목차

How to configure the Fencing on Pacemaker? 3
환경(시연목적) 4
fence_xvm을 사용하도록 KVM 호스트 구성 4
클러스터 노드에서 펜싱 구성 7
STONITH의 유효성 확인 8
출처 8

How to configure the Fencing on Pacemaker?

원본출처 : https://www.unixarena.com/2016/01/rhel-7-configure-fencing-pacemaker.html/

펜싱(STONITH)은 공유 스토리지의 데이터 손상을 피하기 위해 클러스터에서 중요한 메커니즘입니다. 또한 노드 사이에 분할 된 두뇌가있을 때 클러스터를 알려진 상태로 가져 오는 데 도움이됩니다. 클러스터 노드는 일반적으로 이더넷과 같은 표준 네트워크 연결 인 통신 채널을 통해 서로 통신합니다. 각 자원과 노드는 클러스터에서 " 상태 " (예 : 시작됨, 중지됨)를 가지며 노드는 자원에서 발생하는 모든 변경 사항을보고합니다. 이보고는 노드 간 통신이 중단 될 때까지 잘 작동합니다. 펜싱은 노드가 서로 통신 할 수 없을 때 재생됩니다. 대부분의 노드는 쿼럼 투표를 기반으로 클러스터를 형성하고 나머지 노드는 재부팅되거나 차단 된 조치에 따라 중단됩니다.

pacemaker에는 두 종류의 펜싱이 있습니다.

- 리소스 수준 펜싱(Resource Level Fencing)
- 노드 레벨 펜싱(Node Level Fencing)

클러스터는 Resource level fencing을 사용하여 노드가 두 노드의 동일한 자원에 액세스 할 수 없음을 확인할 수 있습니다. Node level fencing은 노드가 자원을 전혀 실행하지 않도록 합니다. 이는 대개 간단하지만 잔인한 방법으로 수행됩니다. 노드는 전원 스위치를 사용하여 간단히 재설정됩니다. 노드가 전혀 응답하지 않을 수 있으므로 궁극적으로 이것이 필요할 수 있습니다. Pacamaker / corosync 클러스터에서 펜싱 방법을 "STONITH"(Shoot The Other Node In The Head; 상대노드헤드샷)라고 부릅니다.

자세한 내용은 clusterlabs.org 를 방문하십시오. 여기에 노드 레벨 펜싱이 보일 것입니다.

클러스터 설정을 살펴봅니다.

```
[root@Node1-LAB ~]# pcs status
Cluster name: GFSCLUS
Last updated: Wed Jan 20 12:43:36 2016
Last change: Wed Jan 20 09:57:06 2016 via cibadmin on Node1
Stack: corosync
Current DC: Node1 (1) - partition with quorum
Version: 1.1.10-29.el7-368c726
2 Nodes configured
2 Resources configured
Online: [ Node1 Node2 ]
PCSD Status:
Node1: Online
Node2: Online
Daemon Status:
corosync: active/enabled
pacemaker: active/enabled
pcsd: active/enabled
[root@Node1-LAB ~]#
```

이 내용에서는 KVM 클러스터 노드에 fence_xvm을 이용하여 stonith/fencing 구성하는 방법을 설명합니다. 이 설정의 목적은 STONITH / FENCING을 시연하는 것입니다.

환경(시연목적)

- node1 및 node2 - pacemaker / corosync 클러스터
- UNIXKB-CP - Node1과 Node2를 호스팅하는 KVM 호스트

fence_xvm을 사용하도록 KVM 호스트 구성

1. KVM 호스트에 로그인합니다. 2. 실행중인 가상 시스템을 나열하십시오.

```
[root@UNIXKB-CP ~]# virsh list
Id    Name                           State
----------------------------------------------------
6     Node1                          running
7     Node2                          running
[root@UNIXKB-CP ~]#
```

3. KVM 호스트 (비 클러스터 노드)에서 필요한 펜싱 패키지를 설치합니다.

```
[root@UNIXKB-CP ~]# yum install fence-virt fence-virtd fence-virtd-libvirt
fence-virtd-multicast fence-virtd-serial
Loaded plugins: langpacks, product-id, subscription-manager
This system is not registered to Red Hat Subscription Management. You can
use subscription-manager to register.
Package fence-virt-0.3.0-16.el7.x86_64 already installed and latest version
Package fence-virtd-0.3.0-16.el7.x86_64 already installed and latest version
Package fence-virtd-libvirt-0.3.0-16.el7.x86_64 already installed and latest
version
Package fence-virtd-multicast-0.3.0-16.el7.x86_64 already installed and
latest version
Package fence-virtd-serial-0.3.0-16.el7.x86_64 already installed and latest
version
Nothing to do
[root@UNIXKB-CP ~]#
```

4. 펜스 키를 저장할 새 디렉토리를 만듭니다. 펜싱에 사용할 랜덤 키를 만듭니다.

```
[root@UNIXKB-CP ~]# mkdir -p /etc/cluster
[root@UNIXKB-CP ~]# cd /etc/cluster/
[root@UNIXKB-CP cluster]# dd if=/dev/urandom of=/etc/cluster/fence_xvm.key
bs=4k count=1
1+0 records in
1+0 records out
4096 bytes (4.1 kB) copied, 0.000506736 s, 8.1 MB/s
[root@UNIXKB-CP cluster]#
```

5. 펜스 키를 클러스터 노드에 복사합니다. (노드 1 및 노드 2)

```
[root@UNIXKB-CP cluster]# scp -r /etc/cluster/fence_xvm.key
root@Node1:/etc/cluster/fence_xvm.key
root@node1's password:
fence_xvm.key
100% 4096     4.0KB/s   00:00
[root@UNIXKB-CP cluster]# scp -r /etc/cluster/fence_xvm.key
root@Node2:/etc/cluster/fence_xvm.key
root@node2's password:
fence_xvm.key
100% 4096     4.0KB/s   00:00
[root@UNIXKB-CP cluster]#
```

참고 : xvm 키를 복사하려면 클러스터 노드에 /etc/cluster
디렉토리를 만들어야합니다.

6. fence_virtd -c명령을 사용하여 /etc/fence_virt.conf파일을 만듭니다.

```
[root@UNIXKB-CP ~]# fence_virtd -c
Module search path [/usr/lib64/fence-virt]:
Available backends:
libvirt 0.1
Available listeners:
multicast 1.2
Listener modules are responsible for accepting requests
from fencing clients.
Listener module [multicast]:
The multicast listener module is designed for use environments
where the guests and hosts may communicate over a network using
multicast.
The multicast address is the address that a client will use to
send fencing requests to fence_virtd.
Multicast IP Address [225.0.0.12]:
Using ipv4 as family.
Multicast IP Port [1229]:
Setting a preferred interface causes fence_virtd to listen only
on that interface.  Normally, it listens on all interfaces.
In environments where the virtual machines are using the host
machine as a gateway, this *must* be set (typically to virbr0).
Set to 'none' for no interface.
Interface [virbr0]: br0:1
The key file is the shared key information which is used to
authenticate fencing requests.  The contents of this file must
be distributed to each physical host and virtual machine within
a cluster.
```

```
Key File [/etc/cluster/fence_xvm.key]:
Backend modules are responsible for routing requests to
the appropriate hypervisor or management layer.
Backend module [libvirt]:
Configuration complete.
=== Begin Configuration ===
backends {
libvirt {
uri = "qemu:///system";
}
}
listeners {
multicast {
port = "1229";
family = "ipv4";
interface = "br0:1";
address = "225.0.0.12";
key_file = "/etc/cluster/fence_xvm.key";
}
}
fence_virtd {
module_path = "/usr/lib64/fence-virt";
backend = "libvirt";
listener = "multicast";
}
=== End Configuration ===
Replace /etc/fence_virt.conf with the above [y/N]? y
[root@UNIXKB-CP ~]#
```

올바른 인터페이스가 브리지인지 확인하십시오. My setup에서 `br0:1` 가상 인터페이스를 사용하여 KVM 게스트와 통신합니다.

7. `fence_virtd` 서비스를 시작하십시오.

```
[root@UNIXKB-CP ~]# systemctl enable fence_virtd.service
[root@UNIXKB-CP ~]# systemctl start fence_virtd.service
[root@UNIXKB-CP ~]# systemctl status fence_virtd.service
fence_virtd.service - Fence-Virt system host daemon
Loaded: loaded (/usr/lib/systemd/system/fence_virtd.service; enabled)
Active: active (running) since Wed 2016-01-20 23:36:14 IST; 1s ago
Process: 3530 ExecStart=/usr/sbin/fence_virtd $FENCE_VIRTD_ARGS
(code=exited, status=0/SUCCESS)
Main PID: 3531 (fence_virtd)
CGroup: /system.slice/fence_virtd.service
└─3531 /usr/sbin/fence_virtd -w
Jan 20 23:36:14 UNIXKB-CP systemd[1]: Starting Fence-Virt system host
daemon...
Jan 20 23:36:14 UNIXKB-CP systemd[1]: Started Fence-Virt system host daemon.
Jan 20 23:36:14 UNIXKB-CP fence_virtd[3531]: fence_virtd starting.
```

```
Listener: libvirt  Backend: multicast
[root@UNIXKB-CP ~]#
```

클러스터 노드에서 펜싱 구성

1. 클러스터 노드 중 하나에 로그인하십시오. 2. 두 노드 모두 fence_virt 패키지를 가지고 있는지 확인하십시오.

```
[root@Node1-LAB ~]# rpm -qa fence-virt
fence-virt-0.3.0-16.el7.x86_64
[root@Node1-LAB ~]#
```

3. 다음 명령은 클러스터에서 펜싱을 구성하기 위해 많이 사용됩니다.

```
[root@Node1-LAB ~]# fence_xvm -o list
Node1                6daac670-c494-4e02-8d90-96cf900f2be9 on
Node2                17707dcb-7bcc-4b36-9498-a5963d86dc2f on
[root@Node1-LAB ~]#
```

4. 클러스터 노드 항목이 /etc/hosts에 있어야합니다.

```
[root@Node1-LAB ~]# cat /etc/hosts |grep Node
192.168.2.10    Node1-LAB  Node1
192.168.2.11    Node2-LAB  Node2
[root@Node1-LAB ~]#
```

5. pacemaker cluster에 fence_xvm fence agent를 구성하십시오.

```
[root@Node1-LAB ~]# pcs stonith create xvmfence  fence_xvm
key_file=/etc/cluster/fence_xvm.key
[root@Node1-LAB ~]#
[root@Node1-LAB ~]# pcs stonith
xvmfence       (stonith:fence_xvm):    Started
[root@Node1-LAB ~]#
[root@Node1-LAB ~]# pcs stonith --full
Resource: xvmfence (class=stonith type=fence_xvm)
Attributes: key_file=/etc/cluster/fence_xvm.key
Operations: monitor interval=60s (xvmfence-monitor-interval-60s)
[root@Node1-LAB ~]#
```

RHEL7 - Pacemaker/Corosync 클러스터에서 펜싱을 성공적으로 구성했습니다. (클러스터는 두 명의 KVM 게스트 사이에서 구성되었습니다).

STONITH의 유효성 확인

stonith구성을 어떻게 테스트해야합니까? 여기에 작은 데모가 있습니다.

1. 클러스터 노드 중 하나에 로그인하십시오. 2. 노드를 펜싱 시켜보십시오.

```
[root@Node1-LAB ~]# pcs stonith fence Node2
Node: Node2 fenced
[root@Node1-LAB ~]#
```

결국 Node2가 재부팅됩니다. 클러스터 속성에 따라 재부트가 발생합니다.

```
[root@Node1-LAB ~]# pcs property --all |grep stonith-action
stonith-action: reboot
[root@Node1-LAB ~]#
```

Stonith는 pcs 속성 명령을 사용하여 ON / OFF 될 수 있습니다.

```
[root@Node1-LAB ~]# pcs property --all |grep stonith-enabled
stonith-enabled: true
[root@Node1-LAB ~]#
```

이 내용이 당신에게 많은 도움이 되었길 바랍니다.

출처

- https://www.unixarena.com/2016/01/rhel-7-configure-fencing-pacemaker.html/

From:
https://atl.kr/dokuwiki/ - **AllThatLinux!**

Permanent link:
https://atl.kr/dokuwiki/doku.php/how_to_configure_the_fencing_on_pacemaker

Last update: **2018/09/11 05:21**