Centro Federal de Educação Tecnológica de Minas Gerais

CEFET-MG

DEPARTAMENTO DE COMPUTAÇÃO (DECOM)
LABORATÓRIO DE ARQUITETURA E ORGANIZAÇÃO DE COMPUTADORES I
Professor: Mateus Felipe Tymburibá Ferreira

Este enunciado está disponível no Sistema Acadêmico

PRÁTICA 1: AVALIAÇÃO DE DESEMPENHO ATRAVÉS DAS UNIDADES DE MONITORAMENTO DE PERFORMANCE (PMU)

OBS.: Trabalho individual. Prazo de entrega do relatório: 1 semana após a prática, impreterivelmente até o horário de início da aula. Enviar por e-mail para "julianasteixeira@hotmail.com" e "andrepgfon@gmail.com" com o seguinte título: "Pratica X – Nome Completo Do Aluno".

O que deve ser entregue

- Arquivo compactado nomeado "pratica-X_nome-completo-aluno.zip". Certifique-se de que o arquivo não está corrompido.
- Este arquivo deverá conter uma pasta com todos os arquivos utilizados na prática (códigos fontes, imagens, resultados, etc, sempre que for o caso).
- Escreva em texto simples e conciso as suas análises e considerações.
- Responda às perguntas realizadas na prática, quando for o caso.

O que deve ser feito

1) Escolha 4 aplicativos a serem executados para avaliar o desempenho do sistema.
2) Identifique o modelo do seu processador e os eventos de monitoramento de performance que ele suporta.
3) Escolha, no mínimo, 4 eventos a serem monitorados.
4) Execute os 4 aplicativos selecionados, monitorando os eventos escolhidos através da aplicação "perf", do Linux.
5) Prepare a apresentação dos resultados em tabelas e/ou gráficos.

O que deve ser respondido

1) Justifique a escolha dos 4 aplicativos selecionados.
2) Justifique a escolha dos eventos monitorados.
3) Apresente uma discussão dos resultados obtidos, elencando possíveis relações de causa e efeito para os eventos e o desempenho observados.

Um pouco de teoria

Unidades de Monitoramento de Desempenho

Unidades de Monitoramento de Desempenho (PMU – *Performance Monitoring Units*) são dispositivos de hardware disponíveis na maioria dos processadores modernos. Essas unidades registram eventos microarquiteturais tais como falhas nos diversos níveis de cache e erros na previsão de desvios, entre outros. Para isso, essas unidades são compostas por registradores de propósito específico denominados Contadores de Performance do Hardware (HPCs – *Hardware Performance Counters*). Esses registradores armazenam a contagem dos referidos eventos microarquiteturais [1].

Ao contrário de monitores via software, os contadores do hardware permitem um acesso preciso e de baixo *overhead* a uma vasta gama de informações detalhadas de desempenho relacionadas a unidades funcionais da CPU [2]. Outra vantagem dos HPCs consiste em não exigir modificações no código-fonte da aplicação a ser monitorada. Na verdade, os contadores do hardware permitem o monitoramento de qualquer binário, incluindo aplicações para as quais não se conheça o código-fonte.

Contudo, os tipos, significados e nomes dos contadores do hardware variam de um modelo de microarquitetura para outro, em função de diferenças na organização das estruturas de hardware. Por conta disso, é necessário pesquisar quais HPCs estão disponíveis em um modelo particular de processador. Além disso, suporte em nível do *kernel* do sistema operacional é necessário para se acessar os contadores de performance do hardware, já que esses registradores só podem ser manipulados através de instruções de nível privilegiado. Felizmente, já existe um módulo adicional para o *kernel* do Linux bastante estável que oferece suporte aos HPCs.

Suporte aos Contadores de Performance no Linux

A partir da versão 2.6.31, o *kernel* do Linux incorporou uma interface genérica *(perf_events*) para acesso aos contadores do hardware, bem como a diversos eventos de software, tais como atividades de escalonamento e chamadas de sistema [3]. Um utilitário em nível de usuário denominado "*perf*" também foi construído a partir dessa interface com o *kernel*. Essa ferramenta provê uma forma de registrar e visualizar eventos relacionados a processos em execução. O "*perf*" é acessado pela linha de comandos e oferece vários subcomandos capazes de registrar estatísticas do sistema como um todo (tanto em nível de *kernel* quanto em nível de usuário). A seguir, estão descritas algumas opções do comando perf:

- list: apresenta uma lista dos eventos suportados e já mapeados (ex: perf list)
- stat: determina que o perf conte eventos.
- -o <arquivo>: indica um arquivo de saída onde os resultados devem ser escritos.
- --apend: adiciona os resultados ao arquivo de saída, caso ele já exista.
- -B: insere separadores de milhares nos números exibidos na saída.
- -r 1: executa a aplicação uma única vez.
- -e rXXXX:uk: lista de eventos a serem monitorados (uk: níveis usuário e *kernel*).

Exemplo de comando perf completo para monitorar 4 eventos não mapeados (indicados via máscara em hexadecimal):

```
perf stat -o out.txt --append -B -r 1 -e r8888:uk,r8889:uk,ra088:uk,ra089:uk firefox
```

Note que é possível registrar eventos ocorridos em nível de usuário(:u), de kernel (:k), ou ambos (:uk). É importante ressaltar também que o perf possui cadastrados apenas os eventos arquiteturais, ou seja, aqueles que independem do modelo específico do processador. Esses são os eventos listados pela opção "list". No entanto, existe uma gama muito maior de eventos específicos suportados por cada modelo de processador. Esses eventos também podem ser monitorados usando-se o perf. Para isso, é necessário passar os valores das máscaras correspondentes aos eventos usando a opção "-r" do perf, conforme exemplo acima. Consulte os eventos suportados pelo seu modelo de processador no Manual para Desenvolvedores de Software da Intel (Cap. Performance Monitoring Events) [4]. Os códigos de eventos indicados no comando perf acima correspondem aos seguintes eventos descritos no manual da Intel:

Event Num.	Umask Value	Event Mask Mnemonic	Description	Comment
88H	08H	BR_INST_EXEC.RETURN_NEAR	Qualify indirect near branches that have a return mnemonic.	Must combine with umask 80H.
89H	08H	BR_MISP_EXEC.RETURN_NEAR	Qualify mispredicted indirect near branches that have a return mnemonic.	Must combine with umask 80H.
88H	20H	BR_INST_EXEC.INDIRECT_NEAR _CALL	Qualify indirect near calls, including both register and memory indirect, executed.	Must combine with umask 80H.
89H	20H	BR_MISP_EXEC.INDIRECT_NEAR _CALL	Qualify mispredicted indirect near calls, including both register and memory indirect, executed.	Must combine with umask 80H.

Note que ao indicar os valores em hexadecimal para o perf (valor que sucede a letra "r"), os bytes devem ser invertidos em relação ao manual da Intel, já que o perf recebe o valor da máscara à esquerda do valor do evento. Exemplo: o evento 89 com máscara 88, indicado no manual da Intel, deve ser apontado para o perf através da opção -r8889.

Para identificar o modelo do seu processador, após executar o comando "lscpu" no Linux, observe os campos "CPU family" e "Model".

Referências

[1] Aman, S. Buchke, A. and Lee, Y., "Performance Monitoring Unit," 2015. [Online]. Available: http://rts.lab.asu.edu/web_438/project_final/Talk%209%20Performance%20Monitoring%20Unit.pdf.

[2] Kufrin, R., "Measuring and Improving Application Performance with PerfSuite," 2015. [Online]. Available: http://perfsuite.ncsa.illinois.edu/publications/LJ135/x27.html.

[3] Kernel.org, "Linux kernel profiling with perf," 2015. [Online]. Available: https://perf.wiki.kernel.org/index.php/Tutorial.

[4] Intel, "Intel 64 and IA-32 Architectures Software Developer's Manual Combined Volumes: 1, 2A, 2B, 2C, 3A, 3B and 3C", 2015. [Online]. Available: http://www.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-manual-325462.pdf