Optymalizacja programów Open-Source

Profilery wysokiego poziomu
część 1

Krzysztof Lichota
lichota@mimuw.edu.pl

strace

strace

- Wypisuje wywołania funkcji systemowych wraz z najważniejszymi parametrami
- Zasada działania: śledzenie funkcji systemowych za pomocą interfejsu do debugowania (ptrace)
- Narzut: średni (każde śledzone wywołanie powoduje przełączenie kontekstu do procesu śledzącego)

Zalety strace

- Nie wymaga rekompilacji programu ani symboli do debugowania, umożliwia śledzenie każdego programu
- Pokazuje czas przebywania w funkcji systemowej i czas wywołania
- Dostępny praktycznie w każdej instalacji Linuksa
- Możliwość ograniczenia narzutu przez śledzenie tylko wybranych funkcji
- Pozwala uzyskać ślad działania programu przez korelację ze źródłami programu
- Można podłączać się do działających już programów

Wady strace

- Nie każde wywołanie funkcji w programie przekłada się na wywołanie systemowe (np. buforowanie wejścia/wyjścia)
- Spory narzut
- Śledzi tylko wywołania systemowe
- Nie pokazuje dostępu do dysku przy mmapowanych plikach

Wywołanie strace

- **strace <opcje> -o plik.strace -- polecenie argumenty**
- W przypadku śledzenia procesów potomnych do osobnych plików wynik jest w plik.strace.PID

Przydatne opcje strace

- -f – śledzi procesy potomne
- -ff – śledzi procesy potomne i zapisuje wynik do osobnych plików
- -T – pokazuje czas przebywania w funkcji systemowej
- -tt – pokazuje czas absolutny wywołania (rozdzielczość mikrosekundowa)
- -ttt – jak -tt ale czas jest w postaci sekund (do łatwiejszego przetwarzania w skryptach)
- -r – wypisuje czas od poprzedniego wywołania funkcji systemowej

Przydatne opcje strace

- -e trace=funkcja1,funkcja2 – śledzi tylko podane funkcje systemowe
- -e trace=file – śledzi wszystkie funkcje na plikach
- -e trace=process – śledzi wszystkie funkcje dotyczące procesów (fork, exec, ...)
- -e trace=network – śledzi wszystkie funkcje sieciowe
- -e trace=signal – śledzi funkcje związane z sygnałami
- -e trace=ipc – śledzi wywołania IPC (InterProcess Communication – semafory, łącza, itp.)
- -e abbrev=funkcje – dla których skracać struktury

Przydatne opcje strace

- Można podłączyć się do działających programów za pomocą opcji **-p pid**
- Za pomocą opcji **-i** można wydrukować miejsce wywołania w programie i przełożyć to na funkcję za pomocą symboli do debugowania

ltrace

ltrace

- Śledzi wywołania funkcji bibliotek dynamicznych
- Zasada działania: breakpointy w kodzie i śledzenie przez interfejs debuggera
- Narzut: średni (śledzenie powoduje przełączanie do kontekstu procesu śledzącego)

Zalety ltrace

- Pokazuje wywołania funkcji bibliotecznych, co daje dużo lepszy ślad

Wady ltrace

- Nie pokazuje argumentów funkcji
- Nie działa dla niektórych programów (np. OpenOffice)
- Działa tylko na niektórych architekturach Linuksa
- Nie działa dla programów używających trików do ładowania (np. kdeinit w KDE)

Przydatne opcje

Obsługuje podobne opcje jak strace i oprócz tego:

- --demangle – tłumaczy symbole C++ na zrozumiałe
- -l biblioteka – śledzi tylko symbole z danej biblioteki
- -S – pokazuje również wywołania systemowe

Google profiler

Google profiler

- Zasada działania – próbkowanie ze zrzutem backtrace
- Narzut – niewielki (w zależności od częstotliwości próbkowania)

Przykład – graf dla Gwenview

Zalety

- Łatwy w użyciu (LD_PRELOAD do istniejącego kodu)
- Intuicyjny i przejrzysty sposób prezentacji wyników
- Wolnodostępny, na licencji BSD
- Łatwa modyfikacja prezentacji wyników (pprof jest napisany w Perlu)

Wady

- Nie pokazuje, gdzie proces śpi
- Nie obsługuje zewnętrznych symboli (pakiety -dbg) – można to łatwo dodać
- Skleja różne instancje template – też można to zmodyfikować
- Mała rozdzielczość (domyślnie 1/100 sekundy), zwiększenie rozdzielczości zwiększa narzut

Sposób użycia

Tworzenie profilu

- CPUPROFILE=ścieżka-do-pliku-profilu LD_PRELOAD=libprofiler.so ścieżka-do-programu
- Za pomocą zmiennej środowiskowej PROFILEFREQUENCY można sterować częstotliwością próbkowania

Generowanie wyniku

- pprof <opcje> ścieżka-do-programu ścieżka-do-profilu >plik-z-grafem.ext

Przydatne opcje pprof

Format wyświetlania

- --text - tekstowy
- --ps – PostScript (można wysłać do kghostview za pomocą „**pprof –ps ... | kghostview -**”)
- --gif – do pliku GIF

Przydatne opcje pprof

Opcje formatu tekstowego:

- --cum – w widoku tekstowym sortuje po skumulowanym czasie
- --disasm=<regexp> - wyświetla kod asemblera dla funkcji pasujących do wyrażenia, wraz z czasami
- --list=<regexp> - wyświetla kod źródłowy funkcji pasujących do wyrażenia, wraz z czasami

Przydatne opcje pprof

Opcje wyświetlania grafu

- --addresses – jeden węzeł na adres w programie
- --lines – jeden węzeł na linię kodu (wymaga symboli)
- --functions – jeden węzeł na funkcję (domyślny, nie wymaga symboli)
- --files – jeden węzeł na plik kodu źródłowego

Przydatne opcje pprof

Opcje do ograniczania wyniku/rozmiaru grafu.

- --focus=<regexp> - wyświetla tylko symbole pasujące do wyrażenia i wszystkie symbole na ścieżce prowadzącej do tego symbolu
- --ignore=<regexp> - ignoruje symbole określone wyrażeniem i wszystkie prowadzące do niego
- opcje do odrzucania krawędzi

Często spotykane problemy

- Generowanie grafu powinno się odbywać na tej samej maszynie, co generowanie profilu (adresy w bibliotekach mogą się różnić)
- Sygnał może spowodować ucięcie profilu lub w ogóle nie będzie on utworzony
- Jeśli biblioteka nie zawiera informacji o symbolach, wszystkie próbki mogą być przypisane do jednego symbolu
- Niektóre krawędzie grafu mogą być niewidoczne (graf jest niespójny) – zależy od opcji wyświetlania.