Debian User Manual

File Name: Debian User Manual.pdf
Size: 2171 KB
Type: PDF, ePub, eBook
Category: Book
Uploaded: 27 May 2019, 15:22 PM
Rating: 4.6/5 from 834 votes.

Status: AVAILABLE

Last checked: 1 Minutes ago!

In order to read or download Debian User Manual ebook, you need to create a FREE account.

Download Now!

eBook includes PDF, ePub and Kindle version

Register a free 1 month Trial Account.
Download as many books as you like (Personal use)
Cancel the membership at any time if not satisfied.
Join Over 80000 Happy Readers

Book Descriptions:

We have made it easy for you to find a PDF Ebooks without any digging. And by having access to our ebooks online or by storing it on your computer, you have convenient answers with Debian User Manual . To get started finding Debian User Manual , you are right to find our website which has a comprehensive collection of manuals listed.
Our library is the biggest of these that have literally hundreds of thousands of different products represented.

Book Descriptions:

Debian User Manual

Debian Users Manuals Web interface VCS interface git clone Active work is being done for the currentHelp is welcome, especially with nonx86Web interface VCS interface git clone Problems and patches should be reported asWeb interface VCS interface git clone Web interface VCS interface git clone Web interface VCS interface git clone Basic tutorials, tips, andWeb interface VCS interface git clone The document includesSome of its content might not be fully uptodate.Web interface VCS interface git clone Web interface VCS interface git clone Web interface VCS interface git clone Web interface VCS interface git clone Web interface VCS interface git clone Web interface VCS interface git clone. These documents come in these basic categories A FAQ is a documentThese files include detailedYou can findNotice that, for larger programs, documentationSome notable referenceAt least basic knowledge of computer. Debian Developers Manuals Web interface VCS interface git clone Web interface VCS interface git clone Debian users and prospective developers using the debmake command.Web interface VCS interface git clone The author is making everyWeb interface VCS interface git clone It teaches prospectiveIn addition to the mainWeb interface VCS interface git clone However, menu tries to provide a more generalWith the updatemenus command from thisInstead of implementing the required logicWeb interface VCS interface git clone Web interface VCS interface git clone. The overallHTML, and so on to printed manuals covering topics such asHere is the Linux Documentation. Project Manifesto and The exercises and heavilycommentedAlso Linux features, such as installation methods for laptops, notebooks and PDAs as well as configurations for different network environments are described.The reader may choose from MicrosoftGuide.http://cienciarazonyfe.com/assets/assets/userfiles/hyundai-excel-manual-gearbox-oil.xml

- **debian user manual, debian user manual pdf, debian 10 user manual, debian user handbook, linux debian user manual, debian add user manual, 1.0, debian user manual, debian user manual pdf, debian 10 user manual, debian user handbook, linux debian user manual, debian add user manual.**

Everybody who wants to make life easier on themselves, power usersBash is available onFor more advancedThis book contains many real life examples derived from the authorsThis document is not a comprehensive list of everyIt is a summary which can be used to learnIt includes information about the tools,The documents primary audience is new TLDPWhereas some ofEach chapter explores a smallVolume Management System EVMS and provide a context for using EVMS. For additional information about EVMS or to ask questions specificYou can view the listIt is meant to bePowerPCbased board. This means makingBeginners and experiencedThe possible range of topics to cover isWeve tried to cover theWeve found that beginners to LinuxMany Linux users wont have accessIt assumes no previous UnixThis manual wont coverThis book is a guideAdditional translations of LDP works and translatedIt takes youThe main focus of TrinityOS is to do this. See the GNU General Public License for more details.If not, see.That said maintaining it takes time and lots of effort, and we appreciate being thanked for this. For example, the Backup section relies on setup that is performed in the Quick Start section. Once pgBackRest is up and running then skipping around is possible but following the user guide in order is recommended the first time through. The only OSspecific commands are those to create, start, stop, and drop PostgreSQL clusters. The pgBackRest commands will be the same on any Unix system though the location to install the executable may vary. Configuration information and documentation for PostgreSQL can be found in the PostgreSQL Manual. A somewhat novel approach is taken to documentation in this user guide.

Each command is run on a virtual machine when the documentation is built from the XML source. This means you can have a high confidence that the commands work correctly in the order presented. Output is captured and displayed below the command when appropriate.http://www.cargoaircon.ru/userfiles/hyundai-excel-manual-transmission-fluid-change.xml

If the output is not included it is because it was deemed not relevant or was considered a distraction from the narrative. All commands are intended to be run as an unprivileged user that has sudo privileges for both the root and postgres users. Its also possible to run the commands directly as their respective users without modification and in that case the sudo commands can be stripped off. 2 Concepts The following concepts are defined as they are relevant to pgBackRest, PostgreSQL, and this user guide. 2.1 Backup A backup is a consistent copy of a database cluster that can be restored to recover from a hardware failure, to perform PointInTime Recovery, or to bring up a new standby. Full Backup pgBackRest copies the entire contents of the database cluster to the backup. The first backup of the database cluster is always a Full Backup.The full backup does not depend on any files outside of the full backup for consistency. Differential Backup pgBackRest copies only those database cluster files that have changed since the last full backup.The advantage of a differential backup is that it requires less disk space than a full backup, however, the differential backup and the full backup must both be valid to restore the differential backup. Incremental Backup pgBackRest copies only those database cluster files that have changed since the last backup which can be another incremental backup, a differential backup, or a full backup. As an incremental backup only includes those files changed since the prior backup, they are generally much smaller than full or differential backups. As with the differential backup, the incremental backup depends on other backups to be valid to restore the incremental backup. Since the incremental backup includes only those files since the last backup, all prior incremental backups back to the prior differential, the prior differential backup, and the prior full backup must all be valid to perform a restore of the incremental backup.

If no differential backup exists then all prior incremental backups back to the prior full backup, which must exist, and the full backup itself must be valid to restore the incremental backup. 2.2 Restore A restore is the act of copying a backup to a system where it will be started as a live database cluster. A restore requires the backup files and one or more WAL segments in order to work correctly. 2.3 Write Ahead Log WAL WAL is the mechanism that PostgreSQL uses to ensure that no committed changes are lost. Transactions are written sequentially to the WAL and a transaction is considered to be committed when those writes are flushed to disk. Afterwards, a background process writes the changes into the main database cluster files also known as the heap. In the event of a crash, the WAL is replayed to make the database consistent. WAL is conceptually infinite but in practice is broken up into individual 16MB files called segments. WAL segments follow the naming convention 0000000100000A1E000000FE where the first 8 hexadecimal digits represent the timeline and the next 16 digits are the logical sequence number LSN. 2.4 Encryption Encryption is the process of converting data into a format that is unrecognizable unless the appropriate password also referred to as passphrase is provided.The repository format has not changed and all nondeprecated options from v1 are accepted, so for most installations it is simply a matter of installing the new version. However, there are a few caveats The deprecated threadmax option is no longer valid. Use processmax instead. The deprecated archivemaxmb option is no longer valid. This has been replaced with the archivepushqueuemax option which has different semantics. Many option names have changed to improve consistency although the old names from v1 are still accepted. PostgreSQL and repository options must be indexed when using the new names introduced in v2, e.g. pg1host, pg1path, repo1path, repo1type, etc.

http://www.jfvtransports.com/home/content/ford-explorer-xlt-2003-owners-manual

When building from source it is best to use a build host rather than building on production. Many of

the tools required for the build should generally not be installed in production.If any dependencies were missed then you will get an error when running pgBackRest from the command line. Usage. CommandsVer Cluster Port Status Owner Data directory Log file. This may not be appropriate for secure installations. This keeps the log output as brief as possible to better illustrate important information. Most db servers will only have one Postgres database cluster and therefore one stanza, whereas backup servers will have a stanza for every database cluster that needs to be backed up. It is tempting to name the stanza after the primary cluster but a better name describes the databases contained in the cluster. Because the stanza name will be used for the primary and all replicas it is more appropriate to choose a name that describes the actual function of the cluster, such as app or dw, rather than the local cluster name, such as main or prod. The name demo describes the purpose of this cluster accurately so that will also make a good stanza name.The path can be requested from PostgreSQL directly but in a recovery scenario the PostgreSQL process will not be available. During backups the value supplied to pgBackRest will be compared against the path that PostgreSQL is running on and they must be equal or the backup will return an error.The default config file will be be loaded if it exists. If it is desirable to load only the files in the specified config include path, then the noconfig option can also be passed. The files are expected to exist. The files are concatenated as if they were one big file; order doesnt matter, but there is precedence based on sections.Path where log files are stored. The log path provides a location for pgBackRest to store log files. Note that. It may be difficult to estimate in advance how much space youll need.

This will give you a general idea of how much space youll need, though of course requirements will likely change over time as your database evolves. For this demonstration the repository will be stored on the same host as the PostgreSQL server. This is the simplest configuration and is useful in cases where traditional backup software is employed to backup the database host. Note that at least one WAL segment will be created during the backup process even if no explicit writes are made to the cluster. The PostgreSQL cluster must be restarted after making these changes and before performing a backup. The archivepush command can be configured with its own options. For example, a lower compression level may be set to speed archiving without affecting the compression used for backups. Encryption is always performed clientside even if the repository type e.g. S3 or other object store supports encryption. It is important to use a long, random passphrase for the cipher key. A good way to generate one is to run openssl rand base64 48. It is recommended that the check command be run after stanzacreate to ensure archiving and backups are properly configured. It detects misconfigurations, particularly in archiving, that result in incomplete backups because required WAL segments did not reach the archive. The command can be run on the database or the repository host. However, an incremental backup must be based on a full backup and since no full backup existed pgBackRest ran a full backup instead. The type option can be used to specify a full or differential backup. While incremental backups can be based on a full or differential backup, differential backups must be based on a full backup. More information about the backup command can be found in the Backup section. 6.10 Schedule a Backup Backups can be scheduled with utilities such as cron.

In the following example, two cron jobs are configured to run; full backups are scheduled for 630 AM every Sunday with differential backups scheduled for 630 AM Monday through Saturday.The stanza status gives a brief indication of the stanzas health. If this is ok then pgBackRest is functioning normally. Note that there may be gaps due to archive retention policies or other reasons. The backups are displayed oldest to newest. The oldest backup will always be a full backup indicated by an F at the end of the label but the newest backup can be full, differential ends with D , or incremental ends with I . The timestamp stop can be used to determine the backup to use when

performing PointInTime Recovery. More information about PointInTime Recovery can be found in the PointInTime Recovery section. The backup command will ensure that this WAL range is in the archive before completing. The database size is the full uncompressed size of the database while backup size is the amount of data actually backed up these will be the same for full backups. The repository size includes all the files from this backup and any referenced backups that are required to restore the database while repository backup size includes only the files in this backup these will also be the same for full backups. Repository sizes reflect compressed file sizes if compression is enabled in pgBackRest or the filesystem. The backup reference list contains the additional backups that are required to restore this backup. 6.12 Restore a Backup Backups can protect you from a number of disaster scenarios, the most common of which are hardware failure and data corruption. The easiest way to simulate data corruption is to remove an important PostgreSQL cluster file. Examine the log output. To restore a backup of the PostgreSQL cluster run pgBackRest with the restore command.

The cluster needs to be stopped in this case it is already stopped and all files must be removed from the PostgreSQL data directory. More information about the restore command can be found in the Restore section. 7 Backup The Backup section introduces additional backup command features. 7.1 Fast Start Option By default pgBackRest will wait for the next regularly scheduled checkpoint before starting a backup. This is convenient for testing and for adhoc backups. For instance, if a backup is being taken at the beginning of a release window it makes no sense to wait for a checkpoint. Alternately, it is possible to override the setting in the configuration file by passing nostartfast on the commandline. This wait time is governed by the pgBackRest archivetimeout option which defaults to 60 seconds. If archiving an individual segment is known to take longer then this option should be increased. 8 Monitoring Monitoring is an important part of any production system. There are many tools available and pgBackRest can be monitored on any of them with a little work.The following example wraps that logic in a function that can be used to perform realtime queries.Test your queries carefully. 9 Retention Generally it is best to retain as many backups as possible to provide a greater window for PointinTime Recovery, but practical concerns such as disk space must also be considered. Retention options remove older backups once they are no longer needed. 9.1 Full Backup Retention The repo1retentionfulltype determines how the option repo1retentionfull is interpreted; either as the count of full backups to be retained or how many days to retain full backups. P00 INFO backup command end completed successfully. These are not useful for recovery — only WAL segments generated after a backup can be used to recover that backup. Differentials only rely on the prior full backup so it is possible to create a rolling set of differentials for the last day or more.

This allows quick restores to recent pointsintime but reduces overall space consumption. An incremental backup is added to demonstrate incremental expiration. Incremental backups cannot be expired independently — they are always expired with their related full or differential backup. Note that full backups are treated as differential backups for the purpose of differential archive retention. Expiring archive will never remove WAL segments that are required to make a backup consistent. However, since PointinTimeRecovery PITR only works on a continuous WAL stream, care should be taken when aggressively expiring archive outside of the normal backup expiration process. To determine what will be expired without actually expiring anything, the dryrun option can be provided on the command line with the expire command. In that case the file ownership will need to be updated by a privileged user before the restore can be retried. If a restore is run as the root user then pgBackRest will attempt to recreate the ownership recorded in the manifest when the backup was made. The delta option allows pgBackRest to automatically determine which files in the database cluster directory can be preserved and which ones need to be restored from the backup — it also removes files not present in the backup manifest so it will dispose of divergent changes. This is accomplished by calculating a SHA1 cryptographic hash for each file in the database cluster

directory. If the SHA1 hash does not match the hash stored in the backup then that file will be restored. This operation is very efficient when combined with the processmax option. Since the PostgreSQL server is shut down during the restore, a larger number of processes can be used than might be desirable during a backup when the PostgreSQL server is running. This could be done for performance reasons or to move selected databases to a machine that does not have enough space to restore the entire cluster backup.

To demonstrate this feature two databases are created test1 and test2. A fresh backup is run so pgBackRest is aware of the new databases. The size of the test1 database is shown here so it can be compared with the disk utilization after a selective restore.Builtin databases template0, template1, and postgres are always restored.This is because the entire database was restored as sparse, zeroed files. PostgreSQL can successfully apply WAL on the zeroed files but the database as a whole will not be valid because key files contain no data. This is purposeful to prevent the database from being accidentally used when it might contain partial data that was applied during WAL replay. While the amount of WAL generated during a backup and applied during recovery can be significant it will generally be a small fraction of the total database size, especially for large databases where this feature is most likely to be useful. It is clear that the test1 database uses far less disk space during the selective restore than it would have if the entire database had been restored.In the case of a hardware failure this is usually the best choice but for data corruption scenarios whether machine or human in origin PointinTime Recovery PITR is often more appropriate. PointinTime Recovery PITR allows the WAL to be played from the last backup to a specified time, transaction id, or recovery point. For common recovery scenarios timebased recovery is arguably the most useful. A typical recovery scenario is to restore a table that was accidentally dropped or data that was accidentally deleted. Recovering a dropped table is more dramatic so thats the example given here but deleted data would be recovered in exactly the same way.This reduces the possibility of unintended timezone conversions and an unexpected recovery result. In practice finding the exact time that the table was dropped is a lot harder than in this example.

It may not be possible to find the exact time, but some forensic work should be able to get you close.Once PostgreSQL has finished recovery the table will exist again and can be queried.It will indicate the time and transaction where the recovery stopped and also give the time of the last transaction to be applied. LOG database system is ready to accept read only connections. LOG archive recovery complete. If a backup after the required time is chosen then PostgreSQL will not be able to recover the lost table. PostgreSQL can only play forward, not backward. To demonstrate this the important table must be dropped again.The info command can be used to find the new backup label.The key is to look for the presence of the recovery stopping before.If a backup set cannot be found, then restore will default to the latest backup which, as shown earlier, may not give the desired result.LOG database system is ready to accept read only connections. The container used to store the repository must be created in advance — pgBackRest will not do it automatically. Commands are run exactly as if the repository were stored on a local disk. The bucket used to store the repository must be created in advance — pgBackRest will not do it automatically. The values given here are for the useast1 region. A role should be created to run pgBackRest and the bucket permissions should be set as restrictively as possible. WARNING Use this command with caution — it will permanently remove all backups and archives from the pgBackRest repository for the specified stanza. To delete a stanza Shut down the PostgreSQL cluster associated with the stanza or use force to override. Run the stop command on the repository host. Run the stanzadelete command on the repository host. Once the command successfully completes, it is the responsibility of the user to remove the stanza from all pgBackRest configuration files.

This separates the backups and WAL archive from the database server so database host failures have less impact. It is still a good idea to employ traditional backup software to backup the

repository host. On PostgreSQL hosts, pg1path is required to be the path of the local PostgreSQL cluster and no pg1host should be configured. When configuring a repository host, the pgbackrest configuration file must have the pghost option configured to connect to the primary and standby if any hosts. The repository host has the only pgbackrest configuration that should be aware of more than one PostgreSQL host. The pgbackrest user is created to own the pgBackRest repository. Any user can own the repository but it is best not to use postgres if it exists to avoid confusion. The primary will be configured as pg1 to allow a standby to be added later. The default for the repo1hostuser option is pgbackrest. If the postgres user does restores on the repository host it is best not to also allow the postgres user to perform backups. However, the postgres user can read the repository directly if it is in the same group as the pgbackrest user. Create the stanza in the new repository. More information about the check command can be found in Check the Configuration. The number of processes to be used for this feature is set using the processmax option. It is usually best not to use more than 25% of available CPUs for the backup command. Backups dont have to run that fast as long as they are performed regularly and the backup process should not impact database performance, if at all possible. The restore command can and should use all available CPUs because during a restore the PostgreSQL cluster is shut down and there is generally no other important work being done on the host. If the host contains multiple clusters then that should be considered when setting restore parallelism.

For very small backups the difference may not be very apparent, but as the size of the database increases so will time savings. 17 Starting and Stopping Sometimes it is useful to prevent pgBackRest from running on a system. For example, when failing over from a primary to a standby its best to prevent pgBackRest from running on the old primary in case PostgreSQL gets restarted or cant be completely killed. This will also prevent pgBackRest from running on cron. If pgBackRest is already stopped then stopping again will generate a warning. The standbys are useful for balancing reads and to provide redundancy in case the primary host fails. 18.1 Installation A new host named pgstandby is created to run the standby.If archiving is not disabled then the repository may be polluted with WAL that can make restores more difficult. Otherwise, connection attempts will be refused. The rest of the configuration is in case the standby is promoted to a primary. Note especially that the cluster has entered standby mode and is ready to accept readonly connections.Since PostgreSQL is pulling WAL segments from the archive to perform replication, changes wont be seen on the standby until the WAL segment that contains those changes is pushed from pgprimary.This results in much less lag between the primary and standby. Streaming replication requires a user with the replication privilege. Be sure to replace the IP address below with the actual IP address of your pgprimary. LOG database system is ready to accept read only connectionsThis option enables asynchronous operation for both the archivepush and archiveget commands. A spool path is required. The commands will store transient data here but each command works quite a bit differently so spool path usage is described in detail in each section.

Asynchronous archiving automatically confers some benefit by reducing the number of connections made to remote storage, but setting processmax can drastically improve performance by parallelizing operations. Be sure not to set processmax so high that it affects normal database operations. This also allows different values for archivepush and archiveget. The spool path holds the current status of WAL archiving. Status files written into the spool directory are typically zero length and should consume a minimal amount of space a few MB at most and very little IO. All the information in this directory can be recreated so it is not necessary to preserve the spool directory if the cluster is moved to new hardware. IMPORTANT In the original implementation of asynchronous archiving, WAL segments were copied to the spool directory before compression and transfer. If asynchronous archiving was utilized in v1.12 or prior, read the v1.13 release notes carefully before upgrading. A good way to test this is to quickly push a number of WAL segments. P01 DETAIL pushed WAL file 000000080000000000000021 to the archive P00 INFO archivepushasync command

end completed successfullyP02 DETAIL pushed WAL file 000000080000000000000023 to the archive. P01 DETAIL pushed WAL file 000000080000000000000024 to the archive. P02 DETAIL pushed WAL file 000000080000000000000025 to the archive P00 INFO archivepushasync command end completed successfullyIf a WAL segment is not found in the queue it is fetched from the repository along with enough consecutive WAL to fill the queue. The maximum size of the queue is defined by archivegetqueuemax. Whenever the queue is less than half full more WAL will be fetched to fill it. Asynchronous operation is most useful in environments that generate a lot of WAL or have a high latency connection to the repository storage i.e., S3 or other object stores. In the case of a high latency connection it may be a good idea to increase processmax.

P00 INFO get 8 WAL files from archive 000000080000000000000001F.000000080000000000000026 P01 DETAIL found 000000080000000000000001F in the archive. P02 DETAIL found 000000080000000000000020 in the archive P01 DETAIL unable to find 000000080000000000000021 in the archive. P02 DETAIL unable to find 000000080000000000000022 in the archive. P00 INFO get 8 WAL files from archive 000000080000000000000021.000000080000000000000028 P02 DETAIL found 000000080000000000000022 in the archive. P01 DETAIL found 000000080000000000000021 in the archive. P02 DETAIL found 000000080000000000000023 in the archive. P01 DETAIL found 000000080000000000000024 in the archive. P01 DETAIL found 000000080000000000000025 in the archive. P02 DETAIL found 000000080000000000000026 in the archive P01 DETAIL unable to find 000000080000000000000027 in the archive. P02 DETAIL unable to find 000000080000000000000028 in the archive. Standby backups require the pgstandby host to be configured and the backupstandby option enabled. If more than one standby is configured then the first running standby found will be used for the backup. The database hosts can be configured in any order.This means that logs and statistics from the primary database will be included in the backup. 21 Upgrading PostgreSQL Immediately after upgrading PostgreSQL to a newer major version, the pgpath for all pgBackRest configurations must be set to the new database location and the stanzaupgrade run on the repository host. If the database is offline use the noonline option. The following instructions are not meant to be a comprehensive guide for upgrading PostgreSQL, rather they outline the general process for upgrading a primary and standby with the intent of demonstrating the steps required to reconfigure pgBackRest. It is recommended that a backup be taken prior to upgrading. Creating script to delete old cluster okThe warning regarding the standby being down is expected since the standby cluster is down.