Configuring an EMS Fault Tolerant Environment On Microsoft Azure

This document provides the steps for configuring and testing EMS F/T in a Red Hat Linux or Microsoft Windows Server operating environment in Microsoft Azure.

<table>
<thead>
<tr>
<th>Version .1</th>
<th>Initial Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version .2</td>
<td>Added Microsoft Windows steps</td>
</tr>
<tr>
<td>Version .3</td>
<td>Linux Encryption (seal) Support added</td>
</tr>
</tbody>
</table>
Copyright Notice
COPYRIGHT© 2018 TIBCO Software Inc. This document is unpublished and the foregoing notice is affixed
to protect TIBCO Software Inc. in the event of inadvertent publication. All rights reserved. No part of this
document may be reproduced in any form, including photocopying or transmission electronically to any
computer, without prior written consent of TIBCO Software Inc. The information contained in this document
is confidential and proprietary to TIBCO Software Inc. and may not be used or disclosed except as
expressly authorized in writing by TIBCO Software Inc. Copyright protection includes material generated
from our software programs displayed on the screen, such as icons, screen displays, and the like.

Trademarks
All brand and product names are trademarks or registered trademarks of their respective holders and are
hereby acknowledged. Technologies described herein are either covered by existing patents or patent
applications are in progress.

Confidentiality
The information in this document is subject to change without notice. This document contains information
that is confidential and proprietary to TIBCO Software Inc. and its affiliates and may not be copied,
published, or disclosed to others, or used for any purposes other than review, without written authorization
of an officer of TIBCO Software Inc. Submission of this document does not represent a commitment to
implement any portion of this specification in the products of the submitters.

Content Warranty
The information in this document is subject to change without notice. THIS DOCUMENT IS PROVIDED "AS
IS" AND TIBCO MAKES NO WARRANTY, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT
NOT LIMITED TO ALL WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR
PURPOSE. TIBCO Software Inc. shall not be liable for errors contained herein or for incidental or
consequential damages in connection with the furnishing, performance or use of this material.

Export
This document and related technical data, are subject to U.S. export control laws, including without limitation
the U.S. Export Administration Act and its associated regulations, and may be subject to export or import
regulations of other countries. You agree not to export or re-export this document in any form in violation of
the applicable export or import laws of the United States or any foreign jurisdiction.

For more information, please contact:
TIBCO Software Inc.
3303 Hillview Avenue
Palo Alto, CA 94304
USA
Table of Contents

Configuring an EMS Fault Tolerant Environment On Microsoft Azure ............................................. 1

1 Overview ........................................................................................................................................ 5
    1.1 Document Purpose .................................................................................................................. 5
    1.2 Assumptions .......................................................................................................................... 5

2 Azure Setup for Linux .................................................................................................................. 6
    2.1 Creating a new Virtual Machine ............................................................................................ 6
    2.2 Setting Up the CIFS/SMB File System .................................................................................. 8
    2.3 Setting up the Red Hat VM for EMS F/T ............................................................................. 11
        2.3.1 Additional Software Installation .................................................................................... 11
        2.3.2 Linux Kernel changes .................................................................................................... 12

3 Azure Setup for Windows Server ................................................................................................ 14
    3.1 Creating a new Virtual Machine ............................................................................................ 14
    3.2 Setting Up the CIFS/SMB File System .................................................................................. 15
    3.3 Setting up the Windows Server for EMS F/T ....................................................................... 15

4 EMS Installation and Configuration ............................................................................................. 17
    4.1 EMS Installation ..................................................................................................................... 17
    4.2 EMS Configuration ................................................................................................................ 17
        4.2.1 Stores.conf .................................................................................................................... 17
        4.2.2 Factories.conf ............................................................................................................... 18
        4.2.3 Tibemsd.conf ................................................................................................................ 18
        4.2.4 Starting the EMS Instances .......................................................................................... 19

5 Testing EMS Fault Tolerance on Azure ..................................................................................... 21
    5.1 EMS Client App Setup .......................................................................................................... 21
    5.2 Performing the EMS Fault Tolerant Test Cases .................................................................. 22
        5.2.1 EMS Process Failure Test .............................................................................................. 22
        5.2.2 Network Failure Test on Linux ...................................................................................... 25
        5.2.3 Network Failure Test on Windows ............................................................................... 27
        5.2.4 System Failure Test ...................................................................................................... 30
Table of Figures

FIGURE 1 - MICROSOFT AZURE DASHBOARD ...........................................................................................................6
FIGURE 2 - CREATE RED HAT LINUX 7.3 VM...............................................................................................................7
FIGURE 3 - VM CREATION INPUTS ............................................................................................................................7
FIGURE 4 - CREATE STORAGE ACCOUNT ..................................................................................................................9
FIGURE 5 - AZURE STORAGE ACCOUNTS ..............................................................................................................10
FIGURE 6 - STORAGE ACCOUNT ACCESS KEYS ......................................................................................................10
FIGURE 7 - FSTAB EXAMPLE ......................................................................................................................................12
FIGURE 8 - DF EXAMPLE ............................................................................................................................................12
FIGURE 9 - MOUNT EXAMPLE ...................................................................................................................................12
FIGURE 10 - SYSCONFIG EXAMPLE ..........................................................................................................................13
FIGURE 11 - CONNECT COMMAND FROM AZURE DASHBOARD ...........................................................................15
FIGURE 12 - CONNECTING TO THE FILE SHARE ......................................................................................................16
FIGURE 13 - STORES,CONF EXAMPLE ........................................................................................................................18
FIGURE 14 - CONNECTION FACTORY SETTINGS ......................................................................................................18
FIGURE 15 - LINUX EMS STARTUP ..........................................................................................................................20
FIGURE 16 - WINDOWS EMS STARTUP ......................................................................................................................20
FIGURE 17 - CREATE THE SYNC QUEUE ...................................................................................................................22
FIGURE 18 - RUNNING TIBEMSMSGPRODPERF .........................................................................................................23
FIGURE 19 - STANDBY EMS BECOMING ACTIVE ON TIBEMS2 ...........................................................................24
FIGURE 20 - PURGE THE SYNC QUEUE FROM TIBEMSADMIN64 ........................................................................25
FIGURE 21 - DROP_CIFS.SH SCRIPT ........................................................................................................................26
FIGURE 22 - RUNNING DROP_CIFS.SH .....................................................................................................................26
FIGURE 23 - DISK WRITE ERROR ON TIBEMS1 .........................................................................................................27
FIGURE 24 - FIREWALL PANE ....................................................................................................................................28
FIGURE 25 - NEW BLOCKOUT445 FIREWALL RULE ...................................................................................................29
FIGURE 26 - DISK WRITE ERROR: S9 ON WIN2012A ..................................................................................................30
FIGURE 27 - TIBEMS1 VM PAGE IN THE AZURE DASHBOARD ..................................................................................31
FIGURE 28 - STANDBY EMS INSTANCE RECOVERING FROM SYSTEM FAILURE OF PRIMARY ................................32
1 Overview

1.1 Document Purpose
The purpose of the document is to provide a guide to install, configure, and run EMS in a fault-tolerant configuration on Microsoft Azure. In addition, the document will provide the steps and expected results for testing EMS F/T on Azure.

The document will outline:

- Setting up Red Hat Linux virtual images on Azure
- Setting up Microsoft Windows Server 2012 R2 virtual images on Azure
- Setting up the shared file system in Azure
- Setting up the CIFS/SMB mount on the Red Hat Linux and Windows Server VMs
- Installing and configuring EMS for F/T on both platforms
- Tuning EMS for Azure
- Running tests on both platforms for:
  - EMS process failure
  - Network failure between the VM running EMS and the shared storage
  - Accidental VM reboot from Azure Dashboard

1.2 Assumptions
The reader of this document is familiar with the following concepts:

- The use of Microsoft Azure
- TIBCO EMS installation and configuration
- Red Hat Linux configuration
- Microsoft Windows Server 2012 R2 configuration

- Document only provides information for Red Hat Linux. Other Linux kernels should be similar
- Document only provides information for Windows Server 2012 R2. However, the same procedures should work with Windows Server 2016
- Windows Server login is done from another Windows machine using Remote Desktop Connection
2 Azure Setup for Linux

The following steps will outline setting up the virtual images on Microsoft Azure. Red Hat Linux 7.3 was used for the virtual machines. Other operating systems or versions are not covered.

2.1 Creating a new Virtual Machine

- Login into the Microsoft Azure Portal
- To create a VM, use the following:
  - In the Azure Dashboard, select the "Virtual Machines" button. Eighth Icon Down. Three VMs are needed: two for EMS F/T and one for the client application.
  - Select the Add button.
  - Select Red Hat Enterprise Linux, then 7.5 (or most current version), use Resource Manager, and then the create button at the bottom of the RH page.
Select:

- Name for the VMs (will need to create 3). For this environment, *tibems1*, *tibems2*, and *tibems3* were created.
- Disk Type SSD/HDD – SSD performs better, but costs more to use.
- Username - Use the same for all VMs.
• Authentication type: Password/SSH Public Key – Password is easiest for non-production environments. Others can then just access the VMs with username/password
• Password - Use the same for all VMs
• Confirm Password
• Subscription - This should be provided as part of your Azure login ID.
• Resource group - For first VM, create a new RG. For all subsequent VMs, USE the SAME Resource Group – VERY IMPORTANT!
• Select Location
  o Click OK
  o Next Screen will be for "Choose a size" for the VM.
  o Select the "View All" button to see the different size VMs available. Pick the appropriate size for the environment
  o Click the Select button
  o Under Settings:
    • Select "No" for Use managed disks
    • Under Storage Accounts:
      - Create a new storage account for the first VM. Use the SAME Storage Account for all subsequent VMs - IMPORTANT
      - Use standard or premium. This is the VM disk for the VM
    • Create a new virtual network for the first VM. Use the SAME Virtual Network for all subsequent VMs. - IMPORTANT
    • Select the default Subnet. Use the SAME Subnet for all subsequent VMs. - IMPORTANT
    • Use the default Public IP address provided - must be different for all VMs.
    • Use the default Network security group (firewall) - must be different for all VMs.
    • Select "None" for High Availability
    • Select the defaults for Monitoring
    • Create the Diagnostics storage account for the first VM. Use the SAME Diagnostics storage account for all subsequent VMs – IMPORTANT
  o Click OK
• Once the VM has started, Use Putty (or ssh on Mac/Linux/UNIX) to access it using the public IP address created by Azure, and the username/password (if used) defined during the VM creation. If username/password are used, a certificate if not required.

2.2 Setting Up the CIFS/SMB File System

To setup the CIFS/SMB file system to mount on your Linux or Windows VM, do the following:
• From the main Azure Dashboard, select the "Storage Accounts" button on the left of the screen. Tenth button down. Then, select Add.
Under "Create storage account"

- Provide a new name, such as "emsdata2"
- Select "Resource manager" for Deployment model
- Select "Standard" for Performance. Standard MUST be used to have "File" access. Premium only provides "Blob", which will not work for EMS shared storage. - IMPORTANT
- Select the default for Replication
- Select "Disabled" for encryption
- Select "Disabled" for Secure transfer required
- Select the Subscription. MUST be the same as the VMs. - IMPORTANT
- Select the Resource Group. This MUST be the SAME as what was used for the VMs. - IMPORTANT
- Select the Location. This MUST be the SAME as what was used for the VMs. - IMPORTANT

Click OK

Once the new Storage Account is created:

- Should still be on the storage account page in the dashboard
  - Refresh the page
  - Select the newly created storage account - Will need this name later for the mount command in Linux or for mounting the share in Windows
Click on Files
Click on the + at the top of the screen to add a new file share
Select a new name - will need this later for the mount in Linux/Windows
Select the size for the file share in GB
  Click OK to create the new File Share
  Go back to the Storage Accounts screen, and click on the newly created account
  Click on Access keys
  Take note/“click to copy” Key1. it is LONG. - Will need this for the mount command in Linux or to mount the share on Windows.
  For a Linux environment, everything from the Azure Dashboard is complete, and can proceed to the next section. For Windows Servers, the Storage Accounts dashboard will be required in Section 3, and should kept up

Figure 6 - Storage Account Access Keys
2.3 Setting up the Red Hat VM for EMS F/T

Use the following to setup two of the newly created Red Hat virtual images for EMS fault tolerance. All steps must be completed on both VMs.

**Note:** Red Hat 5.7 (Maipo) now supports CIFS/SMB encryption for full end to end encryption for EMS data. Only Red Hat 7.5 supports this feature.

### 2.3.1 Additional Software Installation

- **Login to the VM using SSH.** Use the public IP address provided by Azure, and the username/password defined when creating the VM.
  - **Update the VM to the latest RH version, for Java, and the cifs-utils.** *Note:* This step must be performed on the Red Hat virtual image used for the EMS client also.
    - `sudo yum update`
    - `sudo yum install cifs-utils`
    - `sudo yum install java-devel`
  - **Create a new mount point on the virtual image.**
    - `sudo mkdir /mnt/data2`
  - **Mount the CIFS/SMB share created in Azure:**
    - `sudo mount -t cifs //<myaccountname>.file.core.windows.net/<mysharename>./<mymountpoint> -o seal,vers=3.0,username=<myaccountname>,password=<Really long key from access keys>,dir_mode=0777,file_mode=0777,serverino,nostrict sync`
      
      - `<myaccountname>` is the Azure Storage Account Name
      - `<mysharename>` is the file share created in the Azure Storage Account
      - `<mymountpoint>` is the new created mount point on the VM
      - `password` is the really long key copied from the access key for the file share under the Azure Storage account
    
    *Note:* Optionally, the username/password can be added to a file, and the file referenced in the CIFS mount. See the Microsoft documentation for further details.

- `nostrict sync` provides for better write performance. – **IMPORTANT**

- `seal` enables encryption between the server and the file share. If encryption is not needed, or a version of RH Linux prior to 7.5 is used, this option can be omitted. – **OPTIONAL**

- The mount can also be added to `/etc/fstab` to make it permanent. The following is an example of the `/etc/fstab` with the CIFS mount *without the seal option.*

```bash
# /etc/fstab
Created by anaconda on Tue May 30 18:01:31 2017

# Accessible filesystems, by reference, are maintained under '/dev/disk'
See man pages fstab(8), findfs(8), mount(8) and/or b Nit(8) for more info

UUID=36775ddad-df9d-4bc6-a3a5-f6e23b63097f / xfs defaults 0 0
UUID=6a3a6ab4-b6a0-4e0b-89e3-60ae8036a42f /boot xfs defaults 0 0
/dev/sda1 /mnt/data2 cifs vers=3.0,dir_mode=0777,fil
```
Figure 7 - Fstab example


Use the `mount` and the `df` commands to verify the CIFS file share is mounted. Examples do not show the mount with the `seal` option.

```
[tiibco@tibems1 ~]
```

Figure 8 - df example

Figure 9 - Mount example

Disable the firewall all three Red Hat virtual images. If not, the EMS heartbeats will not be seen. Alternatively, use `iptables` to configure the firewall between the virtual images.

- `sudo systemctl stop firewalld`
- `sudo systemctl disable firewalld`

2.3.2 Linux Kernel changes

The Linux kernel by default can keep the `tcp_keepalives` for up to twenty minutes. This can have a delayed affect on EMS fail-over. To shorten this time, the Linux kernel property `tcp_retries2` can be modified. To modify this property, do the following:

- Edit `/etc/sysctl.conf`, and add the following value: `net.ipv4.tcp_retries2 = 4`

Following is an example of `/etc/sysctl.conf` with the change:

```ini
sysctl settings are defined through files in
# /usr/lib/sysctl.d/, /run/sysctl.d/, and /etc/sysctl.d/.
#
# Vendors settings live in /usr/lib/sysctl.d/.
# To override a whole file, create a new file with the same in
# /etc/sysctl.d/ and put new settings there. To override
# only specific settings, add a file with a lexically later
# name in /etc/sysctl.d/ and put new settings there.
#
# For more information, see sysctl.conf(5) and sysctl.d(5).
# Update to shorten tcp keep alives
net.ipv4.tcp_retries2 = 4

Figure 10 - Sysctl.conf example

- Reboot the virtual image/system
3 Azure Setup for Windows Server

The following steps will outline setting up the Windows Server virtual images on Microsoft Azure. Windows Server 2012 R2 was used. Windows Server 2016 will be similar.

3.1 Creating a new Virtual Machine

- Login into the Microsoft Azure Portal
- To create a VM, use the following:
  - In the Azure Dashboard, select the "Virtual Machines" button. Eighth Icon Down. Three VMs are needed: two for EMS F/T and one for the client application. Figure 1 is an example of the Azure Dashboard.
  - Select the Add button.
  - Select the desired Microsoft Server image. For this configuration and testing, Windows Server 2012 R2 Datacenter was used. Click on Resource Manager, and then the create button at the bottom of the RH page.
  - Select:
    - Name for the VMs (will need to create 3). For this environment, win2012a and win2012b were created. In this document, a RH Linux VM created previously, was used for the client apps
    - Disk Type SSD/HDD
    - Username - Use the same for all VMs
    - Authentication type: Password/SSH Public Key – Password is easiest for non-production environments. Others can then just access the VMs with username/password
    - Password - Use the same for all VMs
    - Confirm Password
    - Subscription - This should be provided as part of your Azure login ID.
    - Resource group - For first VM, create a new RG. For all subsequent VMs, USE the SAME Resource Group – VERY IMPORTANT!
    - Select Location
    - If a Windows Server 2012 license is available, click on Already have a Windows Server License?", and then confirm. If not, select No
  - Click OK
  - Next Screen will be for "Choose a size" for the VM.
  - Select the "View All" button to see the different size VMs available. Pick the appropriate size for the environment
  - Click the Select button
  - Under Settings:
    - Select "No" for Use managed disks
    - Under Storage Accounts:
      - Create a new storage account for the first VM, if one doesn’t already exist. Use the SAME Storage Account for all subsequent VMs - IMPORTANT
      - Use standard or premium. This is the VM disk for the VM
    - Create a new virtual network for the first VM, if one doesn’t already exist. Use the SAME Virtual Network for all subsequent VMs. - IMPORTANT
    - Select the default Subnet. Use the SAME Subnet for all subsequent VMs. - IMPORTANT
    - Use the default Public IP address provided - must be different for all VMs
    - Use the default Network security group (firewall) - must be different for all VMs
    - Select "None" for High Availability
    - Select the defaults for Monitoring
• Create the Diagnostics storage account for the first VM, if one doesn’t already exist. Use the SAME Diagnostics storage account for all subsequent VMs – IMPORTANT
  o Click OK
• Once the VM has been created and started, go back to the main Azure Dashboard, and then click on the newly create Windows VM. Click on the Connect button on the top of the Virtual Machine dashboard. This will download and start a new Remote Desktop Connection to the appropriate public IP address created by Azure. Login using the username/password that were previously created. 
  Note: If using a Mac, or Linux desktop, use the appropriate application to login to the Windows virtual machine using the public IP address provided by Azure.

3.2 Setting Up the CIFS/SMB File System
To setup the CIFS/SMB file system to follow the steps outlined in Section 2 on setting up the file system. Make sure to copy Key 1.

3.3 Setting up the Windows Server for EMS F/T
The Windows Server virtual machine will have everything installed and configured for TIBCO EMS. Only the file share created previously needs to be mounted in order to install and configure EMS. The following steps will be required on both Windows Servers used for the F/T EMS servers.
  • Use Remote Desktop Connection to login to one of the newly created Windows Server virtual machines.
  • Once logged into the Windows Server, open a new PowerShell window.
  • Go back to the Storage Accounts dashboard in the Azure portal, click on the Files button, and then on the appropriate file share.
  • Click on the Connect button. On the right side of the dashboard the command to connect the file share for a Windows Server will be displayed, along with the copy button. The following figure shows an example:

![Figure 11 - Connect command from Azure Dashboard](image)

  • Click on the copy button to copy the command
  • Paste the command in the PowerShell window, and replace [drive letter] with an appropriate value, such a “Z:”. Run the command. Should get The command completed successfully.
  
  Note: It is highly recommended that after running the command, open up notepad on the VM, and paste and save the command in an easily accessible file. When the Windows Server VM is stopped and restarted, connection to the share is NOT restored. The command must be re-run, or the share must be re-connected from File Explorer every time the Windows Server is restarted. Having the command and/or password in a text file will simplify restoring the share.
Figure 12 - Connecting to the file share

- From the File Explorer, the directories for the EMS shared configuration files can now be created on the file share.
- Perform the same steps on the second Windows Server virtual machine.
4 EMS Installation and Configuration

This section will outline the installation and configuration of EMS on Linux and Windows.

4.1 EMS Installation

Install EMS on all virtual image, either Linux or Windows. EMS 8.4 must be used. Nothing specific or custom is required to the base configuration of EMS on either platform, so a typical EMS installation can be performed.

Once EMS is installed, use the following to configure EMS for F/T on Azure.

- On one of the VMs used for the EMS server:
  - Create the directory on the Azure file system for the shared EMS configuration files and data stores.
    Ex: /mnt/data2/tibco/cfgmgmt/ems/data and /mnt/data2/tibco/cfgmgmt/ems/data/datastore on Linux, or Z:\tibco\cfgmgmt\ems\data and Z:\tibco\cfgmgmt\ems\data\datastore on Windows.
  - Copy the all EMS configuration files (.conf) installed during the EMS installation to the newly created directory.
  - Copy the tibemsd.conf to $TIBCO_HOME/ems/8.4/bin
  - Create the $TIBCO_HOME/ems/8.4/bin/logs directory

- On the second VM used for the EMS server:
  - Copy the tibemsd.conf to $TIBCO_HOME/ems/8.4/bin
  - Create the $TIBCO_HOME/ems/8.4/bin/logs directory

4.2 EMS Configuration

There are specific configuration changes which must be made to provide better write performance and reliability of EMS F/T on Azure. This section will discuss these changes. See the EMS User Guide for additional information on setting or the use of, any properties discussed.

4.2.1 Stores.conf

In stores.conf, modify/add the following:

- The file_minimum=xxGB should be added to each synchronous data store. By adding this property, EMS will pre-allocate the space on the shared storage the data store. This will provide a better message write throughput on disk. The minimum should be 1GB. Expect the initial startup of EMS to take longer as it creates and allocates the space for the store file.

- The file_crc=enabled should be added. The enables EMS to check for data integrity of the data store. This is now the default on EMS 8.4.

The following is an example of stores.conf with the changes.

```
[$sys.failsafe]
  type=file
  file=sync-msgs.db
  mode=sync
  file_minimum=2GB
  file_crc=enabled

[sync2]
  type=file
  file=sync2-msgs.db
  mode=sync
  file_minimum=2GB
  file_crc=enabled
```
4.2.2 Factories.conf

The EMS client reconnect properties must be set to enable the EMS client to reconnect to the EMS server in the event of an EMS server failure in an F/T configuration. The reconnect properties can be defined in a number of ways, including in the java/C code, TIBCO application’s configuration file, and/or through the connection factory when they are used.

The default values are too low in Azure to reliably allow the EMS client to reconnect to the EMS server after a failover, especially with network or system failure.

It is recommended to set the reconnect_attempt_count to 100, and the reconnect_attempt_delay to 5000. With these values, the EMS client will attempt to reconnect 100 times, every 5 seconds.

The following example shows the values for the FTConnectionFactory in factories.conf.

Note: In the following example for the url, <server1> is tibems1 and the <port1> is 7222, and <server2> is tibems2 and the <port2> is 7222. Substitute with the appropriate values for the environment.

```
[FTConnectionFactory]
type               = generic
url                = tcp://tibems1:7222,tcp://tibems2:7222
reconnect_attempt_count = 100
reconnect_attempt_delay = 5000
```

4.2.3 Tibemsd.conf

The tibemsd.conf for both EMS Servers needs to be updated for multiple properties. The examples shown are for a Linux based tibemsd.conf. For a Windows based environment, instead of “/mnt/data2”, it could be “Z:”.

These include:

- Location of all configuration files – The location must be on the Azure CIFS/SMB shared storage device.

```
# Configuration files.
########################################################################
users                   = "/mnt/data2/tibco/cfgmgmt/ems/data/users.conf"
groups                  = "/mnt/data2/tibco/cfgmgmt/ems/data/groups.conf"
topics                  = "/mnt/data2/tibco/cfgmgmt/ems/data/topics.conf"
queues                  = "/mnt/data2/tibco/cfgmgmt/ems/data/queues.conf"
acl_list                = "/mnt/data2/tibco/cfgmgmt/ems/data/acl.conf"
factories               = "/mnt/data2/tibco/cfgmgmt/ems/data/factories.conf"
routes                  = "/mnt/data2/tibco/cfgmgmt/ems/data/routes.conf"
bridges                 = "/mnt/data2/tibco/cfgmgmt/ems/data/bridges.conf"
transports              = "/mnt/data2/tibco/cfgmgmt/ems/data/transports.conf"
tibrvcm                 = "/mnt/data2/tibco/cfgmgmt/ems/data/tibrvcm.conf"
durables                = "/mnt/data2/tibco/cfgmgmt/ems/data/durables.conf"
channels                = "/mnt/data2/tibco/cfgmgmt/ems/data/channels.conf"
stores                  = "/mnt/data2/tibco/cfgmgmt/ems/data/stores.conf"
########################################################################

# Persistent Storage.
#
# store: directory to store persistent messages.
########################################################################
store                   = "/mnt/data2/tibco/cfgmgmt/ems/data/datastore1"
```

- Log File location – The location must be on the local disk for the VM. The following example locates in the ems/8.4/bin/logs directory.
logfile = "logs/tibemsd2-1.log"

- Server and Client Heartbeat and timeout values – These properties determine how long the client/server listen for the heartbeat from the the client/server, before disconnecting. These properties must be set longer than what is normally configured for a local F/T environment. The values shown below work well on Azure.
  
  server_heartbeat_client = 10
  server_timeout_client_connection = 120
  client_heartbeat_server = 10
  client_timeout_server_connection = 120

- Enabling exiting disk error property – New property for EMS 8.4. This property defines to EMS to exit when there is a disk error reading/writing to the shared storage device. This property will help prevent “Dual Active Server” conditions, sometimes seen in networked storage devices.
  
  always_exit_on_disk_error = enable

- FT properties – Normal properties for the defining the peer EMS server instance, heartbeat between instances, and etc.

- Define a value for destination_backlog_swapout. This will help limit excessive reads to the shared disk. A minimum of 10000 is recommended. If the queues, will persistent a larger number of messages, increase the size.
  
  destination_backlog_swapout = 10000

### 4.2.4 Starting the EMS Instances

Once the configuration files are updated, EMS can be started. Ensure the --forceStart parameter is used when starting EMS. Start both instances, taking note of which instance is the active EMS instance.

**Note:** Leave the both windows to the EMS server instances open. This will be needed for the testing.
Figure 15 – Linux EMS Startup

Figure 16 – Windows EMS Startup
5 Testing EMS Fault Tolerance on Azure

Once EMS has been started on the Azure Red Hat or Windows VMs, the failover testing can be performed.

This section will outline several test cases, including EMS Server process failure, Machine failure, and network failure. Tests are performed using queues with persistence set. This guarantees that the shared file system will be accessed during the tests.

Note: The testing for the process and machine failure for Linux and Windows are the same. The network tests, differ, and will be noted.

5.1 EMS Client App Setup

The third virtual machine (Linux or Windows) is used to run the test applications. EMS is shipped with sample Java applications which can be used for the testing. The tibjmsMsgProducerPerf utility should be used for the testing. All samples Java application are located in $TIBCO_HOME/ems/8.4/samples/java. Use the following to setup the environment:

- Ensure the Java 1.8 development environment is installed.
- Install EMS 8.4 on the third virtual machine following the EMS installation procedures.
- After the installation of EMS is completed:
  - Cd to $TIBCO_HOME/ems/8.4/samples/java
  - . ./setup.sh on Linux, or ./setup.bat on Windows
  - javac *.java – This should compile all java apps in the directory

- Ensure that at least one of the EMS server instances is running (both should be running)
- Use the TIBCO EMS Administration Tool to create the EMS Queue sync utilizing the $sys.failsafe data store. This is required for testing with a synchronous data store:
  - Cd to $TIBCO_HOME/ems/8.4/bin
  - ./tibemsadmin tcp://<server>:port

Note: In the following examples, <server> is win2012a and the <port> is 7222. Substitute with the appropriate values for the environment.
5.2 Performing the EMS Fault Tolerant Test Cases

Three different tests should be performed:

1. EMS Process failure – Active EMS is stopped
2. Network failure – Network failure between the Active EMS Server machine, and the Azure File Share
3. System failure – Accidental restart of the Azure Red Hat Linux VM running the Active EMS server instance

This section will outline how to run these three tests, and what the expected results should be.

Note: All test cases must be run from the third virtual machine where the java sample app was compiled.

5.2.1 EMS Process Failure Test

This test verifies that an EMS client continues to function correctly, with no message loss during an EMS server process failover.

Two EMS server instances will be running in a F/T configuration, while messages are being sent. The active EMS instance will be stopped, the stand-by EMS instance should take over, and continue processing messages until the EMS java completes publishing messages.

Note: In the following examples, <server1> is tibems1 and the <port1> is 7222, and <server2> is tibems2 and the <port2> is 7222. Tibems3 is the VM running the java apps. Substitute with the appropriate values for the environment. The examples show the Linux environment. However, the test on Windows is the same.

5.2.1.1 Running the Process Failure Test

- Three ssh terminal/PowerShell sessions are needed for this test; one for tibems1, one for tibems2, and one for tibems3
- Start EMS on tibems1 and tibems2 in the foreground. EMS on tibems1 should be the active EMS instance.
- From tibems3, start the java application
  - Cd to $TIBCO_HOME/ems/8.4/samples/java
  - ../setup.sh
figure 18 - running tibjmsMsgProducerPerf

- Immediately kill/stop the EMS instance on tibems1, with cntrl-C
- The standby EMS instance on tibems2 will become active, and recover all messages. It should be possible to stop and start the EMS instances a few times while the java test application is running. The number of recovered messages will increase.
Figure 19 - Standby EMS becoming active on tibems2

- After the java application completes, run `tibemsadmin64 tcp://tibems2:7222` (or `tibems1`, if it is active), to verify that there is a minimum of 20000 messages in the sync queue.
- Restart the EMS instance on `tibems1`, and stop the EMS instance on `tibems2`. EMS on `tibems1` should become active, and recover all 20K messages with no errors.
- Use `tibemsadmin64`, and `purge` the sync queue in preparation for the next test.
Stop and restart EMS on tibems1 and tibems2 in the foreground. EMS on tibems1 should be the active EMS instance.

5.2.1.2 Expected Results
The java test application should complete, with a slight pause during failover but should resume sending messages once the failover is complete. No messages should be lost. There may be more than 20K messages, but never should there be less than 20K. Depending on the number of messages that must be recovered, the fail-over should be within a few seconds.

5.2.2 Network Failure Test on Linux
This test verifies that an EMS client continues to function correctly, with no message loss during a network failure between the active EMS server instance, and the Azure shared file system.

Two EMS server instances will be running in a F/T configuration, while messages are being sent. The TCP port will be blocked between then active EMS instance and the Azure file system via `iptables`. The active EMS instance should get a write error, and exit, allowing the stand-by EMS instance to gain the locks on the EMS data stores, and take over. The EMS java application should continue processing messages until it completes.

Note: In the following Linux examples, <server1> is tibems1 and the <port1> is 7222, and <server2> is tibems2 and the <port2> is 7222. Tibems3 is the VM running the java apps. Substitute with the appropriate values for the environment.

5.2.2.1 Running the Network Failure Test
- Four `ssh` terminal sessions are needed for this test; two for tibems1, one for tibems2, and one for tibems3
- A script will be needed to block the CIFS ports on tibems1 while the java app is publishing messages. The following figure shows the `drop_cifs.sh` script. Cut and past the following to create the script. The script must be created in the second ssh terminal window on tibems1.
Figure 21 - drop_cifs.sh script

- From tibems3, start the java application
  - Cd to STIBCO_HOME/ems/8.4/samples/java
  - . ./setup.sh
    FTConnectionFactory --delivery PERSISTENT --connections 10 --threads 8 --count 20000 --size 1000
    --queue sync

- From the second ssh terminal window on tibems1, run sudo drop_nfs.sh

  [[tibco@tibems1 ~]$ sudo ./drop_cifs.sh
  [sudo] password for tibco:
  Saving existing IP table definitions
  Thu Aug 10 19:23:49 UTC 2017
  Sleeping 2 minutes...
  Restoring original IP table definitions
  Done.

  Figure 22 - Running drop_cifs.sh
The active EMS instance on \textit{tibems1} should terminate with a disk write error:

\begin{verbatim}
2017-08-10 19:23:02.183 Process started from "/tibemsd64".
2017-08-10 19:23:02.184 Process Id: 14335
2017-08-10 19:23:02.184 Hostname: tibems1
2017-08-10 19:23:02.184 Hostname IP address: [fe80::28d:3aff:fe19:5396]
2017-08-10 19:23:02.184 Hostname IP address: 10.0.0.4
2017-08-10 19:23:02.184 Reading configuration from 'tibemsd2-1.conf'.
2017-08-10 19:23:02.187 Logging into file 'logs/tibemsd2-1.log'
2017-08-10 19:23:02.189 Server name: 'EMS-SERVER-AZURE'.
2017-08-10 19:23:02.189 Storage Location: '/mnt/data2/tibco/cfg_mgmt/ems/data/datastore1'.
2017-08-10 19:23:02.189 Routing is disabled.
2017-08-10 19:23:02.189 Authorization is disabled.
2017-08-10 19:23:02.189 The server will attempt to trace warnings about destinations that are growing unbounded above 53687091 bytes or 50000 messages.
2017-08-10 19:23:02.189 Set server properties 'large_destination_memory' and 'large_destination_count' respectively to alter these thresholds.
2017-08-10 19:23:03.717 Active server 'tcp://tibems2:7222' not found.
2017-08-10 19:23:03.717 Continuing as active server.
2017-08-10 19:23:03.717 Accepting connections on tcp://tibems1/0.0.0.0:7222.
2017-08-10 19:23:03.717 Recovering state, please wait.
2017-08-10 19:23:05.085 Server is active.
2017-08-10 19:23:12.049 Standby server 'EMS-SERVER-AZURE=tibems2' has connected.
2017-08-10 19:24:18.748 FATAL: Exiting on non-retryable disk error: 5
[tibco@tibems1 bin]$ !
\end{verbatim}

\textit{Figure 23 - Disk Write Error on tibems1}

The standby EMS instance on \textit{tibems2} should determine EMS on \textit{tibems1} is no longer producing a heartbeat, will attempt to become active. Depending on the amount of data, this may take several minutes. There can be other warnings, especially is it take more then than the 120 seconds set for the timeout in \textit{tibemsd.conf}.

After the java application completes, run \textit{tibemsadmin64 tcp://tibems2:7222} (or \textit{tibems1, if it is active}), to verify that there is a minimum of 20000 messages in the sync queue.

Restart the EMS instance on \textit{tibems1}, and stop the EMS instance on \textit{tibems2}. EMS on \textit{tibems1} should become active, and recover all 20K messages with no errors.

While still in \textit{tibemsadmin64}, purge the sync queue in preparation for the next test.

Stop and restart EMS on \textit{tibems1} and \textit{tibems2} in the foreground. EMS on \textit{tibems1} should be the active EMS instance.

\subsection*{5.2.2.2 Expected Results}

The java test application should complete, pausing during the failover, but should resume sending messages once the failover is complete. No messages should be lost. There can be more than 20K messages, depending on the number of connections/threads, but never should there be less than 20K. Depending on the number of messages that must be recovered, the fail-over can take from a few to several minutes. It has been observed with the Azure file system that with the network failure test, the EMS recovery of the messages is much slower then with the other fail-over tests.

\subsection*{5.2.3 Network Failure Test on Windows}

This test verifies that an EMS client continues to function correctly, with no message loss during a network failure between the active EMS server instance on Windows, and the Azure shared file system.

Two EMS server instances will be running in a F/T configuration, while messages are being sent. The TCP port will be blocked between then active EMS instance and the Azure file system via a Windows Firewall rule. The active EMS instance should get a write error, and exit, allowing the stand-by EMS instance to gain the locks on the EMS data stores, and take over. The EMS java application should continue processing messages until it completes.
Note: In the following examples, <server1> is win2012a and the <port1> is 7222, and <server2> is win2012b and the <port2> is 7222. Tibems3 is a Linux VM running the java apps. Substitute with the appropriate values for the environment.

5.2.3.1 Running the Network Failure Test

- Two PowerShell sessions are needed for this test; one for win2012a, one for win2012b. An ssh terminal will be required for tibems3, since it is a Linux VM.
- A Windows Firewall rule will be needed to block the CIFS port on win2012a while the java app is publishing messages. Use the following procedures to create the firewall rule.
  - From the Windows Start button, click on Administrative Tools, then on the Windows Firewall with Advanced Security shortcut.
  - Click on Outbound Rules in the left pane
  - Click on New Rule in the right pane
  - The New Outbound Rule Wizard will display
  - Click on Port, and then Next
  - Click on TCP, specific remote ports, and enter 445. Click Next
  - Click on Block the Connection, and then on Next
  - Ensure that Domain, Private, and Public are all checked, and then click Next
  - Enter a name for the rule, such as blockout445, an optional description, and then Finish
  - The new rule be added, and enabled. Disable the rule for now, but leave the Firewall screen up. It will be needed in the next few steps.
• Start the EMS instances win2012a and win2012b. The active EMS instance must be running on the same server where the firewall rule was created.
• From tibems3, start the java application
  o Cd to $TIBCO_HOME/ems/8.4/samples/java
  o ./setup.sh
  o java tibjmsMsgProducerPerf --server tcp://tibems1:7222, tcp://tibems2:7222 --factory FTConnectionFactory --delivery PERSISTENT --connections 10 --threads 8 --count 20000 --size 1000 --queue sync
• From win2012a, enable the blockout445 firewall rule
• The active EMS instance on win2012a should terminate with non-retryable disk write error: 59. It may take several minutes for this to occur:
5.2.3.2 Expected Results

The java test application should complete, pausing during failover, but should resume sending messages once the failover is complete. No messages should be lost. There can be more than 20K messages, depending on the number of connections/threads, but never should there be less than 20K messages. Even with a large number of messages that must be recovered, the recovery should only take a few seconds. It has been observed with the Windows Server and the Azure file system that the time-out of the port connection (445) can be several minutes, causing the fail-over to take several minutes. However, once the stand-by EMS instance becomes active, the recovery of the data is very short.

5.2.4 System Failure Test

This test verifies that an EMS client continues to function correctly, with no message loss during a system failure on the virtual machine running the active EMS server instance. This is not a normal occurrence. However, it is possible to accidentally restart the virtual machine from the Azure dashboard.

Two EMS server instances will be running in a F/T configuration, while messages are being sent. From the Azure Dashboard, the virtual machine where the active EMS instance is running will be restarted. The stand-by EMS instance should be able to gain the locks on the EMS data stores, and take over. The EMS java application should continue processing messages until it completes.
Note: In the following examples, <server1> is tibems1 and the <port1> is 7222, and <server2> is tibems2 and the <port2> is 7222. Tibems3 is the VM running the java apps. These are Linux VMs. With Windows Servers, the results should be identical. Substitute with the appropriate values for the environment.

5.2.4.1 Running the System Failure Test

- Three ssh terminal/PowerShell sessions are needed for this test; one one each of the virtual machine.
- The Azure Dashboard must also be available, and be on the virtual machine page that will be restarted.

![TIBCO Azure Virtual Machine Page](image)

Figure 27 - tibems1 VM page in the Azure Dashboard

- From tibems3, start the java application
  - Cd to $TIBCO_HOME/ems/8.4/samples/java
  - ./setup.sh

- In the Azure portal, go to the virtual machine page where the active EMS is running, and click on the restart button. This will restart the tibems1 VM.
- The ssh terminal/PowerShell window to that virtual machine should immediately terminate, and the stand-by EMS instance should recover all messages, and become active within a few seconds, depending on the number of messages to be recovered.
Figure 28 - Standby EMS instance recovering from system failure of primary

- After the java application completes, run `tibemsadmin64 tcp://tibems2:7222` (or to the active EMS instance), to verify that there is a minimum of 20000 messages in the sync queue.
- Restart the EMS instance on the restarted virtual machine, and stop the currently active EMS instance on the second virtual machine. EMS should become active on the restarted VM, and recover all 20K messages with no errors.
- Use `tibemsadmin64` to verify, then purge the sync queue.
- Stop EMS on both virtual machines.
- This concludes the tests, so all processes, terminal/PowerShell windows, and virtual machines can be stopped.

5.2.4.2 Expected Results

The java test application should complete, pausing during failover, but should resume sending messages once the failover is complete. No messages should be lost. There can be more than 20K messages, depending on the number of connections/threads, but there should never be less than 20K messages. Depending on the number of messages that must be recovered, the fail-over can take from a few to several seconds. It has been observed with the Azure restart of the virtual machine, that the EMS recovery is virtually no longer than a EMS process failure.