

SLURM 기초 교육

2024.11.12

다원컴퓨팅(주)
윤민수

DAWON
COMPUTING

CONTENTS

1. SLURM 이란?
2. SLURM 기본용어
3. SLURM Command
4. 이용내역 확인
5. 기타사항

1. SLURM 이란?

1. SLURM 이란?

1) SLURM(Simple Linux Utility for Resource Management)

각각의 독립적인 컴퓨팅 자원들을 Slurm을 이용하여 하나의 클러스터로 구성된 HPC(High Performance Computing)환경을 제공 합니다.

전 세계의 많은 슈퍼컴퓨터와 컴퓨터 클러스터에서 사용되는 Linux 및 Unix 계열 커널을 위한 무료 오픈 소스 작업 스케줄러 입니다.

글로벌 슈퍼컴퓨터 TOP500 의 약 60%의 작업 부하 관리자입니다.

오픈소스 기반의 배치 스케줄러로서 모든 규모의 클러스터를 위해 디자인 되었습니다.



1. SLURM 이란?

2) SLURM 주요기능

▣ 리소스 모니터링

- 클러스터 자원들의 상태를 모니터링
- 노드 상태, 리소스 가용성, 작업 상태 등

▣ 리소스 관리

- 클러스터의 리소스를 관리하여 작업이 효율적으로 실행
- CPU, GPU, 메모리, Disk 공간, 네트워크 등

▣ 작업 스케줄링

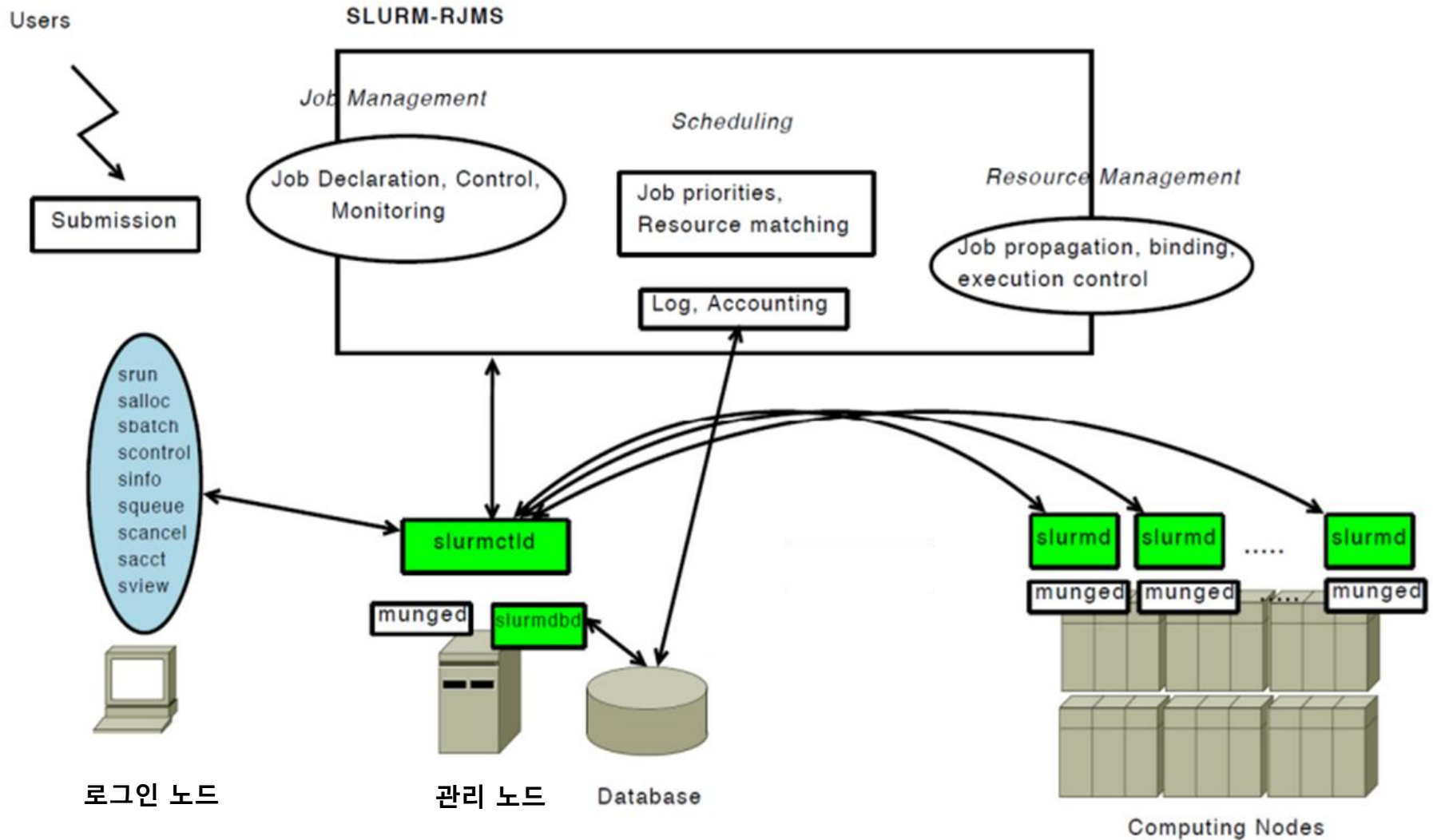
- 사용자들이 제출한 작업을 스케줄링하여 클러스터에 구성된 리소스 사용을 최적화
- 작업의 우선순위, 리소스 가용성 등 여러 요인을 고려하여 스케줄링

▣ 작업제출 및 관리

- 사용자가 작업을 제출하고 관리 할 수 있는 인터페이스를 제공
- 작업 제출, 중지, 취소 등

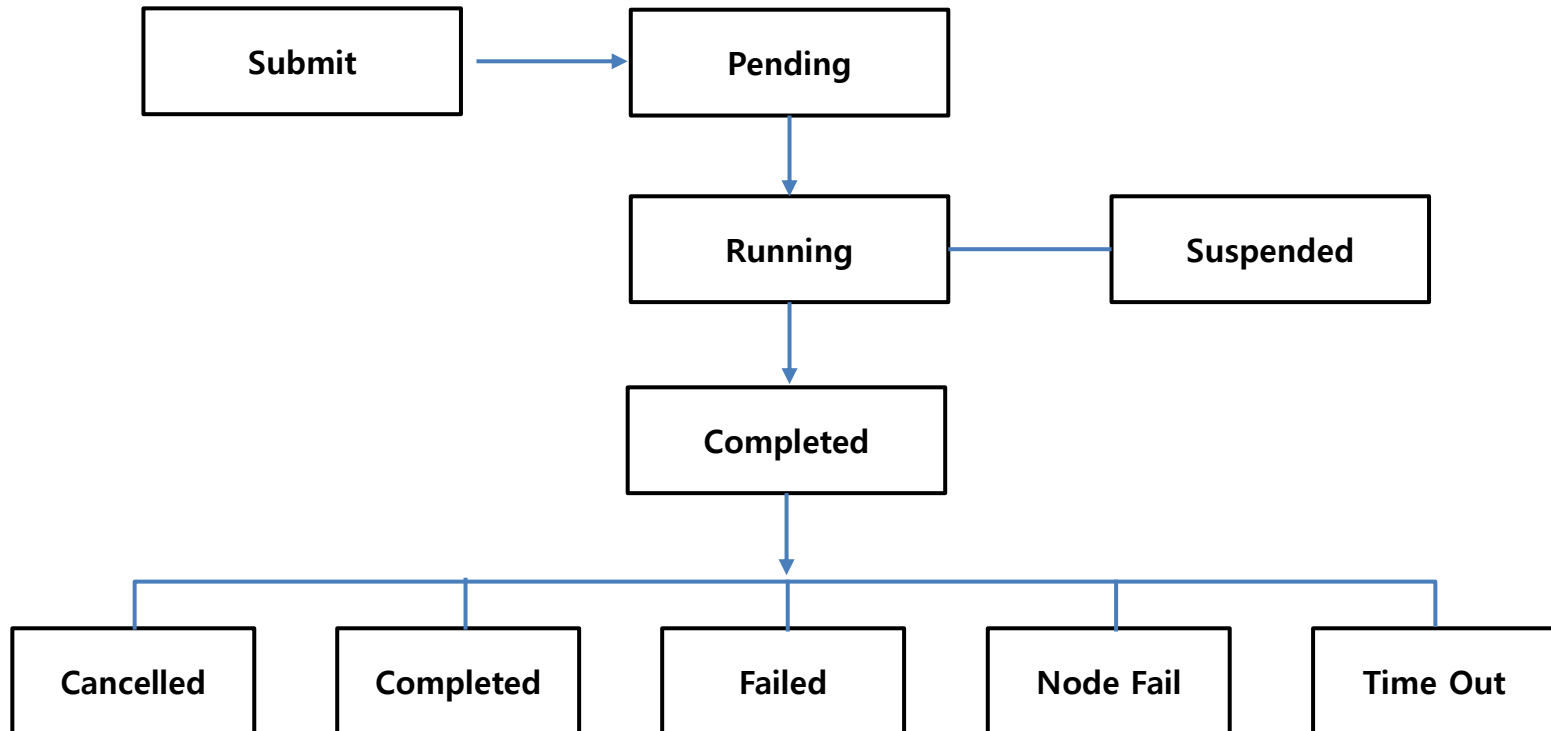
1. SLURM 이란?

3) Slurm 작업 흐름



1. SLURM 이란?

4) Slurm 작업 상태



1. SLURM 이란?

4) Slurm 작업 상태

PENDING

제출한 작업을 위해 클러스터의 리소스를 할당 받기 위한 대기 상태

RUNNING

현재 작업이 실행중인 상태

COMPLETING

작업이 완료중인 상태
일부 노드의 일부 프로세스가 활성화 되어있는 상태

COMPLETED

작업이 종료코드 0으로 모든 노드의 모든 프로세스를 종료된 상태

CANCELLED

사용자 또는 관리자가 작업을 취소된 상태

FAILED

종료코드 0이 아닌 또는 기타 실패조건으로 인해 작업이 종료된 상태

SUSPENDED

필요에 따라 작업이 일시 중지되고 리소스 할당이 해제된 상태
작업이 다시 시작될 수 있는 상태이며 이전 중지된 시점부터 다시 시작

Time Out

작업실행 시간이 최대 실행시간에 도달하여 작업이 종료된 상태

2. SLURM 기본용어

▣ Cluster

- SLURM으로 구성되어 있는 노드, 파티션, 계정(Account), 사용자(User), 작업(Job) 등을 효율적으로 관리하기 위한 자원 Pool을 의미 함

▣ Node

- Cluster 내의 개별적인 기능 수행을 위해 구성된 각각의 computer 자원

▣ Login Node

- 단지, job을 제출만 할 수 있는 node.

▣ Control Node (Master Node)

- 클러스터 전체의 자원상태를 모니터링하고, 자원관리와 작업(Job) 스케줄링을 담당하는 노드

▣ Compute Node

- 실제로 Job이 실행되는 노드로서, Slurm 클러스터를 관리하는 마스터 노드와 통신하여 자원상태를 보고하고 요청받은 Job을 실행하는 노드

▣ Job

- 사용자(User)가 실행하는 프로세스 또는 스크립트 단위로서 실행을 위해 slurm에 제출 된 작업

▣ Partition (Queue)

- Computing 노드들을 논리적으로 그룹화하여 작업이 실행될 수 있는 자원들의 그룹
- 동일한 자원들로 구성된 계산노드들로 구성

▣ Resource

- Job을 수행하는데 이용 가능한 자원 → 예 : CPU, GPU, 메모리 등

▣ Account

- 사용자의 리소스 사용을 관리하고 제어하기 위해 사용하는 개념
(사용자관리, 자원추적, 우선순위 설정, 사용량 확인 을 위한 목적으로 사용)

▣ User (사용자)

- slurm에 작업을 제출하여 사용하는 사용자 명

3. SLURM Command

3. Slurm Command

사용자용 Command

Category	Command	Description
INFO	sinfo	SLURM의 파티션 및 노드 들의 정보를 확인
	squeue	현재 실행중인 작업 내역을 확인
	sacct	기존에 제출된 작업 이력을 확인
	sshare	Fair-Share 정책에서 사용되는 자원정보 및 자원 사용량 확인
RUN	srun	간단하고 즉각적인 작업을 제출용
	salloc	사용자가 특정 리소스를 예약하고 작업을 실행할 환경을 구성
	sbatch	배치 스크립트를 이용한 작업 제출시 사용
CONTROL	scontrol	제출된 작업 및 자원의 상태를 확인
CANCEL	scancel	제출된 작업을 종료

※ 각 명령어들의 사용방법 확인

`$command --help | --usage, man command`

3. Slurm Command

1) INFO (sinfo)

SLURM의 클러스터 정보(파티션 및 노드)를 확인하기 위하여 사용합니다.

```
[dawon@olaf2 ~]$ sinfo
PARTITION    AVAIL  TIMELIMIT  NODES  STATE NODELIST
AIP           up 3-00:00:00      1 down* olaf-g003
AIP           up 3-00:00:00      7   mix olaf-g[001-002,004-008]
mig-3g.40gb   up 3-00:00:00      1   mix olaf-g009
mig-1g.10gb   up 3-00:00:00      1   mix olaf-g010
normal_cpu    up 3-00:00:00     94 alloc olaf-c[001-018,033-035,052-059,0
long_cpu      up 14-00:00:0     94 alloc olaf-c[001-018,033-035,052-059,0
large_cpu     up 3-00:00:00      8 plnd olaf-c[060-063,109-112]
large_cpu     up 3-00:00:00     84 alloc olaf-c[019-032,036-051,072-073,0
large_cpu     up 3-00:00:00      8 idle olaf-c[093-094,156-159,163-164]
olaf_c_core   up 14-00:00:0      1   mix olaf-cl97
olaf_c_core   up 14-00:00:0     15 alloc olaf-c[195-196,198-210]
normal*       up 3-00:00:00      2   mix olaf-cu[1,5]
normal*       up 3-00:00:00      3 alloc olaf-cu[2-4]
long          up 14-00:00:0      2   mix olaf-cu[1,5]
long          up 14-00:00:0      3 alloc olaf-cu[2-4]
jepyc         up infinite      18 idle jepyc[01-14,17-20]
jepyc-rtx     up infinite       1 idle jepyc50
HQ2comp       up infinite      18 idle HQ2comp[03-05,07-16,22-25,28]
HQmem         up infinite       4 idle HQmem[01-04]
[dawon@olaf2 ~]$
```

3. Slurm Command

1) INFO (sinfo)

필드	설명
PARTITION	파티션 이름
AVAIL	해당 파티션의 사용 가능 여부를 확인
TIMELIMIT	해당 파티션을 사용할 수 있는 최대 사용제한 시간
NODES	파티션에 할당된 계산 노드 수
STATE	해당 파티션이 계산 노드들의 상태정보 - Idle : 모든 노드가 사용가능한 상태 - mix : 작업을 수락할 수 있으며 일부 리소스가 다른 작업에서 사용 중인 상태 - drain / draining : 노드에서 실행중인 작업이 완료되기를 기다리는 상태 (새로운 작업할당 불가) - alloc : 모든 리소스가 다른 작업에 사용 중으로 작업을 수락할 수 없는 상태 - down : 서비스 또는 노드가 비정상적으로 동작하여 노드를 사용할 수 없는 상태 - plnd: 해당 노드가 현재 예약되어 있거나 특정 작업을 위해 예약된 상태
NODELIST	파티션을 구성하고 있는 계산 노드 정보

3. Slurm Command

1) INFO (sinfo)

[root@xen2 ~]# sinfo -l

전체 파티션 상세 정보를 출력

PARTITION	AVAIL	TIMELIMIT	JOB_SIZE	ROOT	OVERSUBS	GROUPS	NODES	STATE	NODELIST
AIP	drain	3-00:00:00	1-infinite	no	NO	all	8	mixed	olaf-g[001-002,004-009]
mig-3g.40gb	up	3-00:00:00	1-infinite	no	NO	all	1	mixed	olaf-g003
mig-1g.10gb	up	3-00:00:00	1-infinite	no	NO	all	1	mixed	olaf-g010
normal_cpu	up	3-00:00:00	1-infinite	no	EXCLUSIV	all	87	allocated	olafc[001-018,...]
long_cpu	up	14-00:00:0	1-infinite	no	EXCLUSIV	all	87	allocated	olaf-c[001-018,...]
large_cpu	up	3-00:00:00	30-100	no	EXCLUSIV	all	84	allocated	olaf-c[019-032,...]
olaf_c_core	up	14-00:00:0	1-infinite	no	NO	all	16	mixed	olaf-c[195-210]
normal*	up	3-00:00:00	1-infinite	no	NO	all	5	mixed	olaf-cu[1-5]
long	up	14-00:00:0	1-infinite	no	NO	all	5	mixed	olaf-cu[1-5]
jepyc	up	infinite	1-infinite	no	EXCLUSIV	all	16	idle	jepyc[01-02,05-14,17-20,...]
jepyc-rtx	up	infinite	1-infinite	no	NO	all	1	idle	jepyc50
HQ2comp	up	infinite	1-infinite	no	NO	all	3	mixed	HQ2comp[03,24-25]
HQmem	up	infinite	1-infinite	no	NO	all	4	idle	HQmem[01-04]

[root@xen2 ~]# sinfo -p AIP -l

지정된 파티션의 상세 정보 출력

PARTITION	AVAIL	TIMELIMIT	JOB_SIZE	ROOT	OVERSUBS	GROUPS	NODES	STATE	NODELIST
AIP	up	infinite	1-infinite	no	NO	all	8	idle	olaf-g[001-008]

3. Slurm Command

1) INFO (sinfo)

파티션	용도	정책	Memory per Core
AIP	Only GPU	Shared	7.9 GB
mig-3g.40gb	Only GPU	Shared	7.9 GB
mig-1g.10gb	Only GPU	Shared	7.9 GB
normal_cpu	Only CPU	Exclusive	UNLIMITED
long_cpu	Only CPU	Exclusive	UNLIMITED
normal	Only GPU	Shared	7.1 GB
long	Only GPU	Shared	7.1 GB
jepyc	Only GPU	Exclusive	UNLIMITED
jepyc-rtx	Only GPU	Shared	3.9 GB
HQ2comp	Only CPU	Shared	2.2 GB
HQmem	Only CPU	Shared	12.5 GB

Exclusive(독점) 모드란?

하나의 노드에 하나의 작업만 돌아가게 되고, 단일 코어를 사용하는 작업이어도 해당 노드의 모든 자원을 점유 (자원을 공유하지 않음)

자원 낭비 및 과금 방지를 위하여 작업의 자원 요구사항에 맞는 파티션에서 작업을 돌릴 수 있도록 한다.

Shared 정책이 적용된 파티션에서는 자원을 공유하여 작업을 수행하게 된다.
(자원을 공유하여 사용)

3. Slurm Command

2) INFO (squeue)

제출된 작업 내역을 확인합니다.

```
[dawon@olaf2 ~]$ squeue --me
      JOBID PARTITION    NAME    USER  ST       TIME  NODES NODELIST(REASON)
[dawon@olaf2 ~]$
```

```
[dawon@olaf2 ~]$ squeue | more
      JOBID PARTITION    NAME    USER  ST       TIME  NODES NODELIST(REASON)
    14689543      AIP  script.s  keh0t0 PD        0:00      1 (Resources)
    14689565      AIP    bash  rlawjdgh PD        0:00      1 (Priority)
    14689550      AIP 1024_Din   ytoh PD        0:00      1 (Priority)
    14688882      AIP     NEP  danielhe PD        0:00      1 (Priority)
    14685673      AIP    lexp  quagmire R 1-09:28:09      1 olaf-g004
    14685664      AIP    lto4  quagmire R 1-09:30:15      1 olaf-g006
    14686064      AIP    bash  rlawjdgh R 1-07:00:26      1 olaf-g005
    14689626      AIP    bash  jsh0212 R        33:29      1 olaf-g002
    14689220      AIP    bash  jsh0212 R         3:15:56      1 olaf-g008
    14688208      AIP smdm_set  eunsupar R         5:57:02      1 olaf-g001
    14688207      AIP smdm_ser  eunsupar R         6:28:06      1 olaf-g007
    14688206      AIP smdm_sed  eunsupar R         8:37:30      1 olaf-g001
    14687073      AIP Dinov2_p   ytoh R 1-03:44:04      1 olaf-g007
    14684995      AIP Harim_Si  harimlee R 2-19:52:54      1 olaf-g007
    14687516      AIP     NEP  danielhe R        17:14:03      1 olaf-g002
```

3. Slurm Command

2) INFO (squeue)

필드	설명
JOBID	실행된 JOB에 자동으로 부여되는 고유 번호 (Job 식별 번호)
PARTITION	해당 JOB이 사용하고 있는 파티션 이름
NAME	제출된 JOB 이름
USER	JOB를 실행한 사용자 명
ST	JOB의 실행 상태정보
TIME	제출된 JOB이 실행되고 있는 시간 정보
NODES	JOB에 할당된 계산노드 수
NODELIST (RESON)	JOB에 할당된 계산노드 이름 (RESON) 작업이 수행(Running)되지 않는 원인을 간단하게 설명

3. Slurm Command

2) INFO (squeue)

Job 에 대한 기본 정보 출력

[root@xen2 ~]# squeue

전체 Job list 확인

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
152	debug	sleep	root	R	0:14	1	node-01
150	test1	sleep	admin	R	4:12	1	node-02
151	test1	sleep	root	R	2:54	1	node-02

[root@xen2 ~]# squeue -u admin

지정된 사용자의 Job 정보 확인

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
150	test1	sleep	admin	R	5:29	1	node-02

[root@xen2 ~]# squeue -p debug

지정된 파티션의 Job 정보 확인

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
152	debug	sleep	root	R	0:29	1	node-01

3. Slurm Command

2) INFO (squeue)

```
[root@xen2 ~]# squeue -i 2
```

지정된 시간(초) 만큼 정보를 반복 출력

```
Fri Mar 04 11:44:42 2024
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
152	debug	sleep	root	R	1:46	1	node-01
150	test1	sleep	admin	R	5:44	1	node-02
151	test1	sleep	root	R	4:26	1	node-02

```
Fri Mar 04 11:44:44 2024
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
152	debug	sleep	root	R	1:48	1	node-01
150	test1	sleep	admin	R	5:46	1	node-02
151	test1	sleep	root	R	4:28	1	node-02

3. Slurm Command

3) INFO (sshare)

Account 또는 사용자의 Fair-Share 값을 확인 합니다.

```
[dawon@olaf1 ~]$ sshare -A bmttest
Account      User      RawShares  NormShares  RawUsage  EffectvUsage  FairShare
-----
bmttest      7         0.014675   45926216    0.002844
[dawon@olaf1 ~]$ sshare -u bmttest | grep bmttest
bmttest      7         0.014675   45926216    0.002844
bmttest      bmttest    1         0.500000   45926216    1.000000    0.699653
[dawon@olaf1 ~]$ sshare -a
Account      User      RawShares  NormShares  RawUsage  EffectvUsage  FairShare
-----
root         0.000000   32922564763  1.000000
external     500        0.499500   16150959943  0.490574
ajou         7          0.014675   2402719     0.000149
ajou_ph      7          1.000000   2402719     1.000000
ajou_ph      mdseo      1          1.000000   2402719     1.000000    0.769097
bmttest      7          0.014675   45926216    0.002844
bmttest      bmttest    1          0.500000   45926216    1.000000    0.699653
bmttest      test_shc   1          0.500000   0           0.000000    0.701389
cathol       7          0.014675   11367735    0.000704
catholic_sbtl 7          1.000000   11367735    1.000000
catholic_sbtl jmchung    1          1.000000   11367735    1.000000    0.760417
cau          7          0.014675   0           0.000000
sitd         7          1.000000   0           0.000000
sitd         sitd2008   1          1.000000   0           0.000000    0.998264
cnu          7          0.014675   0           0.000000
cnu          donna      1          1.000000   0           0.000000    0.998264
edu          7          0.014675   15517       0.000001
edu          edu01      1          0.024390   0           0.000000    0.854167
edu          edu02      1          0.024390   0           0.000000    0.854167
edu          edu03      1          0.024390   0           0.000000    0.854167
```

3. Slurm Command

3) INFO (sshare)

Account 또는 사용자의 Fair-Share 값을 확인 합니다.

필드	설명
Account	Account 이름
User	사용자(User) 이름
RawShare	Account에 할당된 자원의 양 - 클러스터 내에서 해당 Account에 할당된 자원의 상대적인 비율
NormShares	자원 할당 비율 - RawShare를 전체 Share 합계로 나눈 값
RawUsage	Account 또는 사용자의 자원 사용량
NormUsage	자원의 사용비율 - RawUsage를 전체 사용량의 합계로 나눈 값.
EffectvUsage	Account의 현재 자원 사용량
FairShare	스케줄링 시 우선순위 결정하는데 사용되는 값 - FairShare 점수는 $\text{NormShares} / (\text{NormUsage} + 1)$ & 기타 요소를 반영

Fair-Share 는 Account 또는 사용자들이 사용하는 자원의 양을 추적하고, 사용자 또는 Account의 우선순위를 조정하여 우선순위에 따라 자원을 공정하게 할당하는 정책입니다.

▣ Fair-Share 특징

- Account 또는 사용자의 자원 사용량에 따라 상대적인 가중치를 반영
- Fair-Share 값이 높을 수록 대기열 중 자원을 할당받는 우선순위가 높아짐
- CPU, GPU 자원의 사용량에 영향을 받음
- 일주일 간격으로 자원 사용량(RawUsage 값)이 1/2로 감소
- 특정 사용자의 과도한 사용을 방지하거나 최적화하기 위한 목적으로 사용

3. Slurm Command

4) RUN (srun)

자원을 할당 받아 하나의 명령을 실행하거나, 디버깅을 위한 Interactive 작업을 수행하는 용도로 사용됩니다.

```
(base) [test1@master ~]$ srun --help
(base) [test1@master ~]$ srun --usage
(base) [test1@master ~]$ srun -J srun_test --pty /bin/bash
(base) [test1@node-01 ~]$ hostname
node-01
(base) [test1@node-01 ~]$ logout
bash: logout: not login shell: use `exit`
(base) [test1@node-01 ~]$ exit
exit
srun: error: node-01: task 0: Exited with exit code 1
(base) [test1@master ~]$ hostname
master
(base) [test1@master ~]$
```

3. Slurm Command

4) RUN (srun)

srun : Command 를 통해 Job Submit

주요 옵션	설명	비고
-N	사용할 노드 수를 지정	--nodes
-n	사용할 프로세스 수 지정	--ntasks
-c	Task당 사용할 코어 수 지정	--cpus-per-task
-t	작업의 최대 실행 시간을 지정	--time
-p	실행할 파티션 지정	--partition
-J	실행할 작업이름 지정	--job-name
-x	실행하지 않을 노드 지정	--exclude
-e	에러 시 에러를 저장할 파일	--error
-o	출력된 결과를 저장할 파일	--output
-v	상세 정보 출력	--verbose
--mem	사용되는 메모리 사용량을 지정(MB)	--mem
--gres	일반 리소스를 지정	--gres
--ntasks-per-node	노드당 프로세스의 수	
--pty	터미널 연결	
--exclusive	노드의 모든 리소스를 독점하여 작업실행	

3. Slurm Command

4) RUN (srun)

사용 예

```
[root@xen2 ~]# srun -p edu --nodes=1 --ntasks=1 --cpus-per-task=2 --nodelist=node-02 sleep 100
```

```
[root@xen2 ~]# squeue --me
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
174	test1	random_name	root	R	0:15	1	node-02

※ 제출작업 설명

edu 파티션의 1개 Node , 1개 프로세스, 2개 CPU core를 할당하고, node-02에서 sleep 100 명령을 실행

3. Slurm Command

4) RUN (srun)

사용 예

```
[root@xen2 ~]# srun -J jobtest -e jobtest_%j.err -o jobtest_%j.out -N 1 -n 2 sleep 100
```

```
[root@xen2 ~]# squeue --me
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
174	edu	jobtest	root	R	0:15	1	node-02

※ 제출작업 설명

job 이름은 jobtest로 설정하고 1개 Node , 2개 프로세스를 사용하여 sleep 100 명령을 수행.

작업진행 중 발생하는 error 로그는 jobtest_%jobID.err , 작업결과는 jobtest_%jobID.out 파일로 저장.

3. Slurm Command

4) RUN (srun)

실 습

▣ TMUX 실행

`$tmux new -s edu??`

`$ctrl + b 후 %`

- 화면이동: `ctrl + b`, 방향키

현재 화면을 세로분할

▣ `srun -p edu -N 2 -n2 hostname`

- 2개의 노드에 2개의 프로세스를 이용하여 `hostname` command 결과를 프롬프트에 출력

▣ `srun -p edu -N 2 -n2 -o srun_%j.out -e srun_%j.err hostname`

- 2개의 노드에 2개의 프로세스를 이용하여 `hostname` command 결과를 `srun_%j.out` 파일에 출력

▣ `srun -p edu -J my_test -N 2 -n2 -c 4 sleep 100`

- `my_test` 작업을 2개의 노드에 2개의 프로세스, 프로세스별 4개의 CPU(2*4)를 이용하여 "sleep 100" 실행

▣ `watch queue --me` (다른 화면에서)

- 현재 실행중인 `my_test` 작업 확인

3. Slurm Command

4) RUN (srun)

실 습

❑ **hostname**

❑ **srun -p edu -N 1 -n 1 -c 1--pty bash**

– 1개의 노드에 1개의 프로세스를 bash 셸을 실행

❑ **squeue --me**

❑ **hostname**

❑ **exit**

❑ **srun -p edu -N 1 -n 2 hostname**

– 1개의 노드에 2개의 프로세스를 이용하여 hostname 명령을 실행

❑ **srun -p edu -N 2 -n 2 hostname**

– 2개의 노드에 2개의 프로세스를 이용하여 hostname 명령을 실행

3. Slurm Command

5) RUN (salloc)

srn 명령과 유사하지만, salloc는 자원만 할당 받은 상태를 구성하게 되며 구성된 환경의 프롬프트 상태에서 command 들을 실행

```
[dawon@olaf1 ~]$ salloc -p AIP -n1
salloc: Granted job allocation 14695850
salloc: Waiting for resource configuration
salloc: Nodes olaf-g002 are ready for job
```

```
[dawon@olaf1 ~]$ squeue --me
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
14695850	AIP	interact	dawon	R	0:14	1	olaf-g002

실 습

❑ **salloc -p edu -J my_test -N 1 -n 1**

– my_test 작업을 위해 1개의 노드에 1개의 프로세스의 Resource 요청

❑ **squeue --me**

– 현재 실행중인 my_test 작업 확인

❑ **hostname**

❑ **exit**

❑ **salloc -p edu -J my_test -N 1 -n 1 srun -N 1 -n 1 hostname**

❑ **hostname**

3. Slurm Command

6) RUN (sbatch)

작성된 배치 스크립트의 디렉티브(#SBATCH)에 따라 작업을 위한 자원구성을 요청하게 되며,
자원이 할당되면 각 STEP에 따라 작업이 수행 됨

sbatch 스크립트 작성 : batch job 실행

```
$ vi test.sh
```

```
-----  
#!/bin/bash  
#SBATCH -p [파티션 명]  
#SBATCH -J my_test  
#SBATCH -o test_%j.out  
#SBATCH -e test_%j.err  
#SBATCH -N 1  
#SBATCH -n 2
```

```
sleep 1000
```

```
-----  
$ sbatch test.sh
```

※ 주의사항

sbatch 명령 없이 배치스크립트만을 실행 할 경우에는 로그인 노드에서 작업이 실행되며

이런 경우 전체 클러스터에 영향을 미칠 수 있으므로 주의 부탁드립니다.

3. Slurm Command

6) RUN (sbatch)

작성된 스크립트의 디렉티브(#SBATCH)에 따라 작업을 위한 자원구성을 요청하게 되며,
자원이 할당되면 각 STEP에 따라 작업이 수행 됨

```
$ vi test.sh
#!/bin/bash
#SBATCH -J test
#SBATCH -p debug
#SBATCH -N 2
#SBATCH -n 2
#SBATCH --gres=gpu:1

module purge
module load example/2024.11.11

cd $SLURM_SUBMIT_DIR
export I_MPI_PMI_LIBRARY=/opt/local/slurm/default/lib64/libpmi.so
srun --mpi-pmi2 -n 2 -N 2 ./test1.exe
srun --mpi-pmi2 -n 2 -N 2 ./test2.exe
```

[test@master ~]\$ sbatch test.sh

3. Slurm Command

6) RUN (sbatch)

(디렉티브)옵션	설명	비고
#SBATCH -J	작업명 지정	--job-name
#SBATCH -t	최대 작업수행 시간	--time
#SBATCH -o	작업 로그 파일 지정	--output
#SBATCH -e	작업 에러 파일 지정	--error
#SBATCH -p	파티션 지정	--partition
#SBATCH --comment	작업에 대한 주석	--comment
#SBATCH -w	작업수행 노드 지정	--odelist
#SBATCH -N	작업수행 노드 수	--nodes
#SBATCH -n	노드당 수행될 프로세스 수 지정	--ntasks
#SBATCH -c	프로세스 당 할당된 CPU core 수	--cpus-per-task
#SBATCH --gres	특정 리소스를 지정	--gres
#SBATCH --cpus-per-gpu	GPU 당 할당 될 CPU Core 수	--cpus-per-gpu
#SBATCH --exclusive	노드의 모든 리소스를 독점적으로 사용	--exclusive

3. Slurm Command

6) RUN (sbatch)

실 습

1. sbatch 스크립트 작성

```
--- vi test.sh -----  
#!/bin/bash  
#SBATCH -p edu  
#SBATCH -J my_test  
#SBATCH -o my_test_%j.out  
#SBATCH -e my_test_%j.err  
#SBATCH -N 1  
#SBATCH -n 1  
  
srun hostname  
sleep 1000  
:wq!  
-----
```

2. 작업 실행 및 결과 확인

```
$ sbatch test.sh  
  
$ squeue --me  
  
$ ls -l *.out  
  
$ cat *.out  
  
$ rm *.out *.err
```

3. Slurm Command

6) RUN (sbatch)

실 습

1. sbatch 스크립트 작성

```
$ vi test.sh
#!/bin/bash
#SBATCH -p edu
#SBATCH -J my_test
#SBATCH -o my_test_%j.out
#SBATCH -e my_test_%j.err
#SBATCH -N 2
#SBATCH -n 2

srun hostname

:wq!
```

2. 작업 실행 및 결과 확인

```
$ sbatch test.sh

$ squeue --me

$ ls -l *.out

$ cat *.out

$ rm *.out *.err

$ scancel
```

3. Slurm Command

6) RUN (sbatch)

EXCLUSIVE 모드에서의 Multi Job 예시 (참고)

```
#!/bin/bash
#SBATCH -J multi_job
#SBATCH -p normal_cpu
#SBATCH -N 1
#SBATCH --ntasks-per-node=7
#SBATCH -ntask=7
#SBATCH --cpus-per-task=20

.....
module load gcc/12.2.0 anaconda/23.09.0
export Prog=[/path/to/Binary/file]
export I_MPI_PMI_LIBRARY=/usr/lib64/libpmi.so
mkdir ${SLURM_JOB_ID}_output

run_Prog() {
    srun -n 1 --exclusive=user --mem-per-cpu=32gb $Prog $1 > ${SLURM_JOB_ID}_output/${SLURM_JOB_ID}_$2.out &
}

for i in $(seq 1 ${SLURM_NTASKS})
do
    run_Prog input $i
done

wait
```

3. Slurm Command

7) CONTROL (scontrol)

클러스터 요소(Job, Partition, node)들의 상태나 정보들을 확인하기 위한 명령어

```
[dawon@olaf1 ~]$ squeue | more
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST (REASON)
14695683	AIP	script.s	keh0t0	PD	0:00	1	(Priority)
14695682	AIP	script.s	keh0t0	PD	0:00	1	(Resources)
14694059	AIP	bash	rlawjdgh	PD	0:00	1	(Priority)
14695653	AIP	bash	rlawjdgh	PD	0:00	1	(Priority)
14685673	AIP	lexp	quagmire	R	2-07:53:10	1	olaf-g004
14685664	AIP	lto4	quagmire	R	2-07:55:16	1	olaf-g006
14686064	AIP	bash	rlawjdgh	R	2-05:25:27	1	olaf-g005
14690983	AIP	sh	train	jsh0212	R	3:02:05	1 olaf-g007

```
[dawon@olaf1 ~]$ scontrol show job 14695682
JobId=14695682 JobName=script.sn
  UserId=keh0t0(46406) GroupId=davian(46400) MCS_label=N/A
  Priority=6520 Nice=0 Account=davian QOS=
  JobState=PENDING Reason=Resources Dependency=(null)
  Requeue=1 Restarts=0 BatchFlag=1 Reboot=0 ExitCode=0:0
  RunTime=00:00:00 TimeLimit=3-00:00:00 TimeMin=N/A
  SubmitTime=2024-10-30T15:19:23 EligibleTime=2024-10-30T15:19:24
  AccrueTime=2024-10-30T15:19:24
  StartTime=2024-10-31T09:56:46 EndTime=2024-11-03T09:56:46 Deadline=N/A
  SuspendTime=None SecsPreSuspend=0 LastSchedEval=2024-10-30T17:53:36 Scheduler=Main
  Partition=AIP AllocNode:Sid=olaf1-mg:2788438
  ReqNodeList=(null) ExcNodeList=(null)
  NodeList=(null) SchedNodeList=olaf-g006
  NumNodes=1-1 NumCPUs=1 NumTasks=1 CPUs/Task=1 ReqB:S:C:T=0:0:*:*
  TRES=cpu=1,mem=7900M,node=1,gres/gpu:a40=4
  Socks/Node=* NtasksPerN:B:S:C=0:0:*:*:1 CoreSpec=*
  MinCPUsNode=1 MinMemoryCPU=7900M MinTmpDiskNode=0
  Features=(null) DelayBoot=00:00:00
```

3. Slurm Command

7) CONTROL (scontrol)

```
[root@xen2 ~]# scontrol show job 1111111(job_id)
  UserId=dawon(14903) GroupId=MIT(14903) MCS_label=N/A
  Priority=10909 Nice=0 Account=mit QOS=
  JobState=RUNNING Reason=None Dependency=(null)
  Requeue=1 Restarts=0 BatchFlag=1 Reboot=0 ExitCode=0:0
  RunTime=00:00:13 TimeLimit=3-00:00:00 TimeMin=N/A
  SubmitTime=2024-11-01T14:47:29 EligibleTime=2024-11-01T14:47:29
  AccrueTime=2024-11-01T14:47:29
  StartTime=2024-11-01T14:47:30 EndTime=2024-11-04T14:47:30 Deadline=N/A
  SuspendTime=None SecsPreSuspend=0 LastSchedEval=2024-11-01T14:47:30
  Partition=AIP AllocNode:Sid=olaf1-mg:3814843
  ReqNodeList=(null) ExcNodeList=(null)
  NodeList=olaf-g005
  BatchHost=olaf-g005
  NumNodes=1 NumCPUs=2 NumTasks=1 CPUs/Task=1 ReqB:S:C:T=0:0:*:*
  TRES=cpu=2,mem=15800M,node=1
```

지정된 작업의 상세 정보 출력

사용자 명

제출된 Job 상태

제출된 Job 실행된 시간, 최대실행시간

Job 제출된 시간

작업실행 시간

Job 할당된 노드

Batch Job을 처리하는 Host

Job 에 할당된 자원 정보

3. Slurm Command

7) CONTROL (scontrol)

```
[root@xen2 ~]# scontrol show job 1111111(job_id)
```

지정된 작업의 상세 정보 출력

```
Socks/Node=* NtasksPerN:B:S:C=0:0:*:1 CoreSpec=*
```

```
MinCPUsNode=1 MinMemoryCPU=7900M MinTmpDiskNode=0
```

```
Features=(null) DelayBoot=00:00:00
```

```
OverSubscribe=OK Contiguous=0 Licenses=(null) Network=(null)
```

```
Command=./test.sh
```

Job에서 실행된 명령

```
WorkDir=/proj/home/ibs/MIT/dawon/test
```

Job 이 실행중인 디렉터리

```
StdErr=/proj/home/ibs/MIT/dawon/test/my_test_14702153.err
```

Job 에러 파일 위치

```
StdIn=/dev/null
```

```
StdOut=/proj/home/ibs/MIT/dawon/test/my_test_14702153.out
```

Job 실행결과 저장 위치

3. Slurm Command

7) CONTROL (scontrol)

```
[root@xen2 ~]# scontrol show partition HQmem
```

지정된 파티션의 상세 정보 출력

```
PartitionName=HQmem
```

```
AllowGroups=ALL AllowAccounts=ALL AllowQos=ALL
```

지정된 파티션을 사용할 수 있는 사용자 정보

```
AllocNodes=ALL Default=NO CpuBind=cores QoS=N/A
```

```
DefaultTime=NONE DisableRootJobs=YES ExclusiveUser=NO GraceTime=0 Hidden=NO
```

```
MaxNodes=UNLIMITED MaxTime=UNLIMITED MinNodes=0 LLN=NO MaxCPUsPerNode=UNLIMITED
```

```
Nodes=HQmem[01-04]
```

지정된 파티션의 노드 정보

```
PriorityJobFactor=2 PriorityTier=2 RootOnly=NO ReqResv=NO OverSubscribe=NO
```

```
OverTimeLimit=NONE PreemptMode=OFF
```

```
State=UP TotalCPUs=80 TotalNodes=4 SelectTypeParameters=NONE
```

지정된 파티션 총 CPU수

```
JobDefaults=(null)
```

```
DefMemPerCPU=12500 MaxMemPerCPU=12800
```

지정된 파티션 Core 당 기본 할당 메모리

```
TRESBillingWeights=CPU=5.5
```

지정된 파티션 사용시 적용되는 가중치

3. Slurm Command

7) CONTROL (scontrol)

```
[root@xen2 ~]# scontrol show node olaf-cu1
```

지정된 노드의 상세 정보 출력

```
NodeName=olaf-cu1 Arch=x86_64 CoresPerSocket=26
```

```
CPUAlloc=24 CPUTot=104 CPULoad=6.92
```

현재 작업이 할당되어 사용중인 CPU 수

```
AvailableFeatures=Gold-6230R,V100
```

```
ActiveFeatures=Gold-6230R,V100
```

```
Gres=gpu:V100:8(S:0-1)
```

현재 작업에 할당되어 사용중인 GPU 수

```
NodeAddr=olaf-cu1 NodeHostName=olaf-cu1 Version=21.08.5
```

```
OS=Linux 4.18.0-372.9.1.el8.x86_64 #1 SMP Tue May 10 14:48:47 UTC 2022
```

```
RealMemory=773685 AllocMem=142000 FreeMem=729119 Sockets=2 Boards=1
```

```
State=MIXED ThreadsPerCore=2 TmpDisk=0 Weight=1 Owner=N/A MCS_label=N/A #현재 노드 상태
```

```
Partitions=normal,long
```

해당노드에 할당된 파티션 이름

```
BootTime=2024-09-20T13:59:55 SlurmdStartTime=2024-09-20T14:05:46
```

```
LastBusyTime=2024-11-07T09:40:32
```

```
CfgTRES=cpu=104,mem=773685M,billing=104
```

```
AllocTRES=cpu=24,mem=142000M,gres/gpu:v100=8
```

현재 사용중인 자원 정보

```
CapWatts=n/a
```

```
CurrentWatts=0 AveWatts=0
```

```
ExtSensorsJoules=n/s ExtSensorsWatts=0 ExtSensorsTemp=n/s
```

3. Slurm Command

7) CONTROL (scontrol)

실 습

1. sbatch 스크립트 작성

```
--- vi test.sh -----  
#!/bin/bash  
#SBATCH -p edu  
#SBATCH -J edu_test  
#SBATCH -o my_test_%j.out  
#SBATCH -e my_test_%j.err  
#SBATCH -N 1  
#SBATCH -n 1  
  
sleep 1000  
:wq!  
-----
```

2. 작업 실행 및 결과 확인

```
$sbatch test.sh  
  
$squeue -me  
  
$scontrol show job jobID
```

8) STOP (scancel)

Job 종료

```
[root@xen2 admin]# squeue
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
157	debug	test1	admin	R	4:56	1	xen2
159	test1	test1	admin	PD	0:00	1	(Resources)
161	test1	test1	root	PD	0:00	1	(Priority)
158	test1	test1	admin	R	4:13	1	xen2

```
[root@xen2 admin]# scancel 158
```

지정된 Job 종료

```
[root@xen2 admin]# squeue
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
157	debug	test1	admin	R	4:56	1	xen2
159	test1	test1	admin	PD	0:00	1	(Resources)
161	test1	test1	root	PD	0:00	1	(Priority)

3. Slurm Command

8) STOP (scancel)

실 습

▣ **squeue -me**

– 현재 실행중인 my_test 작업 ID를 확인

▣ **scancel job_id**

4. 이용내역 확인

4. 이용내역 조회

2) 이용료 조회 (billing)

- **Path** : `/opt/bin/billing.pl`
- **Options** : `./billing.pl -help`

```
[olaf1] ~$ /opt/bin/billing.pl -help
Usage: ./billing.pl [-u User_Name] [-P Partition_Name] [-S Start_date] [-E End_date] [-m Month] [-g Group_Name]
```

- **Example** : `./billing.pl -S 2024-09-01 -E 2024-10-16`

```
[edu11@olaf1 bin]$ ./billing.pl -P AIP -S 2024-07-01 -E 2024-08-01
=====
| User      | Partition | Usage(days) | Unit          | Rate  | Cost(원) |
|-----|-----|-----|-----|-----|-----|
| edu11     | AIP       | 1           | GPU * Day    | 60600 | 60600    |
|-----|-----|-----|-----|-----|-----|
| olaf-g total: 60600 원
```

- 로그인 노드에서 작업이 실행되지 않도록 주의
- 다른 사용자의 파일접근 금지
- 중요한 파일의 퍼미션은 700 으로 설정
- 소프트웨어 라이선스 정책 준수
- 제출된 작업이 정상적으로 수행되지 않는 등 문의사항이 있는 경우
작업했던 내용과 메시지/에러로그 등을 포함하여 메일로 전달
- VPN을 통하지 않는 일반적인 시스템 접근을 금지

