to your
Landscape
Included with your Ubuntu Pro subscription. Automates security patching, auditing, access management and compliance tasks across your Ubuntu estate.
Ubuntu Pro subscription

2.3.8 UP4W - Graphical User Interface (GUI)

UP4W has a small GUI to help users with:

- Providing or acquiring an *Ubuntu Pro token*.
- Providing the *Landscape configuration*.



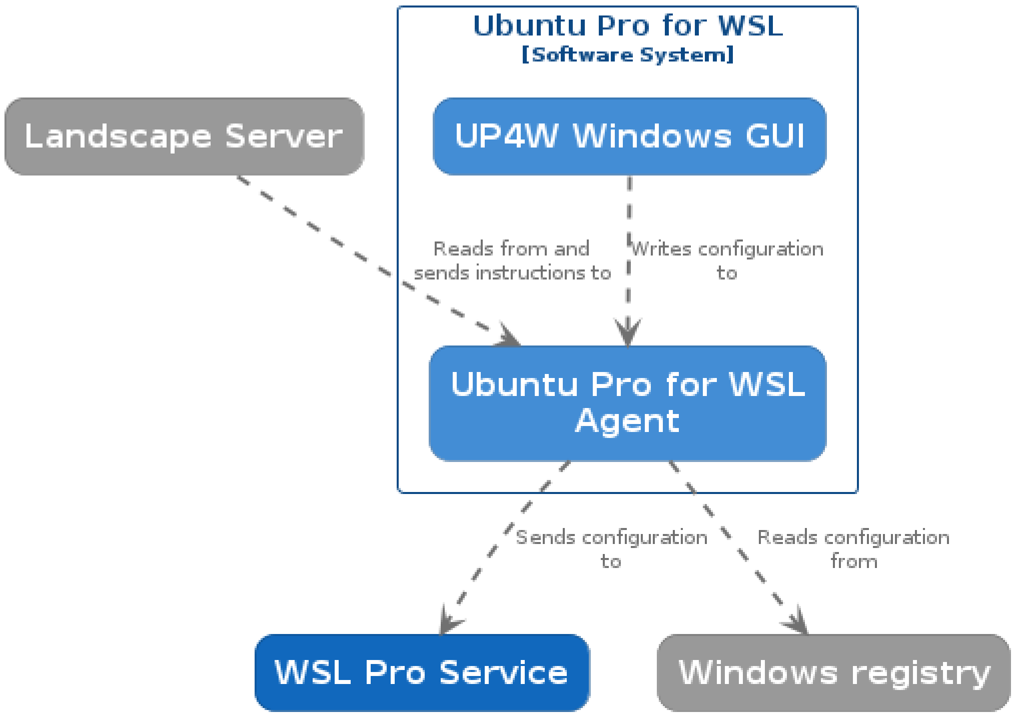
Interaction between the GUI and the agent

When the GUI starts, it attempts to establish a connection to the *UP4W Windows Agent*. If this fails, the agent is restarted. For troubleshooting purposes, you can restart the Agent by stopping the Windows process `ubuntu-pro-agent-launcher.exe` and starting the GUI.

2.3.9 UP4W - Windows Agent

UP4W's Windows Agent is a Windows application running in the background. It starts automatically when the user logs in to Windows. If it stops for some reason, it can also be started by launching the UP4W GUI.

The Windows agent is UP4W's central hub that communicates with all the components to coordinate them.



2.3.10 UP4W - WSL Pro service

A `systemd` unit running inside every Ubuntu WSL instance. The *Windows Agent* running on the Windows host sends commands that the WSL Pro Service executes, such as *pro-attaching* or configuring the *Landscape client*.

You can check the current status of the WSL Pro Service in any particular distro with:

```
systemctl status wsl-pro.service
```

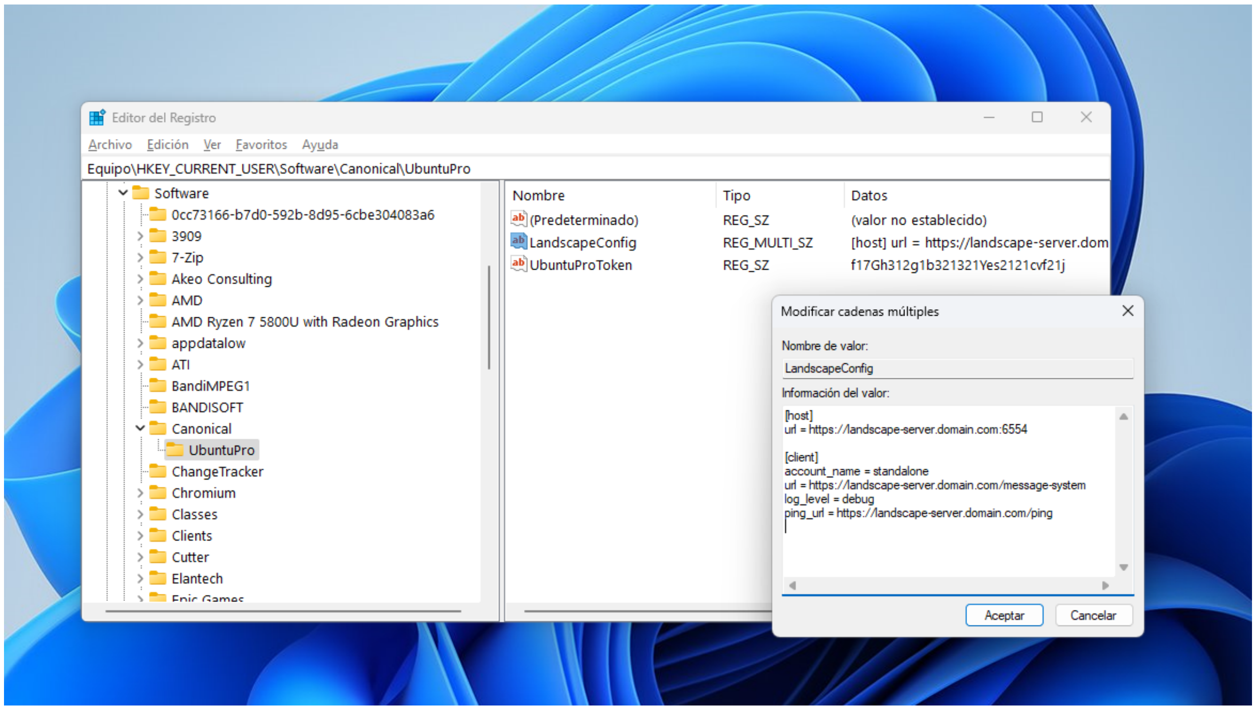
2.3.11 Windows registry

The Windows registry is a database provided by Windows where programs can read and write information. UP4W uses it as a read-only source of configuration.

See more: Microsoft Learn | Windows registry information for advanced users

In UP4W, you can use the Windows registry to supply the configuration for *Ubuntu Pro* and *Landscape* to the *Windows Agent*.

See more: *install UP4W and add a Pro token*



Equipo\HKEY_CURRENT_USER\Software\Canonical\UbuntuPro

Nombre	Tipo	Datos
(Predeterminado)	REG_SZ	(valor no establecido)
LandscapeConfig	REG_MULTI_SZ	[host] url = https://landscape-server.dom
UbuntuProToken	REG_SZ	f17Gh312g1b321321Yes2121cvf21j

Nombre de valor: LandscapeConfig

Información del valor:

```
[host]
url = https://landscape-server.domain.com:6554

[client]
account_name = standalone
url = https://landscape-server.domain.com/message-system
log_level = debug
ping_url = https://landscape-server.domain.com/ping
```

Expected contents of the UbuntuPro registry key

The Windows agent will read the following values from the key at `HK_CURRENT_USER\Software\Canonical\UbuntuPro`:

- Value `UbuntuProToken` (type `String`) expects the *Ubuntu Pro token* for the user.
- Value `LandscapeConfig` (type `String` or `Multi-line string`) expects the *Landscape configuration*.

2.3.12 Windows Subsystem for Linux (WSL)

Windows Subsystem for Linux (WSL) is a Microsoft product, distributed as part of Windows 10 and Windows 11. It allows the user to run a Linux environment on the Windows machine, without the need for a traditional virtual machine or a dual boot setup.

See more: WSL documentation

2.4 Ubuntu Pro for WSL - developer documentation

Welcome

2.4.1 How-to guides

These how-to guides cover key operations and processes in Ubuntu Pro for WSL.

How to install UP4W

This guide will show you how to install UP4W for local development and testing.

Requirements:

- A Windows machine with access to the internet
- Appx from the Microsoft Store:

  - Windows Subsystem For Linux
  - Either Ubuntu, Ubuntu 22.04, or Ubuntu (Preview)
- The Windows Subsystem for Windows optional feature enabled

1. Download the Windows Agent and the WSL Pro Service

1. Go to the repository actions page.
2. Click the latest successful workflow run.
3. Scroll down past any warnings or errors, until you reach the Artifacts section.
4. Download:
   - Windows agent: UbuntuProForWSL+…-production
   - wsl-pro-service: Wsl-pro-service_…

Notice that, for the step above, there is also an alternative version of the MSIX bundle enabled for end-to-end testing. Most likely, that's not what you want to download.

2. Install the Windows Agent

This is the Windows-side agent that manages the distros.

1. Uninstall Ubuntu Pro for WSL if you had installed previously:

```
Get-AppxPackage -Name CanonicalGroupLimited.UbuntuPro | Remove-AppxPackage
```

2. Follow the download steps to download UbuntuProForWSL
3. Unzip the artefact
4. Find the certificate inside. Install it into `Local Machine/Trusted people`.
5. Double click on the MSIX bundle and complete the installation.
6. The Firewall may ask for an exception. Allow it.
7. The GUI should show up. You're done.

3. Install the WSL Pro Service

This is the Linux-side component that talks to the agent. Choose one or more distros Jammy or greater, and follow the instructions.

1. Uninstall the WSL-Pro-Service from your distro if you had it installed previously:

```
sudo apt remove wsl-pro-service
```

2. Follow the download steps to download the WSL-Pro-Service.
3. Unzip the artifact.
4. Navigate to the unzipped directory containing the .deb file. Here is a possible path:

```
cd /mnt/c/Users/WINDOWS-USER/Downloads/wsl-pro-service_*
```

5. Install the deb package.

```
sudo apt install ./wsl-pro-service_*.deb
```

6. Ensure it works via systemd:

```
systemctl status wsl-pro.service
```

How to restart UP4W

Some configuration changes only apply when you restart UP4W. Here is a guide on how to restart it. There are two options.

Option 1: Restart your UP4W host machine

This is the simple one. If you're not in a hurry to see the configuration updated, just wait until next time you boot your machine.

Option 2: Restart only UP4W

1. Stop the agent:

```
Get-Process -Name Ubuntu-Pro-Agent | Stop-Process
```

2. Stop the distro, or distros you installed WSL-Pro-Service in:

```
wsl --terminate DISTRO_NAME_1
wsl --terminate DISTRO_NAME_2
# etc.

# Alternatively, stop all distros:
wsl --shutdown
```

3. Start the agent again:
   1. Open the start Menu and search for "Ubuntu Pro for WSL".
   2. The GUI should start.
   3. Wait a minute.
   4. Click on "Click to restart it".
4. Start the distro, or distros you installed WSL-Pro-Service in.

How to access UP4W logs

At some point you may want to read the UP4W logs, most likely for debugging purposes. The agent and the service store their logs separately. This guide shows you where to find each of the logs.

Access the logs for the WSL Pro service

To access the logs of a specific distribution's WSL-Pro-Service, you must first launch the distribution and then query the journal:

```
journalctl -u wsl-pro.service
```

For more information on using the journal, you can check out its man page with `man journalctl` or online.

These logs may be insufficient for proper debugging, so you may be interested in looking at the agent's logs as well.

Access the logs for the Windows Agent

To access the logs for the Windows Agent:

1. Go to your home directory
   - Open the file explorer
   - Write `%USERPROFILE%` at the address
2. In the home directory, find the `.ubuntupro` directory and double-click on it.
3. In the `.ubuntupto` folder, find file `log` and open it with any text editor.
   - This file contains the logs sorted with the oldest entries at the top and the newest at the bottom.

How to enable opt-in features

Some features in UP4W are opt-in or can be toggled on and off via the Windows Registry. While the code is arranged such that CI always tests with those features enabled, when running UP4W on your machine, you may need to toggle them on and off explicitly via the Windows registry. This guide shows you how to do that.

Enable subscribing via the Microsoft Store

1. Open the Windows registry editor: press `Win + R`, type `regedit` and press `Enter`.
2. Navigate to the following key: `HKEY_CURRENT_USER\Software\Canonical\UbuntuPro\`.
3. Create a new DWORD value named `AllowStorePurchase` and set it to 1.

The next time you open the GUI you'll find the button to subscribe to Ubuntu Pro via the Microsoft Store.

Warning

Beware that can incur in real charges if you proceed with the purchase.

Disable Landscape configuration in the GUI

Landscape configuration page and related buttons are enabled by default, but can be disabled via registry.

1. Open the Windows registry editor: press `Win + R`, type `regedit` and press `Enter`.
2. Navigate to the following key: `HKEY_CURRENT_USER\Software\Canonical\UbuntuPro\`.
3. Create a new DWORD value named `LandscapeConfigVisibility` and set it to `0`.

The next time you open the GUI, you'll find that the Landscape configuration page can be shown via the set up wizard or by clicking on the 'Configure Landscape' button. If that value is not present or set to anything other than `0`, Landscape configuration page and related buttons will be visible.

How to reset UP4W back to factory settings

You can reset Ubuntu Pro for WSL to factory settings following these steps:

1. Shut down WSL

   ```
   wsl --shutdown
   ```

2. Uninstall the package and shut down WSL:

```
Get-AppxPackage -Name "CanonicalGroupLimited.UbuntuPro" | Remove-AppxPackage`
```

3. Remove the public directory:

```
Remove-Item -Recurse -Force "${env:UserProfile}\.ubuntupro\"
```

4. Remove the registry key:
   1. Press Win+R
   2. Type `regedit.exe` and click OK
   3. Write `HKEY_CURRENT_USER\Software\Canonical\UbuntuPro` at the address bar
      - If this fails, you are done (the key does not exist).
   4. Find the `UbuntuPro` key on the left
   5. Right-click on it
   6. Click delete
5. Install the Windows Agent package again (see the section on *how to install*). You do not need to re-install the WSL-Pro-Service.
6. You're done. Next time you start the GUI it'll be like a fresh install.

2.4.2 Reference

The reference material in this section provides technical descriptions of Ubuntu Pro for WSL.

Windows Agent CLI

See first: *UP4W - Windows Agent*

Usage

User commands

ubuntu-pro-agent

Ubuntu Pro for WSL agent

Synopsis

Ubuntu Pro for WSL agent for managing your pro-enabled distro.

```
ubuntu-pro-agent COMMAND [flags]
```

Options

```
  -c, --config string     configuration file path
  -h, --help              help for ubuntu-pro-agent
  -v, --verbosity count   issue INFO (-v), DEBUG (-vv) or DEBUG with caller (-vvv) output
```

ubuntu-pro-agent clean

Removes all the agent's data and exits

```
ubuntu-pro-agent clean [flags]
```

Options

```
  -h, --help   help for clean
```

Options inherited from parent commands

```
  -c, --config string     configuration file path
  -v, --verbosity count   issue INFO (-v), DEBUG (-vv) or DEBUG with caller (-vvv) output
```

ubuntu-pro-agent completion

Generate the autocompletion script for the specified shell

Synopsis

Generate the autocompletion script for ubuntu-pro-agent for the specified shell. See each sub-command's help for details on how to use the generated script.

Options

```
  -h, --help   help for completion
```

Options inherited from parent commands

```
  -c, --config string     configuration file path
  -v, --verbosity count   issue INFO (-v), DEBUG (-vv) or DEBUG with caller (-vvv) output
```

ubuntu-pro-agent completion bash

Generate the autocompletion script for bash

Synopsis

Generate the autocompletion script for the bash shell.

This script depends on the 'bash-completion' package. If it is not installed already, you can install it via your OS's package manager.

To load completions in your current shell session:

```
source <(ubuntu-pro-agent completion bash)
```

To load completions for every new session, execute once:

Linux:

```
ubuntu-pro-agent completion bash > /etc/bash_completion.d/ubuntu-pro-agent
```

macOS:

```
ubuntu-pro-agent completion bash > $(brew --prefix)/etc/bash_completion.d/ubuntu-pro-
↪agent
```

You will need to start a new shell for this setup to take effect.

```
ubuntu-pro-agent completion bash
```

Options

```
  -h, --help              help for bash
      --no-descriptions   disable completion descriptions
```

Options inherited from parent commands

```
  -c, --config string     configuration file path
  -v, --verbosity count   issue INFO (-v), DEBUG (-vv) or DEBUG with caller (-vvv) output
```

ubuntu-pro-agent completion fish

Generate the autocompletion script for fish

Synopsis

Generate the autocompletion script for the fish shell.

To load completions in your current shell session:

```
ubuntu-pro-agent completion fish | source
```

To load completions for every new session, execute once:

```
ubuntu-pro-agent completion fish > ~/.config/fish/completions/ubuntu-pro-agent.fish
```

You will need to start a new shell for this setup to take effect.

```
ubuntu-pro-agent completion fish [flags]
```

Options

```
  -h, --help              help for fish
      --no-descriptions   disable completion descriptions
```

Options inherited from parent commands

```
  -c, --config string     configuration file path
  -v, --verbosity count   issue INFO (-v), DEBUG (-vv) or DEBUG with caller (-vvv) output
```

ubuntu-pro-agent completion powershell

Generate the autocompletion script for powershell

Synopsis

Generate the autocompletion script for powershell.

To load completions in your current shell session:

```
ubuntu-pro-agent completion powershell | Out-String | Invoke-Expression
```

To load completions for every new session, add the output of the above command to your powershell profile.

```
ubuntu-pro-agent completion powershell [flags]
```

Options

```
  -h, --help              help for powershell
      --no-descriptions   disable completion descriptions
```

Options inherited from parent commands

```
  -c, --config string     configuration file path
  -v, --verbosity count   issue INFO (-v), DEBUG (-vv) or DEBUG with caller (-vvv) output
```

ubuntu-pro-agent completion zsh

Generate the autocompletion script for zsh

Synopsis

Generate the autocompletion script for the zsh shell.

If shell completion is not already enabled in your environment you will need to enable it. You can execute the following once:

```
echo "autoload -U compinit; compinit" >> ~/.zshrc
```

To load completions in your current shell session:

```
source <(ubuntu-pro-agent completion zsh)
```

To load completions for every new session, execute once:

Linux:

```
ubuntu-pro-agent completion zsh > "${fpath[1]}/_ubuntu-pro-agent"
```

macOS:

```
ubuntu-pro-agent completion zsh > $(brew --prefix)/share/zsh/site-functions/_ubuntu-pro-
↪agent
```

You will need to start a new shell for this setup to take effect.

```
ubuntu-pro-agent completion zsh [flags]
```

Options

```
  -h, --help              help for zsh
      --no-descriptions   disable completion descriptions
```

Options inherited from parent commands

```
  -c, --config string     configuration file path
  -v, --verbosity count   issue INFO (-v), DEBUG (-vv) or DEBUG with caller (-vvv) output
```

ubuntu-pro-agent version

Returns version of agent and exits

```
ubuntu-pro-agent version [flags]
```

Options

```
  -h, --help   help for version
```

Options inherited from parent commands

```
  -c, --config string     configuration file path
  -v, --verbosity count   issue INFO (-v), DEBUG (-vv) or DEBUG with caller (-vvv) output
```

Hidden commands

Those commands are hidden from help and should primarily be used by the system or for debugging.

WSL Pro Service CLI

See first: *UP4W - WSL Pro Service*

Usage

User commands

wsl-pro-service

WSL Pro Service

Synopsis

WSL Pro Service connects Ubuntu Pro for WSL agent to your distro.

```
wsl-pro-service COMMAND [flags]
```

Options

```
  -c, --config string     configuration file path
  -h, --help              help for wsl-pro-service
  -v, --verbosity count   issue INFO (-v), DEBUG (-vv) or DEBUG with caller (-vvv) output
```

wsl-pro-service completion

Generate the autocompletion script for the specified shell

Synopsis

Generate the autocompletion script for wsl-pro-service for the specified shell. See each sub-command's help for details on how to use the generated script.

Options

```
  -h, --help   help for completion
```

Options inherited from parent commands

```
  -c, --config string     configuration file path
  -v, --verbosity count   issue INFO (-v), DEBUG (-vv) or DEBUG with caller (-vvv) output
```

wsl-pro-service completion bash

Generate the autocompletion script for bash

Synopsis

Generate the autocompletion script for the bash shell.

This script depends on the 'bash-completion' package. If it is not installed already, you can install it via your OS's package manager.

To load completions in your current shell session:

```
source <(wsl-pro-service completion bash)
```

To load completions for every new session, execute once:

Linux:

```
wsl-pro-service completion bash > /etc/bash_completion.d/wsl-pro-service
```

macOS:

```
wsl-pro-service completion bash > $(brew --prefix)/etc/bash_completion.d/wsl-pro-service
```

You will need to start a new shell for this setup to take effect.

```
wsl-pro-service completion bash
```

Options

```
  -h, --help              help for bash
      --no-descriptions   disable completion descriptions
```

Options inherited from parent commands

```
  -c, --config string     configuration file path
  -v, --verbosity count   issue INFO (-v), DEBUG (-vv) or DEBUG with caller (-vvv) output
```

wsl-pro-service completion fish

Generate the autocompletion script for fish

Synopsis

Generate the autocompletion script for the fish shell.

To load completions in your current shell session:

```
wsl-pro-service completion fish | source
```

To load completions for every new session, execute once:

```
wsl-pro-service completion fish > ~/.config/fish/completions/wsl-pro-service.fish
```

You will need to start a new shell for this setup to take effect.

```
wsl-pro-service completion fish [flags]
```

Options

```
  -h, --help              help for fish
      --no-descriptions   disable completion descriptions
```

Options inherited from parent commands

```
  -c, --config string     configuration file path
  -v, --verbosity count   issue INFO (-v), DEBUG (-vv) or DEBUG with caller (-vvv) output
```

wsl-pro-service completion powershell

Generate the autocompletion script for powershell

Synopsis

Generate the autocompletion script for powershell.

To load completions in your current shell session:

```
wsl-pro-service completion powershell | Out-String | Invoke-Expression
```

To load completions for every new session, add the output of the above command to your powershell profile.

```
wsl-pro-service completion powershell [flags]
```

Options

```
  -h, --help              help for powershell
      --no-descriptions   disable completion descriptions
```

Options inherited from parent commands

```
  -c, --config string     configuration file path
  -v, --verbosity count   issue INFO (-v), DEBUG (-vv) or DEBUG with caller (-vvv) output
```

wsl-pro-service completion zsh

Generate the autocompletion script for zsh

Synopsis

Generate the autocompletion script for the zsh shell.

If shell completion is not already enabled in your environment you will need to enable it. You can execute the following once:

```
echo "autoload -U compinit; compinit" >> ~/.zshrc
```

To load completions in your current shell session:

```
source <(wsl-pro-service completion zsh)
```

To load completions for every new session, execute once:

Linux:

```
wsl-pro-service completion zsh > "${fpath[1]}/_wsl-pro-service"
```

macOS:

```
wsl-pro-service completion zsh > $(brew --prefix)/share/zsh/site-functions/_wsl-pro-
↪service
```

You will need to start a new shell for this setup to take effect.

```
wsl-pro-service completion zsh [flags]
```

Options

```
  -h, --help              help for zsh
      --no-descriptions   disable completion descriptions
```

Options inherited from parent commands

```
  -c, --config string     configuration file path
  -v, --verbosity count   issue INFO (-v), DEBUG (-vv) or DEBUG with caller (-vvv) output
```

wsl-pro-service version

Returns version of agent and exits

```
wsl-pro-service version [flags]
```

Options

```
  -h, --help   help for version
```

Options inherited from parent commands

```
  -c, --config string     configuration file path
  -v, --verbosity count   issue INFO (-v), DEBUG (-vv) or DEBUG with caller (-vvv) output
```

Hidden commands

Those commands are hidden from help and should primarily be used by the system or for debugging.

QA Process

Generalities

`wsl-pro-service` is seeded only on WSL images.

```
Build-dep: golang-go (\>= 2:1.21\~)
```

At any point in time, only the latest two versions of the Go toolchain receive security patches. Hence, we need to keep backporting new releases to fix vulnerabilities. They follow an approximate 6-month release cycle, so Go 1.21 should fall out of support by August 2024.

Process

WSL Pro Service follows a robust continuous integration and testing process. It is covered by a comprehensive automated test suite.

The team applies the following quality criteria:

- All changes are thoroughly reviewed and approved by core team members before integration.
- Each change is thoroughly tested at the unit, integration and system levels. All the tests pass in all supported architectures.
- Releases are reviewed as part of the SRU exception.

The test plan is **completely automated** and runs **every time a change is merged**, as well as **during packaging**. This covers integration and end-to-end tests. Integration tests run on each LTS affected by the SRU to ensure compatibility.

Testing also covers the upgrade from the current version to the proposed version.

Tests are not executed on different versions of Windows due to testing environment limitations.

Packaging QA

To prepare the release to LTS, the following procedure is being completed to ensure quality:

- All autopkgtests pass. Unit tests are executed as autopkgtests. Running higher-level tests would require a Windows VM. It is not available in autopkgtest at the moment. Even if wsl-pro-service tests could run in a VM, they wouldn't test anything real.
- The package does not break when upgrading.
- The binary is identical to the CI build, with only Debian packaging changes.
- The copyrights and changelog are up to date.
- An upgrade test from the previous package version has been performed using apt install/upgrade.

Code sanity

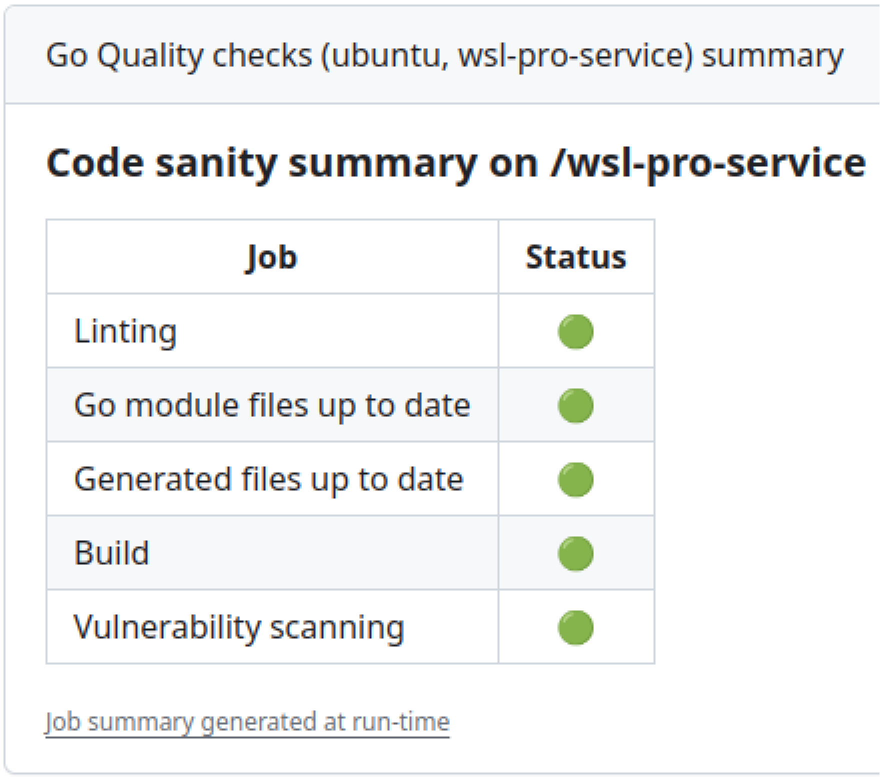
Code sanity checks are performed automatically on each build. They verify:

- Code linting.
- Go module files are up to date.
- Generated files are up to date.
- Any binary in the project builds.
- The Debian package builds.
- Vulnerabilities. It is a run of `govulncheck`.

All the layers are tested from APIs to mocks to the service itself

Example reports

- Code sanity and unit testing: QA workflow
- Integration tests: end-to-to-end tests workflow

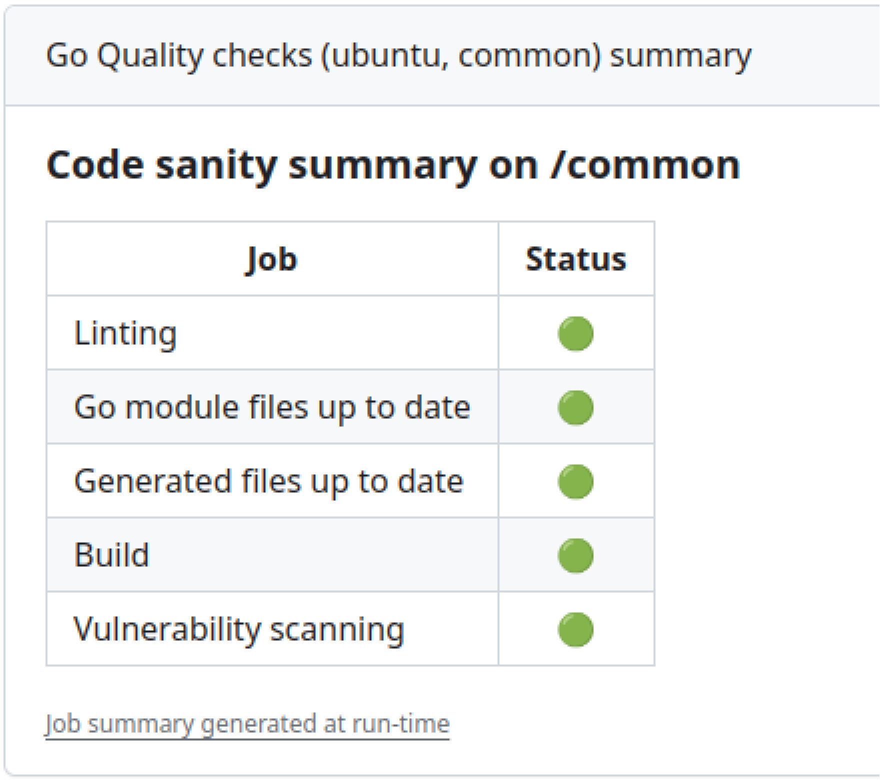


Go Quality checks (ubuntu, wsl-pro-service) summary

Code sanity summary on /wsl-pro-service

Job	Status
Linting	🟢
Go module files up to date	🟢
Generated files up to date	🟢
Build	🟢
Vulnerability scanning	🟢

Job summary generated at run-time



Go Quality checks (ubuntu, common) summary

Code sanity summary on /common

Job	Status
Linting	🟢
Go module files up to date	🟢
Generated files up to date	🟢
Build	🟢
Vulnerability scanning	🟢

Job summary generated at run-time

Code coverage

There is no Codecov report due to the limitations of private projects. However, code coverage is calculated and displayed at testing time. Coverage is manually reviewed by the engineers.

Bug reporting

The main bug tracker remains on GitHub. GitHub Templates are available to help the user with the bug-reporting process and provide the right information.

Wsl-pro supports ubuntu-bug reporting to Launchpad with an apport hook but we are not collecting any data at the moment.

References

- Project Documentation
- Ubuntu Pro for WSL SRU exception
- Ubuntu Pro tools SRU exception