

BaroPAM 가이드 (RADIUS)

목차

목차.....	0
1. RADIUS 인증.....	1
1.1 개요.....	1
1.2 구성 아키텍처.....	1
1.3 구성 요소.....	2
1.4 방화벽 요구사항.....	2
2. FreeRADIUS 설치 및 설정.....	3
2.1 FreeRADIUS 설치.....	3
2.2 FreeRADIUS 설정.....	3
3. BaroPAM 설치 및 설정.....	11
3.1 BaroPAM 설치 전 준비사항.....	11
3.2 BaroPAM 설치 모듈 다운로드.....	12
3.3 BaroPAM 환경 설정 파일 생성.....	13
3.4 BaroPAM 환경 설정.....	17
4. MySQL/MariaDB 설치 및 구성.....	25
4.1 MariaDB 설치.....	25
4.2 MariaDB 구성.....	26
5. FreeRADIUS 연동 테스트.....	37
5.1 PAM, cURL 인증.....	37
5.2 SQL 인증.....	41
6. About BaroPAM.....	46

1. RADIUS 인증

1.1 개요

RADIUS(Remote Authentication Dial In User Service, 원격 인증 전화 사용자 서비스 위치)는 네트워크 프로토콜로 사용자가 네트워크에 연결하고 네트워크 서비스를 받기 위한 중앙 집중화된 인증, 인가, 회계(AAA, 회계 Accounting은 인증, 인가 후 각종 사후 처리를 맡는다.) 관리를 제공한다. RADIUS는 1991년 서버 접근 인증, 회계 프로토콜로써 Livingston Enterprises, Inc. 에서 개발했다. 그리고 후에 IETF표준으로 등재되었다.

지원 범위가 넓고 유비쿼터스 환경에서도 사용이 가능하기에 ISP와 기업들이 인터넷이나 인트라넷 접근을 관리하거나 무선 네트워크 인증, 통합 메일 서비스 등에 자주 쓰인다. 모뎀, 무선 액세스 포인트, 디지털 가입자 회선, 가상 사설망, TCP 및 UDP 포트, 웹 서버 등에 사용된다.

RADIUS는 응용 계층에서 작동하는 클라이언트 및 서버 프로토콜이며 사용자 데이터그램 프로토콜을 통해서 전송된다. 원격 접속 서버, 가상 사설망, 네트워크 스위치의 포트 인증, 네트워크 액세스 서버(NAS) 이들 모두는 RADIUS서버와 통신하는 컴포넌트를 가진다. RADIUS는 종종 IEEE 802.1X 인증의 기반이 되기도 한다.

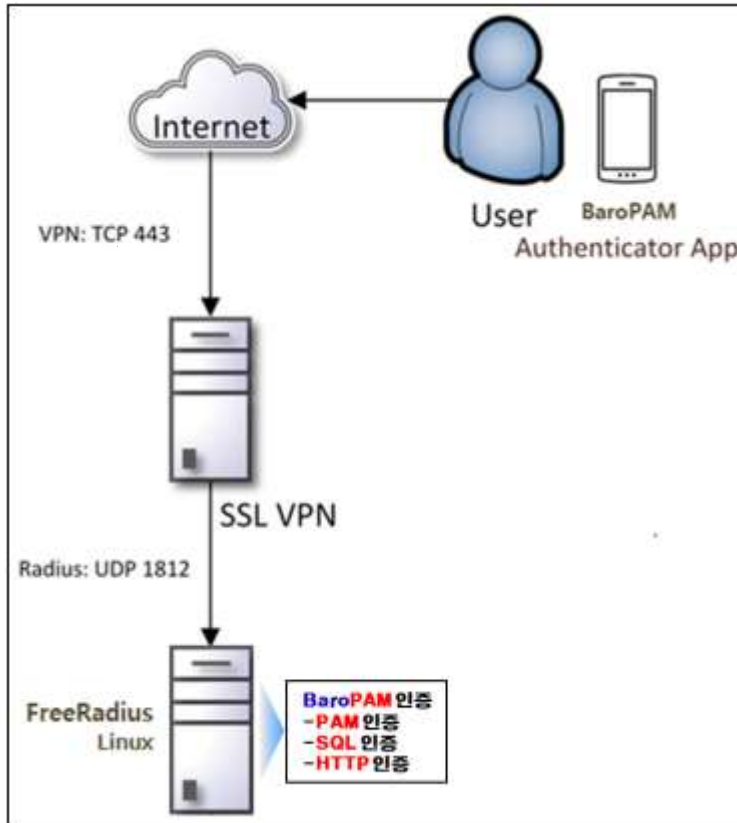
RADIUS 서버는 리눅스/유닉스 시스템이나 윈도우 서버에서 자주 백그라운드 프로세스로 실행된다.

RADIUS 인증 과정인 프로세스 흐름은 비교적 간단하지만 이해하는 것이 중요하다.

1. 어플리케이션 클라이언트는 어플리케이션 서버에 연결을 시도하고 **일회용 인증키**와 함께 자격 증명(로그인-ID와 비밀번호)을 제공한다.
2. 어플리케이션 서버가 이 정보를 수신하고 로컬 카탈로그에서 사용자를 찾고 인증 유형이 PAM인지 확인한다.
3. 그런 다음 외부 사용자를 인증할 위치를 결정한다.
4. RADIUS 연결 정보를 사용하여 어플리케이션 서버는 자격 증명 세부 정보를 RADIUS 서버로 전달한다.
5. RADIUS 서버는 먼저 PAM, HTTP, SQL, Active Directory, LDAP 서비스 등이 될 수 있는 PAM를 사용하여 로그인-ID / 비밀번호로 "**기본 인증(Primary Authentication)**"을 한다.
6. 유효성이 확인되면 RADIUS 서버는 **2차 인증** 서비스인 BaroPAM 모듈로 **일회용 인증키**로 "**2차 인증(Secondary Authentication)**"을 한다.
7. 유효성이 확인된 경우 RADIUS 서버는 "**Access-Accept**" 응답을 어플리케이션 서버로 다시 전달한 다음 연결을 수락하고 완료한다.

일관되고 정확한 시간은 제안된 솔루션의 작동을 위한 핵심 요구 사항이다. RADIUS 호스트와 시간 기반 **일회용 인증키(One-Time Authentication key)**를 사용하는 경우 BaroPAM 서비스와 BaroPAM 앱이 설치된 기기는 일관된 시간을 가져야 하기 때문에 NTP(Network Time Protocol)가 설정되어 있어야 한다.

1.2 구성 아키텍처



1.3 구성 요소

- Rocky linux
- FreeRADIUS
- BaroPAM** PAM Library, Service, & APP
- Pluggable Authentication Module (PAM)

1.4 방화벽 요구사항

- 1) User → NAS
 - TCP 443: SSL VPN
- 2) NAS → RADIUS
 - UDP 1812: RADIUS

2. FreeRADIUS 설치 및 설정

2.1 FreeRADIUS 설치

```
[root@localhost ~]# dnf -y install freeradius freeradius-utils
```

설치된 FreeRADIUS를 제거 하려고 할 때 → `dnf -y erase freeradius freeradius-utils`

만약, MySQL/MariaDB를 연동하는 경우에는 MySQL/MariaDB 모듈을 추가로 설치해야 한다.

```
[root@localhost ~]# dnf -y install freeradius-mysql
[root@localhost ~]# dnf -y install freeradius-mysql --enablerepo=devel → Rocky 9.x
```

FreeRADIUS를 설치 후에는 반드시 EAP에 대한 인증서를 생성해야 한다.

```
$ cd /etc/raddb/certs
$ ./bootstrap
```

EAP에 대한 인증서를 생성하지 않으면 다음과 같은 오류가 발생한다.

```
Failed reading private key file /etc/raddb/certs/server.pem
:error:06065064:digital envelope routines:EVP_DecryptFinal_ex:bad decrypt
rlm_eap_tls: Failed initializing SSL context
rlm_eap (EAP): Failed to initialise rlm_eap_tls
/etc/raddb/mods-enabled/eap[17]: Instantiation failed for module "eap"
```

2.2 FreeRADIUS 설정

1) PAM, cURL 인증

사용자와 그룹을 모두 `root`로 다음과 같이 변경한다.

```
[root@localhost ~]# vi /etc/raddb/radiusd.conf
#user = radiusd
#group = radiusd
user = root
group = root
```

FreeRADIUS 사용은 사용자의 홈 디렉토리에 있는 [BaroPAM](#)에 액세스하려면 `root`로 실행해야 한다.

보다 쉬운 문제 해결을 위해 `log` 지시문에서 `/etc/raddb/radiusd.conf`를 편집하여 추가 로깅을 활성화한다.

```
auth = yes
auth_badpass = yes
auth_goodpass = yes
```

PAM을 활성화하려면 `/etc/raddb/sites-enabled/default`를 편집해야 한다. PAM이 활성화되도록 `#PAM` 라인의

주석을 해제한다.

```
[root@localhost ~]# vi /etc/raddb/sites-enabled/default

#Pluggable Authentication Modules.
pam
```

FreeRADIUS 로그인 "auth_log"와 "reply_log"을 활성화하기 위하여 주석(#) 처리를 제거한다.

```
[root@localhost ~]# vi /etc/raddb/sites-enabled/default

#
# If you want to have a log of authentication requests,
# un-comment the following line, and the 'detail auth_log'
# section, above.
#
auth_log

#
# If you want to have a log of authentication replies,
# un-comment the following line, and the 'detail reply_log'
# section, above.
#
reply_log
```

다음과 같이 PAM을 활성화하기 위하여 /etc/raddb/mods-enabled/에서 PAM용 소프트 링크로 생성한다.

```
[root@localhost ~]# ln -s /etc/raddb/mods-available/pam /etc/raddb/mods-enabled/
```

Ipv4addr과 secret를 다음과 같이 변경한다.

```
[root@localhost ~]# vi /etc/raddb/clients.conf

client localhost {
    ipaddr = 127.0.0.1
    ipv4addr = * # any. 127.0.0.1 == localhost
    secret = baropam
    require_message_authenticator = no
    nas_type = other
}
```

인증 유형을 PAM으로 다음과 같이 변경한다.

```
[root@localhost ~]# vi /etc/raddb/users

DEFAULT Group = "disabled", Auth-Type := Reject
Reply-Message = "Your account has been disabled."
DEFAULT Auth-Type := PAM
```

2) SQL 인증

다음과 같이 SQL을 활성화하기 위하여 /etc/raddb/mods-enabled/에서 SQL용 소프트 링크로 생성한다.

```
[root@localhost ~]# ln -s /etc/raddb/mods-available/sql /etc/raddb/mods-enabled/
```

ipv4addr과 secret를 다음과 같이 변경한다.

```
[root@localhost ~]# vi /etc/raddb/clients.conf

client localhost {
    ipaddr = 127.0.0.1
    ipv4addr = * # any. 127.0.0.1 == localhost
    secret = baropam
    require_message_authenticator = no
    nas_type = other
}
```

이제 MySQL/MariaDB를 사용하도록 FreeRADIUS를 구성한다. /etc/raddb/mods-available/sql 파일을 다음과 같이 편집한다.

```
driver = "rlm_sql_null" => driver = "rlm_sql_mysql"
dialect = "sqlite"      => dialect = "mysql"
```

MySQL/MariaDB와 함께 작동하도록 FreeRADIUS를 구성할 때 MySQL 구성은 기본적으로 TLS를 사용한다고 가정한다. 여기에서는 SSL 인증서를 사용하지 않으므로 TLS 섹션을 주석 처리한다.

```
mysql {
# If any of the files below are set, TLS encryption is enabled
# tls {
# ca_file = "/etc/ssl/certs/my_ca.crt"
# ca_path = "/etc/ssl/certs/"
# certificate_file = "/etc/ssl/certs/private/client.crt"
# private_key_file = "/etc/ssl/certs/private/client.key"
# cipher = "DHE-RSA-AES256-SHA:AES128-SHA"
# tls_required = yes
# tls_check_cert = no
# tls_check_cert_cn = no
#}
# If yes, (or auto and libmysqlclient reports warnings are
# available), will retrieve and log additional warnings from
# the server if an error has occurred. Defaults to 'auto'
warnings = auto
```

MySQL/MariaDB의 연결 정보를 설정하기 위하여 서버, 포트, 로그인 및 비밀번호의 주석을 해제하고 일부 값도 변경한다.

```
# Connection info:
#
server = "localhost"
port = 3306
login = "radius"
password = "baropam"
# Database table configuration for everything except Oracle
radius_db = "radius"
```

```
read_clients = yes
```

편집한 /etc/raddb/mods-available/sql 파일의 그룹 권한을 radiusd로 변경한다.

```
chgrp -h radiusd /etc/raddb/mods-enabled/sql
```

3) 방화벽 허용

RADIUS 패킷을 허용하도록 Firewalld를 구성해야 한다.

RADIUS는 인증에 포트 1812를 사용하고 계정에 포트 1813을 사용하므로 이러한 포트에서 트래픽을 허용해야 한다. FreeRADIUS를 설치하면 방화벽에 구성이 추가되므로 RADIUS 트래픽을 허용하고 다음 명령을 실행한다.

```
[root@localhost ~]# firewall-cmd --permanent --zone=public --add-port=1812/udp
success
[root@localhost ~]# firewall-cmd --permanent --zone=public --add-port=1813/udp
success
```

또는

```
[root@localhost ~]# firewall-cmd --add-service=radius --permanent
success
```

변경 사항을 적용하려면 방화벽을 다시 로드하기 위하여 해야 다음 명령을 실행한다.

```
[root@localhost ~]# firewall-cmd --reload
success
```

참고) 방화벽 firewall-cmd 설치 실행

```
[root@localhost ~]# firewall-cmd
usage: see firewall-cmd man page
No option specified.
```

```
[root@localhost ~]# firewall-cmd --zone=public --permanent --add-port=21/tcp
Firewalld is not running
```

1. 작동여부 확인

1-1. 방법

```
[root@localhost ~]# systemctl status firewalld
* firewalld.service - firewalld - dynamic firewall daemon
  Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)
  Active: inactive (dead)
  Docs: man:firewalld(1)
```

1-2. 방법

```
[root@localhost ~]# firewall-cmd --state
not running
```

2. firewalld 설치확인

```
[root@localhost ~]# yum list installed firewalld
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
* base: ftp.daumkakao.com
* epel: ftp.riken.jp
* extras: ftp.daumkakao.com
* updates: ftp.daumkakao.com
* webtatic: sp.repo.webtatic.com
Installed Packages
firewalld.noarch                0.4.4.4-6.el7                @base
```

3. 패키지 삭제

```
[root@localhost ~]# yum remove firewalld.noarch
```

4. firewall 설치

```
[root@localhost ~]# yum install firewalld
```

5. firewall 실행

```
[root@localhost my.cnf.d]# systemctl start firewalld
```

6. firewall 상태 확인

```
[root@localhost my.cnf.d]# systemctl status firewalld
* firewalld.service - firewalld - dynamic firewall daemon
  Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
  Active: active (running) since Tue 2018-01-30 17:12:19 KST; 4s ago
  Docs: man:firewalld(1)
  Main PID: 21620 (firewalld)
  CGroup: /system.slice/firewalld.service
          └─21620 /usr/bin/python -Es /usr/sbin/firewalld --nofork --nopid
```

4) 자동 시작을 위한 RADIUS 서비스를 생성

자동 시작을 위한 RADIUS 서비스를 생성한다.

```
[root@localhost ~]# systemctl enable radiusd.service
Created symlink /etc/systemd/system/multi-user.target.wants/radiusd.service ->
/usr/lib/systemd/system/radiusd.service.
```

5) Authentication server 설정

예) OpenVPN의 Authentication: RADIUS 설정 화면

RADIUS Server
Specify the RADIUS server connection details below.

Hostname or IP Address	Shared Secret	Authentication Port	Accounting Port
10.190.140.63	*****	1812	1813
		1812	1813
		1812	1813
		1812	1813
		1812	1813

* Enable RADIUS authentication: Yes, Authentication Method: PAP, Shared Secret: baropam (PAP, Ppassword authentication protocol password Authentication Protocol)

예) Sophos SSLVPN의 Authentication server 설정 화면

Edit external server

Feedback | How-to guides | Log viewer | Help | admin | Ebyyn

Servers | Services | Groups | Users | One-time password | Web authentication | Guest users | Clientless users | STAS | ***

Server type: RADIUS server

Server name *: SF_RADIUS

Server IP *: 10.180.39.67

Authentication port *: 1812

Time-out *: 3

Enable accounting

Accounting port: 1813

Shared secret *: ***** [Change Shared secret](#)

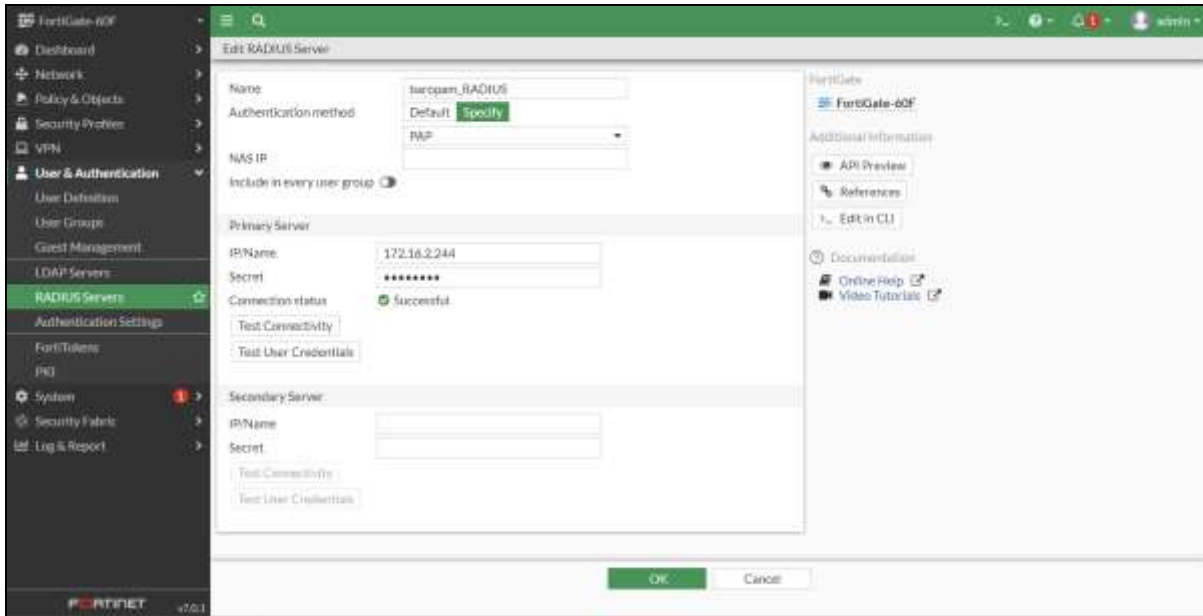
Domain name: radius-ess

Group name attribute *: RADIUS-GROUP

Enable additional settings

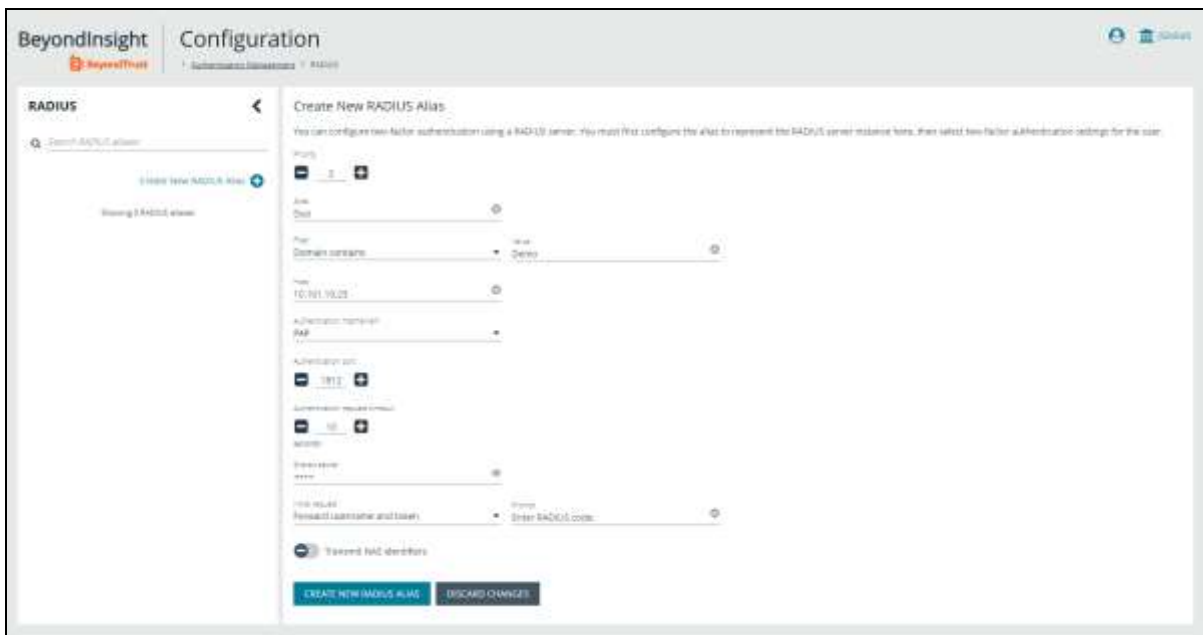
* Shared Secret: baropam

예) FortiGate SSLVPN의 Authentication server 설정 화면



* Authentication Mechanism: PAP 선택, Secret: baropam

예) Beyondtrust Password Safe의 Authentication server 설정 화면



* Authentication Mechanism: PAP, Initial Request: Forward User Name and token 선택, Secret: baropam

예) Cisco SSLVPN Authentication: RADIUS settings 화면

Radius Settings Help

Settings | Radius Users | Test

Global RADIUS Settings

- RADIUS Server Timeout: (seconds) (Range:1-60, Default: 3)
- Retries: (Range:0-10, Default:2)

Radius Servers

Radius Servers:

Primary Server

- IP Address:
- Shared Secret: (Length: 1 to 64 characters)
- Port Number: (Range:1-65535, Default:1812)

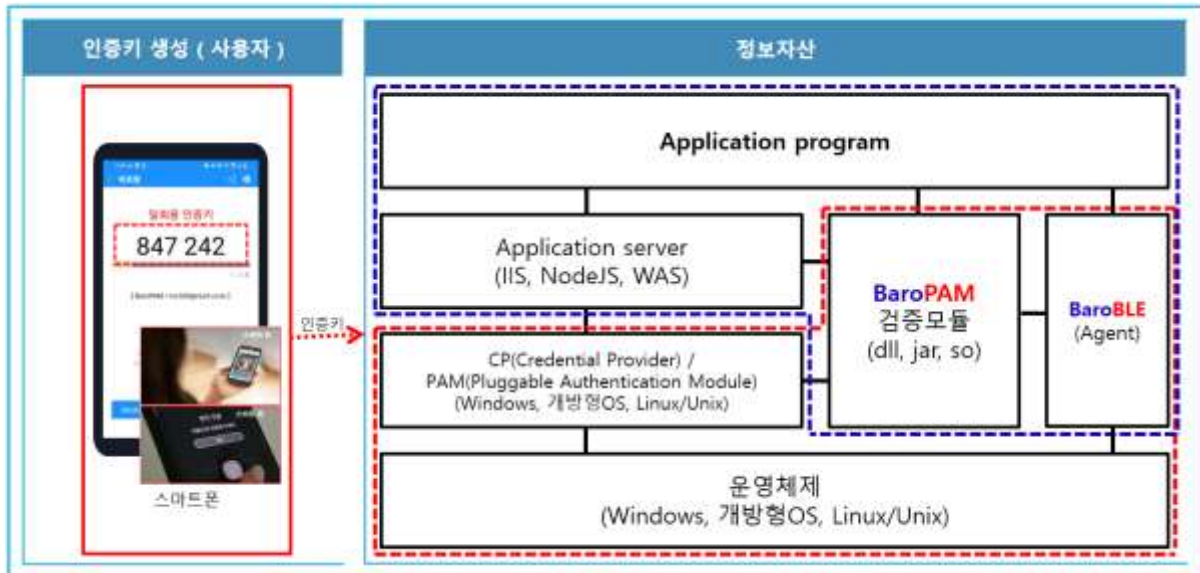
Secondary Server

- IP Address:
- Shared Secret: (Length: 1 to 64 characters)
- Port Number: (Range:1-65535, Default:1812)

* Shared Secret : baropam

3. BaroPAM 설치 및 설정

BaroPAM 솔루션은 **제로 트러스트(Zero Trust) 보안 모델**로 정보자산의 보안 강화를 위하여 **2차 인증(추가 인증)**이 필요한 다양한 운영체제와 애플리케이션에 누구나 손쉽게 곧바로 적용할 수 있는 **플러그인 가능한 인증 모듈(PAM, Pluggable Authentication Module) 방식**을 기반으로 하는 보안에 최적화된 **생체인식이 적용된 3단계 인증 솔루션**이다.



3.1 BaroPAM 설치 전 준비사항

PAM 모듈을 사용하기 위해서는 기본적으로 PAM 패키지가 반드시 설치되어 있어야 한다. 설치 확인은 다음의 명령어를 실행하여 확인한다. 만약, 설치되어 있지 않으면 Redhat 계열은 "dnf -y install *pam*" 그외는 "sudo apt-get install pam" 명령어로 설치하면 된다.

```
[root]# rpm -qa | grep pam
pam_smb-1.1.7-7.2.1
pam_passwdqc-1.0.2-1.2.2
pam-0.99.6.2-14.el5_11
pam_krb5-2.2.14-22.el5
pam-devel-0.99.6.2-14.el5_11
pam_ccreds-3-5
pam_smb-1.1.7-7.2.1
pam_pkcs11-0.5.3-26.el5
pam-devel-0.99.6.2-14.el5_11
pam_passwdqc-1.0.2-1.2.2
pam-0.99.6.2-14.el5_11
pam_ccreds-3-5
pam_krb5-2.2.14-22.el5
pam_pkcs11-0.5.3-26.el5
```

Redhat 계열인 경우 "Selinux"는 "Security Enhanced Linux"의 약자로 기본의 리눅스보다 더욱 뛰어난 보안정책을 제공하는데, 너무 뛰어난 나머지 활성화 되어 있을 경우 보안문제로 막혀서 BaroPAM이 안되는 부분이 발생(Failed to open tmp secret file "/usr/baropam/.baro_auth~" [Permission denied])한다. 그래

서 웬만하면 대부분이 비활성화(SELinux=enforcing → disabled)한다.

```
[root] /etc > vi /etc/sysconfig/selinux
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - SELinux is fully disabled.
SELINUX=disabled
# SELINUXTYPE= type of policy in use. Possible values are:
#     targeted - Only targeted network daemons are protected.
#     strict - Full SELinux protection.
SELINUXTYPE=targeted

# SETLOCALDEFS= Check local definition changes
SETLOCALDEFS=0
```

바로 적용은 되지 않으며 재부팅을 해야 적용이 된다.

재부팅을 하지 않고 현재 접속된 터미널에 한해 변경된 내용을 적용하고 싶을 경우 다음의 명령어를 실행하면 된다.

```
[root] /etc > /usr/sbin/setenforce 0
```

BaroPAM 인증 모듈을 다운로드 및 설치하기 위해서 root 계정으로 접속한 후 모듈을 다운로드 및 설치하기 위한 디렉토리(/usr/baropam)를 다음과 같이 생성한다.

```
[root]# mkdir /usr/baropam
```

BaroPAM 모듈을 다운로드 및 설치하기 위한 디렉토리의 권한(읽기, 쓰기, 실행)을 다음과 같이 부여한다.

```
[root]# chmod 777 /usr/baropam
```

3.2 BaroPAM 설치 모듈 다운로드

BaroPAM 인증 모듈은 root 계정으로 접속한 후 모듈을 다운로드 및 설치하기 위한 디렉토리(/usr/baropam)로 이동하여 모듈을 다운로드 하는 방법은 다음과 같다.

```
[root] /usr/baropam > wget http://nuriapp.com/download/libpam_baro_auth-x.x.tar
```

BaroPAM 인증 모듈의 다운로드가 완료되면 tar 파일의 압축을 해제하는 방법은 다음과 같다.

```
[root] /usr/baropam > tar -xvf libpam_baro_auth-x.x.tar
```

BaroPAM 인증 모듈의 압축을 해제하면 baropam 디렉토리에 다음과 같은 BaroPAM 관련 모듈이 생성된다.

```
[root] /usr/baropam > ls -al
합계 180
drwxrwxrwx 7 root root 4096 8월 23 09:59 .
drwxr-xr-x 17 root root 4096 2월 10 2017 ..
```

```

-r--r--r-- 1 root root 8 3월 24 2021 .baro_acl
-r--r--r-- 1 root root 305 7월 2 14:41 .baro_auth
-r--r--r-- 1 root root 290 6월 30 12:55 .baro_curl
-r--r--r-- 1 root root 287 2월 28 12:19 .baro_sql
-rwxr-xr-x 1 root root 69149 4월 6 19:12 baro_auth
-rwxr-xr-x 1 root root 65072 6월 29 16:36 baro_curl
-rwxr-xr-x 1 root root 57074 2월 28 12:18 baro_sql
drwxr-xr-x 2 root root 4096 7월 20 2021 jilee
-rwxr-xr-x 1 root root 152649 6월 9 08:19 pam_baro_auth.so
-rwxr-xr-x 1 root root 116158 6월 30 12:54 pam_baro_curl.so
-rwxr-xr-x 1 root root 170863 2월 28 12:18 pam_baro_sql.so
-rw-r--r-- 1 root root 221 6월 27 15:59 setauth.sh
-rw-r--r-- 1 root root 150 6월 29 16:29 setcurl.sh
-rw-r--r-- 1 root root 180 2월 28 12:19 setsql.sh
    
```

3.3 BaroPAM 환경 설정 파일 생성

1) PAM 인증(.baro_auth): 환경 설정 정보를 File에 설정

BaroPAM 환경 설정 파일은 baro_auth 프로그램을 실행하여 반드시 생성하는데, BaroPAM 인증 모듈의 디렉토리인 /usr/baropam 밑에 위치하도록 한다.

형식)

```
baro_auth -r rate_limit -R rate_time -t cycle_time -k key_method -e encrypt_flag -A acl_type -a acl_filename -S secure_key -s filename
```

BaroPAM 환경설정 파일의 설정 옵션에 대한 내용은 다음과 같다.

옵션	설명	설정값	비고
-r	일회용 인증키의 제한횟수(1~10)	3	
-R	일회용 인증키의 제한시간(초, 15~600초)	30	
-t	일회용 인증키의 인증주기(초, 3~60초)	30	
-k	일회용 인증키의 인증방식(app1, app256, app384, app512).	app512	
-e	환경설정 파일의 암호화 여부(yes or no)	no	
-A	2차 인증에서 허용(allow) 또는 제외(deny)할지 선택	deny	
-a	2차 인증에서 허용(allow) 또는 제외(deny)할 계정에 대한 ACL 파일명(파일 접근권한은 444)	/usr/baropam/.baro_acl	
-S	반드시 벤더에서 제공하는 Secure key(라이선스 키)	j1q1chbVqdpj7b4PzBpM2DileBvmHFV/	
-s	BaroPAM 환경설정 파일을 생성할 디렉토리를 포함한 파일명	/usr/baropam/.baro_auth	

주의) -s 옵션의 filename는 BaroPAM 환경설정 파일을 생성할 디렉토리를 포함한 파일명(파일 접근권한은 444)이다.

사용 예)

```
[root] /usr/baropam > ./baro_auth -r 3 -R 30 -t 30 -k app512 -e no -A deny -a /usr/baropam/.baro_acl -S j1q1chbVqdpj7b4PzBpM2DileBvmHFV/ -s /usr/baropam/.baro_auth
```

만약, 계정마다 BaroPAM 환경 설정파일을 각각 설정하는 경우 해당 계정으로 접속하여 작업을 진행한다.

(Not root)

```
[root] /usr/baropam > ./baro_auth -r 3 -R 30 -t 30 -k app512 -e no -A deny -a ~/.baro_acl -S j1q1chbVqdpj7b4PzBpM2Di1eBvmHFV/ -s ~/.baro_auth
```

1) Your emergency one-time authentication keys are :

응급 일회용 인증키는 **일회용 인증키** 생성기인 **BaroPAM** 앱을 사용할 수 없을 때 분실한 경우를 대비하여 SSH 서버에 다시 액세스하는데 사용할 수 있는 접속이 가능한 Super 인증키 이므로 어딘가에 적어 두는 것이 좋다.

2) 다음에 나오는 물음에 대해서는 모두 "y"를 입력한다.

"/usr/baropam/.baro_auth" 파일을 업데이트하시겠습니까 (y/n) **y**
 중간자(man-in-the-middle) 공격을 예방할 것인가 (y/n) **y**

BaroPAM 환경 설정 파일인 **.baro_auth**에 설정한 내용은 다음과 같다.

```
[root] /usr/baropam > cat .baro_auth
" AUTH_KEY
" RATE_LIMIT 3 30
" KEY_METHOD app512
" CYCLE_TIME 30
" SECURE_KEY j1q1chbVqdpj7b4PzBpM2Di1eBvmHFV/
" ACL_NAME /usr/baropam/.baro_acl
" ACL_TYPE deny
" DISALLOW_REUSE
33458936
19035576
15364353
54649370
84342192
```

BaroPAM 환경설정 파일인 **.baro_auth**의 설정 항목에 대한 내용은 다음과 같다.

항목	설명	설정값	비고
AUTH_KEY	인증 구분자(고정)		
RATE_LIMIT	일회용 인증키 의 제한횟수(1~10), 제한시간(초, 15~600초)	3 30	
KEY_METHOD	일회용 인증키 의 인증방식(app1, app256, app384, app512)	app512	
CYCLE_TIME	일회용 인증키 의 인증주기(초, 3~60초)	30	
SECURE_KEY	반드시 벤더에서 제공하는 Secure key(라이선스 키)	j1q1chbVqdpj7b4PzBpM2Di1eBvmHFV/	
ACL_TYPE	2차 인증 에서 허용(allow) 또는 제외(deny) 구분	deny	
ACL_NAME	2차 인증 에서 허용 또는 제외할 계정에 대한 ACL Filename(파일 접근권한은 444)	/usr/baropam/.baro_acl	
DISALLOW_REUSE or ALLOW_REUSE	중간자(man-in-the-middle) 공격을 예방할 경우는 "DISALLOW_REUSE"을 설정한 경우 일회용 인증키 의 인증주기 동안은 다른 사용자가 로그인 할 수 없으며, 만약 허용할 경우는 "ALLOW_REUSE"을 설정한다.	DISALLOW_REUSE	

2) PAM 인증(.baro_sql): 환경 설정 정보를 MariaDB에 설정

BaroPAM 환경 설정 정보가 존재하는 Mariadb와 연동하기 위한 접속 정보는 baro_sql 프로그램을 실행하여 반드시 생성하는데, BaroPAM 인증 모듈의 디렉토리인 /usr/baropam 밑에 위치하도록 한다.

형식)

```
baro_sql -H hostname -u username -p password -d dbname -P portno -e encrypt_flag -s filename
```

BaroPAM 환경설정 파일의 설정 옵션에 대한 내용은 다음과 같다.

옵션	설명	설정값	비고
-H	MariaDB 서버의 호스트 이름 또는 IP 주소	nur it.co.kr	
-u	MariaDB 사용자 이름	nur it	
-p	MariaDB 사용자의 비밀번호	baropam	
-d	연결할 MariaDB 이름	baropamdb	
-P	MariaDB 서버의 포트 번호	3308	
-e	환경설정 파일의 암호화 여부(yes or no)	no	
-s	BaroPAM 환경설정 파일을 생성할 디렉토리를 포함한 파일명	/usr/baropam/.baro_sql	

주의) -s 옵션의 filename는 BaroPAM 환경설정 파일을 생성할 디렉토리를 포함한 파일명(파일 접근권한은 444)이다.

사용 예)

```
[root] /usr/baropam > ./baro_sql -H nurit.co.kr -e no -u nurit -p baropams -d baropamdb -P 3306 -s /usr/baropam/.baro_sql
```

- 1) 다음에 나오는 물음에 대해서는 모두 "y"를 입력한다.
"/usr/baropam/.baro_sql" 파일을 업데이트하시겠습니까 (y/n) y

BaroPAM 환경 설정 파일인 .baro_sql에 설정한 내용은 다음과 같다.

```
[root] /usr/baropam > cat .baro_sql
" AUTH_KEY
" HOSTNAME nurit.co.kr
" USERNAME nurit
" PASSWORD baropams
" DBNAME baropamdb
" PORTNO 3306
" RATE_LIMIT 3 30
" KEY_METHOD app512
" CYCLE_TIME 30
" SECURE_KEY j!qlcHbVqdpj7b4PzBpM2Di!eBvmHFV/
" ACL_TYPE deny
" MIDDLE_TYPE DISALLOW_REUSE
" MIDDLE_TIME 58014762
" ENV_TYPE share
```

BaroPAM 환경설정 파일인 .baro_sql의 설정 항목에 대한 내용은 다음과 같다.

항목	설명	설정값	비고
AUTH_KEY	인증 구분자(고정)		
HOSTNAME	MariaDB 서버의 호스트 이름 또는 IP 주소	nur it.co.kr	

USERNAME	MariaDB 사용자 이름	nurit	
PASSWORD	MariaDB 사용자의 비밀번호	baropam	
DBNAME	연결할 MariaDB 이름	baropamdb	
PORTNO	MariaDB 서버의 포트 번호	3308	
그외	나머지는 내부용으로 사용함.		

3) curl 인증(.baro_curl)

curl의 명칭은 "client URL"을 대표하는 것으로 1997년에 처음 출시되었다. 즉 클라이언트가 스크립트로서 서버에 데이터를 요청하는 것으로 BaroPAM은 curl로 http/https 되어 있는 인증 사이트를 호출하여 인증을 요청한다.

BaroPAM 환경 설정 파일은 baro_curl 프로그램을 실행하여 반드시 생성하는데, BaroPAM 인증 모듈의 디렉토리인 /usr/baropam 밑에 위치하도록 한다.

형식)

```
baro_curl -r rate_limit -R rate_time -t cycle_time -k key_method -e encrypt_flag -H hostname -u auth_url -s filename
```

BaroPAM 환경설정 파일의 설정 옵션에 대한 내용은 다음과 같다.

옵션	설명	설정값	비고
-r	일회용 인증키의 제한횟수(1~10)	3	
-R	일회용 인증키의 제한시간(초, 15~600초)	30	
-t	일회용 인증키의 인증주기(초, 3~60초)	30	
-k	일회용 인증키의 인증방식(app1, app256, app384, app512).	app512	
-e	환경설정 파일의 암호화 여부(yes or no)	no	
-H	서버의 호스트명(uname -n)	nurit.co.kr	
-u	호출할 URL로 호스트명(hostname), 사용자 계정(username), 인증주기(cycle_time), 일회용 인증키(auth_key) 등의 파라미터가 포함되어 호출	http://1.23.456.789/baropam/web/result_curl.jsp	
-s	BaroPAM 환경설정 파일을 생성할 디렉토리를 포함한 파일명	/usr/baropam/.baro_curl	

주의) -s 옵션의 filename는 BaroPAM 환경설정 파일을 생성할 디렉토리를 포함한 파일명(파일 접근권한은 444)이며, 설정한 서버의 호스트명(hostname)이 맞지 않는 경우 BaroPAM이 정상적으로 작동되지 않을 수 있으니, 호스트명(hostname)가 변경되는 경우 반드시 환경 설정의 해당 항목에 반영해야 한다.

사용 예)

```
[root] /usr/baropam > ./baro_curl -r 3 -R 30 -t 30 -k app512 -e no -H nurit.co.kr -u http://1.23.456.789/baropam/web/result_curl.jsp
```

- 1) 다음에 나오는 물음에 대해서는 모두 "y"를 입력한다.
 "/usr/baropam/.baro_curl" 파일을 업데이트하시겠습니까 (y/n) y
 중간자(man-in-the-middle) 공격을 예방할 것인가 (y/n) y

BaroPAM 환경 설정 파일인 .baro_curl에 설정한 내용은 다음과 같다.

```
[root] /usr/baropam > cat .baro_curl
```

```
" AUTH_KEY
" RATE_LIMIT 3 30
" AUTH_URL http://1.23.456.789/baropam/web/result_curl.jsp
" KEY_METHOD app512
" CYCLE_TIME 30
" HOSTNAME baropam
" DISALLOW_REUSE
```

BaroPAM 환경설정 파일인 .baro_curl의 설정 항목에 대한 내용은 다음과 같다.

항목	설명	설정값	비고
AUTH_KEY	인증 구분자(고정)		
RATE_LIMIT	일회용 인증키의 제한횟수(1~10), 제한시간(초, 15~600초)	3 30	
AUTH_URL	호출할 URL로 호스트명(hostname), 사용자 계정(username), 인증주기(cycle_time), 일회용 인증키(auth_key) 등의 파라미터가 포함되어 호출	http://1.23.456.789/baropam/web/result_curl.jsp	
KEY_METHOD	일회용 인증키의 인증방식(app1, app256, app384, app512)	app512	
CYCLE_TIME	일회용 인증키의 인증주기(초, 3~60초)	30	
HOSTNAME	서버의 호스트명(uname -n)	nurit.co.kr	
DISALLOW_REUSE or ALLOW_REUSE	중간자(man-in-the-middle) 공격을 예방할 경우는 "DISALLOW_REUSE"을 설정한 경우 일회용 인증키의 인증주기 동안은 다른 사용자가 로그인 할 수 없으며, 만약 허용할 경우는 "ALLOW_REUSE"을 설정한다.	DISALLOW_REUSE	

3.4 BaroPAM 환경 설정

1) PAM 인증: 환경 설정 정보를 File에 설정

BaroPAM 모듈을 설정하기 위해서 radiusrd 파일에 설정하는 방법은 다음과 같이 최 상단에 입력해 준다.

```
[root] /usr/baropam > vi /etc/pam.d/radiusd
auth required /usr/baropam/pam_baro_auth.so forward_pass secret=/usr/baropam/.baro_auth
encrypt=no
```

참고로 secret 파라미터는 BaroPAM 환경설정 파일명, encrypt 파라미터는 BaroPAM 환경설정 파일의 암호화 플래그(yes or no)를 설정한다.

만약, 계정마다 BaroPAM 환경 설정파일을 각각 설정하는 경우 BaroPAM 모듈을 설정하기 위해서 radiusrd 파일에 설정하는 방법은 다음과 같이 최 상단에 입력해 준다.

```
[root] /usr/baropam > vi /etc/pam.d/radiusd
##PAM-1.0
auth required /usr/baropam/pam_baro_auth.so forward_pass secret=${HOME}/.baro_auth
encrypt=no
```

계정마다 BaroPAM 환경 설정파일을 각각 설정하지 않고 특정 디렉토리에 계정별로 BaroPAM 환경 설정파일을 다르게 설정하고자 하는 경우 BaroPAM 모듈을 설정하기 위해서 sshd 파일에 설정하는 방법은 다음과 같

이 최 상단에 입력해 준다.

```
[root] /usr/baropam > vi /etc/pam.d/radiusd
#%PAM-1.0
auth                required                /usr/baropam/pam_baro_auth.so    forward_pass
secret=/usr/baropam/radius/.${USER}_auth encrypt=no
```

filezilla처럼 "Interactive process"가 불가능한 프로그램들을 위해서는 PAM에서 **forward_pass** 옵션을 사용하여 암호 입력 시에 암호와 **일회용 인증키**를 같이 입력하도록 하는 수밖에 없다. 이 경우, openssh client나 Windows의 RDP(Remote Desktop Protocol), Radius, filezilla 등 모두 이렇게 입력을 하는 수밖에 없다.



forward_pass를 이용하여 암호 입력창(Password:)에 암호와 같이 **일회용 인증키**를 입력할 경우, 암호를 먼저 입력하고 공백 없이 이어서 **일회용 인증키**를 입력하면 된다. 예를 들어 암호가 "baropam" 이고 **일회용 인증키**가 "123456" 이라면 "baropam123456"으로 입력하면 된다.

forward_pass를 이용하면 인증을 필요로 하는 대부분의 서비스에 **2-factor 인증**을 가능하게 할 수 있다.

2) PAM 인증: 환경 설정 정보를 MariaDB에 설정

BaroPAM 모듈을 설정하기 위해서 **radiusd** 파일에 설정하는 방법은 다음과 같이 최 상단에 입력해 준다.

```
[root] /usr/baropam > vi /etc/pam.d/radiusd
auth    required    /usr/baropam/pam_baro_sql.so forward_pass secret=/usr/baropam/.baro_sql
encrypt=no auth=radiusd
```

참고로 **secret** 파라미터는 **BaroPAM** 환경설정 파일명, **encrypt** 파라미터는 **BaroPAM** 환경설정 파일의 암호화 플래그(yes or no)를 설정한다.

filezilla처럼 "Interactive process"가 불가능한 프로그램들을 위해서는 PAM에서 **forward_pass** 옵션을 사용하여 암호 입력 시에 암호와 **일회용 인증키**를 같이 입력하도록 하는 수밖에 없다. 이 경우, openssh client나 Windows의 RDP(Remote Desktop Protocol), Radius, filezilla 등 모두 이렇게 입력을 하는 수밖에 없다.



`forward_pass`를 이용하여 암호 입력창(Password:)에 암호와 같이 **일회용 인증키**를 입력할 경우, 암호를 먼저 입력하고 공백 없이 이어서 **일회용 인증키**를 입력하면 된다. 예를 들어 암호가 "baropam" 이고 **일회용 인증키**가 "123456" 이라면 "baropam123456"으로 입력하면 된다.

`forward_pass`를 이용하면 인증을 필요로 하는 대부분의 서비스에 **2-factor 인증**을 가능하게 할 수 있다.

3) curl 인증

BaroPAM 모듈을 설정하기 위해서 `radiusd` 파일 등에 설정하는 방법은 다음과 같이 최 상단에 입력해 준다.

```
[root] /usr/baropam > vi /etc/pam.d/radiusd
auth      required      /usr/baropam/pam_baro_curl.so forward_pass secret=/usr/baropam/.baro_curl
encrypt=no
```

참고로 `secret` 파라미터는 **BaroPAM** 환경설정 파일명, `encrypt` 파라미터는 **BaroPAM** 환경설정 파일을 암호화 플래그(yes or no) 설정한다.

filezilla처럼 "Interactive process"가 불가능한 프로그램들을 위해서는 PAM에서 `forward_pass` 옵션을 사용하여 암호 입력 시에 암호와 **일회용 인증키**를 같이 입력하도록 하는 수밖에 없다. 이 경우, openssh client나 Windows의 RDP(Remote Desktop Protocol), Radius, filezilla 등 모두 이렇게 입력을 하는 수밖에 없다.



`forward_pass`를 이용하여 암호 입력창(Password:)에 암호와 같이 **일회용 인증키**를 입력할 경우, 암호를 먼저 입력하고 공백 없이 이어서 **일회용 인증키**를 입력하면 된다. 예를 들어 암호가 "baropam" 이고 **일회용 인증키**가 "123456" 이라면 "baropam123456"으로 입력하면 된다.

`forward_pass`를 이용하면 인증을 필요로 하는 대부분의 서비스에 **2-factor 인증**을 가능하게 할 수 있다.

3) ACL(Access Control list) 설정

1) PAM 인증(환경 설정 정보를 File에 설정)인 경우 **BaroPAM** 모듈 사용 시 **2차 인증**에서 제외할 계정에 대한 ACL에 제외해야 하는 경우 **BaroPAM** 환경 설정 시 설정한 디렉토리에 ACL 파일을 생성한 후 제외할 계정을 다음과 같이 입력한다. (.baro_acl에 대한 파일 접근권한을 444로 설정해야 한다.)

```
[root] /usr/baropam > vi .baro_acl
barokey
baropam
```

2) PAM 인증(환경 설정 정보를 MariaDB에 설정)인 경우는 Mariadb의 ACL 설정 테이블을 사용해야 한다.

4) NTP(Network Time Protocol) 설정

BaroPAM은 시간 동기화 방식이므로 서버의 시간이 현재 시간과 다를 경우 **일회용 인증키**가 서로 일치하지 않아서 서버에 로그인을 못하는 경우가 발생할 수 있다.

최근에는 정보자산에 대한 시간 동기화(타임서버 시간 동기화)하는 방법으로 NTP(Network Time Protocol)을 이용하여 **root** 계정에서 시스템의 시각을 현재 시각으로 설정할 수 있다.

NTP를 사용하기 위해서는 기본적으로 NTP 패키지가 반드시 설치되어 있어야 한다. 설치 확인은 다음의 명령어를 실행하여 확인한다. 만약, 설치되어 있지 않으면 Redhat, CentOS 8 이하 버전은 "yum install ntp" 그외는 "sudo apt-get install ntp" 명령어로 설치하면 된다.

```
[root]# rpm -qa | grep ntp
ntp-4.2.2p1-18.el5.centos
chkfontpath-1.10.1-1.1
```

ntpd 서비스를 서버 부팅 시 시작프로그램에 등록 및 ntp 활성화 여부 확인은 다음과 같은 명령어로 확인할 수 있다.

```
[root]# chkconfig ntpd on
[root]# chkconfig --list | grep ntp
ntpd          0:해제 1:해제 2:활성 3:활성 4:활성 5:활성 6:해제
```

chkconfig 이용하여 서버 부팅시 ntpd 데몬 활성화 여부 확인 3, 5 level에 off(해제)가 되어 있으면 자동 활성화되지 않는다. 자동 활성화하기 위해서는 3, 5에 on(활성)으로 다음과 같은 명령어로 변경해야 한다.

```
[root]# chkconfig --level 3 ntpd on
[root]# chkconfig --level 5 ntpd on
```

우리나라에서 운영되고 있는 NTP 서버는 다음과 같다.

```
server kr.pool.ntp.org
server time.bora.net
```

우리나라에서 운영되고 있는 NTP 서버를 ntpd 데몬 설정을 위한 설정 파일인 "/etc/ntp.conf"에 다음과 같이 설정한다.

```
[root]# vi /etc/ntp.conf
# Use public servers from the pool.ntp.org project.
# Please consider joining the pool (http://www.pool.ntp.org/join.html).
#server 0.centos.pool.ntp.org
#server 1.centos.pool.ntp.org
#server 2.centos.pool.ntp.org
#server 3.centos.pool.ntp.org
server kr.pool.ntp.org iburst
server time.bora.net iburst
```

iburst 옵션은 일종의 옵션 설정으로써 동기화 하는데 걸리는 시간을 짧게 줄여주는 옵션임.

ntpd 데몬 설정을 위한 설정이 끝나면 반드시 NTP 설정이 제대로 추가되었는지 확인한 후 NTP 데몬의 Restart 작업이 반드시 필요하다.

```
[root]# /etc/init.d/ntpd restart
```

```
ntpd를 종료 중: [ OK ]
ntpd (을)를 시작 중: [ OK ]
```

ntpd 시간 확인은 다음과 같은 명령어로 확인할 수 있다.

```
[root]# ntpq -p
      remote           refid      st t when poll reach  delay  offset  jitter
-----
static.betaidc. 106.247.248.106 3 u   7  64   1  2.884 287.718  0.001
time.bora.net   .INIT.         16 u   -  64   0  0.000  0.000  0.000
183.110.225.61 .INIT.         16 u   -  64   0  0.000  0.000  0.000
LOCAL(0)       .LOCL.         10 l   4  64   1  0.000  0.000  0.001
```

* 표시된 ip 가 현재 시간을 가져오고 있는 ntp 서버임

NTP를 사용하기 위해서는 기본적으로 NTP 패키지가 반드시 설치되어 있어야 한다. 설치 확인은 다음의 명령어를 실행하여 확인한다. 만약, 설치되어 있지 않으면 Redhat, CentOS 8 이상 버전은 "yum install chrony" 명령어로 설치하면 된다.

```
[root@baropam ~]# rpm -qa | grep chrony
chrony-3.5-1.el8.x86_64
```

우리나라에서 운영되고 있는 NTP 서버는 다음과 같다.

```
server kr.pool.ntp.org
server time.bora.net
```

우리나라에서 운영되고 있는 NTP 서버를 ntpd 데몬 설정을 위한 설정 파일인 "/etc/chrony.conf"에 다음과 같이 설정한다.

```
[root@baropam ~]# vi /etc/chrony.conf

# Use public servers from the pool.ntp.org project.
# Please consider joining the pool (http://www.pool.ntp.org/join.html).
#pool 2.centos.pool.ntp.org iburst
server kr.pool.ntp.org iburst
server time.bora.net iburst

# Record the rate at which the system clock gains/losses time.
driftfile /var/lib/chrony/drift

# Allow the system clock to be stepped in the first three updates
# if its offset is larger than 1 second.
makestep 1.0 3

# Enable kernel synchronization of the real-time clock (RTC).
rtcsync

# Enable hardware timestamping on all interfaces that support it.
#hwtimestamp *

# Increase the minimum number of selectable sources required to adjust
# the system clock.
#minsources 2
```

```
# Allow NTP client access from local network.
allow 192.168.0.0/16

# Serve time even if not synchronized to a time source.
#local stratum 10

# Specify file containing keys for NTP authentication.
keyfile /etc/chrony.keys

# Get TAI-UTC offset and leap seconds from the system tz database.
leapsectz right/UTC

# Specify directory for log files.
logdir /var/log/chrony

# Select which information is logged.
#log measurements statistics tracking
```

ntpd 데몬 설정을 위한 설정이 끝나면 반드시 NTP 설정이 제대로 추가되었는지 확인한 후 NTP 데몬의 Restart 작업이 반드시 필요하다. (chrony 서비스 시작 및 부팅시 구동 등록)

```
[root@baropam ~]# sudo systemctl enable chronyd
[root@baropam ~]# sudo systemctl restart chronyd
```

ntpd 시간 확인은 다음과 같은 명령어로 확인할 수 있다.

시간을 받아오는 서버 리스트 / chrony.conf 파일에 등록된 server 리스트)

```
[root@baropam ~]# chronyc sources
210 Number of sources = 2
MS Name/IP address          Stratum Poll Reach LastRx Last sample
=====
^* ec2-54-180-134-81.ap-nor>  2  6  377  43  -349us[-1059us] +/-  24ms
^~ time.bora.net              2  6  377  42  +1398us[+1398us] +/-  90ms
```

시간을 받아 오는 서버 정보)

```
[root@baropam ~]# chronyc tracking
Reference ID    : 36B48651 (ec2-54-180-134-81.ap-northeast-2.compute.amazonaws)
Stratum        : 3
Ref time (UTC) : Sun Mar 22 07:07:43 2020
System time    : 0.000130027 seconds slow of NTP time
Last offset    : -0.000710122 seconds
RMS offset     : 0.000583203 seconds
Frequency      : 19.980 ppm fast
Residual freq  : +0.142 ppm
Skew           : 3.235 ppm
Root delay     : 0.013462566 seconds
Root dispersion: 0.017946836 seconds
Update interval: 65.0 seconds
Leap status    : Normal
```

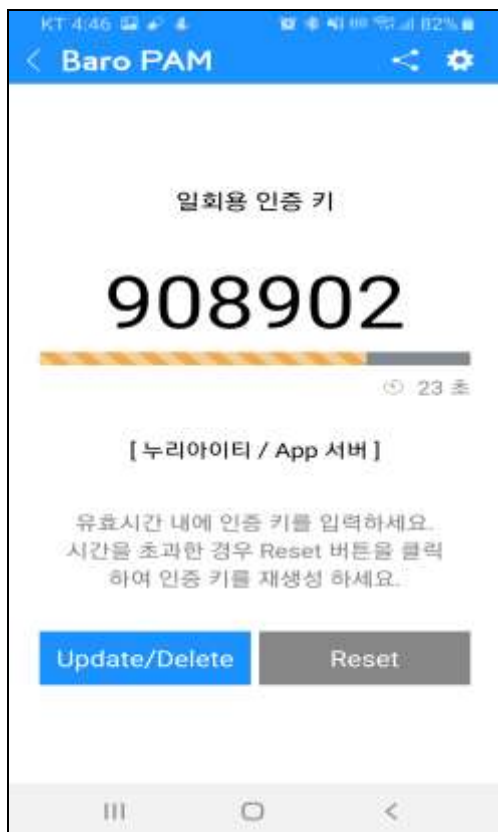
시간 상태 및 동기화 등 정보 확인)

```
[root@baropam ~]# timedatectl status
Local time: Sun 2020-03-22 16:08:45 KST
Universal time: Sun 2020-03-22 07:08:45 UTC
RTC time: Sun 2020-03-22 07:08:44
Time zone: Asia/Seoul (KST, +0900)
System clock synchronized: yes
NTP service: active
RTC in local TZ: no
```

5) BaroPAM 앱과 BaroPAM 웹 사이트



일회용 인증키 발생기인 BaroPAM 앱의 안드로이드 폰용 다운로드 (<https://play.google.com/store/apps/details?id=com.baro.pam>)는 구글의 "Play 스토어"에서 아이폰 용은 애플의 "App store"에서 가능하며, 설치하는 일반 앱의 설치와 동일하다.



BaroPAM 앱(안드로이드 폰용과 아이폰 용)에 문제가 발생 시 서비스의 중단이 발생하지 않도록 BaroPAM 웹 사이트(www.barpam.com)를 통해서도 서비스를 제공한다.



4. MySQL/MariaDB 설치 및 구성

4.1 MariaDB 설치

```
[root@localhost ~]# dnf -y install mariadb-server
```

MariaDB 설치 작업이 정상적으로 끝나면 다음과 같은 명령어를 사용하여 MariaDB를 기동시킨다.

```
[root@localhost ~]# systemctl start mariadb
```

MariaDB와 함께 제공되는 스크립트를 실행하여 MariaDB 보안 옵션을 개선하기 위한 몇 가지 단계를 진행해야 한다.

```
[root@localhost ~]# mysql_secure_installation
```

일련의 프롬프트가 표시되며, 비밀번호를 설정한 것을 모르는 경우 프롬프트가 표시되면 Enter 키를 누른다.

```
Enter current password for root (enter for none): Enter
```

다음으로 새 root 비밀번호를 설정할 것인지 확인하고 강력한 비밀번호를 설정한다.

```
Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.
Set root password? [Y/n] y
New password: baropam
Re-enter new password: baropam
Password updated successfully!
Reloading privilege tables..
... Success!
```

다음으로 이어지는 프롬프트에 대해 Enter 키를 누르기만 하면 된다.

익명 사용자를 제거한다.

```
Remove anonymous users? [Y/n] y
... Success!
```

root 로그인을 원격으로 허용하지 않는다.

```
Disallow root login remotely? [Y/n] y
... Success!
```

테스트 데이터베이스를 제거한다.

```
Remove test database and access to it? [Y/n] y
- Dropping test database...
... Success!
- Removing privileges on test database...
```

```
... Success!
```

권한 테이블을 다시 로드한다.

```
Reload privilege tables now? [Y/n] y
... Success!
Cleaning up...
```

4.2 MariaDB 구성

1) PAM 인증: 환경 설정 정보를 MariaDB에 설정

먼저 FreeRADIUS에 대한 데이터베이스와 데이터베이스 사용자를 생성한 다음 데이터베이스와 비밀번호로 식별되는 사용자를 생성한다.

예)

```
Database: baropamdb
User: nurit
Password: baropams
```

사용자와 비밀번호를 원하는 대로 바꿀 수 있지만 값을 적절하게 바꾸려면 나중에 수행할 구성에 주의를 기울여야 한다.

root로 MySQL/MariaDB 콘솔에 액세스하여 시작한다.

```
[root@baropam /]# mysql -uroot -p
Enter password: baropam
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.26 Source distribution

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\w' to clear the current input statement.
```

명령을 실행하여 데이터베이스 및 사용자를 생성한다.

```
MariaDB [(none)]> CREATE DATABASE baropamdb;
MariaDB [(none)]> CREATE USER 'nurit'@'localhost' IDENTIFIED BY 'baropams';
MariaDB [(none)]> GRANT ALL ON baropamdb.* TO 'nurit'@'localhost';
MariaDB [(none)]> FLUSH PRIVILEGES;
MariaDB [(none)]> use baropamdb
```

참고) 원격에서 MariaDB를 접속하려면

MariaDB를 원격에서 접속하려면 서버 설정 파일 수정과 데이터베이스 사용자 권한 설정, 그리고 방화벽 설

정 세 가지 주요 단계를 거쳐야 한다.

① MariaDB 설정 파일 수정

MariaDB는 기본적으로 보안을 위해 로컬(127.0.0.1)에서만 접속을 허용하도록 설정되어 있다.

설정 파일은 사용 중인 OS와 버전에 따라 설정 파일의 위치가 다를 수 있다.

일반적으로 /etc/my.cnf, /etc/mysql/my.cnf, 또는 /etc/mysql/mariadb.conf.d/50-server.cnf 등에서 찾을 수 있다.

설정 파일을 열고 [mysqld] 섹션에서 bind-address 설정을 찾아 수정한다.

전체 원격 접속 허용)

```
# bind-address = 127.0.0.1 ← 이 줄을 찾아서 주석 처리하거나 아래와 같이 변경
bind-address = 0.0.0.0
```

0.0.0.0으로 설정하면 서버의 모든 네트워크 인터페이스를 통해 접근을 허용한다.

설정을 적용하기 위해 MariaDB 서비스를 재시작한다.

```
sudo systemctl restart mariadb
# 또는 sudo service mariadb restart
```

② 원격 접속 사용자 권한 부여

설정 파일 수정 후, 데이터베이스 자체에서 원격 접속을 허용하는 사용자 계정을 생성하거나 기존 계정에 권한을 부여해야 한다.

MariaDB 접속하여 특정 사용자('newuser')가 모든 IP(%)에서 접속하고 모든 데이터베이스(*.*)에 대한 권한을 갖도록 설정한다.

새로운 원격 접속 사용자 생성 및 권한 부여)

```
GRANT ALL PRIVILEGES ON *.* TO 'newuser'@'%' IDENTIFIED BY 'your_password';
FLUSH PRIVILEGES;
```

만약 기존 사용자('root')에게 모든 IP(`%`)에서의 접속 권한을 부여하려면)

```
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'root_password_if_changed';
FLUSH PRIVILEGES;
```

특정 IP(`192.168.1.100`)만 허용하려면)

```
GRANT ALL PRIVILEGES ON *.* TO 'newuser'@'192.168.1.100' IDENTIFIED BY 'your_password';
FLUSH PRIVILEGES;
```

③ 방화벽 설정

서버 자체의 방화벽이 MariaDB 포트(기본값: 3306/TCP)로의 외부 접속을 차단하고 있을 수 있다.

방화벽 포트 허용: 사용 중인 방화벽(예: firewalld, ufw)에 따라 명령어는 달라진다.

Firewalld를 사용하는 경우(CentOS, RHEL 등):

```
sudo firewall-cmd --permanent --add-port=3306/tcp
sudo firewall-cmd -reload
```

UFW를 사용하는 경우 (Ubuntu, Debian 등):

```
sudo ufw allow 3306/tcp
sudo ufw reload
```

다음으로 RADIUS MySQL 스키마를 새로 생성된 데이터베이스로 생성한다.

```
#
# Table structure for table 'TB_BARO_HOST'
#

CREATE TABLE IF NOT EXISTS TB_BARO_HOST (
  HOSTNAME      VARCHAR(30) NOT NULL default '',
  RATE_CNT      VARCHAR(2)  NOT NULL default '5',
  RATE_SEC      VARCHAR(3)  NOT NULL default '30',
  RATE_TIME     VARCHAR(110) NULL default '',
  KEY_METHOD    VARCHAR(6)  NOT NULL default 'app512',
  CYCLE_TIME    VARCHAR(2)  NOT NULL default '60',
  SECURE_KEY    VARCHAR(32) NOT NULL default 'j1q1cHbVqdpj7b4PzBpM2Di1eBvmHFV/',
  ACL_TYPE      VARCHAR(5)  NOT NULL default 'deny',
  MIDDLE_TYPE   VARCHAR(14) NOT NULL default 'DISALLOW_REUSE',
  MIDDLE_TIME   VARCHAR(8)  NULL default '',
  ENV_TYPE      VARCHAR(8)  NOT NULL default 'share',
  PRIMARY KEY (HOSTNAME)
) ENGINE = INNODB;

#
# Table structure for table 'TB_HOST_EOTA'
#

CREATE TABLE IF NOT EXISTS TB_HOST_EOTA (
  HOSTNAME      VARCHAR(30) NOT NULL default '',
  EMERGENCY_KEY VARCHAR(8)  NOT NULL default '',
  PRIMARY KEY (HOSTNAME,EMERGENCY_KEY)
) ENGINE = INNODB;

#
# Table structure for table 'TB_BARO_ACL'
#

CREATE TABLE IF NOT EXISTS TB_BARO_ACL (
  HOSTNAME      VARCHAR(30) NOT NULL default '',
  USERNAME      VARCHAR(40) NOT NULL default '',
  PRIMARY KEY (HOSTNAME,USERNAME)
) ENGINE = INNODB;

#
```

```

# Table structure for table 'TB_BARO_USER'
#
CREATE TABLE IF NOT EXISTS TB_BARO_USER (
  HOSTNAME      VARCHAR(30) NOT NULL default '',
  USERNAME      VARCHAR(40) NOT NULL default '',
  RATE_CNT      VARCHAR(2)  NOT NULL default '5',
  RATE_SEC      VARCHAR(3)  NOT NULL default '30',
  RATE_TIME     VARCHAR(110) NULL default '',
  KEY_METHOD    VARCHAR(6)  NOT NULL default 'app512',
  CYCLE_TIME    VARCHAR(2)  NOT NULL default '60',
  SECURE_KEY    VARCHAR(32) NOT NULL default 'j|q|c|H|b|V|q|d|p|j|7|b|4|P|z|B|p|M|2|D|i|e|B|v|m|H|F|V|',
  ACL_TYPE      VARCHAR(5)  NOT NULL default 'deny',
  MIDDLE_TYPE   VARCHAR(14) NOT NULL default 'DISALLOW_REUSE',
  MIDDLE_TIME   VARCHAR(8)   NULL default '',
  PRIMARY KEY (HOSTNAME,USERNAME)
) ENGINE = INNODB;

#
# Table structure for table 'TB_USER_EOTA'
#
CREATE TABLE IF NOT EXISTS TB_USER_EOTA (
  HOSTNAME      VARCHAR(30) NOT NULL default '',
  USERNAME      VARCHAR(40) NOT NULL default '',
  EMERGENCY_KEY VARCHAR(8)  NOT NULL default '',
  PRIMARY KEY (HOSTNAME,USERNAME,EMERGENCY_KEY)
) ENGINE = INNODB;

#
# Table structure for table 'TB_BARO_LOG'
#
CREATE TABLE IF NOT EXISTS TB_BARO_LOG (
  AUTH_DTTM     VARCHAR(10) NOT NULL default '',
  HOSTNAME      VARCHAR(30) NOT NULL default '',
  USERNAME      VARCHAR(40) NOT NULL default '',
  REMOTE_IP     VARCHAR(30)  NULL default '',
  AUTH_MSG      VARCHAR(200) NOT NULL default '',
  PRIMARY KEY (AUTH_DTTM, HOSTNAME, USERNAME)
) ENGINE = INNODB;

```

참고) 스키마 정보

1. 서버 설정 마스터 (TB_BARO_HOST)

NO	한글명	영문명	Type	길이	필수	Key	비고
1	호스트명	HOSTNAME	VAR	30	Y	PK	서버의 호스트명(uname -n)
2	제한횟수	RATE_CNT	VAR	2	Y		인증키의 제한횟수(1~10)
3	제한기간	RATE_SEC	VAR	3	Y		인증키의 제한시간(초, 15~600초)
4	제한시간	RATE_TIME	VAR	110	N		인증키의 제한시간(내부적으로 사용)
5	인증방식	KEY_METHOD	VAR	6	Y		인증키의 인증방식(app1, app256, app384, app512)
6	인증주기	CYCLE_TIME	VAR	2	Y		인증키의 인증주기(초, 3~60초)
7	Secure키	SECURE_KEY	VAR	32	Y		벤더에서 제공하는 Secure key(라이선스 키)
8	ACL구분	ACL_TYPE	VAR	5	Y		2차 인증에서 허용(allow) 또는 제외(deny) 구분
9	중간자공격방어구분	MIDDLE_TYPE	VAR	14	Y		중간자(man-in-the-middle) 공격을 예방할 경우는 "DISALLOW_REUSE"을 설정한 경우 일회용 인증키의 인증주기 동안은 다른 사용자가 로그인 할 수 없으며, 만약 허용할 경우는 "ALLOW_REUSE"을 설정
10	중간자공격방어시간	MIDDLE_TIME	VAR	8	N		중간자공격을 방어할 경우에 사용(내부적으로 사용)
11	환경설정구분	ENV_TYPE	VAR	8	Y		환경설정 정보를 공유(share)해서 사용할 것인지, 사용자별(username)로 설정하지 구분

2. 서버 응급일회용인증키 마스터 (TB_HOST_EOTA)

NO	한글명	영문명	Type	길이	필수	Key	비고
1	호스트명	HOSTNAME	VAR	30	Y	PK,FK	서버의 호스트명(uname -n)
2	응급일회용키	EMERGENCY_KEY	VAR	8	Y	PK	인증키 생성기인 BaroPAM 앱을 사용할 수 없을 때 분실한 경우를 대비하여 SSH 서버에 다시 액세스하는데 사용

3. ACL 설정 마스터 (TB_BARO_ACL)

NO	한글명	영문명	Type	길이	필수	Key	비고
1	호스트명	HOSTNAME	VAR	30	Y	PK,FK	서버의 호스트명(uname -n)
2	계정(사용자)명	USERNAME	VAR	40	Y	PK	서버에 로그인하는 계정

4. 계정(사용자)별 설정 마스터 (TB_BARO_USER)

NO	한글명	영문명	Type	길이	필수	Key	비고
1	호스트명	HOSTNAME	VAR	30	Y	PK,FK	서버의 호스트명(uname -n)
2	계정(사용자)명	USERNAME	VAR	40	Y	PK	서버에 로그인하는 계정
3	제한횟수	RATE_CNT	VAR	2	Y		인증키의 제한횟수(1~10)
4	제한기간	RATE_SEC	VAR	3	Y		인증키의 제한시간(초, 15~600초)
5	제한시간	RATE_TIME	VAR	110	N		인증키의 제한시간(내부적으로 사용)
6	인증방식	KEY_METHOD	VAR	6	Y		인증키의 인증방식(app1, app256, app384, app512)
7	인증주기	CYCLE_TIME	VAR	2	Y		인증키의 인증주기(초, 3~60초)
8	Secure키	SECURE_KEY	VAR	32	Y		벤더에서 제공하는 Secure key(라이선스 키)
9	ACL구분	ACL_TYPE	VAR	5	Y		2차 인증에서 허용(allow) 또는 제외(deny) 구분
10	중간자공격방어구분	MIDDLE_TYPE	VAR	14	Y		중간자(man-in-the-middle) 공격을 예방할 경우는 "DISALLOW_REUSE"을 설정한 경우 일회용 인증키의 인증주기 동안은 다른 사용자가 로그인 할 수 없으며, 만약 허용할 경우는 "ALLOW_REUSE"을 설정
11	중간자공격방어시간	MIDDLE_TIME	VAR	8	N		중간자공격을 방어할 경우에 사용(내부적으로 사용)

5. 계정(사용자)별 응급일회용인증키 마스터 (TB_USER_EOTA)

NO	한글명	영문명	Type	길이	필수	Key	비고
1	호스트명	HOSTNAME	VAR	30	Y	PK,FK	서버의 호스트명(uname -n)
2	계정(사용자)명	USERNAME	VAR	40	Y	PK,FK	서버에 로그인하는 계정
3	응급일회용키	EMERGENCY_KEY	VAR	8	Y	PK	인증키 생성기인 BaroPAM 앱을 사용할 수 없을 때 분실한 경우를 대비하여 SSH 서버에 다시 액세스하는데 사용

6. 인증 로그 마스터 (TB_BARO_LOG)

NO	한글명	영문명	Type	길이	필수	Key	비고
1	인증시간	AUTH_DTTM	VAR	10	Y	PK	국제적인 표준시간
2	호스트명	HOSTNAME	VAR	30	Y	PK,FK	서버의 호스트명(uname -n)
3	사용자명	USERNAME	VAR	40	Y	PK,FK	서버에 로그인하는 계정
4	인증메시지	AUTH_MSG	VAR	200	Y		

2) SQL 인증인 경우

먼저 FreeRADIUS에 대한 데이터베이스와 데이터베이스 사용자를 생성한 다음 데이터베이스와 비밀번호로 식별되는 사용자를 생성한다.

예)

```
Database: radius
User: radius
Password: baropam
```

사용자와 비밀번호를 원하는 대로 바꿀 수 있지만 값을 적절하게 바꾸려면 나중에 수행할 구성에 주의를 기울여야 한다.

root로 MySQL/MariaDB 콘솔에 액세스하여 시작한다.

```
[root@baropam /]# mysql -uroot -p
Enter password: baropam
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.26 Source distribution

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\w' to clear the current input statement.
```

명령을 실행하여 데이터베이스 및 사용자를 생성한다.

```
MariaDB [(none)]> CREATE DATABASE radius;
MariaDB [(none)]> GRANT ALL ON radius.* TO radius@localhost IDENTIFIED BY "baropam";
MariaDB [(none)]> FLUSH PRIVILEGES;
MariaDB [(none)]> use radius
```

참고) 원격에서 MariaDB를 접속하려면

MariaDB를 원격에서 접속하려면 서버 설정 파일 수정과 데이터베이스 사용자 권한 설정, 그리고 방화벽 설정 세 가지 주요 단계를 거쳐야 한다.

① MariaDB 설정 파일 수정

MariaDB는 기본적으로 보안을 위해 로컬(127.0.0.1)에서만 접속을 허용하도록 설정되어 있다.

설정 파일은 사용 중인 OS와 버전에 따라 설정 파일의 위치가 다를 수 있다.

일반적으로 /etc/my.cnf, /etc/mysql/my.cnf, 또는 /etc/mysql/mariadb.conf.d/50-server.cnf 등에서 찾을 수 있다.

설정 파일을 열고 [mysqld] 섹션에서 bind-address 설정을 찾아 수정한다.

전체 원격 접속 허용)

```
# bind-address = 127.0.0.1 ← 이 줄을 찾아서 주석 처리하거나 아래와 같이 변경
bind-address = 0.0.0.0
```

0.0.0.0으로 설정하면 서버의 모든 네트워크 인터페이스를 통해 접근을 허용한다.

설정을 적용하기 위해 MariaDB 서비스를 재시작한다.

```
sudo systemctl restart mariadb
# 또는 sudo service mariadb restart
```

② 원격 접속 사용자 권한 부여

설정 파일 수정 후, 데이터베이스 자체에서 원격 접속을 허용하는 사용자 계정을 생성하거나 기존 계정에 권한을 부여해야 한다.

MariaDB 접속하여 특정 사용자('newuser')가 모든 IP(%)에서 접속하고 모든 데이터베이스(*.*)에 대한 권한을 갖도록 설정한다.

새로운 원격 접속 사용자 생성 및 권한 부여)

```
GRANT ALL PRIVILEGES ON *.* TO 'newuser'@'%' IDENTIFIED BY 'your_password';
FLUSH PRIVILEGES;
```

만약 기존 사용자('root')에게 모든 IP(`%`)에서의 접속 권한을 부여하려면)

```
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'root_password_if_changed';
FLUSH PRIVILEGES;
```

특정 IP(`192.168.1.100`)만 허용하려면)

```
GRANT ALL PRIVILEGES ON *.* TO 'newuser'@'192.168.1.100' IDENTIFIED BY 'your_password';
FLUSH PRIVILEGES;
```

③ 방화벽 설정

서버 자체의 방화벽이 MariaDB 포트(기본값: 3306/TCP)로의 외부 접속을 차단하고 있을 수 있다.

방화벽 포트 허용: 사용 중인 방화벽(예: firewalld, ufw)에 따라 명령어는 달라진다.

Firewalld를 사용하는 경우(CentOS, RHEL 등):

```
sudo firewall-cmd --permanent --add-port=3306/tcp
sudo firewall-cmd -reload
```

UFW를 사용하는 경우 (Ubuntu, Debian 등):

```
sudo ufw allow 3306/tcp
sudo ufw reload
```

다음으로 RADIUS MySQL 스키마를 새로 생성된 데이터베이스로 생성한다.

```
#####
# $Id: 2d0fb7e137f9d6f6a2a48d65e013841ed2bfadf9 $ #
# # #
# schema.sql rlm_sql - FreeRADIUS SQL Module #
# # #
# Database schema for MySQL rlm_sql module #
# # #
# To load: #
# mysql -uroot -prootpass radius < schema.sql #
# # #
# Mike Machado <mike@innercite.com> #
#####
#
# Table structure for table 'radacct'
```

```

#
CREATE TABLE IF NOT EXISTS radacct (
  radacctid bigint(21) NOT NULL auto_increment,
  acctsessionid varchar(64) NOT NULL default '',
  acctuniqueid varchar(32) NOT NULL default '',
  username varchar(64) NOT NULL default '',
  realm varchar(64) default '',
  nasipaddress varchar(15) NOT NULL default '',
  nasportid varchar(15) default NULL,
  nasporttype varchar(32) default NULL,
  acctstarttime datetime NULL default NULL,
  acctupdatetime datetime NULL default NULL,
  acctstoptime datetime NULL default NULL,
  acctinterval int(12) default NULL,
  acctsessiontime int(12) unsigned default NULL,
  acctauthentic varchar(32) default NULL,
  connectinfo_start varchar(50) default NULL,
  connectinfo_stop varchar(50) default NULL,
  acctinputoctets bigint(20) default NULL,
  acctoutputoctets bigint(20) default NULL,
  calledstationid varchar(50) NOT NULL default '',
  callingstationid varchar(50) NOT NULL default '',
  acctterminatecause varchar(32) NOT NULL default '',
  servicetype varchar(32) default NULL,
  framedprotocol varchar(32) default NULL,
  framedipaddress varchar(15) NOT NULL default '',
  framedipv6address varchar(45) NOT NULL default '',
  framedipv6prefix varchar(45) NOT NULL default '',
  framedinterfaceid varchar(44) NOT NULL default '',
  delegatedipv6prefix varchar(45) NOT NULL default '',
  PRIMARY KEY (radacctid),
  UNIQUE KEY acctuniqueid (acctuniqueid),
  KEY username (username),
  KEY framedipaddress (framedipaddress),
  KEY framedipv6address (framedipv6address),
  KEY framedipv6prefix (framedipv6prefix),
  KEY framedinterfaceid (framedinterfaceid),
  KEY delegatedipv6prefix (delegatedipv6prefix),
  KEY acctsessionid (acctsessionid),
  KEY acctsessiontime (acctsessiontime),
  KEY acctstarttime (acctstarttime),
  KEY acctinterval (acctinterval),
  KEY acctstoptime (acctstoptime),
  KEY nasipaddress (nasipaddress)
) ENGINE = INNODB;

#
# Table structure for table 'radcheck'
#

CREATE TABLE IF NOT EXISTS radcheck (
  id int(11) unsigned NOT NULL auto_increment,

```

```
username varchar(64) NOT NULL default '',
phone_no varchar(32) NOT NULL default '',
cycle_time varchar(6) NOT NULL default '60',
attribute varchar(64) NOT NULL default '',
op char(2) NOT NULL DEFAULT '=',
value varchar(253) NOT NULL default '',
PRIMARY KEY (id),
KEY username (username(32))
);

#
# Table structure for table 'radgroupcheck'
#

CREATE TABLE IF NOT EXISTS radgroupcheck (
  id int(11) unsigned NOT NULL auto_increment,
  groupname varchar(64) NOT NULL default '',
  attribute varchar(64) NOT NULL default '',
  op char(2) NOT NULL DEFAULT '=',
  value varchar(253) NOT NULL default '',
  PRIMARY KEY (id),
  KEY groupname (groupname(32))
);

#
# Table structure for table 'radgroupreply'
#

CREATE TABLE IF NOT EXISTS radgroupreply (
  id int(11) unsigned NOT NULL auto_increment,
  groupname varchar(64) NOT NULL default '',
  attribute varchar(64) NOT NULL default '',
  op char(2) NOT NULL DEFAULT '=',
  value varchar(253) NOT NULL default '',
  PRIMARY KEY (id),
  KEY groupname (groupname(32))
);

#
# Table structure for table 'radreply'
#

CREATE TABLE IF NOT EXISTS radreply (
  id int(11) unsigned NOT NULL auto_increment,
  username varchar(64) NOT NULL default '',
  attribute varchar(64) NOT NULL default '',
  op char(2) NOT NULL DEFAULT '=',
  value varchar(253) NOT NULL default '',
  PRIMARY KEY (id),
  KEY username (username(32))
);
```

```
#
# Table structure for table 'radusergroup'
#

CREATE TABLE IF NOT EXISTS radusergroup (
  id int(11) unsigned NOT NULL auto_increment,
  username varchar(64) NOT NULL default '',
  groupname varchar(64) NOT NULL default '',
  priority int(11) NOT NULL default '1',
  PRIMARY KEY (id),
  KEY username (username(32))
);

#
# Table structure for table 'radpostauth'
#

CREATE TABLE IF NOT EXISTS radpostauth (
  id int(11) NOT NULL auto_increment,
  username varchar(64) NOT NULL default '',
  pass varchar(64) NOT NULL default '',
  reply varchar(32) NOT NULL default '',
  authdate timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (id),
  KEY username (username(32))
) ENGINE = INNODB;

#
# Table structure for table 'nas'
#

CREATE TABLE IF NOT EXISTS nas (
  id int(10) NOT NULL auto_increment,
  nasname varchar(128) NOT NULL,
  shortname varchar(32),
  type varchar(30) DEFAULT 'other',
  ports int(5),
  secret varchar(60) DEFAULT 'secret' NOT NULL,
  server varchar(64),
  community varchar(50),
  description varchar(200) DEFAULT 'RADIUS Client',
  PRIMARY KEY (id),
  KEY nasname (nasname)
);
```

5. FreeRADIUS 연동 테스트

5.1 PAM, cURL 인증

1) 테스트할 계정 생성

테스트할 로컬 계정을 다음과 같이 생성한다.

```
[root@localhost ~]# useradd raduser
[root@localhost ~]# passwd raduser
Changing password for user raduser.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

사용자의 로그인 정보(로그인-ID, 전화번호) 관련 쉘 스크립트를 실행하기 위하여 BaroPAM 등록 디렉토리 (/usr/baropam/radius)로 이동하여 실행한다.

참고) 쉘 스크립트 실행 시 인수나 인수값의 구분은 공백(space)으로 해야 한다.

예) 사용자(로그인-ID)를 생성하는 쉘 스크립트 (auths=pam인 경우) - setuser.sh

```
#!/bin/sh

export LANG=C
ENV_HOME=/usr/baropam/radius;
ACC_HOME=/home/$1

userdel -rf $1
Wrm ${ENV_HOME}/.$1_auth

useradd -d ${ACC_HOME} -m -s /bin/bash $1
echo "$1:$2" | chpasswd

Wcp ${ENV_HOME}/.baro_auth ${ENV_HOME}/.$1_auth

sed -i "s/01012341234/$3/g" ${ENV_HOME}/.$1_auth
```

사용자(로그인-ID)를 생성하는 쉘 스크립트(setuser.sh) 실행 시 파라미터는 다음과 같다.

\$1 : 생성할 로그인-ID
 \$2 : 로그인-ID의 비밀번호
 \$3 : 로그인-ID의 전화번호

```
[root@localhost ~]# sh setuser.sh nurit baropam 01027714076
```

예) 사용자(로그인-ID)의 비밀번호를 변경하는 쉘 스크립트 (auths=pam인 경우) - setpasswd.sh

```
#!/bin/sh
```

```
export LANG=C
echo "$1:$2" | chpasswd
```

사용자(로그인-ID)의 비밀번호를 변경하는 쉘 스크립트(setpasswd.sh) 실행 시 파라미터는 다음과 같다.
 \$1 : 로그인-ID
 \$2 : 변경할 비밀번호

```
[root@localhost ~]# sh setpasswd.sh nurit !@Baropam#
```

예) 사용자(로그인-ID)의 폰번호를 변경하는 쉘 스크립트 (auths=pam인 경우) - setphone.sh

```
#!/bin/sh
export LANG=C
ENV_HOME=/usr/baropam/radius;
sed -i "s/$2/$3/g" ${ENV_HOME}/.$1_auth
```

사용자(로그인-ID)의 폰번호를 변경하는 쉘 스크립트(setphone.sh) 실행 시 파라미터는 다음과 같다.
 \$1 : 로그인-ID
 \$2 : 변경전 폰번호
 \$3 : 변경후 폰번호

```
[root@localhost ~]# sh setphone.sh nurit 01012341234 01027714076
```

예) 사용자(로그인-ID)의 비밀번호와 폰번호를 변경하는 쉘 스크립트 (auths=pam인 경우) - chpasswd.sh

```
#!/bin/sh
export LANG=C
echo "$1:$2" | chpasswd
sed -i "s/$3/$4/g" ${ENV_HOME}/.$1_auth
```

사용자(로그인-ID)의 비밀번호와 폰번호를 변경하는 쉘 스크립트(setpasswd.sh) 실행 시 파라미터는 다음과 같다.
 \$1 : 로그인-ID
 \$2 : 변경할 비밀번호
 \$3 : 변경전 폰번호
 \$4 : 변경후 폰번호

```
[root@localhost ~]# sh chpasswd.sh nurit !@Baropam# 01012341234 01027714076
```

예) 사용자(로그인-ID)를 삭제하는 쉘 스크립트 (auths=pam인 경우) - deluser.sh

```
#!/bin/sh
export LANG=C
ENV_HOME=/usr/baropam/radius;
ACC_HOME=/home/$1
```

```
userdel -rf $1
Wrm ${ENV_HOME}/.$1_auth
```

사용자(로그인-ID)를 삭제하는 쉘 스크립트(deluser.sh) 실행 시 파라미터는 다음과 같다.
\$1 : 삭제할 로그인-ID

```
[root@localhost ~]# sh deluser.sh nurit
```

예) 사용자(로그인-ID)를 생성하는 쉘 스크립트 (auth=http인 경우) - setuser.sh

```
#!/bin/sh

export LANG=C
ENV_HOME=/usr/baropam/radius;
ACC_HOME=/home/$1

userdel -rf $1
Wrm ${ENV_HOME}/.$1_auth

useradd -d ${ACC_HOME} -m -s /bin/bash $1
echo "$1:$1" | chpasswd

Wcp ${ENV_HOME}/.baro_auth ${ENV_HOME}/.$1_auth
```

사용자(로그인-ID)를 생성하는 쉘 스크립트(setuser.sh) 실행 시 파라미터는 다음과 같다.
\$1 : 생성할 로그인-ID

```
[root@localhost ~]# sh setuser.sh nurit
```

예) 사용자(로그인-ID)를 삭제하는 쉘 스크립트 (auth=http인 경우) - deluser.sh

```
#!/bin/sh

export LANG=C
ENV_HOME=/usr/baropam/radius;
ACC_HOME=/home/$1

userdel -rf $1
Wrm ${ENV_HOME}/.$1_auth
```

사용자(로그인-ID)를 삭제하는 쉘 스크립트(deluser.sh) 실행 시 파라미터는 다음과 같다.
\$1 : 삭제할 로그인-ID

```
[root@localhost ~]# sh deluser.sh nurit
```

2) FreeRADIUS 디버그 모드로 실행

새 ssh 세션을 열고 디버그 모드에서 **radiusd**를 다음과 같이 실행한다.

```
[root@localhost ~]# radiusd -X
```

디버그 모드에서 **radiusd**를 실행 시 다음과 같은 오류가 발생하면

Then run `radiusd -X` but I have this error:

```
Failed reading private key file /etc/raddb/certs/server.pem:error:06065064:digital envelope
routines:EVP_DecryptFinal_ex:bad decrypt
rlm_eap_tls: Failed initializing SSL context
rlm_eap (EAP): Failed to initialise rlm_eap_tls
/etc/raddb/mods-enabled/eap[17]: Instantiation failed for module "eap"
```

다음과 같은 작업을 진행한 후 `radiusd`를 다음과 같이 실행한다.

```
[root@localhost ~]# cd /etc/raddb/certs
[root@localhost ~]# ./bootstrap
```

첫 번째 ssh 세션으로 전환하고 테스트한다.

형식)

```
radtest <username> (<password><verification code>) <hostname> <port> <shared_secret>
```

username: 테스트할 사용자 이름

password: 테스트할 사용자 비밀번호

verification code: BaroPAM 앱에서 생성한 일회용 인증키

hostname: RADIUS 서버의 IP 주소 또는 호스트 이름

port: RADIUS 서버의 인증 포트 - 0을 설정하는 것은 표준 포트(1812)를 사용하겠다는 의미.

shared_secret: RADIUS 서버와 공유하는 비밀 키

BaroPAM으로 FreeRADIUS 테스트를 다음과 같이 실행한다.

```
[root@localhost ~]# radtest root baropam024747 localhost 0 baropam
Sent Access-Request Id 243 from 0.0.0.0:39668 to 127.0.0.1:1812 length 74
  User-Name = "root"
  User-Password = "baropam903481"
  NAS-IP-Address = 10.0.2.15
  NAS-Port = 0
  Message-Authenticator = 0x00
  Cleartext-Password = "baropam903481"
Received Access-Accept Id 243 from 127.0.0.1:1812 to 127.0.0.1:39668 length 20
```

3) FreeRADIUS 데몬 실행

FreeRADIUS 데몬인 `radiusd`를 다음과 같이 백그라운드로 실행한다.

```
[root@localhost ~]# radiusd -s &
[1] 1961
```

FreeRADIUS 데몬인 `radiusd`가 정상적으로 기동되었는지 다음과 같이 확인한다.

```
[root@localhost ~]# ps -ef|grep radiusd
root      1961    1818  0 14:11 pts/1    00:00:00 radiusd -s
root      1964    1818  0 14:11 pts/1    00:00:00 grep --color=auto radiusd
```

FreeRADIUS 데몬인 `radiusd`가 사용하는 UDP 포트인 1812를 다음과 같이 확인한다.

```
[root@localhost ~]# netstat -an | grep 1812
udp        0      0 127.0.0.1:18120      0.0.0.0:*
udp        0      0 0.0.0.0:1812        0.0.0.0:*
udp6       0      0 :::1812              :::*
```

FreeRADIUS를 연동하여 BaroPAM에서 인증한 로그는 다음과 같이 확인한다.

```
[root@localhost ~]# tail -f /var/log/secure
Mar 26 13:54:11 localhost radiusd(pam_baro_auth)[1857]: Try to update RATE_LIMIT line.[3 30
1616734451]
Mar 26 13:56:46 localhost radiusd(pam_baro_auth)[1857]: Try to update RATE_LIMIT line.[3 30
1616734606]
Mar 26 14:00:48 localhost radiusd(pam_baro_auth)[1934]: Try to update RATE_LIMIT line.[3 30
1616734848]
Mar 26 14:00:48 localhost radiusd(pam_baro_auth)[1934]: Invalid verification code
Mar 26 14:00:48 localhost radiusd[1934]: pam_unix(radiusd:auth): authentication failure;
logname=root uid=0 euid=0 tty= ruser= rhost= user=scjool
Mar 26 14:01:13 localhost radiusd(pam_baro_auth)[1934]: Try to update RATE_LIMIT line.[3 30
1616734873]
Mar 26 14:01:36 localhost radiusd(pam_baro_auth)[1934]: Try to update RATE_LIMIT line.[3 30
1616734873 1616734896]
Mar 26 14:02:15 localhost radiusd(pam_baro_auth)[1934]: Try to update RATE_LIMIT line.[3 30
1616734935]
```

4) FreeRADIUS 데몬 종료

먼저 FreeRADIUS 데몬인 **radiusd**가 실행되었는지 다음과 같이 확인한다.

```
[root@localhost ~]# ps -ef|grep radiusd
root      1961    1818  0 14:11 pts/1    00:00:00 radiusd -s
root      1964    1818  0 14:11 pts/1    00:00:00 grep --color=auto radiusd
```

FreeRADIUS 데몬인 **radiusd**의 프로세스 ID(1961)를 확인 한 후 다음과 같이 종료한다.

```
[root@localhost ~]# kill -9 1961
```

참고) FreeRADIUS 관련 로그 파일: /var/log/radius/radius.log

5.2 SQL 인증

1) 사전 준비

MariaDB의 UDF(User Define Function)는 MariaDB에서 C 또는 C++로 작성된 외부 프로그램을 호출하거나 데이터를 전송할 때 사용된다.

C 또는 C++로 작성되어야 하며 운영 체제는 동적 로드를 지원해야 한다.

첨부 파일(libbaroudf.so)을 mysql shell에서 다음의 명령 결과를 디렉토리에 복사한다.

```
mysql> SHOW VARIABLES LIKE 'plugin_dir';
+-----+-----+
| Variable_name | Value                |
+-----+-----+
| plugin_dir    | /usr/lib64/mysql/plugin/ |
+-----+-----+
1 row in set (0.00 sec)
```

① BaroPAM 모듈 복사

```
[root@localhost ~]# cp libbaroudf.so /usr/lib64/mysql/plugin/
```

② BaroPAM 함수 생성

```
mysql> create function TO_VERIFYKEYL returns string soname 'libbaroudf.so';
Query OK, 0 rows affected (0.01 sec)
```

참고) TO_VERIFYKEYL 함수

- NAME

TO_VERIFYKEYL

- SYNOPSIS

void * TO_VERIFYKEYL(const void *username, const void *phone_no, const void *cycle_time)

- DESCRIPTION

일회용 인증키 생성 함수

username : 로그인 화면의 로그인-ID 항목에 입력한 ID를 설정.

phone_no: 사용자별 스마트 폰 번호를 숫자만 설정.

cycle_time: 사용자별로 지정한 **일회용 인증키**의 생성 주기(3-60초)를 설정.

- RETURN VALUES

생성된 **일회용 인증키**를 반환

참고) 만약, 사용자별로 스마트 폰 번호 및 개인별로 지정한 **일회용 인증키**의 생성 주기가 **일회용 인증키**의 생성기와 다른 경우 **일회용 인증키**가 달라서 검증에 실패할 수 있다. 반드시 정보를 일치시켜야 한다.

③ authorize_check_query 변경

/etc/raddb/mods-config/sql/main/mysql/queries.conf 파일 내용 중 authorize_check_query에 TO_VERIFYKEYL 함수로 변경한다.

예) **일회용 인증키**만 검증하는 경우

```
authorize_check_query = "W
    SELECT id, username, attribute, TO_VERIFYKEYL(username, phone_no, cycle_time) as value, op
W
    FROM ${authcheck_table} W
    WHERE username = '%{SQL-User-Name}' W
    ORDER BY id"
```

예) (비밀번호 + **일회용 인증키**)를 결합된 형태로 검증하는 경우

```
authorize_check_query = "W
    SELECT id, username, attribute, CONCAT(value, TO_VERIFYKEYL(username, phone_no,
cycle_time)) as value, op W
    FROM ${authcheck_table} W
    WHERE username = '%{SQL-User-Name}' W
    ORDER BY id"
```

④ 테스트할 사용자 생성

테스트 하기 위하여 다음과 같은 값을 Table에 추가한다.

```
MariaDB [(none)]> insert into radgroupreply (groupname,attribute,op,value) values ('baropam','Auth-
Type',':=','Local');
```

```
MariaDB [(none)]> insert into radcheck (username ,phone_no,cycle_time,attribute,op,value) values
('admin1','01027714076','60','Cleartext-Password',':=','admin1');
```

```
MariaDB [(none)]> insert into radusergroup (username,groupname) values ('admin1', 'baropam');
```

2) FreeRADIUS 디버그 모드로 실행

새 ssh 세션을 열고 디버그 모드에서 **radiusd**를 다음과 같이 실행한다.

```
[root@localhost ~]# radiusd -X
```

디버그 모드에서 **radiusd**를 실행 시 다음과 같은 오류가 발생하면

Then run radiusd -X but I have this error:

```
Failed reading private key file /etc/raddb/certs/server.pem:error:06065064:digital envelope
routines:EVP_DecryptFinal_ex:bad decrypt
rlm_eap_tls: Failed initializing SSL context
rlm_eap (EAP): Failed to initialise rlm_eap_tls
/etc/raddb/mods-enabled/eap[17]: Instantiation failed for module "eap"
```

다음과 같은 작업을 진행한 후 **radiusd**를 다음과 같이 실행한다.

```
[root@localhost ~]# cd /etc/raddb/certs
[root@localhost ~]# ./bootstrap
```

첫 번째 ssh 세션으로 전환하고 테스트한다.

형식)

```
radtest <username> (<password><verification code>) localhost 0 baropam

or

radtest <username> (<verification code>) localhost 0 baropam
```

BaroPAM으로 FreeRADIUS 테스트를 다음과 같이 실행한다.

예) **일회용 인증키**를 검증하는 경우

```
[root] /usr/baropam > radtest admin1 969533 localhost 0 baropam
Sent Access-Request Id 66 from 0.0.0.0:36066 to 127.0.0.1:1812 length 76
  User-Name = "admin1"
  User-Password = "969533"
  NAS-IP-Address = 10.0.2.15
  NAS-Port = 0
  Message-Authenticator = 0x00
  Cleartext-Password = "969533"
Received Access-Accept Id 66 from 127.0.0.1:1812 to 127.0.0.1:36066 length 20
```

예) (비밀번호 + 일회용 인증키)를 결합된 형태로 검증하는 경우

```
[root] /usr/baropam > radtest admin1 admin1969533 localhost 0 baropam
Sent Access-Request Id 66 from 0.0.0.0:36066 to 127.0.0.1:1812 length 76
  User-Name = "admin1"
  User-Password = "admin1969533"
  NAS-IP-Address = 10.0.2.15
  NAS-Port = 0
  Message-Authenticator = 0x00
  Cleartext-Password = "admin1969533"
Received Access-Accept Id 66 from 127.0.0.1:1812 to 127.0.0.1:36066 length 20
```

3) FreeRADIUS 데몬 실행

FreeRADIUS 데몬인 **radiusd**를 다음과 같이 백그라운드로 실행한다.

```
[root@localhost ~]# radiusd -s &
[1] 1961
```

FreeRADIUS 데몬인 **radiusd**가 정상적으로 기동되었는지 다음과 같이 확인한다.

```
[root@localhost ~]# ps -ef|grep radiusd
root      1961   1818  0 14:11 pts/1    00:00:00 radiusd -s
root      1964   1818  0 14:11 pts/1    00:00:00 grep --color=auto radiusd
```

FreeRADIUS 데몬인 **radiusd**가 사용하는 UDP 포트인 1812를 다음과 같이 확인한다.

```
[root@localhost ~]# netstat -an | grep 1812
udp        0      0 127.0.0.1:18120      0.0.0.0:*
udp        0      0 0.0.0.0:1812        0.0.0.0:*
udp6       0      0 :::1812              :::*
```

FreeRADIUS를 연동하여 **BaroPAM**에서 인증한 로그는 다음과 같이 확인한다.

```
$ tail -f /var/log/secure
Mar 26 13:54:11 localhost radiusd(pam_baro_auth)[1857]: Try to update RATE_LIMIT line.[3 30
1616734451]
Mar 26 13:56:46 localhost radiusd(pam_baro_auth)[1857]: Try to update RATE_LIMIT line.[3 30
1616734606]
Mar 26 14:00:48 localhost radiusd(pam_baro_auth)[1934]: Try to update RATE_LIMIT line.[3 30
1616734848]
Mar 26 14:00:48 localhost radiusd(pam_baro_auth)[1934]: Invalid verification code
Mar 26 14:00:48 localhost radiusd[1934]: pam_unix(radiusd:auth): authentication failure;
```

```
logname=root uid=0 euid=0 tty= ruser= rhost= user=scjool
Mar 26 14:01:13 localhost radiusd(pam_baro_auth)[1934]: Try to update RATE_LIMIT line.[3 30
1616734873]
Mar 26 14:01:36 localhost radiusd(pam_baro_auth)[1934]: Try to update RATE_LIMIT line.[3 30
1616734873 1616734896]
Mar 26 14:02:15 localhost radiusd(pam_baro_auth)[1934]: Try to update RATE_LIMIT line.[3 30
1616734935]
```

4) FreeRADIUS 데몬 종료

먼저 FreeRADIUS 데몬인 **radiusd**가 실행되었는지 다음과 같이 확인한다.

```
[root@localhost ~]# ps -ef|grep radiusd
root      1961    1818  0 14:11 pts/1    00:00:00 radiusd -s
root      1964    1818  0 14:11 pts/1    00:00:00 grep --color=auto radiusd
```

FreeRADIUS 데몬인 **radiusd**의 프로세스 ID(1961)를 확인한 후 다음과 같이 종료한다.

```
[root@localhost ~]# kill -9 1961
```

참고) FreeRADIUS 관련 로그 파일: /var/log/radius/radius.log

6. About BaroPAM



Version 1.0 – Official Release – 2016.12.1
Copyright © Nurit corp. All rights reserved.
<http://www.nurit.co.kr>

제 조 사 : 주식회사 누리아이티
등록번호 : 258-87-00901
대표이사 : 이종일
대표전화 : 02-2665-0119(영업문의/기술지원)
이 메 일 : mc529@nurit.co.kr
주 소 : 서울시 강서구 마곡중앙2로 15, 913호(마곡동, 마곡테크노타워2)