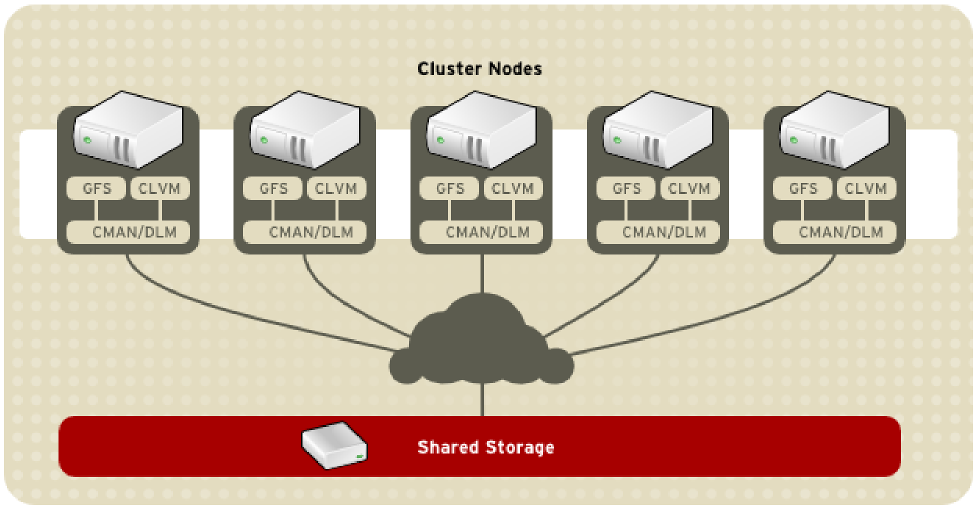
FIXEdge Failover Cluster installation

*** The following instructions are for CentOS/RedHat 7. They don't work for CentOS/RedHat 6.**

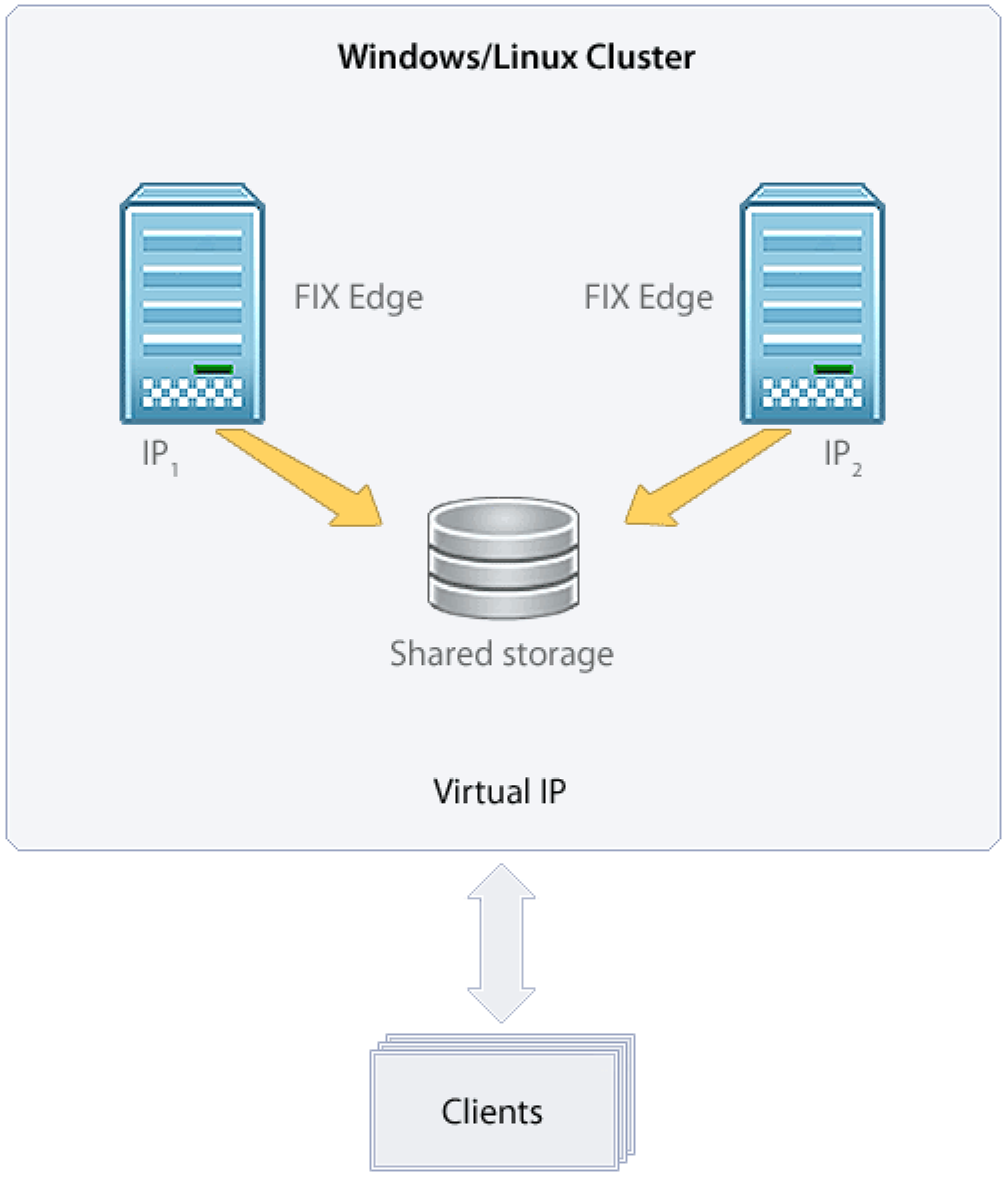
Deployment scheme

In general, the cluster might look like this (https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/5/html/Cluster_Suite_Overview/s1-hasci-overview-CSO.html):

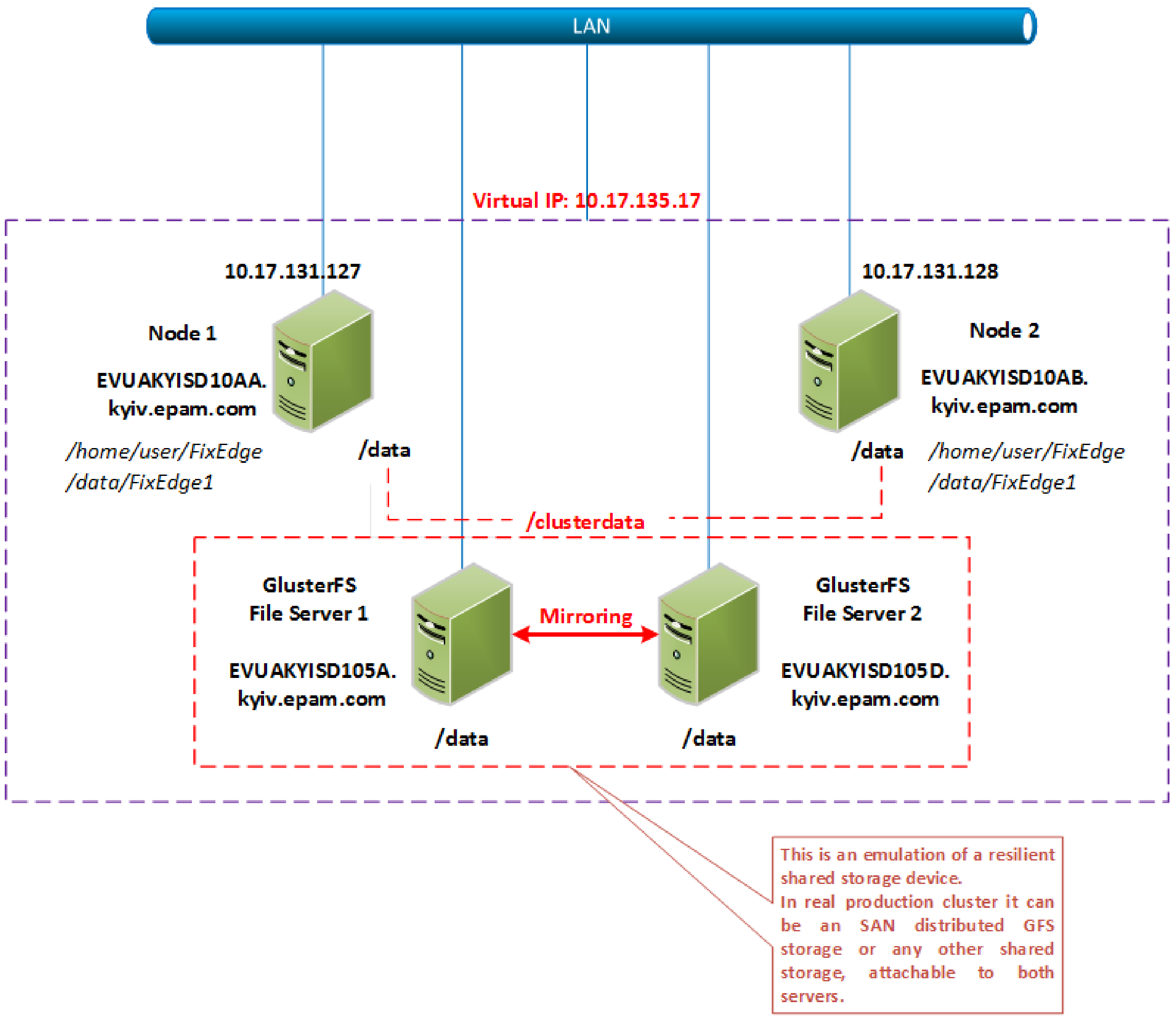


Shared storage might be a SAN-attached device, Fibre channel attached device, or TCP/IP attached device. The device might be attached to all nodes simultaneously, or cluster resource manager can attach it to the active node only. The device might be in turn a resilient one, presenting the distributed file system with software or hardware replication between filesystem nodes. In the case of a geographically distributed cluster, the shared storage also can be distributed geographically in the same way as cluster nodes, and cluster resource manager can attach to the node the storage instance in the same geo location.

Typical FIXEdge cluster setup is the two-node failover cluster with shared storage:



In this setup, the TCP/IP attached device is used. A two nodes GlusterFS file server with mirroring is used as the shared device. The device is mounted to both nodes simultaneously. See the diagram below:



Setting up GlusterFS Server

Reference articles:

http://blog.bobbyallen.me/2013/01/26/creating-a-highly-available-file-server-cluster-for-a-web-farm-using-ubuntu-12-04-lts/

http://www.sohailriaz.com/glusterfs-howto-on-centos-6-x/

Install software

Do these steps on both servers: *EVUAKYISD105A.kyiv.epam.com* and *EVUAKYISD105D.kyiv.epam.com*

Download and install

```
$ sudo wget -P /etc/yum.repos.d http://download.gluster.org/pub/gluster/glusterfs/LATEST/CentOS/glusterfs-epel.
repo
$ sudo yum install glusterfs
$ sudo yum install glusterfs-fuse
$ sudo yum install glusterfs-server
```

Check installed version

```
$ glusterfsd --version

glusterfs 3.6.2 built on Jan 22 2015 12:58:10
Repository revision: git://git.gluster.com/glusterfs.git
Copyright (c) 2006-2013 Red Hat, Inc. <http://www.redhat.com/>
GlusterFS comes with ABSOLUTELY NO WARRANTY.
It is licensed to you under your choice of the GNU Lesser
General Public License, version 3 or any later version (LGPLv3
or later), or the GNU General Public License, version 2 (GPLv2),
in all cases as published by the Free Software Foundation.
```

Start glusterfs services on all servers with enable them to start automatically on startup

```
$ sudo /etc/init.d/glusterd start
$ sudo chkconfig glusterfsd on
```

Configure

Do these steps only on one server. For example, on *EVUAKYISD105A.kyiv.epam.com*

add EVUAKYISD105D to the trusted storage pool

```
$ sudo gluster peer probe EVUAKYISD105D.kyiv.epam.com
```

Now we can check the status

```
$ sudo gluster peer status
Number of Peers: 1
Hostname: EVUAKYISD105D.kyiv.epam.com
Uuid: 04126ea4-a0e5-4854-a38e-69c24d9c05aa
State: Peer in Cluster (Connected)
```

On both nodes create a folder to be shared

```
$ cd /
$ sudo mkdir data
$ sudo chmod a+rwx /data
```

Create the storage share and start it

```
$ sudo gluster volume create clusterdata replica 2 transport tcp EVUAKYISD105A.kyiv.epam.com:/data
EVUAKYISD105D.kyiv.epam.com:/data force

volume create: clusterdata: success: please start the volume to access data

$ sudo gluster volume start clusterdata

volume start: clusterdata: success

$ sudo gluster volume info


Volume Name: clusterdata
Type: Replicate
Volume ID: fa31c2de-f7d6-4f05-97e5-e310d2680d94
Status: Started
Number of Bricks: 1 x 2 = 2
Transport-type: tcp
Bricks:
Brick1: EVUAKYISD105A.kyiv.epam.com:/data
Brick2: EVUAKYISD105D.kyiv.epam.com:/data
```

Mounting GlusterFS storage on FIXEdge cluster nodes

These commands shall be run on both FIXEdge nodes: *EVUAKYISD103D.kyiv.epam.com* and *EVUAKYISD103C.kyiv.epam.com*

Install client components

```
$ sudo wget -P /etc/yum.repos.d http://download.gluster.org/pub/gluster/glusterfs/LATEST/CentOS/glusterfs-epel.
repo
$ sudo yum install glusterfs-client
```

Mount remote storage

```
$ sudo mkdir /data
$ sudo chmod a+rwx /data
$ sudo mount.glusterfs EVUAKYISD105D.kyiv.epam.com:/clusterdata /data

$ mount
/dev/mapper/VolGroup00-LogVol00 on / type ext4 (rw,noatime,nodiratime,noacl,commit=60,errors=remount-ro)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
tmpfs on /dev/shm type tmpfs (rw)
/dev/sda1 on /boot type ext4 (rw)
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
EVUAKYISD105D.kyiv.epam.com:/clusterdata on /data type fuse.glusterfs (rw,default_permissions,allow_other,
max_read=131072)
```

And now add the following line to /etc/fstab to ensure mounting after reboot:

```
EVUAKYISD105A.kyiv.epam.com:/clusterdata        /data   glusterfs       defaults,_netdev        0 0
```

Setting up FIXEdge instances

Copy FixEdge-5.8.2.68334-Linux-2.6.32-gcc447-x86_64.tar.gz to /home/user to both nodes: *EVUAKYISD103D.kyiv.epam.com* and *EVUAKYISD103C.kyiv.epam.com*.

On the Node1

Unzip. Untar.

```
$ gunzip FixEdge-5.8.2.68334-Linux-2.6.32-gcc447-x86_64.tar.gz
$ tar xf FixEdge-5.8.2.68334-Linux-2.6.32-gcc447-x86_64.tar
```

Move installation folder to /home/user/FixEdge:

```
$ cd FixEdge
$ cd v.5.8.2.68334
$ mv * /home/user/FIXEdge
$ cd ..
$ rmdir v.5.8.2.68334
```

Upload licenses (engine.license and fixaj2-license.bin) to /home/user/FIXEdge folder.

Move FIXEdge instance folder to shared storage:

```
mv FixEdge1 /data/
```

Edit scripts in /home/user/FixEdge/bin to direct them to the new FIXEdge instance location:

- replace ".." with "/data".

Edit FIXICC Agent configuration:

- /home/user/FixEdge/fixicc-agent/conf/wrapper.conf
    - wrapper.daemon.pid.dir = ${wrapper_home}
- /home/user/FixEdge/fixicc-agent/conf/agent.properties
    - EngineProperty = /data/FixEdge1/conf/engine.properties
    - FIXEdgeFileSettings = /data/FixEdge1/conf/FIXEdge.properties
    - LogUrl = /data/FixEdge1/log

Edit FIXEdge and engine configuration:

- /data/FixEdge1/conf/engine.properties
    - EngineRoot = /data
    - LicenseFile = /home/user/FixEdge/engine.license
- /data/FixEdge1/conf/FIXEdge.properties
    - FIXEdge.RootDir = /data/FixEdge1
    - Log.File.RootDir = /data/FixEdge1
    - TransportLayer.SmtpTA.DllName = /home/user/FixEdge/bin/libSMTPTA.so

Install and Start FIXICC Agent daemon:

```
$ cd /home/user/FixEdge/fixicc-agent/bin
$ ./installDaemon.sh
$ ./startDaemon.sh
```

Now everything is ready to run FIXEdge on Node 1.

Prepare to copy the installation to Node 2:

```
$ cd /home/user
$ tar cvf FixEdge.tar FixEdge
$ gzip FixEdge.tar
```

Copy file FixEdge.tar.gz to Node2:/user/home

On the Node2

Unzip, untar

```
$ cd /home/user
$ gunzip FixEdge.tar.gz
$ tar xf FixEdge.tar
```

Install and Start FIXICC Agent daemon:

```
$ cd /home/user/FixEdge/fixicc-agent/bin
$ ./installDaemon.sh
$ ./startDaemon.sh
```

Open ports on both nodes

8005 for FIXICC Agent and 8901 for FIX Engine

```
$ sudo iptables -I INPUT -p tcp -m state --state NEW -m tcp --dport 8005 -j ACCEPT
$ sudo iptables -I INPUT -p tcp -m state --state NEW -m tcp --dport 8901 -j ACCEPT
$ sudo service iptables save
```

Configuring a failover cluster for FIXEdge with Pacemaker, Corosync & PCS

Reference articles:

http://jensd.be/?p=156

https://github.com/feist/pcs/issues/

On both nodes install needed software

```
$ sudo yum install corosync pcs pacemaker
```

On both nodes set the password for hacluster user ('epmc-cmcc' was used)

```
$ sudo passwd hacluster
```

Configure Firewall on both nodes to allow cluster traffic:

```
$ sudo iptables -I INPUT -m state --state NEW -p udp -m multiport --dports 5404,5405 -j ACCEPT
$ sudo iptables -I INPUT -p tcp -m state --state NEW -m tcp --dport 2224 -j ACCEPT
$ sudo iptables -I INPUT -p igmp -j ACCEPT
$ sudo iptables -I INPUT -m addrtype --dst-type MULTICAST -j ACCEPT
$ sudo service iptables save
```

Start the pcsd service on both nodes

```
$ sudo systemctl start pcsd
```

From now on all commands needs to be executed on one node only. We can control the cluster by using PCS from on of the nodes.

Since we will configure all nodes from one point, we need to authenticate on all nodes before we are allowed to change the configuration. Use the previously configured hacluster user and password to do this:

```
$ sudo pcs cluster auth EVUAKYISD10AA.kyiv.epam.com EVUAKYISD10AB.kyiv.epam.com
Username: hacluster
Password:
EVUAKYISD10AA.kyiv.epam.com: Authorized
EVUAKYISD10AB.kyiv.epam.com: Authorized
```

Create the cluster and add nodes. This command command creates the cluster node configuration in /etc/corosync.conf.

```
$ sudo pcs cluster setup --name fixedge_cluster EVUAKYISD10AA.kyiv.epam.com EVUAKYISD10AB.kyiv.epam.com
Shutting down pacemaker/corosync services...
Redirecting to /bin/systemctl stop  pacemaker.service
Redirecting to /bin/systemctl stop  corosync.service
Killing any remaining services...
Removing all cluster configuration files...
EVUAKYISD10AA.kyiv.epam.com: Succeeded
EVUAKYISD10AB.kyiv.epam.com: Succeeded
```

We can start cluster now:

```
$ sudo pcs cluster start --all
EVUAKYISD10AB.kyiv.epam.com: Starting Cluster...
EVUAKYISD10AA.kyiv.epam.com: Starting Cluster...
```

We can check cluster status:

```
$ sudo pcs status cluster
Cluster Status:
 Last updated: Tue Jan 27 22:11:15 2015
 Last change: Tue Jan 27 22:10:48 2015 via crmd on EVUAKYISD10AA.kyiv.epam.com
 Stack: corosync
 Current DC: EVUAKYISD10AA.kyiv.epam.com (1) - partition with quorum
 Version: 1.1.10-32.el7_0.1-368c726
 2 Nodes configured
 0 Resources configured


$ sudo pcs status nodes
Pacemaker Nodes:
 Online: EVUAKYISD10AA.kyiv.epam.com EVUAKYISD10AB.kyiv.epam.com
 Standby:
 Offline:

$ sudo corosync-cmapctl | grep members
runtime.totem.pg.mrp.srp.members.1.config_version (u64) = 0
runtime.totem.pg.mrp.srp.members.1.ip (str) = r(0) ip(10.17.131.127)
runtime.totem.pg.mrp.srp.members.1.join_count (u32) = 1
runtime.totem.pg.mrp.srp.members.1.status (str) = joined
runtime.totem.pg.mrp.srp.members.2.config_version (u64) = 0
runtime.totem.pg.mrp.srp.members.2.ip (str) = r(0) ip(10.17.131.128)
runtime.totem.pg.mrp.srp.members.2.join_count (u32) = 1
runtime.totem.pg.mrp.srp.members.2.status (str) = joined


$ sudo pcs status corosync
Membership information
----------------------
    Nodeid      Votes Name
         1          1 EVUAKYISD10AA.kyiv.epam.com (local)
         2          1 EVUAKYISD10AB.kyiv.epam.com
```

Disable the STONITH option as we don't have STONITH devices in our demo virtual environment:

```
$ sudo pcs property set stonith-enabled=false
```

For two-nodes cluster we must disable the quorum:

```
$ sudo pcs property set no-quorum-policy=ignore
$ sudo pcs property
Cluster Properties:
 cluster-infrastructure: corosync
 dc-version: 1.1.10-32.el7_0.1-368c726
 no-quorum-policy: ignore
 stonith-enabled: false
```

Add Virtual IP as a resource to the cluster:

```
$ sudo pcs resource create virtual_ip ocf:heartbeat:IPaddr2 ip=10.17.135.17 cidr_netmask=32 op monitor
interval=30s
$ sudo pcs status resources
 virtual_ip (ocf::heartbeat:IPaddr2): Started
```

Add FIXEdge as a resource to cluster:

```
$ sudo pcs resource create FIXEdge ocf:heartbeat:anything params binfile="/home/user/FixEdge/bin/FIXEdge"
cmdline_options="/data/FixEdge1/conf/FIXEdge.properties" user="user" logfile="/home/user/FIXEdge_resource.log"
errlogfile="/home/user/FIXEdge_resource_error.log"
```

For some reason in the /usr/lib/ocf/resource.d/ of the installed cluster there are many missing agents, including ocf:heartbeat:anything. You need to modify the original version (which you can download here: https://github.com/ClusterLabs/resource-agents/blob/master/heartbeat/anything) to make it working. The working version of the agent is attached.

This file should be copied to /usr/lib/ocf/resource.d/ and make executable:

```
$ sudo cp anything /usr/lib/ocf/resource.d/heartbeat/
$ sudo chmod a+rwx /usr/lib/ocf/resource.d/heartbeat/anything
```

Also, to make this agent works the following lines shall be added to sudoers file:

```
$ sudo visudo
Defaults    !requiretty
user    ALL=(user)      NOPASSWD: ALL
root    ALL=(user)      NOPASSWD: ALL
```

In order to make sure that the Virtual IP and FIXEdge always stay together, we can add a constraint:

```
$ sudo pcs constraint colocation add FIXEdge virtual_ip INFINITY
```

To avoid the situation where the FIXEdge would start before the virtual IP is started or owned by a certain node, we need to add another constraint which determines the order of availability of both resources:

```
$ sudo pcs constraint order virtual_ip then FIXEdge
Adding virtual_ip FIXEdge (kind: Mandatory) (Options: first-action=start then-action=start)
```

After configuring the cluster with the correct constraints, restart it and check the status:

```
$ sudo pcs cluster stop --all && sudo pcs cluster start --all
EVUAKYISD10AB.kyiv.epam.com: Stopping Cluster...
EVUAKYISD10AA.kyiv.epam.com: Stopping Cluster...
EVUAKYISD10AA.kyiv.epam.com: Starting Cluster...
EVUAKYISD10AB.kyiv.epam.com: Starting Cluster...
```

Cluster configuration is now completed.

FIXEdge Recovery Time Objective and Recovery Point Objective

The current article describes the Recovery Time Objective (RTO) Recovery Point Objective(RPO) for Disaster recovery in case of active-passive Cluster configuration (see FIXEdge Failover Cluster installation).

Recovery Time Objective (RTO)

- Some functions are more important than others and may need a quicker recovery.
- The period of time within which systems, applications, or functions must be recovered after an outage is referred to as Recovery Time Objectives (RTOs).
- All business functions, applications or vendors that require a business continuity plan must have an RTO.

Recovery Point Objective(RPO)

- An RPO is the maximum allowable amount of data loss an organization can tolerate as a result of an event which causes systems recovery (the transfer of work) to an alternative system environment or location.
- An RPO is applicable only to applications that store data and are evaluated using similar to RTO requirements.
- The RPO is not the time it takes to restore data as part of recovery. The data restoration time is part of the RTO.

Possible RTO Values are described as "Greater than **X** time, up to and including **Y** time"

Possible RPO Values are described as "Recent sync point: Greater than **X** time, up to and including **Y** time"

Failover scenario #1. Application or cluster node failure

Failure of software or hardware within a node (e.g. application failure, node unavailability, resource exhausting). Failover Cluster consists of several application nodes. In case of one node fails – the application (or DB) is moved to another node and start there within approximately 2 minutes.

The session recovery procedure happens automatically. The missing messages should be recovered with a resend request procedure automatically.

Session recovery requires reset sequence in case of damaged storage. The messages from the beginning of the day will be lost when resetting the sequence. See Recovery procedure for a session with corrupted storages

It is possible to manually request missing messages since sequence number equal one using the Logon (A) with the desired sequence and the subsequent Request Request (2) messages (see the FIX specification https://www.fixtrading.org/standards/)

Achievable RTO Values: **"Greater than 0 seconds, up to and including 2 minutes"**

Achievable RPO Values: **"Recent synch point: Greater than 0 seconds, up to and including few seconds"**

Failover scenario #2. Whole cluster failure

Disaster recovery environment should expect the connection. In case of disaster, the production environment is moved to DR.

Achievable RTO Values: **"Greater than 0 seconds, up to and including few seconds"**

Achievable RPO Values: **"Recent synch point: Greater than 0 seconds, up to and including 1 day, i.e. since the start of the day".**

The better RPO values can be achieved with additional configuration or tools like log replication tool

This procedure requires additional configuration for each client individually.

Attachments

Unknown macro: 'attachment-table'