

High storage usage

Last updated by | Hamza Aqel | Mar 8, 2023 at 2:31 AM PST

This is part of GT, to update please refer to haaqel@microsoft.com

Contents

- Check customer storage consumption
- Case 1: Wal upload/archival is fine but we have a high wor...
- Case 2: Inactive replication slot
- Case 3: Archival/upload is not working due to Blob Lease R...
- Case 4: Archival/upload is not working due to other failures
- Case 5: High temporary files usage:

Some customers are reporting that the storage consumption is increasing, and they need to know why that is happening, this TSG will check some corners on the customer server to see if there is any service issue related to that:

Check customer storage consumption

Take a look on the customer storage, most of the cases are related to the WAL size, so we will check first around this area if there is a problem there:

Before running the below command(s), you need to get the orcas_instance_id from ASC properties tab or from XTS view orcasbreadth\orcasbreadth server instance details.xls

readth\orcasbreadth server instance details.xls

Select Start/End Time; Press OK

Enter value

Start Date Time: 11/21/2022 8:21:14 AM

End Date Time: 11/21/2022 9:21:14 AM

Custom

☐ Auto refresh

Interval (sec): 30

End at (min): 10

name	replication_role	role	physical_count	server_type	server_edition	orcas_instance_id	id	orcas_instance_id1
122 7:28:12 AM	Primary		1	PostgreSQL	MemoryOptimized	448b744f-3706-4d7b-8d0b-e6726304a36e	448b744f-3706-4d7b-8d0b-e6726304a36e	448b744f-3706-4d7b-8d0b-e6726304a36e

Physical Instances. orcas_server_entity_id: 448b744f-3706-4d7b-8d0b-e6726304a36e

state	name	create_time	replication_role	orcas_instance_id	availability_zone	id	name1	orcas_instance_id1
Succeeded	b6a87f32cfa8	6/24/2022 7:28:13 AM	Primary	448b744f-3706-4d7b-8d0b-e6726304a36e	1	448b744f-3706-4d7b-8d0b-e6726304a36e	b6a87f32cfa8	448b744f-3706-4d7b-8d0b-e6726304a36e

Invoke-OrcasBreadthExecuteScriptWithRunCommand -OrcasInstanceId "448b744f-3706-4d7b-8d0b-e6726304a36e" - AzureVmRunCommandScriptName "df_kh"

```

The Environment Prod and Cluster wasd-prod-westeurope1-a-cr2 have been pre-selected for you.
PS C:\Users\haaqel\SqlAzureConsole> Invoke-OrcasBreadthExecuteScriptWithRunCommand -OrcasInstanceId "448b744f-3706-4d7b-8d0b-e6726304a36e" -AzureVmRunCommandScriptName "df_kh"
[stdout]
ps_hostname,b6a87f32cfa8
ps_now_utc,2022-11-21T12:15:22Z

Filesystem              Size  Used Avail Use% Mounted on
udev                    32G   0   32G   0% /dev
tmpfs                   6.3G  1.4M  6.3G   1% /run
/dev/sdc1               62G   27G   36G  43% /
tmpfs                   32G   0   32G   0% /dev/shm
tmpfs                   5.0M   0   5.0M   0% /run/lock
tmpfs                   32G   0   32G   0% /sys/fs/cgroup
/dev/sdc15              105M  4.4M  100M   5% /boot/efi
/dev/sdb                503G  83G  395G  18% /datadrive
/dev/mapper/encryptedssd 126G  33G  88G  27% /mnt
overlay                 62G   27G   36G  43% /var/lib/docker/overlay2/983734fbc7a10fdfe90c3c7838d0468e73f38ed29a28718fcaa8a8649230d669/merged
overlay                 62G   27G   36G  43% /var/lib/docker/overlay2/34686670c6b8ef39604ae118acce28596b4d70a49187380dccc6e07ac587db3/merged
overlay                 62G   27G   36G  43% /var/lib/docker/overlay2/e2e3a4dfdf129003734d61aa157dfda02191f523c9980448b9fee42394b2b15/merged
overlay                 62G   27G   36G  43% /var/lib/docker/overlay2/edcd258cd7724445ba2977f141d2970606db4d1f4eb3bbe6bd4374ca881db857/merged
overlay                 62G   27G   36G  43% /var/lib/docker/overlay2/7376c369a4001f5280e2120120713992f6a4db134c8f1f523bbf7306b8b4f652/merged
overlay                 62G   27G   36G  43% /var/lib/docker/overlay2/79f1d62c573c6c29358cf1d7627c15904b33d1ce008b7dfb17d9de05d543ecf5/merged
[stderr]

```

Invoke-OrcasBreadthExecuteScriptWithRunCommand -OrcasInstanceId "448b744f-3706-4d7b-8d0b-e6726304a36e" -AzureVmRunCommandScriptName "disk_usage_check_data_folder"

```

PS C:\Users\haaqel\SqlAzureConsole> Invoke-OrcasBreadthExecuteScriptWithRunCommand -OrcasInstanceId "448b744f-3706-4d7b-8d0b-e6726304a36e" -AzureVmRunCommandScriptName "disk_usage_check_data_folder"
[stdout]
ps_hostname,b6a87f32cfa8
ps_now_utc,2022-11-21T12:17:33Z

34G /datadrive/pg/data
71G /datadrive/pg/data/base
8.0M /datadrive/pg/data/base/1
8.0M /datadrive/pg/data/base/14029
9.6M /datadrive/pg/data/base/14030
8.1M /datadrive/pg/data/base/16384
8.8M /datadrive/pg/data/base/24577
71G /datadrive/pg/data/base/24836
192K /datadrive/pg/data/conf.d
784K /datadrive/pg/data/global
4.0K /datadrive/pg/data/pg_commit_ts
4.0K /datadrive/pg/data/pg_dynshmem
16K /datadrive/pg/data/pg_logical
4.0K /datadrive/pg/data/pg_logical/mappings
4.0K /datadrive/pg/data/pg_logical/snapshots
72M /datadrive/pg/data/pg_multixact
52M /datadrive/pg/data/pg_multixact/members
21M /datadrive/pg/data/pg_multixact/offsets
4.0K /datadrive/pg/data/pg_notify
4.0K /datadrive/pg/data/pg_replslot
4.0K /datadrive/pg/data/pg_serial
4.0K /datadrive/pg/data/pg_snapshots
4.0K /datadrive/pg/data/pg_stat
8.0K /datadrive/pg/data/pg_stat_statements
104K /datadrive/pg/data/pg_stat_tmp
212K /datadrive/pg/data/pg_subtrans
4.0K /datadrive/pg/data/pg_tblspc
4.0K /datadrive/pg/data/pg_tupdesc
13G /datadrive/pg/data/pg_wal
60K /datadrive/pg/data/pg_wal/archive_status
57M /datadrive/pg/data/pg_xact
8.0K /datadrive/pg/data/postgis
[stderr]

```

Notes:

[1] From df_kh , we know disk usage percentage.

[2] From disk_usage_check_data_folder , we know if data (tables) folder usage high or pg_wal folder usage is high

Case 1: Wal upload/archival is fine but we have a high workload

Rerun the above command after 15 mins for example:

Invoke-OrcasBreadthExecuteScriptWithRunCommand -OrcasInstanceId "448b744f-3706-4d7b-8d0b-e6726304a36e" -AzureVmRunCommandScriptName "disk_usage_check_data_folder"

```

PS C:\Users\haaqe1\SqlAzureConsole> Invoke-OrcasBreadthExecuteScriptWithRunCommand -OrcasInstanceId "448b744f-3706-4d7b-8d0b-e6726304a36e" -AzureVmRunCommandScriptName "disk_usage_check
Folder"
Enable succeeded:
[stdout]
ps_hostname,b6a87f32cfa8
ps_now_utc,2022-11-21T12:36:53z
35G /datadrive/pg/data
73G /datadrive/pg/data/base
8.0M /datadrive/pg/data/base/1
8.0M /datadrive/pg/data/base/14029
9.6M /datadrive/pg/data/base/14030
8.1M /datadrive/pg/data/base/16384
8.8M /datadrive/pg/data/base/24577
73G /datadrive/pg/data/base/24836
192K /datadrive/pg/data/conf.d
784K /datadrive/pg/data/global
4.0K /datadrive/pg/data/pg_commit_ts
4.0K /datadrive/pg/data/pg_dynshmem
16K /datadrive/pg/data/pg_logical
4.0K /datadrive/pg/data/pg_logical/mappings
4.0K /datadrive/pg/data/pg_logical/snapshots
73M /datadrive/pg/data/pg_multixact
52M /datadrive/pg/data/pg_multixact/members
21M /datadrive/pg/data/pg_multixact/offsets
4.0K /datadrive/pg/data/pg_notify
4.0K /datadrive/pg/data/pg_repislot
4.0K /datadrive/pg/data/pg_serial
4.0K /datadrive/pg/data/pg_snapshots
4.0K /datadrive/pg/data/pg_stat
8.0K /datadrive/pg/data/pg_stat_statements
304K /datadrive/pg/data/pg_stat_tmp
124K /datadrive/pg/data/pg_subtrans
4.0K /datadrive/pg/data/pg_tblspc
4.0K /datadrive/pg/data/pg_twophase
12G /datadrive/pg/data/pg_wal
40K /datadrive/pg/data/pg_wal/archive_status
57M /datadrive/pg/data/pg_xact
8.0K /datadrive/pg/data/postgis
[stderr]

```

If the upload is working, we should have a decrease in the WAL size folder as in the above, before was 13GB and after re-execute it is 12GB, and you can check the below too to confirm:

```
let STARTTIME_ARG = datetime('xxxx-xx-xx xx:xx:xx');
```

```
let ENDTIME_ARG = datetime('xxxx-xx-xx xx:xx:xx');
```

```
let SERVERNAME_ARG = 'pgflexservername';
```

```
MonOBPGSqlXLogFileCount
```

```
| where TIMESTAMP >= STARTTIME_ARG and TIMESTAMP <= ENDTIME_ARG
```

```
| where LogicalServerName == SERVERNAME_ARG
```

```
| order by TIMESTAMP desc | project TIMESTAMP, LogicalServerName, Ready_count, Done_count, File_count
```

TIMESTAMP	LogicalServerName	Ready_count	Done_count	File_count
2022-11-21 12:27:20.0000000	pgflexservernam	1	475	1246
2022-11-21 12:22:20.0000000	pgflexservername	1	325	1096
2022-11-21 12:17:20.0000000	pgflexservername	0	467	1254
2022-11-21 12:12:20.0000000	pgflexservername	1	317	1104
2022-11-21 12:07:20.0000000	pgflexservername	0	459	1267

As long as Ready_count is always below 2 and we have Done_count, so the upload is working, all of that is an indication that we have a high workload which you can check too:

```
let STARTTIME_ARG = datetime('xxxx-xx-xx xx:xx:xx');
```

```
let ENDTIME_ARG = datetime('xxxx-xx-xx xx:xx:xx');
```

```
let SERVERNAME_ARG = 'pgflexservername';
```

MonOBPgsqITransactionStats

| where TIMESTAMP >= STARTTIME_ARG and TIMESTAMP <= ENDTIME_ARG

| where LogicalServerName == SERVERNAME_ARG

| summarize sum(Tup_inserted), sum(Tup_updated), sum(Tup_deleted), sum(Commits) by TIMESTAMP

TIMESTAMP	sum_Tup_inserted	sum_Tup_updated	sum_Tup_deleted	sum_Commits
2022-11-21 09:02:10.0000000	273994147	571265485	10614	478192929
2022-11-21 09:17:10.0000000	273994147	571265604	10614	478195145
2022-11-21 09:22:10.0000000	273994147	571265644	10614	478196034
2022-11-21 09:57:20.0000000	273994147	571265922	10614	478201638
2022-11-21 10:02:20.0000000	273994148	571265962	10614	478202332
2022-11-21 10:07:20.0000000	273994148	571266003	10614	478203235
2022-11-21 10:12:20.0000000	273994148	571266041	10614	478204053
2022-11-21 10:17:20.0000000	473993982	571266793	12368	478204881
2022-11-21 10:37:20.0000000	481749245	571267008	12375	478362969
2022-11-21 10:47:20.0000000	485887000	571267124	12380	478447171
2022-11-21 11:12:20.0000000	496669411	571267399	12390	478666478
2022-11-21 11:17:20.0000000	498842511	571267440	12390	478710664
2022-11-21 11:22:20.0000000	501038063	571267496	12392	478755299
2022-11-21 11:27:20.0000000	503183666	571267556	12395	478798933
2022-11-21 11:42:20.0000000	509641871	571267712	12400	478930293
2022-11-21 12:07:20.0000000	520293727	571267950	12405	479146977
2022-11-21 12:12:20.0000000	522452679	571268006	12407	479190872
2022-11-21 12:27:20.0000000	529000635	571268166	12412	479324014
2022-11-21 12:37:20.0000000	533297185	571268244	12412	479411431
2022-11-21 12:42:20.0000000	535471738	571268303	12415	479455629

```
let STARTTIME_ARG = datetime('xxxx-xx-xx xx:xx:xx');
```

```
let ENDTIME_ARG = datetime('xxxx-xx-xx xx:xx:xx');
```

```
let SERVERNAME_ARG = 'pgflexservername';
```

OBvmagentsidecarpgsql

| where TIMESTAMP >= STARTTIME_ARG and TIMESTAMP <= ENDTIME_ARG

| where LogicalServerName == SERVERNAME_ARG

| project originalEventTimestamp, VirtualMachineName, LogLevel, MessageString

| where MessageString contains "Found" and MessageString contains "WAL files" or MessageString contains "PostgreSQLDbWalUpload"

| where MessageString !contains "Number of 'ready' files: 0 before"

| where MessageString !contains "Found 0"

| order by originalEventTimestamp asc

originalEventTimestamp	VirtualMachineName	LogLevel	MessageString
b6a87f32cfa8		1	[PostgreSQLDbWalUpload].GetCpuProcessorCount: CPU processor count is 8
b6a87f32cfa8		2	[PostgreSQLDbWalUpload].UploadWalWithLease: Function Exit. Wal files had been uploaded and been truncated
b6a87f32cfa8		2	[PostgreSQLDbWalUpload].UploadWal: Number of 'ready' files: 1 before, 0 after upload. upload duration ms=1078, catchup threshold=20, allow rerun=True, rerun=False, consecutive executions=1.
b6a87f32cfa8		1	[PostgreSQLDbWalUpload].UploadWal: FunctionExit
b6a87f32cfa8		1	[PostgreSQLDbWalUpload].UploadWal: FunctionEnter
b6a87f32cfa8		1	[PostgreSQLDbWalUpload].GetCpuProcessorCount: CPU processor count is 8
b6a87f32cfa8		1	[PostgreSQLDbWalUpload].GetMemSizeInKb: CPU memory size in kilobytes is 65928320
b6a87f32cfa8		2	[PostgreSQLDbWalUpload].ShouldSidecarUploadWal: FSPG non-burstable SKU detected or could not find /datadrive/enable_pg_wal_upload.txt. WAL files will only be archived by sidecar with lease mechanism...
b6a87f32cfa8		1	[PostgreSQLDbWalUpload].UploadWalWithLease: FunctionEnter
b6a87f32cfa8		2	[PostgreSQLDbWalUpload].UploadWalWithLease: Function Exit. Wal files had been uploaded and been truncated
b6a87f32cfa8		1	[PostgreSQLDbWalUpload].UploadWal: FunctionExit
b6a87f32cfa8		2	[PostgreSQLDbWalUpload].UploadWalWithLease: Function Exit. Wal files had been uploaded and been truncated
b6a87f32cfa8		2	[PostgreSQLDbWalUpload].UploadWal: Number of 'ready' files: 38 before, 17 after upload. upload duration ms=4467, catchup threshold=20, allow rerun=True, rerun=False, consecutive executions=1.
b6a87f32cfa8		1	[PostgreSQLDbWalUpload].UploadWal: FunctionExit
b6a87f32cfa8		1	[PostgreSQLDbWalUpload].UploadWal: FunctionEnter
b6a87f32cfa8		1	[PostgreSQLDbWalUpload].GetCpuProcessorCount: CPU processor count is 8

On that case we can let the customer knows that his workload is generating these WALs, and it will be minimized when that high workload decreases.

Case 2: Inactive replication slot

When the customer has inactive replication slot, the WAL files will keep building up due to that slot, so we can check if the customer has any and recommend to drop these slots:

let STARTTIME_ARG = datetime('xxxx-xx-xx xx:xx:xx');

let ENDTIME_ARG = datetime('xxxx-xx-xx xx:xx:xx');

let SERVERNAME_ARG = 'pgflexservername';

MonOBPqSqlReplicationStats

| where TIMESTAMP >= STARTTIME_ARG and TIMESTAMP <= ENDTIME_ARG

| where LogicalServerName == SERVERNAME_ARG

| where Active == 0

| distinct Slot_name, Slot_type, Active

Slot_name	Slot_type	Active
mon_slot	logical	False

Or you can run the below CAS command:

Invoke-OrcasBreadthExecuteScriptWithRunCommand -OrcasInstanceId "8d6606f9-0ade-4d04-b60a-bff64212c3bf" -AzureVmRunCommandScriptName "docker_exec_psql_pg_replication_slots"

```
Enable succeeded:
[stdout]
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn	wal_status
azure_asyncreplica_eadbf284033		physical			f	t	121			19/780dFAB8		reserved
mon_slot	test_decoding	logical	14030	postgres	f	f			977903	19/780A4B78	19/780A4B80	reserved

(2 rows)

In that case, ask the customer to drop the inactive replication slot:

From the customer side, he can execute from his end:

```
select * from pg_replication_slots where not active;
```

```
select pg_drop_replication_slot('slot_name') FROM pg_replication_slots WHERE NOT active;
```

```
select * from pg_replication_slots where not active;
```

```
postgres=> select * from pg_replication_slots where not active;
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn	wal_status	safe_wal_size
mon_slot	test_decoding	logical	14030	postgres	f	f			977986	19/79000580	19/79000588	reserved	

(1 row)

Time: 2.707 ms

```
postgres=> SELECT pg_drop_replication_slot('mon_slot') FROM pg_replication_slots WHERE NOT active;
```

```
pg_drop_replication_slot
```

--

(1 row)

Time: 9.875 ms

```
postgres=> select * from pg_replication_slots where not active;
```

slot_name	plugin	slot_type	datoid	database	temporary	active	active_pid	xmin	catalog_xmin	restart_lsn	confirmed_flush_lsn	wal_status	safe_wal_size
-----------	--------	-----------	--------	----------	-----------	--------	------------	------	--------------	-------------	---------------------	------------	---------------

(0 rows)

Case 3: Archival/upload is not working due to Blob Lease Renew Issue

If the cases ##1 and ##2 did not help and the WAL folder is keep increasing, there is a possibility that the WAL upload is not working for some reason, one of the reasons is due to Blob lease renew issue, to check that you can use the below query:

```
let STARTTIME_ARG = datetime('xxxx-xx-xx xx:xx:xx');
```

```
let ENDTIME_ARG = datetime('xxxx-xx-xx xx:xx:xx');
```

```
let SERVERNAME_ARG = 'pgflexservername';
```

```
OBvmagentsidecarpgsql
```

```
| where TIMESTAMP >= STARTTIME_ARG and TIMESTAMP <= ENDTIME_ARG
```

```
| where LogicalServerName == SERVERNAME_ARG
```

```
| project TIMESTAMP, VirtualMachineName, LogLevel, MessageString, LogicalServerName
```

```
| where MessageString contains "cancellation in lease renew"
```

```
| extend lease_failed_wal_file_v1 = extract("UploadWalFiles: Failed to upload file for ([0-9A-Fa-f]+) due to cancellation in lease renew", 1, MessageString)
```

```
| extend lease_failed_wal_file_v2 = extract(@"\[Archive\].UploadWalFiles: Failed: ([\\w\\.]+) ", 1, MessageString)
```

```
| extend lease_failed_wal_file = iff(isnotempty(lease_failed_wal_file_v1), lease_failed_wal_file_v1, lease_failed_wal_file_v2)
```

```
| summarize min(TIMESTAMP), max(TIMESTAMP) by lease_failed_wal_file, MessageString
```

```
| order by lease_failed_wal_file asc
```

lease_failed_wal_file	MessageString
000000010000000F00000021	[Archive].UploadWalFiles: Failed: 000000010000000F00000021 PostgreSQL Failed to upload due to cancellation in lease renew : System.Threading.Tasks.TaskCanceledException: The operation was canceled. at Azure.Core.CancellationHelper.ThrowOperationCanceledException(Exception innerException, CancellationToken cancellationToken) at Azure.Core.CancellationHelper.ThrowIfCancellationRequested(CancellationToken cancellationToken) at Azure.Core.Pipeline.ResponseBodyPolicy.ProcessAsync(HttpMessage message, ReadOnlyMemory`1 pipeline, Boolean async) at Azure.Core.Pipeline.HttpPipelineSynchronousPolicy.<ProcessAsync>g__ProcessAsyncInner4_0(HttpMessage message, ReadOnlyMemory`1 pipeline) at Azure.Core.Pipeline.RedirectPolicy.ProcessAsync(HttpMessage message, ReadOnlyMemory`1 pipeline, Boolean async) at Azure.Core.Pipeline.RetryPolicy.ProcessAsync(HttpMessage message, ReadOnlyMemory`1 pipeline, Boolean async)
000000010000000F00000022	[Archive].UploadWalFiles: Failed: 000000010000000F00000022 PostgreSQL Failed to upload due to cancellation in lease renew : System.Threading.Tasks.TaskCanceledException: The operation was canceled. at Azure.Core.CancellationHelper.ThrowOperationCanceledException(Exception innerException, CancellationToken cancellationToken) at Azure.Core.CancellationHelper.ThrowIfCancellationRequested(CancellationToken cancellationToken) at Azure.Core.Pipeline.ResponseBodyPolicy.ProcessAsync(HttpMessage message, ReadOnlyMemory`1 pipeline, Boolean async) at Azure.Core.Pipeline.HttpPipelineSynchronousPolicy.<ProcessAsync>g__ProcessAsyncInner4_0(HttpMessage message, ReadOnlyMemory`1 pipeline) at Azure.Core.Pipeline.RedirectPolicy.ProcessAsync(HttpMessage message, ReadOnlyMemory`1 pipeline, Boolean async) at Azure.Core.Pipeline.RetryPolicy.ProcessAsync(HttpMessage message, ReadOnlyMemory`1 pipeline, Boolean async)

If you see such errors, you file an ICM and engage the engineering team to fix it.

Case 4: Archival/upload is not working due to other failures

Use the below query to check if there is any other failures in the upload process, and engage the engineering team if needed:

```
let STARTTIME_ARG = datetime('xxxx-xx-xx xx:xx:xx');
```

```
let ENDTIME_ARG = datetime('xxxx-xx-xx xx:xx:xx');
```

```
let SERVERNAME_ARG = 'pgflexservername';
```

```
OBvmagentsidecarpgsql
```

```
| where TIMESTAMP >= STARTTIME_ARG and TIMESTAMP <= ENDTIME_ARG
```

```
| where LogicalServerName == SERVERNAME_ARG
```

```
| project originalEventTimestamp, VirtualMachineName, LogLevel, MessageString, LogicalServerName
```

```
| where MessageString contains "UploadWalFiles" and MessageString contains "fail"
```

```
| order by originalEventTimestamp asc
```

originalEventTimestamp	VirtualMachineName	LogLevel	MessageString
			Headers: Server: Windows-Azure-Blob/1.0,Microsoft-HTTPAPI/2.0 [Archive].UploadWalFiles: Failed: Lease 0000000500001B680000004E.lease PostgreSQL Fail acquire the lease with : Azure.RequestFailedException: There is already a lease present. RequestId:bfe64a66-001e-0101-199d-fd700a000000 Time:2022-11-21T11:34:10.2394226Z Status: 409 (There is already a lease present.) ErrorCode: LeaseAlreadyPresent
	a06ccdba98d7	4	
			Headers: Server: Windows-Azure-Blob/1.0,Microsoft-HTTPAPI/2.0 [Archive].UploadWalFiles: Failed: Lease 0000000500001B7200000009B.lease PostgreSQL Fail acquire the lease with : Azure.RequestFailedException: There is already a lease present. RequestId:407e3b92-301e-001e-50b2-fd16eb000000 Time:2022-11-21T14:09:29.6296829Z Status: 409 (There is already a lease present.) ErrorCode: LeaseAlreadyPresent
	fe1f2ee96050	4	
			Headers: Server: Windows-Azure-Blob/1.0,Microsoft-HTTPAPI/2.0

Case 5: High temporary files usage:

Our Product Group engineers have developed new workbooks for the Azure database for PostgreSQL flexible server that will aid the support engineers in resolving a variety of customer issues:

- High CPU.
- High Memory.
- High IOPs.
- Autovacuum/vacuum monitoring.
- Autovacuum blockers and Wraparound identification.
- High temporary file usage.

You can use the High temporary file usage to see if this is related or not:

Workbook Name	Link
High temporary file usage	https://ms.portal.azure.com/#@microsoft.onmicrosoft.com/resource/subscriptions/b1172cf5a-4917-ac31-6f06460ff9b2/resourceGroups/prod-internal-perf-workbooks/providers/microsoft.insights/workbooks/b1f4f9fd-9d92-4eb7-a388-711ccb482789/workbook 