

Dell EMC DD BoostFS for Linux

Configuration Guide

7.0

Notes, cautions, and warnings

 **NOTE:** A NOTE indicates important information that helps you make better use of your product.

 **CAUTION:** A CAUTION indicates either potential damage to hardware or loss of data and tells you how to avoid the problem.

 **WARNING:** A WARNING indicates a potential for property damage, personal injury, or death.

Figures.....	5
Tables.....	6
Chapter 1: Introduction to BoostFS for Linux.....	7
Revision history.....	7
Introduction to BoostFS.....	7
Supported environments.....	8
Supported applications.....	8
Chapter 2: Preparing the system for BoostFS.....	9
Prepare the system for BoostFS.....	9
Set the system hostname and domain name.....	10
Prepare the system for Kerberos authentication.....	10
Join a PowerProtect or Data Domain system to an Active Directory domain.....	10
Configure PowerProtect or Data Domain systems for the UNIX KDC.....	11
BoostFS and existing DD OS commands.....	11
Assign multiple users to BoostFS.....	11
Create storage units.....	12
Logical stream limits for storage units (optional).....	13
Client Groups and BoostFS.....	13
Distributed segment processing option.....	13
Chapter 3: Installing BoostFS for Linux.....	14
Installation overview.....	14
Components of the BoostFS for Linux client.....	14
BoostFS on Linux systems	14
The role of FUSE in BoostFS for Linux.....	15
Upgrade the BoostFS client.....	15
Chapter 4: Configuring and using BoostFS for Linux.....	16
The BoostFS for Linux configuration file.....	16
BoostFS for Linux command overview.....	18
BoostFS and high availability.....	18
Authentication methods.....	18
RSA Lockbox-based authentication.....	18
Shared lockbox files.....	18
Kerberos-based authentication.....	20
Considerations for Kerberos authentication.....	22
boostfs mount.....	23
Command options for boostfs mount.....	23
Compressed restoration.....	24
Maximum connections for boostfs mount.....	25
BoostFS and the Linux mount command.....	25
Automounter.....	26

BoostFS client connection details.....	27
Chapter 5: Troubleshooting.....	28
Log information.....	28
Known issues.....	28
Appendix A: Appendix.....	31
About Puppet.....	31
Adding principals to the Unix KDC.....	31
Add DD principals to the UNIX KDC.....	31
Add client principals to the KDC.....	31
References.....	32

1	ddboost show connections display.....	27
---	---------------------------------------	----

1	Revision history of BoostFS for Linux Configuration Guide, version 7.0.....	7
2	Terminology.....	8
3	Command options for boostfs mount.....	23
4	mount command options.....	25

Introduction to BoostFS for Linux

Topics:

- [Revision history](#)
- [Introduction to BoostFS](#)
- [Supported environments](#)
- [Supported applications](#)

Revision history

The following table presents the revision history of this document.

Table 1. Revision history of BoostFS for Linux Configuration Guide, version 7.0

Revision	Date	Description
04	June 2021	Editorial updates.
03	September 2020	Editorial updates.
02	August 2020	Kerberos usage modified.
01	September 2019	This revision includes support for the PowerProtect DD6900, PowerProtect DD9400, and PowerProtect DD9900 systems.

Introduction to BoostFS

DD Boost Filesystem (BoostFS) 7.0 provides a general file-system interface to the DD Boost library, allowing standard backup applications to take advantage of DD Boost features.

Advantages of BoostFS

By leveraging the DD Boost technology, BoostFS helps reduce bandwidth, can improve backup-times, offers load-balancing, allows in-flight encryption, and supports the DD multi-tenancy feature set.

As a file server system implementation, the BoostFS workflow is similar to NFS but also leverages the DD Boost protocol. In addition, BoostFS improves backup times compared to NFS and various copy-based solutions.

BoostFS supports single-node Data systems, high-availability (HA) systems, DD Virtual Edition, and Extended Distance Protection.

Purpose

This document describes how to install and configure BoostFS on client systems.

Terminology

Table 2. Terminology

Term	Definition
FUSE	Filesystem in User Space (FUSE) is an open-source interface that enables non-privileged users to securely create and mount their own file-system implementations.
Puppet	An open-source software configuration management tool. For more information, see About Puppet on page 31.
Push	A process that involves using a centralized server to connect to specified clients and run commands remotely; for BoostFS, that means downloading and remotely installing the installation package on each client.

Supported environments

Environments that use BoostFS 7.0 must meet the following specifications.

BoostFS for Linux requires the following:

- DD OS version 6.0 or later
- FUSE 2.8 or later

The following Linux distributions are supported:

- Red Hat Enterprise Linux versions 6 and 7
- CentOS 7
- SUSE Linux Enterprise Server versions 11 and 12
- Ubuntu 14.04 and 15
- Oracle Linux version 7


Supported applications

BoostFS for Linux supports the following applications:

- Commvault Simpana versions 10 and 11
- MySQL Community 5.6. and 5.7
- MySQL Enterprise Manager 5.6 and 5.7
- MongoDB Community 2.6, 3.0, and 3.2

Boost features supported by BoostFS

Transport Layer Security (TLS) anonymous authentication is supported to provide encryption.

 **NOTE:** If you select TLS, be aware that there is no configuration option to enable TLS from the client. It must be enabled through the PowerProtect or Data Domain System.

Boost features not supported by BoostFS

- Managed File Replication (MFR)
- DD Boost-over-Fibre Channel (DFC)
- Retention Lock

Preparing the system for BoostFS

Topics:

- [Prepare the system for BoostFS](#)
- [Set the system hostname and domain name](#)
- [Prepare the system for Kerberos authentication](#)
- [BoostFS and existing DD OS commands](#)
- [Assign multiple users to BoostFS](#)
- [Create storage units](#)
- [Logical stream limits for storage units \(optional\)](#)
- [Client Groups and BoostFS](#)
- [Distributed segment processing option](#)

Prepare the system for BoostFS

Every system that is enabled for DD Boost deduplication must have a unique name. You can use the system DNS name, which is always unique.

Prerequisites

Ensure that all your systems can access the Key Distribution Center (KDC). In a Windows environment, the Windows server that hosts the Microsoft Active Directory service acts as the KDC and the domain name system (DNS). If the systems cannot reach the KDC, check the DNS settings at `/etc/resolv.conf`.

Steps

1. On the PowerProtect or Data Domain system, log in as an administrative user.
2. Verify that the file system is enabled and running by entering:

```
$ filesystem status
The file system is enabled and running.
```

3. Verify that DD Boost is already enabled:

```
$ ddboost status
DD Boost status: enabled
```

If the DD Boost status is reported as disabled, enable it by entering:

```
$ ddboost enable
DD Boost enabled
```

4. Verify that distributed segment processing is enabled:

```
ddboost option show
```

You should see the following output:

Option	Value
distributed-segment-processing	enabled
virtual-synthetics	enabled
fc	disabled
global-authentication-mode	none
global-encryption-mode	medium

If distributed segment processing is shown as disabled, enable it by entering:

```
ddboost option set distributed-segment-processing enabled
```

NOTE:

- If secure multi-tenancy (SMT) is used, the user role must be set as none.
- Users who run backup applications that connect to Power Protect or Data Domain systems must have their user names configured on that system. For more information, see the *DD OS Administration Guide*.
- Multiple applications can use DD Boost to access a Power Protect or Data Domain system, and multiple users can be configured for DD Boost access. The username, password, and role must have already been set up using the DD OS `user add` command:

```
user add <user> [password <password>]  
[role {admin | limited-admin | security | user | backup-operator | data-access}]  
[min-days-between-change <days>] [max-days-between-change <days>]  
[warn-days-before-expire <days>] [disable-days-after-expire <days>]  
[disable-date <date>] [force-password-change {yes | no}]
```

For example, to add a user with a login name of **jsmith** and a password of **mP34\$muk*E** with administrative privilege, enter:

```
$ user add jsmith password mP34$muk*E role admin
```

Once the user has been created, the user must be made a DD Boost user. To add **jsmith** to the DD Boost user list, enter:

```
$ ddboost user assign jsmith
```

Set the system hostname and domain name

Set the system host name and the domain name in DD OS using the **net set** CLI command.

Steps

In DD OS, type the following:

```
# net set hostname [host]  
# net set {domain name [local-domain-name]}
```

For more information on **net** commands, see the *DD OS Command Reference Guide*.

Prepare the system for Kerberos authentication

Join a PowerProtect or Data Domain system to an Active Directory domain

About this task

Joining the system to an Active Directory domain is required for Kerberos authentication in an Active Directory environment. If you do not plan to use Kerberos in your implementation, this procedure is not required.

For more information about Kerberos authentication, see [Configure the BoostFS client for Kerberos authentication](#) on page 21.

NOTE: You must use a NIS server to map Kerberos users to Unix user IDs.

Steps

1. To join a system to an Active Directory domain, type the following command:

```
# authentication kerberos set realm <domain> kdc-type windows
```

You are prompted to type credentials for the domain.

2. Type the domain username and password.

Results

If the credentials are valid, the system is joined to the Active Directory domain. The use of this command does not enable CIFS.

Configure PowerProtect or Data Domain systems for the UNIX KDC

Steps

1. Rename the `keytab_file_for_ddsystem` file located on the `/ddvar/releases` directory to `krb5.keytab`.
See [Add client principals to the KDC](#) on page 31 for information on creating the DD OS keytab file.
2. In DD OS, import the keytab file moved in Step 1 to `/ddr/etc` using the following command:
authentication kerberos keytab import
3. Confirm the configuration using the `authentication` command:
authentication kerberos show config
4. In DD OS, set the realm using the `authentication` command:
authentication kerberos set realm <realm> kdc-type unix kdc <KDC-hostname>

BoostFS and existing DD OS commands

You must create one or more storage units on each PowerProtect or Data Domain system enabled for BoostFS. System administrators can use existing DD OS CLI commands to create and manage storage units used by BoostFS.

Assign multiple users to BoostFS

When, as a system administrator, you create the storage units that users employ with the backup applications, you associate a username with each storage unit. This associated username can be changed after creation of the storage unit.

Storage units are accessible only to applications with the username that owns the storage unit.

Each storage unit is owned by one username, and the same username can own multiple storage units. The application passes the username and password to BoostFS, and DD Boost passes them to the PowerProtect or Data Domain system when attempting to connect to the system. The system then authenticates the username and password. The username and password can be shared by different applications.

When a storage unit is created with a valid local user but not assigned to DD Boost, the user is automatically added to the DD Boost users list in the same way that a user is added via the `ddbboost user assign` command.

Assign one or more users to the DD Boost users list:

```
$ ddbboost user assign user1 user2
User "user1" assigned to DD Boost.
User "user2" assigned to DD Boost.
```

To verify and display the users in the users list, enter:

```
$ ddbboost user show
```

DD Boost user	Default tenant-unit	Using Token Access
user1	Unknown	Yes
user2	Unknown	-
user3	Unknown	Yes
user4	Unknown	-
user5	Unknown	-
user6	Unknown	-
user7	Unknown	Yes

```
user8          Unknown          -
-----
```

To unassign the user from the users list, enter:

```
$ ddboost user unassign user1
User "user1" unassigned from DD Boost.
```

Create storage units

You need to create one or more storage units on each PowerProtect or Data Domain system enabled for BoostFS.

Steps

1. Create a storage unit in DDOS:

```
$ ddboost storage-unit create NEW_STU1 user user1
Created storage-unit "NEW_STU1" for "user1".
```

A storage unit name must be unique on any given PowerProtect or Data Domain system. However, the same storage unit name can be used on different systems.

The username owns the storage unit and ensures that only connections with this username's credentials are able to access this storage unit. See the section on `ddboost storage-unit` commands in the *DD OS Command Reference Guide* for details on command options.

2. Repeat the previous step for each storage-unit needed in DD OS.
3. If you want to modify a DD OS storage unit, enter:

```
$ ddboost storage-unit modify NEW_STU1 user user2
Storage-unit "NEW_STU1" modified for user "user2".
```

The `ddboost storage-unit modify` command allows the backup application to change the username ownership of the storage unit. Changing the username does not require that attributes of every file on the storage unit be changed.

4. Display the users list for the storage units:

```
$ ddboost storage-unit show
```

After entering the command, the output you see should be similar to the following:

```
# ddboost storage-unit show
Name                Pre-Comp (GiB)   Status   User          Report Physical
                  Size (MiB)
-----
backup              3.0             RW       sysadmin      -
DDBOOST_STRESS_SU   60.0            RW       sysadmin      -
task2               0.0             RW       sysadmin      -
tasking1            0.0             RW       sysadmin      -
DD1                 0.0             RW       sysadmin      -
D6                  5.0             RW       sysadmin      -
TEST_DEST           0.0             D        sysadmin      -
STU-NEW             0.0             D        ddul          -
getevent            0.0             RW       ddul          -
DDP-5-7             120.0           RW       sysadmin      -
TESTME              150.0           RW       sysadmin      -
DDP-5-7-F           100.0           RW       sysadmin      -
testSU              0.0             RW       sysadmin      200
-----
D      : Deleted
Q      : Quota Defined
RO     : Read Only
RW     : Read Write
RD     : Replication Destination
```

Next steps

If you are using Kerberos authentication in your implementation, you must create an Active Directory user with the same name as the storage-unit user.

Logical stream limits for storage units (optional)

BoostFS is restricted to the same stream limit and storage quota features as DD Boost. See the *DD Boost for Partner Integration Administration Guide* for more information.


Client Groups and BoostFS

The Client Group feature identifies specific client loads when clients are associated with groups.

The `client_group` command set is supported only for clients that use DD Boost or NFS protocols. For more information about Client Groups, see the *DD OS Command Reference Guide*.

Distributed segment processing option

BoostFS supports distributed segment processing as supported by DD Boost. For more information, refer to the *DD OS Administration Guide*.

 **NOTE:** Enabling or disabling the distributed segment processing option does not require a file system restart.

Installing BoostFS for Linux

Topics:

- [Installation overview](#)
- [Components of the BoostFS for Linux client](#)
- [The role of FUSE in BoostFS for Linux](#)
- [Upgrade the BoostFS client](#)

Installation overview

There is a single RPM installation package for BoostFS for Linux that both enterprise and small-scale users can download. It is available in both RPM and .deb formats. The RPM package includes the boostfs executable.

Check the following before beginning the process:

- The FUSE version on the client must be 2.8 or higher.

While the BoostFS process is running:

- BoostFS mount points must be deactivated.
- You cannot upgrade BoostFS.
- You cannot uninstall BoostFS.

Components of the BoostFS for Linux client

The BoostFS for Linux client is composed of the following:


- A daemon process that supports various commands
- Two shared libraries: `libDDBoost.so` and `libDDBoostFS.so`
- `.rsalib`: A hidden directory that contains redistributable RSA libraries
- A configuration file
- A manual page

`libDDBoost.so`, a FUSE-agnostic library built on the DD Boost library, provides such services as connection management, a retry mechanism, and client logging.

The packaging defaults to the Red Hat Package Manager (RPM) format, but the native packaging for other operating systems is also supported.

The following packages are available:

- Ubuntu: `DDBoostFS-1.1.0.1-565134_amd64.deb`
- Red Hat: `DDBoostFS-1.1.0.1-565134.rhel.x86_64.rpm`
- SUSE: `DDBoostFS-1.1.0.1-565134.sles.x86_64.rpm`

 **NOTE:** Verify that you are using the appropriate package for your client OS.

BoostFS on Linux systems

Employing the Linux Filesystem Hierarchy Standard 3.0, the BoostFS for Linux client is installed in `/opt/emc/boostfs` and contains the following subdirectories:

- `bin`: boostfs command(s) are installed here.
- `lib`: Contains these libraries.
 - `libDDBoost.so`

- `libDDBoostFS.so`
- `.rsalib`: Contains redistributable RSA libs.
- `etc`: Contains configuration files (sample and production).
- `man`: Contains standard man pages.

The role of FUSE in BoostFS for Linux

BoostFS for Linux uses FUSE, an open-source software interface that enables non-privileged users to securely create and mount their own file-system implementations.

FUSE allows you to export a virtual file system to the Linux kernel. Write operations through BoostFS and FUSE benefit from DD OS distributed segment processing.


Using FUSE and the DD Boost plug-in, BoostFS exports a storage unit on a PowerProtect or Data Domain system to a mountpoint on a client. On the client, file system operations conducted on the mountpoint are captured by the kernel before being passed through FUSE to BoostFS.

BoostFS runs as a daemon on a client. As a software module, BoostFS serves as a layer between FUSE and DD Boost.

BoostFS in this release is only supported on some Linux systems in the initial release. For a list of supported environments, see [Supported applications](#) on page 8

FUSE consists of three parts:

- A kernel module: `fuse.ko`
- A user space library: `libfuse`
- A mount utility: `fusermount`


 **NOTE:** BoostFS requires the "user_allow_other" option for FUSE; it will add the option to the `/etc/fuse.conf` file if it is not already present. Be aware that this may change the behavior of other FUSE-based applications you are using.

Upgrade the BoostFS client

Upgrade BoostFS for Linux using the BoostFS RPM package. Before performing the upgrade, you must stop all BoostFS processes.

About this task

The shared lockbox feature is introduced in BoostFS 1.1. When you upgrade from BoostFS 1.0 to BoostFS 1.1 or later, you must create a new lockbox and add current user credentials.

 **NOTE:** If you are upgrading from BoostFS 1.1 or later, this procedure is not required.

If you use the BoostFS lockbox for user authentication, you must perform the following steps to upgrade:

Steps

1. Upgrade BoostFS to 1.1 or later.
2. Remove all previous lockbox files:

```
# rm /opt/emc/boostfs/lockbox/*
```

3. Create the new lockbox by entering user credentials with the `boostfs lockbox set` command:

```
# /opt/emc/boostfs/bin/boostfs lockbox set <parameters>
```

4. Enter the remaining user credential pairs as needed.

Results

BoostFS is upgraded with the new lockbox ready for authentication use. See [Shared lockbox files](#) on page 18 for more information about configuring a common lockbox file for all BoostFS clients.

Configuring and using BoostFS for Linux

Topics:

- [The BoostFS for Linux configuration file](#)
- [BoostFS for Linux command overview](#)
- [BoostFS and high availability](#)
- [Authentication methods](#)
- [boostfs mount](#)
- [BoostFS and the Linux mount command](#)
- [Automounter](#)
- [BoostFS client connection details](#)

The BoostFS for Linux configuration file

The Boost Filesystem has two configuration options.

- Command-line interface (CLI)
- The configuration file: `boostfs.conf`

This file is located in `/opt/emc/boostfs/etc`, and can be edited by the "root" user or someone with sudo privileges.

Parameters can be specified either in the config file or on the command line, or both.

The configuration file has a global section and a mount-point specific section. Configuration parameters configured using the command line take the highest priority and override any values in the config file. Mount-specific parameter values override global parameter values.

The following is a sample configuration file:

```
#####
# BoostFS 1.3 example input file
#
#
# The configuration file is divided into sections, delineated by brackets [].
# Options that are to apply to all mount points are in the [global] section.
# More details on the various configuration options can be found in the
# BoostFS manual. Command line options override what is in this file.
#
# Format:
# # - Identifies a comment line, and must be at the start. Configuration
# parameters can be disabled by adding a "#" to the start of the line.
#
# Values which contains spaces should use double quotations around the
# entire value.
#
# No whitespace is allowed between the option and the value, i.e.
# log-dir = /path is not allowed.
#
# Comments are not allowed after the option value pair.
#
#####

[global]
# Data Domain Hostname or IP address
# data-domain-system=dd2500-1.yourdomain.com

# Storage Unit
# storage-unit=su-name

# Security option used for authentication (default: lockbox)
```



```

# security=<krb5|lockbox>

# Storage Unit Username (should only be used in conjunction with Kerberos authentication)
# storage-unit-username=sysadmin

# Subdirectory within the storage-unit to mount to
# directory-name=path/to/subdir

# Lockbox path (default: /opt/emc/boostfs/lockbox/boostfs.lockbox)
# lockbox-path=path/to/lockbox

# Enable logging (default: true)
# log-enabled=<true|false>

# Log level (default: info)
# log-level=<debug|info|warning|error>

# Directory for log files (default: /opt/emc/boostfs/log)
# log-dir=/path/to/log

# Log file name (default: ddbostfs_<uid>_<gid>.log)
# log-file=output.log

# Maximum log size in MB (default: 100MB)
# log-maxsize=100

# Number of log files to save (default: 8)
# log-rotate-num=10

# Text string that describes the application using boostfs with additional information
such as the version.
# app-info="text_string"

# Allow users other than the owner of the mount to access the mount
# allow-others=<true|false>

# Automatically renew Kerberos tickets when Kerberos authentication is used (default: true)
# krb-auto-renew=<true|false>

#
# Mount point sections are delineated by [mountpoint]
#

# [/path/to/mount]
# Data Domain Hostname or IP address
# data-domain-system=dd2500-1.yourdomain.com

# Storage Unit
# storage-unit=su-name

# Security option used for authentication (default: lockbox)
# security=<krb5|lockbox>

# Storage Unit Username (should only be used in conjunction with Kerberos authentication)
# storage-unit-username=sysadmin

# Subdirectory within the storage-unit to mount to
# directory-name=path/to/subdir

# Enable Boost multithreading (default: true)
# mtboost-enabled=<true|false>

# Number of threads to use in multithreaded Boost mode for writing each file (default: 2)
# This does not have any significance if mtboost-enabled=false
# Min value is 0 (this means mtboost-threads will be intelligently calculated by boostfs
by querying CPU information)
# Max value is 16
#
# mtboost-threads=16

# Maximum number of connections that can be used at the same time (default: 128).
# Min value is 64. Max value is 256.
# max-connections=128

```

```
# Enable compressed restoration (default: false).
# When set to true, the server conducts data compression before sending to the client.
# Correspondingly, when the client receives data, it needs to conduct decompression first.
# By sending compressed data over the network, bandwidth usage can be reduced. However,
# use this option with caution since it requires significant amount of CPU power to conduct
# compression on the server and to conduct decompression on the client.
# ddboost-read-compression=<true|false>
```

BoostFS for Linux command overview

You use the `boostfs` command to establish the FUSE mount, create the lockbox (if desired), and set up Kerberos credentials if you choose Kerberos as the authentication method.

For detailed information about a BoostFS command, see the corresponding `man` page entry.

BoostFS and high availability

If you are configuring a high availability (HA) system, you should make sure the IP address (or hostname) that you specify for the system is one of the floating IP addresses. Only the floating IP addresses in an HA system are accessible after a failover.

If you incorrectly specify one of the fixed HA addresses, you will not be able to connect to the PowerProtect or Data Domain system in the event of a recoverable failure.

Authentication methods

BoostFS has two authentication options:

- RSA Lockbox (default)
- Kerberos

RSA Lockbox-based authentication

RSA Lockbox is the default password manager for BoostFS for Linux.

To use RSA Lockbox, you need to set the lockbox using the `boostfs lockbox set` command. Beginning with BoostFS 1.1, you can also set up a shared BoostFS lockbox file.

Shared lockbox files

Beginning with BoostFS 1.1, you can create a common lockbox file for all BoostFS clients. This feature allows you to avoid having to create a separate lockbox file for each unique BoostFS client.

Sharing a common lockbox file enables you to create a single management point for BoostFS clients to access BoostFS mount points on PowerProtect or Data Domain systems.

Lockbox files created with BoostFS 1.0 are incompatible with BoostFS 1.1. To solve this problem, you must erase a BoostFS 1.0 installation if you have one, and then install BoostFS 1.1.


Due to a change in the Lockbox format, you must recreate your Lockbox when upgrading from BoostFS 1.0 to BoostFS 1.1. To do this, remove the files located under `/opt/emc/boostfs/lockbox/`. Then after upgrading, re-enter any credentials using the `boostfs lockbox set` command.

i NOTE: A BoostFS 1.0 client can use a lockbox created with a 1.1 or later client, but a 1.1 or later client cannot use a lockbox created with a 1.0 client. A BoostFS 1.1 or later client can use a lockbox created with a 1.1 or later client, even if they are not the same version.

Create the Lockbox on the primary client

Prerequisites

Verify that BoostFS is installed on the server that manages access to the shared Lockbox.

 **NOTE:** The command `boostfs lockbox set` fails if there is an existing Lockbox file in the same location.

About this task

In this example, `/mnt/share/lockbox/dir/` represents the NFS path that is accessible by all clients.

Steps

1. Create the Lockbox with the `-l` option:

```
boostfs lockbox set -u <storage-unit-username> -d <data-domain-system> -s <storage-unit>
-l /mnt/share/lockbox/dir/boostfs.lockbox
```

You can also specify the `lockbox-path` in the configuration file.

2. Repeat the `lockbox set` command for each PowerProtect or Data Domain system or storage unit that needs to be accessed by the Lockbox.

Use the shared Lockbox on other clients

Prerequisites

Create a shared Lockbox and add credentials for the PowerProtect or Data Domain systems and storage units that need access to the Lockbox.

About this task

In this example, `/mnt/share/lockbox/dir/` represents the NFS path that is accessible by all clients.

Steps

1. To allow access to the Lockbox for the other clients, type the following command on the primary client:

```
boostfs lockbox add-hosts -l /mnt/share/lockbox/dir/boostfs.lockbox
client1.dell.com client2.dell.com
```

In this example, clients with the hostname `client1.dell.com` and `client2.dell.com` are allowed access to the shared Lockbox.

2. On each client that needs access to the shared Lockbox, specify the path to the shared Lockbox by either:
 - Using the `mount` command:

```
boostfs mount -d <data-domain-system> -s <storage-unit> -l /mnt/share/lockbox/dir/
boostfs.lockbox
```

- Editing the configuration file:

```
[global]
lockbox-path=/mnt/share/lockbox/dir/boostfs.lockbox
```

View clients accessing the shared lockbox

About this task

Clients allowed to access the lockbox can be viewed by using `lockbox show-hosts` command as below:

```
/boostfs lockbox show-hosts -l /mnt/nfsshare/boostfs.lockbox
```

In this example the output will be:

```
./boostfs lockbox show-hosts -l /mnt/nfsshare/boostfs.lockbox
Security Access List:
  client1.dell.com
  client2.dell.com
```

Modify the shared Lockbox

About this task

Only the primary client can modify the Lockbox file. Other clients encounter an error when they try to modify the Lockbox. Other clients are still able to query the Lockbox.

In this example, `/mnt/share/lockbox/dir/` represents the NFS path that is accessible by all clients.

Steps

1. To remove client access:

```
boostfs lockbox delete-hosts -l /mnt/share/lockbox/dir/boostfs.lockbox client2.dell.com
```

NOTE: After removing a client from the Lockbox, the client can no longer use the Lockbox and can no longer access any of the PowerProtect or Data Domain systems defined in the Lockbox.

2. To remove a Lockbox entry:

```
boostfs lockbox remove -d <data-domain-system> -s <storage-unit> -l /mnt/share/lockbox/dir/boostfs.lockbox
```

NOTE: After removing a PowerProtect or Data Domain system or storage unit from those that the Lockbox grants access to, none of the clients that use the Lockbox can access the system or storage unit.

Kerberos-based authentication

BoostFS Linux supports the MIT implementation of Kerberos authentication as an alternative to RSA lockbox authentication.

There are three main entities involved with Kerberos Authentication:

- BoostFS client
- Kerberos Key Distribution Center (KDC), which can be on either one of the following:
 - An Active Directory server on a domain controller in a Windows environment
 - A POSIX-based operating system with optional NIS lookups
- PowerProtect or Data Domain system running DD OS version 6.0 or later

The Kerberos file contains a "shared secret" (a password, pass phrase, or other unique identifier) between the KDC server and the PowerProtect or Data Domain system.

In an Active Directory environment, the Windows server that hosts the Active Directory service also acts as the Key Distribution Center (KDC) and also a domain name system (DNS). When using a UNIX KDC, the DNS server does not have to be the KDC server; it can be a separate server.


NOTE: Before using Kerberos for BoostFS, you should verify that the Kerberos client libraries for your Linux distribution are installed on your machine.

Kerberos tickets

To authenticate using Kerberos, you must acquire a Ticket Granting Ticket (TGT) for two types of user accounts:

- A Kerberos Ticket Granting Ticket (TGT)
- A Kerberos ticket for various services (service tickets) that the client will use (BoostFS, DNS, CIFS, NFS, etc.)

Each user only has access to the tickets they create with the BoostFS Kerberos commands. Users cannot access tickets that others have created.

 **NOTE:** Because the Kerberos authentication implementation in BoostFS 1.3 is different than in previous releases, you must re-acquire your tickets before using this release.

For more detailed information about using Kerberos with BoostFS, see [Considerations for Kerberos authentication](#) on page 22.

Configure the BoostFS client for Kerberos authentication

Kerberos authentication uses tickets to authenticate instead of a username and password.

Prerequisites

Verify that each of the following requirements are met:

- The PowerProtect or Data Domain system and the client resolve DNS for each other.
- The client points to the correct Key Distribution Center (KDC). Verify by checking the `/etc/krb5.conf` file.
- The PowerProtect or Data Domain system, client, and KDC system clocks must be within five minutes of each other. Using an NTP server is a reliable way to keep the clocks synchronized.
- There must be a user in the Kerberos realm with the same name as the storage-unit user local to the PowerProtect or Data Domain system. You must use the Kerberos realm credentials to acquire the storage-unit user ticket, not the credentials local to the PowerProtect or Data Domain system.

If you are using a Unix KDC, you must add the following principals:

- Service principal for the PowerProtect or Data Domain system
- Host principal for the PowerProtect or Data Domain system
- Service principal for BoostFS
- User principal for the storage-unit user

This step is not required if an Active Directory server is acting as the KDC. For more information, see [Adding principals to the Unix KDC](#) on page 31.

Steps

1. Acquire a storage-unit user TGT.
This TGT grants access to the mount point and is required to mount BoostFS. For more information, see [Acquire the storage-unit user ticket](#) on page 21.
2. Mount BoostFS as the storage-unit user.
For more information, see [Mount BoostFS](#) on page 22.
3. Acquire a primary Kerberos user TGT.
This TGT determines access to files and directories within the mount point. Each user that requires file access after mounting BoostFS must have a primary user ticket. For more information, see [Acquire the primary user ticket](#) on page 22.

Acquire the storage-unit user ticket

The storage-unit user TGT grants access to the mount point and is required to mount BoostFS.

Prerequisites

Review the prerequisites in [Configure the BoostFS client for Kerberos authentication](#) on page 21.

Steps

1. To create a storage-unit user ticket, use the `kerberos set` command with the `-u` option and specify the storage-unit username:

```
# boostfs kerberos set -u <storage-unit-username>
```

 **NOTE:**

- You must use the Kerberos realm credentials to acquire the storage-unit user ticket, not the credentials local to the PowerProtect or Data Domain system.

- To allow other users on the client system to mount BoostFS, include the option `-o allow-others=true`. This option can only be changed by the root user.

2. (Optional) To verify the creation of the storage-unit user ticket, use the `-u` option and specify the storage-unit username:

```
# boostfs kerberos query -u <storage-unit-username>
```

Mount BoostFS

About this task

For more information about mounting BoostFS, see [Considerations for Kerberos authentication](#) on page 22 and [BoostFS and the Linux mount command](#) on page 25.

Steps

Mount BoostFS and specify Kerberos authentication:

```
boostfs mount -d <data-domain-system> -s <storage-unit> -o security=krb5 -o <storage-unit-username=mount-point>
```

Results

BoostFS is mounted, but inaccessible.

Acquire the primary user ticket

Steps

1. To create a primary Kerberos user ticket, use the `-m` option and specify the primary Kerberos username:

```
# boostfs kerberos set -m <primary-username>
```

2. (Optional) To verify the creation of a primary Kerberos user ticket, use the `-m` option without specifying a username:

```
# boostfs kerberos query
```

Results

The client configuration is complete.

Considerations for Kerberos authentication

Kerberos implementation

BoostFS uses MIT Kerberos, which has a separate configuration file located at `/etc/krb5.conf`. This configuration file can be used to control ticket lifetime and make other changes to the Kerberos implementation. For additional information about changing the Kerberos configuration, and other information not specific to the BoostFS implementation, refer to MIT Kerberos documentation.

The credential for the storage-unit user is stored in `/opt/emc/boostfs/kerberos`. The credential for the primary Kerberos user is stored in `/opt/emc/boostfs/kerberos/<primary-username>/<process-username>`.

Security and file permissions

To allow other users on the client system to mount BoostFS, include the option `-o allow-others=true` when using `boostfs kerberos set -u <storage-unit-username>`. When this option is used, the storage-unit user ticket is

shared with any user of the mount point. However, other users must still have their own primary user ticket to access the files and directories within the mount. This option can only be changed by the root user.

By default, the `local-user-security` parameter is set to `false`. When using RSA Lockbox authentication, this setting can be changed to `true`. When using Kerberos authentication it is always set to `true` and ignores any conflicting options in the configuration file.

Any files created through the primary Kerberos user's connection to the BoostFS mount are owned by that primary user. These files can only be changed by a user with the same TGT.

You can optionally configure the client access list for DD Boost on the PowerProtect or Data Domain system to only use Kerberos authentication by typing the following command on the system: `# ddboost client add <client-name> authentication-mode kerberos`

NOTE: If you perform this optional step, note that a BoostFS client configured to use Kerberos must use Kerberos for the connection to succeed. If that BoostFS client uses RSA Lockbox, the connection will fail.

Renewing tickets

When the `krb-auto-renew` option is used, tickets are automatically renewed up to their renewable time. Once the renewable time has been exceeded, you must manually acquire the ticket again using the BoostFS Kerberos commands.

boostfs mount

The `boostfs mount` command allows you to establish the BoostFS FUSE mount.
`boostfs mount [-d|--data-domain-system] <data-domain-system> [-s|--storage-unit] <storage-unit> [[-o|--option <param>=<value>] ...] <mount-point>`

Mount the BoostFS file system. Role required: none.

`boostfs umount <mount-point>`

Unmount the BoostFS file system. Role required: none.

Argument Definitions

- mount-point** The mount-point for the BoostFS system.
- storage-unit** The target storage-unit on the protection system.

Command options for boostfs mount

The following options are valid for the `boostfs mount` command.

Table 3. Command options for boostfs mount

Option	Description
<code>-o allow-others=<true false></code>	Allow users on the client system other than the owner of the mount to mount BoostFS. Default value: <code>false</code> For more information, see Considerations for Kerberos authentication on page 22
<code>-o app-info="text_string"</code>	Display a text string describing the application using BoostFS. Default value: <code>FUSE version</code>
<code>-o ddboost-read-compression=<true false></code>	Enable compressed restoration. Default value: <code>false</code>

Table 3. Command options for `boostfs mount` (continued)

Option	Description
	For more information, see Compressed restoration on page 24
<code>-o directory-name=path/to/subdir</code>	Subdirectory within the storage-unit you select for mounting (default: root of the storage unit). You must create the subdirectory after mounting at the root path, unmounting, and adding the parameter to the subsequent mount command or configuration file.
<code>-o krb-auto-renew=<true false></code>	Allow tickets to be automatically renewed up to their renewable time. Once the renewable time is exceeded, you must manually acquire the ticket again using the BoostFS Kerberos commands. Default value: <code>false</code>
<code>-o log-enabled=<true false></code>	Enable or disable logging. Default value: <code>true</code>
<code>-o log-level=<debug info warning error></code>	Set the log detail level. Default value: <code>info</code>
<code>-o log-dir=/path/to/logfile</code>	Specify the directory for log files. Default value: <code>/opt/emc/boostfs/log</code>
<code>-o log-file=unique-file-name.log</code>	Specify the log file name. Default value: <code>ddboostfs_<uid>_<gid>.log</code>
<code>-o log-maxsize=100</code>	Specify the maximum log size in MB. Default value: <code>100</code>
<code>-o log-rotate-num=8</code>	Specify the number of log files to save. Default value: <code>8</code>
<code>-o max-connections=128</code>	Specify the maximum number of connections that can be used at the same time. Default value: <code>128</code> For more information, see Maximum connections for boostfs mount on page 25.
<code>-o security=<krb5 lockbox></code>	Specify the security option used for authentication Default value: <code>lockbox</code>
<code>-o storage-unit-username=sysadmin</code>	Specify the storage unit user name. Use only with Kerberos.

Compressed restoration

This option reduces bandwidth usage when sending and receiving data, but increases CPU usage.

When the mount option `ddboost-read-compression` is set to `true`, data is compressed on the server before being sent to the client. When the client receives the data, it must decompress the data. Sending and receiving compressed data uses less network bandwidth, but compressing and decompressing the data requires a significant amount of CPU power. By default, this option is set to `false`.

This option can be used in one of the following two ways:

- As a command-line option:



```
boostfs mount -o ddbboost-read-compression=true /mnt/bfs-mount
```

- As an option configured in the `boostfs.conf` file:

```
ddbboost-read-compression=true
```

Maximum connections for boostfs mount

You can use the `max-connections` mount option to specify the maximum number of simultaneous open files on the BoostFS mount point. The default value is 128, and the value can be set to any value between 64 and 256.

 **NOTE:** Increasing the number of simultaneous open files increase the amount of memory BoostFS uses.

BoostFS and the Linux mount command

BoostFS allows you to mount a BoostFS file system using the Linux/UNIX `mount` command.

Mounting a BoostFS file system with the `mount` command works the same way mounting NFS or any other file system works. Because the standard `mount` command is supported, other standard facilities that use the `mount` command also work.

 **NOTE:** BoostFS does not support files being executed on the mount point.

Basic use of mount

The most basic use of the `mount` command is as follows:

```
mount -t boostfs myddr:/mystu /mnt
```

In the example, **myddr** is the hostname of the DDR, **mystu** is the name of the DD Boost storage unit, and **/mnt** is the mount point where the file system is to be mounted.

Use of the file systems table

During system start and some other times, the `mount` command consults the file systems table (`fstab`) that is stored in the `/etc/fstab` file for direction on what file systems should be mounted. For example, if the `mount -a` command is executed, `mount` tries to mount all of the file systems that are documented in the `/etc/fstab` file according to the `fstab` rules.

Use of mount with fstab

In this example, the `fstab` entry shown mounts the storage unit **mystu** from the DDR **myddr** onto `/mnt` as a BoostFS filesystem.

```
myddr:/mystu /mnt boostfs defaults,_netdev 0 0
```

When using this command, set the mount point to the location where you want the file system to be mounted, such as `/mnt`.

Allowing multiple users

In this example, the `fstab` entry includes the BoostFS option `allow-others`, which allows access to users other than the user that mounted the file system. Because the file system is mounted during system start, the user that mounted the file system is the root user.

```
myddr:/mystu /mnt boostfs defaults,_netdev,bfsopt(allow-others=true) 0 0
```

Table 4. `mount` command options

Command Option	Description
<code>[-o]</code>	Precedes an option.
<code>[username]=<valid user name></code>	The specified username is used when root is the mounter and the administrator wants the mount to be performed on behalf of the specified user.

Table 4. `mount` command options (continued)

Command Option	Description
<code>[uid]=###</code>	If the username option is not specified, the uid option is used. The specified uid is used when root is the mounter and the administrator wants the mount to be performed on behalf of the specified uid user. If the username option is specified, the uid option is ignored.
<code>[gid]=###</code>	If the uid option is used and the gid option is also specified, the specified gid is used as the effective gid of the mount. If the username option is specified, the gid option is ignored.
<code>[umask]=###</code>	If the uid or the username option is used and the umask option is specified, the specified umask is used as the effective umask of the mount point at the time of the mount. This option is necessary to change the permissions of transient mountpoints as is the case with automounter and other automatic mount mechanisms. It is recommended that the umask be set to "0000" (" <code>umask=0000</code> ").
<code>[bfsopt](<valid boostfs option>,...)</code>	The bfsopt is used to introduce boostfs options on the mount command line. These options are passed directly to boostfs at the time of the mount. Any valid boostfs option may be specified.

Automounter

To mount file systems dynamically, use the Linux automounter with the `autofs` command. Mounts created with the `automount` command are automatically unmounted when not in use.

To enable the automounter, edit the `/etc/auto.master` file. If a program refers to a file within an automount-defined file system, the system mounts the file system to honor the request. The mounting process is transparent to the user and application.

The `auto.master` file introduces the file system to be mounted to automount and refers the automount facility to a script that controls the mount. This file is read when automount is started, usually by an `init.d` or `systemd` script.

In recent versions of Linux, the `systemctl` command is used to perform a service operation such as `systemctl [start | stop | restart] autofs`, where the `start`, `stop`, or `restart` option is specified.

For more information about the automount facility, refer to the Linux man pages for `mount`, `automount`, `auto.master`, and `autofs`.

Using the automounter with BoostFS

In this example, the script to which the `auto.master` file refers the automount facility is `auto.boost`. The `auto.boost` script receives the directory to be mounted as a parameter. The script returns the mount options that are used.

Sample line in `/etc/auto.master` that enables `/etc/auto.boost` to mount to `/boost`:

```
/boost program,sun:/etc/auto.boost --timeout=10
```

A sample `/etc/auto.boost` script file:

```
#!/bin/bash
opts="-fstype=boostfs,rw,noauto,exec,bfsopt(allow-others=true)"
opts2="-fstype=boostfs,rw,umask=0000,username=auser,exec,bfsopt(allow-others=false)"
case "$1" in
    userdir)
        echo "$opts2 myddr:/mystu"
        ;;
    backup)
        echo "$opts myddr:/mystu"
        ;;
    *)
        ;;
esac
```

In this example, the directory `/boost` is automatically created when `automount` is started. When a program or shell command refers to `/boost/userdir`, `automount` creates the directory `/boost/userdir` and mounts the BoostFS file system to that mount point. When the mount operation completes successfully, the user process executes with the files at that mount point. If the mount point remains dormant for more than 10 seconds, it is automatically unmounted.

This example shows an additional mount point, `/boost/backup`, with different options. When using the automounter, you must specify the user for whom the file system is mounted or use the `boostfs` option **allow-others**. The options for the mount point `/boost/backup` show the **allow-others** option.

Because the file `/etc/auto.boost` is an executable script, you must give it `+x` permissions. To test the script, run it with a specified parameter and check the printed response.

BoostFS client connection details

After mount points are created, you can use the `ddboost show connections` command to see details about clients that use BoostFS to connect to the PowerProtect or Data Domain system.

The details displayed in the output include the BoostFS version number and the Boost library, as shown in the following example:

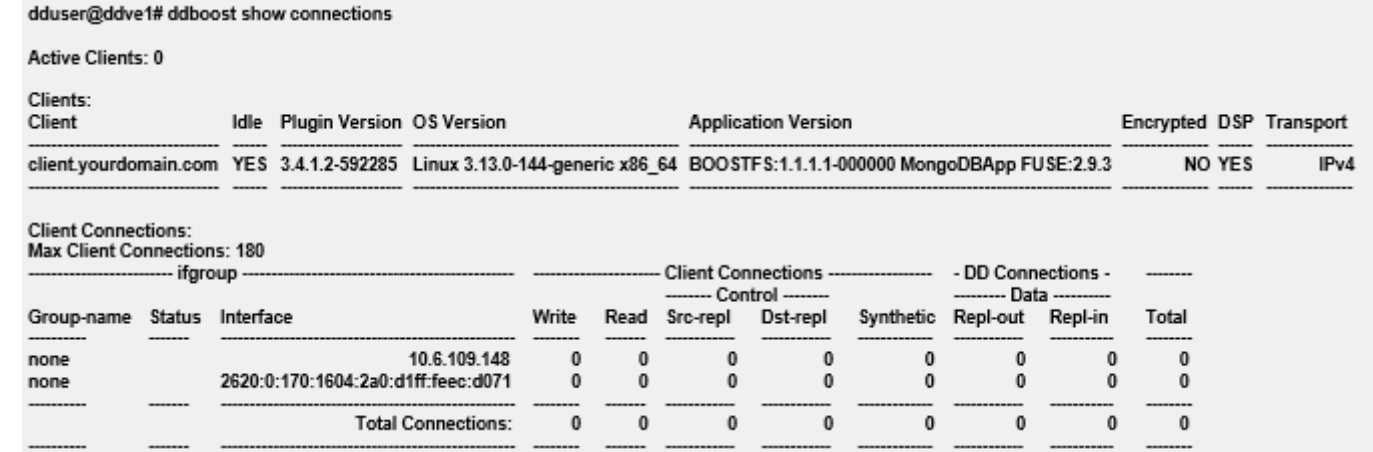


Figure 1. `ddboost show connections` display

See the *DD OS Command Reference Guide* for more information about the `ddboost show connections` command.

Troubleshooting

Topics:

- [Log information](#)
- [Known issues](#)

Log information

You can use the following log files to diagnose BoostFS problems:

- BoostFS log file

By default, the BoostFS log file is found in the directory `/opt/emc/boostfs/log`. The default name of the file is `ddboostfs_<uid>_<gid>.log`, where:

- `<uid>` is the user id of boostfs user
- `<gid>` is the group id of boostfs user

A typical BoostFS log message appears in the following format:

```
Date + Time + Procss-ID + Thread-ID + [logging-level: E - error, W - warning, I - info,
D - debug] + Message-Text
```

The following is an example information message:

```
May 23 12:53:51 2996 4014012160 [I] bfs_close_open_nodsp: File /000000004 opened in non-
DSP mode
```

- DD Boost SDK precert log file
- DD File System logs

DD File System logs are found on the system in the directory `/ddr/var/log/debug`. See the *DD OS Administration Guide* for more information.

BoostFS generates a local log file that contains its internal status, activities, warnings, and errors. You can specify the logging level in addition to the name and location of the log file by using the CLI or the BoostFS configuration file.

If you want to initiate troubleshooting during BoostFS operations, you can use the `kill -s SIGUSR2 <boostfs_pid>` to rotate the BoostFS log level, where `<boostfs_pid>` is the process identifier of the BoostFS process.

You might need to set a size limit on the log file to ensure that when the size of the log file reaches that limit, BoostFS will rotate log files.

You can configure the maximum size of the BoostFS log file in the configuration file. You can also configure the number of older log files you wish to keep.

When the log file size reaches the maximum specified size (in MB), the log file is renamed by appending ".1" to the log file name. If there is already an existing log file that ends in ".1," that file is renamed to replace ".1" with ".2." As each log file reaches the maximum size, log files with numbers (n) appended are renamed .n+1 up to the maximum log rotate number.

Known issues

Unable to establish a BoostFS mount point after upgrade to 1.1

[E] There is an incompatible lockbox version. When you upgrade from BoostFS 1.0 to 1.1, you must create a new lockbox and add user credentials. See [Upgrade the BoostFS client](#) on page 15 for more details.

bfs_get_passp
hrase: unable

```
to get
passphrase -
incompatible
lockbox
version
```

Unable to establish a BoostFS mount point

The following section describes other mount common errors and solutions:

The mount point mount-point is nonempty. BoostFS cannot be mounted on a nonempty mount point. Try mounting BoostFS again on an empty mount point. This error can occur if the user already has a mount point established. You might want to check to see if this might be the case. You can either use the already established mount point or use **boostfs umount** to unmount the existing mount point and establish a new one. This error can also appear if the directory on the client being mounted to already contains files. In this case, these files need to be removed or an empty mount point directory must be selected instead.

Cannot mount mount-point: unexpected error, please see log for details. Most often seen if the DD Boost protocol is not enabled and configured on the PowerProtect or Data Domain System. You should check the BoostFS log files for more details and confirm DD Boost is enabled using the `ddboost status` command.

Invalid mount point option and value pair [option=key from config file] [value= value from config file] /mnt/ test: Configuration initialization failed This message can appear when errors occur during the processing of the BoostFS configuration file. A best practice is to review the specific key and value printed out in the BoostFS configuration file and make any corrections.

Cannot mount mount-point: unexpected error This error is most often seen when using Kerberos authentication and an error exists in the setup. Review the Kerberos instructions in this configuration guide and ensure the values are set properly.

Unable to unmount a BoostFS mountpoint

fusermount: failed to unmount mount-point: Invalid argument This message can appear if the BoostFS mount point has not yet been established. There should be no issues if a mount point has already been unmounted, but this error can still appear.

Unable to access a BoostFS mount point

A permission or privileges error can appear when attempting to use the mount point. This error is most often seen when the user does not have the necessary permissions to access a mount point. By default, the only user allowed to access a mount point is the one that established it. To allow other users to share this mount point, you must include the `-o allow-others=true` option either on the command line or in the BoostFS configuration file.

Configuration values are not taking effect

Typically configuration parameters are not taking effect because the `[global]` label at the top of the BoostFS configuration file has not been uncommented. A best practice is to check the configuration file to confirm `# [global]` has been changed to `[global]`.

BoostFS does not mount after reboot

If BoostFS fails to mount after rebooting the system, you can add the `_netdev` option to `/etc/fstab` as shown in the following example:

```
10.98.88.93:/user1-stu /home/user1/boostfs boostfs umask=0000,user,_netdev 0 0
```

Kerberos

The following section describes Kerberos common errors and solutions:

Insufficient access to or storage-unit storage-unit does not exist

If this error is encountered while mounting BoostFS, use the `boostfs kerberos query` command to confirm that a valid Kerberos ticket exists for the storage-unit user. Use the `boostfs kerberos set` command to reconfigure the expired ticket if necessary.

NOTE: If the BoostFS debug log contains the error message `Server not found in Kerberos database`, confirm that the DNS entries are correct and you can perform a forward and reverse DNS lookup of the server hostname.

Too many open files

If this error is encountered while trying to access BoostFS mount point, confirm that a valid Kerberos ticket exists for the user accessing the BoostFS mount point.

You should also confirm that the Kerberos user name exists in NIS so the Data Domain system is able to map the Kerberos user name.

Appendix

Topics:

- [About Puppet](#)
- [Adding principals to the Unix KDC](#)
- [References](#)

About Puppet

If you have an enterprise/remote environment, you can install and configure Puppet to distribute the BoostFS configuration file – and any future updates – to clients. This is a best practice.

Puppet is open-source software that allows you to manage clients in a primary-server manner. When you install Puppet, you designate one system as the primary. BoostFS uses Puppet to distribute or “push” BoostFS to different machines in a large enterprise environment.

In smaller-scale environments, you can simply install BoostFS on individual machines.

For more information about Puppet, see the Puppet Labs website at <https://puppet.com/product/puppet-enterprise-and-open-source-puppet>.

Adding principals to the Unix KDC

Add DD principals to the UNIX KDC

Steps

1. Log in to the Key Distribution Center (KDC).
2. Enter KDC admin mode using the following command: **kadmin**
The commands in the subsequent steps apply to the KDC after entering **kadmin** mode.
3. Add DD principals to the Key Distribution Center (KDC) using the Kerberos **addprinc** command: **# addprinc boostfs/<ddsystem-hostname>@<realm>**
4. Confirm the client principals have been added by entering the following Kerberos command: **listprincs**
5. Import host and BoostFS credentials to a temporary keytab file on the KDC by entering the Kerberos **ktadd -k** command: **# ktadd -k /tmp/<keytab-file-name-for-ddsystem> boostfs/<ddsystem-hostname>@<realm>**
The keytab file for the PowerProtect or Data Domain system is generated and needs to be imported to the system.
6. Rename the file to **krb5.keytab** and copy it to **/ddr/var** folder.
7. Copy the keytab file generated in Step 3 from the KDC to the PowerProtect or Data Domain system directory **/ddr/var/releases**.

Add client principals to the KDC

Steps

1. Add the host and BoostFS service principals to the KDC using the Kerberos **addprinc** command:

```
# addprinc host/<client-hostname>@<realm>
```

2. Confirm the client principals have been added using the following Kerberos command: **listprincs**

3. Import the host and BoostFS credentials to a temporary keytab file on the KDC by entering the Kerberos `ktadd -k` command:

```
# ktadd -k /tmp/<keytab-file-name-for-client> boostfs/<client-hostname>@<realm>
# ktadd -k /tmp/<keytab-file-name-for-client> host/<client-hostname>@<realm>
```

4. Copy the keytab file generated in Step 3 from the KDC to the client as `/etc/krb5.keytab` file.

References

The following documents, located at [Online Support](#), provide additional and relevant information. Access to these documents depends on your login credentials. If you do not have access to a document, contact a sales representative.

- *Data Domain BoostFS Integration Guide: Application Validation and Best Practices*, available on <https://community.emc.com>
- *DD OS Administration Guide*