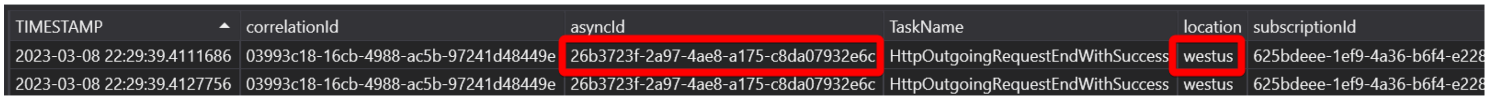
Troubleshooting Read Replica creation issues

Last updated by | Ulisses Alves | Mar 16, 2023 at 1:16 PM PDT

To thorubleshooting cxs have issues creating PGFS Read Replica:

1. Run the following Kusto query on ARMProd (update the Tracking ID and start date):

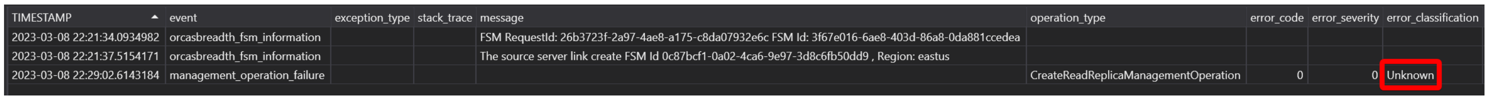
```
HttpOutgoingRequests
| where TIMESTAMP >= datetime('2023-03-08T22:28:00Z')
//| where subscriptionId =~ "customersubscriptionid"
| where serviceRequestId =~ 'aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee' //Tracking ID from Customer
//| where correlationId =~ "correlationID"
| where operationName contains "AZUREASYNCOPERATION"
| extend location = extract('locations/([a-z|A-Z|0-9|]+)', 1, targetUri, typeof(string))
| extend asyncId = extract('azureAsyncOperation/([a-z|A-Z|0-9|-]+)', 1, targetUri, typeof(string))
| where isnotempty( asyncId) //and location contains "AddRegionHere"
| project TIMESTAMP, correlationId, asyncId,TaskName, location, subscriptionId, operationName, targetUri, dura
```



TIMESTAMP	correlationId	asyncId	TaskName	location	subscriptionId
2023-03-08 22:29:39.4111686	03993c18-16cb-4988-ac5b-97241d48449e	26b3723f-2a97-4ae8-a175-c8da07932e6c	HttpOutgoingRequestEndWithSuccess	westus	625bdeee-1ef9-4a36-b6f4-e228
2023-03-08 22:29:39.4127756	03993c18-16cb-4988-ac5b-97241d48449e	26b3723f-2a97-4ae8-a175-c8da07932e6c	HttpOutgoingRequestEndWithSuccess	westus	625bdeee-1ef9-4a36-b6f4-e228

2. Using the RequestID from #1 (asyncId) run the following Kusto query in the region from #1 (location), note that I'm checking the operations with TIMESTAMP a few minutes before the error:

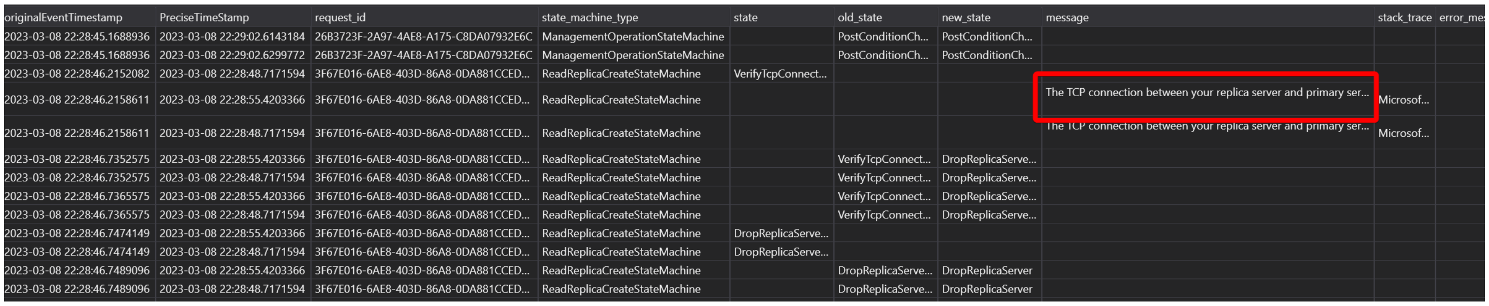
```
MonOrcasBreadthRp
| where TIMESTAMP >= datetime('2023-03-08T22:00:00Z')
| where request_id =~ "26b3723f-2a97-4ae8-a175-c8da07932e6c"
| where (event contains "cancel" or event contains "timeout" or event contains "fail") or isnotempty (message)
| project TIMESTAMP, event, exception_type, stack_trace, message, operation_type, error_code, error_severity,
```



TIMESTAMP	event	exception_type	stack_trace	message	operation_type	error_code	error_severity	error_classification
2023-03-08 22:21:34.0934982	orcasbreadth_fsm_information			FSM RequestId: 26b3723f-2a97-4ae8-a175-c8da07932e6c FSM Id: 3f67e016-6ae8-403d-86a8-0da881ccedea				
2023-03-08 22:21:37.5154171	orcasbreadth_fsm_information			The source server link create FSM Id 0c87bcf1-0a02-4ca6-9e97-3d8c6fb50dd9 , Region: eastus				
2023-03-08 22:29:02.6143184	management_operation_failure				CreateReadReplicaManagementOperation	0	0	Unknown

3. Still in the same location execute the following Kusto query, again note the TIMESTAMP a few minutes earlier than the error occured:

```
let requestid = "26b3723f-2a97-4ae8-a175-c8da07932e6c"; // Request id of replica operation
let ['_startTime']=datetime('2023-03-08T22:00:00Z'); // Start time of the request
let ['_endTime']=now(); //datetime('2022-11-13T03:03:34Z');  // End time of the request, if it in progress use
let replica_request_completed = toscalar(MonOrcasBreadthRp
        | where TIMESTAMP between (_startTime .. _endTime)
        | where request_id =~ requestid
        | where AppTypeName == "OrcasBreadthRp"
        | where operation_type in ("CreateReadReplicaManagementOperation", "PromoteReadReplicaManagementOperat
        | where event in ("management_operation_failure", "management_operation_success")
        | extend replica_drop_operation_id = extract("\\<ReadReplicaDropOperationId\\>(.+?)\\</ReadReplicaDrop
        | extend replica_promote_operation_id = extract("\\<ReadReplicaPromoteOperationId\\>(.+?)\\</ReadRepli
        | extend replica_create_operation_id = extract("\\<ReadReplicaCreateOperationId\\>(.+?)\\</ReadReplica
        | extend replica_request_id = case (
                              isnotempty(replica_create_operation_id), replica_create_operation_id,
                              isnotempty(replica_drop_operation_id), replica_drop_operation_id,
                              isnotempty(replica_promote_operation_id), replica_promote_operation_id,
                              requestid)
        | extend replica_request_id = iff (replica_request_id =~ "00000000-0000-0000-0000-000000000000", reque
        | project replica_request_id);
let replica_request_inprogress = toscalar(MonOrcasBreadthRp
        | where isempty(replica_request_completed)
        | where TIMESTAMP between (_startTime .. _endTime)
        | where request_id =~ requestid
        | where AppTypeName == "OrcasBreadthRp"
        | where state  in ("PerformPreChecks")
        | where event == "orcasbreadth_fsm_information"
        | extend replica_request_id = extract("FSM RequestId: (.+?) FSM Id: (.*)", 2, message)
        | project toupper(trim(" ", replica_request_id)));
let replica_source_request = toscalar(MonOrcasBreadthRp
        | where isnotempty(replica_request_inprogress) or isnotempty(replica_request_completed)
        | where TIMESTAMP between (_startTime .. _endTime)
        | where request_id =~ requestid
        | where AppTypeName == "OrcasBreadthRp"
        | where state in ("NotifyCreateSourceServerLink", "NotifyDropSourceServerLinkCommunication")
        | where event == "orcasbreadth_fsm_information"
        | extend source_request_id = extract("The source server link create FSM Id (.+?) , Region:(.+?)", 1, m
        | extend source_request_id = iff (isempty(trim(" ", source_request_id)), extract("The source server li
        | project toupper(trim(" ", source_request_id)));
let replica_request_id = case (
                          isnotempty(replica_request_completed), replica_request_completed,
                          isnotempty(replica_request_inprogress), replica_request_inprogress,
                          requestid);
let replica_source_request_id = iff(isempty(replica_source_request) or replica_request_id =~ requestid, reques
let requestids = set_union(pack_array(toupper(requestid),toupper(replica_request_id), toupper(replica_source_r
MonOrcasBreadthRp
| where TIMESTAMP between (_startTime .. _endTime)
| where request_id in (requestids)
| where AppTypeName == "OrcasBreadthRp"
//| where isnotempty(message)
| project originalEventTimestamp,PreciseTimeStamp, request_id, state_machine_type, ['state'], old_state, new_s
| order by  originalEventTimestamp asc
```



originalEventTimestamp	PreciseTimeStamp	request_id	state_machine_type	state	old_state	new_state	message	stack_trace	error_me
2023-03-08 22:28:45.1688936	2023-03-08 22:29:02.6143184	26B3723F-2A97-4AE8-A175-C8DA07932E6C	ManagementOperationStateMachine		PostConditionCh...	PostConditionCh...			
2023-03-08 22:28:45.1688936	2023-03-08 22:29:02.6299772	26B3723F-2A97-4AE8-A175-C8DA07932E6C	ManagementOperationStateMachine		PostConditionCh...	PostConditionCh...			
2023-03-08 22:28:46.2152082	2023-03-08 22:28:48.7171594	3F67E016-6AE8-403D-86A8-0DA881CCED...	ReadReplicaCreateStateMachine	VerifyTcpConnect...					
2023-03-08 22:28:46.2158611	2023-03-08 22:28:55.4203366	3F67E016-6AE8-403D-86A8-0DA881CCED...	ReadReplicaCreateStateMachine				The TCP connection between your replica server and primary ser...	Microsof...	
2023-03-08 22:28:46.2158611	2023-03-08 22:28:48.7171594	3F67E016-6AE8-403D-86A8-0DA881CCED...	ReadReplicaCreateStateMachine				The TCP connection between your replica server and primary ser...	Microsof...	
2023-03-08 22:28:46.7352575	2023-03-08 22:28:55.4203366	3F67E016-6AE8-403D-86A8-0DA881CCED...	ReadReplicaCreateStateMachine		VerifyTcpConnect...	DropReplicaServe...			
2023-03-08 22:28:46.7352575	2023-03-08 22:28:48.7171594	3F67E016-6AE8-403D-86A8-0DA881CCED...	ReadReplicaCreateStateMachine		VerifyTcpConnect...	DropReplicaServe...			
2023-03-08 22:28:46.7365575	2023-03-08 22:28:55.4203366	3F67E016-6AE8-403D-86A8-0DA881CCED...	ReadReplicaCreateStateMachine		VerifyTcpConnect...	DropReplicaServe...			
2023-03-08 22:28:46.7365575	2023-03-08 22:28:48.7171594	3F67E016-6AE8-403D-86A8-0DA881CCED...	ReadReplicaCreateStateMachine		VerifyTcpConnect...	DropReplicaServe...			
2023-03-08 22:28:46.7474149	2023-03-08 22:28:55.4203366	3F67E016-6AE8-403D-86A8-0DA881CCED...	ReadReplicaCreateStateMachine	DropReplicaServe...					
2023-03-08 22:28:46.7474149	2023-03-08 22:28:48.7171594	3F67E016-6AE8-403D-86A8-0DA881CCED...	ReadReplicaCreateStateMachine	DropReplicaServe...					
2023-03-08 22:28:46.7489096	2023-03-08 22:28:55.4203366	3F67E016-6AE8-403D-86A8-0DA881CCED...	ReadReplicaCreateStateMachine		DropReplicaServe...	DropReplicaServer			
2023-03-08 22:28:46.7489096	2023-03-08 22:28:48.7171594	3F67E016-6AE8-403D-86A8-0DA881CCED...	ReadReplicaCreateStateMachine		DropReplicaServe...	DropReplicaServer			

Error:
The TCP connection between your replica server and primary server failed. If replica and primary server is in different virtual networks make sure both virtual networks peered. If you create Network Security Groups (NSG) to deny traffic flow to or from your Flexible Server within the subnet where it's deployed, please make sure to allow both inbound and outbound traffic to destination port 5432. Refer to https://docs.microsoft.com/en-us/azure/postgresql/flexible-server/concepts-networking#virtual-network-concepts for more details.

For a read replica in another region, the VNets need to be peered or use a hub-spoke network. Also, make sure the PostgreSQL port (5432) is open in the network firewall.

How good have you found this content?