GiantVM 설치 가이드

- 다음의 내용은 2대의 머신(*node0*, *node1*)을 GiantVM으로 묶어서 가상 1머신으로 구동하기 위한 절차입니다.

- 권장 환경은 2대의 real 머신이 infiniband 네트워크로 연결된 구성입니다.

  (아직, VirtualBox 등의 가상 환경에서는 설치 절차가 검증되지 않은 상황입니다.
  변경 사항 있을 때마다 다시 업로드하겠습니다.)

1. 준비물

1) H/W

- 인피니밴드(infiniband) 네트워크로 연결된 2대의 real 머신

  (참고: original GiantVM은 infiniband 네트워크에 맞춰서 개발되었으며,
  이더넷 환경은 정상 지원하지 않음)

2) S/W

- GiantVM 패치 소스 2종 (참조: https://github.com/ememos/GiantVM)
  : **Linux-DSM-4.18** (GiantVM KVM patch for kernel 4.18)
  : **QEMU-gvm-vcpupin** (GiantVM QEMU vcpu pin patch)

- 환경 설정 파일
  : config-4.18.20-gvm (Linux-DSM-4.18 컴파일용)

- Guest OS
  : GiantVM 상에서 기동할 guest OS 이미지

- Guest OS 기동용 스크립트

2. GiantVM 패치 소스 2종 준비

```
$ git clone https://github.com/ememos/GiantVM.git
```

(결과로서, GiantVM 디렉토리 아래에
Linux-DSM-4.18, **QEMU-gvm-vcpupin** 디렉토리와
config-4.18.20-gvm 파일 등 생성됨)

3. 호스트용 커널 설치 (Linux-DSM-4.18)

3-1. 커널 컴파일 준비

```
$ sudo apt-get install -y build-essential libncurses5-dev gcc libssl-dev grub2 bc
$ sudo apt install flex
$ sudo apt install bison
$ sudo apt install net-tools
$ sudo apt install libelf-dev
```

3-2. 호스트용 커널 컴파일 (Linux-DSM-4.18)

```
$ cd GiantVM
$ cd Linux-DSM-4.18
$ vi Makefile   (확인 또는 명칭 수정)

   --> EXTRAVERSION = -gvm

$ cp ../config-4.18.20-gvm .config

$ sudo make oldconfig
$ sudo make deb-pkg -j 10 LOCALVERSION=1

$ cd ..
$ sudo dpkg -i linux-image-4.18.20-gvm1_4.18.20-gvm1-1_amd64.deb
$ sudo dpkg -i linux-libc-dev_4.18.20-gvm1-1_amd64.deb
$ sudo dpkg -i linux-headers-4.18.20-gvm1_4.18.20-gvm1-1_amd64.deb
$ sudo dpkg -i linux-image-4.18.20-gvm1-dbg_4.18.20-gvm1-1_amd64.deb
```

3-3. grub 파일(/etc/default/grub) 수정

3-4. 호스트 재부팅

4. 호스트용 QEMU 설치 (QEMU-gvm-vcpupin)

4-1. kvm.h 파일 수정

```
$ cd QEMU-gvm-vcpupin
$ vi linux-headers/linux/kvm.h

   (Line 873) #define KVM_CAP_X86_DSM        133 -> 156 (수정)
```

4-2. configure 실행

```
$ sudo ./configure —target-list=x86_64-softmmu —enable-kvm —disable-werror
```

4-3. make 실행
$ sudo make -j 10

여기까지 진행되면 GiantVM을 실행할 수 있는 준비가 된 상태가 됨

5. GuestOS 기동

- guest OS를 GiantVM에서 기동
  : 아래의 예제는 vcpu를 total 8개 구성하고, node0 머신에서는 0-3번 vcpu를 기동, node1 머신에서는 4-7번 vcpu를 기동하는 절차입니다.

5-1. guest OS 준비 (ubuntu1804.img 파일)
  # qemu-img create <image name> <image size>
    ==> # qemu-img create *ubuntu1804.img 10G*

  # qemu-system-x86_64 -cpu host -smp 8 -m 8192 -hda ubuntu1804.img
     -cdrom <ubuntu ISO filename>.iso -boot d -enable-kvm

    ... 이후 ubuntu 설치 진행 ...
   (설치 중에는 GUI 필요. 설치 후에는 -nographic 옵션으로 text 기반 사용 가능)

5-2. node0 머신에서 기동 스크립트 실행
  # ./run_gvm.sh 0

5-3. node1 머신에서 기동 스크립트 실행
  # ./run_gvm.sh 4

(run_gvm.sh 스크립트 내용)

```
#!/bin/bash
START_VCPU=$1
if [ "$START_VCPU" == "" ]; then
        echo "usage: $0 START_VCPU"
        exit
fi

VM_IMAGE=/home/GiantVM/ubuntu1804.img

sudo setfacl -m "u:etri:rw" /dev/kvm
COMMAND="./QEMU/x86_64-softmmu/qemu-system-x86_64 "
COMMAND+="--nographic --enable-kvm "
```

```
COMMAND+="-hda ${VM_IMAGE} "
COMMAND+="-cpu host,-kvm-asyncpf -machine kernel-irqchip=off "
COMMAND+="-smp 8 -m $((8*1024)) -serial mon:stdio "
COMMAND+="-monitor telnet:127.0.0.1:1234,server,nowait "

# for NUMA configuration
#COMMAND+="-object memory-backend-ram,size=4G,id=ram0 "
#COMMAND+="-numa node,nodeid=0,cpus=0-3,memdev=ram0 "
#COMMAND+="-object memory-backend-ram,size=4G,id=ram1 "
#COMMAND+="-numa node,nodeid=1,cpus=4-7,memdev=ram1 "

if [ "$START_VCPU" == "0" ]; then
        COMMAND+="-redir tcp:5556::22 "
fi

sudo $COMMAND \
        -local-cpu 4,start=$START_VCPU,iplist="10.10.20.14  10.10.20.16"
```