Canonical

Integration of Ubuntu Desktop with Microsoft Active Directory

January 2024

Executive Summary

Microsoft Active Directory is a widely-deployed directory service that is commonly used for identity management and authentication across the enterprise. Its ubiquity makes it a fixture of the IT landscape. As a result, interoperability with Active Directory is often a necessity when deploying services based on non-Microsoft operating systems.

We have worked to make Ubuntu as integrated as possible with Active Directory to streamline the experience of IT administrators and security teams alike. Ubuntu has been engineered to support standard operations like login/logout through SSSD, as well as more advanced flows like Group Policy Objects (GPOs), script execution, network shares, certificate enrollment, apparmor profiles and proxy settings through ADsys.

System Security Services Daemon (SSSD) and ADSys are tools whose purpose is precisely to ease integration of non-Microsoft operating systems into an existing Active Directory architecture. SSSD automates a number of settings that previously required time-consuming and error-prone manual configuration, while ADSys enables system administrators to manage and control Ubuntu Desktop clients from a central Microsoft Active Directory. Together these tools enable integration of Ubuntu with Active Directory quick and painless as it takes the guesswork out of the process.

It is important to understand that Active Directory was never meant to be a cross-platform directory service in the first place. From the ground up, it was built with Microsoft operating systems and software in mind. As such, complete Active Directory integration of third-party operating systems, such as Ubuntu, is not straightforward. The identity management component of Active Directory (authentication of users and groups) is open enough for tools such as SSSD and ADSys to achieve a functional level of interoperability. However, do not assume further Active Directory tasks, such as system management and extensive provisioning for example, are interoperable with Ubuntu.

Extended Features with Ubuntu Pro Desktop

The extended Active Directory integration through the use of ADSys is only available through an active Ubuntu Pro subscription, while basic integration with Active Directory is still freely available.

Features	Free	Ubuntu Pro
Join an Active Directory domain	✓	✓
Authenticate using Active Directory	✓	✓
Manage desktop settings (dconf support)	✓	✓
Privilege management		✓
Remote script execution		✓
Manage apparmor profiles		✓
Mount network shares		✓
Configure proxy settings		✓
Certificate autoenrollment		✓

Table of contents

Introduction

Overview

This whitepaper discusses the use of System Security Services Daemon (SSSD) and the extended Active Directory features (ADSys) on Ubuntu 22.04 LTS. We will demonstrate how an Ubuntu desktop can be configured to enable Active Directory users to login using SSSD and how ADSys implements host configuration via Active Directory and Group Policy Objects (GPO). We will also cover the new Active Directory script execution feature, which enables an administrator to take their powers to new levels.

The overview sections explain what SSSD, ADSys, and its companion tools are, and briefly presents its architecture with an eye toward understanding how it fulfils its functions in Ubuntu.

Pre-requisites and configuration checklists for Active Directory, Ubuntu members, and the service itself are discussed in fairly sequential order. 'Setting up Active Directory', 'Joining Ubuntu to an Active Directory Domain' are necessary to use any of the policy managers provided. To use the extended features of ADSys and script execution, you will also need an active Ubuntu Pro subscription. These sections are not specific to our examples; it holds true for any deployment of SSSD and ADSys, and can be used as a starting point of your own.

The section 'Troubleshooting SSSD' presents some details about gathering information and understanding what is going on when the service is malfunctioning.

We will take an in-depth look at ADsys, its setup process, and how its features can be used to unlock some of the familiar Windows desktop management commands on ubuntu.

Finally, we present some best practices, troubleshooting, and further reading for the ever curious administrator.

Intended audience

This whitepaper has been written with Windows system administrators new to Ubuntu in mind. We assume a basic level of knowledge of administering Ubuntu, including the ability to use a command-line shell, understanding of sudo as a means for privilege escalation and the ability to use a text editor to edit configuration files.

Overview of SSSD

The purpose of System Security Services Daemon (SSSD) is to simplify integration of non-Microsoft systems (Linux in particular) into Microsoft Active Directory. It is a collection of daemons (long-running system services) that handle authentication, authorisation, and user and group information from a variety of network providers. Its purpose is to act as a gateway to Microsoft domains for authentication and identity resolution of users, and to provide consistent mapping of users and groups. It basically enables Microsoft domain users and groups to appear to be local on the non-Microsoft system. This is very useful in a number of scenarios.

At its core, it has support for:

- Active Directory
- LDAP
- Kerberos

To integrate these remote sources as sources of local users and groups on Linux, recognised as valid users, including group membership, requires a fair bit of effort. The Name Service Switch (NSS) framework needs to be configured to resolve users and groups. The Pluggable Authentication Module (PAM) stack similarly needs to be configured to funnel authentication requests. SSSD consolidates all these operations in a consistent suite of tools and delivers a clean configuration for the common use-case in a few easy steps.

SSSD, through a PAM module, provides a generic mechanism for system services on Ubuntu to validate user's credentials against Active Directory. This preempts the need to keep multiple redundant authentication databases by centralising user accounts management, and enables organisations to make full use of their existing Active Directory infrastructure and know-how when deploying Ubuntu.

Once SSSD is installed and configured, users from the Active Directory will appear as if they are local to the Ubuntu system. User attributes that are standard in Unix/Linux but not present in Active Directory are either generated on the fly (i.e; the numerical user ID), or through configuration directive (home directory location and preferred user shell.) In the same manner, groups from the Active Directory will also appear to be regular Unix groups. This is achieved through a Name Service Switch (NSS) module, a mechanism that is standard across all Linux distributions.

SSSD allows disconnected operations by caching this information, so that users can continue to login in the event of a network failure, or other problem of the same sort.

The SSSD tool suite provides command-line tools to manage the services. The command-line tools '`realm`' makes joining an Active Directory domain very straightforward and provides various sanity checks on the Ubuntu computer's configuration - more on that later. Under the hood, SSSD is running several daemons called `sssd`, `sssd_pam`, `sssd_nss`, etc for each service it provides. The SSSD service provides a service control architecture for starting and stopping all `sssd` daemons and drivers based on dependencies. The `sssd` daemon itself is managed using a standard systemd unit.

Advantages of SSSD

- No software to install on the Active Directory, and no change to its configuration required.
- Centralised authentication use for existing users and groups when deploying Ubuntu, no need to maintain a duplicate user database.
- Unix user and group IDs are coherent across all machines running SSSD, no need to maintain an ID map.
- Offline mode allows disconnected users to authenticate using their cached Active Directory credentials.
- SSSD can work with multiple authentication sources.

Overview of ADSys

ADSys: Active Directory Group Policy integration

ADSys is the Active Directory Group Policy client for Ubuntu. It provides system administrators the ability to manage and control Ubuntu Desktop clients from a central Microsoft Active Directory. The project contains everything you need to integrate Ubuntu into your Active Directory, including .admx and .adml template files.

There are multiple policy managers for different types of settings. As of now, only a dconf manager is available for Linux.

The role of ADSys is solely the configuration of the host via Active Directory. Authentication of the users, the initial security policy of the "Default Domain Policy", and creation of the home directory are still the responsibility of SSSD and PAM.

Once an Ubuntu client is configured, Active Directory Group Policies are applied on boot for the machine and at login time for each user, then refreshed periodically.

It is composed of 2 parts:

- The daemon - `adsysd` - implements the Group Policy protocol. It relies on Kerberos, Samba and LDAP for authentication and policy retrieval.

- The command line interface - `adsysctl` - controls the daemon and its status.

Supported releases

ADSys is supported on Ubuntu starting from 22.04.2 LTS, and tested with Windows Server 2019.

Ubuntu supports Active Directory Domain Services and Azure Active Directory Domain Services. Azure Active Directory (AAD) is not yet supported on an LTS release, but on the roadmap (No ETA at the moment).

Setting up Active Directory

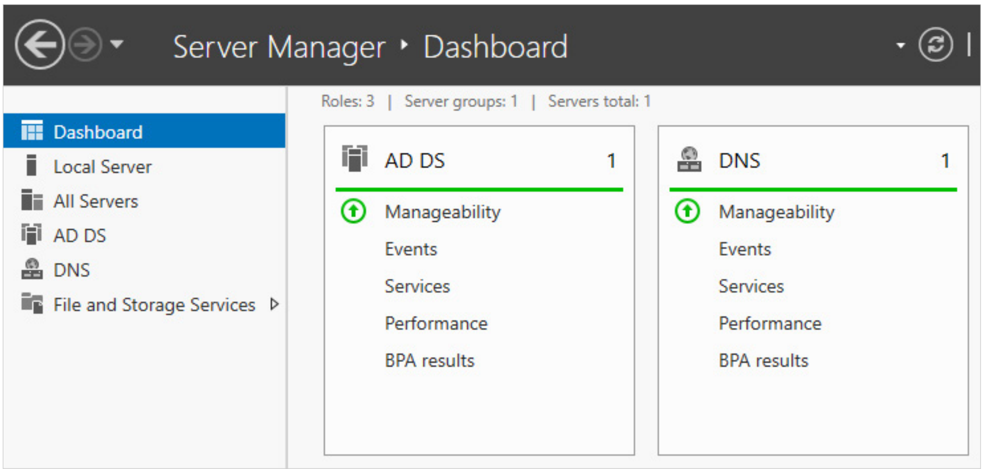
On the Active Directory server you have to verify that you have an account with enough privileges to join a machine to a domain, that the Active Directory service and the DNS are configured and operational, and that the user accounts have been added to the directory.

Administrative privileges

You will need an account with sufficient privileges to add the Ubuntu computers to the Active Directory. Typically, this would be an account member of the Domain Administrator group (such as the ubiquitous Administrator account), although this can vary according to your Active Directory configuration.

Active Directory and DNS

Active Directory and a DNS must be installed, configured, and running on your domain controller. Depending on your network configuration, the DNS can run on a separate server but you will need to ensure that a DNS entry has been created for the machine in question.



Windows Server 2019 Dashboard

Another setup is to run the DHCP on the same server as Active Directory and the DNS. By doing so, DNS records will be updated when a DHCP lease is given.

From above, we see that there are different configurations possible to run the services and it will all depend on the setup of your own organisation.

While not strictly a requirement, it is better to have a reverse lookup zone configured - containing pointer (PTR) records - for your domain in the Active Directory DNS, as many services in Linux do make use of reverse lookup. From the Ubuntu command line, you can easily check if the reverse lookups have been configured properly by using the dig or the host command.

First, verify that the hostname can be resolved:

```
ubuntu@ad-desktop-1:~$ host ad-desktop-1.warthogs.biz
ad-desktop-1.warthogs.biz has address 10.148.231.109
```

Then verify that the reverse lookup is configured properly with `host`:

```
ubuntu@ad-desktop-1:~$ host 10.2.250.153
109.231.148.10.in-addr.arpa domain name pointer ad-desktop-1.
warthogs.biz.
```

Or `dig`:

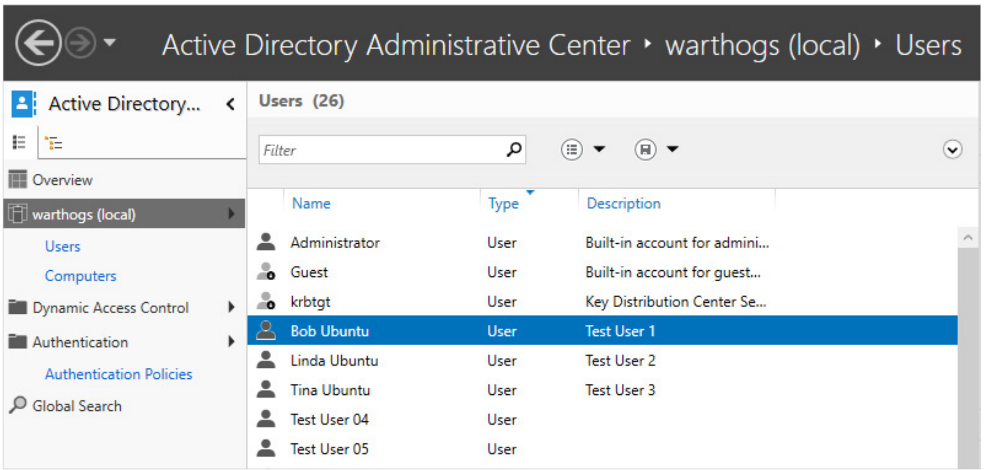
```
ubuntu@ad-desktop-1:~$ dig -x 10.2.250.109

; <<>> DiG 9.16.1-Ubuntu <<>> -x 10.148.231.109
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 23902
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;109.231.148.10.in-addr.arpa.    IN    PTR

;; ANSWER SECTION:
109.231.148.10.in-addr.arpa. 0    IN    PTR    ad-desktop-1.
warthogs.biz.

;; Query time: 0 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: lun. sept. 14 14:31:06 CEST 2021
;; MSG SIZE  rcvd: 92
```

Alternatively, we can rely on the join procedure to update the DNS records for us. SSSD will attempt to do so, and it will work as long as `hostname -f` returns a fully qualified hostname (FQDN).

A simple way to achieve this without pre-configuring DNS, or changing `/etc/hosts`, is to set the FQDN in `/etc/hostname`. Normally that file holds the short name, but changing it to the FQDN might be worth this small compromise.

Organisational Units

If your Active Directory domain is divided into organisational units (OU), you will need to determine into which OUs you want to join the Ubuntu computers. In this whitepaper, for the sake of simplicity, the structure of the directory will be flat and we will not make use of OUs.



Name	Type	Description
Administrator	User	Built-in account for admini...
Guest	User	Built-in account for guest...
krbtgt	User	Key Distribution Center Se...
Bob Ubuntu	User	Test User 1
Linda Ubuntu	User	Test User 2
Tina Ubuntu	User	Test User 3
Test User 04	User	
Test User 05	User	

Windows Server 2019 Active Directory Administrative Center - List of test users

Joining Ubuntu to an Active Directory domain

There are 2 ways to join an Ubuntu client to an AD domain:

1. At installation time with the Ubuntu Desktop installer.
2. After installation, by manually setting up the connection to AD.

Joining During Installation Time

Joining during installation is only supported by the Ubuntu Desktop graphical installer Ubiquity. So, start an installation of Ubuntu Desktop as you would usually do and proceed through the installation, making sure to:

Fill out the hostname info correctly (see note below)

Who are you?: Be sure to check the box "Use Active Directory"

Configure Active Directory: Enter the address of the Active Directory controller and credentials of the user allowed to add machines to the domain.

Note about the hostname

In order to set and resolve the host name properly, you must enter the Fully Qualified Domain Name (FQDN) of the machine in the field "Your computer's name". For example, "ad-desktop-1.warthogs.biz" instead of only the host name "ad-desktop-1".

After installation you can check if it is correct with the commands `hostname` and `hostname -f` which must return the name of the machine ("ad-desktop-1") and the full name of the machine with the domain ("ad-desktop-1.warthogs.biz") respectively.

Logging in as a user of the domain

By default, there is no default domain configured in SSSD. You have to enter the full user name with one of the forms `USER@DOMAIN` or `DOMAIN/USER`. To log in as a user of the domain, press the link "Not listed?" in the greeter. Then enter the username followed by the password.

On the first log in the user's home directory is created.

All of this (default domain, default path for home directories, default shell, etc.) is configurable in `/etc/sssd/sssd.conf`.

Joining After Installation via SSSD

An existing install of Ubuntu can be configured to join an Active Directory domain too. In order to do this, an administrator needs to first set up the Ubuntu desktop and then configure SSSD - so keep reading.

Once configured, SSSD will retrieve the credentials and the initial security policy of the "Default Domain Policy". Then the user can authenticate their Ubuntu login against an Active Directory server.

Preparing Ubuntu Desktop for SSSD Setup

On Ubuntu's side, besides a correct network setup and a user with administrative privileges, you'll need to ensure that time is synchronised with the Active Directory controller and host name resolution is working.

Administrative privileges

As is usual in Ubuntu, all examples of commands requiring super-user (administrative) privileges in the text have been prefixed with `sudo`. You are not expected to have access to the root account (it is disabled by default on Ubuntu), but you are expected to have a user account member of the admin group, who is allowed to escalate privileges using the `sudo` command. The first user account, created during installation, is a member of the admin group in question. In our case, this user is called, quite simply, `ubuntu`.

Network settings

Using SSSD obviously requires network connectivity to the Active Domain Controller. While it is possible to use SSSD on a machine configured for dynamic IP addressing using DHCP, our example will assume fixed IP settings with a single network interface for the sake of simplicity.

If a firewall is mitigating IP connectivity between the Ubuntu machine and the Active Directory Controller, you will need to ensure that the required ports are open for connection between the Active Directory Controller and the Ubuntu machines.

Depending on the topology of the network, this is the list of ports and protocols used and that must be open in order for SSSD to operate successfully. These ports must be open between each client and every domain controller of the domain:

Service	Protocol	Port
DNS	UDP, TCP	53
LDAP	UDP, TCP	389, 636
Kerberos	UDP, TCP	88, 464
SMB	UDP, TCP	445
NTP	UDP	123

Host name resolution

It is important to ensure that the fully qualified domain name (FQDN) of the Ubuntu machine matches the DNS record used in the Active Directory DNS. This information is stored in the `/etc/hostname` configuration file on Ubuntu. You can check the FQDN of the Ubuntu machine from the command line using the `hostname` command, for example:

```
ubuntu@ad-desktop-1:~$ hostname -f
ad-desktop-1.warthogs.biz
```

SSSD will attempt to update the DNS record for this hostname right after the join succeeds, and will keep it up-to-date if the IP changes. But it's important that `hostname -f` returns the fully qualified domain name for that to work.

Time synchronisation

The Kerberos protocol, used internally by Active Directory for authentication, is sensitive to clock skew between computers participating in a Kerberos domain. The default clock skew tolerance is 300 seconds (five minutes). If the Ubuntu machine and the Active Directory Controller clock drift apart for more than five minutes, authentication against the Active Directory controller will systematically fail.

Traditionally, in the Unix/Linux world, time synchronisation is achieved using the Network Time Protocol (NTP). This is usually completed against an external time source, such as one of the many public NTP servers on the internet. By default, Ubuntu is configured to synchronise time with the ntp.ubuntu.com NTP server each time a network interface is brought up, which happens at least at every boot.

In our case, it is not desirable to have the Ubuntu machine synchronise time with an outside source, as this source may differ from the Active Directory Controller. Hence, the default NTP server needs to be changed to one of the ADC. Since Ubuntu 16.04 LTS, Ubuntu, by default, uses `timedatectl / timesyncd` (which are part of systemd). It replaces most of ntpdate / ntp.

The current status of time and time configuration via *timedatectl* and *timesyncd* can be checked with `timedatectl status`:

```
# timedatectl status
            Local time: Mon 2020-09-14 14:56:28 CEST
            Universal time: Mon 2020-09-14 12:56:28 UTC
                   RTC time: Mon 2020-09-14 12:56:29
                   Time zone: Europe/Paris (CEST, +0200)
System clock synchronized: yes
            NTP service: active
            RTC in local TZ: no
```

Via *timedatectl* an administrator can control the timezone, how the system clock should relate to the hwclock and if permanent synchronisation should be enabled or not. See `man timedatectl` for more details.

The nameserver to fetch time for *timedatectl* and *timesyncd* from can be specified in `/etc/systemd/timesyncd.conf` and additional config files can be stored in `/etc/systemd/timesyncd.conf.d/`. The entries for NTP= and FallbackNTP= are space separated lists. See `man timesyncd.conf` for more.

timesyncd will generally do the right thing keeping your time in sync, and *chrony* will help with more complex cases.

The content of timesyncd.conf for our example setup is pasted below:

```
#  This file is part of systemd.
#
#  systemd is free software; you can redistribute it and/or
modify it
#  under the terms of the GNU Lesser General Public License as
published by
#  the Free Software Foundation; either version 2.1 of the
License, or
#  (at your option) any later version.
#
# Entries in this file show the compile time defaults.
# You can change settings by editing this file.
# Defaults can be restored by simply deleting this file.
#
# See timesyncd.conf(5) for details.

[Time]
NTP=adc1.warthogs.biz
FallbackNTP=ntp.ubuntu.com
#RootDistanceMaxSec=5
#PollIntervalMinSec=32
#PollIntervalMaxSec=2048
```

Setting up SSSD

Installing the package

SSSD is available from the official Ubuntu repository and its mirrors. The first task is to install the required packages from this repository.

To install the packages, open a terminal (from an Ubuntu Desktop press CTRL+ALT+T or Super then type terminal and select the terminal application), update the indices of the repository to make sure the latest version will be installed:

```
$ sudo apt update
```

then run the command:

```
$ sudo apt install sssd-ad sssd-tools realmd adcli
```

If all goes well it will return without any error. You can verify that it is correctly installed with the standard `apt` command and check that the version numbers of the installed and candidate packages match:

```
$ apt policy sssd-ad sssd-tools realmd adcli
sssd-ad:
  Installed: 2.2.3-3
  Candidate: 2.2.3-3
  Version table:
 *** 2.2.3-3 500
      500 http://archive.ubuntu.com/ubuntu focal/main amd64
Packages
      100 /var/lib/dpkg/status
sssd-tools:
  Installed: 2.2.3-3
  Candidate: 2.2.3-3
  Version table:
 *** 2.2.3-3 500
      500 http://archive.ubuntu.com/ubuntu focal/main amd64
Packages
      100 /var/lib/dpkg/status
realmd:
  Installed: 0.16.3-3
  Candidate: 0.16.3-3
  Version table:
 *** 0.16.3-3 500
      500 http://archive.ubuntu.com/ubuntu focal/universe amd64
```

```
Packages
      500 http://archive.ubuntu.com/ubuntu focal-updates/main amd64
Packages
      100 /var/lib/dpkg/status
adcli:
  Installed: 0.9.0-1
  Candidate: 0.9.0-1
  Version table:
 *** 0.9.0-1 500
      500 http://archive.ubuntu.com/ubuntu focal/universe amd64
Packages
      500 http://archive.ubuntu.com/ubuntu focal-updates/main amd64
Packages
      100 /var/lib/dpkg/status
```

Removing the package

If you want to remove the machine from the domain and remove the packages, you can do it the usual way from the command line:

```
$ sudo apt autoremove --purge  sssd-ad sssd-tools realmd adcli
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be REMOVED:
  adcli* ldap-utils* libbasicobjects0* libc-ares2* libcollection4*
libdhash1* libini-config5* libipa-hbac0* libnfsidmap2* libnss-sss*
  libpam-pwquality* libpam-sss* libpath-utils1* libref-array1*
libsss-certmap0* libsss-idmap0* libsss-nss-idmap0* libsss-sudo*
  python3-sss* realmd* sssd* sssd-ad* sssd-ad-common* sssd-
common* sssd-ipa* sssd-krb5* sssd-krb5-common* sssd-ldap* sssd-
proxy*
  sssd-tools*
0 upgraded, 0 newly installed, 30 to remove and 0 not upgraded.
After this operation, 10,9 MB disk space will be freed.
Do you want to continue? [Y/n]
```

Joining a domain

Once the SSSD suite packages are installed, you can proceed to a first verification to ensure it can contact the domain with the command `realm`. Realmd is the tool to manage enrollment in Active Directory and more generally Kerberos Realms. It is used to discover and join domains and perform domain integration and configure SSSD to connect to the domain.

To discover the domain with `realmd`, run the following command:

```
$ realm discover adc01.warthogs.biz
warthogs.biz
  type: kerberos
  realm-name: WARTHOGS.BIZ
  domain-name: warthogs.biz
  configured: no
  server-software: active-directory
  client-software: sssd
  required-package: sssd-tools
  required-package: sssd
  required-package: libnss-sss
  required-package: libpam-sss
  required-package: adcli
  required-package: samba-common-bin
```

Note in the output that ‘`configured: no`’. We can now join the domain and configure SSSD with the `join` command or `realm`:

```
$ sudo realm join adc01.warthogs.biz
[sudo] password for ubuntu: ********
Password for Administrator: ********
```

You can substitute `‘warthogs.biz’` for your own domain name, and `‘Administrator’` for an account with sufficient privileges to join computers in your domain with the `-U` flag. `man realm`, `realm -h`, and `realm <command> -h` provide all the details about this command.

Realmd generated the configuration for SSSD in `/etc/sssd/sssd.conf`:

```
/etc/sssd/
├── conf.d
└── sssd.conf
```

Here is the default configuration file generated when a client has successfully joined a domain:

```
[sssd]
domains = warthogs.biz
config_file_version = 2
services = nss, pam

[domain/warthogs.biz]
default_shell = /bin/bash
ad_server = adc01.warthogs.biz
krb5_store_password_if_offline = True
cache_credentials = True
krb5_realm = WARTHOGS.BIZ
realmd_tags = manages-system joined-with-adcli
id_provider = ad
fallback_homedir = /home/%u@%d
ad_domain = warthogs.biz
use_fully_qualified_names = True
ldap_id_mapping = True
access_provider = ad
```

Some default settings worth mentioning are:

- **`cache_credentials:`** With this set to True, the credentials will be cached and the users will be allowed to login even if the machine is disconnected from the network.

- **`fallback_homedir:`** This is used if no homedir is provided by the domain’s data provider. For instance, with the default, the home directory for user ‘`linda`’ is of the form: ‘`/home/linda@warthogs.biz’`.

- **`use_full_qualified_names`:** users will be of the form '`user@domain`', not just the username. This should only be changed if you are certain no other domains will ever join the AD forest, via one of the several possible trust relationships.

Creating a home directory

What the `realm` tool didn't do for us is setup `pam_mkhomedir` (this should be fixed with [Bug #1894135 "Enable homedir creation" : Bugs : realmd package : Ubuntu](#)), so that network users can get a home directory automatically when they login. This remaining step can be done by running the following command:

```
# pam-auth-update --enable mkhomedir
```

This will enable `pam_mkhomedir.so` in the `/etc/pam.d/common-session` pam configuration file.

Alternatives to SSSD

If, for some reason, SSSD is not suitable for your environment, there are other options that can help you integrate Ubuntu into Active Directory for authentication and identity management.

NSS and PAM configured for LDAP and Kerberos

Microsoft Active Directory is based, in part, on the LDAP and Kerberos standards. Both of these protocols are well supported in the Linux world. LDAP can be used as a database for NSS, and both LDAP and Kerberos can be used as an authentication backend with PAM. Combined together, you can get the same result as using SSSD, without the disconnected operations.

Unix user and group accounts require a certain number of attributes that are not present by default in Active Directory. Starting with Windows Server 2003 R2, the Active Directory schema have been extended to include attributes conforming to RFC 2307 - 'An Approach for Using LDAP as a Network Information Service', which defines the LDAP attributes required by Unix and Unix-like system, such as Ubuntu. A role service, Identity Management for UNIX, is available in Active Directory to extend functionalities precisely for that purpose. From there on, Active Directory can be used as a store of Unix users and groups as is, without the help of third-party tools. This approach may prove more flexible than using SSSD, at the cost of being more management heavy.

On the Active Directory side, RFC2307 attributes of users and groups would need to be set and managed. While installing the Identity Management for UNIX role service extends the Active Directory Users and Computers MMC snap-in to expose these attributes and allows for setting their value manually, a large deployment will certainly need some sort of tool to automate the process. This needs to be taken into consideration.

Plain winbind

Winbind includes PAM and NSS modules to authenticate and resolve users and groups to an Active Directory. It is generally used alongside Samba for file sharing services, where it exposes Windows domain users as local Unix users.

Its use, however, is not restricted to Samba: it would also work well with any services that use NSS and PAM for user management.

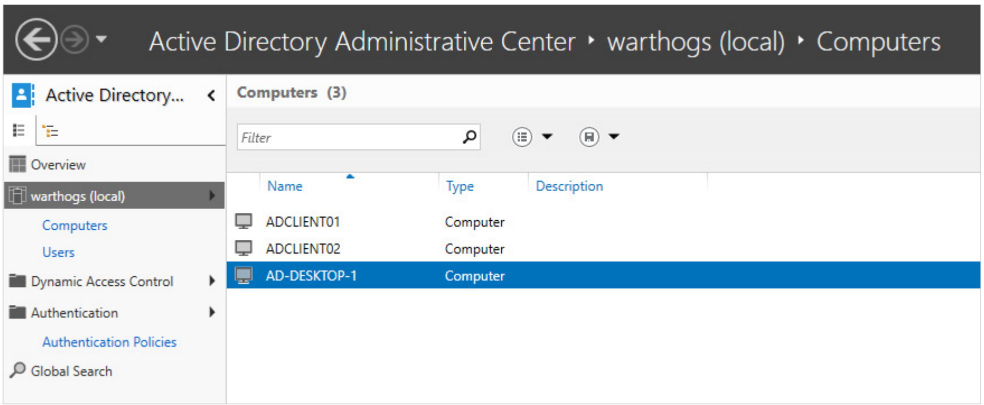
One of winbind's tasks is to keep a mapping of Unix numerical user and group ID for Windows users and groups. By default, winbind assigns Unix uid and gid sequentially as users and groups are being looked up. The result is, obviously, rather random and will vary from one machine running winbind to another. This will pose a problem if your infrastructure requires Unix and Unix-like machines to share a coherent uid and gid namespace; that would be the case if, for example, you were to share files using the Unix-native NFS protocol. Fortunately, winbind can be configured to use a so-called idmap backend that can be shared among multiple winbind instances. The job of these idmap backends is to store the Windows SID to Unix id mapping, ensuring that users and groups ID are consistent across all machines using the same backend. For example, such an idmap could ensure that the Active Directory user WARTHOGS\bob has a user ID 13897 on both server ubuntusrv1 and ubuntusrv2, making file permissions manageable and consistent when files are being shared between the two. Various idmap backends are available, and winbind can also be configured to derive Unix ID algorithmically based on the Windows RID.

It is worth noting that realmd can also use winbind with the command line option `--client-software=winbind`

If you wish to learn more about winbind, the best reference remains the Official Samba How to and reference guide at:
http://samba.org/samba/docs/man/Samba-HOWTO-Collection/

Verifying proper domain operations

Whether the Ubuntu system was joined to Active Directory at install time or afterwards with SSSD, you may want to check that the Ubuntu server is listed in Computers in the Active Directory Administrative Centre.



Computers (3)

Name	Type	Description
ADCLIENT01	Computer	
ADCLIENT02	Computer	
AD-DESKTOP-1	Computer	

List of computers in Active Directory Administrative Center

On the Ubuntu computer, you can use the `realm` command (provided by the `realmd` package) to confirm that it has indeed joined the domain by querying SSSD with the realm command, for example:

```
# realm list
warthogs.biz
  type: kerberos
  realm-name: WARTHOGS.BIZ
  domain-name: warthogs.biz
  configured: kerberos-member
  server-software: active-directory
  client-software: sssd
  required-package: sssd-tools
  required-package: sssd
  required-package: libnss-sss
  required-package: libpam-sss
  required-package: adcli
  required-package: samba-common-bin
  login-formats: %U@warthogs.biz
  login-policy: allow-realm-logins
```

The output shows that the domain is now 'configured', and displays additional information like the login format. This is the format we will use to log into the machine or identify remote accounts on Active Directory.

Next, verify that users from the domain can be resolved using the `getent` command, for example, using this format we can retrieve the entry for a user:

```
# getent passwd linda@warthogs.biz
linda@warthogs.biz:*:1899001103:1899000513:Linda Ubuntu:/home/
linda@warthogs.biz:/bin/bash
```

The `getent` command is used to query NSS databases. In the above case, we ask getent to query the password (passwd) database for the `'linda@warthogs.biz'` entry. The entry format is the same as is used in the `/etc/passwd` system user database, except the entry is not actually from `/etc/passwd`; it is pulled from Active Directory. You may also use `getent` to resolve a group entry, as below:

```
root@ad-desktop-1:~# getent group “domain users”@warthogs.biz
domain users@warthogs.biz:*:1899000513:linda@warthogs.biz
root@ad-desktop-1:~#
root@ad-desktop-1:~# getent group marketing@warthogs.biz
marketing@warthogs.biz:*:1899001134:linda@warthogs.biz,bob@
warthogs.biz
```

Notice that both users and groups from Active Directory are suffixed with the domain name according to the usual convention NAME@DOMAIN.

Other standard tools can be used to list the groups a user is a member of, for example:

```
# groups bob@warthogs.biz
bob@warthogs.biz : domain users@warthogs.biz marketing@warthogs.biz
```

> **Note**
>
> If you just changed the group membership of a user, it may be a while before sssd notices due to caching. If it is taking too long, you can manually stop the service, delete the cache, and restart the service.

Additional verification with sssd-tools

`sssd-tools` is a package that provides several utilities to manage and display information about users, groups, and sssd in general.

`sssctl` is one of these utilities. From the man page, it “*provides a simple and unified way to obtain information about SSSD status, such as active server, auto-discovered servers, domains and cached objects. In addition, it can manage SSSD data files for troubleshooting in such a way that is safe to manipulate while SSSD is running*”.

`sssctl` depends on the package `sssd-dbus` to provide the InfoPipe responder. If it is not already installed, use `apt` to do it.

```
# apt install sssd-dbus
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  sssd-dbus
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 93,5 kB of archives.
After this operation, 357 kB of additional disk space will be
used.
[installation of the package …]
root@ad-desktop-1:/var/log/sssd#
```

To use `sssctl`, `sssd` must be running and the InfoPipe responder must be enabled.

The InfoPipe responder is enabled in `/etc/sssd/sssd.conf` by adding it to the list of services:

```
[sssd]
domains = warthogs.biz
config_file_version = 2
services = nss, pam, ifp
```

Then you can restart `sssd` and verify the status of `sssd` with `systemctl`:

```
root@ad-desktop-1:~# systemctl restart sssd
root@ad-desktop-1:~# systemctl status sssd
● sssd.service - System Security Services Daemon
      Loaded: loaded (/lib/systemd/system/sssd.service; enabled;
vendor preset: enabled)
      Active: active (running) since Tue 2020-09-15 11:28:55 CEST;
49s ago
   Main PID: 8138 (sssd)
      Tasks: 5 (limit: 4657)
      Memory: 36.6M
      CGroup: /system.slice/sssd.service
             ├─8138 /usr/sbin/sssd -i --logger=files
             ├─8159 /usr/libexec/sssd/sssd_be --domain warthogs.
biz --uid 0 --gid 0 --logger=files
             ├─8162 /usr/libexec/sssd/sssd_nss --uid 0 --gid 0
--logger=files
             ├─8163 /usr/libexec/sssd/sssd_pam --uid 0 --gid 0
--logger=files
             └─8164 /usr/libexec/sssd/sssd_ifp --uid 0 --gid 0
--logger=files
sept. 15 11:28:55 ad-desktop-1 sssd[be[8159]: Starting up
sept. 15 11:28:55 ad-desktop-1 sssd[8163]: Starting up
sept. 15 11:28:55 ad-desktop-1 sssd[8164]: Starting up
sept. 15 11:28:55 ad-desktop-1 sssd[8162]: Starting up
sept. 15 11:28:55 ad-desktop-1 systemd[1]: Started System Security
Services Daemon.
```

`sssctl` provides a set of commands to check the status of the domain and list users and groups.

- **`domain-status`:** Print information about domain

```
# sssctl domain-status warthogs.biz
Online status: Online

Active servers:
AD Global Catalog: adc01.warthogs.biz
AD Domain Controller: adc01.warthogs.biz

Discovered AD Global Catalog servers:
- adc01.warthogs.biz

Discovered AD Domain Controller servers:
- adc01.warthogs.biz
```

- **`user-checks`:** Print information about a user and check authentication

```
# sssctl user-checks bob@warthogs.biz
user: bob@warthogs.biz
action: acct
service: system-auth

SSSD nss user lookup result:
 - user name: bob@warthogs.biz
 - user id: 1899001102
 - group id: 1899000513
 - gecos: Bob Ubuntu
 - home directory: /home/bob@warthogs.biz
 - shell: /bin/bash
[...]
```

In this whitepaper, we won't cover the other commands. These commands are documented in the help of `sssctl`.

Additional verification with samba-tool

`samba-tool` is the main Samba (A Windows AD and SMB/CIFS file server for UNIX) administration tool. It is provided by the package `samba-common-bin`:

```
~# apt install samba-common-bin
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  python3-ldb python3-samba python3-tdb samba-dsdb-modules
Suggested packages:
  heimdal-clients python3-markdown python3-dnspython
The following NEW packages will be installed:
  python3-ldb python3-samba python3-tdb samba-common-bin samba-
dsdb-modules
0 upgraded, 5 newly installed, 0 to remove and 0 not upgraded.
Need to get 3 270 kB of archives.
After this operation, 23,0 MB of additional disk space will be
used.
Do you want to continue? [Y/n] y
[...]
```

With this tool you can directly manage domain users and groups, domain Group Policy, domain sites, DNS services, domain replication and other critical domain functions.

Like `sssctl`, `samba-tools` provides a set of commands to check the status of the domain and list users and groups:

- **`domain`:** domain management

```
~# samba-tool domain info adc01.warthogs.biz
Forest              : warthogs.biz
```

```
Domain             : warthogs.biz
Netbios domain    : WARTHOGS
DC name            : ADC01.warthogs.biz
DC netbios name   : ADC01
Server site        : Default-First-Site-Name
Client site        : Default-First-Site-Name
```

- **`user`:** user management

```
~# samba-tool user list -U Administrator -H ldap://adc01.warthogs.
biz
Password for [WORKGROUP\Administrator]:
Administrator
Guest
krbtgt
bob
linda
Tina
[...]
```

Note the syntax used to indicate the host. It uses an LDAP URI instead of just the domain or the address of the domain controller.

- **`group`:** Group management

```
~# samba-tool group list -U Administrator -H ldap://adc01.
warthogs.biz
Password for [WORKGROUP\Administrator]:
Administrators
Users
Guests
[...]
Key Admins
Enterprise Key Admins
Marketing
```

- **`computer`:** Computer management

```
~# samba-tool computer list -U Administrator -H ldap://adc01.
warthogs.biz
Password for [WORKGROUP\Administrator]:
ADC01$
ADCLIENT02$
ADCLIENT01$
AD-DESKTOP-1$
```

- **`computer show`:** Display a computer AD object

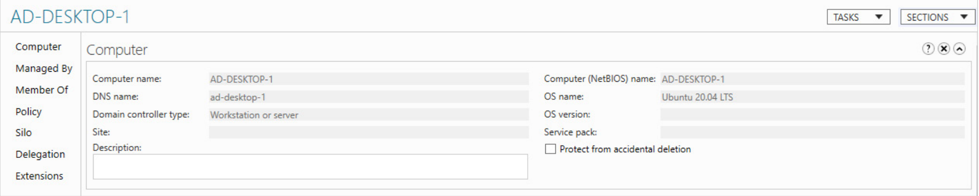
```
~# samba-tool computer show AD-DESKTOP-1 -U Administrator -H
ldap://adc01.warthogs.biz
Password for [WORKGROUP\Administrator]:
dn: CN=AD-DESKTOP-1,CN=Computers,DC=warthogs,DC=biz
objectClass: top
objectClass: person
objectClass: organizationalPerson
```

```
objectClass: user
objectClass: computer
cn: AD-DESKTOP-1
distinguishedName: CN=AD-DESKTOP-1,CN=Computers,DC=warthogs,DC=biz
[...]
operatingSystem: pc-linux-gnu
dNSHostName: ad-desktop-1
[...]
```

- **`computer edit`:** Modify a computer AD object

```
# samba-tool computer edit AD-DESKTOP-1 -U Administrator -H
ldap://adc01.warthogs.biz
Password for [WORKGROUP\Administrator]:
[
This opens the record of the object in a text editor
Change the text ‘pc-gnu-linux’ to ‘Ubuntu 22.04 LTS’
]
Modified computer ‘AD-DESKTOP-1’ successfully
root@ad-desktop-1:~#
:~# samba-tool computer show AD-DESKTOP-1 -U Administrator -H
ldap://adc01.warthogs.biz
Password for [WORKGROUP\Administrator]:
dn: CN=AD-DESKTOP-1,CN=Computers,DC=warthogs,DC=biz
[...]
operatingSystem: Ubuntu 22.04 LTS
[...]
```

The attribute `operatingSystem` has been modified from ‘`pc-gnu-linux`’ to ‘`Ubuntu 22.04 LTS`’, which is also visible in Active Directory Administrative Center.



Computer name: AD-DESKTOP-1
DNS name: ad-desktop-1
Domain controller type: Workstation or server
Site:
Description:
Computer (NetBIOS) name: AD-DESKTOP-1
OS name: Ubuntu 20.04 LTS
OS version:
Service pack:
□ Protect from accidental deletion

Windows Server AD Admin Center - Detail of a computer object

Finally `samba-tool`, without any arguments, lists the available commands and `-h` with any command displays the subcommands and the help for that subcommand.

User authentication

You can test authentication by using `login` and an Active Directory user:

```
# login
ad-desktop-1 login: bob@warthogs.biz
Password:********
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.4.0-48-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro
```

```
0 updates can be installed immediately.
0 of these updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in
the individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in
the individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

Last login: mar. sept. 15 12:09:50 CEST 2020 on pts/1
Creating directory '/home/bob@warthogs.biz'.
bob@warthogs.biz@ad-desktop-1:~$
```

If you used SSSD to connect the Ubuntu system to AD, then the home directory isn't automatically created for you, as mentioned earlier. In this case, since it is the first login and the home directory doesn't exist yet, we can see the output of the mkhomedir PAM module that indicates the creation of the directory.

If however you joined the Ubuntu system to AD at installation time, you will already have a home directory.

Remote connection can be tested with an SSH server running on the system. We can simply open an SSH connection to '`localhost`' using an Active Directory user, for example:

```
# ssh linda@warthogs.biz@ad-desktop-1.warthogs.biz
The authenticity of host 'ad-desktop-1.warthogs.biz
(192.168.122.109)' can't be established.
ECDSA key fingerprint is
SHA256:NaOHo8x+mUw4DwxCwGVQuKT0GszZFZZ5qTCYyYhHAFM.
Are you sure you want to continue connecting (yes/no/
[fingerprint])? yes
Warning: Permanently added 'ad-desktop-1.warthogs.
biz,192.168.122.109' (ECDSA) to the list of known hosts.
linda@warthogs.biz@ad-desktop-1.warthogs.biz's password:
Creating directory '/home/linda@warthogs.biz'.
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.4.0-48-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

0 updates can be installed immediately.
0 of these updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in
the individual files in /usr/share/doc/*/copyright.
```

```
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

linda@warthogs.biz@ad-desktop-1:~$
```

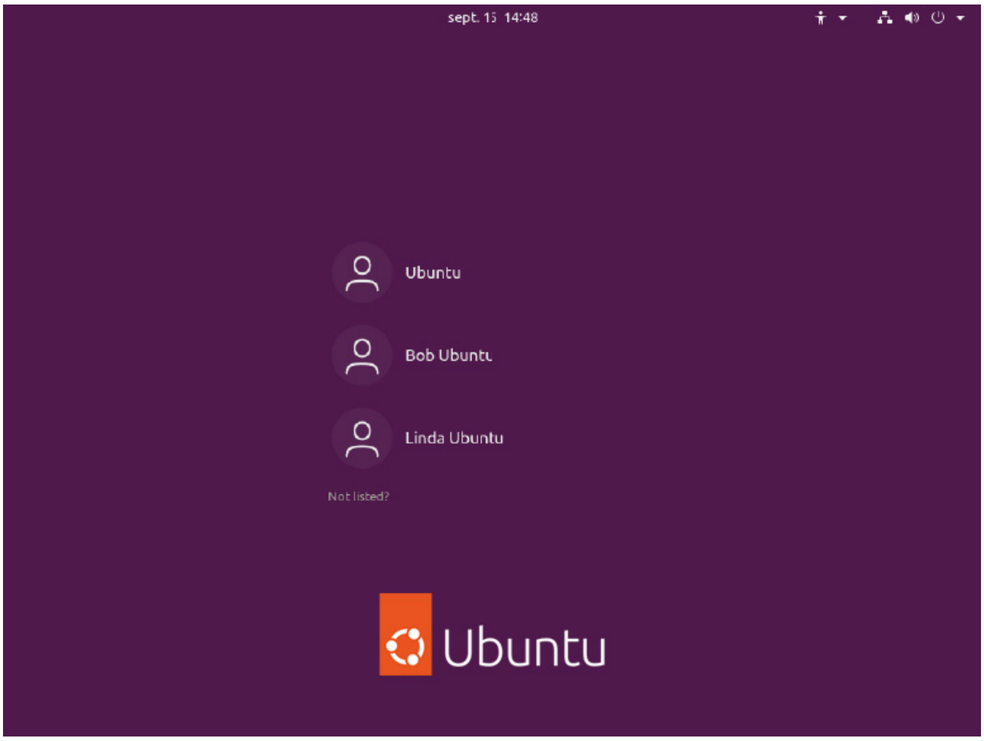
Note the double '@' syntax in the connection string. The first @ is for the user@ domain and the second @ is for the user@ssh_server.

As for the `login` command, the mkhomedir PAM module indicates that the home directory has been created for user '`linda`'.

From the above, we can see that we were able to establish an SSH connection using the credentials of an Active Directory user, confirming the ability to authenticate to the Active Directory. The `whoami` and `id` commands further confirm that resolving users and groups work as expected. We have finished testing the installation.

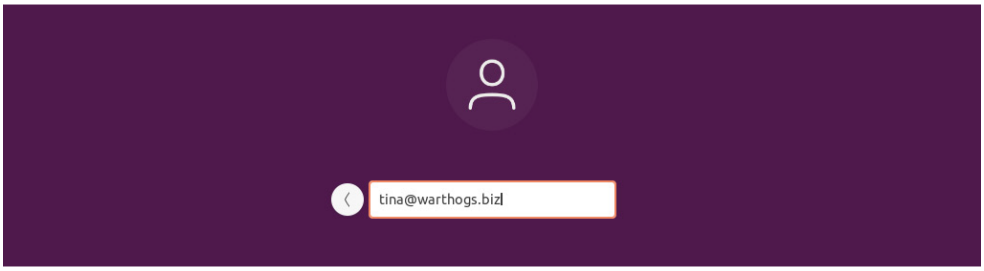
```
$ whoami
linda@warthogs.biz
$
$ id
uid=1899001103(linda@warthogs.biz) gid=1899000513(domain
users@warthogs.biz) groups=1899000513(domain users@warthogs.
biz),1899001134(marketing@warthogs.biz)
$
$ groups
domain users@warthogs.biz marketing@warthogs.biz
```

Last, you can verify that a user can authenticate with a graphical interface from GDM. On the login screen the list of users includes the local users, in our example 'Ubuntu', and the list of Active Directory users which have already logged in at least once, in our example, 'Linda Ubuntu' and 'Tina Ubuntu' .



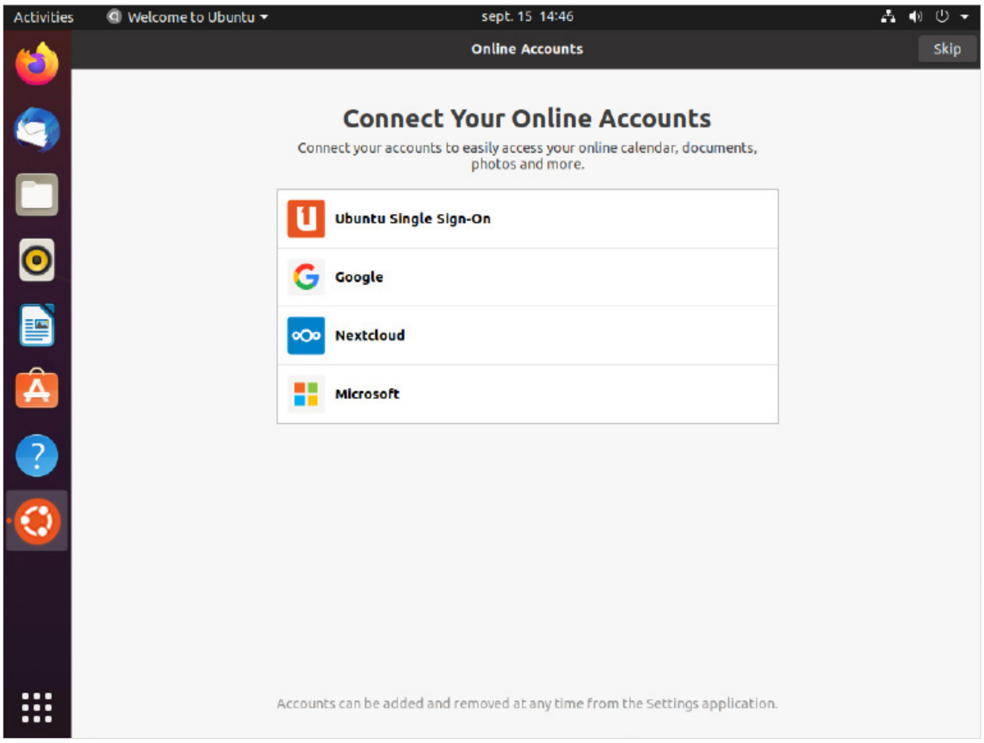
Ubuntu Desktop 22.04 LTS - Desktop Greeter - List of users

User ‘Tina’ is not listed and wants to log into this machine. She selects “Not Listed?” to display a login prompt and enter her username followed by the domain name.



Ubuntu Desktop 22.04 LTS - First connection on a machine of a domain user

Upon entering her password, she successfully starts her session.



Connect Your Online Accounts

Connect your accounts to easily access your online calendar, documents, photos and more.

Accounts can be added and removed at any time from the Settings application.

Ubuntu Desktop 22.04 LTS - First connection of a user

Additional configuration

Allowing sudo access to the admin group

We can now allow the administrator of the Active Directory controller to have administrator privileges on the machines controlled by this controller. In Ubuntu, it means all privileges to run `sudo` commands to the group `administrator` .

For instance, if we want to grant admin privileges on the Ubuntu machines to the group of user ‘Administrator’, the first thing is to determine the group of this user. This is done with the standard command `id`:

```
$ id Administrator@warthogs.biz
uid=1899000500(administrator@warthogs.biz)
gid=1899000513(domain users@warthogs.biz)
```

```
groups=1899000513(domain users@warthogs.biz),
      1899000512(domain admins@warthogs.biz),
      1899000572(denied rodc password replication group@warthogs.
biz),
      1899000520(group policy creator owners@warthogs.biz),
      1899000519(enterprise admins@warthogs.biz),
      1899000518(schema admins@warthogs.biz)
```

The command returns 6 records. In our example, the Active Directory admin group we are interested in is “`domain admins@warthogs.biz`”. Granting `sudo` access to the group is accomplished by adding the group to `/etc/sudoers` and allowing it to run any command.

```
root@ad-desktop-1:~# visudo
[the default editor opens /etc/sudoers]
# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL
“%domain admins@warthogs.biz” ALL=(ALL) ALL
```

Note

There are double quotes around the domain name to protect the space in the name of the group. It is recommended to use `visudo` to edit the sudoers file and avoid any mistake when entering AD group names that may contain spaces.

Now the administrator of the Active Directory controller can run any command with `sudo`.

```
root@ad-desktop-1:~# sudo -i -u Administrator@warthogs.biz
administrator@warthogs.biz@ad-desktop-1:~$ whoami
administrator@warthogs.biz
administrator@warthogs.biz@ad-desktop-1:~$ sudo ls /
[sudo] password for administrator@warthogs.biz:
bin  boot  cdrom  dev  etc  home  lib  lib32  lib64  libx32
lost+found  media   mnt  opt  proc   root  run  sbin  snap
srv  swapfile  sys  tmp   usr  var
administrator@warthogs.biz@ad-desktop-1:~$
```

Alternatively, you could create a specific group in Active Directory, eg `UbuntuAdmins`, and add this group to sudoers like:

```
# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL
%UbuntuAdmins@warthogs.biz ALL=(ALL) ALL
```

Configure default domain, HOME directory and default shell

With SSSD, it is possible to customise the user’s desktop environment in multiple ways. You can configure the HOME directory, login shell, domain prefix, etc.

This is accomplished through its configuration file `/etc/sssd/sssd.conf`.
As seen previously, this file has been created by `realm` during the join operation.

Note

If you modify this file manually, something very important to remember is that this file must have permission 0600 and ownership root:root, or else sssd won't start.

Some useful settings are:

- **`default_domain_suffix`:** This string will be used as a default domain name for all names without a domain name component. The main use case is environments where the primary domain is intended for managing host policies and all users are located in a trusted domain. The option allows those users to login just with their username, without giving a domain name as well.

- **`override_homedir`:** Override the user's home directory. If you provide a template, variables can be substituted at runtime.

- **`override_shell`:** Override the login shell for all users. Alternatively, `default_shell` can be used if none is returned during lookup.

`sssd` must be restarted with '`systemctl restart sssd`' for the changes to this file to take effect.

At this point we have working logins via GDM, the console or a remote shell like ssh, bash is set as the default shell, and the home directory structure matches standard user directories.

Leaving a domain

To remove your Ubuntu Desktop machine from the domain simply run the following command:

```
~$ sudo realm leave warthogs.biz
~$
```

If there is no error printed, then we successfully left the domain. We can then confirm that the machine really left the domain with:

```
root@ad-desktop-1:~# realm list
root@ad-desktop-1:~#
```

It doesn't list any domains.

Setting up ADSys

Get ADsys through Ubuntu Pro

You need an active Ubuntu Pro subscription and a system already joined to Active Directory to run ADsys and perform the actions enabled by the tools. Please refer to the previous sections to perform these steps.

Enabling Ubuntu Pro

Please follow the instructions in the following link to get an ubuntu Pro subscription and attach it to the machine you wish to use ADsys on.

Note

Subscriptions are not just for enterprise customers. Anyone can get a personal subscription for free on up to 5 machines, or 50 if you are an official Ubuntu Community member.

Installing ADSys

ADSys facilitates the use of some advanced Active Directory management features and it is not currently installed by default. This must be done manually by the local administrator of the machine. To do so, log in on first boot, update the repositories and install ADSys with the following commands:

```
~$ sudo apt update
~$ sudo apt install adsys
```

Reboot and the system will automatically do its policy refresh.

Active Directory Setup

Active Directory needs policy files (*.admx* and *.adml*) to know which settings can be changed on given clients and actions to be performed on users of the domain.

We generally separate the Ubuntu configuration from the Windows one to avoid incompatibilities:

- Namespace conflicts
- Forward vs backward slashes
- Lets the administrator know exactly what settings are supported on which client version.

Note

If you need more information on in-depth Active Directory setup and operations, then please review Microsoft's documentation on the central store.

Ubuntu administrative template generations

ADSys ships with pre-built Active Directory administrative templates that you can install on your Active Directory server. You will find two flavours of them:

1. One listing only Long Term Supported (LTS) Ubuntu versions.
2. One listing all currently supported Ubuntu versions, including non-LTS.

To get them, run:

```
~$ adsysctl policy admx lts-only
```

It will create 2 files: `Ubuntu.adml` and `Ubuntu.admx` in the current directory. As described, those will list only LTS versions of Ubuntu. To get all of the supported releases, replace “`lts-only`“ with “`all`”.

Those are the 2 files that must be copied onto your Active Directory server.

You can find the latest version of those files in a dedicated directory of the upstream repository. Note that not all of the keys may be supported by our local ADSys installation. Only the templates generated by the `adysctl` that match the version of your client.

Deployment of ADM files on the Active Directory server

The administrative templates for Ubuntu must be deployed on your Active Directory server in the policy definition directory corresponding to your forest root. For example:

- .admx file: “\\example.com\sysvol\example.com\Policies\PolicyDefinitions”
- .adml file: “\\example.com\sysvol\example.com\Policies\PolicyDefinitions\en-US”

These directories can be created manually if they do not exist.

For more information read the Microsoft documentation
Create and manage the Central Store.

Enabling ADsys management features

Adsys enables system administrators to configure and use the following Active Directory functionalities on Ubuntu Desktop, using a similar experience as the one available on Windows.

Using GPOs with Ubuntu

Group Policy Objects can be used to change any of the dconf settings. Compatibility to additional policy managers might be extended in the future based on usage and customer demand.

Similar to Windows, we offer both user and computer policies, which can be accessed by navigating to the Ubuntu administrative template section of Active Directory. GPO rules can have the traditional enabled, disabled and not configured states and their precedence follows the same, default Active Directory constructs, (machine policies take precedence over user ones).

As in Windows, GPOs are applied:

- On boot for the computer policies
- On login for the user policies
- At a configurable time interval for active, connected clients (the default is set to the standard 90 minutes)

The settings are applied to the relevant users on the client and they can be overwritten only by local machine administrators.

For more information on how to configure and use GPOs with Ubuntu, please refer to the dedicated section of the ADsys documentation.

Privilege management

The Admin privilege manager enables you to grant or revoke superuser privileges for the default local user, and Active Directory users and groups.

By default members of the local sudo group are administrators on the machine. If the local User setting is set to Disabled the sudo group members are not considered administrators on the client. This means that only valid Active Directory users are able to log in to the machine.

Similarly it is possible to grant administrator privileges to specific Active Directory users and groups, or a combination of both. Using groups is an essential feature to allow you to securely manage administrators across machines, as privileged access reviews will be reduced to reviewing membership to a single or a few Active Directory groups.

For more information on how to perform privilege management and remove local sudoers please refer to the dedicated section of the ADsys documentation.

Remote script execution

The remote script execution functionality allows the execution of shell scripts or any supported binary on the target machine (including Powershell if the relevant package is installed on Ubuntu). Active Directory can be configured to execute the script on behalf of the client or by impersonating other users.

To be executed, the scripts have to be copied in the Active Directory sysvol folder and they are specific per distribution. To expose a new version to the system it is also necessary to create an appropriate GPT.ini file, and make sure it is updated every time a new version of the script is available. This can be done manually or through a daemon.

Once in the folder, scripts can be enabled or disabled by navigating to the relevant Computer Scripts or User Scripts folder on the Ubuntu administrative templates, depending on whether you want them to tie them to the machine (startup/shutdown) or the users (login/logoff).

It is important that, like in Windows, the script sessions are transactional, meaning that whenever the machine boots up and connects to the domain controller it will download the latest available version of the script. However, if a new version becomes available throughout the session it will not be executed until the next reboot/login.

For more information on how to configure and operate the remote script execution feature please refer to the dedicated section of the ADsys documentation.

Managing AppArmor profiles

AppArmor is a Linux Security Module implementation of name-based mandatory access controls. AppArmor confines individual programs to a set of listed files and posix 1003.1e draft capabilities.

On Ubuntu, AppArmor is installed and loaded by default. It uses profiles of an application to determine what files and permissions the application requires. Some packages will install their own profiles, and additional profiles can be found in the apparmor-profiles package.

ADsys bundles an AppArmor manager that allows admins to enforce custom AppArmor profiles on the client through Active Directory GPOs.

Custom AppArmor profiles can be enforced on a:

- **System-wide level**, which are loaded on the /etc/apparmor.d/adsys/machine folder of the machine.

- **User level**, supporting the confinement of executables on a user-by-user basis via the pam_apparmor PAM module. The module allows applications to confine authenticated users into subprofiles based on group names, user names, or a default profile.

You can find more information on how to develop AppArmor profiles in the following documentation article.

For more information on how to enforce custom AppArmor profiles using Active Directory, PAM module configuration and parsing behaviours please refer to the dedicated section of the ADsys documentation.

Attach network shares

The ADsys mount manager enables AD administrators to specify network shares that must be mounted on the machine that is connected to the domain.

The mount process for these mounts is triggered at the moment a user logs in. System mounts are handled by systemd through unit files and happen at root level. Therefore, users do not have control over the mounting / unmounting process.

All protocols supported by the mount command should work out of the box. However, the only tested ones are smb, ftp and nfs.

The backends for the protocols smb and nfs are automatically enabled when installing the adsys package. In order to enable the backend for ftp mounts, the user must install the recommended curlftpfs package. This behaviour is tested on Ubuntu and might differ on other Linux distributions.

For more information on how to mount and manage network shares please refer to the dedicated section of the ADsys documentation.

Configure proxy settings

The proxy manager enables AD administrators to apply proxy settings on Ubuntu Desktop. Currently, only system-wide proxy settings are supported.

ADsys currently supports configuration for the following proxy settings:

- HTTP Proxy
- HTTPS Proxy
- FTP Proxy
- SOCKS Proxy
- Ignored hosts
- Autoconfiguration URL

For more information on how to configure and manage proxy settings using Active Directory refer to the dedicated section of the ADsys documentation.

Certificate autoenrollment

Active Directory Certificate Services provides customizable services for issuing and managing digital certificates used in software security systems that employ public key technologies.

The digital certificates that AD CS provides can be used to encrypt and digitally sign electronic documents and messages. These digital certificates can be used for authentication of computer, user, or device accounts on a network.

The certificate policy manager allows Ubuntu Desktops to enrol certificates from Active Directory Certificate Services. Certificates are then continuously monitored and refreshed by the certmonger daemon. Currently, only machine certificates are supported.

Unlike the other ADSys policy managers which are configured in the special Ubuntu section provided by the ADMX files (Administrative Templates), settings for certificate auto-enrollment are configured in the Microsoft GPO tree:

This feature relies on the certmonger and cepces packages respectively to monitor/ update certificates and communicate with Active Directory Certificate Services.

For more information on how to configure and manage certificates using Active Directory refer to the dedicated section of the ADsys documentation.

The adsys Daemon and Client

Socket Activation

The ADSys daemon is started on demand by systemd's socket activation and only runs when it's required. It will gracefully shutdown after idling for a short period of time (by default 120 seconds).

Configuration

ADSys doesn't ship a configuration file by default. However, such a file can be created to modify the behaviour of the daemon and the client.

The configuration file can be system-wide in `/etc/adsys.yaml`. This will thus apply to both daemon and client. You can have per-user configuration by creating `$HOME/adsys.yaml`. It will then affect only the client for this user.

The current directory is also searched for an `adsys.yaml` file.

Finally, each `adsysd` and `adsysctl` commands accept a `--config|-c <configuration_file_path>` flag to set the path to a configuration file at run time. It can be used for testing purposes, for instance.

Here is an example of a configuration file with all the options:

```
# Service and client configuration
verbose: 2
socket: /tmp/adsysd/socket

# Service only configuration
service_timeout: 3600
ad_server: ldap://adc.example.com
ad_domain: example.com
cache_dir: /tmp/adsysd/cache
run_dir: /tmp/adsysd/run
dconf_dir: /etc/dconf
sss_cache_dir: /var/lib/sss/db

# Client only configuration
client_timeout: 60
```

Configuration common between service and client

- **verbose**: Increase the verbosity of the daemon or client. By default, only warnings and error logs are printed. This value is set between 0 and 3. This has the same effect as the `-v` and `-vv` flags.

- **socket**: Path of the unix socket for communication between clients and daemon. This can be overridden by the `--socket` option. Defaults to `/run/adsysd.sock` (monitored by systemd for socket activation).

Service only configuration

- **service_timeout**: Time in seconds without any active request before the service exits. This can be overridden by the `--timeout` option. Defaults to 120 seconds.

- **ad_server**: Active directory LDAP server to contact. This overrides the field `ad_server` from `/etc/sssd.conf`. This can be overridden by the `--ad-server` option.

- **ad_domain**: Active directory domain to use for the machine and users, if not provided. This overrides the field `ad_domain` from `/etc/sssd.conf`. This can be overridden by the `--ad-domain` option.

- **cache_dir**: The cache directory contains the policy files downloaded from an Active Directory server and the rules currently applied to the machine and users. This can be overridden by the `--cache-dir` option. Defaults to `/var/cache/adsys/`.

- **run_dir**: The run directory contains the links to the kerberos tickets for the machine and the active users. This can be overridden by the `--run-dir` option. Defaults to `/run/adsys/`.

- **dconf_dir**: Setting specific to the dconf provider. It is the directory containing the dconf configuration. By default `/etc/dconf/`.

- **sss_cache_dir**: The directory that stores Kerberos tickets used by SSSD. The default is `/var/lib/sss/db/`.

Client only configuration

- **client_timeout**: Maximum time in seconds between 2 server activities before the client returns and aborts the request. This can be overridden by the `--timeout` option. The default is 30 seconds.

Debugging with logs

It is possible to follow the exchanges between all clients and the daemon with the `cat` command. It forwards all logs and messages printed from the daemon alone.

Only privileged users have access to this information. As with any other command, the verbosity can be increased with `-v` flags (it's independent of the daemon or client current verbosity). More flags increases the verbosity further, up to 3.

> Do not hesitate to use the shell completion and the help subcommands to get more information about various commands.

Keep reading for more information on the `adsysctl cat` command.

Authorisations

ADSys uses a privilege mechanism based on polkit to manage authorisations. Many commands require elevated privileges to be executed. If the adsys client is executed with insufficient privileges to execute a command, the user will be prompted to enter its password. If allowed, then the command will be executed or denied otherwise.

This is configurable by the administrator, as any service controlled by polkit. For more information, see `man polkit`.

The `adsysctl` command

`adsysctl` is a command line utility to request actions from the daemon and query its current status. You can get more verbose output with the `-v` accumulative flags, which will stream all logs from the service corresponding to your specific request.

As a general rule, favour shell completion and the `help` command for discovering various parts of the `adsysctl` user interface. It will help you by completing subcommands, flags, users and even chapters of the offline documentation:

```
$ adsysctl doc
   Table of Contents
   1. [Welcome] ADSys: Active Directory Group Policy Integration
   2. [Prerequisites] Prerequisites and Installation
[...]
```

Best Practices

SSSD

It’s important to keep a few things in mind when considering the security of such an authentication solution.

External authentication

We are entrusting an external system with the decision of authenticating a user, and, ultimately, allowing or not a login on this computer. To avoid a *KDC spoofing* attack, SSSD has a setting called `krb5_validate` which, when set to `True`, will verify the KDC server it is talking to. This setting defaults to `True` when using the `ad` `id_provider`, which is the case of this whitepaper.

Offline login

In order to allow disconnected operation, SSSD defaults to allowing offline logins. This is done by keeping a local cache of the credentials and account information that were used the last time the system was connected to the network and the user logged in. It’s a nice feature, but it means all this information is stored locally on the system, and for all users that logged in.

The directory where this information is stored is `/var/lib/sss/db` and it is protected using filesystem permissions, so that only root can access it.

There are many options to control the behaviour of offline logins. The most important ones are:

- **`cache_credentials`:** when set to True, user credentials will be stored on disk in a hashed format after a successful login

- **`offline_credentials_expiration`:** for how long, in days, should cached logins be allowed, if the authentication provider is offline

- **`account_cache_expiration`:** number of days that the whole user account information is kept in the cache

For more information, consult the `sssd.conf(5)` manpage.

System Keytab

When joining a domain, encryption keys are stored in the system keytab file located in `/etc/krb5.keytab:`

```
~$ sudo klist -k
Keytab name: FILE:/etc/krb5.keytab
KVNO Principal
----------------------------------
   5 AD-DESKTOP-1$@WARTHOGS.BIZ
   5 AD-DESKTOP-1$@WARTHOGS.BIZ
   5 AD-DESKTOP-1$@WARTHOGS.BIZ
   5 host/AD-DESKTOP-1@WARTHOGS.BIZ
   5 host/AD-DESKTOP-1@WARTHOGS.BIZ
   5 host/AD-DESKTOP-1@WARTHOGS.BIZ
```

```
    5 RestrictedKrbHost/AD-DESKTOP-1@WARTHOGS.BIZ
    5 RestrictedKrbHost/AD-DESKTOP-1@WARTHOGS.BIZ
    5 RestrictedKrbHost/AD-DESKTOP-1@WARTHOGS.BIZ
~$
```

These are secrets which identify the host and its services with the Active Directory controller, and can be treated as such. The filesystem permissions are by default `0600` and owner and group are `root`.

ADSys

Some group policies are directly managed by SSSD. For those, ADSys is not involved at all. This is the case of the Security Settings. Additionally, there are many Security Settings as defined in Windows and not managed by ADSys but still partially supported through SSSD.

In Windows Group Policy Management Editor, these keys are located in

```
    [FOREST.ROOT] > Computer Configuration > Windows Settings >
Security Settings
```

Troubleshooting SSSD

Logs and debug level

SSSD’s log files are located in /var/log/sssd/. There is one log file per service (pam, nss, sssd itself, ...). Increasing debug level is a way to collect useful information in these log files and help understand what is going on.

There are 2 ways to increase SSSD’s debug level. First, on the fly, with sssctl. For example to set it to level 6 run:

```
# sssctl debug-level 6
```

The second way is to add it to the configuration file then restart sssd. In the configuration file, you can set a different debug level for each service. For example:

```
[sssd]
domains = warthogs.biz
config_file_version = 2
services = nss, pam, ifp
[nss]
debug_level=6
[pam]
debug_level=6
[domain/warthogs.biz]
debug_level=6
default_shell = /bin/bash
[...]
```

The sssctl method doesn’t make the setting persistent but has the advantage of not having to restart the service.

Cache management

Caching is useful to speed things up, but it can get in the way big time when troubleshooting. It’s useful to be able to remove the cache while chasing down a problem. This can also be done with the `sssctl` tool from the `sssd-tools` package.

You can either remove the whole cache:

```
root@ad-desktop-1:~# sssctl cache-remove
SSSD must not be running. Stop SSSD now? (yes/no) [yes]
Creating backup of local data...
Removing cache files...
SSSD needs to be running. Start SSSD now? (yes/no) [yes]
root@ad-desktop-1:~#
```

Or just one element:

```
root@ad-desktop-1:~# sssctl cache-expire -u linda@warthogs.biz
```

Or expire everything:

```
root@ad-desktop-1:~# sssctl cache-expire -E
```

Conclusion

Using Active Directory to manage the identities of users and their desktops is a cornerstone requirement of the security and compliance strategy of many businesses. Ubuntu has been engineered to help IT managers streamline the device integration and management process. This document has explored methods of integration for reasons of IT policy compliance, ease of administration, and end-user experience. We have looked at various ways to enable user accounts configured in AD environments to seamlessly login to Ubuntu workstations and configure them through the use of GPOs and script execution.

There are both free and paid-for options to make the integration of Active Directory straightforward and manageable. Depending on your IT policy requirements, the time you have to commit to a solution and the expertise of your team your business should be able to achieve a good level of integration.

Further reading

SSSD reference documentation

- SSSD - System Security Services Daemon

The Ubuntu Server Guide

- A good starting point for everything Ubuntu-related, including sections on LDAP, Kerberos and Samba: https://ubuntu.com/server/docs

IETF RFC 2307 - An Approach for Using LDAP as a Network Information Service

- For directory administrator interested in knowing the exact purpose of all LDAP attributes used by NSS. http://www.ietf.org/rfc/rfc2307.txt

Kerberos Explained

- A dated but excellent article on Microsoft's implementation of the Kerberos protocol. The explanations are platform-agnostic. http://technet.microsoft.com/en-us/library/bb742516.aspx

Active Directory

There is an assumed knowledge of Active Directory setup and operations, for which Microsoft has developed an abundance of documentation:

- Active Directory Domain Services: https://docs.microsoft.com/en-us/windows-server/identity/ad-ds/active-directory-domain-services
- GPO management: https://docs.microsoft.com/en-us/troubleshoot/windows-client/group-policy/create-and-manage-central-store

ADSys

- All of the ADSys documentation is included in `adsysctl doc` but can also be found online: https://github.com/ubuntu/adsys/wiki/

Every effort has been made by Canonical to ensure the accuracy of this document but Canonical disclaims, to the extent possible at law, any liability for any error or omission. © Canonical Limited 2024. Ubuntu and associated logos are registered trademarks of Canonical Ltd., all rights reserved. All other trademarks are the properties of their respective owners. Any information contained in this document may change without notice and Canonical is not held responsible for any such changes.

Canonical Limited, Registered in Isle of Man, Company number 110334C, Registered Office: 2nd Floor, Clarendon House, Victoria Street, Douglas IM1 2LN, Isle of Man, VAT Registration: GB 003 2322 47

Canonical