ARM Templates - Deployment failed with timeout error

Last updated by | Lisa Liu | Nov 6, 2020 at 10:34 AM PST

ARM Templates- Deployment failed with timeout error

Firewall Rule/Virtual Network Rule/Server Parameters/Databases in Single Server Deployment Cancelled/Timed out

Recommended Steps

- If your deployment is failing:
  - If you are creating a new server, make sure your the server name is globally unique
  - If you are deploying or updating multiple server attributes which includes firewall rules, Virtual Network rules, server parameters or databases for a given server, make sure you are deploying these serially, in any order. By default, ARM deploys resources in parallel and a deployment may fail if configured in parallel. Familiarize yourself with Sample ARM templates
  - Required parameters are set and valid. See the Azure Database for PostgreSQL REST API to understand the valid values of the parameters.
- Poll the status of the operation after you issue the request. Most operations are asynchronous and can take a few minutes to complete.

Recommended Documents

- Deploy multiple properties of a server including Firewall rules, Virtual Network rules, server parameters or databases
- Create databases in Azure Database for PostgreSQL
- Create Azure Database for PostgreSQL with multiple firewall rules
- Deploy Azure Database for PostgreSQL with CanNotDelete lock

Deployment failed with timeout error

To troubleshoot this issues you need to query two tables

```
MonManagementOperations
| where elastic_server_name =~ "{ServerName}"
//| where TIMESTAMP >= datetime(2019-08-21 14:37:12.3520000) and TIMESTAMP < now()
| project originalEventTimestamp, request_id, operation_type, ['state'], error_message
| order by originalEventTimestamp desc
```

And then to learn more about the operation collect request ID from previous query and run the following query

```
MonManagement
| where request_id == "168D1E11-1344-4679-8CC2-32BD3EAB2EDF"
```

```
Please note that if the customer ran deployment while another deployment in process it will timeout
```

Example below:
Reason for the issue in the following example:
The reason why the first deployment went through is because server was idle at that time, and all update firewall rule requests takes less time. For the second deployment, server was already up and running, thus, each updating firewall rule takes longer time and hit this issue.
According to the provided template, all firewall rules are deployed in parallel. We suggest the customer to use "dependsOn" element in ARM template to define the order of deployment and avoid failures. Here is the documentation and examples on how to use "dependsOn" element.

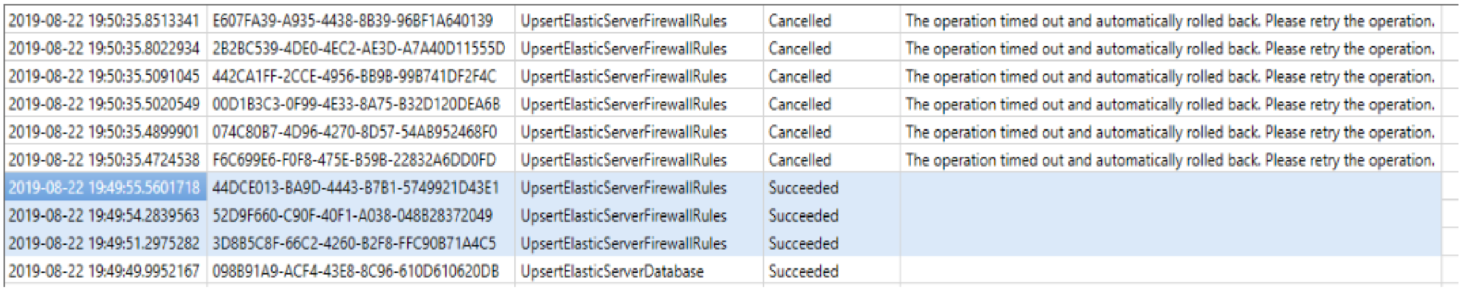
From <https://icm.ad.msft.net/imp/v3/incidents/details/141389847/home>

```
['ERROR: Deployment failed. Correlation ID: c6a9355f-5ac3-4877-ac36-ad1c0644d2c7. {', '  \\'status\\': \\'Canceled\\',', '  \\'error\\': {', '    \\'code\\': \\'ResourceDeploymentFailure\\',', '    \\'message\\': \\'The resource operation completed with terminal provisioning state 'Canceled'.\\',', '    \\'details\\': [', '      {', '        \\'code\\': \\'OperationTimedOut\\',', '        \\'message\\': \\'The operation timed out and automatically rolled back. Please retry the operation.\\'',;
```

The UpsertElasticServerFirewallRules operation failed since 2019-08-21 14:36:48.067.

```
MonManagement
| where request_id == 'F6C699E6-F0F8-475E-B59B-22832A6DD0FD'
| project originalEventTimestamp,state, event, state_machine_type, action, old_state, new_state,message, operation_type,
operation_parameters , elastic_server_name, error , error_message, ClusterName, keys,  stack_trace, fabric_name_uri ,
fabric_service_uri , fabricApplicationUri, correlation_id
| order by originalEventTimestamp asc
```

In this case Customer updated the firewall rule several times within 1 min and the first 3 operation succeeded.
The succeeding operations failed because it was unable to obtain the lock.
The finite state machine of type Microsoft.Xdb.InstanceManager.ElasticServerStateMachine with key(s) of [629f844a-5914-4720-8a36-0cb8c5fbb979] was unable to obtain the lock in 00:00:10.



2019-08-22 19:50:35.8513341	E607FA39-A935-4438-8B39-96BF1A640139	UpsertElasticServerFirewallRules	Cancelled	The operation timed out and automatically rolled back. Please retry the operation.
2019-08-22 19:50:35.8022934	2B2BC539-4DE0-4EC2-AE3D-A7A40D11555D	UpsertElasticServerFirewallRules	Cancelled	The operation timed out and automatically rolled back. Please retry the operation.
2019-08-22 19:50:35.5091045	442CA1FF-2CCE-4956-BB9B-99B741DF2F4C	UpsertElasticServerFirewallRules	Cancelled	The operation timed out and automatically rolled back. Please retry the operation.
2019-08-22 19:50:35.5020549	00D1B3C3-0F99-4E33-8A75-B32D120DEA6B	UpsertElasticServerFirewallRules	Cancelled	The operation timed out and automatically rolled back. Please retry the operation.
2019-08-22 19:50:35.4899901	074C80B7-4D96-4270-8D57-54AB952468F0	UpsertElasticServerFirewallRules	Cancelled	The operation timed out and automatically rolled back. Please retry the operation.
2019-08-22 19:50:35.4724538	F6C699E6-F0F8-475E-B59B-22832A6DD0FD	UpsertElasticServerFirewallRules	Cancelled	The operation timed out and automatically rolled back. Please retry the operation.
2019-08-22 19:49:55.5601718	44DCE013-BA9D-4443-B7B1-5749921D43E1	UpsertElasticServerFirewallRules	Succeeded	
2019-08-22 19:49:54.2839563	52D9F660-C90F-40F1-A038-048B28372049	UpsertElasticServerFirewallRules	Succeeded	
2019-08-22 19:49:51.2975282	3D8B5C8F-66C2-4260-B2F8-FFC90B71A4C5	UpsertElasticServerFirewallRules	Succeeded	
2019-08-22 19:49:49.9952167	098B91A9-ACF4-43E8-8C96-610D610620DB	UpsertElasticServerDatabase	Succeeded	

Customer send multiple update firewall rule requests at the same time.
For example, the cancelled UpsertElasticServerFirewallRules request `F6C699E6-F0F8-475E-B59B-22832A6DD0FD` happened during `datetime(2019-08-22 19:48:59.8370000), datetime(2019-08-22 19:50:35.7910000)`
while another UpsertElasticServerFirewallRules request: `3D8B5C8F-66C2-4260-B2F8-FFC90B71A4C5` is still in process druing `datetime(2019-08-22 19:48:59.7228693), datetime(2019-08-22 19:49:51.3001880)` and finally succeeded.

Solution: ask the customer to run the command serially (one by one)

Created with Microsoft OneNote 2016.

How good have you found this content?