SSL Provider The specified data could not be decrypted

Last updated by | Vitor Tomaz | Feb 24, 2023 at 3:27 AM PST

Contents

Issue

The error "SSL Provider: The specified data could not be decrypted" has been seen in several scenarios:

- Stand-alone databases
- One or more databases in an elastic pool
- Queries involving Elastic Query scenarios, including external tables and calling stored procedures remotely through sp_execute_remote

It may occur for either client applications like Azure Functions or manual database queries executed in an SSMS query window.

Typical error messages are reported as either of:

> An error occurred while establishing connection to **remote data source**:
> [Microsoft][ODBC Driver 17 for SQL Server]SSL Provider: The specified data could not be decrypted.
> [Microsoft][ODBC Driver 17 for SQL Server]Client unable to establish connection

> An error occurred while excecuting query on **remote server**:
> [Microsoft][ODBC Driver 17 for SQL Server]TCP Provider: An existing connection was forcibly closed by the remote host.
> [Microsoft][ODBC Driver 17 for SQL Server]Communication link failure

> ExceptionType=System.Data.SqlClient.SqlException
> ExceptionMessage=A transport-level error has occurred when receiving results from the server. (provider: SSL Provider, error: 0 - The specified data could not be decrypted.)
> InnerExceptionType=System.ComponentModel.Win32Exception
> InnerExceptionMessage=The specified data could not be decrypted
> InnerExceptionNumber=-2146893008

Note that the "remote" key word on the error message is indicating the Elastic Query scenario.

Investigation / Analysis

The common factor on previous support cases was a spike in workload shortly before the error, and/or high CPU consumption or CPU spikes at that time. High workload or high login rates can both relate to high CPU.

Common troubleshooting approach

The recommended troubleshooting approach is to follow the Workflow for High CPU troubleshooting and also look further into the CPU_Troubleshooting article. Make sure to look at the maximum CPU values, not the average values, as short-time spikes to 100% can also cause this error.

When looking at the ASC troubleshooter output, check for Sub Core SLOs which are prone to inconsistent behaviour during workload spikes and CPU pressure.

Specific step to check for Elastic Query scenarios

To get an idea about the remote queries and remote stored proceduure executions, the `MonGlobalQueryEvents` Kusto table can show you both the local and the remote server/database that are involved in the failed queries. Cross-check the CPU and workload for the remote server if the local server looks OK.

```
let srv = "servername";
let db = "databasename";
let startTime = datetime(2022-11-22 11:00:00);
let endTime = datetime(2022-11-22 11:30:00);
let timeRange = ago(7d);
MonGlobalQueryEvents
| where originalEventTimestamp  >= startTime
| where originalEventTimestamp <= endTime
//| where  TIMESTAMP >= timeRange
| where LogicalServerName == srv
| where logical_database_name == db
| where event == "global_query_extractor_fail"
| distinct request_id
| join kind = inner (
    MonGlobalQueryEvents
    | where originalEventTimestamp  >= startTime
    | where originalEventTimestamp <= endTime
    //| where  TIMESTAMP >= timeRange
) on request_id
| extend local_server = LogicalServerName
| extend local_database= logical_database_name
| extend remote_server = server_name
| extend remote_database = database_name
| project TIMESTAMP, NodeName, AppName, local_server, local_database, remote_server, remote_database, external
```

```
TIMESTAMP                    NodeName  AppName       local_server  local_database  remote_server
2022-11-22 11:01:06.8417250  DB.6      b58ab16c064e  servername    databasename    remoteserver.database.windo
2022-11-22 11:01:06.8417250  DB.6      b58ab16c064e  servername    databasename
2022-11-22 11:01:06.8417250  DB.6      b58ab16c064e  servername    databasename
2022-11-22 11:01:36.8414202  DB.6      b58ab16c064e  servername    databasename    remoteserver.database.windo
2022-11-22 11:01:36.8414202  DB.6      b58ab16c064e  servername    databasename
2022-11-22 11:01:36.8414202  DB.6      b58ab16c064e  servername    databasename
```

Note that for the SSL error, you would see error_code=12 or 13. I have added the row with error_code=14 only to demonstrate how other error types might look like.

> An error occurred while establishing connection to remote data source
> const HRESULT EXT_USER_GLOBALQUERY_ESTABLISHING_REMOTE_CONNECTION_ERROR =
> MAKE_HRESULT(SEVERITY_ERROR, FACILITY_EXTRACTOR_USER, 12);
> An error occurred while excecuting query on remote server
> const HRESULT EXT_USER_GLOBALQUERY_REMOTE_QUERY_EXECUTION_ERROR =
> MAKE_HRESULT(SEVERITY_ERROR, FACILITY_EXTRACTOR_USER, 13);

Mitigation

Short term

If the database is a Sub Core SLO, then scale up to at least Standard S3. S0~S2 is not recommended for production environments. By scaling to at least S3, it will move you away from the unpredictable experience and provide more stability for your workload.

If the issue occurs for an elastic pool database, consider moving the database out of the pool instead of scaling the pool.

Mid/Long term

Review the top CPU-consuming queries on the database and tune them as best as possible. See the resources section below for further links.

Internal Doc References

- Query tuning - *customer-facing*
- Query Tuning Troubleshooting - *Kusto telemetry*
- High CPU Utilization - *customer-facing*
- CPU_Troubleshooting
- Workflow for High CPU troubleshooting

How good have you found this content?

🙂 🙁