[CSSTSG-Orcas-PGSS-Error] Error: Could not read block xxxx in file "base...": Invalid argument error - after K63

Last updated by | Shawn Xiao | Jan 16, 2023 at 9:53 PM PST

Contents

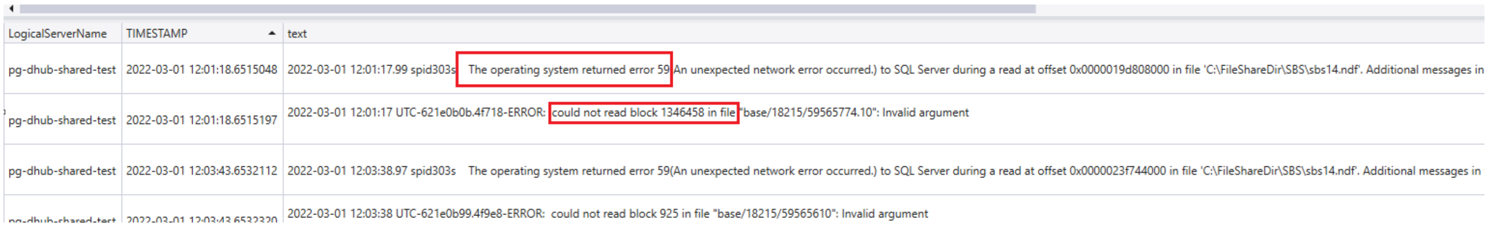
Issue Description

After K63 deployment , we are seeing some incidents reported where the customer is facing block corruption in his database server with error like

```
ERROR:  could not read block xxxx in file "base/xxxxx/xxxxxx": Invalid argument
```

To validate, use the below Kusto query:

```
let TimeCheckStart = datetime('2022-xx-xx xx:xx:xx');
let TimeCheckEnd = datetime('2022-xx-xx xx:xx:xx');
MonRdmsPgSqlSandbox
| where LogicalServerName == "pgservername"
| where originalEventTimestamp > TimeCheckStart and originalEventTimestamp < TimeCheckEnd
| where text contains "Error 59" or text contains "could not read block"
| project LogicalServerName, TIMESTAMP, text
```



LogicalServerName	TIMESTAMP	text
pg-dhub-shared-test	2022-03-01 12:01:18.6515048	2022-03-01 12:01:17.99 spid303s The operating system returned error 59(An unexpected network error occurred.) to SQL Server during a read at offset 0x0000019d808000 in file 'C:\FileShareDir\SBS\sbs14.ndf'. Additional messages in
pg-dhub-shared-test	2022-03-01 12:01:18.6515197	2022-03-01 12:01:17 UTC-621e0b0b.4f718-ERROR: could not read block 1346458 in file "base/18215/59565774.10": Invalid argument
pg-dhub-shared-test	2022-03-01 12:03:43.6532112	2022-03-01 12:03:38.97 spid303s The operating system returned error 59(An unexpected network error occurred.) to SQL Server during a read at offset 0x0000023f744000 in file 'C:\FileShareDir\SBS\sbs14.ndf'. Additional messages in
pg-dhub-shared-test	2022-03-01 12:03:43.6532320	2022-03-01 12:03:38 UTC-621e0b99.4f9e8-ERROR: could not read block 925 in file "base/18215/59565610": Invalid argument

Mitigation

Mitigation is to restart the server to clear these invalid handles.

Root Cause

Azure Database for Postgres service uses Azure premium file share storage for storing data. Azure Files support SMB Transparent Failover with SMB3.0 and above. This is achieved using persisted handles. SMB layer would try to reconnect during such scenarios to maintain uninterrupted IOs. But due to a rare bug, the handles are not re-established causing the IOs to fail.

Fix Plan

A fix has been delivered by Azure SMB team which is currently being tested by Azure database for Postgres service team. Post testing will be rolled out to production in the future deployment.