

Linux – Perf

:Linux profiling with performance counters

Kosslab-kr/linux-perf

멘토 : 송태웅

-팀원-

김동현 김선영

김재훈 김종빈

권욱제 안이수

양낙영 조성수

최왕용 하남봉

INDEX

I. 목표

II. 최종 성과물

III. 진행 과정

I. 목표

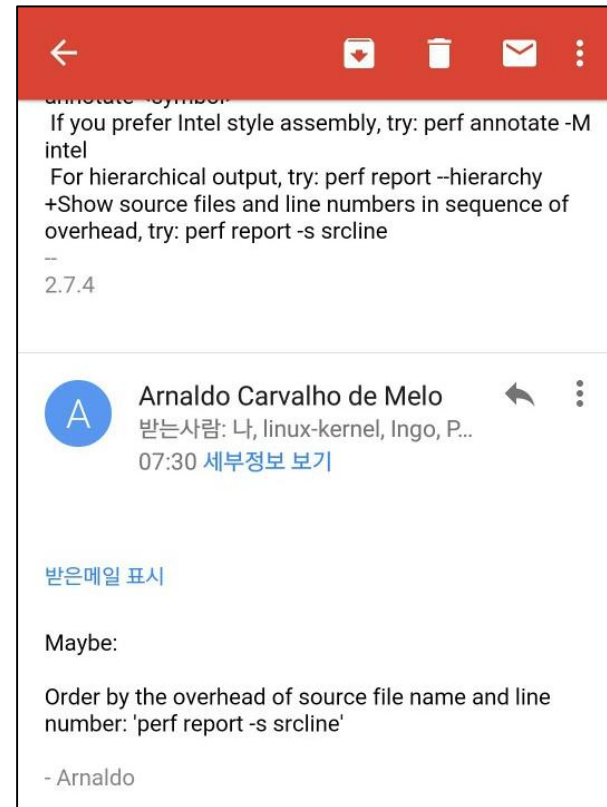
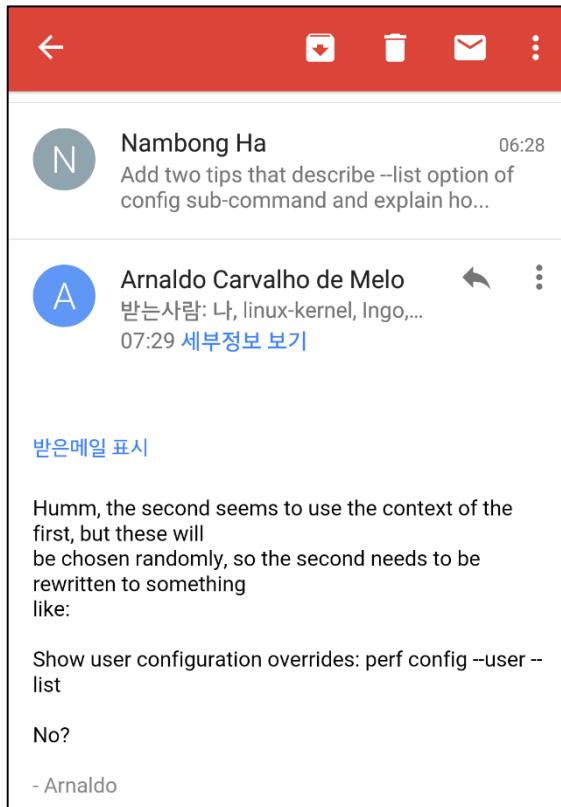
- 이 프로젝트의 가장 큰 목적은 리눅스 커널 Contribution 을 하기위한 큰 장벽을 허무는데 있습니다.
- git 을 이용해서 수정한 내용에 대해 패치를 만들고 그것을 메일링 리스트에 전송하거나, github 에 pull request 를 보내는 것을 학습하여 리눅스 커널이 개발되는 과정을 학습합니다.
- 해커톤이 끝난 이후에도 지속적으로 perf 툴의 개발 진행상황에 관심을 가지고 지속적인 contribution 을 할 수 있기를 희망합니다.

II. 최종 성과물

- 5개의 커널 패치 전송 (2개 답변 완료, 적용 대기 중)
 - Perf 사용을 위한 Tip
 - Perf diff 관련 내용
 - Perf config 관련 permission 버그 수정
- Perf stat diff 아이디어 제시(진행 중)
- 도커를 활용한 perf 개발 테스트 환경 구축 작업
- Perf 초보 개발자를 위한 필요 라이브러리 설치 및 컴파일 자동 스크립트
- perf 초보 개발자를 위한 가이드 문서

II. 최종 성과물 - 5개의 커널 패치 전송

- 5개의 커널 패치 전송
 - Perf 사용을 위한 Tip - 답변 완료, 적용 대기 중



II. 최종 성과물 - 5개의 커널 패치 전송

- 5개의 커널 패치 전송
 - Perf diff 관련 내용

```

nexus29@nexus: /tmp/work/linux-perf/tools/perf$ git send-email --to "Arnaldo Carvalho de Melo <acme@kernel.org>" --cc "Peter Zijlstra <peterz@infradead.org>" 0001-perf-diff-introduce-the-new-rules-of-colored-printin.patch
(mbox) Adding cc: SeongSoo Cho <nexus29@gmail.com> from line 'From: SeongSoo Cho <nexus29@gmail.com>'
(body) Adding cc: SeongSoo Cho <nexus29@gmail.com> from line 'Signed-off-by: SeongSoo Cho <nexus29@gmail.com>'
(body) Adding cc: Namhyung Kim <namhyung@kernel.org> from line 'Cc: Namhyung Kim <namhyung@kernel.org>'
(body) Adding cc: Jiri Olsa <jolsa@kernel.org> from line 'Cc: Jiri Olsa <jolsa@kernel.org>'
(body) Adding cc: Taeung Song <taeung@kosslab.kr> from line 'Cc: Taeung Song <taeung@kosslab.kr>'

From: SeongSoo Cho <nexus29@gmail.com>
To: Arnaldo Carvalho de Melo <acme@kernel.org>
Cc: linux-kernel@vger.kernel.org,
    Ingo Molnar <mingo@kernel.org>,
    Peter Zijlstra <peterz@infradead.org>,
    SeongSoo Cho <nexus29@gmail.com>,
    Namhyung Kim <namhyung@kernel.org>,
    Jiri Olsa <jolsa@kernel.org>,
    Taeung Song <taeung@kosslab.kr>
Subject: [PATCH RESEND] perf diff: Introduce the new rules of colored printing of delta.
Date: Fri, 30 Sep 2016 08:44:19 +0900
Message-Id: <1475192659-27164-1-git-send-email-nexus29@gmail.com>
X-Mailer: git-send-email 2.7.4

Send this email? (Y|es|N|o|A|ut|a|I|): y
Password for 'smtp://nexus29@gmail.com@smtp.gmail.com:587':
OK. Log says:
Server: smtp.gmail.com
MAIL FROM: <nexus29@gmail.com>
RCPT TO: <acme@kernel.org>
RCPT TO: <linux-kernel@vger.kernel.org>
RCPT TO: <mingo@kernel.org>
RCPT TO: <peterz@infradead.org>
RCPT TO: <nexus29@gmail.com>
RCPT TO: <namhyung@kernel.org>
RCPT TO: <jolsa@kernel.org>
RCPT TO: <taeung@kosslab.kr>
From: SeongSoo Cho <nexus29@gmail.com>
To: Arnaldo Carvalho de Melo <acme@kernel.org>
Cc: linux-kernel@vger.kernel.org,
    Ingo Molnar <mingo@kernel.org>,
    Peter Zijlstra <peterz@infradead.org>,
    SeongSoo Cho <nexus29@gmail.com>,
    Namhyung Kim <namhyung@kernel.org>,
    Jiri Olsa <jolsa@kernel.org>,
    Taeung Song <taeung@kosslab.kr>
Subject: [PATCH RESEND] perf diff: Introduce the new rules of colored printing of delta.
Date: Fri, 30 Sep 2016 08:44:19 +0900
Message-Id: <1475192659-27164-1-git-send-email-nexus29@gmail.com>
X-Mailer: git-send-email 2.7.4

Result: ZS0
nexus29@nexus: /tmp/work/linux-perf/tools/perf$

```

perf diff: Introduce the new rules of colored printing of delta.

As you know, there are the common colored printing of percents so overhead(%) can be colored with the rule. But Delta means difference percents from percents of overhead between two files e.g. perf.data and perf.data.old. Although the rule is for overhead(%), Delta value also follow the same rule.

So, I think that it would be better to use the new colored rule for the Delta as below.

Increment: background colored in red (e.g. +0.50%)
 Decrement: colored in blue (e.g. -5.50%)
 Same: default color (e.g. +0.00%)

Instead of percent_color_snprintf() function, use new delta_color_snprintf() function.

Signed-off-by: SeongSoo Cho <nexus29@gmail.com>
 Cc: Namhyung Kim <namhyung@kernel.org>
 Cc: Jiri Olsa <jolsa@kernel.org>
 Cc: Taeung Song <taeung@kosslab.kr>

nexus29 committed 19 hours ago commit 41c7681a9d147001d8405160c8a8d85858ec905

```

2 ===== tools/perf/builtin-diff.c
808 808 @@ -808,7 +808,7 @@ static int __http_color_compare(struct perf_hpp_fmt *fmt,
809 809 diff = compute_delta(hc, pair);
810 810
811 811     snprintf(fmt, 20, "%08.2f%%", dft->header_width - 1);
812 812 -    return percent_color_snprintf(hpp->buf, hpp->size,
813 813 +    return delta_color_snprintf(hpp->buf, hpp->size,
814 814         pfmt, diff);
815 815
816 816     case COMPUTE_RATIO:
817 817         if (hc->dummy)

```

```

21 ===== tools/perf/util/color.c
219 219 @@ -219,3 +219,24 @@ int percent_color_len_snprintf(char *bf, size_t size, const char *fmt, ...)
220 220     color = get_percent_color(percent);
221 221     return color_snprintf(bf, size, color, fmt, len, percent);
222 222
223 223 +
224 224 +int delta_color_snprintf(char *bf, size_t size, const char *fmt, ...)
225 225 +{
226 226 +    va_list args;
227 227 +    double diff, percent;
228 228 +    const char *color = PERF_COLOR_NORMAL;
229 229 +
230 230 +    va_start(args, fmt);
231 231 +    diff = va_arg(args, double);
232 232 +    va_end(args);
233 233 +
234 234 +    /* diff command printed second digit after the decimal point. */
235 235 +    percent = round(diff * 100) / 100;
236 236 +    if (percent < 0)
237 237 +        color = PERF_COLOR_BLUE;
238 238 +    else {
239 239 +        if (percent > 0)
240 240 +            color = PERF_COLOR_BG_RED;
241 241 +        return color_snprintf(bf, size, color, fmt, diff);
242 242 +    }

```

```

1 ===== tools/perf/util/color.h
40 40 @@ -40,6 +40,7 @@ int value_color_snprintf(char *bf, size_t size, const char *fmt, double value);
41 41 int percent_color_snprintf(char *bf, size_t size, const char *fmt, ...);
42 42 int percent_color_len_snprintf(char *bf, size_t size, const char *fmt, ...);

```

II. 최종 성과물 - 5개의 커널 패치 전송

- 5개의 커널 패치 전송
 - Perf config 관련 permission 버그 수정

perf config: Fix a bug about permission checking state of config file

perf_config_set__init() check state of user config file before opening it. But there is a bug when checking uid and euid of current user. Although current user have superuser permission, a error occurs as below.

Before:

```
user01@localhost:~$ ls -l ~/.perfconfig
-rw-rw-r-- 1 user01 user01 89 2016-09-30 01:52 /home/user01/.perfconfig
```

```
user01@localhost:~/linux-perf/tools/perf/util$ sudo perf config --list
Warning: File /home/user01/.perfconfig not owned by current user or root, ignoring it.
Warning: File /home/user01/.perfconfig not owned by current user or root, ignoring it.
```

So, Fix it allowing a user who have superuser permission to open user config file.

After:

```
user01@localhost:~$ ls -l ~/.perfconfig
-rw-rw-r-- 1 user01 user01 89 2016-09-30 01:52 /home/user01/.perfconfig
```

```
user01@localhost:~$ sudo perf config --list
annotate.hide_src_code=false
report.queue-size=0
tui.report=on
colors.top=red, default
```

Cc: Taeung Song <taeung@kosslab.kr>
Cc: Namhyung Kim <namhyung@kernel.org>
Cc: Jiri Olsa <jolsa@kernel.org>
Signed-off-by: Wookje Kwon <aweeee@gmail.com>

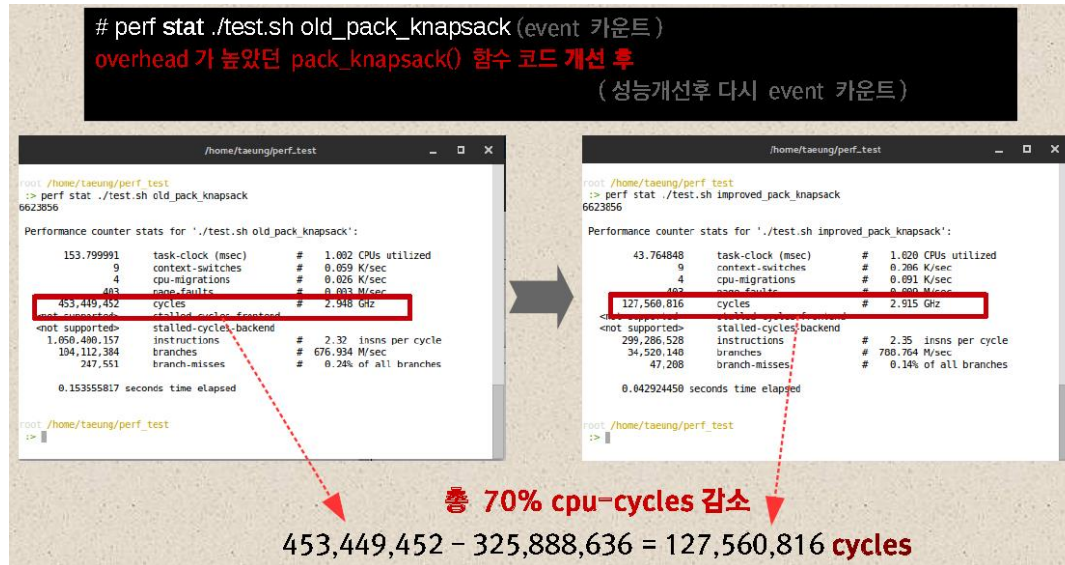


wooooook committed 18 hours ago

commit 08ad8a6b2

II. 최종 성과물

- Perf stat diff 아이디어 제시(진행 중)



Performance counter stats for './perf stat diff perf.data perf.data.old':

perf.data	perf.data.old	delta			
168.595515	46.177321	-122.418194	+72%	task-clock (msec)	#
52	21	-31	+59%	context-switches	#
4	5	1	-25%	cpu-migrations	#
401	404	3	0.0%	page-faults	#
449,585,944	121,963,831	-327,622,113	+72%	cycles	#
1,058,043,486	298,496,345	-759,547,141	+71%	instructions	#
103,968,811	34,390,673	-69,578,138	+66%	branches	#
245,160	40,873	-204,287	+83%	branch-misses	#

II. 최종 성과물

- 도커를 활용한 perf 개발 테스트 환경 구축 작업

```
[root@jouet ~]# cat ~/bin/dm
output=$(mktemp /tmp/dm.log.XXXXXX)
ln -sf ${output} /tmp/dm.log

for img in $(docker images | grep perf- | cut -d' ' -f1 ) ; do
    distro=${img:11}
    echo -n "$distro: "
    echo $distro >> ${output}
    docker run -v /home/acme/git:/git --privileged=true --rm=true -t -i $img >> ${output}
    [[ $? = 0 ]] && msg="Ok" || msg="FAIL"
    echo $msg
    if [ "$msg" == "FAIL" ] ; then
        tail $output
    fi
    echo "$distro: $msg" >> ${output}
done
[root@jouet ~]#
```

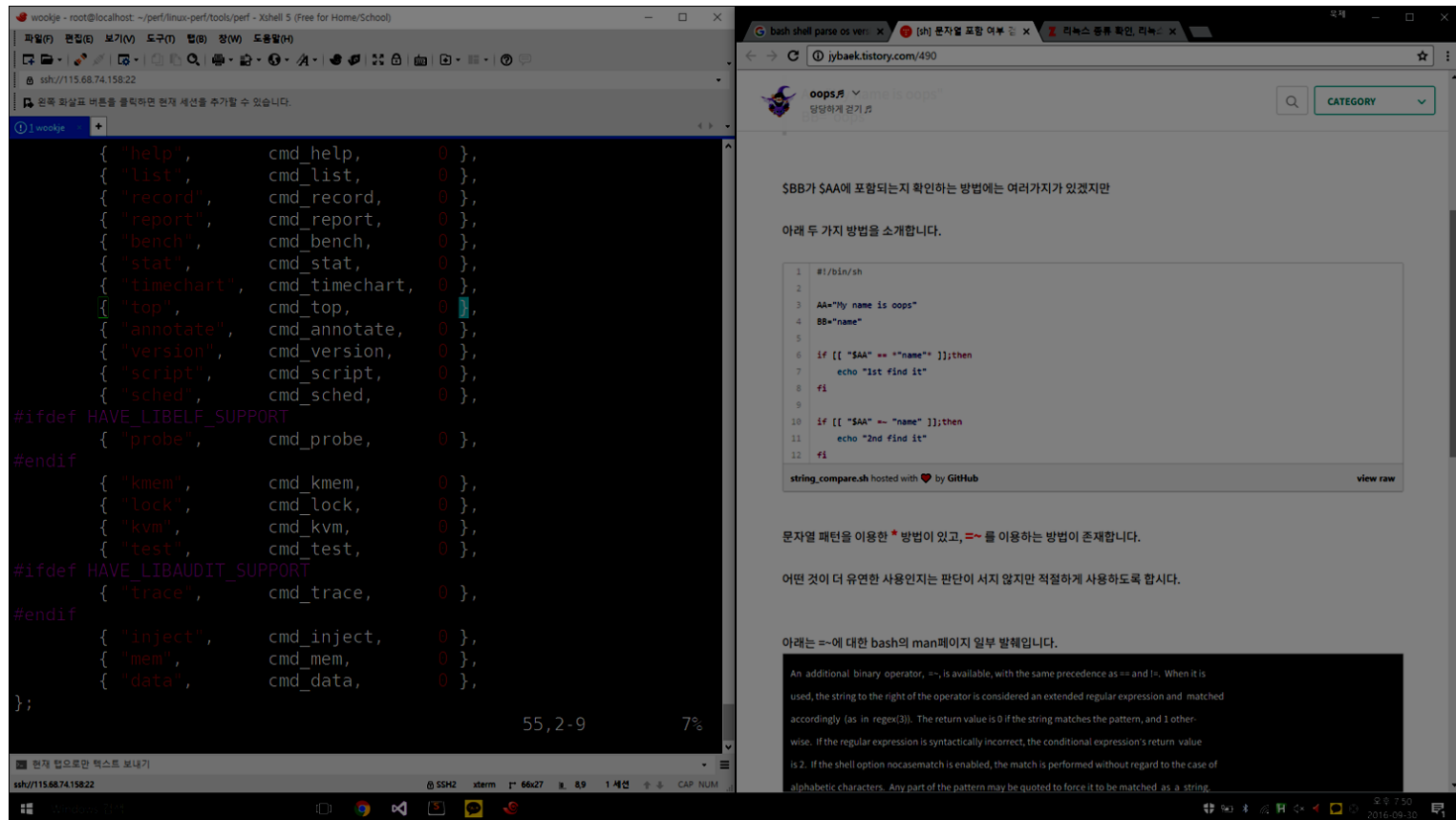
To use it I had to first build images for each distro I want to test. In my workstation I have these images built:

```
[root@jouet ~]# docker images | grep perf-
```

perf-build-alldeps-fedora-rawhide	latest	e57278404926	4 weeks ago	1.298 GB
perf-build-alldeps-ubuntu-14.04	latest	c36091c85ed1	5 weeks ago	632.9 MB
perf-build-alldeps-ubuntu-16.04	latest	72f6f5434e6d	5 weeks ago	669.3 MB
perf-build-alldeps-fedora-20	latest	7a8e670f7e51	7 weeks ago	944.2 MB
perf-build-alldeps-ubuntu-12.04	latest	f55b8c3182bf	8 weeks ago	508.7 MB
perf-build-minimal-debian-experimental-x-mips64	latest	9cd0c6b31282	9 weeks ago	595.3 MB
perf-build-minimal-debian-experimental-x-mips64el	latest	f37a0e73c696	9 weeks ago	595.4 MB
perf-build-minimal-debian-experimental-x-mipsel	latest	97aa2321eeea	9 weeks ago	589.4 MB
perf-build-minimal-ubuntu-x-arm	latest	a9450fa330a4	9 weeks ago	380.5 MB
perf-build-minimal-ubuntu-x-arm64	latest	c236781b734e	9 weeks ago	357.2 MB
perf-build-minimal-ubuntu-x-ppc64	latest	02adec2f8e15	9 weeks ago	384.3 MB
perf-build-minimal-ubuntu-x-ppc64el	latest	5b4e2e62a3f3	9 weeks ago	372.3 MB
perf-build-alldeps-debian	latest	aaf8db4ad122	11 weeks ago	678.7 MB

II. 최종 성과물


- Perf 초보 개발자를 위한 필요 라이브러리 설치 및 컴파일 자동 스크립트




II. 최종 성과물

- Perf 초보개발자를 위한 가이드 문서

초보자를 위한 linux perf contribution 가이드 #153

 nexusz99 opened this issue 14 hours ago · 0 comments



nexusz99 commented 14 hours ago · edited

1. 개발 환경 구축

linux kernel 코드 다운로드

linux kernel 코드는 <https://git.kernel.org> git 저장소 에서 관리되고 있습니다.

이 저장소에서 주요 커미터 별 그리고 linux kernel 안의 프로젝트 별로 저장소가 구분되어 있습니다.

우리가 contribution 할 프로젝트 'perf'는 커널 소스 안의 **tools/perf** 에 존재하는 프로젝트입니다. 이 프로젝트의 maintainer 는 'Arnaldo Carvalho de Melo' 이고, perf 툴의 모든 수정사항은 이 사람의 저장소에 가장 먼저 반영됩니다.


따라서, 우리는 'Arnaldo Carvalho de Melo'의 저장소를 clone 받아서 작업을 진행해야합니다.

아래 명령어로 저장소를 clone 받고, perf/core 브랜치를 checkout 합니다.

```
mkdir ~/work
cd work
git clone git://git.kernel.org/pub/scm/linux/kernel/git/acme/linux.git
git checkout perf/core
```

의존 라이브러리 설치

perf 를 컴파일하는데 필요한 라이브러리들이 있습니다. 아래 명령어를 실행하여 라이브러리를 설치합니다.

 debian 계열의 linux 를 기준의 명령어입니다. 다른 배포판의 경우 배포판에서 제공하는 패키지 설치 툴을 이용해서 라이브러리를 설치해주시기 바랍니다.

```
sudo apt-get install -y libdw-dev libelf-dev libnewt-dev libunwind8-dev elfutils libaudit-c
```

III. 진행 과정

- 마일스톤 관리(미션과 진행사항 표시)
 - 총 14개의 pull-request 연습 약 10회의 PATCH 메일 연습
 - 총 158개 issues(미션, 과제) 중 114개 완료
- 온라인, 오프라인 모임
 - Offline – 9.24(토), 9.26(월), 9.29(목), 9.30(금)
 - Online – Skype, 카카오톡

III. 진행 과정

- 마일스톤 관리(미션과 진행사항 표시)
 - 총 14개의 pull-request 연습 약 10회의 PATCH 메일 연습
 - 총 158개 issues(미션, 과제) 중 114개 완료

Labels Milestones New milestone

10 Open 0 Closed

Sort

김선영
▲ Past due by about 9 hours Last updated about 2 hours ago
KOSSHACKATHON 2016 진행상황
100% complete 0 open 13 closed
Edit Close Delete

하남봉
▲ Past due by about 9 hours Last updated about 3 hours ago
KOSSHACKATHON 2016 진행상황
100% complete 0 open 13 closed
Edit Close Delete

조성수
▲ Past due by about 9 hours Last updated about 4 hours ago
100% complete 0 open 14 closed
Edit Close Delete

최왕용
No due date Last updated about 7 hours ago
KOSSHACKATHON 2016 진행상황
100% complete 0 open 10 closed
Edit Close Delete

권옥제
▲ Past due by about 9 hours Last updated about 9 hours ago
100% complete 0 open 23 closed
Edit Close Delete

김동현
No due date Last updated about 11 hours ago
0% complete 0 open 0 closed
Edit Close Delete

안이수
▲ Past due by about 9 hours Last updated 1 day ago
80% complete 3 open 12 closed
Edit Close Delete

김재훈
▲ Past due by about 9 hours Last updated 3 days ago
KOSS HACKATHON 2016 진행상황
61% complete 5 open 8 closed
Edit Close Delete

송태웅(예시)
▲ Past due by about 9 hours Last updated 4 days ago
KOSSHACKATHON 2016 진행상황
0% complete 15 open 0 closed
Edit Close Delete

김중빈
No due date Last updated 4 days ago
57% complete 3 open 4 closed
Edit Close Delete

Unwatch 12 Star 7 Fork 10

Code Issues Pull requests 14 Projects Wiki Pulse Graphs

송태웅(예시)
Due by September 30, 2016 0% complete
KOSSHACKATHON 2016 진행상황

15 Open 0 Closed

perf 소스리딩 미션 "subcommand는 어떻게 실행되는가?" (9/28)
#91 opened 4 days ago by Taewung

perf 소스리딩 미션 "perf.data에서 이벤트 샘플링 정보 읽어서 evlister에 어떻게 세팅하는가?" (9/28)
#92 opened 4 days ago by Taewung

perf 소스리딩 tool 먹이기 미션 (9/27)
#90 opened 4 days ago by Taewung

perf의 Documentation/tips.txt 실제적용 및 추가러인 만들기 (9/27)
#89 opened 4 days ago by Taewung

perf config 명령에서 --list 옵션이 파싱되는 과정을 이해하자 (9/26)
#88 opened 4 days ago by Taewung

pull-request 하기 미션 (9/26)
#87 opened 4 days ago by Taewung

PATCH mail 보내기 미션 (Documentation/tips.txt 파일 수정) (9/26)
#86 opened 4 days ago by Taewung

(필수작업) 엔티 각자 Milestones 만들기
#17 opened 4 days ago by Taewung

perf 관련 링크 공유 (9/25)
#10 opened 4 days ago by Taewung

Git 기본 명령(add, commit, push) 학습 (9/25)
#11 opened 4 days ago by Taewung

perf 컴파일 및 개발환경 구축 (9/25)
#13 opened 4 days ago by Taewung

perf 실습 (QuickStart) (9/25)
#15 opened 4 days ago by Taewung

perf 기본실습 (9/25)
#16 opened 4 days ago by Taewung

perf 소스 중 config --list가 실행되는 과정분석 (9/25)
#14 opened 4 days ago by Taewung

Git 고급 명령(checkout, reset, rebase) 학습 (9/26)
#12 opened 4 days ago by Taewung

III. 진행 과정

- 마일스톤 관리(미션과 진행사항 표시)
 - 총 14개의 pull-request 연습 약 10회의 PATCH 메일 연습
 - 총 158개 issues(미션, 과제) 중 114개 완료

[권옥제] 'ctags - vi에서 함수의 내부로 점프하기' 정리

woooooo opened this issue 5 days ago · 2 comments

woooooo commented 5 days ago · edited

ctags 설치 및 사용법

ctags의 특징

- 소스코드 내의 함수나 변수를 인식시켜 해당 함수/변수가 선언된 위치로 이동할 수 있게 도와줌
- 함수와 변수를 인식(index)하는 유틸리티
- 간단한 설정으로 vi 내에서 사용 가능

ctags의 설치

설치확인

```
$ ctags -help
```

설치

```
$ sudo apt-get install ctags
```

Path 에 상관없이 perf 실행하기 #115

memnoth opened this issue 3 days ago · 0 comments



memnoth commented 3 days ago

매번 tools/perf 로 가서서 실행하는게 불편하다면 환경변수 폴더에 perf 바이너리 파일을 이동하여 실행하면 됩니다.
일단 perf 바이너리가 생성되고 tools/perf 로 이동했다는 가정하에 아래의 커맨드를 이용하여 이동시켜줍니다.

```
sudo mv ./perf /usr/local/bin
```

후에 홈 폴더로 이동하여 perf -version으로 실행이 되는지 확인하시면 됩니다.

nexusz99 commented 3 days ago · edited

builtin-config.c 에서 아래 부분에서 질문이 있습니다.

```
switch (actions) {
  case ACTION_LIST:
    if (argc) {
      pr_err("Error: takes no arguments\n");
      parse_options_usage(config_usage, config_options, "l", 1);
    } else {
      ret = show_config(set);
      if (ret < 0) {
```

if(argc) 에서 perf config --list a b 와 같이 --list 다음에 추가 argument 가 올 경우, 에러를 출력하도록 되어있는데, 굳이 추가 인자를 넣을 때, 이 옵션은 추가 인자가 없다고 오류를 줘야할 필요가 있는지 궁금합니다.

1

Taeung commented 3 days ago Korean Open Source Software Developers LAB member

말씀하신내용은 사실 문화(?) 관습(?) 적인 느낌이 좀 있는것 같습니다. 이렇게 해야 무조건 좋고 저렇게 하면 무조건 틀리다는 식은 아닌것 같습니다. 사실 perf 소스는 git 소스에서 그 틀을 많이 가져왔는데요. 저도 보여 주신 소스를 구현할때 git의 비슷함부분이 있는지 있다면 어떻게 했는지 등을 많이 참고 했습니다. git 같은경우도 아래와 같이

```
$ git config -l a
error: wrong number of arguments
usage: git config [<options>]

Config file location
--global          use global config file
--system          use system config file

...(선택)..
```

오류가 출력되는데, argument를 받지 않는 옵션이 argument가 들어오면 오류를 출력하는게 맞다고 저도 공감해서 그렇게 구현을 했습니다. (물론 반드시 굳이 그래야하는건 아니죠. 개발자의 판단에 따라 달라질 수 있다고 봅니다.) 그래서 아시는것 처럼 아래와 같이 오류메시지가 출력 됩니다.

```
$ perf config --list a
Error: takes no arguments

Usage: perf config [<file-option>] [options]

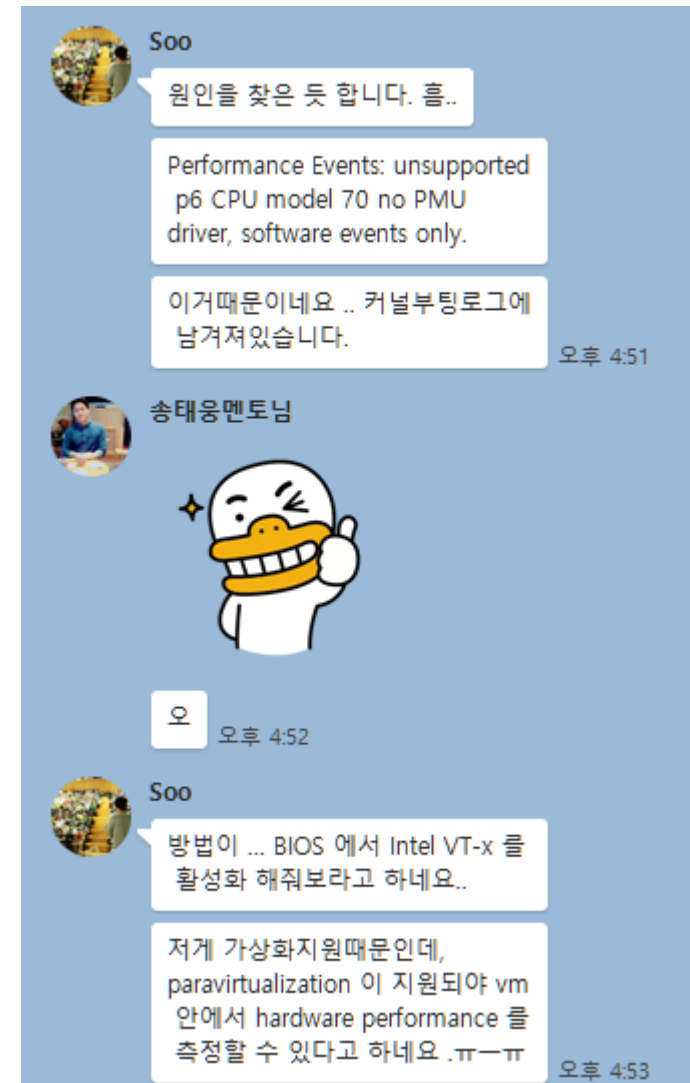
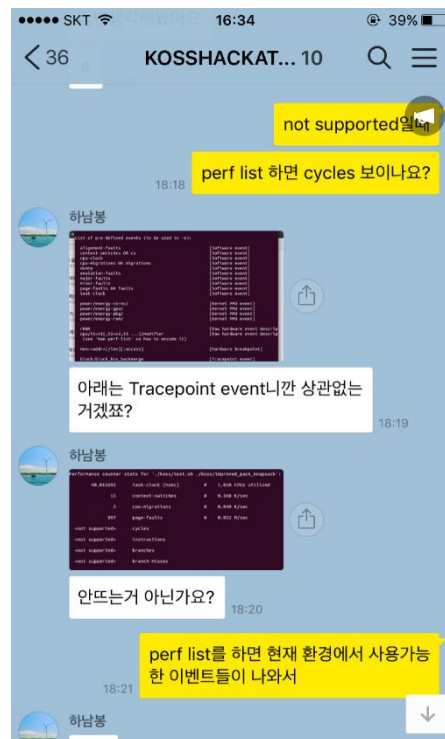
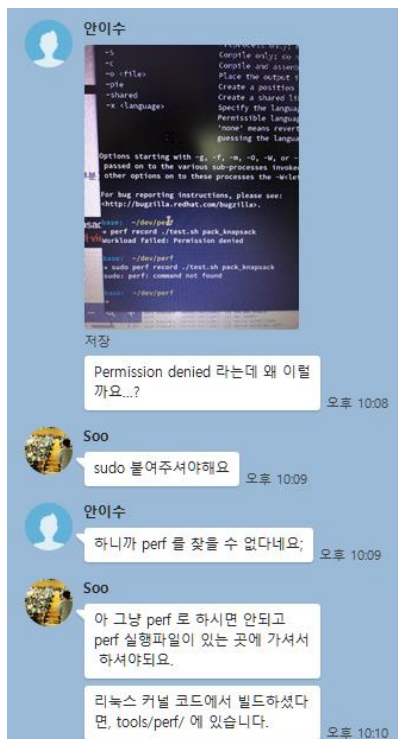
-l, --list          show current config variables
```

1

III. 진행 과정

• 온라인 모임 – 9.25(일)

- Perf 개발 환경 구축
- Git 기초 명령어 학습 (add, commit, push, pull)
- Git 심화 명령어 학습 (checkout, reset, rebase)
- Perf에 대해서 개별 조사



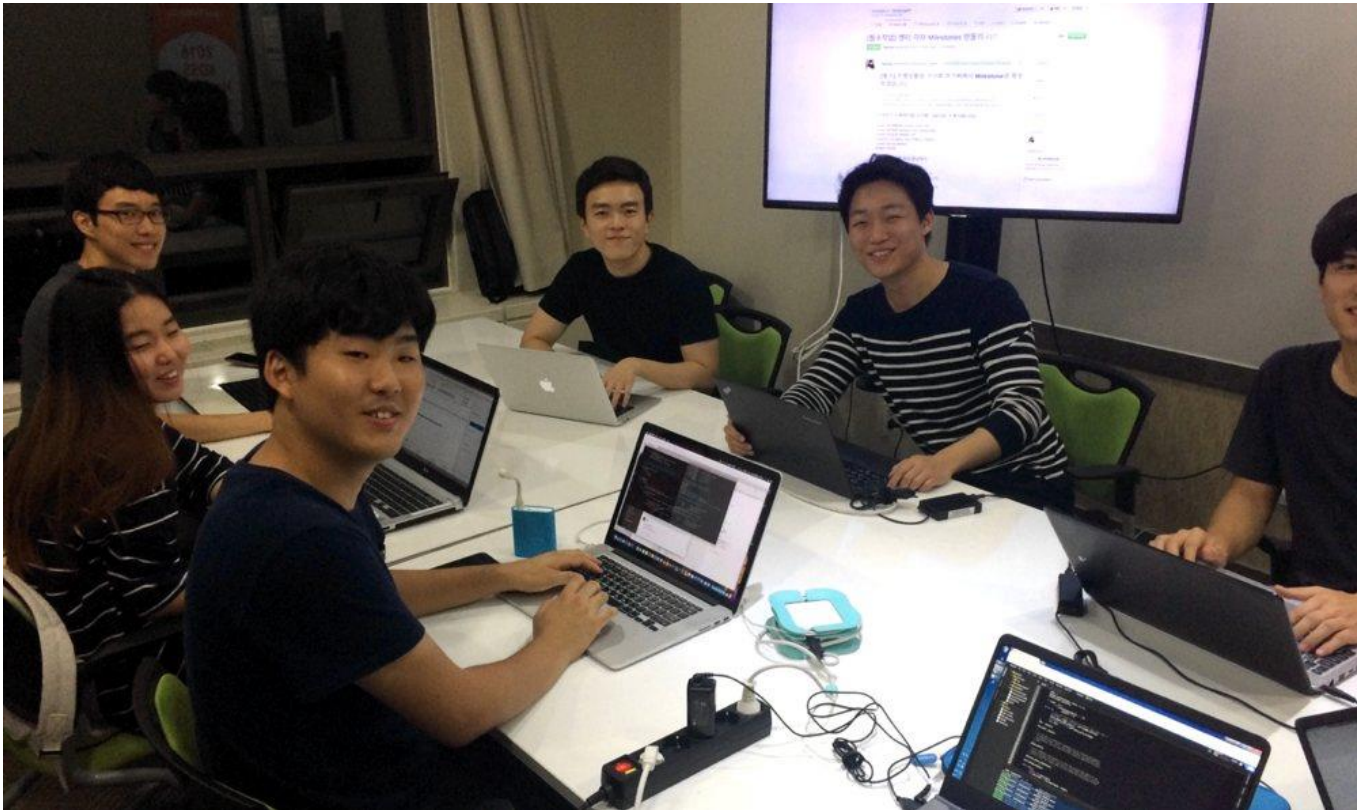
III. 진행 과정

- 오프라인 모임 – 9.26(월)

- 장소 : kossilab 센터
- 시간 : 19:00 ~ 20:30
- 작업 내용
 - Git 기초/ 심화 명령어 실습
 - Github으로 pull request 생성 실습
 - Perf로 간단 프로그램 성능 테스트
 - vim + ctags 를 이용한 perf 개발 환경 구축
 - Uftrace를 이용한 프로그램 실행 시 함수 호출 trace 실습
 - Git 을 이용하여 수정 내용 패치 생성 및 메일링 리스트에 패치 전송 실습

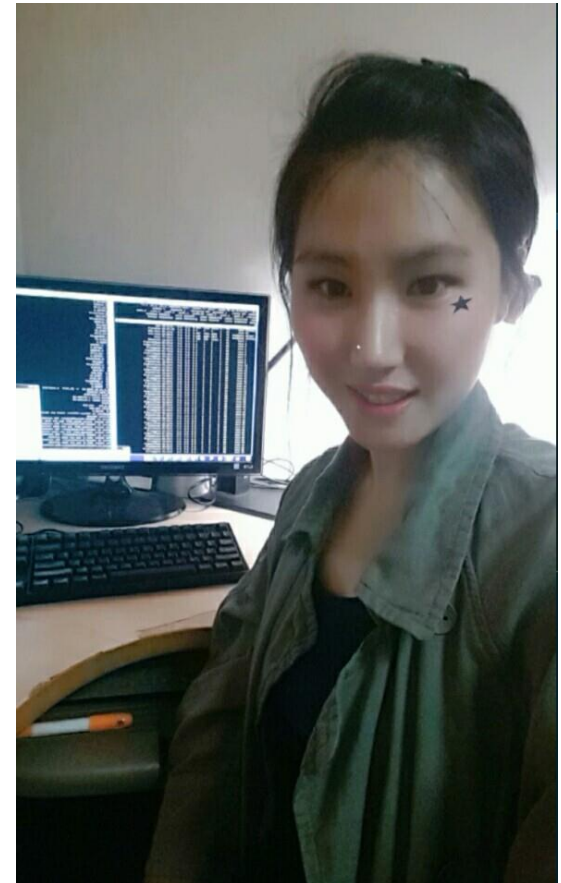
III. 진행 과정

- 오프라인 모임 – 9.26(월)
 - 장소 : kossilab 센터
 - 시간 : 19:00 ~ 20:30



III. 진행 과정

- 온라인 모임 – 9.27(화)
 - Perf의 Documentation/tips.txt 실제 적용 및 추가 라인 만들기
 - Perf 소스 리딩 tool 익히기 미션



III. 진행 과정

• 온라인 모임 – 9.28(수)

- Perf 소스 리딩 미션 ‘perf.data에서 이벤트 샘플 정보를 어떻게 읽고 셋팅 하는가?’
- Perf 소스 리딩 미션 ‘subcommand는 어떻게 실행 되는가?’
- Perf 새로운 stat diff 만들기 관련 토의

```

root@nb-500R5K-501R5K-500R5Q:~/koss# ./perf report perf.data
root@nb-500R5K-501R5K-500R5Q:~/koss# ./perf report
root@nb-500R5K-501R5K-500R5Q:~/koss# ./perf diff old perf
perf      perf.data
root@nb-500R5K-501R5K-500R5Q:~/koss# ./perf diff old perf.data
# Event 'cycles:pp'
#
# Baseline   Delta   Shared Object      Symbol
# .....
#
83.35%      old_pack_knapsack  [.] get_cond_maxprice
12.71%      old_pack_knapsack  [.] pack_knapsack
0.48%      [kernel.kallsyms]  [k] clear_page_c_e
0.45%      libc-2.19.so       [.] __mbrtowc
0.42%      bash               [.] sh_xmalloc
0.40%      bash               [.] _IO_putc@plt
0.37%      [kernel.kallsyms]  [k] unmap_page_range
0.31%      [kernel.kallsyms]  [k] page_add_file_rmap
0.30%      [kernel.kallsyms]  [k] rw_verify_area
0.25%      [kernel.kallsyms]  [k] copy_page
0.24%      [kernel.kallsyms]  [k] fget_light
0.22%      bash               [.] close_buffered_stream
0.17%      [kernel.kallsyms]  [k] module_put
0.15%      [kernel.kallsyms]  [k] __call_rcu
0.10%      [kernel.kallsyms]  [.] native_irq_return_iret
0.02%      [kernel.kallsyms]  [k] __put_user_4
0.02%      [kernel.kallsyms]  [k] __zone_watermark_ok
0.02%      [kernel.kallsyms]  [k] perf_event_comm_output
0.02%      [kernel.kallsyms]  [k] __do_page_fault
0.01%      [kernel.kallsyms]  [k] native_sched_clock
0.00%      [kernel.kallsyms]  [k] pgd_free
0.00%      [kernel.kallsyms]  [k] native_read_tsc
0.00%      [kernel.kallsyms]  [k] nmi_restore
0.00%      [kernel.kallsyms]  [k] native_write_msr_safe
+0.00%      [kernel.kallsyms]  [.] pack_knapsack
+0.00%      [kernel.kallsyms]  [.] get_cond_maxprice
+0.00%      [kernel.kallsyms]  [k] unlock_page
+1.81%      bash               [.] 0x00000000000023817
+1.72%      bash               [.] __mempcpy_sse2_unaligned
+1.44%      libc-2.19.so       [.] reader_loop
+1.38%      [kernel.kallsyms]  [k] file_update_time
+0.95%      bash               [k] perf_event_aux
+0.95%      [kernel.kallsyms]  [k] 0x0000000000045b7c0
+0.77%      [unknown]          [k] get_page_from_freelist
+0.51%      [kernel.kallsyms]  [k] number_isra.13
+0.06%      [kernel.kallsyms]  [k] __perf_event_header__init_id
+0.06%      [kernel.kallsyms]  [k] raw_spin_lock
+0.04%      [kernel.kallsyms]  [k] finish_task_switch
+0.00%      [kernel.kallsyms]  [k] perf_sample_event_took

```

```

/home/taeung/lighttaeung/linux-perf/tools/perf
40  if (!prefixcmp(cmd, "trace")) {
41  #ifdef HAVE_LIBAUDIT_SUPPORT
42  setup_path();
43  argv[0] = "trace";
44  return cmd_trace(argc, argv, NULL);
45  #else
46  fprintf(stderr,
47  "trace command not available: missing audit-libs devel package at build time.\n");
48  goto out;
49  #endif
50
51  /* Look for flags.. */
52  argv++;
53  argc--;
54  handle_options(&argv, &argc, NULL);
55  commit_page_choices();
56
57  if (argc > 0) {
58  if (!prefixcmp(argv[0], "-"))
59  argv[0] += 2;
60  } else {
61  /* The user didn't specify a command, give them help */
62  printf("Usage: %s\n", perf_usage_string());
63  list_common_cmds_help();
64  printf("Use '%s %s' for more info.\n", argv[0], argv[1]);
65  goto out;
66  }
67  cmd = argv[0];
68
69  test_attr_init();
70
71  /*
72  * We use PATH to find perf commands, but we precede some higher
73  * precedence paths, the --exec-path option, the PERF_PATH
74  * environment, and the $PERFEXECDIR from the Makefile at build
75  * time.
76  */
77  setup_path();
78
79  /* Block SIGWINCH notifications so that the thread that wants it can
80  * unblock and get signals like select interrupted instead of waiting
81  * forever while the signal goes to some other non interested thread
82  */
83  pthread_block_sigwinch();
84
85  perf_debug_setup();
86
87  while (1) {
88  static int done_help;
89  int was_alias = run_argv(&argc, &argv);
90
91  if (errno != ENOENT)
92  break;
93
94  if (was_alias) {
95  fprintf(stderr, "Expansion of alias '%s' failed: "
96  "'%s' is not a perf-command\n",
97  cmd, argv[0]);
98  goto out;
99  }
100  if (!done_help) {
101  cmd = argv[0] = help_unknown_cmd(cmd);
102  done_help = 1;
103  } else break;
104  }
105
106  fprintf(stderr, "Failed to run command '%s': %s",
107  cmd, str_error_r(errno, sbuf, sizeof(sbuf)));
108  goto out;
109
110  return 1;
111 }

```

III. 진행 과정

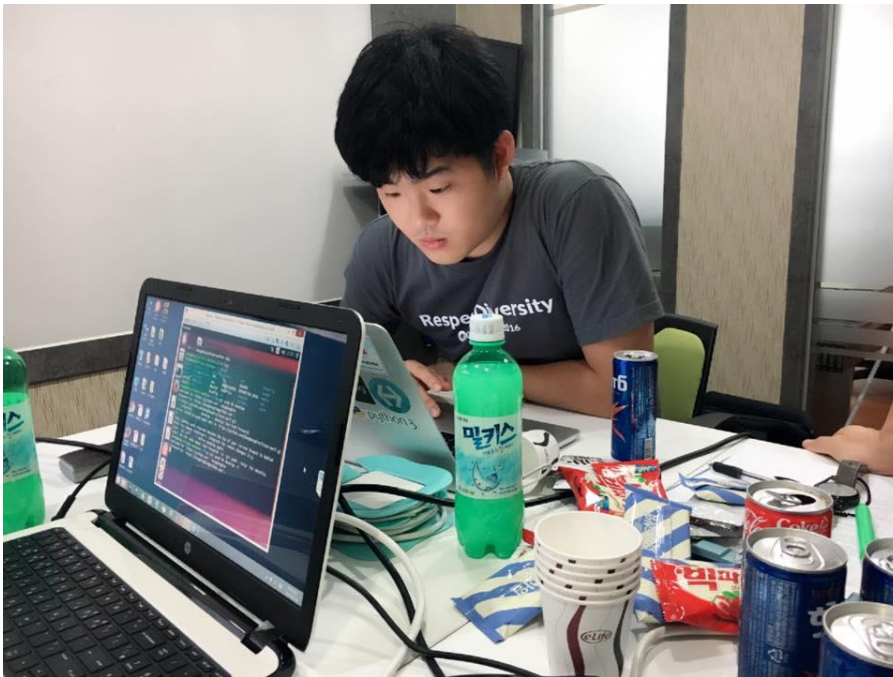
- 오프라인 모임 – 9.29(목)

- 장소 : kossilab 센터
- 시간 : 19:00 ~ 07:30 ★ 9.30(금) ★
- 작업 내용
 - (분석 정리) Perf 소스 리딩 미션 'perf.data에서 이벤트 샘플 정보를 어떻게 읽고 셋팅 하는가?
 - (분석 정리) Perf 소스 리딩 미션 'subcommman는 어떻게 실행 되는가?
 - Perf의 Documentation/tips.txt 실제 적용 및 추가라인 만들기(각자 실제 작업 진행)
 - 총 5명 리눅스 커널 메인라인에 PATCH 메일 전송 성공(2명은 적용 대기중)

III. 진행 과정

- 오프라인 모임 – 9.29(목)

- (분석 정리) Perf 소스 리딩 미션 'perf.data에서 이벤트 샘플 정보를 어떻게 읽고 셋팅 하는가?'
- (분석 정리) Perf 소스 리딩 미션 'subcomman는 어떻게 실행 되는가?'
- Perf의 Documentation/tips.txt 실제 적용 및 추가라인 만들기(각자 실제 작업 진행)
- 총 5명 리눅스 커널 메인라인에 PATCH 메일 전송 성공(2명은 적용 대기중)



III. 진행 과정

- 오프라인 모임 – 9.29(목)
 - (분석 정리) Perf 소스 리딩 미션 'perf.data에서 이벤트 샘플 정보를 어떻게 읽고 셋팅 하는가?'
 - (분석 정리) Perf 소스 리딩 미션 'subcomman는 어떻게 실행 되는가?'
 - Perf의 Documentation/tips.txt 실제 적용 및 추가라인 만들기(각자 실제 작업 진행)
 - 총 5명 리눅스 커널 메인라인에 PATCH 메일 전송 성공(2명은 적용 대기중)



III. 진행 과정

- 온라인, 오프라인 모임 – 9.30(금)
 - 장소 : kossilab 센터
 - 시간 : 18:00 ~ 21:00
 - 작업 내용
 - 도커를 활용한 perf 개발 테스트 환경 구축 작업 (오프라인)
 - Linux Kernel – perf 프로젝트 초보개발자를 위한 가이드 문서 (온라인)
 - 발표자료 작업 진행 (온라인)

혼자가면 빨리가지만
함께 가면 멀리간다.

