Service Fabric bug causing performance issues after planned maintenance

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

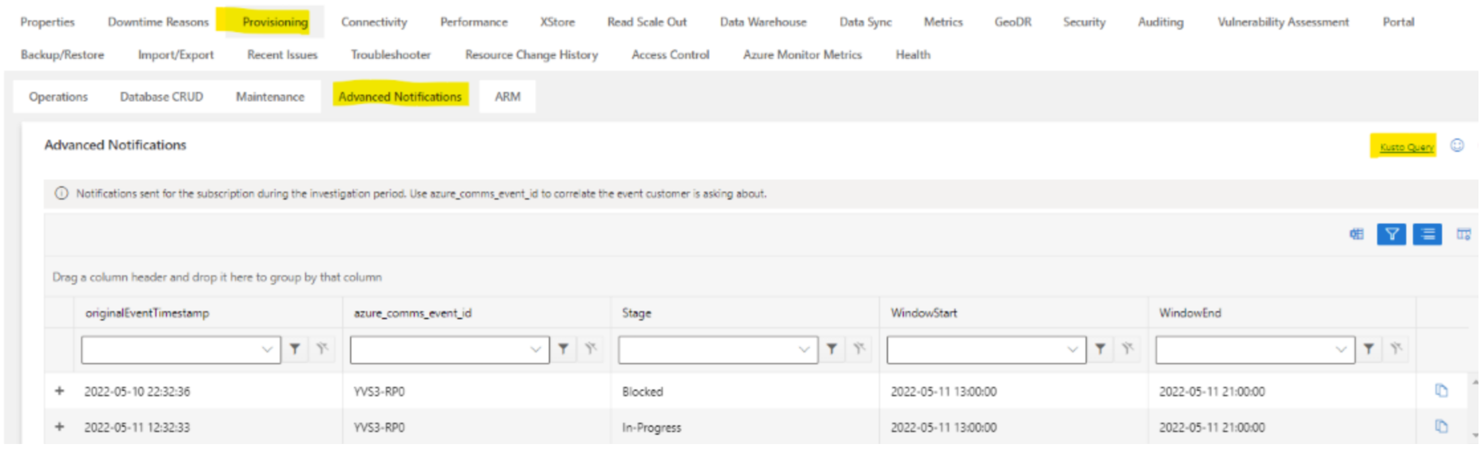
Contents

Issue

Performance issues in SQL Databases such as high CPU, IO and query timeouts are experienced following a planned maintenance. This is due to a code defect in Service Fabric. The fix is completed and released on Cluster version 9.1 CU. The planned train to rollout the update is T-67 set for the beginning of 2023.

Investigation/Analysis

Confirm in ASC that performance issues coincided with planned maintenance.



Advanced Notifications

Notifications sent for the subscription during the investigation period. Use azure_comms_event_id to correlate the event customer is asking about.

Drag a column header and drop it here to group by that column

originalEventTimestamp	azure_comms_event_id	Stage	WindowStart	WindowEnd
2022-05-10 22:32:36	YVS3-RP0	Blocked	2022-05-11 13:00:00	2022-05-11 21:00:00
2022-05-11 12:32:33	YVS3-RP0	In-Progress	2022-05-11 13:00:00	2022-05-11 21:00:00

Run the following Kusto to confirm the replica state is NOT SYNCHRONIZING and NOT_HEALTHY during the maintenance time period.

```
MonDmDbHadrReplicaStates
| where LogicalServerName =~ "{LogicalServerName}"  and AppName =~ "{AppName}"  //and  logical_database_name
| where TIMESTAMP between (datetime({StartDateTime})..datetime({EndDateTime}))
| where synchronization_state_desc == "NOT SYNCHRONIZING" or synchronization_health_desc == "NOT_HEALTHY"
| project TIMESTAMP, NodeName , synchronization_state_desc , synchronization_health_desc
| order by TIMESTAMP asc nulls last
```

In Kusto, run this query to analysis the service fabric logs.

```
WinFabLogs
| where TaskName == "CRM" and EventType == "Operation"
| where Text contains "Phase: " and Text contains "Action: " and Text contains "DecisionId: "
| where ETWTimestamp >= ago(4d)
| extend ServiceName = extract("Service: ([\\.\\-/:a-zA-Z|0-9]+) \r", 1, Text , typeof(string)),
DecisionId = toguid(extract("DecisionId: ([[a-z|A-Z|0-9|-]+)\r", 1, Text , typeof(string))),
Phase = extract("Phase: ([A-Za-z]+) \r", 1, Text , typeof(string)),
Action = extract("Action: ([A-Za-z]+) \r", 1, Text , typeof(string)),
SourceNodeId = extract("SourceNode: ([a-zA-Z0-9]+)\r", 1, Text , typeof(string)),
TargetNodeId = extract("TargetNode: ([a-zA-Z0-9]+)\r", 1, Text , typeof(string))
| where ServiceName startswith "fabric:/Worker.ISO.Premium/{AppName}"
| project ETWTimestamp , ClusterName , ServiceName , DecisionId, Phase , Action , SourceNodeId , TargetNodeId
| join kind = leftouter (
WinFabLogs // Violations
    | where ETWTimestamp >= ago(4d)
    | where TaskName == "PLB" and EventType == "SchedulerAction"
    | where Text contains "Constraint Violations: " and ((Text contains " NodeCapacity" and Text contains "[Me
    | extend DecisionId = toguid(extract("DecisionId: (.+)[[:space:]]Affects Service", 1, Text , typeof(string
ConstraintType = extract("--\\[([a-zA-Z0-9_]+), [0-9]+, [0-9]+, [a-zA-Z0-9]+, [A-Z0-9\\.]+, N/A\\]", 1, Text ,
    | extend ConstraintType = iff(isempty(ConstraintType) and Text contains "Affinity" and Text !contains "[Me
"Affinity", ConstraintType)
    | project ETWTimestamp , ClusterName , DecisionId , ConstraintType , Text
) on ClusterName, DecisionId
| join kind= leftouter (
WinFabLogs // Source node name
    | where ETWTimestamp >= ago(4d)
    | where EventType == "NodeLoads" and TaskName == "PLB"
    | project ETWTimestamp , ClusterName ,EventType , TaskName , Id , Level , Text
    | extend node_info = split(Text, "\r")
    | mvexpand node_info
    | extend SourceNodeId = extract("([a-z0-9]+) [A-Z0-9\\.]+ \\(", 1, trim("\t", tostring(node_info)), typeof
SourceNodeName = extract("[a-z0-9]+ ([A-Z0-9\\.]+) \\(", 1, trim("\t", tostring(node_info)), typeof(string))
    | where isnotempty(SourceNodeId) and isnotempty(SourceNodeName)
    | summarize by ClusterName, SourceNodeId, SourceNodeName
) on ClusterName , SourceNodeId
| join kind= leftouter (
WinFabLogs // Target node name
    | where ETWTimestamp >= ago(4d)
    | where EventType == "NodeLoads" and TaskName == "PLB"
    | project ETWTimestamp , ClusterName ,EventType , TaskName , Id , Level , Text
    | extend node_info = split(Text, "\r")
    | mvexpand node_info
    | extend TargetNodeId = extract("([a-z0-9]+) [A-Z0-9\\.]+ \\(", 1, trim("\t", tostring(node_info)), typeof
TargetNodeName = extract("[a-z0-9]+ ([A-Z0-9\\.]+) \\(", 1, trim("\t", tostring(node_info)), typeof(string))
    | where isnotempty(TargetNodeId) and isnotempty(TargetNodeName)
    | summarize by ClusterName, TargetNodeId, TargetNodeName
) on ClusterName , TargetNodeId
| extend RingName = toupper(extract("(.*)\\.(.*)\\.worker.(database.windows.net|sqltest-eg1.mscds.com)", 1, Cl
| project ETWTimestamp , RingName , ServiceName , Phase , Action , ConstraintType , SourceNodeName , TargetNod
| order by ETWTimestamp asc
```

Example output

										f886f01ea4ce
2022-05-10T19:54:24.175Z	TR2005	fabric:/Worker.ISO.Premium/e93cde2054c7/SQL.UserDb/01798326-465a-40d0-85a1-24dd70b56448	ConstraintCheck	MovePrimary				503b8d12747ef40013f94d2ceb29afc8	89c127699f34476af9ffcade2c90214e	7c8a2075-bc53-4013-a773-0a68228ab1c8
2022-05-10T19:54:24.176Z	TR2005	fabric:/Worker.ISO.Premium/e93cde2054c7/SQL.LogicalServer/e93cde2054c7	ConstraintCheck	MovePrimary				503b8d12747ef40013f94d2ceb29afc8	89c127699f34476af9ffcade2c90214e	7c8a2075-bc53-4013-a773-0a68228ab1c8
2022-05-10T19:54:55.732Z	TR2005	fabric:/Worker.ISO.Premium/e93cde2054c7/SQL.LogicalServer/e93cde2054c7	ConstraintCheck	MoveSecondary				503b8d12747ef40013f94d2ceb29afc8	ba48f1cb4aca26bca5eab5febb99c6f2	c0d7445b-560f-4e5b-a148-6bbe5edd7f6d
2022-05-10T20:19:13.359Z	TR2005	fabric:/Worker.ISO.Premium/e93cde2054c7/SQL.LogicalServer/e93cde2054c7	ConstraintCheck	MoveSecondary	Affinity			ba48f1cb4aca26bca5eab5febb99c6f2	26ff4cfa36545e150fa0840aa46bcbb3	396574ee-1625-4f31-8beb-462bea729317
2022-05-10T20:19:17.348Z	TR2005	fabric:/Worker.ISO.Premium/e93cde2054c7/SQL.LogicalServer/e93cde2054c7	NewReplicaPlacement	DropSecondary				ba48f1cb4aca26bca5eab5febb99c6f2	0	f3e549e7-f031-4e0f-9808-df6ae527efd9
2022-05-10T20:19:35.049Z	TR2005	fabric:/Worker.ISO.Premium/e93cde2054c7/SQL.UserDb/01798326-465a-40d0-85a1-24dd70b56448	NewReplicaPlacement	AddSecondary				0	26ff4cfa36545e150fa0840aa46bcbb3	8e897bec-c4a2-4167-b90a-8a3b3db2ef40
2022-05-10T20:19:49.814Z	TR2005	fabric:/Worker.ISO.Premium/e93cde2054c7/SQL.UserDb/01798326-465a-40d0-85a1-24dd70b56448	NewReplicaPlacement	AddSecondary				0	26ff4cfa36545e150fa0840aa46bcbb3	e8578ae1-e23d-4611-8b51-31120c08f9be
2022-05-11T02:43:05.959Z	TR2005	fabric:/Worker.ISO.Premium/e93cde2054c7/SQL.LogicalServer/e93cde2054c7	NewReplicaPlacement	DropSecondary				503b8d12747ef40013f94d2ceb29afc8	0	0c4dbec8-bdfc-43c0-8cd8-8ffe5250713c
2022-05-11T02:43:05.959Z	TR2005	fabric:/Worker.ISO.Premium/e93cde2054c7/SQL.UserDb/01798326-465a-40d0-85a1-24dd70b56448	NewReplicaPlacement	DropSecondary				503b8d12747ef40013f94d2ceb29afc8	0	0c4dbec8-bdfc-43c0-8cd8-8ffe5250713c

In this example, node **503b8d12747ef40013f94d2ceb29afc8** was moved from Primary to Secondary and then was dropped from the replica after 8 hours. The planned maintenance window was for tenant upgrade. This consists of Azure infrastructure jobs which perform upgrades by picking up new GuestOS and other payloads if they are available. A side effect of this planned maintenance is a cluster reconfiguration which includes SQL Server restart and moving around of Primary/Secondary nodes. Generally this is a quick operation and only causes a momentary blip. However, in this case the planned maintenance of the tenant ring hit a placement affinity constraint violation while placing a replica. Although the addition of the replica (AddReplica) succeeded, the dropping of the replica (DropReplica) was stuck. Essentially what happened is that for multi Azure clusters, which only have 3 fault domains (and this database in question has 4 replicas), when one of the replicas goes down due to an upgrade, PLB (load balancer) will attempt to fix the fault domain constraint, even though it should just wait for the upgrade to finish. This lead to cluster becoming in an unhealthy state. A healthy cluster has 1 primary and 3 active secondaries, however, this cluster was in a state where there was 1 primary, 3 active secondaries and 1 idle secondary. The issue did eventually self-mitigated by moving the cluster to a healthy state after a number of hours.

Mitigation

This is rare scenario and does not commonly happen. The internal monitoring system should catch this before customers start seeing impact. Once determined if the underlying cause of the issue is the same and the issue remains unresolved, create an IcM for the Product Group to manually force removing the DB from that node.

RCA Template

This issue is caused by a recently identified Service Fabric code defect which erroneously moves a database when upgrading the service. During the maintenance process this resulted in Service Fabric preemptively building a new replica, which subsequently blocked the previous replica from dropping. The code fix to resolve this issue is actively being worked on.

We apologize sincerely for the inconvenience that this issue has caused your business. We are constantly taking steps to improve our service, which includes, but is not limited to adding additional telemetry to enhance detection and mitigation processes.

Internal Reference

IcM: 307012678 - High CPU Consumption
IcM: 292012983 - RCA for PLB ResourceBalancing Post Maintenance 12:07 20220228 UTC
IcM: 293008243 - Multiple Databases in a Degraded State in one SQL server
IcM: 307200973 - Resource health event alerts causing connection failures

Code fix Work Item: 7141674

Root Cause Classification

Cases resolved by this TSG should be coded to the following root cause:
/Azure SQL v3/Performance/<related cause - e.g. CPU or IO>

How good have you found this content?