AWS SSM 을 통한 VPC Private Network 접근

목적

- VPC 내의 Private Network 에 외부 공인 대역에서 Shell 로 접근 (VPN 역할)

특징

- SSM 을 통해 AWS 연결을 하는 것으로, IGW 나 EIP 등의 VPC내 별도 제약이 필요없다
- EC2 SSH 접근 용도의 Password 나 Key-Pair 가 필요없다.
- Shell 전환으로 SSH 와 거의 동일하게 사용할 수 있다.
- AWS Client VPN 과 비교 시, 비용적으로 유리하다.

요구사항

- AWS CLi 를 설치 및 AWS 리소스에 접근할 수 있는 공인망 환경 (VM / CT / Server)
  일부 리전의 경우 AWS Console 내에서 Cloudshell 사용 가능

설치 순서

- 1) Private Network 에 접근할 수 있는 IAM 권한 설정
- 2) EC2 에 IAM Role 추가
- 3) EC2 에 SSM Agent 설치
- 4) AWS CLI 를 통한 EC2 연결

1. Key 방식 전용의 IAM 생성 후 권한 설정


– 관리형 정책을 만들어서IAM 에 할당한다.

arn 은 리전과 어카운트와 접근할 EC2 ID 를 적는다

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                 "arn:aws:ec2:us-west-2:1234567890:instance/i-ahe52134fxed6"

            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:username}-*"
            ]
        }
    ]
}
```

2. **IAM Custom Role 을 생성한 후 VPC 내의 EC2 에 설정한다**

– Role 은 SSM InstanceCore 만 선택

– Role Name 지정 후 생성

EC2 의 보안 설정에서 해당 롤을 추가한다

3. **EC2 에 SSM-Agent 를 설치한다**

Aamazon 2 리눅스 인스턴스의 경우에는 기본 설정되어 있을 수 있다.

```
[root@ip-10-10-20-201 ~]# sudo yum install -y
```

https://s3.region.amazonaws.com/amazon-ssm-region/latest/linux_amd64/amazon-ssm-agent.rpm
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Cannot open: https://s3.region.amazonaws.com/amazon-ssm-region/latest/linux_amd64/amazon-ssm-agent.rpm. Skipping.
Error: Nothing to do
[root@ip-10-10-20-201 ~]# wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_amd64/amazon-ssm-agent.rpm
--2022-03-29 02:49:06-- https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_amd64/amazon-ssm-agent.rpm
Resolving s3.amazonaws.com (s3.amazonaws.com)... 52.217.196.240
Connecting to s3.amazonaws.com (s3.amazonaws.com)|52.217.196.240|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 26724168 (25M) [binary/octet-stream]
Saving to: ‘amazon-ssm-agent.rpm’

100%[==========>] 26,724,168 12.7MB/s in 2.0s

2022-03-29 02:49:08 (12.7 MB/s) - ‘amazon-ssm-agent.rpm’ saved [26724168/26724168]

설치 후 자동시작 활성화한다.
[root@ip-10-10-20-201 ~]# **rpm -Uvh amazon-ssm-agent.rpm**
warning: amazon-ssm-agent.rpm: Header V4 RSA/SHA1 Signature, key ID 693eca21: NOKEY
Preparing...
################################# [100%]
Updating / installing...
1:amazon-ssm-agent-3.1.1080.0-1
################################# [100%]
Created symlink from /etc/systemd/system/multi-user.target.wants/amazon-ssm-agent.service to /etc/systemd/system/amazon-ssm-agent.service.
[root@ip-10-10-20-201 ~]# **systemctl enable amazon-ssm-agent**

```
[root@ip-10-10-20-201 ~]# systemctl start amazon-ssm-agent
```

```
[root@ip-10-10-20-201 ~]# systemctl status amazon-ssm-agent
 -- amazon-ssm-agent.service - amazon-ssm-agent
     Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2022-03-29 02:53:54 UTC; 21s ago
 Main PID: 3355 (amazon-ssm-agen)
   CGroup: /system.slice/amazon-ssm-agent.service
           ├─3355 /usr/bin/amazon-ssm-agent
           └─3382 /usr/bin/ssm-agent-worker

Mar 29 02:53:54 ip-10-10-20-201.us-west-2.compute.internal amazon-ssm-agent[3355]: 2022-03-29 02:53:54 INFO Agent will take identity f...EC2
Mar 29 02:53:54 ip-10-10-20-201.us-west-2.compute.internal amazon-ssm-agent[3355]: 2022-03-29 02:53:54 INFO [amazon-ssm-agent] using n...IPC
Mar 29 02:53:55 ip-10-10-20-201.us-west-2.compute.internal amazon-ssm-agent[3355]: 2022-03-29 02:53:54 INFO [amazon-ssm-agent] using n...IPC
Mar 29 02:53:55 ip-10-10-20-201.us-west-2.compute.internal amazon-ssm-agent[3355]: 2022-03-29 02:53:54 INFO [amazon-ssm-agent] using n...IPC
Mar 29 02:53:55 ip-10-10-20-201.us-west-2.compute.internal amazon-ssm-agent[3355]: 2022-03-29 02:53:54 INFO [amazon-ssm-agent] amazon-...0.0
Mar 29 02:53:55 ip-10-10-20-201.us-west-2.compute.internal amazon-ssm-agent[3355]: 2022-03-29 02:53:54 INFO [amazon-ssm-agent] OS: lin...d64
Mar 29 02:53:55 ip-10-10-20-201.us-west-2.compute.internal amazon-ssm-agent[3355]: 2022-03-29 02:53:54 INFO [CredentialRefresher] Iden...her
Mar 29 02:53:55 ip-10-10-20-201.us-west-2.compute.internal amazon-ssm-agent[3355]: 2022-03-29 02:53:55 INFO [amazon-ssm-agent] [LongRu...ess
Mar 29 02:53:55 ip-10-10-20-201.us-west-2.compute.internal amazon-ssm-agent[3355]: 2022-03-29 02:53:55 INFO [amazon-ssm-
```

```
agent] [LongRu...ted
Mar 29 02:53:55 ip-10-10-20-201.us-west-2.compute.internal
amazon-ssm-agent[3355]: 2022-03-29 02:53:55 INFO [amazon-ssm-
agent] [LongRu...nds
Hint: Some lines were ellipsized, use -l to show in full.
```

4. AWS Cli 에서 SSM 을 통한 EC2 접속

외부 CT 에 AWSCli 를 설치 후 IAM API Key 권한을 설정한다.

```
## 접속할 Client의 IP 확인
$ curl http://icanhazip.com
1.2.3.4

## AWSCli 설치
(중요) SSM의 Session-Plugin 을 사용하기 위해서는 AWSCli 1.16 버전 이상
이어야 한다.
$ curl
"https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
"awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install

## Linux 용 SSM Session-Manager Plugin 설치
$ curl
"https://s3.amazonaws.com/session-manager-downloads/plugin/lat
est/linux_64bit/session-manager-plugin.rpm" -o "session-
manager-plugin.rpm"
$ rpm -Uvh session-manager-plugin.rpm
$ session-manager-plugin
The Session Manager plugin was installed successfully. Use the
AWS CLI to start a session.

## 접근 정보 설정
본문 내용의 키는 임의 생성한 내용.
$ aws configure
AWS Access Key ID [None]: AKIAQ25632EPN7T7FFVT
AWS Secret Access Key [None]:
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
Default region name [None]: us-west-2
Default output format [None]: json
```

SSM 을 통한 실제 EC2 에 접근

```
[root@node1 ~]# aws ssm start-session --target i-064f7ebc0bed75c74

Starting session with SessionId: SSM-Only-0a9041d6b13f368ce
sh-4.2$ bash
[ssm-user@ip-10-10-20-201 bin]$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 9001
        inet 10.10.20.201  netmask 255.255.255.0  broadcast 10.10.20.255
        inet6 fe80::aa:14ff:fed7:abd  prefixlen 64  scopeid 0x20<link>
        ether 02:aa:14:d7:0a:bd  txqueuelen 1000  (Ethernet)
        RX packets 31846  bytes 8338621 (7.9 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 29180  bytes 6068149 (5.7 MiB)
         TX errors 0  dropped 0 overruns 0  carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
         TX errors 0  dropped 0 overruns 0  carrier 0 collisions 0

[ssm-user@ip-10-10-20-201 bin]$
```

※ 참고자료
https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html
https://docs.aws.amazon.com/ko_kr/systems-manager/latest/userguide/session-manager-working-with-install-plugin.html
https://docs.aws.amazon.com/ko_kr/systems-manager/latest/userguide/session-manager-getting-started.html

암호화된 AMI 이미지를 AWS 계정간 공유하려면?

CMK 공유 설정 없이, 암호화된 볼륨이 있는 프라이빗 AMI 를 AWS 계정간에 공유하면
새 인스턴스 시작 시에 에러 메세지 없이 생성 후 그냥 인스턴스가 자동 종료(Terminated)되어 버립니다.

키 또한 권한을 공유하여 사용해야 합니다.

```
# 공유할 이미지를 지정해 주는 부분입니다.
# 원본 AMI 계정에 IAM 관리형 정책을 만들고 IAM 에 할당합니다.

{
   "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ModifyImageAttribute"
            ],
            "Resource": [
                                                "arn:aws:ec2:us-
west-2::image/<0e9fcdb7ae40e8f4c>"
## 공유할 이미지의 리전 위치와 id ( ami-xxxxxxxxxxxxxxxxx  에서 xxx 부분입니다 )
            ]
        }
 ]
}

# KMS 서비스에서 Key 를 공유 가능한 어카운트를 설정합니다.
# 원본 AMI 계정의 KMS 서비스에 가서 공유 할 AWS Account 넘버를 공유에 등록해 줍니다.
```

```
# 공유 받은 쪽에서 복호화할 CMK 를 요청하는 부분입니다.
# 공유 받은 계정의 IAM 에 관리형 정책을 등록합니다.
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:DescribeKey",
                "kms:ReEncrypt*",
                "kms:CreateGrant",
                "kms:Decrypt"
            ],
            "Resource": [
                                                "arn:aws:kms:us-
west-2:891977874274:key/bc52517a-676b-4f1a-9d1a-d50241563abc"
## 원본 AMI 의 KMS Key 권한을 요청하는 정책입니다.
            ]
        }
    ]
}
```

이제 정상적으로 공유 받은 이미지에서 EC2 가 생성되고 정상 시작되는 것을 확인할 수 있습니다.

참고 AWS Docs : https://aws.amazon.com/ko/blogs/security/how-to-share-encrypted-amis-across-accounts-to-launch-encrypted-ec2-instances/

CentOS 7에서 APM 최신 버전 yum 으로 설치하기

고정적인 버전 관리가 필요하지 않거나 간단한 설치가 필요할 경우 yum 을 이용할 수 있습니다.

Image OS : CentOS 7.6.1810 Minimal

공통사항
추가 패키지를 설치하기 위해 우선 확장 레포지토리를 설치해 줍니다
```
yum install -y epel-release
```

Apache 2.4.52
아파치는 보통 빌드를 통해 버전 관리용 RPMs 패키지를 만듭니다.
우선 외부 레포지토리(codeit)를 사용합니다.

```
cd /etc/yum.repos.d/ && wget https://repo.codeit.guru/codeit.el`rpm -q --qf "%{VERSION}" $(rpm -q --whatprovides redhat-release)`.repo
```

사용가능 버전 확인
```
yum info httpd
```

```
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirror.kakao.com
 * epel: hk.mirrors.thegigabit.com
 * extras: mirror.kakao.com
 * remi-safe: mirror.bebout.net
 * updates: mirror.navercorp.com
Installed Packages
Name        : httpd
Arch        : x86_64
Version     : 2.4.52
Release     : 1.codeit.el7
Size        : 4.3 M
Repo        : installed
From repo   : CodeIT
Summary     : Apache HTTP Server
URL         : https://httpd.apache.org/
License     : ASL 2.0
Description : The Apache HTTP Server is a powerful, efficient,
and extensible
            : web server.
```

Apache 설치
```
yum --enablerepo=CodeIT install httpd mod_ssl
```

PHP 7.4 설치를 위한 Remi Repository 를 설치합니다.
```
yum install
```

```
https://rpms.remirepo.net/enterprise/remi-release-7.rpm

# 전체 PHP 버전 리스트를 확인 후 7.4 버전 설치
yum repolist all | grep -i php
yum --enablerepo=remi-php74 install php  php-opcache php-gd
php-mysql php-xml

# MariaDB 10.3 설치를 위한 공식 레포지토리 설정
cat << EOF | tee /etc/yum.repos.d/MariaDB.repo
[mariadb]
name = MariaDB
baseurl = http://yum.mariadb.org/10.3/centos7-amd64
gpgkey=https://yum.mariadb.org/RPM-GPG-KEY-MariaDB
gpgcheck=1
enabled=0
EOF


# MariaDB 10.3 설치
yum -y install --enablerepo=mariadb MariaDB-server MariaDB-
client MariaDB-backup


# APM 버전 확인
[root@localhost ~]# php -v
PHP 7.4.28 (cli) (built: Feb 15 2022 13:23:10) ( NTS )
Copyright (c) The PHP Group
Zend Engine v3.4.0, Copyright (c) Zend Technologies
     with Zend OPcache v7.4.28, Copyright (c), by Zend
Technologies
[root@localhost ~]# mysql --version
mysql  Ver 15.1 Distrib 10.3.34-MariaDB, for Linux (x86_64)
using readline 5.1
[root@localhost ~]# httpd -v
Server version: Apache/2.4.52 (codeit)
Server built:   Dec 20 2021 11:29:54
```

Windows 2012 원격 세션 늘리기

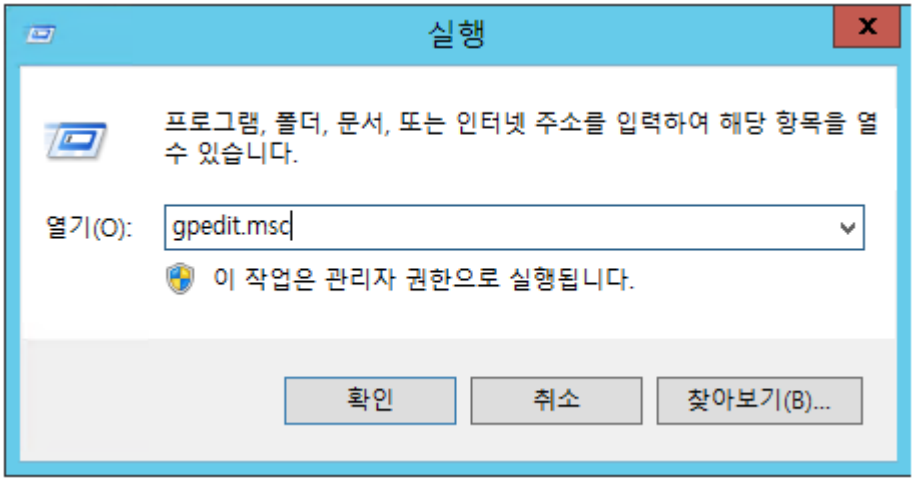
Windows Server 2012 설치 이후 고객사에서 원격 접속으로 접속하여 작업시 다른 세션의 접속으로 인하여 세션이 끊기며 서로 뺏어가는 경우가 자주 발생하게 됩니다.

Windows 2012의 경우 1개의 세션만을 공유하도록 기본으로 설정되어 있어 발생하는 문제입니다.

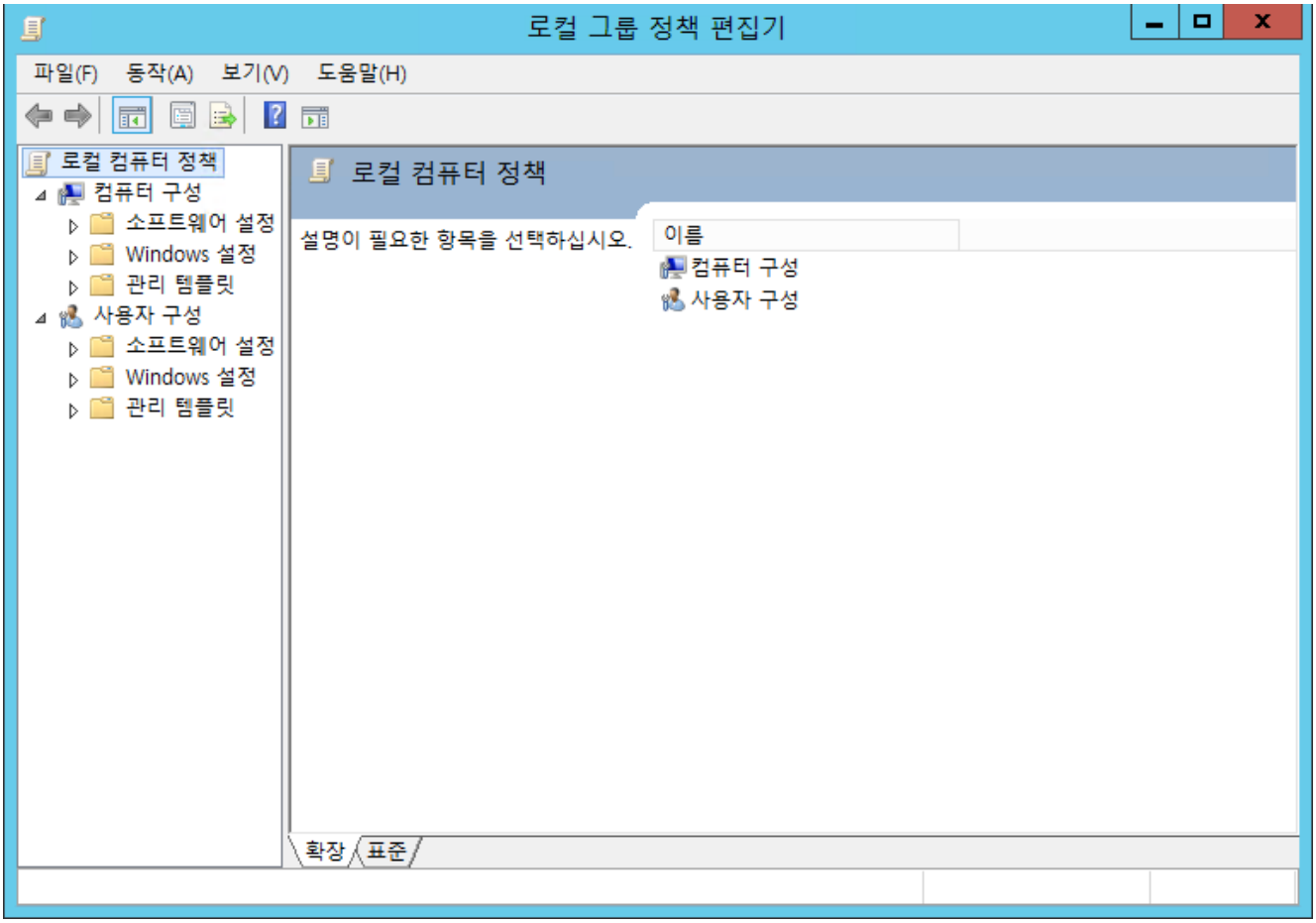
이에 Windows Server 2012에서 세션 설정을 변경하는 방법을 안내드리도록 하겠습니다.

Windows Server 2012의 경우 세션 설정에 대하여 로컬 컴퓨터 정책에서 설정해야 합니다.

실행창에서 gpedit.msc 를 입력하여 실행합니다.

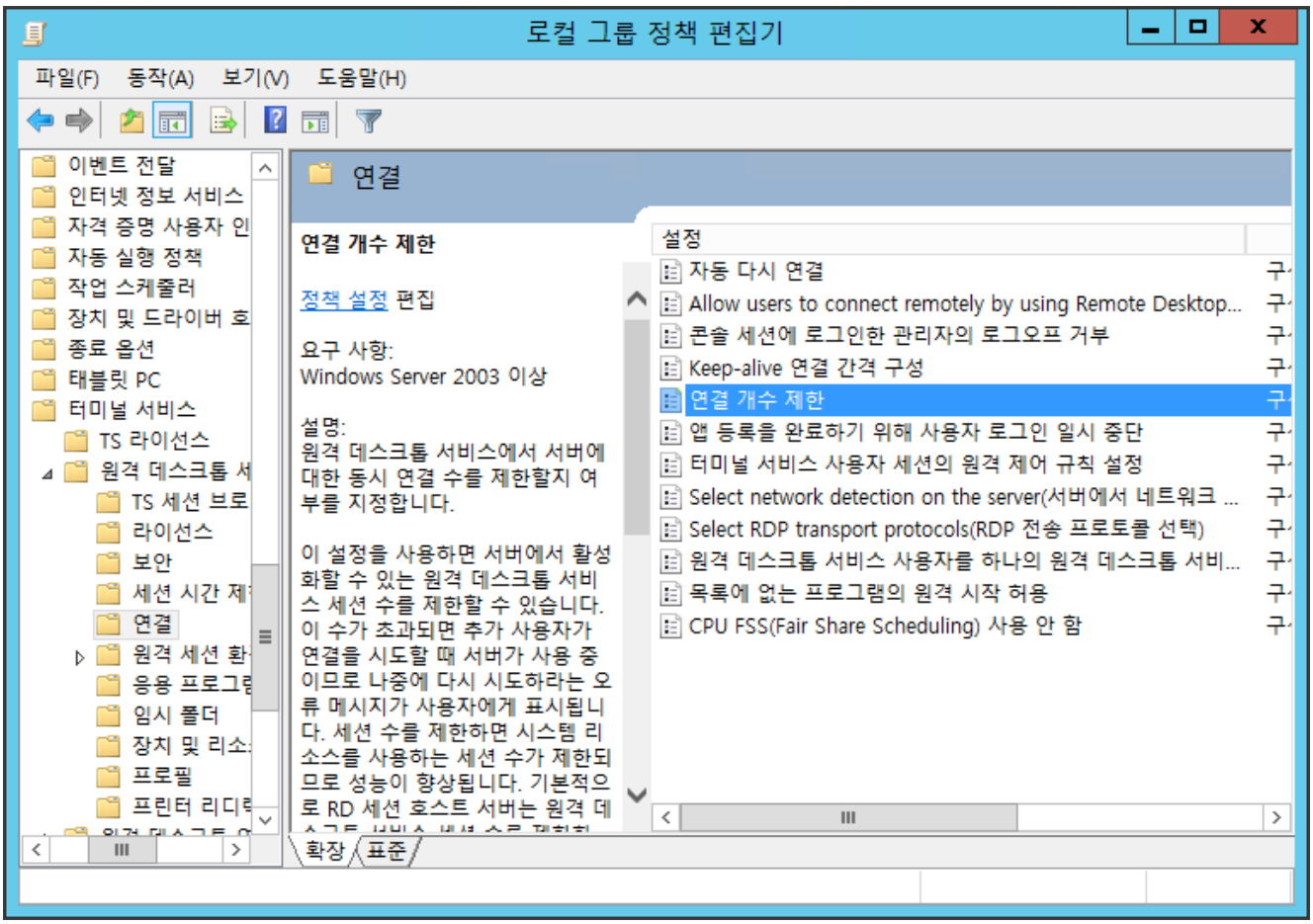


로컬 그룹 정책 편집기가 나타납니다.

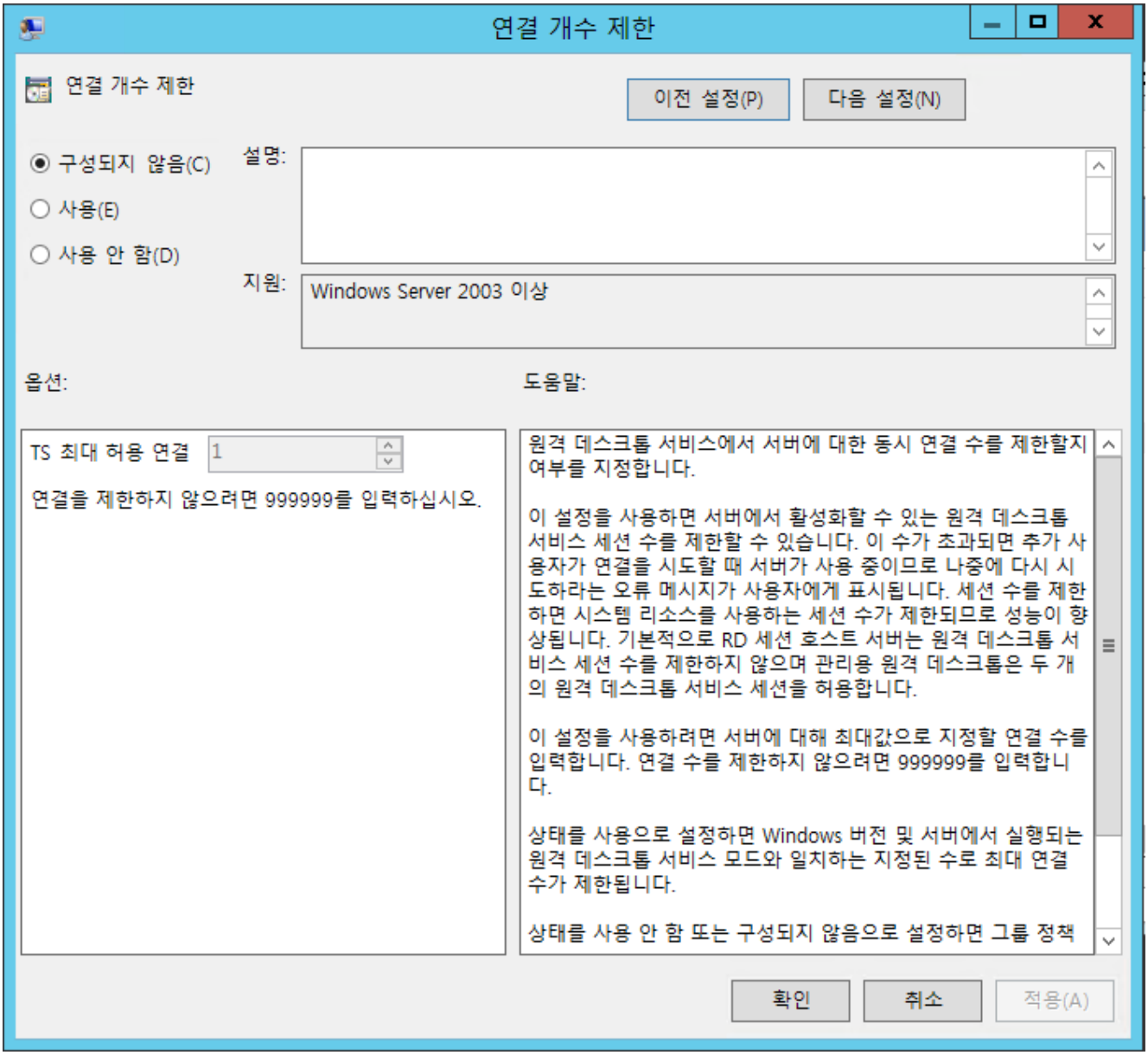


원격 세션 연결 설정 변경을 위하여 [컴퓨터 구성] – [관리 템플릿] - [Windows 구성 요소] – [터미널 서비스] – [원격 데스크톱 세션 호스트] – [연결] 메뉴를 선택 한후

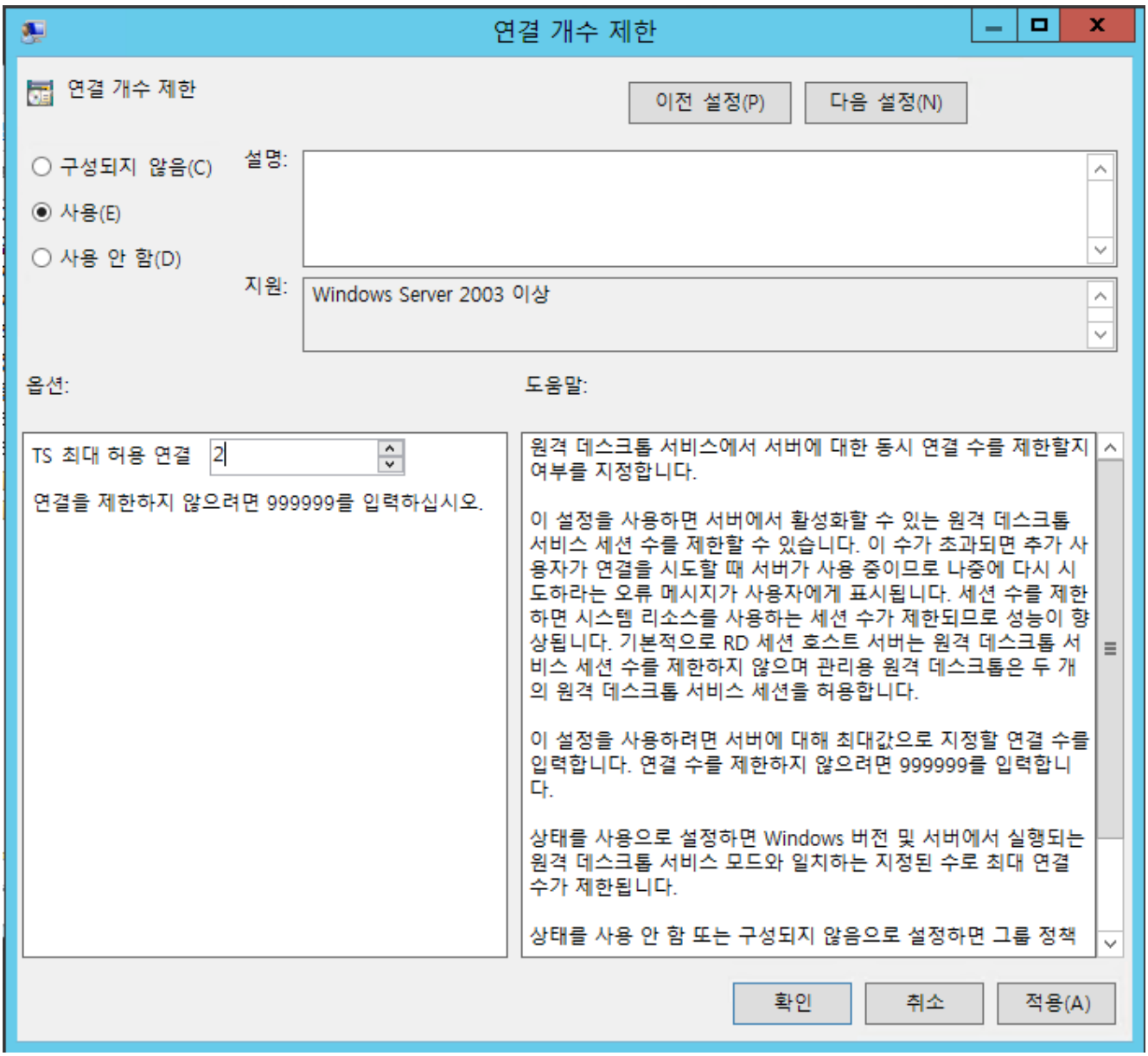
[연결 개수 제한]을 항목을 선택합니다.



기본적인 [연결 개수 제한]은 구성되지 않음으로 설정되어 있습니다.



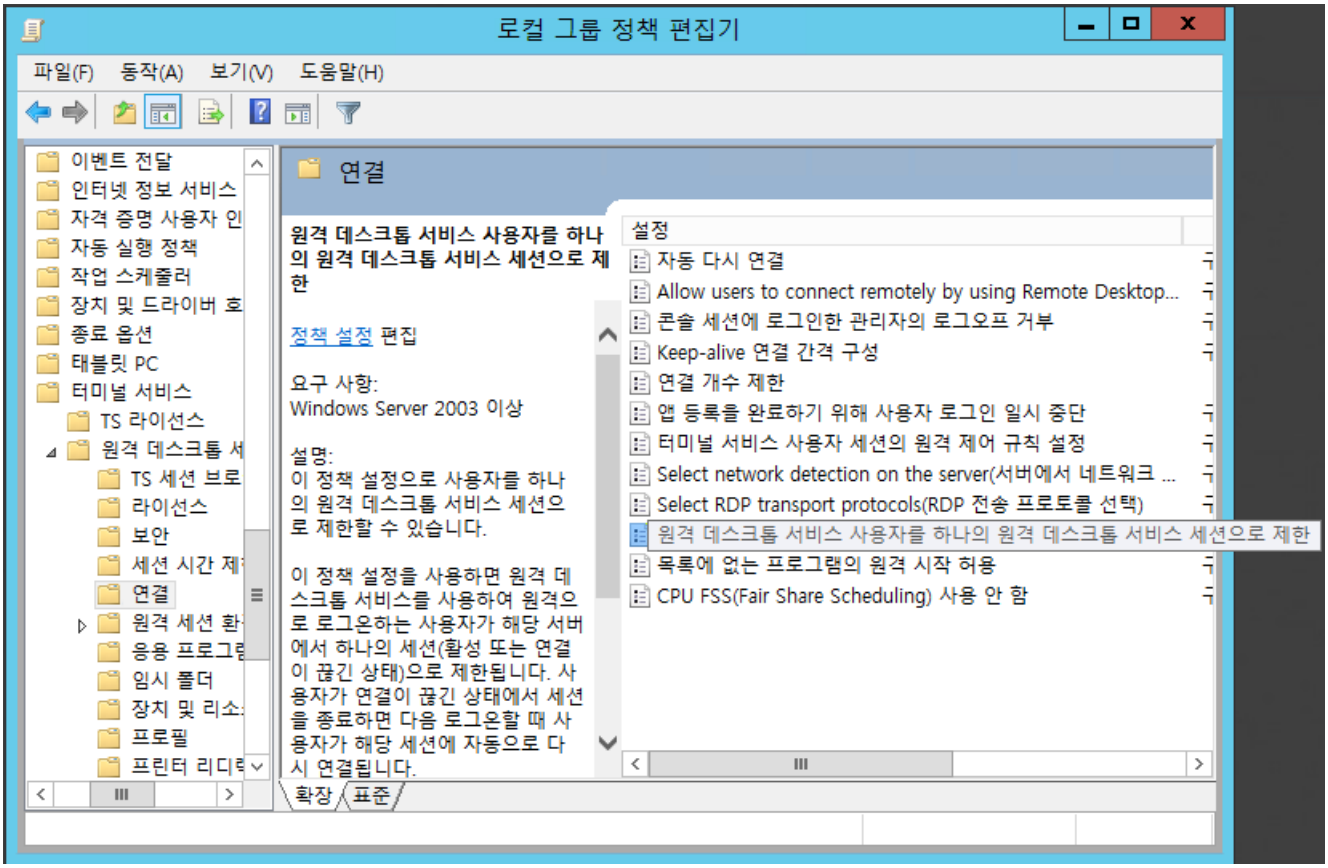
[연결 개수 제한]을 사용으로 선택하며, 옵션에서 TS 최대 허용 연결을 임의의 값으로 설정한 후 확인을 클릭 합니다.(여기서는 2개 선택)



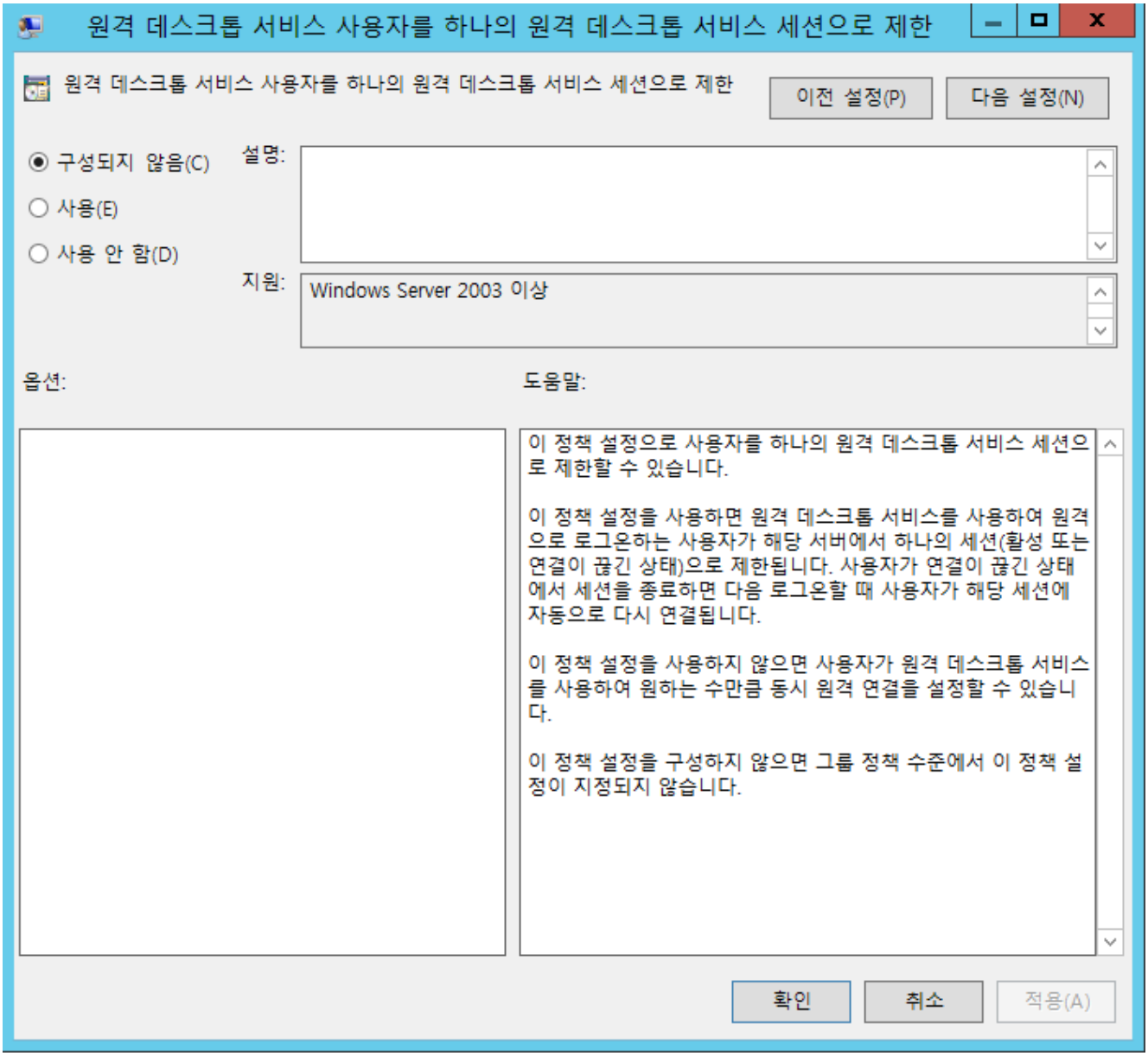
여기 까지의 설정으로 세션 연결은 2개로 늘어났지만, 아직까지는 하나의 세션으로 접속이 됩니다.

이에 각자 별도의 세션으로 접속하기 위한 작업을 다음과 같이 추가적으로 적용해 줍니다.

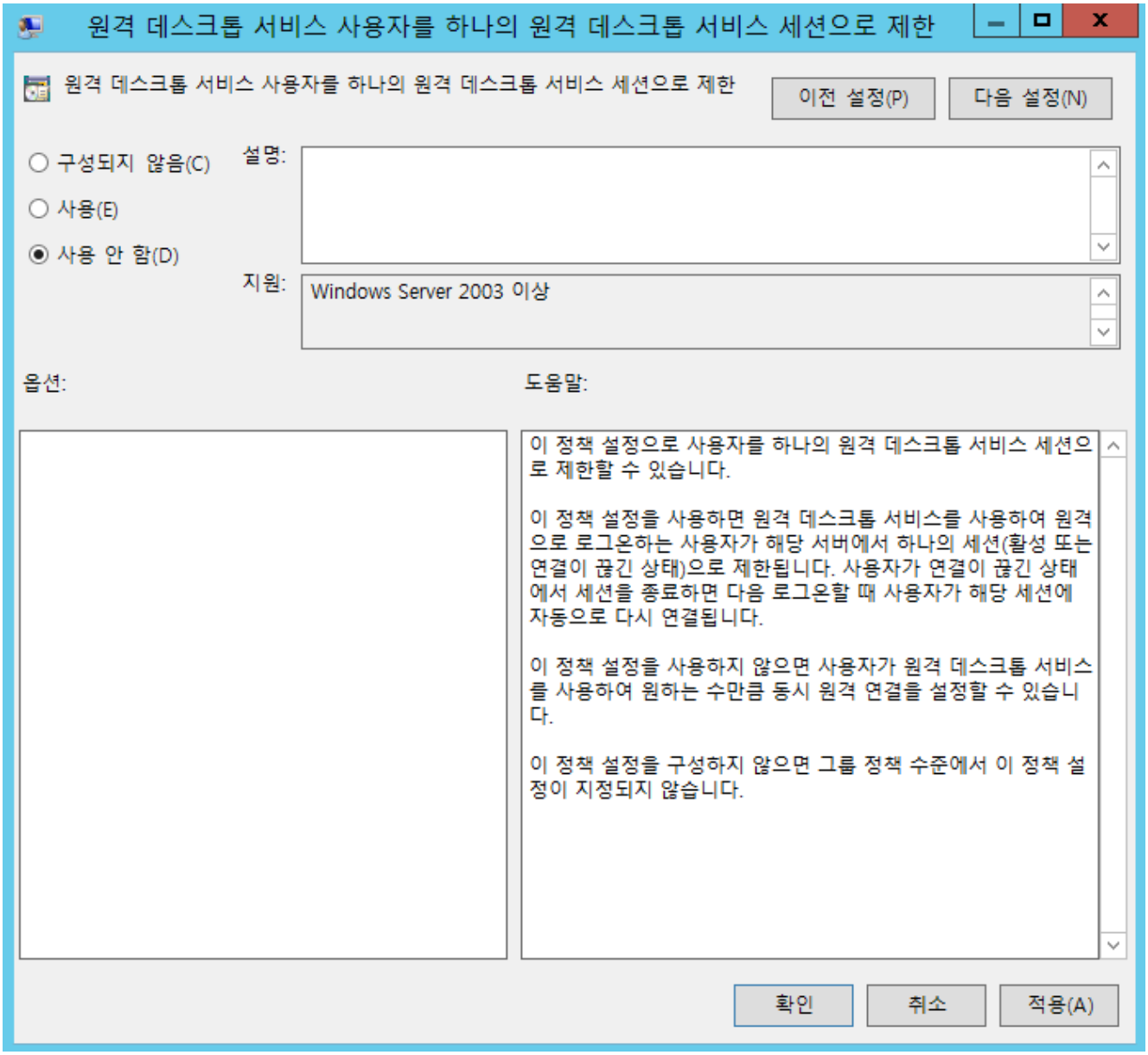
[원격 데스크톱 서비스 사용자를 하나의 원격 데스크톱 서비스 세션으로 제한] 을 선택합니다.



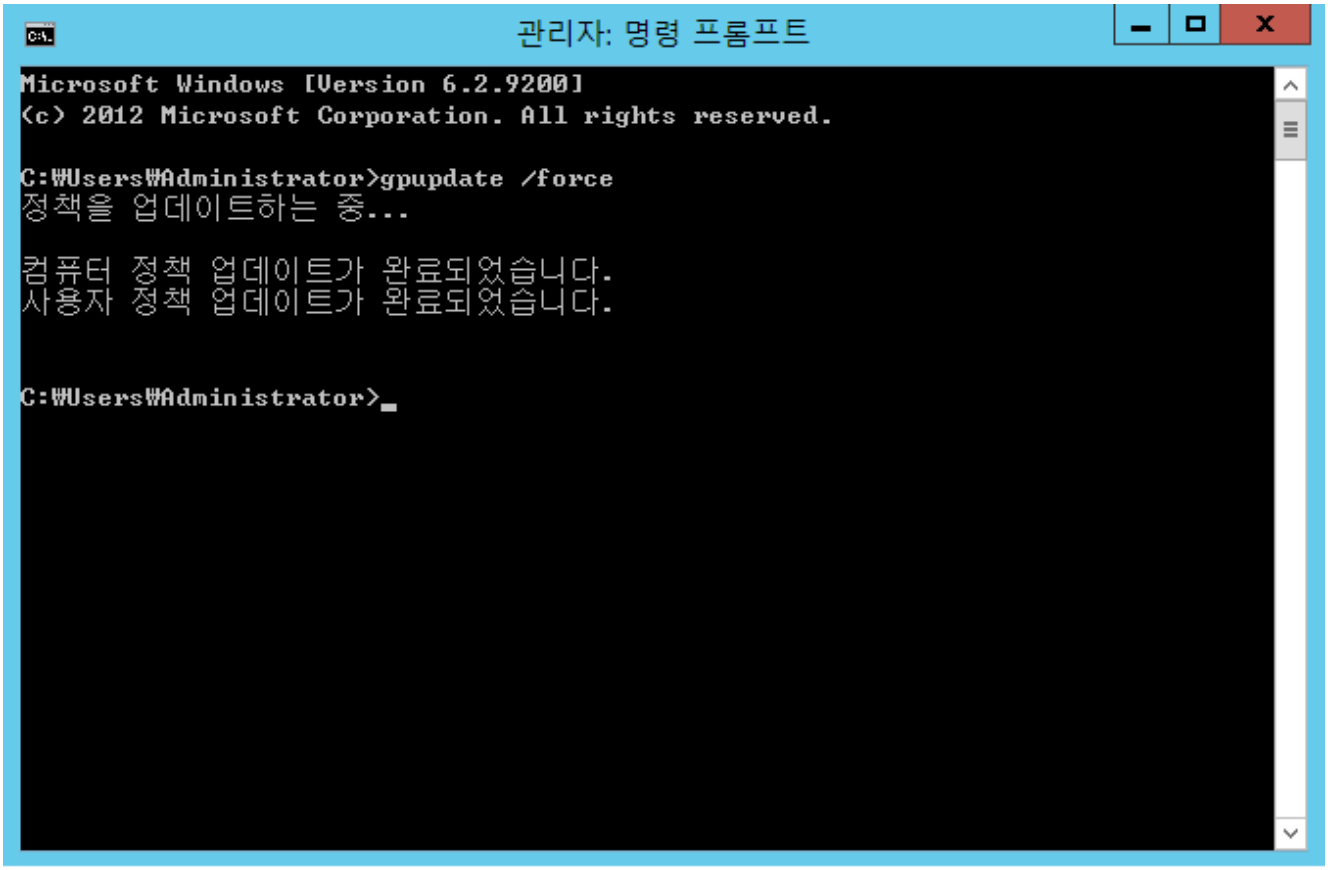
기본적인 메뉴는 다음과 같습니다.



각각의 세션에서 작업을 하기 위하여 사용 안 함 으로 선택 후 확인을 클릭합니다.



변경된 정책이 적용 될 수 있도록 정책 업데이트를 적용 해줍니다.



CentOS 7 설치 후 할 일 – 파티셔닝 및 설정 (1)

CentOS 7 설치 후 할 일

실제로 서비스를 제공하는 서버라면 몇 가지 기본적인 설정, 패키지, 보안에 신경을 써줘야 한다.

파티셔닝 및 설정, 필수 및 권장 패키지 설치, 서버 운영을 위한 보안 설정(하드닝) 세가지로 분류해 작성한다.

파티셔닝

LVM : Default로 설정되어 있으며 스토리지 확장 시 유연하게 변경 가능, 관리 난이도가 높아 일반적으로 **표준 파티션** 사용 권장

표준 파티션 : 디스크 성능이 좋아짐에 따라 속도 면에서 **LVM**과 크게 차이가 없음, xfs 형식 사용하여 이전보다 파티션 조절 수월함

1. 네트워크 설정

(OS 설치 과정에서 네트워크 기설정되어 있는 경우 해당 내용 생략)

```
# ip addr
# vi /etc/sysconfig/network-scripts/ifcfg-eth0

. . .
BOOTPROTO=none
. . .
IPV6INIT=no
IPV6_AUTOCONF=no
IPV6_DEFROUTE=no
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
. . .
ONBOOT=yes                                  # 부팅/네트워크 재시작 시  자동으로 인터페이스 활성화를 위해  yes  설정
IPV6_PRIVACY=no
IPADDR=192.168.122.243
NETMASK=255.255.255.0
GATEWAY=192.168.122.1
DNS1=8.8.8.8
DNS2=8.8.4.4

# systemctl restart network
# ip addr
eth0 영역  IP 확인



# ping -c 4 google.com                  # 네트워크 상태 확인
--- google.com ping statistics ---
```

```
4 packets transmitted, 4 received, 0% packet loss, time 3094ms
rtt min/avg/max/mdev = 82.692/83.010/83.554/0.492 ms
```

2. 바이오스 타이머 조정

CentOS 7 부터 timedatectl 명령어로 시간, 타임존 설정을 할 수 있다.

```
# timedatectl

      RTC time : 하드웨어 시간
      NTP enabled : NTP 활성화 여부
      NTP synchronized : NTP 동기화 여부
      RTC in local TZ : RTC 와 Time zone 시간 동기화 여부

# timedatectl list-timezones | grep -i Asia*

# timedatectl set-timezone Asia/Seoul

# timedatectl
```

3. Hostname 설정

CentOS 7 설치 후 기본적으로 hostname이 localhost.localdomain 으로 설정되어있으므로 변경해주어야 함

```
# hostnamectl
Static hostname: localhost.localdomain

# hostnamectl set-hostname newhostname

# hostnamectl
Static hostname: newhostname
```

4. SELinux 비활성화

SELinux는 기본적으로 활성화 되어있으나, 해당 메뉴얼에서는 disabled(=미사용) 설정한다.

```
# vi /etc/sysconfig/selinux

. . .
SELINUX=disabled     . . .

# shutdown -r now

# getenforce
Disabled
```

5. root 계정 원격 접근 불가 설정

보안을 위해 root로의 원격 접속은 거부하고 타 계정을 사용하여 su 명령어를 통해 사용하도록 한다.

```
# ps -ef | grep sshd
# systemctl enable sshd

# vi /etc/ssh/sshd_config
. . .
PermitRootLogin=no
. . .

# systemctl restart sshd
```

6. 자동 로그아웃 설정

보안을 위해 일정 시간 응답이 없으면 자동으로 로그아웃 하도록 설정한다.

```
# vi /etc/profile.d/timeout.sh
TMOUT=600
export TMOUT
```

```
chmod +x /etc/profile.d/timeout.sh

# source /etc/profile
# echo $TMOUT
600
```

7. 히스토리 포맷 설정

명령어의 내역을 출력하는 명령어인 history의 날짜 및 시간을 확인하기 위해 설정한다.

```
# vi /etc/profile.d/history.sh

HISTTIMEFORMAT="%F %T -- "
export HISTTIMEFORMAT


# chmod 644 /etc/profile.d/history.sh
# source /etc/profile.d/history.sh

# hisotry
999  2022-04-06 14:50:10 -- vi /etc/profile.d/history.sh
1000       2022-04-06    14:50:19    --    chmod    644
/etc/profile.d/history.sh
1001  2022-04-06 14:50:28 -- source /etc/profile.d/history.sh
1002  2022-04-06 14:50:30 -- history
```

8. 문자셋 설정

설치시 언어와 문자셋을 잘못 선택했을 경우에 설정한다.

```
# localectl
  System Locale: LANG=en_US.UTF-8
      VC Keymap: us
     X11 Layout: us
```

```
# localectl list-locales | grep -i kr
ko_KR
ko_KR.euckr
ko_KR.utf8

# localectl set-locale LANG=ko_KR.UTF-8
# localectl set-keymap kr
# localectl set-x11-keymap kr

# localectl
   System Locale: LANG=ko_KR.UTF-8
       VC Keymap: kr
      X11 Layout: kr
```

다음 장에선 필수적으로 설치해야할 패키지 및 권장 패키지들에 대해서 설명한다.