



EDB

Lasso

Release 4.11.0

EnterpriseDB Corporation.

Dec 26, 2023

Contents

1	Release notes	3
2	Lasso - Version 4.11.0 (2023-12-14)	5
2.1	New features	5
2.2	Bug fix	5
2.3	Lasso - Version 4.10.0 (2023-11-07)	5
2.3.1	New features	5
2.3.2	Deprecation notes	6
2.4	Lasso - Version 4.9.0 (2023-10-17)	6
2.4.1	New features	6
2.4.2	Bug fix	6
2.5	Lasso - Version 4.8.0 (2023-07-06)	6
2.5.1	New features	6
2.6	Lasso - Version 4.7.0 (2023-06-01)	6
2.6.1	New features	6
2.6.2	Security fixes	7
2.6.3	Other bug fixes	7
2.7	Lasso - Version 4.6.0 (2022-12-06)	7
2.7.1	New feature	7
2.7.2	Improvements	7
2.7.3	Bug fix	7
2.8	Lasso - Version 4.5.0 (2022-11-15)	7
2.8.1	Improvements	7
2.9	Lasso - Version 4.4.0 (2022-09-22)	7
2.9.1	Improvements	7
2.10	Lasso - Version 4.3.0 (2022-08-25)	8
2.10.1	Improvements	8
2.10.2	Bug	8
2.11	Lasso - Version 4.2.0 (2022-05-18)	8
2.11.1	Improvements	8
2.11.2	Other changes	8
2.11.3	Bug	8
2.12	Lasso - Version 4.1.1 (2022-05-10)	9
2.12.1	Bug	9
2.13	Lasso - Version 4.1.0 (2022-05-03)	9
2.13.1	Improvements	9

2.13.2	Other changes	9
2.13.3	Bug	9
2.14	Lasso - Version 4.0.0 (2022-03-30)	9
2.14.1	Features	9
3	Installing Lasso	11
3.1	Linux	11
3.2	Windows	12
3.3	Configuration file	12
3.4	Executing Lasso	13
4	Usage	15
5	Configuration	19
6	Security considerations	21
7	Report types	23
7.1	Local/Full PostgreSQL report	23
7.2	Remote/Database-only PostgreSQL report	24
7.3	Barman report	25
7.3.1	Managing custom installations of Barman	25
7.4	Replication Manager (repmgr) report	25
7.4.1	How Lasso finds the repmgr configuration file	25
7.4.2	Managing custom installations of repmgr	26
7.5	Postgres Enterprise Manager (PEM) report	26
7.6	Failover Manager (EFM) report	26
7.6.1	How Lasso finds the EFM configuration file	26
7.6.2	How Lasso finds the EFM binary	26
7.6.3	Managing custom installations of EFM	27
7.7	Replication Server (xDB) report	27
7.7.1	How Lasso finds the xDB configuration file	27
7.7.2	How Lasso finds the xDB binary	28
7.7.3	Managing custom installations of xDB	28
7.8	PgBouncer report	28
7.8.1	How Lasso finds the PgBouncer configuration file	29
7.9	HARP report	29
7.9.1	How Lasso finds the HARP configuration file	29
7.9.2	Managing custom installations of HARP	29
7.9.3	Running Lasso on a HARP proxy node	29
7.10	PGD Proxy report	30
7.10.1	How Lasso finds the PGD Proxy configuration file	30
7.10.2	Managing custom installations of PGD Proxy	30
7.10.3	Running Lasso on a PGD Proxy node	30
7.11	etcd report	30
7.11.1	How Lasso finds the etcd configuration file	30
7.12	System-only report	31
8	Returning the Lasso report	33
9	What do we gather in our Lasso report?	35
10	Detailed breakdown of gathered data	37
10.1	GNU/Linux Operating System	37
10.1.1	barman crontab/cron (barman_crontab_cron)	37

10.1.2	debug_sources (debug_sources)	37
10.1.3	EFM CLI (efm_cli)	38
10.1.4	EFM configuration (efm_configuration)	38
10.1.5	EFM systemctl (efm_systemctl)	38
10.1.6	etcd CLI (etcd_cli)	38
10.1.7	etcd configuration (etcd_configuration)	39
10.1.8	etcd systemctl (etcd_systemctl)	39
10.1.9	HARP CLI (harp_cli)	39
10.1.10	HARP configuration (harp_configuration)	40
10.1.11	HARP systemctl (harp_systemctl)	40
10.1.12	Block devices layout (linux_block_devices_layout)	40
10.1.13	Processor governor (linux_cpu_governor)	40
10.1.14	Mounted file systems and available space (linux_devices_info)	41
10.1.15	File systems configuration (linux_disk_configuration)	41
10.1.16	OS distribution, kernel and device data (linux_distro_collector)	41
10.1.17	Hardware (linux_hardware_info)	41
10.1.18	Hypervisor (linux_hypervisor_collector)	42
10.1.19	Kernel (linux_kernel_info)	42
10.1.20	Kernel limits (linux_kernel_limits)	42
10.1.21	Processor usage statistics (linux_mpstat)	42
10.1.22	Network interfaces (linux_network_interfaces)	43
10.1.23	Installed packages via rpm or dpkg (linux_packages_info)	43
10.1.24	PostgreSQL disk layout (linux_postgresql_disk_layout)	43
10.1.25	SELinux (linux_sestatus)	43
10.1.26	System identification (linux_system_identity)	44
10.1.27	dmesg and /proc information (linux_system_info)	44
10.1.28	System status — iostat (linux_system_status_iostat)	45
10.1.29	System status — nfsiostat (linux_system_status_nfsiostat)	45
10.1.30	System status — ps (linux_system_status_ps)	45
10.1.31	System status — sar (linux_system_status_sar)	45
10.1.32	System status — top (linux_system_status_top)	46
10.1.33	System status — vmstat (linux_system_status_vmstat)	46
10.1.34	systemctl units (linux_systemctl_units)	46
10.1.35	tuned (linux_tuned)	46
10.1.36	PEM configuration (pem_configuration)	47
10.1.37	PEM systemctl (pem_systemctl)	47
10.1.38	PgBouncer configuration (pgbouncer_configuration)	47
10.1.39	PgBouncer systemctl (pgbouncer_systemctl)	47
10.1.40	PGD Proxy configuration (pgd_proxy_configuration)	48
10.1.41	PGD Proxy systemctl (pgd_proxy_systemctl)	48
10.1.42	postgres/enterprisedb crontab (postgres_enterprisedb_crontab)	48
10.1.43	repmgr CLI (repmgr_cli)	48
10.1.44	repmgr configuration (repmgr_configuration)	49
10.1.45	repmgr systemctl (repmgr_systemctl)	49
10.1.46	xDB CLI (xdb_cli)	49
10.1.47	xDB configuration (xdb_configuration)	49
10.1.48	xDB systemctl (xdb_systemctl)	50
10.2	Microsoft Windows Operating System	50
10.2.1	PEM configuration — Windows (pem_configuration_windows)	50
10.2.2	PEM sc (pem_sc)	50
10.2.3	Disk information (win_disk_information)	50
10.2.4	Hosts file (win_hosts)	51
10.2.5	MsInfo (win_msinfo)	51
10.2.6	ODBC/64 (win_odbc32_info)	51

10.2.7	ODBC/32 (win_odbc64_info)	52
10.2.8	systeminfo (win_systeminfo)	52
10.2.9	Disk volumes (win_volumes)	52
10.2.10	xDB CLI — Windows (xdb_cli_windows)	52
10.2.11	xDB configuration — Windows (xdb_configuration_windows)	53
10.2.12	xDB sc (xdb_sc)	53
10.3	PostgreSQL/BDR3 Instance	53
10.3.1	Current archiver stats (postgresql_archiver)	53
10.3.2	Available extensions (postgresql_available_extensions)	53
10.3.3	Current bg_writer stats (postgresql_bgwriter)	54
10.3.4	Directory with binaries (postgresql_bin_dir)	54
10.3.5	Current configuration (postgresql_configuration)	54
10.3.6	Configuration files (postgresql_configuration_files)	54
10.3.7	Databases (postgresql_databases)	55
10.3.8	postgresql_db_bdr_tables_and_views (postgresql_db_bdr_tables_and_views)	55
10.3.9	postgresql_db_pglogical_tables_and_views (postgresql_db_pglogical_tables_and_views)	55
10.3.10	Database/Role Setting (postgresql_db_role_setting)	55
10.3.11	Node and snapshot data (postgresql_node)	56
10.3.12	pg_config (postgresql_pg_config)	56
10.3.13	pg_controldata (postgresql_pg_controldata)	56
10.3.14	Version (postgresql_pg_version)	56
10.3.15	Current pg_prepared_xacts contents (postgresql_prepared_xacts)	57
10.3.16	Current pg_replication_origin_status contents (postgresql_replication_origin)	57
10.3.17	Current pg_replication_slots contents (postgresql_replication_slots)	57
10.3.18	Roles (postgresql_roles)	57
10.3.19	Current activity stats (postgresql_running_activity)	58
10.3.20	Active locks (postgresql_running_locks)	58
10.3.21	pg_server_limits (postgresql_server_limits)	58
10.3.22	Current pg_shmem_allocations contents (postgresql_shmem_allocations)	58
10.3.23	Current pg_stat_progress_analyze contents (postgresql_stat_progress_analyze)	58
10.3.24	Current pg_stat_progress_basebackup contents (postgresql_stat_progress_basebackup)	59
10.3.25	Current pg_stat_progress_copy contents (postgresql_stat_progress_copy)	59
10.3.26	Current pg_stat_progress_vacuum contents (postgresql_stat_progress_vacuum)	59
10.3.27	Current pg_stat_replication contents (postgresql_stat_replication)	59
10.3.28	Server subscription statistics (postgresql_subscription_statistics)	60
10.3.29	Server subscriptions (postgresql_subscriptions)	60
10.3.30	Tablespaces (postgresql_tablespaces)	60
10.3.31	Workload characteristics using waits (postgresql_waits_stats)	60
10.4	Details for every PostgreSQL/BDR3 Database	60
10.4.1	BDR1 replication slots (postgresql_db_bdr1_replication_slots)	60
10.4.2	BDR2 replication slots (postgresql_db_bdr2_replication_slots)	61
10.4.3	BDR conflict_history_summary aggregation (postgresql_db_bdr3_conflict_history_summary_agg)	61
10.4.4	BDR current activity stats (postgresql_db_bdr3_stat_activity)	61
10.4.5	BDR sequences (postgresql_db_bdr_sequences)	61
10.4.6	BDR version (postgresql_db_bdr_version)	62
10.4.7	Database extensions (postgresql_db_extensions)	62
10.4.8	Database indexes (postgresql_db_indexes)	62
10.4.9	Database procedural languages (postgresql_db_languages)	62
10.4.10	BDR monitor_group_raft (postgresql_db_monitor_group_raft)	62
10.4.11	BDR monitor_group_versions (postgresql_db_monitor_group_versions)	63
10.4.12	BDR monitor_local_replslots (postgresql_db_monitor_local_replslots)	63

10.4.13	Database operators (postgresql_db_operators)	63
10.4.14	Database partitioned tables (postgresql_db_partitioned_tables)	63
10.4.15	Database PEM pem.agent table tuples (postgresql_db_pem_agent)	63
10.4.16	Database PEM pem.agent_config table tuples (postgresql_db_pem_agent_config) . . .	64
10.4.17	Database PEM pem.agent_heartbeat table tuples (postgresql_db_pem_agent_heartbeat)	64
10.4.18	Database PEM pem.agent_server_binding table tuples (postgresql_db_pem_agent_server_binding)	64
10.4.19	Database PEM pem.config table tuples (postgresql_db_pem_config)	64
10.4.20	Database PEM pem.email_group table tuples (postgresql_db_pem_email_group) . . .	65
10.4.21	Database PEM pem.email_group_option table tuples (postgresql_db_pem_email_group_option)	65
10.4.22	Database PEM pem.probe table tuples (postgresql_db_pem_probe)	65
10.4.23	Database PEM pem.probe_schedule table tuples (postgresql_db_pem_probe_schedule)	65
10.4.24	Database PEM pem.schema_version() output (postgresql_db_pem_schema_version) .	66
10.4.25	Database PEM pem.server table tuples (postgresql_db_pem_server)	66
10.4.26	Database PEM pem.server_heartbeat table tuples (postgresql_db_pem_server_heartbeat)	66
10.4.27	Database PEM pem.smtp_pool table tuples (postgresql_db_pem_smtp_pool)	66
10.4.28	Database PEM pem.snmp_pool table tuples (postgresql_db_pem_snmp_pool)	67
10.4.29	Pglogical subscription status (postgresql_db_pglogical_subscription_status) . .	67
10.4.30	Database functions (postgresql_db_procs)	67
10.4.31	Database publication tables (postgresql_db_publication_tables)	67
10.4.32	Database publications (postgresql_db_publications)	68
10.4.33	Database repmgr repmgr.events table tuples (postgresql_db_repmgr_events)	68
10.4.34	Database repmgr repmgr.monitoring_history table tuples (postgresql_db_repmgr_monitoring_history)	68
10.4.35	Database repmgr repmgr.nodes table tuples (postgresql_db_repmgr_nodes)	68
10.4.36	Database repmgr repmgr.replication_status table tuples (postgresql_db_repmgr_replication_status)	69
10.4.37	Database repmgr repmgr.show_nodes table tuples (postgresql_db_repmgr_show_nodes)	69
10.4.38	Database schema (postgresql_db_schemas)	69
10.4.39	Database statistics (postgresql_db_statistics)	69
10.4.40	Database subscription tables (postgresql_db_subscription_tables)	70
10.4.41	Database tables (postgresql_db_tables)	70
10.4.42	Database types (postgresql_db_types)	70
10.4.43	Database xDB _edb_replicator_pub.xdb_mmr_pub_group table tuples (postgresql_db_xdb_mmr_pub_group)	70
10.4.44	Database xDB _edb_replicator_pub.xdb_pub_database table tuples (postgresql_db_xdb_pub_database)	71
10.4.45	Database xDB _edb_replicator_pub.xdb_pub_replog table tuples (postgresql_db_xdb_pub_replog)	71
10.4.46	Database xDB _edb_replicator_pub.xdb_publication_subscriptionstable tuples (postgresql_db_xdb_publication_subscriptions)	71
10.4.47	Database xDB _edb_replicator_pub.xdb_publications table tuples (postgresql_db_xdb_publications)	71
10.4.48	Database xDB _edb_replicator_pub.rrep_mmr_pub_group table tuples (postgresql_db_xdb_rrep_mmr_pub_group)	72
10.4.49	Database xDB _edb_replicator_pub.rrep_mmr_txset table tuples (postgresql_db_xdb_rrep_mmr_txset)	72
10.4.50	Database xDB _edb_replicator_pub.rrep_properties table tuples (postgresql_db_xdb_rrep_properties)	72
10.4.51	Database xDB _edb_replicator_pub.rrep_publication_tablestable tuples (postgresql_db_xdb_rrep_publication_tables)	72

10.4.52	Database	xDB	_edb_replicator_pub.rrep_txset	table	tuples	
	(postgresql_db_xdb_rrep_txset)					73
10.4.53	Oracle-compatible partitioning key view	(postgresql_epas_all_part_key_columns)				73
10.4.54	Oracle-compatible all partitioned table view	(postgresql_epas_all_part_tables)				73
10.4.55	Oracle-compatible subpartitioning key view	(postgresql_epas_all_subpart_key_columns				
	)					73
10.4.56	Oracle-compatible all table partitions view	(postgresql_epas_all_tab_partitions)				74
10.4.57	Oracle-compatible all table subpartitioning view	(postgresql_epas_all_tab_subpartitions				
	)					74
10.4.58	EPAS specific dblink information	(postgresql_epas_dblink)				74
10.5	Barman (Backup and Recovery Manager)					74
10.5.1	Barman check	(barman_check)				74
10.5.2	Barman diagnose	(barman_diagnose)				75
10.5.3	Barman executable location	(barman_executable)				75
10.5.4	Barman module location	(barman_modules_path)				75

11 Servers accepting upload of reports 77

EDB has developed a small multi-platform application called Lasso that can safely gather relevant diagnostics data on a system where Postgres and other relevant supported software, such as Barman, is running.

You can also run Lasso on systems where Postgres isn't installed to gather all relevant information regarding the underlying operating system.

Consult the specific information for your operating system.

Important: Lasso can't affect your data. It gathers statistics and diagnostics information from your Postgres server, with imperceptible effects on the workload. *No actual data* from the rows of your Postgres user tables is gathered.¹

inspection.

¹ : Gathered diagnostics data files are available for your

CHAPTER 1

Release notes

Lasso - Version 4.11.0 (2023-12-14)

2.1 New features

- Implement EPAS and 2ndQPG flavour detection (DC-875)
- Gathering EPAS Catalog - Views related to partitioned tables in Oracle syntax (DC-266)
- Support for PostgreSQL 16 (DC-887)
- Add support for RHEL9 (DC-802, DC-803)

2.2 Bug fix

- Fix situation when `ls pg_data` command takes much time to complete (when number of files is extremely large). Adding a default timeout of 120s for this cases. (DC-845)

2.3 Lasso - Version 4.10.0 (2023-11-07)

2.3.1 New features

- Implement a new feature (filelock) to not rely on an EPEL dependency (DC-863)

2.3.2 Deprecation notes

- Deprecate Ubuntu Bionic (18.04) as it is EOL since June 2023 (DC-877)

2.4 Lasso - Version 4.9.0 (2023-10-17)

2.4.1 New features

- Include the limits for the postmaster process (DC-162)
- Add progress reporting views (DC-170)
- Add waits profiling via `pg_stat_activity` (DC-841)

2.4.2 Bug fix

- Fix issue related to the report filename when the hostname is a socket path (DC-769)
- Fix issue for the SET ROLE syntax (DC-842)
- Fix issue on Hostname from config file (DC-864)

2.5 Lasso - Version 4.8.0 (2023-07-06)

2.5.1 New features

- Add support for PGD Proxy:
- New option `--pgd-proxy-configuration` for custom installations of `pgd-proxy` (Default: `/etc/edb/pgd-proxy/pgd-proxy-config.yml`) (DC-696);
- Collect the contents of the PGD Proxy configuration file (DC-697);
- Collect the output of `systemctl status pgd-proxy` (DC-698).
- Collect the output of `systemctl cat pgd-proxy` (DC-699).

2.6 Lasso - Version 4.7.0 (2023-06-01)

2.6.1 New features

- Collect output of `systemd-detect-virt` (DC-76);
- Collect the output of `lsmod` (DC-197);
- Collect the content of `harp.cluster.init.yml` (DC-628).

2.6.2 Security fixes

- Redaction of password for Lasso with `--password` seen in `ps` and `top` outputs (DC-627).

2.6.3 Other bug fixes

- Fix a bug where it wasn't possible to run `lasso --help` without a configuration file (DC-490).

2.7 Lasso - Version 4.6.0 (2022-12-06)

2.7.1 New feature

- Lasso packages open to all customers through EnterpriseDB repository (<https://www.enterprisedb.com/repos>) (DC-443)

2.7.2 Improvements

- Updated the [main Lasso Knowledge Base article](#) to reflect the installation from packages and the configuration file (KB-54)

2.7.3 Bug fix

- Improve Debian/Ubuntu packages to comply with the Lintian (DC-435)

2.8 Lasso - Version 4.5.0 (2022-11-15)

2.8.1 Improvements

- Support for PostgreSQL 15 (DC-428)
- Improved execution on situations where there is a short timeout or when Lasso is executed on the standby (DC-169).

2.9 Lasso - Version 4.4.0 (2022-09-22)

2.9.1 Improvements

- Collect output of `pg_shmem_allocations` in PG $\geq$ 13 (DC-195)
- Document options for `edb-lasso.conf` (DC-308)

2.10 Lasso - Version 4.3.0 (2022-08-25)

2.10.1 Improvements

- Lasso support for etcd (DC-367, DC-371, DC-370, DC-369, DC-368)

2.10.2 Bug

- Gather databases and tablespaces sizes regardless of CONNECT privilege (DC-325)

2.11 Lasso - Version 4.2.0 (2022-05-18)

2.11.1 Improvements

- Collect systemd information for HARP (DC-331)
- Collect configuration files included in pgbouncer.ini (DC-322)
- Allow the user to specify custom paths to pgbouncer configuration files (DC-321)
- Gather etcdctl endpoint commands, if this is the consensus database (DC-309)
- Add an option to lasso to specify a different path for harp's config.yml (DC-306)
- Helper function to get harp configuration file (DC-305)
- Get harpctl outputs (DC-304)
- Get HARP config.yml (DC-303)
- Identify all the pgbouncer instances on the node (DC-157)
- Collect systemctl output for pgbouncer (DC-156)
- Collect pgbouncer.ini configuration file (DC-151)

2.11.2 Other changes

- Fix SLES smoke tests which are failing (DC-349)
- Add support for Ubuntu 22.04 (DC-284)

2.11.3 Bug

- Redact sslpassword (DC-348)

2.12 Lasso - Version 4.1.1 (2022-05-10)

2.12.1 Bug

- Remove the psycpg2-binary warning from the bundle execution

2.13 Lasso - Version 4.1.0 (2022-05-03)

2.13.1 Improvements

- Build PPC64le Lasso bundle (DC-276)
- Add .exe extension to Windows artifacts (DC-285)

2.13.2 Other changes

- Add smoke tests for Almalinux 8 (x86_64) (DC-294)
- Add smoke tests for RHEL (x86_64 and PPC64le) (DC-295)
- Add smoke tests for SLES 12 (x86_64 and PPC64le) (DC-296)
- Add smoke tests for SLES 15 (x86_64 and PPC64le) (DC-297)

2.13.3 Bug

- Lasso was unable to use MD5 hashing of tarball in FIPS enable systems (DC-310)

2.14 Lasso - Version 4.0.0 (2022-03-30)

2.14.1 Features

- Rebranding of Data Collector to Lasso, the diagnostic tool from EDB
- Update of Copyright notes (DC-239)
- Update license on all files (DC-228)
- Change the name of the bundle to reflect the new brand of the tool (DC-208)
- The generated output tarball now has the new brand and is identified with EDB (DC-49)
- The connection Lasso creates to the database now sets the `application_name` to `edb-lasso` to identify the process Lasso is running (DC-222)
- Update the documentation to reflect the new brand name (DC-184)

CHAPTER 3

Installing Lasso

EDB distributes the application through the [EDB Website](#) and grants usage to its customers using their *company token* as a means of authentication.

3.1 Linux

You can install Lasso on any major supported Linux distribution by following the corresponding Linux installation option for your system from the [EDB Website Downloads](#) , or directly from the [EDB repositories for Linux](#) page , choosing Lasso and following the installation instructions.

After installing the EDB repository for your subscription on your system, you will be able to install Lasso on Linux using the package manager tool for your Linux distribution, for example, including but not limited to:

- Debian / Ubuntu:

```
apt install edb-lasso
```

- RHEL / CentOS / Oracle Linux 7:

```
yum install edb-lasso
```

- RHEL / Rocky / AlmaLinux / Oracle Linux 8+:

```
dnf install edb-lasso
```

If your system has access to the Internet, then installing the `edb-lasso` package using your package manager will already automatically install the Lasso dependencies, which are:

- `python3`
- `python3-psycpg2`
- `python3-setuptools`

- python3-psutil

If your system does not have access to the Internet, then these dependencies need to be downloaded and installed manually. Consult EDB Support for more details.

3.2 Windows

Lasso for Windows is a single-file binary executable called `lasso-windows-X.Y.Z.exe`, where X.Y.Z is the current Lasso version, which can be downloaded from the [EDB Website Downloads page](#).

On Windows, Lasso doesn't require installing any dependencies, but Lasso can only run on Windows Server 2008 R2 or newer.

3.3 Configuration file

If you try to run Lasso without a configuration file, it will error out with:

```
ERROR: no configuration file for Lasso could be found. Please create a configuration_
↪file and try again.
```

Lasso requires a configuration file, which can be one of the following options (it uses the first match):

1. `./edb-lasso.cfg` in the same directory where Lasso is running
2. `./edb-lasso.conf` in the same directory where Lasso is running
3. `$HOME/.edb-lasso.conf`
4. `/etc/edb-lasso.conf`

The minimum configuration file should look like this:

```
[customer]
id=XXXXX
token=YYYYYYYYY
```

Replacing the `id` and `token` above with the information found in your company page in the Support Portal (in the left menu bar, click on `Company info` then click on `Company`). The `Company` code from the page should be filled in the `id` field in the configuration file, and the `Token` from the page should be filled in the `token` field in the configuration file.

Important: A configuration file for Lasso is mandatory, at least with the customer id and token. !!!

For more details about the Lasso configuration, see [Configuration](#).

3.4 Executing Lasso

After installing Lasso and creating an appropriate configuration file, then a standard Lasso execution on Linux consists of simply:

```
lasso
```

On Windows, in order to run Lasso, you need to do the following:

1. In the Start Menu, search for “cmd” ;
2. Right-click the “Command Prompt” and choose “Run as administrator” ;
3. If Windows asks whether you allow “Command Prompt” to do changes in the system, click “Yes” ;
4. Inside the “Command Prompt” , cd to the directory where the Lasso .exe file is located;
5. Execute the command “lasso-windows-X.Y.Z.exe” , where X.Y.Z is the Lasso version:

```
lasso-windows-X.Y.Z.exe -H IP_ADDRESS -p PORT --password PASSWORD DATABASE_NAME USER_NAME
```

Alternatively, if you fill the settings under the `postgresql` section in the configuration file telling Lasso how to connect to Postgres, then you can execute Lasso from the Windows explorer by right-clicking on the .exe file, then choosing `Run as administrator`.

CHAPTER 4

Usage

Lasso uses the `libpq` environment variables to get the Postgres connection parameters. You can find the list of the environment variables in the [PostgreSQL documentation](#).

The connection parameters, among other options, can also be passed as command line arguments:

```
usage: lasso [-h] [-H HOST_NAME] [-p PORT] [--password PASSWORD]
            [--lock-timeout LOCK_TIMEOUT]
            [--statement-timeout STATEMENT_TIMEOUT] [--bindir BINDIR]
            [--depth [{surface,shallow,deep}]]
            [--describe [{short,json,full}]] [--version] [--latest-version]
            [--system-only | --barman]
            [--barman-configuration BARMAN_CONFIGURATION]
            [--repmgr-configuration REPMGR_CONFIGURATION]
            [--efm-configuration EFM_CONFIGURATION]
            [--xdb-pubserver-configuration XDB_PUBSERVER_CONFIGURATION]
            [--xdb-subserver-configuration XDB_SUBSERVER_CONFIGURATION]
            [--pgbouncer-configuration PGBOUNCER_CONFIGURATION]
            [--harp-configuration HARP_CONFIGURATION]
            [--etcd-configuration ETCD_CONFIGURATION]
            [--pgd-proxy-configuration PGD_PROXY_CONFIGURATION]
            [--upload] [--keep-report] [--is-latest-version]
            [dbname] [user]
```

EDB Lasso

positional arguments:

dbname	Database name to connect to (default root)
user	Database user name (default root)

optional arguments:

-h, --help	show this help message and exit
-H HOST_NAME, --host-name HOST_NAME	

(continues on next page)

(continued from previous page)

```

Database host name or socket directory (default local
socket)
-p PORT, --port PORT Database server port (default 5432)
--password PASSWORD Database server password
--lock-timeout LOCK_TIMEOUT
Database connection lock timeout (default 3s)
--statement-timeout STATEMENT_TIMEOUT
Database connection statement timeout (default 5min)
--bindir BINDIR PostgreSQL binaries directory (autodetect by default)
--depth [{surface,shallow,deep}]
Depth of the report (default deep)
--describe [{short,json,full}]
Describes every single module, in terms of action and
output
--version Shows Lasso version and modules revision
--latest-version Shows the latest available version of EDB Lasso, taken
from the EDB Web Services
--system-only Gather only system-related information - without
requiring a PostgreSQL connection
--barman Gather Barman status, enabled by default when Lasso is
run as barman user and the executable exists.
Defaults to False. When Barman reporting is enabled,
we do not gather PostgreSQL related information
--barman-configuration BARMAN_CONFIGURATION
Barman configuration file. By default use the native
algorithm in Barman to find the configuration file.
Valid only if Barman reporting is enabled
--repmgr-configuration REPMGR_CONFIGURATION
Path to the repmgr.conf file, if using a non-default
path
--efm-configuration EFM_CONFIGURATION
Path to the EFM properties file, if using a non-
default path
--xdb-pubserver-configuration XDB_PUBSERVER_CONFIGURATION
Path to the xDB publication server configuration file,
if using a non-default path
--xdb-subserver-configuration XDB_SUBSERVER_CONFIGURATION
Path to the xDB subscription server configuration
file, if using a non-default path
--pgbouncer-configuration PGBOUNCER_CONFIGURATION
Path to the pgbouncer.ini file. You can specify multiple
files separated by comma
--harp-configuration HARP_CONFIGURATION
Path to the config.yml file, if using a non-default
path
--etcd-configuration ETCD_CONFIGURATION
Path to the etcd.conf file, if using a non-default
path
--pgd-proxy-configuration PGD_PROXY_CONFIGURATION
Path to the pgd-proxy-config.yml file, if using a
non-default path
--upload Report tarball file is sent to EDB at the end of the

```

(continues on next page)

(continued from previous page)

	execution. The file will be removed if successfully uploaded unless --keep-report is specified
--keep-report	Keep a local copy of the report even after a successful upload to EDB
--is-latest-version	Only check if this is the latest available version of Lasso and returns exit code 0 if this is the latest version and 1 otherwise

CHAPTER 5

Configuration

Besides being mandatory for the customer `id` and `token`, the Lasso configuration file also allows you to omit most of the command line options that your environment might require (you can see a list of all the command line options in Processor usage statistics (`linux_mpstat`) <Processor usage statistics (`linux_mpstat`)>`).

Lasso looks for configuration files in the following paths, in this order, and uses the first match:

1. `./edb-lasso.cfg` in the same directory where Lasso is running
2. `./edb-lasso.conf` in the same directory where Lasso is running
3. `$HOME/.edb-lasso.conf`
4. `/etc/edb-lasso.conf`

A template file for the configuration file looks like this:

```
; Lasso template configuration file
;
; Copyright (C) EnterpriseDB UK Limited 2015-2024 - All Rights Reserved.
; Licensed only for use with an EnterpriseDB subscription

[customer]
; the "Company code" from the customer page in the EDB Portal
id=
; the "Token" from the customer page in the EDB Portal
token=
; the depth of the Lasso report. Must be one between: surface, shallow and deep
depth=deep

[postgresql]
; Lasso uses the following connection string parameters to connect to your
; PostgreSQL cluster and take diagnostics. By default it will attempt a peer
; connection to socket under directory /var/run/postgresql
; dbname=
```

(continues on next page)

(continued from previous page)

```

; user=
; port=
; hostname can be a host or a socket directory
; hostname=/var/run/postgresql
; password=

; session variable to control timeout on lock waits
; lock_timeout=3s
; session variable to control statement execution time
; statement_timeout=5min

[environment]
; PostgreSQL binaries directory. Lasso will try to detect this automatically if
; not set
; bindir=
; if the host is able to access the internet and reach the EDB servers to upload
; Lasso reports
; external_access=yes

[barman]
; path to barman.conf, if using a non-default one
; configuration=

[repmgr]
; path to repmgr.conf, if using a non-default one
; configuration=

[efm]
; path to efm.properties, if using a non-default one
; configuration=

[xdb]
; path to xdb_pubserver.conf, if using a non-default one
; pubserver_configuration=
; path to xdb_subserver.conf, if using a non-default one
; subserver_configuration=

[pgbouncer]
; path to pgbouncer.ini, if using a non-default one
; configuration=

[harp]
; path to config.yml, if using a non-default one
; configuration=

[pgd-proxy]
; path to pgd-proxy-config.yml, if using a non-default one
; configuration=

```

You can use this template to set up your configuration file. Uncomment the desired parameters and set their values according to your environment.

On Linux, you can find that same template configuration file at `/etc/edb-lasso.conf.templ`.

Security considerations

When running queries in the database, Lasso tries to use a role that has enough privileges to gather the required information from the tool from which metrics are being gathered.

The following are the tools and the roles that Lasso tries to use for each of them. Lasso tries to use the first available role in each tool role list. *Initial connection role* means the role that was provided through Lasso CLI when running the tool—usually `postgres` or `enterprisedb`.

- PostgreSQL:
 - `pg_monitor`
 - initial connection role
- PgLogical:
 - `pglogical_superuser`
 - initial connection role
- PGD:
 - `bdr_monitor`
 - initial connection role
- PEM:
 - `pem_user`
 - initial connection role
- Repmgr: - initial connection role
- xDB: - initial connection role

Most of the PGD gatherings try using the `bdr_monitor` role. However, one of them, which is in charge of gathering conflicts, tries to use the role `bdr_read_all_conflicts` for that purpose. That's the only exception.

7.1 Local/Full PostgreSQL report

This type of report is useful when the server has a PostgreSQL cluster which is up, running and accessible. Besides all information that is gathered in the system only report, it will also gather configurations and metrics from the PostgreSQL instance.

For this you need to run Lasso as the `postgres` or `enterprisedb` user.

Important: The same user that runs the Postgres server process must run the command. The user must have permission to write in the current working directory. !!!

A standard Lasso run consists of:

```
lasso
```

By default, Lasso tries to connect using Unix sockets, the default Postgres port (5432), the default database and user name (`postgres`), but you can specify different values in the command line like this:

```
lasso -p PORT DATABASE_NAME USER_NAME
```

If a password is required, Lasso will ask for a password, but you can specify a password using the `PGPASSWORD` environment variable:

```
PGPASSWORD=xxxx lasso -p PORT DATABASE_NAME USER_NAME
```

By default Lasso looks for a Unix socket under `/var/run/postgresql/`, but you can specify any socket listed by `SHOW unix_socket_directories` using the `-H` argument. For example, usually a Lasso report for the EnterpriseDB Advanced Server is gathered with:

```
lasso -H /tmp -p 5444 edb
```

On Windows, typically you run Lasso like this:

```
lasso-windows-X.Y.Z.exe -H IP_ADDRESS -p PORT --password PASSWORD DATABASE_NAME USER_NAME
```

For a list of all command line arguments detailed in Processor usage statistics (`linux_mpstat`) <Processor usage statistics (`linux_mpstat`)>, run:

```
lasso --help
```

Postgres settings can be specified in the Lasso configuration file, under the `postgresql` setting. For more details see [Configuration](#).

Important: When run on a local instance, Lasso requires a user that belongs to the `pg_monitor` role, which is available in Postgres 10 and later. Before 10, you must run as a superuser. For more information, see [Default Roles](#) in the PostgreSQL documentation. !!!

Lasso transparently gathers information related to all supported versions of EDB Postgres Distributed (PGD) and pg-logical.

7.2 Remote/Database-only PostgreSQL report

If you need to gather data from a remote PostgreSQL instance, you can run a remote PostgreSQL report.

As explained in the previous section, you can change the connection string parameters of Lasso. For example, to gather a PostgreSQL report from a remote server located in the IP 192.168.0.10, running on port 5433, you could invoke Lasso from another server as below:

```
lasso -H 192.168.0.10 -p 5433
```

This type of report will bring only PostgreSQL related info from the PostgreSQL instance that Lasso was connected to. It means it will not be able to gather files like `postgresql.conf` and `pg_hba.conf`, nor will it be able to gather configurations and metrics from the underlying operational system or information from PostgreSQL related tools, like EFM configuration files. Having that in mind, it's always preferable running a local PostgreSQL report, if possible.

Important: This is the only option you have to gather PostgreSQL information from an instance hosted on a DBaaS provider such as EDB Big Animal or Amazon RDS. !!!

Important: This is the only option you have to gather PostgreSQL information from an instance running under Kubernetes. !!!

7.3 Barman report

Lasso can run on systems where Barman is installed. In that case, it gathers Barman-related information, as well as system information, that helps during analysis.

Important: You must run the command as the same user that runs the Barman process, and you must have enough permission to write in the current working directory. If you installed Barman using RPM/DEB packages, Barman is configured to run as the *barman* user. !!!

The Barman report is enabled by default if Lasso runs as the *barman* user and looks for configuration files in the expected locations, typically `/etc/barman.conf`. See `man 5 barman` for details.

In general, if you installed Barman using EDB-certified RPM and DEB packages, all you need to do is execute Lasso as the *barman* user.

7.3.1 Managing custom installations of Barman

If you have custom installations of Barman, you can enable the barman report by passing the `--barman` option to Lasso:

```
lasso --barman
```

You can also point to a specific global configuration file by using the `--barman-configuration` option as follows:

```
lasso --barman-configuration /opt/barman/barman.conf
```

7.4 Replication Manager (repmgr) report

Lasso can run on systems where repmgr is installed. In that case, it gathers repmgr-related information.

In general, if you installed repmgr using EDB-certified RPM and DEB packages, all you need to do is execute Lasso.

7.4.1 How Lasso finds the repmgr configuration file

Lasso uses the following approach while trying to identify the repmgr configuration file. It uses the first one it finds.

1. Use the one provided through the `--repmgr-configuration` option, if given.
2. Check the paths provided by packages. For example, in CentOS/RHEL packages, the configuration file is usually put under `/etc/repmgr/<PG_VERSION>/repmgr.conf`. Lasso inspects those folders, if they exist, in descending order, and uses the first repmgr configuration file that it finds.
3. Check if a `repmgr.conf` file exists in the current directory.
4. Check if a `/etc/repmgr.conf` file exists.
5. Check the path given by `pg_config --sysconfdir` for the `repmgr.conf` file.
6. Check the output of `systemctl cat <service_name>`, where `<service_name>` is any service whose name contains `repmgr`, and try to get the `repmgr.conf` file from the service unit file. It checks the units in descending order, just as it does in approach 2, and uses the first match.

7.4.2 Managing custom installations of repmgr

You can point to a specific repmgr configuration file by using the `--repmgr-configuration` option as follows:

```
lasso --repmgr-configuration /opt/repmgr/repmgr.conf
```

This approach is preferred, as it guarantees Lasso will use the correct repmgr configuration file instead of trying to find it automatically.

7.5 Postgres Enterprise Manager (PEM) report

Lasso can run on systems where PEM is installed. In that case, it also gathers PEM-related information, like some configuration files and information about services.

Lasso inspects the well-known paths for configuration files, as well as well-known service names. If you have custom PEM installations, Lasso doesn't gather the related information.

7.6 Failover Manager (EFM) report

Lasso can run on systems where EFM is installed. In that case, it gathers EFM-related information.

In general, if you installed EFM using EDB-certified RPM and DEB packages, all you need to do is execute Lasso.

7.6.1 How Lasso finds the EFM configuration file

Lasso uses the following approach while trying to identify the EFM configuration file. It uses the first one it finds.

1. Use the one provided through the `--efm-configuration` option, if given.
2. Check the paths provided by packages. The configuration file is usually put under `/etc/edb/efm-<EFM_VERSION>/efm.properties`. Lasso inspects those folders, if they exist, in descending order, and uses the first EFM configuration file that it finds.
3. Check the output of `systemctl cat <service_name>`, where `<service_name>` is replaced with any service whose name starts with `edb-efm-`. Then try to get the `efm.properties` file from the service unit file. It checks the units in descending order, just as it does in approach 2, and uses the first match.

7.6.2 How Lasso finds the EFM binary

Lasso inspects the paths provided by packages. The binary file is usually put under `/usr/edb/efm-<EFM_VERSION>/bin/efm`. Lasso inspects these folders, if they exist, in descending order, and uses the first EFM binary file that it finds.

7.6.3 Managing custom installations of EFM

You can point to a specific EFM configuration file by using the `--efm-configuration` option, as follows:

```
lasso --efm-configuration /opt/EFM/efm.properties
```

This approach is preferred, as it guarantees Lasso will use the correct EFM configuration file instead of trying to find it automatically.

7.7 Replication Server (xDB) report

Lasso can run on systems where xDB is installed. In that case, it gathers xDB-related information.

In general, if you installed xDB using EDB-certified RPM packages or the bitrock installer, all you need to do is execute Lasso.

7.7.1 How Lasso finds the xDB configuration file

Lasso uses the following approach while trying to identify the xDB configuration files. It uses the first one it finds.

On Linux:

1. Use the ones provided through the `--xdb-pubserver-configuration` and `--xdb-subserver-configuration` options, if given.
2. Check the paths provided by xDB 7 RPM package and bitrock installer. The configuration files are usually put under `/etc/edb/xdb/etc`. Lasso inspects this folder, if it exists, and uses the `xdb_pubserver.conf` and `xdb_subserver.conf` files found under that folder.
3. Check the paths provided by xDB 6 RPM package and bitrock installer, which are usually `/usr/ppas-xdb-<XDB_VERSION>/etc` and `/opt/PostgreSQL/EnterpriseDB-xDBReplicationServer/etc`, respectively. Lasso inspects these folders and, if they exist, uses the `xdb_pubserver.conf` and `xdb_subserver.conf` files found under the folder. In the case of RPM packages, as there can be a lot of folders, they're inspected from newest XDB version to oldest.

On Windows:

1. Use the ones provided through `--xdb-pubserver-configuration` and `--xdb-subserver-configuration` options, if given.
2. Check the paths provided by xDB 7 packages. The configuration files are usually put under `<drive>:\Program Files\edb\EnterpriseDB-xDBReplicationServer\etc`. Lasso inspects this folder, if it exists, and uses the `xdb_pubserver.conf` and `xdb_subserver.conf` files found under that folder.
3. Check the paths provided by xDB 6 packages. The configuration files are usually put under `<drive>:\Program Files\PostgreSQL\EnterpriseDB-xDBReplicationServer\etc`. Lasso inspects this folder, if it exists, and uses the `xdb_pubserver.conf` and `xdb_subserver.conf` files found under that folder.

7.7.2 How Lasso finds the xDB binary

Lasso uses the following approach while trying to identify the xDB binary file. It uses the first one it finds.

On Linux:

1. Check the paths provided by xDB 7 RPM package and bitrock installer. The binary files are usually put under `/etc/edb/xdm/bin`. Lasso inspects this folder, if it exists, and uses the `edb-repcli.jar` file found under that folder.
2. Check the paths provided by the xDB 6 RPM package and bitrock installer, which are usually `/usr/ppas-xdb-<XDB_VERSION>/bin` and `/opt/PostgreSQL/EnterpriseDB-xDBReplicationServer/bin`, respectively. Lasso inspects these folders and, if they exist, uses the `edb-repcli.jar` file found under the folder. In the case of RPM packages, as there can be a lot of folders, they're inspected from newest XDB version to oldest.

On Windows:

1. Check the paths provided by xDB 7 packages. The binary files are usually put under `<drive>:\Program Files\edb\EnterpriseDB-xDBReplicationServer\bin`. Lasso inspects this folder, if it exists, and uses the `edb-repcli.jar` file found under that folder.
2. Check the paths provided by xDB 6 packages. The binary files are usually put under `<drive>:\Program Files\PostgreSQL\EnterpriseDB-xDBReplicationServer\bin`. Lasso inspects this folder, if it exists, and uses the `edb-repcli.jar` file found under that folder.

7.7.3 Managing custom installations of xDB

You can point to a specific xDB publication or subscription server configuration file by using the `--xdb-pubserver-configuration` and `--xdb-subserver-configuration` options, as follows:

On Linux:

```
lasso --xdb-pubserver-configuration /opt/xdm/pubserver.conf --xdb-subserver-  
configuration /opt/xdm/subserver.conf
```

On Windows:

```
lasso-windows-X.Y.Z.exe --xdb-pubserver-configuration C:\xdm\pubserver.conf --xdb-  
subserver-configuration C:\xdm\subserver.conf
```

This approach is preferred, as it guarantees Lasso will use the correct xDB configuration files instead of trying to find it automatically.

7.8 PgBouncer report

Lasso can run on systems where PgBouncer is installed. In that case, it gathers PgBouncer-related information.

In general, if you installed PgBouncer using EDB-certified RPM and DEB packages, and PgBouncer instances are up and running, all you need to do is execute Lasso.

7.8.1 How Lasso finds the PgBouncer configuration file

Lasso uses the list of running processes to identify the PgBouncer processes and uses the configuration file retrieved from there.

You can also specify one or more PgBouncer configuration files through the Lasso argument `--pgbouncer-configuration`. It's mainly useful in scenarios where a PgBouncer instance isn't running or if Lasso can't detect the configuration file from the process list.

Lasso collects information from all identified PgBouncer instances.

7.9 HARP report

Lasso can run on systems where HARP is installed. In that case, it gathers HARP-related information.

In general, if you installed HARP using EDB-certified RPM and DEB packages, all you need to do is execute Lasso.

7.9.1 How Lasso finds the HARP configuration file

Lasso uses the following approach while trying to identify the HARP configuration file. It uses the first one it finds.

1. Use the one provided through the `--harp-configuration` option, if given.
2. Check if the `/etc/harp/config.yml` file exists.

7.9.2 Managing custom installations of HARP

You can point to a specific HARP configuration file by using the `--harp-configuration` option, as follows:

```
lasso --harp-configuration /opt/harp/config.yml
```

This approach is preferred, as it guarantees Lasso will use the correct HARP configuration file instead of trying to find it automatically.

7.9.3 Running Lasso on a HARP proxy node

The HARP proxy nodes don't have a Postgres or EDB Postgres Advanced Server service running on them, as they're intended only as a proxy for connections coming in from the applications.

When using Lasso to gather information from a HARP proxy node, you must use the `--system-only` option so it doesn't try to connect to the database server.

7.10 PGD Proxy report

Lasso can be run on systems where PGD Proxy is installed. In that case, it will gather PGD Proxy related information. In general, if you have installed PGD Proxy using EDB certified RPM and DEB packages, all you need to do is execute Lasso.

7.10.1 How Lasso finds the PGD Proxy configuration file

Lasso will use the below approach while trying to identify the PGD Proxy configuration file, and will use the first one which is found:

1. Use the one provided through `--pgd-proxy-configuration` option, if given
2. Check if `/etc/edb/pgd-proxy/pgd-proxy-config.yml` file exists

7.10.2 Managing custom installations of PGD Proxy

You can point to a specific PGD Proxy configuration file, by using the `--pgd-proxy-configuration` option, as follows:

```
lasso --pgd-proxy-configuration /opt/pgd-proxy/config.yml
```

This is actually the preferred approach, as it guarantees Lasso will use the correct PGD Proxy configuration file instead of trying to find it automatically.

7.10.3 Running Lasso on a PGD Proxy node

There are cases where the PGD Proxy nodes do not have a Postgres/EPAS service running on them, as they are only intended as a proxy for connections coming in from the applications.

When using Lasso to gather information from a standalone PGD Proxy node you will have to use the `--system-only` option so it doesn't try to connect to the database server.

7.11 etcd report

Lasso can run on systems where etcd is installed. In that case, it gathers etcd-related information.

7.11.1 How Lasso finds the etcd configuration file

Lasso uses the following approach while trying to identify the etcd configuration file. It uses the first one it finds.

1. Use the one provided through `--etcd-configuration` option, if given.
2. Check if the `/etc/etcd/etcd.conf` file exists.

7.12 System-only report

Lasso can also run on systems where Postgres or Barman aren't installed to gather all relevant information regarding the underlying operating system.

You can run a system-only report with the following command:

```
lasso --system-only
```

Returning the Lasso report

The script produces a *TAR file* in the same directory where you executed Lasso, containing the gathered data: this is the “Lasso report” .

You can also use Lasso on a server that has no Postgres installation. In that case, use the `--system-only` option, and the *TAR file* produced will contain only system-related information.

Unless you are running Lasso on an isolated network system, without external access to EnterpriseDB’ s infrastructure, you can upload the produced tarball directly through Lasso using the `--upload` option. For more information, see *Servers accepting upload of reports* .

What do we gather in our Lasso report?

A Lasso report is designed to allow any EDB engineer to rapidly feel the pulse of your database by getting relevant information from your operating system and your Postgres server.

You can get a list with a detailed description of the information that's gathered from your system through `lasso` by entering:

```
lasso --describe
```

If you need more information, contact your account manager at EDB.

Lasso informs its level of depth when scraping your system. There are three levels:

- **Surface:** Noninvasive gathering process, mainly used to perform a supportability check of the instance and focused on the current moment. In terms of Postgres, the focus is on the instance, its configuration, and its general health. This level is the minimal level of scraping.
- **Shallow:** Gathering of performance-related information, including basic trending data. In terms of Postgres data, its focus is moved down to the database. This level is suitable for support-incident resolution.

Detailed breakdown of gathered data

10.1 GNU/Linux Operating System

10.1.1 barman crontab/cron (barman_crontab_cron)

Output from `crontab -l`, if running as barman. Content of `/etc/cron.d/barman`, if exists.

Report output:

- file `/linux/barman_cron.data` : Content of `/etc/cron.d/barman`, if exists
- file `/linux/barman_crontab.data` : Output from barman `crontab -l`, if barman user

Depth: surface

Security Impact: *low* — May have entries in crontab/cron with sensitive data

10.1.2 debug_sources (debug_sources)

Count files under `/usr/src/debug` to detect which applications' source code is present in the system and facilitate live debugging.

Report output:

- file `/linux/debug_sources.data` : Sources for GNU debugger

Depth: surface

Security Impact: *low* — no known security impact.

10.1.3 EFM CLI (`efm_cli`)

Get output of `efm cluster-status` command.

Report output:

- file `/tools/efm/cli/cluster_status.out` : Output of `efm cluster-status cluster_name` command

Depth: surface

Security Impact: *low* — no known security impact.

10.1.4 EFM configuration (`efm_configuration`)

EFM properties and nodes configuration files

Report output:

- file `/tools/efm/config/efm.nodes` : EFM nodes file
- file `/tools/efm/config/efm.properties` : EFM properties file

Depth: surface

Security Impact: *low* — no known security impact.

10.1.5 EFM systemctl (`efm_systemctl`)

When EFM services are detected, collects status and cat of the corresponding services. Will check for any service which name starts with `edb-efm-`

Report output:

- file `/tools/efm/systemd/service_name_cat.data` : Output of `' systemctl cat service_name`
- file `/tools/efm/systemd/service_name_status.data` : Output of `' systemctl status service_name`

Depth: surface

Security Impact: *low* — no known security impact.

10.1.6 etcd CLI (`etcd_cli`)

Gathers the output of some `etcdctl` commands, if `etcdctl` is available in the server. The commands are: `endpoint status` and `endpoint health`.

Report output:

- file `/tools/etcd/cli/endpoint_status.out` : Output of `etcdctl endpoint status` command
- file `/tools/etcd/cli/endpoint_health.out` : Output of `etcdctl endpoint health` command

Depth: surface

Security Impact: *low* — no known security impact.

10.1.7 etcd configuration (etcd_configuration)

Collects etcd configuration file that is found in the server

Report output:

- file /tools/etcd/config/basename : etcd configuration file

Depth: surface

Security Impact: *low* — no known security impact.

10.1.8 etcd systemctl (etcd_systemctl)

When etcd services are detected, collects status and cat of the corresponding services. Will check for any service which name starts with etcd

Report output:

- file /tools/etcd/systemd/service_name_cat.data : Output of ' systemctl cat service_name
- file /tools/etcd/systemd/service_name_status.data : Output of ' systemctl status service_name

Depth: surface

Security Impact: *low* — no known security impact.

10.1.9 HARP CLI (harp_cli)

Gathers output of a few harpctl command outputs using the config.yml file which is found in the server. The commands are: cluster, proxies, locations, nodes, and version.

Report output:

- file /tools/harp/cli/version.out : Output of harp -f conf_file_path version command
- file /tools/harp/cli/proxies.out : Output of harp -f conf_file_path get proxies -o yaml command
- file /tools/harp/cli/nodes.out : Output of harp -f conf_file_path get nodes -o yaml command
- file /tools/harp/cli/locations.out : Output of harp -f conf_file_path get locations -o yaml command
- file /tools/harp/cli/cluster.out : Output of harp -f conf_file_path get cluster -o yaml command

Depth: surface

Security Impact: *low* — no known security impact.

10.1.10 HARP configuration (harp_configuration)

Collects HARP configuration file that is found in the server

Report output:

- file /tools/harp/config/harp.cluster.init.yml : HARP bootstrap configuration file
- file /tools/harp/config/basename : HARP configuration file

Depth: surface

Security Impact: *low* — no known security impact.

10.1.11 HARP systemctl (harp_systemctl)

When HARP services are detected, collects status and cat of the corresponding services. Will check for any service which name starts with harp

Report output:

- file /tools/harp/systemd/service_name_cat.data : Output of ' systemctl cat service_name
- file /tools/harp/systemd/service_name_status.data : Output of ' systemctl status service_name

Depth: surface

Security Impact: *low* — no known security impact.

10.1.12 Block devices layout (linux_block_devices_layout)

Information on block devices layout from the lsblk command

Report output:

- file /linux/lsbk.data : lsblk command output

Depth: surface

Security Impact: *low* — no known security impact.

10.1.13 Processor governor (linux_cpu_governor)

Processor scaling governor from the files in /sys/devices/system/cpu

Report output:

- file /linux/sys/energy_perf_bias.data : Intel Performance and Energy Bias attributes
- file /linux/sys/intel_pstate.data : Intel pstate configuration
- file /linux/sys/cpu_scaling_driver.data : available CPU scaling driver
- file /linux/sys/cpu_scaling_available_governors.data : available CPU scaling governors
- file /linux/sys/cpu_scaling_governor.data : active CPU scaling governor

Depth: surface

Security Impact: *low* — no known security impact.

10.1.14 Mounted file systems and available space (linux_devices_info)

List mounted filesystems through the `mount` command and free space using `df` .

Report output:

- file `/linux/diskspace.data` : amount of available disk space
- file `/linux/mount.data` : Output of the `mount` command

Depth: surface

Security Impact: *low* — no known security impact.

10.1.15 File systems configuration (linux_disk_configuration)

Disk configuration obtained through the `/etc/fstab` file

Report output:

- file `/linux/fstab.data` : contents of `/etc/fstab`

Depth: surface

Security Impact: *low* — no known security impact.

10.1.16 OS distribution, kernel and device data (linux_distro_collector)

Information about the Linux distribution currently in use, as returned by the `lsb_release` command.

Report output:

- file `/linux/release.data` : The Linux distribution currently in use
- file `/linux/release_source.data` : The name of the collected file or the executed command

Depth: surface

Security Impact: *low* — no known security impact.

10.1.17 Hardware (linux_hardware_info)

Hardware info through `lspci`

Report output:

- file `/linux/lspci.data` : Hardware info from `lspci`

Depth: surface

Security Impact: *low* — no known security impact.

10.1.18 Hypervisor (`linux_hypervisor_collector`)

Information about the type of virtualization used, as returned by the `systemd-detect-virt` command.

Report output:

- file `/linux/hypervisor.data` : The name of the collected file or the executed command

Depth: surface

Security Impact: *low* — no known security impact.

10.1.19 Kernel (`linux_kernel_info`)

Kernel info, transparent huge pages status and disk scheduler configuration obtained by combining the output of the commands `uname` and `ipcs` with the contents of the `/proc` and `/sys` filesystems

Report output:

- file `/linux/read_ahead.data` : Info on the read ahead
- file `/linux/schedulers.data` : Scheduler info from `/sys` dir
- file `/linux/sys/kernel/mm/transparent_hugepage.data` : Transparent huge pages info
- file `/linux/ipcs.data` : `ipcs` command output
- file `/linux/uname.data` : `uname` command output

Depth: surface

Security Impact: *low* — no known security impact.

10.1.20 Kernel limits (`linux_kernel_limits`)

Configuration file for the `pam_limits` module

Report output:

- file `/linux/limits.data` : content of the `limits.conf` file

Depth: surface

Security Impact: *low* — no known security impact.

10.1.21 Processor usage statistics (`linux_mpstat`)

Processor statistics from the `mpstat` command.

Report output:

- file `/linux/mpstat.data` : Output from `'mpstat -P ALL 1 10'`

Depth: surface

Security Impact: *low* — no known security impact.

10.1.22 Network interfaces (`linux_network_interfaces`)

Network interface information from the `ip` and `ifconfig` commands.

Report output:

- file `/linux/ifconfig.data` : Output from `'ifconfig'`
- file `/linux/ip_address_list.data` : Output from `'ip address list'`

Depth: surface

Security Impact: *low* — no known security impact.

10.1.23 Installed packages via rpm or dpkg (`linux_packages_info`)

Information about the system packages installed using `rpm` or `dpkg`

Report output:

- file `/linux/packages-dpkg.data` : List of packages installed using `dpkg`
- file `/linux/packages-rpm.data` : List of packages installed using `rpm`

Depth: surface

Security Impact: *low* — no known security impact.

10.1.24 PostgreSQL disk layout (`linux_postgresql_disk_layout`)

List all files in the PostgreSQL data directory using `find` for links and `ls` for files

Report output:

- file `/linux/pg_ls.data` : List of files inside the data directory
- file `/linux/pg_links.data` : List of links inside the data directory

Depth: surface

Security Impact: *low* — no known security impact.

10.1.25 SELinux (`linux_sestatus`)

SELinux status from `sestatus`

Report output:

- file `/linux/sestatus.data` : Output from `'setstatus'`

Depth: surface

Security Impact: *low* — no known security impact.

10.1.26 System identification (`linux_system_identity`)

Collect hostname, network interfaces, system info (uname), system identifier and release info.

Report output:

- file `/linux/id/system_release.data` : OS information from `/etc/system-release`
- file `/linux/id/os_release.data` : OS information from `/etc/os-release`
- file `/linux/id/machine_id.data` : Machine ID contained in `/etc/machine-id`
- file `/linux/id/uname.data` : Information about the running kernel
- file `/linux/id/hostname.data` : Fully qualified domain name
- file `/linux/id/interfaces.data` : Network addresses of the host

Depth: surface

Security Impact: *low* — no known security impact.

10.1.27 dmesg and /proc information (`linux_system_info`)

System info from the contents of the `/proc` filesystem and through the output of `dmesg` command.

Report output:

- file `/linux/lsmmod.data` : Lsmmod output
- file `/linux/dmesg_with_timestamp.data` : Dmesg output (human readable timestamps)
- file `/linux/dmesg.data` : Dmesg output
- file `/linux/proc/sys_net_ipv4.data` : Network info from `/proc`
- file `/linux/proc/sys_vm.data` : vm info from `/proc`
- file `/linux/proc/sys_kernel.data` : Kernel info from `/proc`
- file `/linux/vmstat.data` : VM statistics from `/proc`
- file `/linux/proc/mounts.data` : Mount points from `/proc`
- file `/linux/proc/uptime.data` : Uptime info from `/proc`
- file `/linux/proc/loadavg.data` : Load avg from `/proc`
- file `/linux/proc/meminfo.data` : Memory info from `/proc`

Depth: surface

Security Impact: *low* — no known security impact.

10.1.28 System status — iostat (linux_system_status_iostat)

System status from the `iotstat` command.

Report output:

- file `/linux/iotstat.data` : Info on I/O statistics

Depth: surface

Security Impact: *low* — no known security impact.

10.1.29 System status — nfsiostat (linux_system_status_nfsiostat)

System status from the `nfsiostat` command.

Report output:

- file `/linux/nfsiostat.data` : nfs I/O statistics

Depth: surface

Security Impact: *low* — no known security impact.

10.1.30 System status — ps (linux_system_status_ps)

System status from the `ps` command.

Report output:

- file `/linux/ps.data` : Active processes info

Depth: surface

Security Impact: *low* — Some processes might contain sensitive data in their names

10.1.31 System status — sar (linux_system_status_sar)

System status from the `sar` command.

Report output:

- file `/linux/sar.data` : Actual sar info
- file `/linux/sar-yesterday.data` : sar info from yesterday

Depth: surface

Security Impact: *low* — no known security impact.

10.1.32 System status — top (linux_system_status_top)

System status from the `top` command.

Report output:

- file `/linux/top.data` : Process information

Depth: surface

Security Impact: *low* — Some processes might contain sensitive data in their names

10.1.33 System status — vmstat (linux_system_status_vmstat)

System status from the `vmstat` command.

Report output:

- file `/linux/vmstat.data` : Info on processes, memory, paging, block IO, traps, disks and cpu activity

Depth: surface

Security Impact: *low* — no known security impact.

10.1.34 systemctl units (linux_systemctl_units)

Systemctl list-units on a `systemd` server

Report output:

- file `/linux/systemctl-list-units.data` : Output of ‘`systemctl list-units`’

Depth: surface

Security Impact: *low* — no known security impact.

10.1.35 tuned (linux_tuned)

Tuned status and profiles

Report output:

- directory `/linux/tuned/tune-profiles` : Files from `/etc/tune-profiles`
- directory `/linux/tuned/tuned` : Files from `/etc/tuned`
- file `/linux/tuned/tuned.conf` : The file `/etc/tuned.conf`
- file `/linux/tuned/tuned-list.data` : Output from `tuned_adm list`
- file `/linux/tuned/tuned-active.data` : Output from `tuned_adm active`

Depth: surface

Security Impact: *low* — no known security impact.

10.1.36 PEM configuration (pem_configuration)

PEM configuration files from PEM agent, PEM server, and PEM Web server

Report output:

- file /tools/pem/config/edb-ssl-pem.conf : PEM Web server SSL configuration file
- file /tools/pem/config/edb-pem.conf : PEM Web server configuration file
- file /tools/pem/config/install-config : PEM server configuration file (installation config file)
- file /tools/pem/config/config_setup.py : PEM server setup configuration file
- file /tools/pem/config/pem.wsgi : PEM server WSGI definition file
- file /tools/pem/config/agent.cfg : PEM agent configuration file

Depth: surface

Security Impact: *low* — no known security impact.

10.1.37 PEM systemctl (pem_systemctl)

When PEM is detected, collects PEM agent and PEM web server status and content

Report output:

- file /tools/pem/systemd/service_name_cat.data : Output of ' systemctl cat service_name
- file /tools/pem/systemd/service_name_status.data : Output of ' systemctl status service_name

Depth: surface

Security Impact: *low* — no known security impact.

10.1.38 PgBouncer configuration (pgbouncer_configuration)

PgBouncer configuration files

Report output:

- file /tools/pgbouncer/num/config/basename : PgBouncer configuration file from instance num

Depth: surface

Security Impact: *low* — no known security impact.

10.1.39 PgBouncer systemctl (pgbouncer_systemctl)

When PgBouncer services are detected, collects status and cat of the corresponding services. Will check for any service which contains any of the PgBouncer configuration files

Report output:

- file /tools/pgbouncer/num/systemd/service_name_cat.data : Output of ' systemctl cat service_name from instance num
- file /tools/pgbouncer/num/systemd/service_name_status.data : Output of ' systemctl status service_name from instance num

Depth: surface

Security Impact: *low* — no known security impact.

10.1.40 PGD Proxy configuration (pgd_proxy_configuration)

Collects PGD Proxy configuration file that is found in the server

Report output:

- file /tools/pgd-proxy/config/basename : PGD Proxy configuration file

Depth: surface

Security Impact: *low* — no known security impact.

10.1.41 PGD Proxy systemctl (pgd_proxy_systemctl)

When PGD Proxy services are detected, collects status and cat of the corresponding services. Will check for any service which name starts with pgd-proxy

Report output:

- file /tools/pgd-proxy/systemd/service_name_cat.data : Output of ' systemctl cat service_name
- file /tools/pgd-proxy/systemd/service_name_status.data : Output of ' systemctl status service_name

Depth: surface

Security Impact: *low* — no known security impact.

10.1.42 postgres/enterprisedb crontab (postgres_enterprisedb_crontab)

Output from crontab -l, if running as postgres or enterprisedb

Report output:

- file /linux/enterprisedb_crontab.data : Output from enterprisedb crontab -l, if enterprisedb user
- file /linux/postgres_crontab.data : Output from postgres crontab -l, if postgres user

Depth: surface

Security Impact: *low* — May have entries in crontab/cron with sensitive data

10.1.43 repmgr CLI (repmgr_cli)

Collects output of repmgr cluster crosscheck and repmgr daemon status using the repmgr.conf file which is found in the server.

Report output:

- file /tools/repmgr/cli/daemon_status.out : Output of repmgr daemon status -f conf_file_path command
- file /tools/repmgr/cli/cluster_crosscheck.out : Output of repmgr cluster crosscheck -f conf_file_path command

Depth: surface

Security Impact: *low* — no known security impact.

10.1.44 repmgr configuration (repmgr_configuration)

Collects repmgr configuration file that is found in the server

Report output:

- file /tools/repmgr/config/repmgr.conf : repmgr configuration file

Depth: surface

Security Impact: *low* — no known security impact.

10.1.45 repmgr systemctl (repmgr_systemctl)

When repmgr services are detected, collects status and cat of the corresponding services. Will check for any service which name starts with repmgr

Report output:

- file /tools/repmgr/systemd/service_name_cat.data : Output of ' systemctl cat service_name
- file /tools/repmgr/systemd/service_name_status.data : Output of ' systemctl status service_name

Depth: surface

Security Impact: *low* — no known security impact.

10.1.46 xDB CLI (xdb_cli)

xDB output from several CLI commands, from the xDB publication and/or subscription server that are running.

Report output:

- directory /tools/xdb/cli : xDB CLI print commands

Depth: surface

Security Impact: *low* — no known security impact.

10.1.47 xDB configuration (xdb_configuration)

xDB configuration files

Report output:

- file /tools/xdb/config/xdbReplicationServer.config : xDB startup configuration`
- file /tools/xdb/config/edb-repl.conf : xDB replication configuration`
- file /tools/xdb/config/xdb_subserver.conf : xDB subscription server configuration
- file /tools/xdb/config/xdb_pubserver.conf : xDB publication server configuration

Depth: surface

Security Impact: *low* — no known security impact.

10.1.48 xDB systemctl (xdb_systemctl)

When xDB services are detected, collects status and cat of edb- xdbpubserver and edb-xdbsubserver

Report output:

- file /tools/xdb/systemd/service_name_cat.data : Output of ' systemctl cat service_name
- file /tools/xdb/systemd/service_name_status.data : Output of ' systemctl status service_name

Depth: surface

Security Impact: *low* — no known security impact.

10.2 Microsoft Windows Operating System

10.2.1 PEM configuration — Windows (pem_configuration_windows)

PEM configuration files from PEM agent, PEM server, and PEM Web server in a Windows environment.

Report output:

- file /tools/pem/config/edb-ssl-pem.conf : PEM Web server SSL configuration file
- file /tools/pem/config/edb-pem.conf : PEM Web server configuration file
- file /tools/pem/config/pem.wsgi : PEM server WSGI definition file
- file /tools/pem/config/agent.cfg : PEM agent configuration file

Depth: surface

Security Impact: *low* — no known security impact.

10.2.2 PEM sc (pem_sc)

When PEM is detected, collects PEM agent and PEM web server status and content

Report output:

- file /tools/pem/sc/service_name_query.data : Output of ' sc query service_name

Depth: surface

Security Impact: *low* — no known security impact.

10.2.3 Disk information (win_disk_information)

Disk and controller information from the system registry.

Report output:

- file /windows/enum_ide.reg : Local machine ide device settings
- file /windows/enum_scsi.reg : Local machine scsi device settings

Depth: surface

Security Impact: *low* — no known security impact.

10.2.4 Hosts file (win_hosts)

Host files and network related information

Report output:

- file /windows/services.data : Windows services file
- file /windows/protocol.data : Windows protocol file
- file /windows/networks.data : Windows networks file
- file /windows/hosts.sam : Windows hosts.sam file
- file /windows/hosts.data : Windows hosts file

Depth: surface

Security Impact: *low* — no known security impact.

10.2.5 MsInfo (win_msinfo)

MsInfo32 report in NFO and TXT format

Report output:

- file /windows/msinfo_report.txt : Information from the MsInfo32 in textual format
- file /windows/msinfo_report.nfo : Information from the MsInfo32 in NFO

Depth: surface

Security Impact: *low* — no known security impact.

10.2.6 ODBC/64 (win_odbc32_info)

ODBC configuration from the 64 bit registry section

Report output:

- file /windows/user_odbc_wow64.reg : User DSN list
- file /windows/localmachine_odbcinst_wow64.reg : List of installed ODBC drivers
- file /windows/localmachine_odbc_wow64.reg : System DSN list

Depth: surface

Security Impact: *medium* — ODBC connection information could expose the presence of other databases or connection information to PostgreSQL that can be used to attack the system

10.2.7 ODBC/32 (win_odbc64_info)

ODBC configuration from the 32 bit registry section

Report output:

- file /windows/user_odbc.reg : User DSN list
- file /windows/localmachine_odbcinst.reg : list of installed ODBC drivers
- file /windows/localmachine_odbc.reg : System DSN list

Depth: surface

Security Impact: *medium* — ODBC connection information could expose the presence of other databases or connection information to PostgreSQL that can be used to attack the system

10.2.8 systeminfo (win_systeminfo)

Output of the systeminfo command

Report output:

- file /windows/systeminfo_report.txt : Information from the systeminfo command

Depth: surface

Security Impact: *low* — no known security impact.

10.2.9 Disk volumes (win_volumes)

Volume list from WMI

Report output:

- file /windows/association_structure : Association between drive letters and physical drives
- file /windows/volume_disk : Volume list from the WMI subsystem
- file /windows/logical_disk_list : Logical disk list from the WMI subsystem
- file /windows/disk_partition_list : Disk partition list from the WMI subsystem
- file /windows/disk_drive_list : Disk list from the WMI subsystem

Depth: surface

Security Impact: *low* — no known security impact.

10.2.10 xDB CLI — Windows (xdb_cli_windows)

xDB output from several CLI commands, from the xDB publication and/or subscription server that are running.

Report output:

- directory /tools/xdb/cli : xDB CLI print commands

Depth: surface

Security Impact: *low* — no known security impact.

10.2.11 xDB configuration — Windows (xdb_configuration_windows)

xDB configuration files

Report output:

- file /tools/xdb/config/xdbReplicationServer.config : xDB startup configuration`
- file /tools/xdb/config/edb-repl.conf : xDB replication configuration`
- file /tools/xdb/config/xdb_subserver.conf : xDB subscription server configuration
- file /tools/xdb/config/xdb_pubserver.conf : xDB publication server configuration

Depth: surface

Security Impact: *low* — no known security impact.

10.2.12 xDB sc (xdb_sc)

When xDB is detected, collects xDB Publication and Subscription server status

Report output:

- file /tools/xdb/sc/service_name_query.data : Output of ' sc query service_name

Depth: surface

Security Impact: *low* — no known security impact.

10.3 PostgreSQL/BDR3 Instance

10.3.1 Current archiver stats (postgresql_archiver)

Statistics about the archiver process' s activity (from pg_stat_archiver)

Report output:

- file postgresql/archiver.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.2 Available extensions (postgresql_available_extensions)

List of extensions available on the server

Report output:

- file postgresql/available_extensions.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.3 Current bg_writer stats (postgresql_bgwriter)

Statistics about the background writer process' s activity (from pg_stat_bgwriter)

Report output:

- file postgresql/bgwriter.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.4 Directory with binaries (postgresql_bin_dir)

PostgreSQL binary directory

Report output:

- file /postgresql/postgresql_bin_path.data : Path to the PostgreSQL bin directory

Depth: surface

Security Impact: *low* — no known security impact.

10.3.5 Current configuration (postgresql_configuration)

PostgreSQL current configuration

Report output:

- file postgresql/configuration.out

Depth: surface

Security Impact: *medium* — postgresql.conf might contain bad security policies

10.3.6 Configuration files (postgresql_configuration_files)

PostgreSQL configuration files and the data directory path (passwords contained in well-known connection strings are automatically redacted for information security reasons).

Report output:

- file /postgresql/pg_ident.conf : PostgreSQL ident configuration file
- file /postgresql/pg_hba.conf : PostgreSQL host-based authentication file
- file /postgresql/postgresql.auto.conf : PostgreSQL auto configuration file
- file /postgresql/recovery.done : PostgreSQL recovery.done file
- file /postgresql/recovery.conf : PostgreSQL recovery.conf file
- file /postgresql/postgresql.conf : PostgreSQL configuration file

Depth: surface

Security Impact: *medium* — pg_hba.conf and pg_ident.conf might expose potential security holes such as trusted connections.

10.3.7 Databases (postgresql_databases)

List of databases in the PostgreSQL node

Report output:

- file postgresql/databases.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.8 postgresql_db_bdr_tables_and_views (postgresql_db_bdr_tables_and_views)

Collect all the tables and views of the BDR extension, except for:

- bdr.apply_log — bdr.conflict_history — bdr.consensus_kv_data
- bdr.internal_node_pre_commit — bdr.replication_status - bdr.state_journal — bdr.stat_activity

Report output:

- file /postgresql/dbs/dbname/bdr/* : Content of all tables under the BDR schema

Depth: shallow

Security Impact: *low* — no known security impact.

10.3.9 postgresql_db_pglogical_tables_and_views (postgresql_db_pglogical_tables_and_views)

Collect all the tables and views of the pglogical extension

Report output:

- file /postgresql/dbs/dbname/pglogical/* : Content of all tables under the pglogical schema

Depth: shallow

Security Impact: *low* — no known security impact.

10.3.10 Database/Role Setting (postgresql_db_role_setting)

List of database/role settings in the PostgreSQL node

Report output:

- file postgresql/db_role_setting.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.3.11 Node and snapshot data (postgresql_node)

Information about the running PostgreSQL node

Report output:

- file postgresql/node.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.12 pg_config (postgresql_pg_config)

PostgreSQL pg_config command output.

Report output:

- file /postgresql/pg_config.data : pg_config command output

Depth: surface

Security Impact: *low* — no known security impact.

10.3.13 pg_controldata (postgresql_pg_controldata)

PostgreSQL pg_controldata information.

Report output:

- file /postgresql/pg_controldata.data : pg_controldata command output

Depth: surface

Security Impact: *low* — no known security impact.

10.3.14 Version (postgresql_pg_version)

PostgreSQL client and server version.

Report output:

- file /postgresql/postgresql_server_version.data : PostgreSQL server version
- file /postgresql/postgresql_client_version.data : PostgreSQL client version

Depth: surface

Security Impact: *low* — no known security impact.

10.3.15 Current pg_prepared_xacts contents (postgresql_prepared_xacts)

Status of prepared xacts (from pg_prepared_xacts)

Report output:

- file postgresql/prepared_xacts.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.16 Current pg_replication_origin_status contents (postgresql_replication_origin)

Status of replication origins (from pg_replication_origin_status)

Report output:

- file postgresql/replication_origins.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.17 Current pg_replication_slots contents (postgresql_replication_slots)

Replication slots (from pg_replication_slots)

Report output:

- file postgresql/replication_slots.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.18 Roles (postgresql_roles)

Database roles from pg_roles

Report output:

- file postgresql/roles.out

Depth: shallow

Security Impact: *medium* — pg_roles might contain bad security policies

10.3.19 Current activity stats (postgresql_running_activity)

Information related to the current activity on running processes (from pg_stat_activity)

Report output:

- file postgresql/running_activity.out

Depth: shallow

Security Impact: *low* — queries in pg_stat_activity could contain user names and application names.

10.3.20 Active locks (postgresql_running_locks)

List of active locks

Report output:

- file postgresql/running_locks.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.21 pg_server_limits (postgresql_server_limits)

Real effective kernel OS limits for the postmaster PID.

Report output:

- file /postgresql/pg_server_limits_PORT.data : prlimit for postmaster PID

Depth: surface

Security Impact: *low* — no known security impact.

10.3.22 Current pg_shmem_allocations contents (postgresql_shmem_allocations)

Status of shared memory allocations (from pg_shmem_allocations)

Report output:

- file postgresql/shmem_allocations.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.23 Current pg_stat_progress_analyze contents (postgresql_stat_progress_analyze)

ANALYZE progress

Report output:

- file postgresql/pg_stat_progress_analyze.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.24 Current pg_stat_progress_basebackup contents (postgresql_stat_progress_basebackup)

BASEBACKUP progress

Report output:

- file postgresql/pg_stat_progress_basebackup.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.25 Current pg_stat_progress_copy contents (postgresql_stat_progress_copy)

COPY progress

Report output:

- file postgresql/pg_stat_progress_copy.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.26 Current pg_stat_progress_vacuum contents (postgresql_stat_progress_vacuum)

VACUUM progress

Report output:

- file postgresql/pg_stat_progress_vacuum.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.27 Current pg_stat_replication contents (postgresql_stat_replication)

Replication connections (from pg_stat_replication)

Report output:

- file postgresql/replication.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.28 Server subscription statistics (postgresql_subscription_statistics)

Statistics of subscriptions

Report output:

- file postgresql/subscription_statistics.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.3.29 Server subscriptions (postgresql_subscriptions)

List of subscriptions

Report output:

- file postgresql/subscriptions.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.3.30 Tablespaces (postgresql_tablespaces)

Tablespaces information and location

Report output:

- file postgresql/tablespaces.out

Depth: surface

Security Impact: *low* — no known security impact.

10.3.31 Workload characteristics using waits (postgresql_waits_stats)

PostgreSQL workload characterisation using built-in wait events

Report output:

- file postgresql/running_waits_sample.out : Workload characterisation using built-in wait events

Depth: surface

Security Impact: *low* — no known security impact.

10.4 Details for every PostgreSQL/BDR3 Database

10.4.1 BDR1 replication slots (postgresql_db_bdr1_replication_slots)

List of replication slots with 9.6 format for BDR1

Report output:

- file bdr1_replication_slots.out

Depth: surface

Security Impact: *low* — no known security impact.

10.4.2 BDR2 replication slots (postgresql_db_bdr2_replication_slots)

List of replication slots with 9.6 format for BDR2

Report output:

- file bdr2_replication_slots.out

Depth: surface

Security Impact: *low* — no known security impact.

10.4.3 BDR conflict_history_summary aggregation (postgresql_db_bdr3_conflict_history_summary)

Collect aggregate count for all types of conflicts

Report output:

- file bdr_conflict_history_summary_agg.out

Depth: surface

Security Impact: *low* — no known security impact.

10.4.4 BDR current activity stats (postgresql_db_bdr3_stat_activity)

Information related to the current activity on running processes (from bdr.stat_activity)

Report output:

- file bdr_stat_activity.out

Depth: shallow

Security Impact: *low* — queries in bdr.stat_activity could contain user names and application names.

10.4.5 BDR sequences (postgresql_db_bdr_sequences)

List of the BDR sequences

Report output:

- file bdr_sequences.out

Depth: surface

Security Impact: *low* — no known security impact.

10.4.6 BDR version (postgresql_db_bdr_version)

Currently used version of BDR

Report output:

- file bdr_version.out

Depth: surface

Security Impact: *low* — no known security impact.

10.4.7 Database extensions (postgresql_db_extensions)

List of extensions in the database

Report output:

- file extensions.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.8 Database indexes (postgresql_db_indexes)

List of indexes in the database

Report output:

- file indexes.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.9 Database procedural languages (postgresql_db_languages)

Procedural languages in the database

Report output:

- file language.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.10 BDR monitor_group_raft (postgresql_db_monitor_group_raft)

Check the raft status in the bdr cluster

Report output:

- file bdr_monitor_group_raft.out

Depth: surface

Security Impact: *low* — no known security impact.

10.4.11 BDR monitor_group_versions (postgresql_db_monitor_group_versions)

Check the version of all BDR nodes

Report output:

- file bdr_monitor_group_versions.out

Depth: surface

Security Impact: *low* — no known security impact.

10.4.12 BDR monitor_local_replslots (postgresql_db_monitor_local_replslots)

Check all the replication slot status

Report output:

- file bdr_monitor_local_replslots.out

Depth: surface

Security Impact: *low* — no known security impact.

10.4.13 Database operators (postgresql_db_operators)

Operators in the database

Report output:

- file operator.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.14 Database partitioned tables (postgresql_db_partitioned_tables)

Information about partitioned tables (using declarative partitioning)

Report output:

- file partitioned_table.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.15 Database PEM pem.agent table tuples (postgresql_db_pem_agent)

Data from metatable pem.agent

Report output:

- file pem_agent.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.16 Database PEM pem.agent_config table tuples (postgresql_db_pem_agent_config)

Data from metatable pem.agent_config

Report output:

- file pem_agent_config.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.17 Database PEM pem.agent_heartbeat table tuples (postgresql_db_pem_agent_heartbeat)

Data from metatable pem.agent_heartbeat

Report output:

- file pem_agent_heartbeat.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.18 Database PEM pem.agent_server_binding table tuples (postgresql_db_pem_agent_server_binding)

Data from metatable pem.agent_server_binding

Report output:

- file pem_agent_server_binding.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.19 Database PEM pem.config table tuples (postgresql_db_pem_config)

Data from metatable pem.config

Report output:

- file pem_config.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.20 Database PEM pem.email_group table tuples (postgresql_db_pem_email_group)

Data from metatable pem.email_group

Report output:

- file pem_email_group.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.21 Database PEM pem.email_group_option table tuples (postgresql_db_pem_email_group_option)

Data from metatable pem.email_group_option

Report output:

- file pem_email_group_option.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.22 Database PEM pem.probe table tuples (postgresql_db_pem_probe)

Data from metatable pem.probe

Report output:

- file pem_probe.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.23 Database PEM pem.probe_schedule table tuples (postgresql_db_pem_probe_schedule)

Data from metatable pem.probe_schedule

Report output:

- file pem_probe_schedule.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.24 Database PEM pem.schema_version() output (postgresql_db_pem_schema_version)

Output from function pem.schema_version()

Report output:

- file pem_schema_version.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.25 Database PEM pem.server table tuples (postgresql_db_pem_server)

Data from metatable pem.server

Report output:

- file pem_server.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.26 Database PEM pem.server_heartbeat table tuples (postgresql_db_pem_server_heartbeat)

Data from metatable pem.server_heartbeat

Report output:

- file pem_server_heartbeat.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.27 Database PEM pem.smtp_spool table tuples (postgresql_db_pem_smtp_spool)

Data from metatable pem.smtp_spool

Report output:

- file pem_smtp_spool.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.28 Database PEM pem.snmp_spool table tuples (postgresql_db_pem_snmp_spool)

Data from metatable pem.snmp_spool

Report output:

- file pem_snmp_spool.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.29 Pglogical subscription status (postgresql_db_pglogical_subscription_status)

List of tables replicated by pglogical

Report output:

- file pglogical_subscription_status.out

Depth: surface

Security Impact: *low* — no known security impact.

10.4.30 Database functions (postgresql_db_procs)

Functions in the database

Report output:

- file proc.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.31 Database publication tables (postgresql_db_publication_tables)

List of tables of publications of the database

Report output:

- file publication_tables.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.32 Database publications (postgresql_db_publications)

List of publications of the database

Report output:

- file publications.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.33 Database repmgr repmgr.events table tuples (postgresql_db_repmgr_events)

Data from metatable repmgr.events

Report output:

- file repmgr/events.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.34 Database repmgr repmgr.monitoring_history table tuples (postgresql_db_repmgr_monitoring_history)

Data from metatable repmgr.monitoring_history

Report output:

- file repmgr/monitoring_history.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.35 Database repmgr repmgr.nodes table tuples (postgresql_db_repmgr_nodes)

Data from metatable repmgr.nodes

Report output:

- file repmgr/nodes.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.36 Database `repmgr` `repmgr.replication_status` `table` `tuples` (`postgresql_db_repmgr_replication_status`)

Data from metatable `repmgr.replication_status`

Report output:

- file `repmgr/replication_status.out`

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.37 Database `repmgr` `repmgr.show_nodes` `table` `tuples` (`postgresql_db_repmgr_show_nodes`)

Data from metatable `repmgr.show_nodes`

Report output:

- file `repmgr/show_nodes.out`

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.38 Database schema (`postgresql_db_schemas`)

List of schemas in the database

Report output:

- file `schemas.out`

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.39 Database statistics (`postgresql_db_statistics`)

Statistics of the database

Report output:

- file `statistics.out`

Depth: deep

Security Impact: *low* — no known security impact.

10.4.40 Database subscription tables (postgresql_db_subscription_tables)

List of tables of subscriptions of the database

Report output:

- file subscription_tables.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.41 Database tables (postgresql_db_tables)

List of tables in the database

Report output:

- file tables.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.42 Database types (postgresql_db_types)

Types in the database

Report output:

- file type.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.43 Database xDB _edb_replicator_pub.xdb_mmr_pub_group table tuples (postgresql_db_xdb_mmr_pub_group)

Data from metatable _edb_replicator_pub.xdb_mmr_pub_group

Report output:

- file xdb_mmr_pub_group.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.44 Database xDB _edb_replicator_pub.xdb_pub_database table tuples (postgresql_db_xdb_pub_database)

Data from metatable _edb_replicator_pub.xdb_pub_database

Report output:

- file xdb_pub_database.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.45 Database xDB _edb_replicator_pub.xdb_pub_relog table tuples (postgresql_db_xdb_pub_relog)

Last 50 rows from metatable _edb_replicator_pub.xdb_pub_relog

Report output:

- file xdb_pub_relog.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.46 Database xDB _edb_replicator_pub.xdb_publication_subscriptions table tuples (postgresql_db_xdb_publication_subscriptions)

Data from metatable _edb_replicator_pub.xdb_publication_subscriptions

Report output:

- file xdb_publication_subscriptions.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.47 Database xDB _edb_replicator_pub.xdb_publications table tuples (postgresql_db_xdb_publications)

Data from metatable _edb_replicator_pub.xdb_publications

Report output:

- file xdb_publications.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.48 Database xDB _edb_replicator_pub.rrep_mmr_pub_group table tuples (postgresql_db_xdb_rrep_mmr_pub_group)

Data from metatable _edb_replicator_pub.rrep_mmr_pub_group

Report output:

- file xdb_rrep_mmr_pub_group.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.49 Database xDB _edb_replicator_pub.rrep_mmr_txset table tuples (postgresql_db_xdb_rrep_mmr_txset)

Last 10 rows from metatable _edb_replicator_pub.rrep_mmr_txset

Report output:

- file xdb_rrep_mmr_txset.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.50 Database xDB _edb_replicator_pub.rrep_properties table tuples (postgresql_db_xdb_rrep_properties)

Data from from metatable _edb_replicator_pub.rrep_properties

Report output:

- file xdb_rrep_properties.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.51 Database xDB _edb_replicator_pub.rrep_publication_tablestable tuples (postgresql_db_xdb_rrep_publication_tables)

Data from from metatable _edb_replicator_pub.rrep_publication_tables

Report output:

- file xdb_rrep_publication_tables.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.52 Database xDB _edb_replicator_pub.rrep_txset table tuples (postgresql_db_xdb_rrep_txset)

Data from metatable _edb_replicator_pub.rrep_txset

Report output:

- file xdb_pub_rrep_txset.out

Depth: shallow

Security Impact: *low* — no known security impact.

10.4.53 Oracle-compatible partitioning key view (postgresql_epas_all_part_key_columns)

Provides partitioning key details (all_part_key_columns, EPAS specific)

Report output:

- file epas_all_part_key_columns.out

Depth: surface

Security Impact: *low* — no known security impact.

10.4.54 Oracle-compatible all partitioned table view (postgresql_epas_all_part_tables)

All partitioned tables view (from all_part_tables, EPAS specific)

Report output:

- file epas_all_part_tables.out

Depth: surface

Security Impact: *low* — no known security impact.

10.4.55 Oracle-compatible subpartitioning key view (postgresql_epas_all_subpart_key_columns)

Provides subpartitioning key details (all_subpart_key_columns, EPAS specific)

Report output:

- file epas_all_subpart_key_columns.out

Depth: surface

Security Impact: *low* — no known security impact.

10.4.56 Oracle-compatible all table partitions view (postgresql_epas_all_tab_partitions)

All partitions of all partitioned tables view (all_tab_partitions, EPAS specific)

Report output:

- file epas_all_tab_partitions.out

Depth: surface

Security Impact: *low* — no known security impact.

10.4.57 Oracle-compatible all table subpartitioning view (postgresql_epas_all_tab_subpartitions)

All subpartitions of all partitioned tables view (all_tab_subpartitions, EPAS specific)

Report output:

- file epas_all_tab_subpartitions.out

Depth: surface

Security Impact: *low* — no known security impact.

10.4.58 EPAS specific dblink information (postgresql_epas_dblink)

Current EPAS dblink information from edb_dblink)

Report output:

- file epas_dblink.out

Depth: surface

Security Impact: *low* — no known security impact.

10.5 Barman (Backup and Recovery Manager)

10.5.1 Barman check (barman_check)

Collect the status of the Barman check framework for all configured servers.

Report output:

- file /barman/barman_check.data : Output of barman check

Depth: surface

Security Impact: *low* — no known security impact.

10.5.2 Barman diagnose (barman_diagnose)

Collect the barman diagnosis information

Report output:

- file `/barman/diagnose.data` : Output from ‘barman diagnose’

Depth: surface

Security Impact: *low* — no known security impact.

10.5.3 Barman executable location (barman_executable)

Collect the ‘barman’ executable location

Report output:

- file `/barman/barman_location.data` : The path of the main barman executable

Depth: surface

Security Impact: *low* — no known security impact.

10.5.4 Barman module location (barman_modules_path)

Collect the location of the ‘barman’ Python modules

Report output:

- file `/barman/barman_python_verbose.data` : The list of Barman python modules

Depth: surface

Servers accepting upload of reports

Note: This information applies only to Lasso executables with external network access. You can verify that your Lasso can access the internet using the `--version` option. Look for the line *External network access enabled* . !!!

The Lasso `--upload` command sends the tarball to one of the front-line application servers that are configured in high availability and load balancing to receive Lasso reports for EDB as part of the Customer Portal.

These servers are part of an ISO 27K1 infrastructure and fully comply with the European Union' s General Data Protection Regulation (GDPR).

Provide this list of servers to your infrastructure team so that they can open outbound connections to port 443 of the following IPv4 addresses:

1. 78.46.204.124
2. 88.99.39.37
3. 88.99.39.42
4. 88.99.80.86
5. 116.203.69.173
6. 116.203.69.177
7. 116.203.69.179
8. 116.203.69.185
9. 116.203.69.248
10. 116.203.69.253
11. 159.69.178.25
12. 159.69.178.96
13. 159.69.181.84

14. 159.69.181.86
15. 195.201.138.157
16. 195.201.99.62

If you also use IPv6, add this list:

1. 2a01:4f8:1c17:482e::1
2. 2a01:4f8:1c17:4f1a::1
3. 2a01:4f8:1c17:5197::1
4. 2a01:4f8:1c17:529d::1
5. 2a01:4f8:c010:1210::1
6. 2a01:4f8:c010:154c::1
7. 2a01:4f8:c010:1550::1
8. 2a01:4f8:c010:1551::1
9. 2a01:4f8:1c1c:c82e::1
10. 2a01:4f8:1c1c:c836::1
11. 2a01:4f8:1c1c:c837::1
12. 2a01:4f8:1c1c:c95f::1
13. 2a01:4f8:1c1c:c97e::1
14. 2a01:4f8:1c1c:ca43::1
15. 2a01:4f8:1c1c:ca4e::1