

pgmoneta

User Guide

Contents

1	Introduction	7
1.1	Features	7
1.2	Platforms	8
2	Installation	9
2.1	Fedora	9
2.2	RHEL 9 / RockyLinux 9	9
2.3	Compiling the source	9
2.3.1	RHEL / RockyLinux	10
2.3.2	FreeBSD	11
2.3.3	Build	11
2.4	Compiling the documentation	12
2.4.1	Build	12
2.5	Extension installation	12
2.5.1	Install pgmoneta_ext	13
2.5.2	Verify success	13
2.5.3	Granting SUPERUSER Privileges	13
3	Quick start	15
3.1	Configuration	15
3.2	Running	17
3.3	Run-time administration	17
3.4	Administration	18
3.5	Next Steps	19
4	Configuration	20
4.1	pgmoneta	20
4.2	Server section	31
4.2.1	extra parameter	33
4.3	pgmoneta_users configuration	34
4.4	pgmoneta_admins configuration	34
5	Tutorials	35
5.1	Install pgmoneta	35
5.1.1	Preface	35
5.1.2	Initialize cluster	35
5.1.3	Remove default access	35

5.1.4	Add access for users and a database	36
5.1.5	Make sure that replication level is set	36
5.1.6	Start PostgreSQL	36
5.1.7	Add users and a database	36
5.1.8	Add Write-Ahead Log (WAL) replication slot	37
5.1.9	Verify access	37
5.1.10	Add pgmoneta user	37
5.1.11	Create pgmoneta configuration	37
5.1.12	Create base directory	38
5.1.13	Start pgmoneta	39
5.1.14	Create a backup	39
5.1.15	View backup	39
5.1.16	Shell completion	39
5.2	Remote administration for pgmoneta	40
5.2.1	Preface	40
5.2.2	Change the pgmoneta configuration	40
5.2.3	Add pgmoneta admin	41
5.2.4	Restart pgmoneta	41
5.2.5	Connect via remote administration interface	41
5.2.6	Using Transport Level Security for access	41
5.3	Prometheus metrics for pgmoneta	42
5.3.1	Preface	42
5.3.2	Change the pgmoneta configuration	42
5.3.3	Restart pgmoneta	42
5.3.4	Get Prometheus metrics	42
5.4	Backup and restore	43
5.4.1	Preface	43
5.4.2	Backup	43
5.4.3	List backups	43
5.4.4	Restore	43
5.5	Verify	43
5.5.1	Preface	44
5.5.2	Verify	44
5.6	Archive	44
5.6.1	Preface	44
5.6.2	Creating an archive	44
5.7	Delete a backup	44
5.7.1	Preface	45

5.7.2	Delete the oldest backup	45
5.8	Encryption and Decryption	45
5.8.1	Preface	45
5.8.2	Enable Encryption and Decryption in pgmoneta workflow	45
5.8.3	Encryption and Decryption Commands	45
5.9	Retention Policy	46
5.9.1	Preface	46
5.9.2	Retention Setup	46
5.9.3	Retention Validation Rule	47
5.10	Retention check	47
5.11	Grafana Dashboard	47
5.11.1	Preface	47
5.11.2	Prometheus Configuration	47
5.11.3	Grafana Dashboard Import	49
5.12	WAL shipping	50
5.12.1	Preface	50
5.12.2	Configuration	50
5.12.3	Prometheus	51
5.13	Use of Transport Level Security (TLS)	51
5.13.1	Preface	51
5.13.2	PostgreSQL	51
5.13.3	Using client certificate	52
5.13.4	More information	53
5.14	Hot standby	53
5.14.1	Preface	53
5.14.2	Configuration	54
5.14.3	Tablespaces	54
5.15	Annotate a backup with comments	54
5.15.1	Preface	54
5.15.2	Add a comment	54
5.15.3	Update a comment	55
5.15.4	Remove a comment	55
5.15.5	View comments	55
5.16	Feature extra	55
5.16.1	Preface	55
5.16.2	Configuration	55
5.16.3	Info	56

6	Command line interface	57
6.1	backup	58
6.2	list-backup	58
6.3	restore	58
6.4	verify	59
6.5	archive	59
6.6	delete	60
6.7	retain	60
6.8	expunge	60
6.9	encrypt	60
6.10	decrypt	61
6.11	compress	61
6.12	decompress	61
6.13	info	61
6.14	ping	61
6.15	shutdown	62
6.16	status	62
6.17	conf	62
6.18	clear	62
6.19	Shell completions	63
7	Prometheus metrics	64
7.1	pgmoneta_state	64
7.2	pgmoneta_version	64
7.3	pgmoneta_logging_info	64
7.4	pgmoneta_logging_warn	64
7.5	pgmoneta_logging_error	64
7.6	pgmoneta_logging_fatal	64
7.7	pgmoneta_retention_days	64
7.8	pgmoneta_retention_weeks	64
7.9	pgmoneta_retention_months	65
7.10	pgmoneta_retention_years	65
7.11	pgmoneta_retention_server	65
7.12	pgmoneta_extension	65
7.13	pgmoneta_compression	65
7.14	pgmoneta_used_space	65
7.15	pgmoneta_free_space	66
7.16	pgmoneta_total_space	66

7.17	pgmoneta_server_valid	66
7.18	pgmoneta_wal_streaming	66
7.19	pgmoneta_server_operation_count	66
7.20	pgmoneta_server_failed_operation_count	66
7.21	pgmoneta_server_last_operation_time	66
7.22	pgmoneta_server_last_failed_operation_time	66
7.23	pgmoneta_wal_shipping	66
7.24	pgmoneta_wal_shipping_used_space	67
7.25	pgmoneta_wal_shipping_free_space	67
7.26	pgmoneta_wal_shipping_total_space	67
7.27	pgmoneta_hot_standby	67
7.28	pgmoneta_hot_standby_free_space	67
7.29	pgmoneta_hot_standby_total_space	67
7.30	pgmoneta_server_timeline	67
7.31	pgmoneta_server_parent_tli	67
7.32	pgmoneta_server_timeline_switchpos	68
7.33	pgmoneta_server_workers	68
7.34	pgmoneta_backup_oldest	68
7.35	pgmoneta_backup_newest	68
7.36	pgmoneta_backup_count	69
7.37	pgmoneta_backup	69
7.38	pgmoneta_backup_version	69
7.39	pgmoneta_backup_throughput	70
7.40	pgmoneta_backup_elapsed_time	70
7.41	pgmoneta_backup_start_timeline	70
7.42	pgmoneta_backup_end_timeline	70
7.43	pgmoneta_backup_start_walpos	71
7.44	pgmoneta_backup_checkpoint_walpos	71
7.45	pgmoneta_backup_end_walpos	71
7.46	pgmoneta_restore_newest_size	71
7.47	pgmoneta_backup_newest_size	72
7.48	pgmoneta_restore_size	72
7.49	pgmoneta_restore_size_increment	72
7.50	pgmoneta_backup_size	72
7.51	pgmoneta_backup_compression_ratio	73
7.52	pgmoneta_backup_retain	73
7.53	pgmoneta_backup_total_size	73
7.54	pgmoneta_wal_total_size	73

7.55	pgmoneta_total_size	74
7.56	pgmoneta_active_backup	74
7.57	pgmoneta_current_wal_file	74
7.58	pgmoneta_current_wal_lsn	74
8	SSH	76
8.1	Prerequisites	76
8.2	Modify the pgmoneta configuration	77
9	Azure	78
9.1	Prerequisites	78
9.2	Modify the pgmoneta configuration	79
10	S3	80
10.1	Prerequisites	80
10.2	Modify the pgmoneta configuration	80
11	Troubleshooting	82
11.1	Could not get version for server	82
12	Acknowledgement	83
12.1	Authors	83
12.2	Committers	83
12.3	Contributing	83
13	License	85
13.1	libart	85

1 Introduction

pgmoneta is a backup / restore solution for PostgreSQL.

Ideally, you would not need to do backups and disaster recovery, but that isn't how the real World works.

Possible scenarios that could happen

- Data corruption
- System failure
- Human error
- Natural disaster

and then it is up to the database administrator to get the database system back on-line, and to the correct recovery point.

Two key factors are

- Recovery Point Objective (RPO): Maximum targeted period in which data might be lost from an IT service due to a major incident
- Recovery Time Objective (RTO): The targeted duration of time and a service level within which a business process must be restored after a disaster (or disruption) in order to avoid unacceptable consequences associated with a break in business continuity

You would like to have both of these as close to zero as possible, since RPO of 0 means that you won't lose data, and RTO of 0 means that your system recovers at once. However, that is easier said than done.

pgmoneta is focused on having features that will allow database systems to get as close to these goals as possible such that high availability of 99.99% or more can be implemented, and monitored through standard tools.

pgmoneta is named after the Roman Goddess of Memory.

1.1 Features

- Full backup
- Restore
- Compression (gzip, zstd, lz4, bzip2)
- AES encryption support
- Symlink support
- WAL shipping support

- Hot standby
- Prometheus support
- Remote management
- Offline mode
- Transport Layer Security (TLS) v1.2+ support
- Daemon mode
- User vault

1.2 Platforms

The supported platforms are

- Fedora 38+
- RHEL 9
- RockyLinux 9
- FreeBSD
- OpenBSD

2 Installation

2.1 Fedora

You need to add the PostgreSQL YUM repository, for example for Fedora 40

```
dnf install -y https://download.postgresql.org/pub/repos/yum/reporpms/F-40-x86_64/pgdg-fedora-repo-latest.noarch.rpm
```

and do the install via

```
dnf install -y pgmoneta
```

Additional information

- PostgreSQL YUM
- Linux downloads

2.2 RHEL 9 / RockyLinux 9

x86_64

```
dnf install -y https://dl.fedoraproject.org/pub/epel/epel-release-latest-9.noarch.rpm
dnf install -y https://download.postgresql.org/pub/repos/yum/reporpms/EL-9-x86_64/pgdg-redhat-repo-latest.noarch.rpm
```

aarch64

```
dnf install -y https://dl.fedoraproject.org/pub/epel/epel-release-latest-9.noarch.rpm
dnf install -y https://download.postgresql.org/pub/repos/yum/reporpms/EL-9-aarch64/pgdg-redhat-repo-latest.noarch.rpm
```

and do the install via

```
dnf install -y pgmoneta
```

2.3 Compiling the source

We recommend using Fedora to test and run **pgmoneta**, but other Linux systems, FreeBSD and MacOS are also supported.

pgmoneta requires

- clang
- cmake
- make
- libev
- OpenSSL
- zlib
- zstd
- lz4
- bzip2
- systemd
- rst2man
- libssh
- libcurl
- libarchive

```
dnf install git gcc clang clang-analyzer cmake make libev libev-devel \
            openssl openssl-devel \
            systemd systemd-devel zlib zlib-devel \
            libzstd libzstd-devel \
            lz4 lz4-devel libssh libssh-devel \
            libcurl libcurl-devel \
            python3-docutils libatomic \
            bzip2 bzip2-devel \
            libarchive libarchive-devel
```

Alternative gcc can be used.

2.3.1 RHEL / RockyLinux

On RHEL / Rocky, before you install the required packages some additional repositories need to be enabled or installed first.

First you need to install the subscription-manager

```
dnf install subscription-manager
```

It is ok to disregard the registration and subscription warning.

Otherwise, if you have a Red Hat corporate account (you need to specify the company/organization name in your account), you can register using

```
subscription-manager register --username <your-account-email-or-login> --
    password <your-password> --auto-attach
```

Then install the EPEL repository,

```
dnf install epel-release
```

Then to enable powertools

```
dnf config-manager --set-enabled codeready-builder-for-rhel-9-rhui-rpms
dnf config-manager --set-enabled crb
dnf install https://dl.fedoraproject.org/pub/epel/epel-release-latest-9.
noarch.rpm
```

Then use the `dnf` command for **pgmoneta** to install the required packages.

2.3.2 FreeBSD

On FreeBSD, `pkg` is used instead of `dnf` or `yum`.

Use `pkg install <package name>` to install the following packages

```
git gcc cmake libev openssl libssh zlib-ng zstd liblz4 bzip2 curl \
py39-docutils libarchive
```

2.3.3 Build

2.3.3.1 Release build The following commands will install **pgmoneta** in the `/usr/local` hierarchy.

```
git clone https://github.com/pgmoneta/pgmoneta.git
cd pgmoneta
mkdir build
cd build
cmake -DCMAKE_INSTALL_PREFIX=/usr/local ..
make
sudo make install
```

See RPM for how to build a RPM of **pgmoneta**.

2.3.3.2 Debug build The following commands will create a `DEBUG` version of **pgmoneta**.

```
git clone https://github.com/pgmoneta/pgmoneta.git
cd pgmoneta
mkdir build
cd build
cmake -DCMAKE_C_COMPILER=clang -DCMAKE_BUILD_TYPE=Debug ..
make
```

2.4 Compiling the documentation

pgmoneta's documentation requires

- pandoc
- texlive

```
dnf install pandoc texlive-scheme-basic \
    'tex(footnote.sty)' 'tex(footnotebackref.sty)' \
    'tex(pagecolor.sty)' 'tex(hardwrap.sty)' \
    'tex(mdframed.sty)' 'tex(sourcesanspro.sty)' \
    'tex(lylenc.def)' 'tex(sourcecodepro.sty)' \
    'tex(titling.sty)' 'tex(csquotes.sty)' \
    'tex(zref-abspace.sty)' 'tex(needspace.sty)'
```

You will need the [Eisvogel](#) template as well which you can install through

```
wget https://github.com/Wandmalfarbe/pandoc-latex-template/releases/
download/2.4.2/Eisvogel-2.4.2.tar.gz
tar -xzf Eisvogel-2.4.2.tar.gz
mkdir -p $HOME/.local/share/pandoc/templates
mv eisvogel.latex $HOME/.local/share/pandoc/templates
```

where `$HOME` is your home directory.

2.4.0.1 Generate API guide This process is optional. If you choose not to generate the API HTML files, you can opt out of downloading these dependencies, and the process will automatically skip the generation.

Download dependencies

```
dnf install graphviz doxygen
```

2.4.1 Build

These packages will be detected during `cmake` and built as part of the main build.

2.5 Extension installation

When you configure the `extra` parameter in the server section of `pgmoneta.conf`, it requires the server side to have the `pgmoneta_ext` extension installed to make it work.

The following instructions can help you easily install `pgmoneta_ext`. If you encounter any problems, please refer to the more detailed instructions in the DEVELOPERS documentation.

2.5.1 Install pgmoneta_ext

After you have successfully installed `pgmoneta`, the following commands will help you install `pgmoneta_ext`:

```
dnf install -y pgmoneta_ext
```

You need to add the `pgmoneta_ext` library for PostgreSQL in `postgresql.conf` as well:

```
shared_preload_libraries = 'pgmoneta_ext'
```

And remember to restart PostgreSQL to make it work.

2.5.2 Verify success

You can use the `postgres` role to test.

1. Log into PostgreSQL

```
psql
```

2. Create a new test database

```
CREATE DATABASE testdb;
```

3. Enter the database

```
\c testdb
```

4. Follow the SQL commands below to check the function

```
DROP EXTENSION IF EXISTS pgmoneta_ext;  
CREATE EXTENSION pgmoneta_ext;  
SELECT pgmoneta_ext_version();
```

You should see

```
pgmoneta_ext_version  
-----  
0.1.0  
(1 row)
```

2.5.3 Granting SUPERUSER Privileges

Some functions in `pgmoneta_ext` require `SUPERUSER` privileges. To enable these, grant the `repl` role superuser privileges using the command below. **Please proceed with caution:** granting superuser

privileges bypasses all permission checks, allowing unrestricted access to the database, which can pose security risks. We are committed to enhancing privilege security in future updates.

```
ALTER ROLE repl WITH SUPERUSER;
```

To revoke superuser privileges from the `repl` role, use the following command:

```
ALTER ROLE repl WITH NOSUPERUSER;
```

3 Quick start

Make sure that **pgmoneta** is installed and in your path by using `pgmoneta -?`. You should see

```
pgmoneta 0.12.0
Backup / restore solution for PostgreSQL

Usage:
pgmoneta [ -c CONFIG_FILE ] [ -u USERS_FILE ] [ -d ]

Options:
-c, --config CONFIG_FILE      Set the path to the
                               pgmoneta.conf file
-h, --host HOST               Set the host name
-p, --port PORT               Set the port number
-U, --user USERNAME           Set the user name
-P, --password PASSWORD       Set the password
-L, --logfile FILE            Set the log file
-v, --verbose                  Output text string of
                               result
-V, --version                  Display version
                               information
-F, --format text|json|raw     Set the output format
-C, --compress none|gz|zstd|lz4|bz2 Compress the wire
                               protocol
-E, --encrypt none|aes|aes256|aes192|aes128 Encrypt the wire
                               protocol
-?, --help                     Display help
```

If you encounter any issues following the above steps, you can refer to the **Installation** chapter to see how to install or compile pgmoneta on your system.

3.1 Configuration

Lets create a simple configuration file called `pgmoneta.conf` with the content

```
[pgmoneta]
host = *
metrics = 5001

base_dir = /home/pgmoneta

compression = zstd

retention = 7

log_type = file
log_level = info
```



```
log_path = /tmp/pgmoneta.log

unix_socket_dir = /tmp/

[primary]
host = localhost
port = 5432
user = repl
wal_slot = repl
```

In our main section called `[pgmoneta]` we setup **pgmoneta** to listen on all network addresses. We will enable Prometheus metrics on port 5001 and have the backups live in the `/home/pgmoneta` directory. All backups are being compressed with `zstd` and kept for 7 days. Logging will be performed at `info` level and put in a file called `/tmp/pgmoneta.log`. Last we specify the location of the `unix_socket_dir` used for management operations and the path for the PostgreSQL command line tools.

Next we create a section called `[primary]` which has the information about our PostgreSQL instance. In this case it is running on `localhost` on port 5432 and we will use the `repl` user account to connect, and the Write+Ahead slot will be named `repl` as well.

The `repl` user must have the `REPLICATION` role and have access to the `postgres` database, so for example

```
CREATE ROLE repl WITH LOGIN REPLICATION PASSWORD 'secretpassword';
```

and in `pg_hba.conf`

local	postgres	repl		scram-sha-256
host	postgres	repl	127.0.0.1/32	scram-sha-256
host	postgres	repl	:::1/128	scram-sha-256
host	replication	repl	127.0.0.1/32	scram-sha-256
host	replication	repl	:::1/128	scram-sha-256

The authentication type should be based on `postgresql.conf`'s `password_encryption` value.

Then, create a physical replication slot that will be used for Write-Ahead Log streaming, like

```
SELECT pg_create_physical_replication_slot('repl', true, false);
```

Alternatively, configure automatic slot creation by adding `create_slot = yes` to `[pgmoneta]` or corresponding server section.

We will need a user vault for the `repl` account, so the following commands will add a master key, and the `repl` password. The master key should be longer than 8 characters.

```
pgmoneta-admin master-key
pgmoneta-admin -f pgmoneta_users.conf user add
```

We are now ready to run **pgmoneta**.

See the **Configuration** chapter for all configuration options.

3.2 Running

We will run **pgmoneta** using the command

```
pgmoneta -c pgmoneta.conf -u pgmoneta_users.conf
```

If this doesn't give an error, then we are ready to do backups.

pgmoneta is stopped by pressing Ctrl-C (^C) in the console where you started it, or by sending the **SIGTERM** signal to the process using `kill <pid>`.

3.3 Run-time administration

pgmoneta has a run-time administration tool called `pgmoneta-cli`.

You can see the commands it supports by using `pgmoneta-cli -?` which will give

```
pgmoneta-cli 0.12.0
  Command line utility for pgmoneta

Usage:
  pgmoneta-cli [ -c CONFIG_FILE ] [ COMMAND ]

Options:
  -c, --config CONFIG_FILE      Set the path to the
                                pgmoneta.conf file
  -h, --host HOST               Set the host name
  -p, --port PORT               Set the port number
  -U, --user USERNAME           Set the user name
  -P, --password PASSWORD       Set the password
  -L, --logfile FILE            Set the log file
  -v, --verbose                 Output text string of
                                result
  -V, --version                 Display version
                                information
  -F, --format text|json|raw    Set the output format
  -C, --compress none|gz|zstd|lz4|bz2
                                Compress the wire
                                protocol
  -E, --encrypt none|aes|aes256|aes192|aes128
                                Encrypt the wire
                                protocol
```

<code>-?, --help</code>	Display help
Commands:	
<code>backup</code>	Backup a server
<code>list-backup</code>	List the backups for a server
<code>restore</code>	Restore a backup from a server
<code>verify</code>	Verify a backup from a server
<code>archive</code>	Archive a backup from a server
<code>delete</code>	Delete a backup from a server
<code>retain</code>	Retain a backup from a server
<code>expunge</code>	Expunge a backup from a server
<code>encrypt</code>	Encrypt a file using master-key
<code>decrypt</code>	Decrypt a file using master-key
<code>ping</code>	Check if pgmoneta is alive
<code>shutdown</code>	Shutdown pgmoneta
<code>status [details]</code>	Status of pgmoneta, with optional details
<code>conf <action></code>	Manage the configuration, with one of
subcommands:	
<code>clear <what></code>	- 'reload' to reload the configuration
	Clear data, with:
	- 'prometheus' to reset the Prometheus statistics

This tool can be used on the machine running **pgmoneta** to do a backup like

```
pgmoneta-cli -c pgmoneta.conf backup primary
```

A restore would be

```
pgmoneta-cli -c pgmoneta.conf restore primary <timestamp> /path/to/restore
```

To shutdown pgmoneta you would use

```
pgmoneta-cli -c pgmoneta.conf shutdown
```

Check the outcome of the operations by verifying the exit code, like

```
echo $?
```

or by using the `-v` flag.

If pgmoneta has both Transport Layer Security (TLS) and `management` enabled then `pgmoneta-cli` can connect with TLS using the files `~/.pgmoneta/pgmoneta.key` (must be 0600 permission), `~/.pgmoneta/pgmoneta.crt` and `~/.pgmoneta/root.crt`.

3.4 Administration

pgmoneta has an administration tool called `pgmoneta-admin`, which is used to control user regis-

tration with **pgmoneta**.

You can see the commands it supports by using `pgmoneta-admin -?` which will give

```
pgmoneta-admin 0.12.0
Administration utility for pgmoneta

Usage:
pgmoneta-admin [ -f FILE ] [ COMMAND ]

Options:
-f, --file FILE          Set the path to a user file
-U, --user USER          Set the user name
-P, --password PASSWORD Set the password for the user
-g, --generate            Generate a password
-l, --length              Password length
-V, --version             Display version information
-?, --help               Display help

Commands:
master-key               Create or update the master key
user <subcommand>       Manage a specific user, where <subcommand> can
    be
    - add to add a new user
    - del to remove an existing user
    - edit to change the password for an existing
      user
    - ls  to list all available users
```

In order to set the master key for all users you can use

```
pgmoneta-admin -g master-key
```

The master key must be at least 8 characters.

Then use the other commands to add, update, remove or list the current user names, f.ex.

```
pgmoneta-admin -f pgmoneta_users.conf user add
```

3.5 Next Steps

Next steps in improving pgmoneta's configuration could be

- Update `pgmoneta.conf` with the required settings for your system
- Enable Transport Layer Security v1.2+ (TLS) for administrator access

See Configuration for more information on these subjects.

4 Configuration

The configuration is loaded from either the path specified by the `-c` flag or `/etc/pgmoneta/pgmoneta.conf`.

The configuration of **pgmoneta** is split into sections using the `[` and `]` characters.

The main section, called `[pgmoneta]`, is where you configure the overall properties of **pgmoneta**.

Other sections doesn't have any requirements to their naming so you can give them meaningful names like `[primary]` for the primary PostgreSQL instance.

All properties are in the format `key = value`.

The characters `#` and `;` can be used for comments; must be the first character on the line.

The `Bool` data type supports the following values: `on`, `yes`, `1`, `true`, `off`, `no`, `0` and `false`.

See a sample configuration for running **pgmoneta** on `localhost`.

4.1 pgmoneta

Property	Default	Unit	Required	Description
host		String	Yes	The bind address for pgmoneta
unix_socket_dir		String	Yes	The Unix Domain Socket location
base_dir		String	Yes	The base directory for the backup
metrics	0	Int	No	The metrics port (disable = 0)

Property	Default	Unit	Required	Description
metrics_cache_max_age	0	String	No	The number of seconds to keep in cache a Prometheus (metrics) response. If set to zero, the caching will be disabled. Can be a string with a suffix, like 2m to indicate 2 minutes
metrics_cache_max_size	256k	String	No	The maximum amount of data to keep in cache when serving Prometheus responses. Changes require restart. This parameter determines the size of memory allocated for the cache even if metrics_cache_max_age or metrics are disabled. Its value, however, is taken into account only if metrics_cache_max_age is set to a non-zero value. Supports suffixes: 'B' (bytes), the default if omitted, 'K' or 'KB' (kilobytes), 'M' or 'MB' (megabytes), 'G' or 'GB' (gigabytes).

Property	Default	Unit	Required	Description
management	0	Int	No	The remote management port (disable = 0)
compression	zstd	String	No	The compression type (none, gzip, client-gzip, server-gzip, zstd, client-zstd, server-zstd, lz4, client-lz4, server-lz4, bzip2, client-bzip2)
compression_level	3	Int	No	The compression level
workers	0	Int	No	The number of workers that each process can use for its work. Use 0 to disable
storage_engine	local	String	No	The storage engine type (local, ssh, s3, azure)

Property	Default	Unit	Required	Description
encryption	none	String	No	The encryption mode for encrypt wal and data none : No encryption aes or aes-256 or aes-256-cbc : AES CBC (Cipher Block Chaining) mode with 256 bit key length aes-192 or aes-192-cbc : AES CBC mode with 192 bit key length aes-128 or aes-128-cbc : AES CBC mode with 128 bit key length aes-256-ctr : AES CTR (Counter) mode with 256 bit key length aes-192-ctr : AES CTR mode with 192 bit key length aes-128-ctr : AES CTR mode with 128 bit key length
create_slot	no	Bool	No	Create a replication slot for all server. Valid values are: yes, no
ssh_hostname		String	Yes	Defines the hostname of the remote system for connection

Property	Default	Unit	Required	Description
ssh_username		String	Yes	Defines the username of the remote system for connection
ssh_base_dir		String	Yes	The base directory for the remote backup
ssh_ciphers	aes-256-ctr, aes-192-ctr, aes-128-ctr	String	No	The supported ciphers for communication. aes or aes-256 or aes-256-cbc : AES CBC (Cipher Block Chaining) mode with 256 bit key length aes-192 or aes-192-cbc : AES CBC mode with 192 bit key length aes-128 or aes-128-cbc : AES CBC mode with 128 bit key length aes-256-ctr : AES CTR (Counter) mode with 256 bit key length aes-192-ctr : AES CTR mode with 192 bit key length aes-128-ctr : AES CTR mode with 128 bit key length. Otherwise verbatim
s3_aws_region		String	Yes	The AWS region

Property	Default	Unit	Required	Description
s3_access_key_id		String	Yes	The IAM access key ID
s3_secret_access_key		String	Yes	The IAM secret access key
s3_bucket		String	Yes	The AWS S3 bucket name
s3_base_dir		String	Yes	The base directory for the S3 bucket
azure_storage_account		String	Yes	The Azure storage account name
azure_container		String	Yes	The Azure container name
azure_shared_key		String	Yes	The Azure storage account key
azure_base_dir		String	Yes	The base directory for the Azure container
retention	7, -, -, -	Array	No	The retention time in days, weeks, months, years
retention_interval	300	Int	No	The retention check interval
log_type	console	String	No	The logging type (console, file, syslog)

Property	Default	Unit	Required	Description
log_level	info	String	No	The logging level, any of the (case insensitive) strings FATAL , ERROR , WARN , INFO and DEBUG (that can be more specific as DEBUG1 thru DEBUG5). Debug level greater than 5 will be set to DEBUG5 . Not recognized values will make the log_level be INFO
log_path	pgmoneta.log	String	No	The log file location. Can be a strftime(3) compatible string.
log_rotation_age	0	String	No	The age that will trigger a log file rotation. If expressed as a positive number, is managed as seconds. Supports suffixes: 'S' (seconds, the default), 'M' (minutes), 'H' (hours), 'D' (days), 'W' (weeks). A value of 0 disables.

Property	Default	Unit	Required	Description
log_rotation_size	0	String	No	The size of the log file that will trigger a log rotation. Supports suffixes: 'B' (bytes), the default if omitted, 'K' or 'KB' (kilobytes), 'M' or 'MB' (megabytes), 'G' or 'GB' (gigabytes). A value of 0 (with or without suffix) disables.
log_line_prefix	%Y-%m-%d %H:%M:%S	String	No	A strftime(3) compatible string to use as prefix for every log line. Must be quoted if contains spaces.
log_mode	append	String	No	Append to or create the log file (append, create)
blocking_timeout	30	Int	No	The number of seconds the process will be blocking for a connection (disable = 0)
tls	off	Bool	No	Enable Transport Layer Security (TLS)
tls_cert_file		String	No	Certificate file for TLS. This file must be owned by either the user running pgmoneta or root.

Property	Default	Unit	Required	Description
tls_key_file		String	No	Private key file for TLS. This file must be owned by either the user running pgmoneta or root. Additionally permissions must be at least 0640 when owned by root or 0600 otherwise.
tls_ca_file		String	No	Certificate Authority (CA) file for TLS. This file must be owned by either the user running pgmoneta or root.
libev	auto	String	No	Select the libev backend to use. Valid options: auto, select, poll, epoll, iouring, devpoll and port
backup_max_rate	0	Int	No	The number of bytes of tokens added every one second to limit the backup rate
network_max_rate	0	Int	No	The number of bytes of tokens added every one second to limit the network backup rate

Property	Default	Unit	Required	Description
manifest	sha256	String	No	The hash algorithm for the manifest. Valid options: <code>crc32c</code> , <code>sha224</code> , <code>sha256</code> , <code>sha384</code> and <code>sha512</code>
keep_alive	on	Bool	No	Have <code>SO_KEEPALIVE</code> on sockets
nodelay	on	Bool	No	Have <code>TCP_NODELAY</code> on sockets
non_blocking	on	Bool	No	Have <code>O_NONBLOCK</code> on sockets
backlog	16	Int	No	The backlog for <code>listen()</code> . Minimum 16
hugepage	try	String	No	Huge page support (<code>off</code> , try , <code>on</code>)
pidfile		String	No	Path to the PID file. If not specified, it will be automatically set to <code>unix_socket_dir/pgmoneta.<host>.pid</code> where <code><host></code> is the value of the <code>host</code> parameter or <code>all</code> if <code>host = *</code> .

Property	Default	Unit	Required	Description
update_process_title	<code>verbose</code>	String	No	<p>The behavior for updating the operating system process title. Allowed settings are: <code>never</code> (or <code>off</code>), does not update the process title; <code>strict</code> to set the process title without overriding the existing initial process title length; <code>minimal</code> to set the process title to the base description; <code>verbose</code> (or <code>full</code>) to set the process title to the full description. Please note that <code>strict</code> and <code>minimal</code> are honored only on those systems that do not provide a native way to set the process title (e.g., Linux). On other systems, there is no difference between <code>strict</code> and <code>minimal</code> and the assumed behaviour is <code>minimal</code> even if <code>strict</code> is used. <code>never</code> and <code>verbose</code> are always honored, on every</p> <hr/> <p>system. On Linux 30 systems the process title is always trimmed to 255 characters, while on</p>

4.2 Server section

Property	Default	Unit	Required	Description
host		String	Yes	The address of the PostgreSQL instance
port		Int	Yes	The port of the PostgreSQL instance
user		String	Yes	The replication user name
wal_slot		String	Yes	The replication slot for WAL
create_slot	no	Bool	No	Create a replication slot for this server. Valid values are: yes, no
follow		String	No	Failover to this server if follow server fails
retention		Array	No	The retention for the server in days, weeks, months, years
wal_shipping		String	No	The WAL shipping directory
hot_standby		String	No	Hot standby directory
hot_standby_overrides		String	No	Files to override in the hot standby directory
hot_standby_tablespaces		String	No	Tablespace mappings for the hot standby. Syntax is [from -> to,?]+

Property	Default	Unit	Required	Description
workers	-1	Int	No	The number of workers that each process can use for its work. Use 0 to disable, -1 means use the global settting
backup_max_rate	-1	Int	No	The number of bytes of tokens added every one second to limit the backup rate. Use 0 to disable, -1 means use the global settting
network_max_rate	-1	Int	No	The number of bytes of tokens added every one second to limit the netowrk backup rate. Use 0 to disable, -1 means use the global settting
manifest	sha256	String	No	The hash algoritm for the manifest. Valid options: crc32c , sha224 , sha256 , sha384 and sha512
tls_cert_file		String	No	Certificate file for TLS. This file must be owned by either the user running pgmoneta or root.

Property	Default	Unit	Required	Description
tls_key_file		String	No	Private key file for TLS. This file must be owned by either the user running pgmoneta or root. Additionally permissions must be at least 0640 when owned by root or 0600 otherwise.
tls_ca_file		String	No	Certificate Authority (CA) file for TLS. This file must be owned by either the user running pgmoneta or root.
extra		String	No	The source directory for retrieval on the server side (details are in the extra section)

The `user` specified must have the `REPLICATION` option in order to stream the Write-Ahead Log (WAL), and must have access to the `postgres` database in order to get the necessary configuration parameters.

Note, that PostgreSQL 13+ is required, as well as having `wal_level` at `replica` or `logical` level.

Note, that if `host` starts with a `/` it represents a path and **pgmoneta** will connect using a Unix Domain Socket.

4.2.1 extra parameter

The `extra` configuration is set in the server section. It is not required, but if you configure this parameter, when you perform a backup using the CLI `pgmoneta-cli -c pgmoneta.conf backup primary`, it will also copy all specified files on the server side and send them back to the client side.

This `extra` feature requires the server side to install the `pgmoneta_ext` extension and also make the role `repl` a `SUPERUSER` (this will be improved in the future). Currently, this feature is only available to the `SUPERUSER` role.

You can set up `pgmoneta_ext` by following the README to easily install the extension. There are also more detailed instructions available in the DEVELOPERS documentation.

The format for the `extra` parameter is a path to a file or directory. You can list more than one file or directory separated by commas. The format is as follows:

```
extra = /tmp/myfile1, /tmp/myfile2, /tmp/mydir1, /tmp/mydir2
```

4.3 pgmoneta_users configuration

The `pgmoneta_users` configuration defines the users known to the system. This file is created and managed through the `pgmoneta-admin` tool.

The configuration is loaded from either the path specified by the `-u` flag or `/etc/pgmoneta/pgmoneta_users.conf`.

4.4 pgmoneta_admins configuration

The `pgmoneta_admins` configuration defines the administrators known to the system. This file is created and managed through the `pgmoneta-admin` tool.

The configuration is loaded from either the path specified by the `-A` flag or `/etc/pgmoneta/pgmoneta_admins.conf`.

If `pgmoneta` has both Transport Layer Security (TLS) and `management` enabled then `pgmoneta-cli` can connect with TLS using the files `~/.pgmoneta/pgmoneta.key` (must be 0600 permission), `~/.pgmoneta/pgmoneta.crt` and `~/.pgmoneta/root.crt`.

5 Tutorials

5.1 Install pgmoneta

This tutorial will show you how to do a simple installation of **pgmoneta**.

At the end of this tutorial you will have a backup of a PostgreSQL cluster, and will be streaming Write-Ahead Log (WAL) to **pgmoneta**.

Please note that inside the brackets at the end of each step it's the user account you should be using, switch the account when needed.

5.1.1 Preface

This tutorial assumes that you have an installation of PostgreSQL 13+ and **pgmoneta**.

For RPM based distributions such as Fedora and RHEL you can add the PostgreSQL YUM repository and do the install via

```
dnf -qy module disable postgresql
dnf install -y postgresql13 postgresql13-server pgmoneta
```

5.1.2 Initialize cluster

```
export PATH=/usr/pgsql-13/bin:$PATH
initdb -k /tmp/pgsql
```

(postgres user)

5.1.3 Remove default access

Remove

host	all	all	127.0.0.1/32	trust
host	all	all	:::1/128	trust
host	replication	all	127.0.0.1/32	trust
host	replication	all	:::1/128	trust

from /tmp/pgsql/pg_hba.conf

(postgres user)

5.1.4 Add access for users and a database

Add

host	mydb	myuser	127.0.0.1/32	md5
host	mydb	myuser	:::1/128	md5
host	postgres	repl	127.0.0.1/32	md5
host	postgres	repl	:::1/128	md5
host	replication	repl	127.0.0.1/32	md5
host	replication	repl	:::1/128	md5

to `/tmp/pgsql/pg_hba.conf`

Remember to check the value of `password_encryption` in `/tmp/pgsql/postgresql.conf` to setup the correct authentication type.

(postgres user)

5.1.5 Make sure that replication level is set

Check that

```
wal_level = replica
```

is set in `/tmp/pgsql/postgresql.conf` - or `logical`

(postgres user)

5.1.6 Start PostgreSQL

```
pg_ctl -D /tmp/pgsql/ start
```

(postgres user)

5.1.7 Add users and a database

```
createuser -P myuser  
createdb -E UTF8 -O myuser mydb
```

with `mypass` as the password.

Then

```
psql postgres
CREATE ROLE repl WITH LOGIN REPLICATION PASSWORD 'secretpassword';
\q
```

(postgres user)

5.1.8 Add Write-Ahead Log (WAL) replication slot

```
psql postgres
SELECT pg_create_physical_replication_slot('repl', true, false);
\q
```

(postgres user)

5.1.9 Verify access

For the user `myuser` using `mypass` as the password

```
psql -h localhost -p 5432 -U myuser mydb
\q
```

For the user `repl` using `secretpassword` as the password

```
psql -h localhost -p 5432 -U repl postgres
\q
```

(postgres user)

5.1.10 Add pgmoneta user

```
sudo su -
useradd -ms /bin/bash pgmoneta
passwd pgmoneta
exit
```

(postgres user)

5.1.11 Create pgmoneta configuration

Switch to the pgmoneta user

```
sudo su -
su - pgmoneta
```

Add the master key and create vault

```
pgmoneta-admin master-key  
pgmoneta-admin -f pgmoneta_users.conf -U repl -P secretpassword user add
```

You have to choose a password for the master key - remember it !

If you see an error saying `error while loading shared libraries: libpgmoneta.so.0: cannot open shared object` running the above command, you may need to locate where your `libpgmoneta.so.0` is. It could be in `/usr/local/lib` or `/usr/local/lib64` depending on your environment. Add the corresponding directory into `/etc/ld.so.conf`, or alternatively, create a file called `pgmoneta_shared_library.conf` under `/etc/ld.so.conf.d/`, and add your directory into it. Remember to run `ldconfig` to make the change effective

Create the `pgmoneta.conf` configuration

```
cat > pgmoneta.conf  
[pgmoneta]  
host = *  
metrics = 5001  
create_slot = yes  
  
base_dir = /home/pgmoneta/backup  
  
compression = zstd  
  
storage_engine = local  
  
retention = 7  
  
log_type = file  
log_level = info  
log_path = /tmp/pgmoneta.log  
  
unix_socket_dir = /tmp/  
  
[primary]  
host = localhost  
port = 5432  
user = repl  
wal_slot = repl
```

and press `Ctrl-d`

(pgmoneta user)

5.1.12 Create base directory

pgmoneta

```
mkdir backup
```

(pgmoneta user)

5.1.13 Start pgmoneta

```
pgmoneta -c pgmoneta.conf -u pgmoneta_users.conf
```

(pgmoneta user)

5.1.14 Create a backup

In another terminal

```
pgmoneta-cli -c pgmoneta.conf backup primary
```

(pgmoneta user)

5.1.15 View backup

In another terminal

```
pgmoneta-cli -c pgmoneta.conf status details
```

(pgmoneta user)

5.1.16 Shell completion

There is a minimal shell completion support for `pgmoneta-cli` and `pgmoneta-admin`. If you are running such commands from a Bash or Zsh, you can take some advantage of command completion.

5.1.16.1 Installing command completions in Bash There is a completion script into `contrib/shell_comp/pgmoneta_comp.bash` that can be used to help you complete the command line while you are typing.

It is required to source the script into your current shell, for instance by doing:

```
source contrib/shell_comp/pgmoneta_comp.bash
```

At this point, the completions should be active, so you can type the name of one the commands between `pgmoneta-cli` and `pgmoneta-admin` and hit `<TAB>` to help the command line completion.

5.1.16.2 Installing the command completions on Zsh In order to enable completion into `zsh` you first need to have `compinit` loaded; ensure your `.zshrc` file contains the following lines:

```
autoload -U compinit
compinit
```

and add the sourcing of the `contrib/shell_comp/pgmoneta_comp.zsh` file into your `~/.zshrc` also associating the `_pgmoneta_cli` and `_pgmoneta_admin` functions to completion by means of `compdef`:

```
source contrib/shell_comp/pgmoneta_comp.zsh
compdef _pgmoneta_cli    pgmoneta-cli
compdef _pgmoneta_admin  pgmoneta-admin
```

If you want completions only for one command, e.g., `pgmoneta-admin`, remove the `compdef` line that references the command you don't want to have automatic completion. At this point, digit the name of a `pgmoneta-cli` or `pgmoneta-admin` command and hit <TAB> to trigger the completion system.

5.2 Remote administration for pgmoneta

This tutorial will show you how to do setup remote management for **pgmoneta**.

5.2.1 Preface

This tutorial assumes that you have an installation of PostgreSQL 13+ and **pgmoneta**.

See `Install pgmoneta` for more detail.

5.2.2 Change the pgmoneta configuration

Change `pgmoneta.conf` to add

```
management = 5002
```

under the `[pgmoneta]` setting, like

```
[pgmoneta]
...
management = 5002
```

(`pgmoneta` user)

5.2.3 Add pgmoneta admin

```
pgmoneta-admin -f pgmoneta_admins.conf -U admin -P admin1234 user add
```

(pgmoneta user)

5.2.4 Restart pgmoneta

Shutdown pgmoneta and start it again with

```
pgmoneta-cli -c pgmoneta.conf shutdown  
pgmoneta -c pgmoneta.conf -u pgmoneta_users.conf -A pgmoneta_admins.conf
```

(pgmoneta user)

5.2.5 Connect via remote administration interface

```
pgmoneta-cli -h localhost -p 5002 -U admin status details
```

and use `admin1234` as the password

(pgmoneta user)

5.2.6 Using Transport Level Security for access

You can security the administration level interface by using Transport Level Security (TLS).

It is done by setting the following options,

```
[pgmoneta]  
tls_cert_file=/path/to/server.crt  
tls_key_file=/path/to/server.key  
tls_ca_file=/path/to/root.crt  
...
```

in `pgmoneta.conf`.

The client side setup must go into `~/ .pgmoneta/` with the following files

```
~/ .pgmoneta/pgmoneta.key  
~/ .pgmoneta/pgmoneta.crt  
~/ .pgmoneta/root.crt
```

They must have 0600 permission.

5.3 Prometheus metrics for pgmoneta

This tutorial will show you how to do setup Prometheus metrics for **pgmoneta**.

5.3.1 Preface

This tutorial assumes that you have an installation of PostgreSQL 13+ and **pgmoneta**.

See Install pgmoneta for more detail.

5.3.2 Change the pgmoneta configuration

Change `pgmoneta.conf` to add

```
metrics = 5001
```

under the `[pgmoneta]` setting, like

```
[pgmoneta]
...
metrics = 5001
```

(pgmoneta user)

5.3.3 Restart pgmoneta

Shutdown pgmoneta and start it again with

```
pgmoneta-cli -c pgmoneta.conf shutdown
pgmoneta -c pgmoneta.conf -u pgmoneta_users.conf
```

(pgmoneta user)

5.3.4 Get Prometheus metrics

You can now access the metrics via

```
http://localhost:5001/metrics
```

(pgmoneta user)

5.4 Backup and restore

This tutorial will show you how to do a backup and a restore using **pgmoneta**.

5.4.1 Preface

This tutorial assumes that you have an installation of PostgreSQL 13+ and **pgmoneta**.

See Install pgmoneta for more detail.

5.4.2 Backup

```
pgmoneta-cli -c pgmoneta.conf backup primary
```

will take a backup of the [primary] host.

(pgmoneta user)

5.4.3 List backups

```
pgmoneta-cli -c pgmoneta.conf list-backup primary
```

(pgmoneta user)

5.4.4 Restore

```
pgmoneta-cli -c pgmoneta.conf restore primary newest current /tmp/
```

will take the latest backup and all Write-Ahead Log (WAL) segments and restore it into the `/tmp/primary-<timestamp>` directory for an up-to-date copy.

(pgmoneta user)

5.5 Verify

This tutorial will show you how to verify a backup using **pgmoneta**.

5.5.1 Preface

This tutorial assumes that you have an installation of PostgreSQL 13+ and **pgmoneta**.

See Install pgmoneta for more detail.

5.5.2 Verify

```
pgmoneta-cli -c pgmoneta.conf verify primary oldest /tmp
```

will verify the oldest backup of the [primary] host.

(pgmoneta user)

5.6 Archive

This tutorial will show you how to do an archive using **pgmoneta**.

5.6.1 Preface

This tutorial assumes that you have an installation of PostgreSQL 13+ and **pgmoneta**.

See Install pgmoneta for more detail.

5.6.2 Creating an archive

```
pgmoneta-cli -c pgmoneta.conf archive primary newest current /tmp/
```

will take the latest backup and all Write-Ahead Log (WAL) segments and create an archive named `/tmp/primary-<timestamp>.tar.zstd`. This archive will contain an up-to-date copy.

(pgmoneta user)

5.7 Delete a backup

This tutorial will show you how to delete a backup from **pgmoneta**.

5.7.1 Preface

This tutorial assumes that you have an installation of PostgreSQL 13+ and **pgmoneta**.

See Install pgmoneta for more detail.

5.7.2 Delete the oldest backup

```
pgmoneta-cli -c pgmoneta.conf delete primary oldest
```

will delete the oldest backup on [primary].

(pgmoneta user)

5.8 Encryption and Decryption

This tutorial will show you how to use encryption and decryption features in **pgmoneta**.

5.8.1 Preface

This tutorial assumes that you have an installation of PostgreSQL 13+ and **pgmoneta**.

See Install pgmoneta for more detail.

5.8.2 Enable Encryption and Decryption in pgmoneta workflow

By default, the encryption is disabled. To enable this feature, modify `pgmoneta.conf`:

```
encryption = aes-256-cbc
```

(pgmoneta user)

Many encryption modes are supported, see Configuration for details.

5.8.3 Encryption and Decryption Commands

pgmoneta use the same key created by `pgmoneta-admin master-key` to encrypt and decrypt files.

Encrypt a file with `pgmoneta-cli encrypt`, the file will be encrypted in place and remove unencrypted file on success.

```
pgmoneta-cli -c pgmoneta.conf encrypt '<path-to-your-file>/file.tar.zstd'
```

Decrypt a file with `pgmoneta-cli decrypt`, the file will be decrypted in place and remove encrypted file on success.

```
pgmoneta-cli -c pgmoneta.conf decrypt '<path-to-your-file>/file.tar.zstd.aes'
```

`pgmoneta-cli encrypt` and `pgmoneta-cli decrypt` are built to deal with files created by `pgmoneta-cli archive`. It can be used on other files though.

5.9 Retention Policy

This tutorial will show you how to configure retention to retain backups.

5.9.1 Preface

This tutorial assumes that you have an installation of PostgreSQL 13+ and **pgmoneta**.

See Install pgmoneta for more detail.

5.9.2 Retention Setup

In `pgmoneta.conf`, you can use `retention = 7, 4, 12, 5` to configure **pgmoneta** to retain backups within the nearest 7 days, 4 weeks, 12 months and 5 years. Specifically, **pgmoneta** will retain all the backups within the nearest 7 days, the latest backup on each Monday within the nearest 4 weeks, the latest backup on the first day of each month in the last 12 months and the latest backup on the first day of each year in the last 5 years. If you input more than 4 values, **pgmoneta** will only read the first 4.

There are a lot of ways to leave a parameter unspecified. For trailing parameters, you can simply omit them. And for parameters in between, you can use placeholders. Currently, placeholders we allow are: `-`, `X`, `x`, `0` or whitespaces (spaces or tabs).

Please note that you should always configure `days` to retain the nearest backups. If you don't configure retention, by default **pgmoneta** keeps backups within the nearest 7 days and other parameters (weeks, months, years) are unspecified. Additionally, if you are using prometheus, unspecified values will be shown as `0`.

5.9.3 Retention Validation Rule

Current validation rule is:

1. Retention days ≥ 1
2. If retention months is specified, then $1 \leq \text{weeks} \leq 4$, otherwise weeks ≥ 1
3. If retention years is specified, then $1 \leq \text{months} \leq 12$, otherwise months ≥ 1
4. Retention years ≥ 1 Please note that the rule above only checks specified parameters, except for days, which should always be specified

5.10 Retention check

The retention check runs every 5 minutes, and will delete one backup per run.

You can change this to every 30 minutes by

```
retention_interval = 1800
```

under the `[pgmoneta]` configuration.

5.11 Grafana Dashboard

This tutorial will show you how to encapsulate Prometheus standard API and use them to monitor state of **pgmoneta** with the help of Grafana dashboard.

5.11.1 Preface

This tutorial assumes that you have an installation of PostgreSQL 13+ and **pgmoneta**.

See Install pgmoneta for more detail.

5.11.2 Prometheus Configuration

First of all, you should install Prometheus in your **pgmoneta** server.

After you successfully installed Prometheus, you should replace `prometheus.yml` with the content below to configure how to query your **pgmoneta** metrics.

```
scrape_configs:
  - job_name: 'pgmoneta'
    metrics_path: '/metrics'
    static_configs:
```


pgmoneta

```
- targets: ['localhost:5001']
```

Then the Prometheus service will query your **pgmoneta** metrics every 15 seconds and package them as time-series data. You can query your **pgmoneta** metrics and watch their changes as time passed in Prometheus web page (default port is 9090).

Prometheus Alerts Graph Status Help

☐ Enable query history

pgmoneta_backup_count

Execute - insert metric at cursor

Load time: 84ms
Resolution: 14s
Total time series: 2

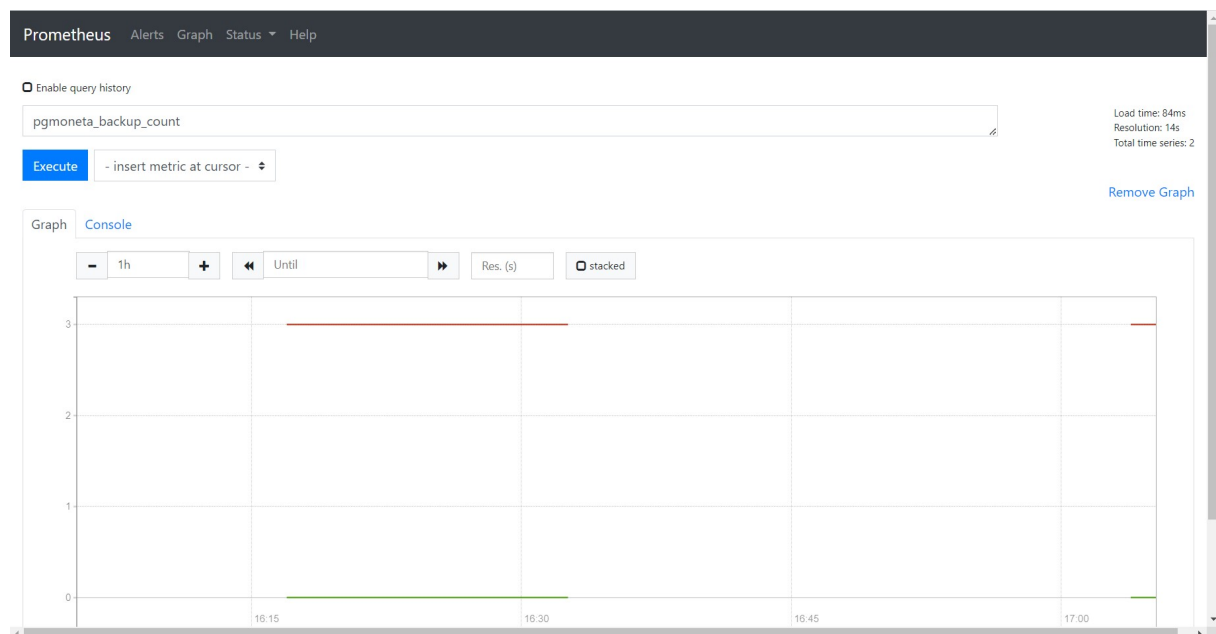
Remove Graph

Graph Console

◀ Moment ▶

Element	Value
pgmoneta_backup_count(instance="localhost:5001",job="pgmoneta",name="primary")	3
pgmoneta_backup_count(instance="localhost:5001",job="pgmoneta",name="secondary")	0

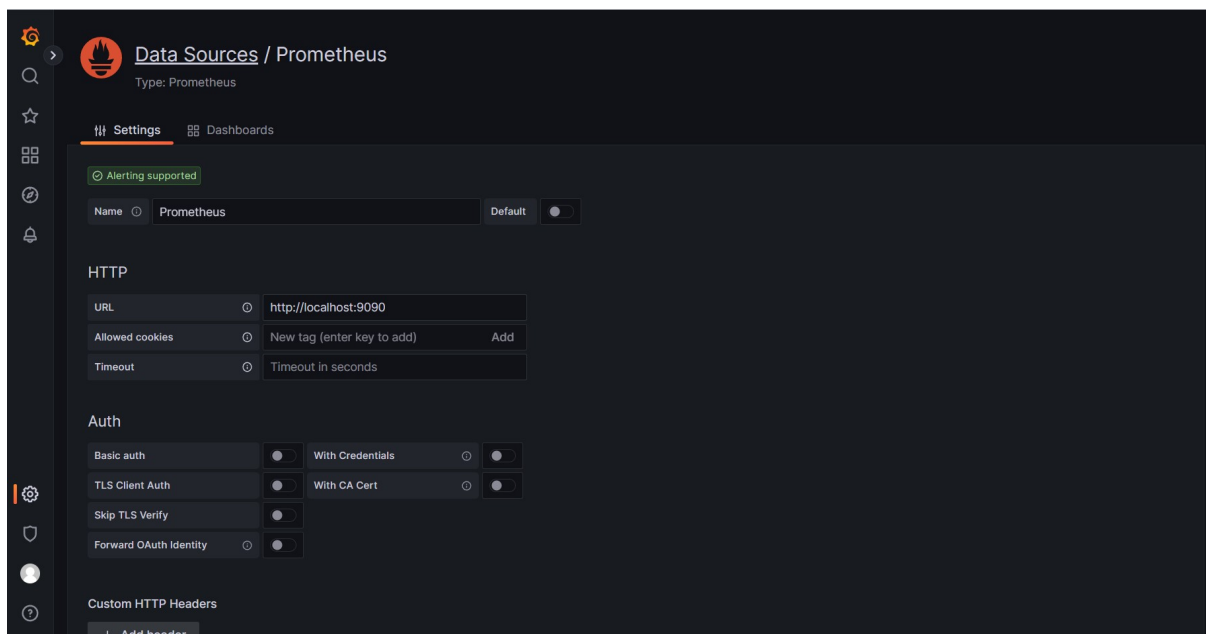
Add Graph



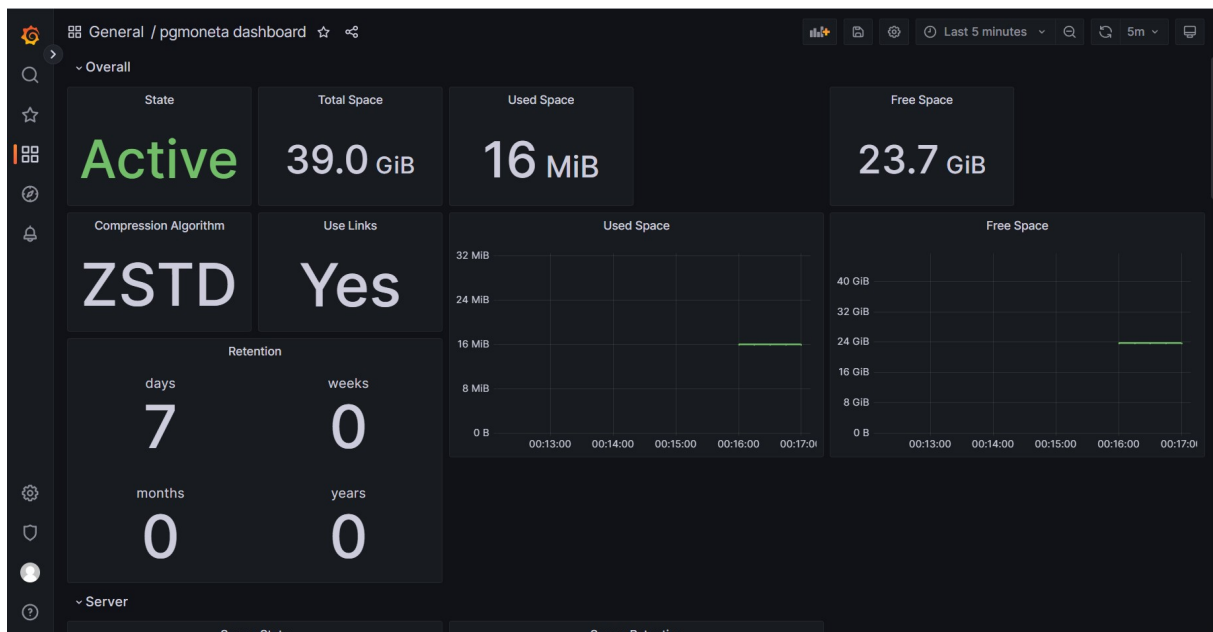
5.11.3 Grafana Dashboard Import

Although Prometheus provides capacity of querying and monitoring metrics, we can not customize graphs for each metric and provide a unified view.

As a result, we use Grafana to help us manage all graphs together. First of all, we should install Grafana in the computer you need to monitor **pgmoneta** metrics. You can browse Grafana web page with default port 3000, default user `admin` and default password `admin`. Then you can create Prometheus data source of **pgmoneta**.



Finally you can create dashboard by importing `contrib/grafana/dashboard.json` and monitor metrics about **pgmoneta**.



5.12 WAL shipping

This tutorial will show you how to configure WAL shipping, so that if the backup server crashes, you will be able to use an older archive to do Point-in-Time recovery with WAL segments shipped to another server/local directory.

Note that this feature is still at its early stages, that it currently only ships WAL segments to another LOCAL directory. You do not need to make sure the directory is unique for each server, the WAL copy will be saved under the subdirectory `server_name/wal/`

5.12.1 Preface

This tutorial assumes that you have an installation of PostgreSQL 13+ and **pgmoneta**.

See Install pgmoneta for more detail.

5.12.2 Configuration

In order to use WAL shipping, simply add

```
wal_shipping = your/local/wal/shipping/directory
```

to the corresponding server section of `pgmoneta.conf`, **pgmoneta** will create the directory if it doesn't exist, and ship a copy of WAL segments under the subdirectory `your/local/wal/shipping/directory/server_name/wal`.

5.12.3 Prometheus

You can monitor the disk usage regarding WAL shipping using prometheus. Here are some metrics.

```
pgmoneta_wal_shipping(name) -- size of the WAL shipping for the server (
    wal/shipping/directory/server_name/wal)
pgmoneta_wal_shipping_used_space -- size of everything under the WAL
    shipping directory; this could include archives (wal/shipping/directory
    /server_name/)
pgmoneta_wal_shipping_free_space -- free size of the WAL shipping
    directory for the server (wal/shipping/directory/server_name/)
pgmoneta_wal_shipping_total_space -- total size of the WAL shipping
    directory for the server (wal/shipping/directory/server_name/)
```

5.13 Use of Transport Level Security (TLS)

This tutorial is about using Transport Level Security (TLS) in PostgreSQL, and how it affects **pgmoneta**.

Note, that this tutorial is an example on how to setup a PostgreSQL TLS environment for development use only !

5.13.1 Preface

This tutorial assumes that you have an installation of PostgreSQL 13+, OpenSSL and **pgmoneta**.

See Install pgmoneta for more detail.

5.13.2 PostgreSQL

Generate the server key

```
openssl genrsa -aes256 8192 > server.key
```

Remove the passphrase

```
openssl rsa -in server.key -out server.key
```

Set the server key permission

```
chmod 400 server.key
```

Generate the server certificate

```
openssl req -new -key server.key -days 3650 -out server.crt -x509
```

Use the server certificate as the root certificate (self-signed)

```
cp server.crt root.crt
```

In `postgresql.conf` change the following settings

```
listen_addresses = '*'
ssl = on
ssl_ca_file = '/path/to/root.crt'
ssl_cert_file = '/path/to/server.crt'
ssl_key_file = '/path/to/server.key'
ssl_prefer_server_ciphers = on
```

In `pg_hba.conf` change

host	all	all	0.0.0.0/0	scram-sha-256
------	-----	-----	-----------	---------------

to

hostssl	all	all	0.0.0.0/0	scram-sha-256
---------	-----	-----	-----------	---------------

In this scenario there are no changes to the `pgmoneta.conf` configuration file.

5.13.3 Using client certificate

Create the client key

```
openssl ecparam -name prime256v1 -genkey -noout -out client.key
```

Create the client request - remember that the CN has to have the name of the replication user

```
openssl req -new -sha256 -key client.key -out client.csr -subj "/CN=repl"
```

Generate the client certificate

```
openssl x509 -req -in client.csr -CA root.crt -CAkey server.key -
CAcreateserial -out client.crt -days 3650 -sha256
```

You can test your setup by copying the files into the default PostgreSQL client directory, like

```
mkdir ~/.postgresql
cp client.crt ~/.postgresql/postgresql.crt
```

```
cp client.key ~/.postgresql/postgresql.key
cp root.crt ~/.postgresql/ca.crt
chmod 0600 ~/.postgresql/postgresql.crt ~/.postgresql/postgresql.key ~/.
    postgresql/ca.crt
```

and then test with the `psql` command.

In `pg_hba.conf` change

hostssl	all	all	0.0.0.0/0	scram-sha-256
---------	-----	-----	-----------	---------------

to

hostssl	all	all	0.0.0.0/0	scram-sha-256
	clientcert=verify-ca			

In `pgmoneta.conf` add the paths to the server in question, like

```
[pgmoneta]
...

[primary]
host=...
port=...
user=repl
tls_cert_file=/path/to/home/.postgresql/postgresql.crt
tls_key_file=/path/to/home/.postgresql/postgresql.key
tls_ca_file=/path/to/home/.postgresql/ca.crt
```

5.13.4 More information

- Secure TCP/IP Connections with SSL
- The `pg_hba.conf` File

5.14 Hot standby

This tutorial will show you how to configure a hot standby instance.

5.14.1 Preface

This tutorial assumes that you have an installation of PostgreSQL 13+ and **pgmoneta**.

See Install `pgmoneta` for more detail.

5.14.2 Configuration

In order to use hot standby, simply add

```
hot_standby = /your/local/hot/standby/directory
```

to the corresponding server section of `pgmoneta.conf`. **pgmoneta** will create the directory if it doesn't exist, and keep the latest backup in the defined directory.

You can use

```
hot_standby_overrides = /your/local/hot/standby/overrides/
```

to override files in the `hot_standby` directory.

5.14.3 Tablespaces

By default tablespaces will be mapped to a similar path than the original one, for example `/tmp/mytblspc` becomes `/tmp/mytblspchs`.

However, you can use the directory name to map it to another directory, like

```
hot_standby_tablespaces = /tmp/mytblspc->/tmp/mybcktblspc
```

You can also use the `OID` for the key part, like

```
hot_standby_tablespaces = 16392->/tmp/mybcktblspc
```

Multiple tablespaces can be specified using a `,` between them.

5.15 Annotate a backup with comments

This tutorial is about adding comments to a backup.

5.15.1 Preface

This tutorial assumes that you have an installation of PostgreSQL 13+, OpenSSL and **pgmoneta**.

See Install pgmoneta for more detail.

5.15.2 Add a comment

You can add a comment by

```
pgmoneta-cli -c pgmoneta.conf annotate primary newest add mykey mycomment
```

5.15.3 Update a comment

You can update a comment by

```
pgmoneta-cli -c pgmoneta.conf annotate primary newest update mykey  
mynewcomment
```

5.15.4 Remove a comment

You can remove a comment by

```
pgmoneta-cli -c pgmoneta.conf annotate primary newest remove mykey
```

5.15.5 View comments

You can view the comments by

```
pgmoneta-cli -c pgmoneta.conf info primary newest
```

5.16 Feature extra

This tutorial will show you how to configure the [extra](#) feature to retrieve extra files and directories (recursively) from the PostgreSQL server instance.

5.16.1 Preface

This tutorial assumes that you have an installation of PostgreSQL 13+, **pgmoneta_ext**, and **pgmoneta**.

See [Install pgmoneta_ext](#) and [Install pgmoneta](#) for more detail.

5.16.2 Configuration

In order to use [extra](#), simply add

```
extra = /tmp/myfile, /tmp/mydir, /tmp/mydirs
```


to the corresponding server section of `pgmoneta.conf`. **pgmoneta** will retrieve `extra` files and directories (recursively) from the PostgreSQL server instance when performing a backup.

You can use

```
pgmoneta-cli -c pgmoneta.conf backup primary
```

to trigger the client to get `extra` files from the server side.

All the files will be stored in the `/extra` directory under the backup directory. For our configuration, they will be stored in the `/tmp/primary/<timestamp>/extra` directory.

5.16.3 Info

You can use

```
pgmoneta-cli -c pgmoneta.conf info primary newest
```

to show all copied `extra` files, something like this (only showing successfully copied files):

```
Extra          : myfile, ...
```

6 Command line interface

The **pgmoneta-cli** command line interface controls your interaction with **pgmoneta**.

It is important that you only use the pgmoneta-cli command line interface to operate on your backup directory

Using other commands on the backup directory could cause problems.

```
pgmoneta-cli 0.16.0
  Command line utility for pgmoneta

Usage:
  pgmoneta-cli [ -c CONFIG_FILE ] [ COMMAND ]

Options:
  -c, --config CONFIG_FILE      Set the path to the
                                pgmoneta.conf file
  -h, --host HOST               Set the host name
  -p, --port PORT               Set the port number
  -U, --user USERNAME           Set the user name
  -P, --password PASSWORD       Set the password
  -L, --logfile FILE            Set the log file
  -v, --verbose                 Output text string of
                                result
  -V, --version                 Display version
                                information
  -F, --format text|json|raw    Set the output format
  -C, --compress none|gz|zstd|lz4|bz2
                                Compress the wire
                                protocol
  -E, --encrypt none|aes|aes256|aes192|aes128
                                Encrypt the wire
                                protocol
  -?, --help                    Display help

Commands:
  backup                        Backup a server
  list-backup                   List the backups for a server
  restore                       Restore a backup from a server
  verify                       Verify a backup from a server
  archive                      Archive a backup from a server
  delete                       Delete a backup from a server
  retain                       Retain a backup from a server
  expunge                      Expunge a backup from a server
  encrypt                       Encrypt a file using master-key
  decrypt                      Decrypt a file using master-key
  compress                     Compress a file using configured method
  decompress                   Decompress a file using configured method
  info                         Information about a backup
  annotate                     Annotate a backup with comments
  ping                         Check if pgmoneta is alive
```

<code>shutdown</code>	Shutdown pgmoneta
<code>status [details]</code>	Status of pgmoneta, with optional details
<code>conf <action></code>	Manage the configuration, with one of
<code>subcommands:</code>	
	- <code>'reload'</code> to reload the configuration
<code>clear <what></code>	Clear data, with:
	- <code>'prometheus'</code> to reset the Prometheus statistics

pgmoneta: <https://pgmoneta.github.io/>
Report bugs: <https://github.com/pgmoneta/pgmoneta/issues>

6.1 backup

Backup a server

Command

```
pgmoneta-cli backup <server>
```

Example

```
pgmoneta-cli backup primary
```

6.2 list-backup

List the backups for a server

Command

```
pgmoneta-cli list-backup <server>
```

Example

```
pgmoneta-cli list-backup primary
```

6.3 restore

Restore a backup from a server

Command

```
pgmoneta-cli restore <server> [<timestamp>|oldest|newest] [[current|name=X  
|xid=X|lsn=X|time=X|inclusive=X|timeline=X|action=X|primary|replica],*]  
    <directory>
```

where

- **current** means copy the Write-Ahead Log (WAL), and restore to first stable checkpoint
- **name=X** means copy the Write-Ahead Log (WAL), and restore to the label specified
- **xid=X** means copy the Write-Ahead Log (WAL), and restore to the XID specified
- **time=X** means copy the Write-Ahead Log (WAL), and restore to the timestamp specified
- **lsn=X** means copy the Write-Ahead Log (WAL), and restore to the Log Sequence Number (LSN) specified
- **inclusive=X** means that the restore is inclusive of the specified information
- **timeline=X** means that the restore is done to the specified information timeline
- **action=X** means which action should be executed after the restore (pause, shutdown)

More information

Example

```
pgmoneta-cli restore primary newest name=MyLabel,primary /tmp
```

6.4 verify

Verify a backup from a server

Command

```
pgmoneta-cli verify <server> <directory> [failed|all]
```

Example

```
pgmoneta-cli verify primary oldest /tmp
```

6.5 archive

Archive a backup from a server

Command

```
pgmoneta-cli archive <server> [<timestamp>|oldest|newest] [[current|name=X|xid=X|lsn=X|time=X|inclusive=X|timeline=X|action=X|primary|replica],*]  
<directory>
```

Example

```
pgmoneta-cli archive primary newest current /tmp
```

6.6 delete

Delete a backup from a server

Command

```
pgmoneta-cli delete <server> [<timestamp>|oldest|newest]
```

Example

```
pgmoneta-cli delete primary oldest
```

6.7 retain

Retain a backup from a server. The backup will not be deleted by the retention policy

Command

```
pgmoneta-cli retain <server> [<timestamp>|oldest|newest]
```

Example

```
pgmoneta-cli retain primary oldest
```

6.8 expunge

Expunge a backup from a server. The backup will be deleted by the retention policy

Command

```
pgmoneta-cli expunge <server> [<timestamp>|oldest|newest]
```

Example

```
pgmoneta-cli expunge primary oldest
```

6.9 encrypt

Encrypt the file in place, remove unencrypted file after successful encryption.

Command

```
pgmoneta-cli encrypt <file>
```

6.10 decrypt

Decrypt the file in place, remove encrypted file after successful decryption.

Command

```
pgmoneta-cli decrypt <file>
```

6.11 compress

Compress the file in place, remove uncompressed file after successful compression.

Command

```
pgmoneta-cli compress <file>
```

6.12 decompress

Decompress the file in place, remove compressed file after successful decompression.

Command

```
pgmoneta-cli decompress <file>
```

6.13 info

Information about a backup.

Command

```
pgmoneta-cli info <server> <timestamp|oldest|newest>
```

6.14 ping

Verify if **pgmoneta** is alive

Command

```
pgmoneta-cli ping
```

Example

```
pgmoneta-cli ping
```

6.15 shutdown

Shutdown **pgmoneta**

Command

```
pgmoneta-cli shutdown
```

Example

```
pgmoneta-cli shutdown
```

6.16 status

Status of **pgmoneta**, with a `details` option

Command

```
pgmoneta-cli status [details]
```

Example

```
pgmoneta-cli status details
```

6.17 conf

Manage the configuration

Command

```
pgmoneta-cli conf [reload]
```

Subcommand

- `reload`: Reload configuration

Example

```
pgmoneta-cli conf reload
```

6.18 clear

Clear data/statistics

Command

pgmoneta

```
pgmoneta-cli clear [prometheus]
```

Subcommand

- **prometheus**: Reset the Prometheus statistics

Example

```
pgmoneta-cli clear prometheus
```

6.19 Shell completions

There is a minimal shell completion support for **pgmoneta-cli**.

Please refer to the Install pgmoneta tutorial for detailed information about how to enable and use shell completions.

7 Prometheus metrics

pgmoneta has the following Prometheus metrics.

7.1 pgmoneta_state

The state of pgmoneta

1 = Running

7.2 pgmoneta_version

The version of pgmoneta

7.3 pgmoneta_logging_info

The number of INFO statements

7.4 pgmoneta_logging_warn

The number of WARN statements

7.5 pgmoneta_logging_error

The number of ERROR statements

7.6 pgmoneta_logging_fatal

The number of FATAL statements

7.7 pgmoneta_retention_days

The retention of pgmoneta in days

7.8 pgmoneta_retention_weeks

The retention of pgmoneta in weeks

7.9 pgmoneta_retention_months

The retention of pgmoneta in months

7.10 pgmoneta_retention_years

The retention of pgmoneta in years

7.11 pgmoneta_retention_server

The retention of a server

Attribute	Description
name	The identifier for the server
parameter	days weeks months years

7.12 pgmoneta_extension

The version of pgmoneta extension

7.13 pgmoneta_compression

The compression used

0 = None

1 = GZip

2 = ZSTD

3 = LZ4

4 = BZIP2

7.14 pgmoneta_used_space

The disk space used for pgmoneta

7.15 pgmoneta_free_space

The free disk space for pgmoneta

7.16 pgmoneta_total_space

The total disk space for pgmoneta

7.17 pgmoneta_server_valid

Is the server in a valid state

7.18 pgmoneta_wal_streaming

The WAL streaming status of a server

7.19 pgmoneta_server_operation_count

The count of client operations of a server

7.20 pgmoneta_server_failed_operation_count

The count of failed client operations of a server

7.21 pgmoneta_server_last_operation_time

The time of the latest client operation of a server

7.22 pgmoneta_server_last_failed_operation_time

The time of the latest failed client operation of a server

7.23 pgmoneta_wal_shipping

The disk space used for WAL shipping for a server

7.24 pgmoneta_wal_shipping_used_space

The disk space used for everything under the WAL shipping directory of a server

7.25 pgmoneta_wal_shipping_free_space

The free disk space for the WAL shipping directory of a server

7.26 pgmoneta_wal_shipping_total_space

The total disk space for the WAL shipping directory of a server

7.27 pgmoneta_hot_standby

The disk space used for hot standby for a server

7.28 pgmoneta_hot_standby_free_space

The free disk space for the hot standby directory of a server

7.29 pgmoneta_hot_standby_total_space

The total disk space for the hot standby directory of a server

7.30 pgmoneta_server_timeline

The current timeline a server is on

Attribute	Description
name	The identifier for the server

7.31 pgmoneta_server_parent_tli

The parent timeline of a timeline on a server

Attribute	Description
name	The identifier for the server
tli	The current/previous timeline ID in the server history

7.32 pgmoneta_server_timeline_switchpos

The WAL switch position of a timeline on a server (showed in hex as a parameter)

Attribute	Description
name	The identifier for the server
tli	The current/previous timeline ID in the server history
walpos	The WAL switch position of this timeline

7.33 pgmoneta_server_workers

The number of workers for a server

Attribute	Description
name	The identifier for the server

7.34 pgmoneta_backup_oldest

The oldest backup for a server

Attribute	Description
name	The identifier for the server

7.35 pgmoneta_backup_newest

The newest backup for a server

Attribute	Description
name	The identifier for the server

7.36 pgmoneta_backup_count

The number of valid backups for a server

Attribute	Description
name	The identifier for the server

7.37 pgmoneta_backup

Is the backup valid for a server

Attribute	Description
name	The identifier for the server
label	The backup label

7.38 pgmoneta_backup_version

The version of PostgreSQL for a backup

Attribute	Description
name	The identifier for the server
label	The backup label
major	The backup PostgreSQL major version
minor	The backup PostgreSQL minor version

7.39 pgmoneta_backup_throughput

The throughput of the backup for a server (bytes/s)

name The identifier for the server label The backup label

7.40 pgmoneta_backup_elapsed_time

The backup in seconds for a server

Attribute	Description
name	The identifier for the server
label	The backup label

7.41 pgmoneta_backup_start_timeline

The starting timeline of a backup for a server

Attribute	Description
name	The identifier for the server
label	The backup label

7.42 pgmoneta_backup_end_timeline

The ending timeline of a backup for a server

Attribute	Description
name	The identifier for the server
label	The backup label

7.43 pgmoneta_backup_start_walpos

The starting WAL position of a backup for a server

Attribute	Description
name	The identifier for the server
label	The backup label
walpos	The backup starting WAL position

7.44 pgmoneta_backup_checkpoint_walpos

The checkpoint WAL pos of a backup for a server

Attribute	Description
name	The identifier for the server
label	The backup label
walpos	The backup checkpoint WAL position

7.45 pgmoneta_backup_end_walpos

The ending WAL pos of a backup for a server

Attribute	Description
name	The identifier for the server
label	The backup label
walpos	The backup ending WAL position

7.46 pgmoneta_restore_newest_size

The size of the newest restore for a server

Attribute	Description
name	The identifier for the server

7.47 pgmoneta_backup_newest_size

The size of the newest backup for a server

Attribute	Description
name	The identifier for the server

7.48 pgmoneta_restore_size

The size of a restore for a server

Attribute	Description
name	The identifier for the server
label	The backup label

7.49 pgmoneta_restore_size_increment

The increment size of a restore for a server

Attribute	Description
name	The identifier for the server
label	The backup label

7.50 pgmoneta_backup_size

The size of a backup for a server

Attribute	Description
name	The identifier for the server
label	The backup label

7.51 pgmoneta_backup_compression_ratio

The ratio of backup size to restore size for each backup

Attribute	Description
name	The identifier for the server
label	The backup label

7.52 pgmoneta_backup_retain

Retain a backup for a server

Attribute	Description
name	The identifier for the server
label	The backup label

7.53 pgmoneta_backup_total_size

The total size of the backups for a server

Attribute	Description
name	The identifier for the server

7.54 pgmoneta_wal_total_size

The total size of the WAL for a server

Attribute	Description
name	The identifier for the server

7.55 pgmoneta_total_size

The total size for a server

Attribute	Description
name	The identifier for the server

7.56 pgmoneta_active_backup

Is there an active backup for a server

Attribute	Description
name	The identifier for the server

7.57 pgmoneta_current_wal_file

The current streaming WAL filename of a server

Attribute	Description
name	The identifier for the server
file	The current WAL filename for this server

7.58 pgmoneta_current_wal_lsn

The current WAL log sequence number

Attribute	Description
name	The identifier for the server
lsn	The current WAL log sequence number

8 SSH

8.1 Prerequisites

First of all, you need to have a remote server where you can store your backups on.

Lets take an EC2 instance as an example, after launching an EC2 instance you need to add new user account with SSH access to the EC2 instance:

1. Connect to your Linux instance using SSH.
2. Use the `adduser` command to add a new user account to an EC2 instance (replace `new_user` with the new account name).

```
sudo adduser new_user --disabled-password
```

3. Change the security context to the `new_user` account so that folders and files you create have the correct permissions:

```
sudo su - new_user
```

4. Create a `.ssh` directory in the `new_user` home directory and use the `chmod` command to change the `.ssh` directory's permissions to 700:

```
mkdir .ssh && chmod 700 .ssh
```

5. Use the `touch` command to create the `authorized_keys` file in the `.ssh` directory and use the `chmod` command to change the `.ssh/authorized_keys` file permissions to 600:

```
touch .ssh/authorized_keys && chmod 600 .ssh/authorized_keys
```

6. Retrieve the public key for the key pair in your local computer:

```
cat ~/.ssh/id_rsa.pub
```

7. In the EC2 instance, run the `cat` command in append mode:

```
cat >> .ssh/authorized_keys
```

8. Paste the public key into the `.ssh/authorized_keys` file and then press Enter.
9. Press and hold `Ctrl+d` to exit `cat` and return to the command line session prompt.

To verify that the new user can use SSH to connect to the EC2 instance, run the following command from a command line prompt on your local computer:

```
ssh new_user@public_dns_name_of_EC2_instance
```

8.2 Modify the pgmoneta configuration

You need to create a directory on your remote server where backups can be stored in.

In addition, your local computer needs to have a storage space for 1 backup.

Change `pgmoneta.conf` to add

```
storage_engine = ssh
ssh_hostname = your-public-dns-name-of-EC2-instance
ssh_username = new_user
ssh_base_dir = the-path-of-the-directory-where-backups-stored-in
```

under the `[pgmoneta]` section.

9 Azure

9.1 Prerequisites

First of all, you need to have an Azure account, an Azure storage account and a blob container.

A container organizes a set of blobs, similar to a directory in a file system. A storage account can include an unlimited number of containers, and a container can store an unlimited number of blobs.

To create an Azure storage account with the Azure portal:

1. Sign in to the Azure portal.
2. From the left portal menu, select Storage accounts to display a list of your storage accounts. If the portal menu isn't visible, click the menu button to toggle it on.
3. On the Storage accounts page, select Create.
4. On the Basics tab, provide a resource group name and storage account name. You can go for the default settings of the other fields.
5. Choose Next: Advanced.
6. On the Advanced tab, you can configure additional options and modify default settings for your new storage account. You can go for the default settings.
7. Choose Next: Networking.
8. On the Networking tab, you can go for the default settings.
9. Choose Next: Data protection.
10. On the Data protection tab, you can for the default settings.
11. Choose Next: Encryption.
12. On the Encryption tab, you can for the default settings.
13. Choose Next: Tags.
14. Choose Next: Review to see all of the choices you made up to this point. When you are ready to proceed, choose Create.

To create a blob container with the Azure portal:

1. In the navigation pane for the storage account, scroll to the [Data storage](#) section and select Containers.
2. Within the Containers pane, select the + [Container](#) button to open the New container pane.

3. Within the New Container pane, provide a Name for your new container.
4. Select Create to create the container.

To get the Azure storage account shared key which is required for pgmoneta configuration:

1. In the navigation pane for the storage account, scroll to the **Security + networking** section and select Access Keys.
2. Under key1, find the Key value. Select the Copy button to copy the account key.

You can use either of the two keys to access Azure Storage, but in general it's a good practice to use the first key, and reserve the use of the second key for when you are rotating keys.

9.2 Modify the pgmoneta configuration

You need to have a storage space for 1 backup on your local computer.

Change `pgmoneta.conf` to add

```
storage_engine = azure
azure_storage_account = the-storage-account-name
azure_container = the-container-name
azure_shared_key = the-storage-account-shared-key
azure_base_dir = directory-where-backups-will-be-stored-in
```

under the `[pgmoneta]` section.

10 S3

10.1 Prerequisites

First of all, you need to have an AWS account, an IAM user and S3 bucket.

To create an IAM user:

1. Sign in to the AWS Management Console and open the IAM console.
2. In the navigation pane, choose Users and then choose Add users.
3. Type the user name for the new user.
4. Select the type of access to be both programmatic access and access to the AWS Management Console.
5. Choose Next: Permissions.
6. On the Set permissions page, select attach existing policies directly, search for AmazonS3FullAccess and choose it, then choose Next: Review, then choose Add permissions.
7. Choose Next: Tags.
8. Choose Next: Review to see all of the choices you made up to this point. When you are ready to proceed, choose Create user.
9. To view the users' access keys (access key IDs and secret access keys), choose Show next to each password and access key that you want to see. To save the access keys, choose Download .csv and then save the file to a safe location.

You are now ready to create a S3 bucket, To create a S3 bucket:

1. Sign in to the AWS Management Console using your IAM user credentials and open the Amazon S3 console.
2. Choose Create bucket.
3. In Bucket name, enter a name for your bucket.
4. In Region, choose the AWS Region where you want the bucket to reside.
5. Keep the default values as it is and Choose Create bucket.

10.2 Modify the pgmoneta configuration

You need to have a storage space for 1 backup on your local computer.

Change `pgmoneta.conf` to add

```
storage_engine = s3
s3_aws_region = the-aws-region
s3_access_key_id = your-access-key-id-from-the-downloaded-file
s3_secret_access_key = your-secret-access-key-from-the-downloaded-file
s3_bucket = your-s3-bucket-name
s3_base_dir = directory-where-backups-will-be-stored-in
```

under the [pgmoneta] section.

11 Troubleshooting

11.1 Could not get version for server

If you get this `FATAL` during startup check your PostgreSQL logins

```
psql postgres
```

and

```
psql -U repl postgres
```

And, check the PostgreSQL logs for any error.

Setting `log_level` to `DEBUG5` in `pgmoneta.conf` could provide more information about the error.

12 Acknowledgement

12.1 Authors

pgmoneta was created by the following authors:

```
Jesper Pedersen <jesper.pedersen@comcast.net>
David Fetter <david@fetter.org>
Will Leinweber <will@bitfission.com>
Luca Ferrari <fluca1978@gmail.com>
Nikita Bugrovsky <nbugrovs@redhat.com>
Mariam Fahmy <mariamfahmy66@gmail.com>
Jichen Xu <kyokitisin@gmail.com>
Saurav Pal <resyfer.dev@gmail.com>
Bokket <bokkett@gmail.com>
Haoran Zhang <andrewzhr9911@gmail.com>
Hazem Alrawi <hazemalrawi7@gmail.com>
Shahryar Soltanpour <shahryar.soltanpour@gmail.com>
Shikhar Soni <shikharish05@gmail.com>
Nguyen Cong Nhat Le <lenguyencongnhat2001@gmail.com>
Chao Gu <chadraven369@gmail.com>
Luchen Zhao <lucian.zlc@gmail.com>
Joan Jeremiah J <joanjeremiah04@gmail.com>
Iury Santos <iuryroberto@gmail.com>
Palak Chaturvedi <palakchaturvedi2843@gmail.com>
Jakub Jirutka <jakub@jirutka.cz>
```

12.2 Committers

```
Jesper Pedersen <jesper.pedersen@comcast.net>
Haoran Zhang <andrewzhr9911@gmail.com>
```

12.3 Contributing

Contributions to **pgmoneta** are managed on GitHub

- Ask a question
- Raise an issue
- Feature request
- Code submission

Contributions are most welcome!

Please, consult our Code of Conduct policies for interacting in our community.

Consider giving the project a star on GitHub if you find it useful. And, feel free to follow the project on Twitter as well.

13 License

Copyright (C) 2024 The pgmoneta community

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, **this** list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, **this** list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from **this** software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

BSD-3-Clause

13.1 libart

Our adaptive radix tree (ART) implementation is based on The Adaptive Radix Tree: ARTful Indexing for Main-Memory Databases and libart which has a 3-BSD license as

Copyright (c) 2012, Armon Dadgar
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, **this** list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, **this** list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of the organization nor the names of its contributors may be used to endorse or promote products derived from **this** software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL ARMON DADGAR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.