

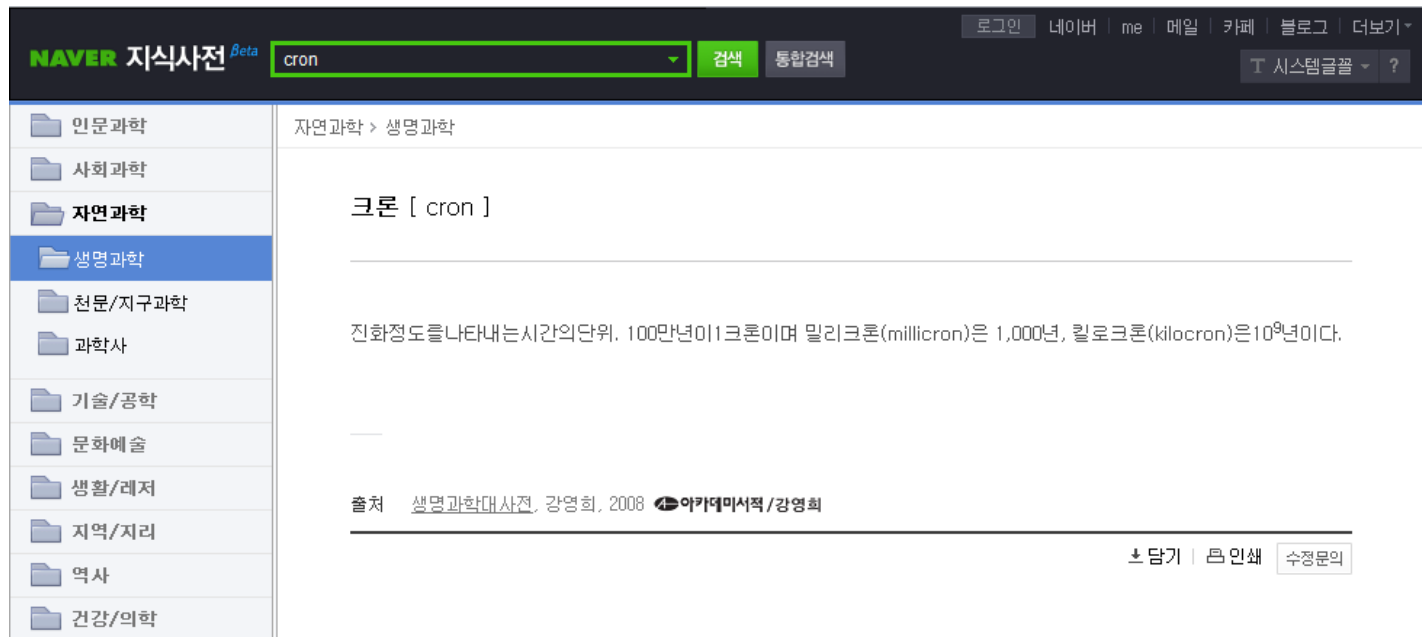
7

기타
시스템
관리

- Cron
 - crond
 - crontab
- System Logs
 - syslogd
 - logrotate
 - logcheck
 - fail2ban

기타
시스템
관리

cron?



The screenshot shows the Naver Knowledge Encyclopedia interface. At the top, there is a search bar with the text 'cron' and a search button. The page is categorized under '자연과학 > 생명과학' (Natural Science > Life Science). The main content area displays the title '크론 [cron]' followed by a definition: '진화정도를 나타내는 시간의 단위. 100만년이 1크론이며 밀리크론(millicron)은 1,000년, 킬로크론(kilocron)은 10⁹년이다.' Below the definition, there is a citation: '출처 생명과학대사전, 강영희, 2008 아카데미서적/강영희'. At the bottom right, there are buttons for '다운로드' (Download), '인쇄' (Print), and '수정문의' (Edit Article).

로그인 | 네이버 | me | 메일 | 카페 | 블로그 | 더보기 ▾

NAVER 지식사전 ^{Beta} cron 검색 통합검색


T 시스템글꼴 ▾ ?

인문과학 | 사회과학 | **자연과학** | 생명과학 | 천문/지구과학 | 과학사 | 기술/공학 | 문화예술 | 생활/레저 | 지역/지리 | 역사 | 건강/의학

자연과학 > 생명과학

크론 [cron]

진화정도를 나타내는 시간의 단위. 100만년이 1크론이며 밀리크론(millicron)은 1,000년, 킬로크론(kilocron)은 10⁹년이다.

출처 [생명과학대사전](#), 강영희, 2008  아카데미서적/강영희

↓ 담기 | 인쇄 수정문의

cron

- 주어진 일정에 따라 명령어를 실행하는 작업 스케줄러 데몬
 - 새벽 5시마다 디스크를 백업하려면?
- Multi-user runlevel에 들어가면서 init에서 실행
- 시스템과 각 계정의 명령어 실행 일정을 읽어서 메모리에 불러온다.

근데 어디서?

crontab

- 어디에서 일정을 읽어올까?
 - 시스템크론 설정파일: /etc/crontab
 - 사용자크론 설정파일: /var/spool/cron/[username]
- 명령어 스케줄이 들어 있는 이런 파일을 crontab(cron table)이라 한다.
- 시스템크론 설정디렉토리
 - /etc/cron.hourly
 - /etc/cron.daily
 - /etc/cron.weekly
 - /etc/cron.monthly
- Debian, Redhat에서는 /etc/cron.d 디렉토리 내에 있는 파일도 crontab으로 취급한다.

crond

- cron 데몬은 매 분마다 crontab의 각 명령어가 실행될 시간인지 확인한다.
- 실행된 명령어의 output, error는 crontab의 owner에게 메일로 보내진다.
- 또한 crontab 파일의 수정시간을 확인하여 변경된 경우 리로드한다.
 - 변경 후 데몬을 새로 시작할 필요가 없다
- 실행확인: `ps -ef | grep crond`
- 수동으로 시작/종료/재시작하려면:
 - `/etc/rc.d/init.d/crond start`
 - `/etc/rc.d/init.d/crond stop`
 - `/etc/rc.d/init.d/crond restart`

user cron

- 개별 사용자가 /var/spool/cron/[username] 파일에 자신의 crontab 설정
- 직접 수정 X
 - root 권한이 없으면 폴더에 접근하지도 못할 수 있다
- crontab 커맨드 사용
 - 자신의 crontab 내용 보기: crontab -l
 - 자신의 crontab 파일 삭제: crontab -r
 - 자신의 crontab 파일 수정: crontab -e
- root는 일반사용자의 crontab을 변경할 수 있다
 - crontab -u [username] [-l / -r / -e]
- 사용자 cron에서 실행되는 명령어의 output, error는 사용자의 메일로 보내진다.

crontab file format

- 빈 행, 행 시작의 스페이스와 탭 무시
- 각 행은 주석, 환경변수 설정, 크론 명령어 중 하나

- 주석(comment)
 - #으로 시작
 - 행의 중간에 주석을 입력할 수 없다

crontab file format

```
cling@sparcs: ~  
cling@sparcs:~$ cat /etc/crontab  
# /etc/crontab: system-wide crontab  
# Unlike any other crontab you don't have to run the `crontab`  
# command to install the new version when you edit this file  
# and files in /etc/cron.d. These files also have username fields,  
# that none of the other crontabs do.  
  
SHELL=/bin/sh  
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin  
  
# m h dom mon dow user  command  
17 * * * * root    cd / && run-parts --report /etc/cron.hourly  
25 6 * * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily  
47 6 * * 7 root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly  
52 6 1 * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly  
#  
  
*/5 * * * * root    /SPARCS/bin/edalias_gather.py  
#10 0 * * * root    /usr/sbin/ntpdate 0.kr.pool.ntp.org  
cling@sparcs:~$
```

} 주석 (comment)

} 환경변수설정

} 크론 명령어

crontab file format

- 환경 변수 설정(environment setting)
 - name = value
 - 등호 양 옆을 제외한 빈 칸은 value로 들어간다
 - value 앞에 빈 칸을 넣으려면 큰/작은 따옴표로 감싼다
 - value는 환경변수로 치환되지 않는다
 - PATH = \$HOME/bin:\$PATH와 같은 사용 불가능
- cron에 의해 자동적으로 설정되는 환경변수: SHELL, LOGNAME, HOME, PATH, MAILTO, EDITOR...
 - SHELL은 기본적으로 /bin/sh로 설정되어 있다 (bourne shell)
 - bash에서 홈 디렉토리를 나타내는 ~ 사용 불가능, \$HOME으로 대체
 - EDITOR는 vim으로 설정되어 있다
 - LOGNAME은 crontab의 소유주를 나타내며 나머지 변수값은 변경 가능
 - MAILTO = root / MAILTO = noname@gmail.com

crontab file format

- 크론 명령어 (cron command)
 - 수행시간을 나타내는 5-6개의 필드와 명령어
 - 현재시간이 모든 시간 필드를 만족할 때 명령어가 수행된다
 - sparcs 서버 crontab에는 유저도 쓰게 되어 있다

필드 이름	필수 여부	허용되는 값	특수 문자
분 (minute)	Yes	0-59	* / , -
시 (hour)	Yes	0-23	* / , -
날짜 (day)	Yes	1-31	* / , - ? L W
월 (month)	Yes	1-12 / JAN-DEC	* / , -
요일 (weekday)	Yes	0-7 / SUN-SAT	* / , - ? L #
년도 (year)	No	1970-2099	* / , -

- 별표(*): 모든 값 / 콤마 (,): 값 구분

crontab file format

cron command	결과
* * * * * [command]	매분마다 실행
17 * * * * [command]	매시 17분마다 실행
25 6 * * * [command]	매일 6시 25분마다 실행
09,39 * * * * [command]	매시 9분, 39분마다 실행
0 23-7 * * * [command]	23시부터 7시까지 매시 실행
0 23-7/2 * * * [command]	23시~7시까지 2시간마다 실행
*/5 * * * * [command]	매 5분마다 실행

crontab file format

string	meaning
@reboot	시작 시 실행
@yearly	1년마다 (0 0 1 1 *)
@annually	= @yearly
@monthly	한달마다 (0 0 1 * *)
@daily	한주마다 (0 0 * * 0)
@midnight	= @daily
@hourly	한시간마다 (0 * * * *)

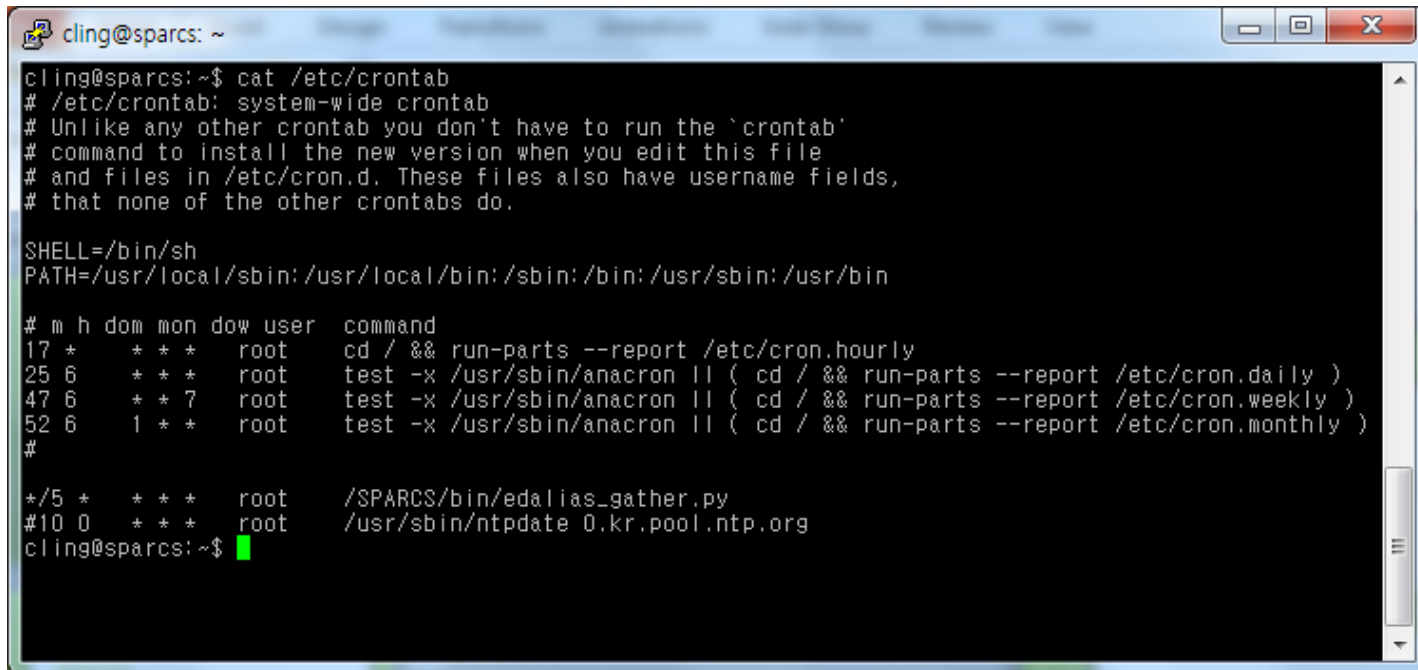
crontab file format

- Redirection을 이용하여 output이나 error를 메일 대신 파일에 연결할 수 있다
 - * * * * * /sbin/ping -c 1 192.168.0.1 > /dev/null
 - output은 무시되고 error만 메일로 보내진다
 - * 0 1 */2 * /sbin/ping -c 192.168.0.1; ls -la >> /var/log/cronrun
 - 두달마다 ping과 ls를 수행해서 결과를 /var/log/cronrun에 저장한다
 - * 0 1 */2 /sbin/ping -c 192.168.0.1; ls -la >> /var/log/cronrun 2>&1
 - output과 error 모두 /var/log/cronrun에 저장한다
- 실행할 내용이 긴 경우 쉘 스크립트를 이용할 수 있다

crontab permission

- /etc/cron.allow 파일이 있는 경우
 - 파일에 등록된 사용자만 crontab 사용 가능 (root도 마찬가지)
- /etc/cron.deny 파일이 있는 경우
 - 파일에 등록된 사용자는 crontab 사용 불가능
- /etc/cron.allow, deny 둘 다 존재할 경우
 - 둘 다 등록된 사용자는 crontab 사용 가능
 - 둘 다 등록되지 않은 사용자도 crontab 사용 가능
- 초기에는 존재하지 않으므로 필요할 때 생성
- 행마다 ID 하나씩 적으면 됨
- echo ALL >>/etc/cron.deny

cron.hourly, daily, weekly...

A terminal window titled 'cling@sparcs: ~' showing the output of the command 'cat /etc/crontab'. The output displays the system-wide crontab configuration, including comments, environment variables (SHELL and PATH), and several cron jobs for hourly, daily, weekly, and monthly tasks, as well as two specific jobs for gathering aliases and updating ntpdate.

```
cling@sparcs:~$ cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 * * * * root    cd / && run-parts --report /etc/cron.hourly
25 6 * * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * 7 root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
#

*/5 * * * * root    /SPARCS/bin/edalias_gather.py
#10 0 * * * root    /usr/sbin/ntpdate 0.kr.pool.ntp.org
cling@sparcs:~$
```



anacron

- 시스템이 지속적으로 동작하지 않더라도 일 단위(혹은 며칠, 몇 주, 몇 달 단위)의 작업을 하도록 만들어졌다.
- 사용되지 않는 파일의 삭제 등 리눅스의 집안일
- 보통 cron에서 새벽 등 시스템 사용이 적은 시간에 수행하도록 되어있지만 그 시간에 시스템이 꺼져있다면?
- anacron을 이용하면 개인 컴퓨터에서도 정해진 간격마다 꼭 작업을 수행할 수 있다.
- 그러니까 스팍스서버에는 없다....

cron practice

- 매 2분마다 "Hello World!"를 출력하고 결과는 자신의 홈 디렉토리의 log 파일에 기록되도록 해 보자

log?

log¹ 미국식 [lɔːg, lɑːg]  영국식 [lɔg]  ★★

1. 통나무
2. 동작이 느린 사람; 무감각한 것, 움직임이 없는 것
3. 항해 일지; 여행 일기; (엔진·보일러 등의) 공정(工程) 일지; (실험 등의) 기록

System Logs

- 로그: 시스템이 부팅할 때부터 종료할 때까지 작동중인 모든 상황을 별도의 파일로 만드는 것
- 로그 파일: 이러한 기록이 저장되는 파일
 - 로그 파일을 분석해서 시스템의 각종 환경과 작업 상태 등을 확인하고 검사할 수 있다
 - 시스템에 이상이 생겼을 때, 해킹을 당했을 때 1차적인 확인을 로그파일에서 하게 된다

System Logs

- syslogd 데몬이 부팅 시마다 실행되어 시스템의 각종 활동을 기록한다
- 로그파일의 기본 위치: /var/log/
 - lastlog 각 유저의 마지막 로그인 정보. lastlog 명령어로 확인한다.
 - messages 운영체제에서 보내지는 실시간 메시지
 - utmp 시스템에 현재 로그인한 유저들의 정보. who, w, finger 커맨드가 사용.
 - wtmp 모든 로그인/로그아웃과 시스템 재부팅을 기록. last 커맨드에서 사용
 - dmesg 시스템 부팅시 나왔던 메시지들을 기록
 - boot 부트 메시지
 - cron 크론 데몬 로그파일
- /etc/syslog.conf 파일의 설정에 따라 각 내용을 기록할 위치가 결정된다.

syslog.conf

- /etc/syslog.conf
 - 주석을 제외한 각 행이 "selector"와 "action" 필드로 구성되어 있다.
 - "selector"는 다시 facilities와 level(priorities)의 값으로 구성되어 있다.
 - 메시지 종류 (selector) 기록될 장소 (action)

*.info; *.notice	/var/log/messages
mail.debug	/var/log/maillog
*.warn	/var/log/syslog
kern.emerg	/dev/console
 - 내용을 변경한 후에는 syslogd에 HUP(hang up) 시그널을 보내서 설정파일을 다시 읽도록 한다

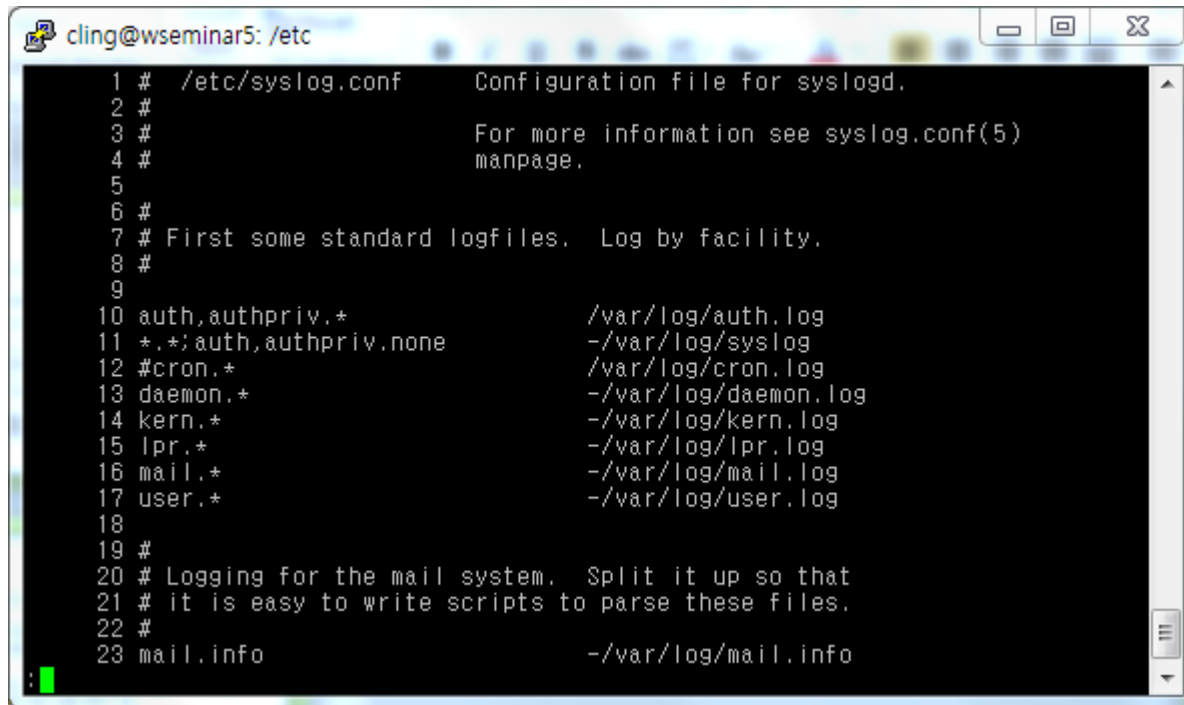
syslog.conf

- 메시지 종류 (selector)
 - facility.level [;facility.level ...]
- facility: 메시지를 생성하는 애플리케이션/기능
 - auth login이나 su와 같은 인증 프로그램
 - authpriv 개인인증을 요구하는 프로그램
 - cron cron이나 at과 같은 프로그램
 - mail 메일시스템
 - news 유즈넷 뉴스 프로그램
 - user 사용자 프로세스
 - * 모든 facility
 - daemon(데몬), kern(커널), syslog....

syslog.conf

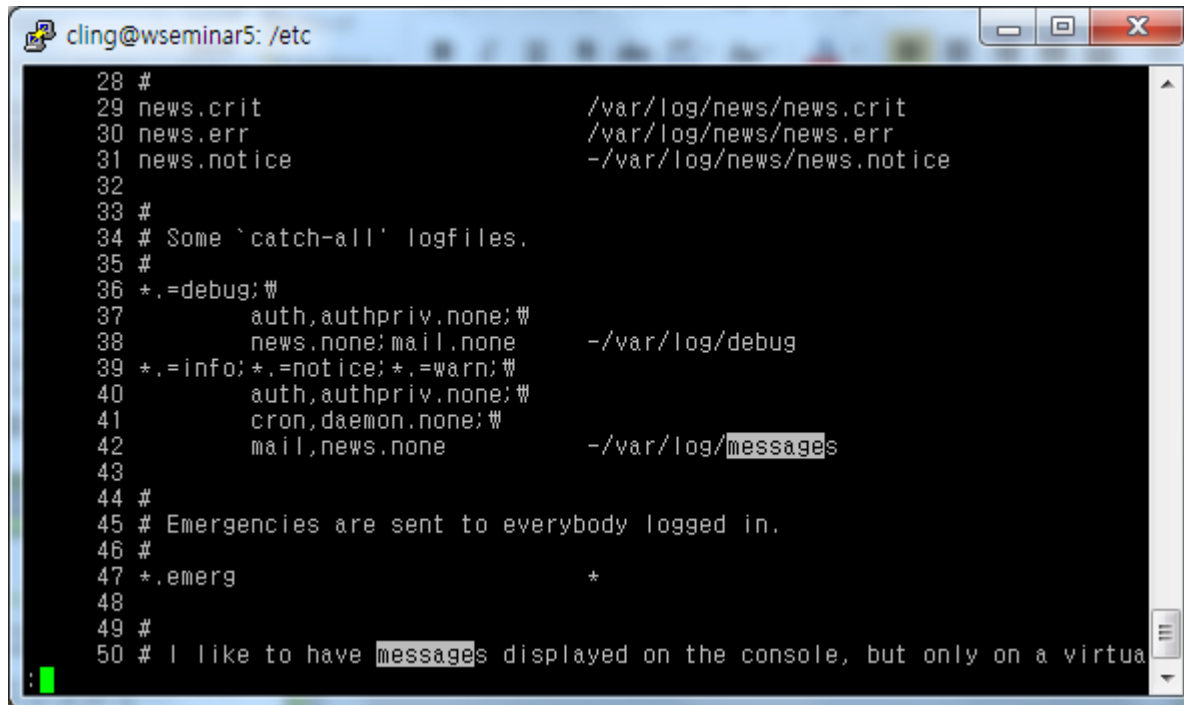
- level: 메시지의 등급
 - 위험 레벨이 낮은 순으로 debug, info, notice, warning, err, crit, alert, emerg
 - debug - 프로그램을 디버깅할 때 발생하는 메시지
 - info - 통계나 기본 정보 메시지
 - notice - 에러는 아니지만 특별한 주의가 필요한 메시지
 - warning - 주의를 요하는 경고 메시지
 - err, error - 에러가 발생하는 경우 메시지
 - crit - 크게 급하지는 않지만 시스템에 문제가 생기는 단계의 메시지
 - alert - 즉각적인 조치가 필요한 상황
 - emerg - 모든 사용자에게 전달되어야 하는 위험한 상황
 - facility.none: 해당 facility를 제외하겠다는 의미

syslog.conf

A terminal window titled 'cling@wseminar5: /etc' displays the configuration file /etc/syslog.conf. The window has standard Linux window controls (minimize, maximize, close) in the top right corner. The text is as follows:

```
1 # /etc/syslog.conf      Configuration file for syslogd.
2 #
3 #                       For more information see syslog.conf(5)
4 #                       manpage.
5
6 #
7 # First some standard logfiles.  Log by facility.
8 #
9
10 auth,authpriv.*        /var/log/auth.log
11 *.*:auth,authpriv.none -/var/log/syslog
12 #cron.*                 /var/log/cron.log
13 daemon.*                -/var/log/daemon.log
14 kern.*                  -/var/log/kern.log
15 lpr.*                   -/var/log/lpr.log
16 mail.*                  -/var/log/mail.log
17 user.*                  -/var/log/user.log
18
19 #
20 # Logging for the mail system.  Split it up so that
21 # it is easy to write scripts to parse these files.
22 #
23 mail.info               -/var/log/mail.info
```

syslog.conf



```
cling@wseminar5: /etc
28 #
29 news.crit          /var/log/news/news.crit
30 news.err           /var/log/news/news.err
31 news.notice       -/var/log/news/news.notice
32
33 #
34 # Some 'catch-all' logfiles.
35 #
36 *,=debug;#
37     auth,authpriv.none;#
38     news.none;mail.none    -/var/log/debug
39 *,=info;*,=notice;*,=warn;#
40     auth,authpriv.none;#
41     cron,daemon.none;#
42     mail,news.none        -/var/log/messages
43
44 #
45 # Emergencies are sent to everybody logged in.
46 #
47 *,=emerg           *
48
49 #
50 # I like to have messages displayed on the console, but only on a virtua
```

Managing System Logs

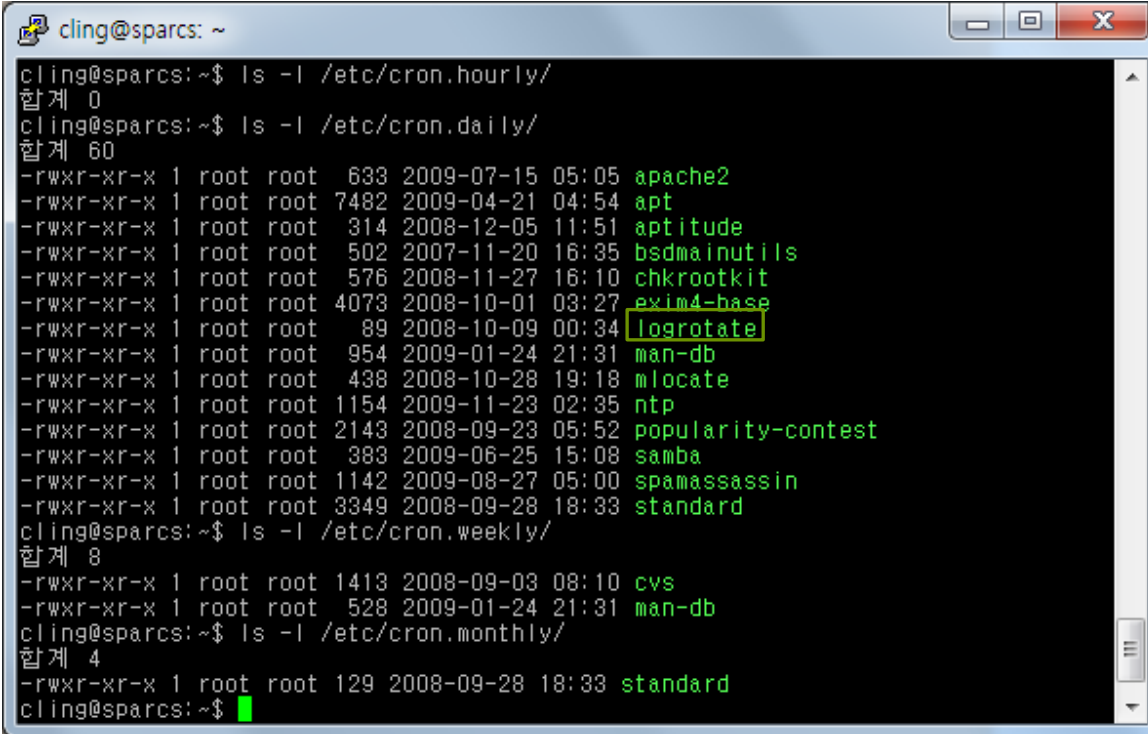
- 로그파일이 쌓이면 용량이 커지며, 디스크 용량이 부족하면 파티션이 너무 빨리 찰 수도 있다
- 무작정 지우기에는 위기 상황에 로그파일이 유용할 수 있다
-> 백업해보자!
- 로그파일을 복사해서 압축하고 비워주면 된다

```
mv /var/log/messages /var/log/messages-backup
cp /dev/null /var/log/messages
gzip /var/log/messages-backup
```
- cron으로 반복 수행하기?
 - 백업파일이 같은 파일명으로 계속 덮어쓰기 될 것이다

Log rotation

- 돌려막기
- 압축파일이 몇 개 이상 (예: 10개) 생기면 가장 오래된 압축파일을 삭제하고 새 압축파일로 덮어쓴다
- 시스템에 따라 savelog, logrotate와 같은 쉘 스크립트가 미리 작성되어 있다
 - 스팍스 서버에서는 logrotate를 사용하여 관리중
- `ls /var/log`

logrotate



```
cling@sparcs: ~  
cling@sparcs:~$ ls -l /etc/cron.hourly/  
합계 0  
cling@sparcs:~$ ls -l /etc/cron.daily/  
합계 60  
-rwxr-xr-x 1 root root 633 2009-07-15 05:05 apache2  
-rwxr-xr-x 1 root root 7482 2009-04-21 04:54 apt  
-rwxr-xr-x 1 root root 314 2008-12-05 11:51 aptitude  
-rwxr-xr-x 1 root root 502 2007-11-20 16:35 bsdmainutils  
-rwxr-xr-x 1 root root 576 2008-11-27 16:10 chkrootkit  
-rwxr-xr-x 1 root root 4073 2008-10-01 03:27 exim4-base  
-rwxr-xr-x 1 root root 89 2008-10-09 00:34 logrotate  
-rwxr-xr-x 1 root root 954 2009-01-24 21:31 man-db  
-rwxr-xr-x 1 root root 438 2008-10-28 19:18 mlocate  
-rwxr-xr-x 1 root root 1154 2009-11-23 02:35 ntp  
-rwxr-xr-x 1 root root 2143 2008-09-23 05:52 popularity-contest  
-rwxr-xr-x 1 root root 383 2009-06-25 15:08 samba  
-rwxr-xr-x 1 root root 1142 2009-08-27 05:00 spamassassin  
-rwxr-xr-x 1 root root 3349 2008-09-28 18:33 standard  
cling@sparcs:~$ ls -l /etc/cron.weekly/  
합계 8  
-rwxr-xr-x 1 root root 1413 2008-09-03 08:10 cvs  
-rwxr-xr-x 1 root root 528 2009-01-24 21:31 man-db  
cling@sparcs:~$ ls -l /etc/cron.monthly/  
합계 4  
-rwxr-xr-x 1 root root 129 2008-09-28 18:33 standard  
cling@sparcs:~$
```

logrotate



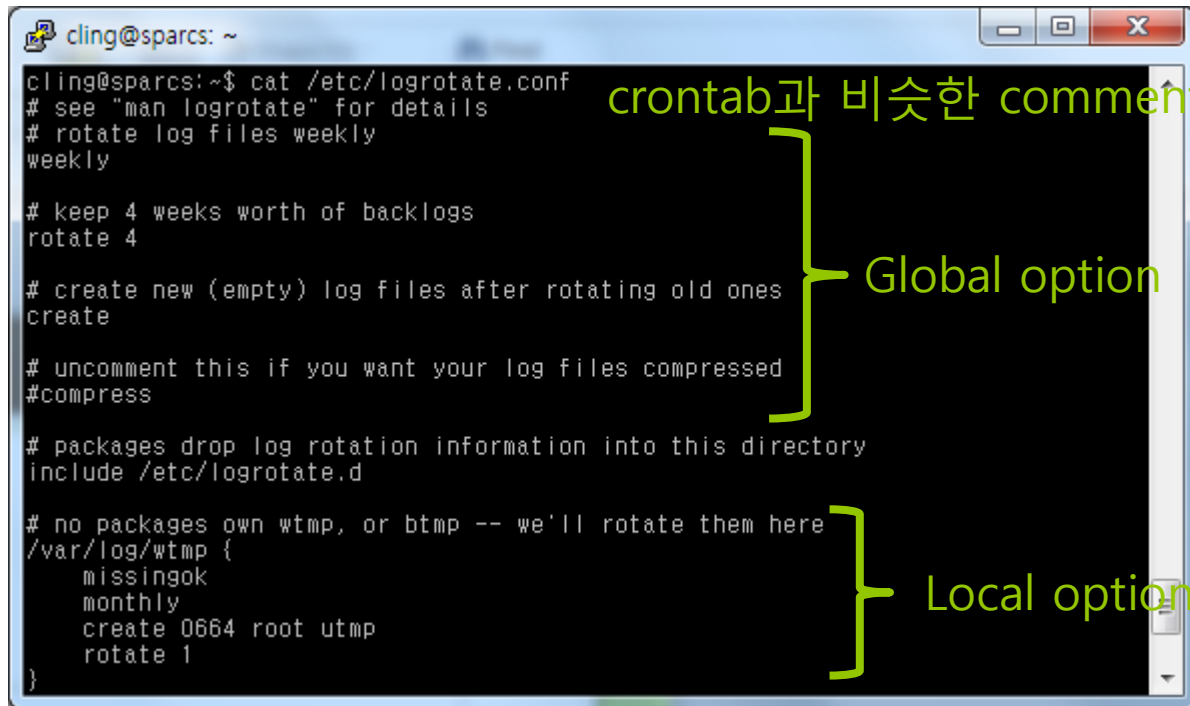
```
cling@sparcs: ~  
cling@sparcs:~$ cat /etc/cron.daily/logrotate  
#!/bin/sh  
  
test -x /usr/sbin/logrotate || exit 0  
/usr/sbin/logrotate /etc/logrotate.conf  
cling@sparcs:~$
```

- test -x FILE: 파일이 존재하고 실행가능한지 검사
 - logrotate 프로그램이 존재하고 실행가능하지 않으면 스크립트 종료
- /etc/logrotate.conf를 argument로 해서 /usr/sbin/logrotate 프로그램 실행
- logrotate.conf: 환경설정파일

logrotate

- logrotate는 보통 cron으로 날마다 실행되도록 설정됨
 - 로그의 로테이션, 압축, 제거, 메일링을 자동화
 - 각 로그파일을 일, 주, 월, 혹은 크기에 따라 관리 가능
- 임의 갯수의 환경설정 파일을 argument로 전달.
 - 뒷쪽 파일이 앞쪽 파일의 설정을 덮어쓰게 된다.
 - 보통 다른 설정파일들을 포함(include)하는 설정파일 하나, /etc/logrotate.conf 를 쓴다
- ㅋ

logrotate configuration file



```
cling@sparcs: ~  
cling@sparcs:~$ cat /etc/logrotate.conf  
# see "man logrotate" for details  
# rotate log files weekly  
weekly  
  
# keep 4 weeks worth of backlogs  
rotate 4  
  
# create new (empty) log files after rotating old ones  
create  
  
# uncomment this if you want your log files compressed  
#compress  
  
# packages drop log rotation information into this directory  
include /etc/logrotate.d  
  
# no packages own wtmp, or btmp -- we'll rotate them here  
/var/log/wtmp {  
    missingok  
    monthly  
    create 0664 root utmp  
    rotate 1  
}
```

crontab과 비슷한 comment

Global option

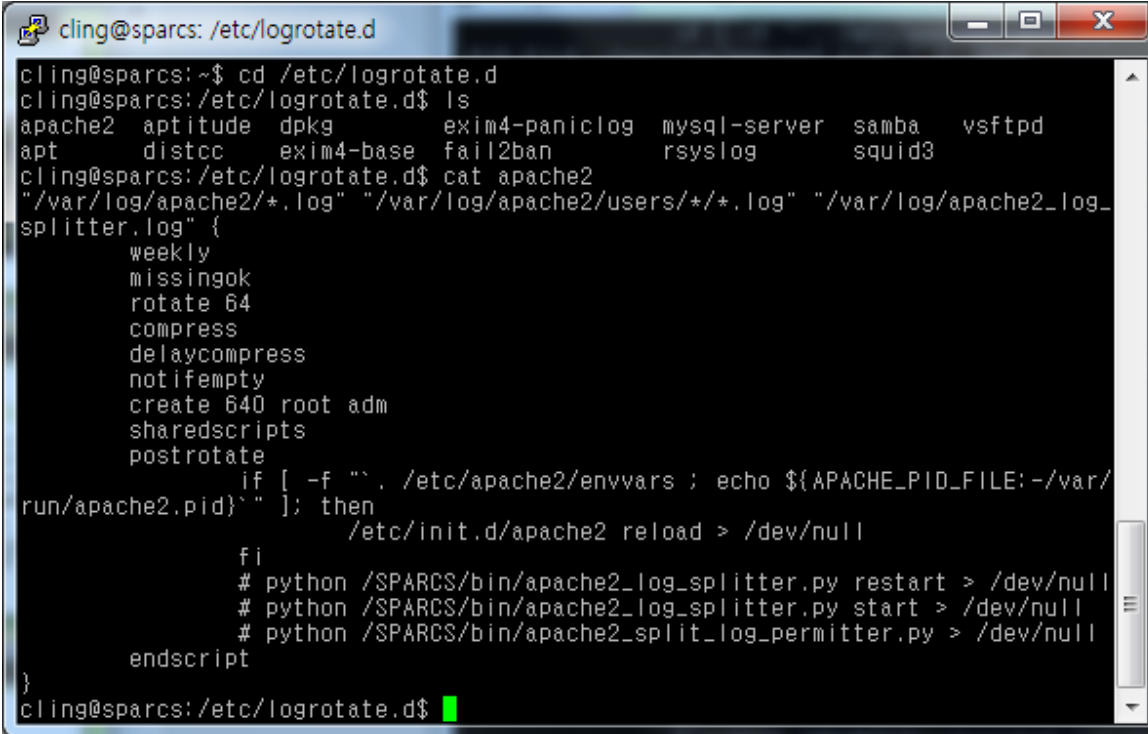
Local option

logrotate configuration file

```
cling@sparcs: ~  
cling@sparcs:~$ cat /etc/logrotate.conf  
# see "man logrotate" for details  
# rotate log files weekly  
weekly  
  
# keep 4 weeks worth of backlogs  
rotate 4  
  
# create new (empty) log files after rotating old ones  
create  
  
# uncomment this if you want your log files compressed  
#compress  
  
# packages drop log rotation information into this directory  
include /etc/logrotate.d  
  
# no packages own wtmp, or btmp -- we'll rotate them here  
/var/log/wtmp {  
    missingok  
    monthly  
    create 0664 root utmp  
    rotate 1  
}  
  
/var/log/btmp {  
    missingok  
    monthly  
    create 0660 root utmp  
    rotate 1  
}  
  
# system-specific logs may be configured here  
cling@sparcs:~$
```

- weekly/monthly/hourly: 주/달/시간마다 로테이션을 한다
- rotate 4: 삭제하기 전 4번의 로테이션을 거친다
- create: 로테이션 후 새로운 로그파일 생성
- compress: 로그파일을 압축한다. 일반적으로 로그가 너무 크지 않으면 압축하지 않는다
- include /etc/logrotate.d
- missingok: 로그 파일이 없어도 무시하고 에러 메시지를 출력하지 않는다
- create mode owner group

logrotate configuration file



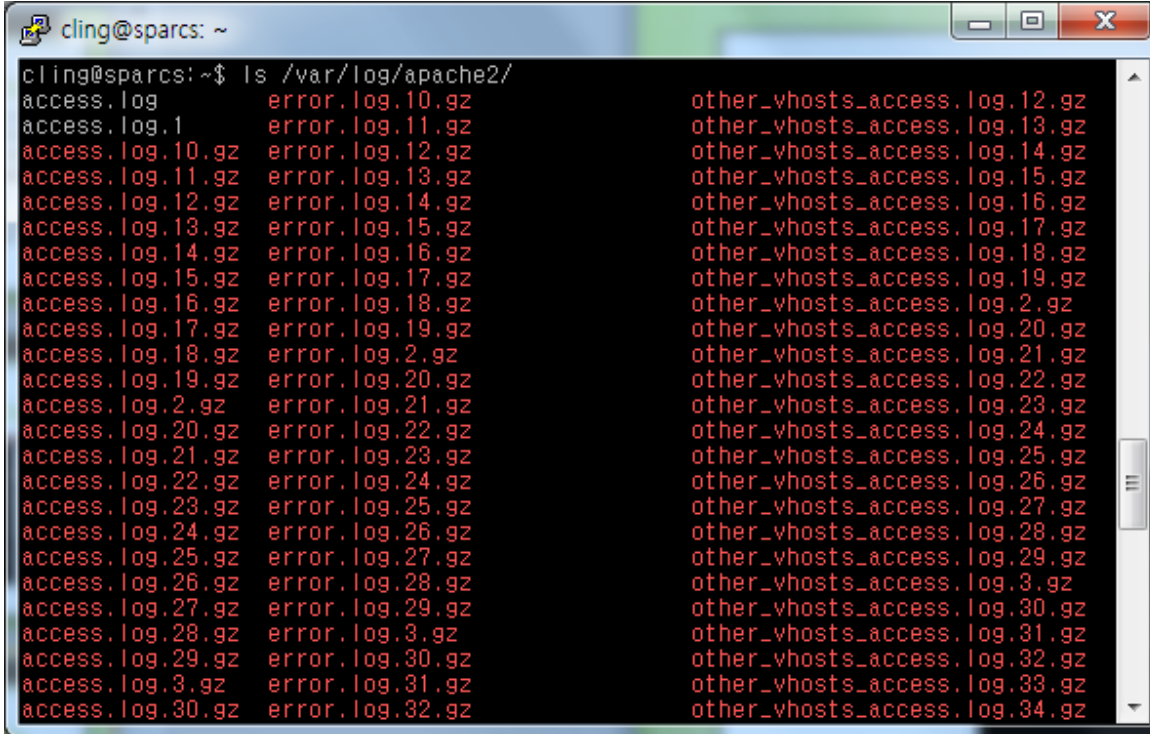
```
cling@sparcs: /etc/logrotate.d
cling@sparcs:~$ cd /etc/logrotate.d
cling@sparcs:/etc/logrotate.d$ ls
apache2  aptitude  dpkg          exim4-paniclog  mysql-server  samba  vsftpd
apt      distcc    exim4-base   fail2ban        rsyslog      squid3
cling@sparcs:/etc/logrotate.d$ cat apache2
"/var/log/apache2/*.log" "/var/log/apache2/users/*/*.log" "/var/log/apache2_log_
splitter.log" {
    weekly
    missingok
    rotate 64
    compress
    delaycompress
    notifempty
    create 640 root adm
    sharedscripts
    postrotate
        if [ -f "/etc/apache2/envvars" ; echo "${APACHE_PID_FILE:-/var/
run/apache2.pid}" ]; then
            /etc/init.d/apache2 reload > /dev/null
        fi
        # python /SPARCS/bin/apache2_log_splitter.py restart > /dev/null
        # python /SPARCS/bin/apache2_log_splitter.py start > /dev/null
        # python /SPARCS/bin/apache2_split_log_permmitter.py > /dev/null
    endscript
}
cling@sparcs:/etc/logrotate.d$
```

logrotate configuration file

```
"/var/log/apache2/*.log"
splitter.log" {
    weekly
    missingok
    rotate 64
    compress
    delaycompress
    notifempty
    create 640 root a
    sharedscripts
    postrotate
        if [ -f "
run/apache2.pid" ]; the
        fi
        # python
        # python
        # python
    endscript
}
```

- notifempty
- delaycompress: 로그 파일의 압축을 다음 로테이션까지 미룬다
- postrotate/endscript 사이에 있는 스크립트를 로테이션 후에 실행한다
 - prerotate/endscript
- sharedscripts
 - 이렇게 *를 사용한 경우 prerotate, postrotate가 각 파일에 대해서 실행되므로 여러번 실행되게 되는데 한 번만 실행되게 한다.
- mail alphamin@gmail.com: 로테이션이 끝나서 폐기되는 내용을 미스평창에게 메일로 보낸다
- size (100/100k/100M/1G): 로그파일 크기에 따라 로테이션

logrotate configuration file

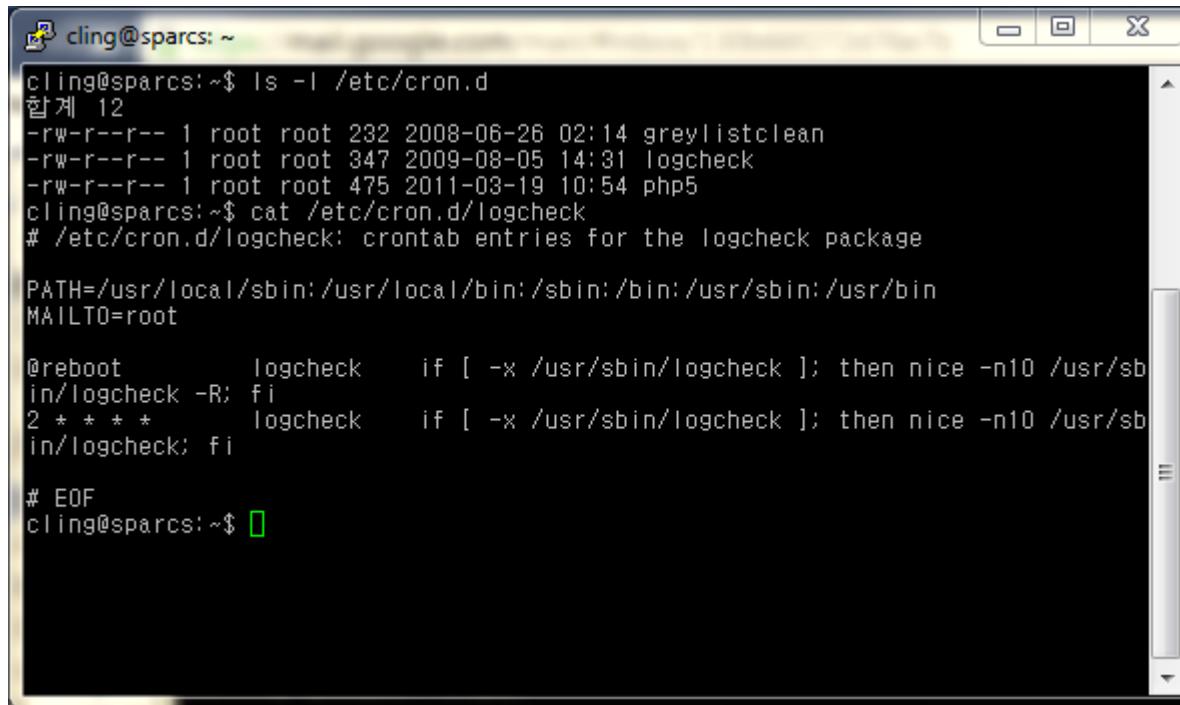


```
cling@sparcs: ~  
cling@sparcs:~$ ls /var/log/apache2/  
access.log          error.log.10.gz    other_vhosts_access.log.12.gz  
access.log.1       error.log.11.gz    other_vhosts_access.log.13.gz  
access.log.10.gz   error.log.12.gz    other_vhosts_access.log.14.gz  
access.log.11.gz   error.log.13.gz    other_vhosts_access.log.15.gz  
access.log.12.gz   error.log.14.gz    other_vhosts_access.log.16.gz  
access.log.13.gz   error.log.15.gz    other_vhosts_access.log.17.gz  
access.log.14.gz   error.log.16.gz    other_vhosts_access.log.18.gz  
access.log.15.gz   error.log.17.gz    other_vhosts_access.log.19.gz  
access.log.16.gz   error.log.18.gz    other_vhosts_access.log.2.gz  
access.log.17.gz   error.log.19.gz    other_vhosts_access.log.20.gz  
access.log.18.gz   error.log.2.gz     other_vhosts_access.log.21.gz  
access.log.19.gz   error.log.20.gz    other_vhosts_access.log.22.gz  
access.log.2.gz    error.log.21.gz    other_vhosts_access.log.23.gz  
access.log.20.gz   error.log.22.gz    other_vhosts_access.log.24.gz  
access.log.21.gz   error.log.23.gz    other_vhosts_access.log.25.gz  
access.log.22.gz   error.log.24.gz    other_vhosts_access.log.26.gz  
access.log.23.gz   error.log.25.gz    other_vhosts_access.log.27.gz  
access.log.24.gz   error.log.26.gz    other_vhosts_access.log.28.gz  
access.log.25.gz   error.log.27.gz    other_vhosts_access.log.29.gz  
access.log.26.gz   error.log.28.gz    other_vhosts_access.log.3.gz  
access.log.27.gz   error.log.29.gz    other_vhosts_access.log.30.gz  
access.log.28.gz   error.log.3.gz     other_vhosts_access.log.31.gz  
access.log.29.gz   error.log.30.gz    other_vhosts_access.log.32.gz  
access.log.3.gz    error.log.31.gz    other_vhosts_access.log.33.gz  
access.log.30.gz   error.log.32.gz    other_vhosts_access.log.34.gz
```

logcheck

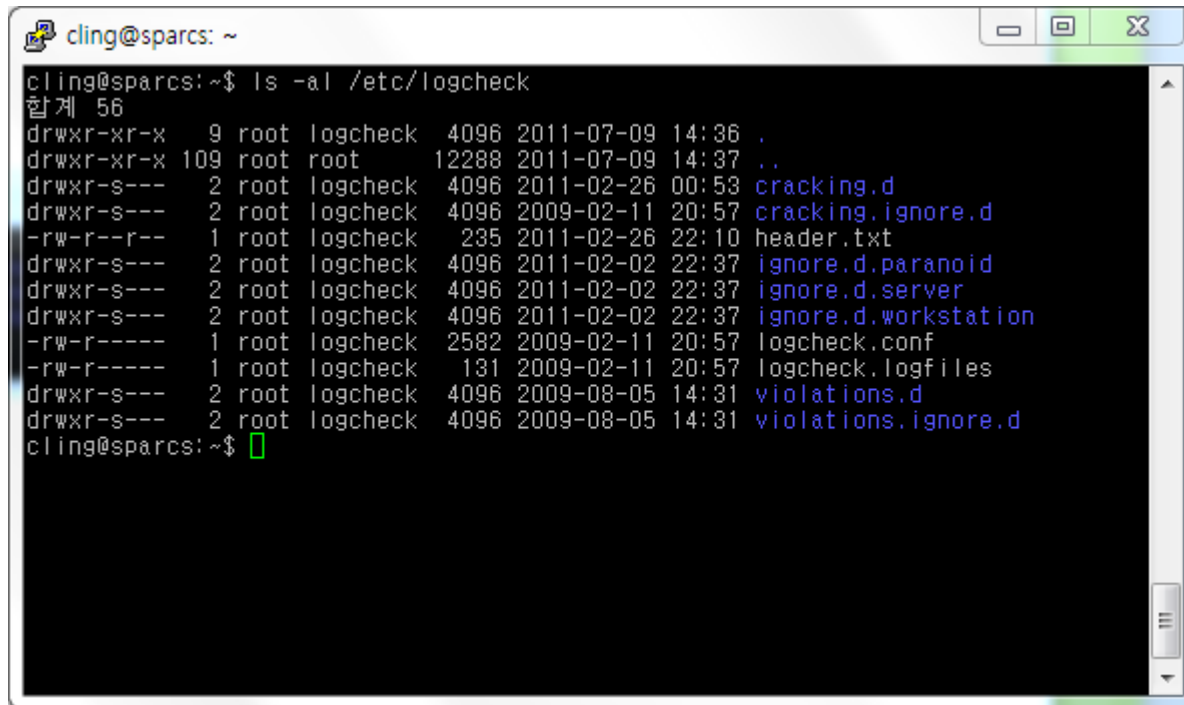
- 로그 파일이 너무 길고 쓸데없는 내용이 많은 것 같다. 좀 중요한 내용만 편하게 볼 수 없을까요?
- 시스템 로그를 분석해서 이를 설정한 사용자에게 메일로 알려주는 쉘 스크립트
 - 매번 시스템에 접속하지 않아도 보안 위반사항이나 비정상적인 행위를 확인할 수 있다!
- 기본적으로 시스템 부팅시, 그리고 매 시간마다 cron에서 실행
 - 중복된 검사를 피하기 위해 로그파일 중 마지막에 읽은 위치를 기억해주는 logtail이라는 프로그램을 사용한다

logcheck



```
cling@sparcs: ~  
cling@sparcs:~$ ls -l /etc/cron.d  
합계 12  
-rw-r--r-- 1 root root 232 2008-06-26 02:14 greylistclean  
-rw-r--r-- 1 root root 347 2009-08-05 14:31 logcheck  
-rw-r--r-- 1 root root 475 2011-03-19 10:54 php5  
cling@sparcs:~$ cat /etc/cron.d/logcheck  
# /etc/cron.d/logcheck: crontab entries for the logcheck package  
  
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin  
MAILTO=root  
  
@reboot          logcheck    if [ -x /usr/sbin/logcheck ]; then nice -n10 /usr/sb  
in/logcheck -R; fi  
2 * * * *      logcheck    if [ -x /usr/sbin/logcheck ]; then nice -n10 /usr/sb  
in/logcheck; fi  
  
# EOF  
cling@sparcs:~$ █
```

/etc/logcheck



```
cling@sparcs: ~  
cling@sparcs:~$ ls -al /etc/logcheck  
합계 56  
drwxr-xr-x  9 root  logcheck  4096 2011-07-09 14:36 .  
drwxr-xr-x 109 root  root    12288 2011-07-09 14:37 ..  
drwxr-s---  2 root  logcheck  4096 2011-02-26 00:53 cracking.d  
drwxr-s---  2 root  logcheck  4096 2009-02-11 20:57 cracking.ignore.d  
-rw-r--r--  1 root  logcheck  235 2011-02-26 22:10 header.txt  
drwxr-s---  2 root  logcheck  4096 2011-02-02 22:37 ignore.d.paranoid  
drwxr-s---  2 root  logcheck  4096 2011-02-02 22:37 ignore.d.server  
drwxr-s---  2 root  logcheck  4096 2011-02-02 22:37 ignore.d.workstation  
-rw-r----- 1 root  logcheck  2582 2009-02-11 20:57 logcheck.conf  
-rw-r----- 1 root  logcheck  131 2009-02-11 20:57 logcheck.logfiles  
drwxr-s---  2 root  logcheck  4096 2009-08-05 14:31 violations.d  
drwxr-s---  2 root  logcheck  4096 2009-08-05 14:31 violations.ignore.d  
cling@sparcs:~$
```

logcheck

- etc.logfiles 파일에는 검사할 로그파일의 목록이 들어있다
- 이 로그파일들을 egrep을 이용하여 필터링한 후 메일로 보내준다.
- 필터링은 세 단계로 나뉜다
 - SECURITY ALERTS: 침입 시도 흔적을 검사하는 레이어
 - cracking.d – 검사에 사용될 regular expression 패턴이 적힌 파일이 들어있다
 - cracking.ignore.d – 필터링한 결과 중 이 디렉토리 내에 있는 패턴과 맞는 것은 잘못된 필터링으로 간주하고 무시한다.
 - SECURITY EVENTS: 덜 치명적이지만 주의가 필요한 사건
 - violations.d
 - violations.ignore.d
 - SYSTEM EVENTS: 나머지 로그 메시지
 - cracking.d나 violations.d에 해당하는 파일 없이 모든 나머지 로그가 SYSTEM EVENTS에 포함될 대상으로 고려된다. (ignore된 결과는 포함하지 않는다)
 - ignore.d.paranoid / ignore.d.server / ignore.d.workstation

logcheck

- REPORTLEVEL – SYSTEM EVENTS를 필터링하는 강도
 - paranoid
 - 소수의 서비스를 제공하는 고보안 시스템에 적합
 - ignore하는 패턴이 가장 적다
 - 수많은 메시지를 다 볼 수 있을 때 사용할 것
 - server
 - 기본적인 필터링 레벨
 - 여러 데몬에 대한 ignore 패턴이 있다
 - workstation
 - 안전한 시스템에 사용. 대부분의 메시지를 필터링한다
- logcheck.conf – 기본 설정파일
 - REPORTLEVEL="server"
 - SENDMAILTO="root"

fail2ban

- /var/log/pwdfail이나 /var/log/apache/error_log 같은 로그 파일을 읽어서 패스워드를 너무 많이 틀리는 IP를 차단한다.



```
cling@sparcs: ~  
cling@sparcs:~$ ls /etc/fail2ban  
action.d fail2ban.conf filter.d jail.conf  
cling@sparcs:~$
```

- filter.d – 로그인 실패 등에 해당하는 regular expression 패턴이 들어있는 디렉토리
- action.d – 상황에 따라 실행할 여러 명령어가 들어있다
- jail.conf – 각 필터와 액션을 이어주는 설정파일

Reference

- [http://www.utm.edu/organizations/tsa/Books/\(OReilly\)%20Running%20Linux,%204th%20Edition.pdf](http://www.utm.edu/organizations/tsa/Books/(OReilly)%20Running%20Linux,%204th%20Edition.pdf)
- <http://www.wikipedia.org/>
- <http://www.pantz.org/software/cron/croninfo.html>
- <http://suya55.tistory.com/32>
- <http://lupusmaru.egloos.com/1090823>
- <http://www.fis.unipr.it/pub/linux/redhat/9/en/doc/RH-DOCS/rhl-cg-ko-9/s1-autotasks-anacron.html>

- http://www.joinc.co.kr/modules/moniwiki/wiki.php/Site/system_programing/Unix_Env/syslog_1
- <http://chyeon.tistory.com/entry/Linux-System-Log-1>
- <http://knamhun.blogspot.com/2008/04/linux-varlog-information.html>
- http://www.fail2ban.org/wiki/index.php/Main_Page
- Linux Man Page