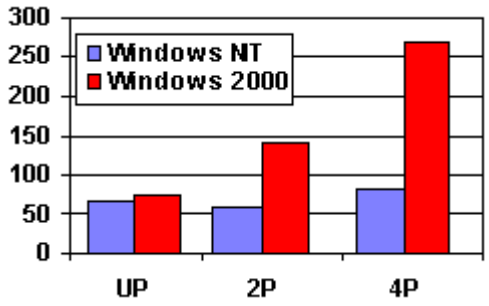
Ottimizzazioni delle prestazioni di un Web server
Ottimizzazioni delle prestazioni di un Web server

Spesso il ***server*** non è in grado di gestire tutto il carico di cui è gravato. Inoltre, una volta messi in produzione, i ***server*** devono avere la possibilità di crescere: un ***server*** può avere inizialmente 100 utenti, ma potrebbe passare a 500 utenti nel giro di poco tempo.

Secondo una definizione generica, il *benchmarking* è il **processo** che raffronta due o più sistemi (o componenti di un sistema) al fine di stabilire quale sia in grado di fornire le migliori prestazioni o di consentire a un maggior numero di utenti di accedere a risorse quali *database* e *file system*. Fondamentalmente, esistono due tipi di ***benchmark*** generici:

- **sintetici/artificiali**: non riflettono l'uso effettivo del ***server*** in un ambiente produttivo. Questi ***benchmark*** applicano inoltre un carico costante e artificiale a uno specifico componente del ***server***, in modo da valutarne le prestazioni massime. Spesso i risultati di questi ***benchmark*** non hanno un grande significato, se non quando vengono usati in modo relativo.
- **mondo/vita reale**: cercano invece di misurare le prestazioni offerte dal sistema sotto i carichi che potrebbero esistere realmente in un ambiente produttivo. Purtroppo, l'unico modo per compiere questo ***benchmark*** è quello di mettere il ***server*** in produzione e vedere che cosa succede - un lusso che ben pochi si possono permettere.



I guasti del sistema possono dipendere dai *driver software* applicativi o a livello di sistema, da una configurazione errata, oppure dalla presenza di *hardware* difettoso. Quando si effettua il *burn-in*, spesso si possono rilevare dei problemi che altrimenti si manifesterebbero soltanto dopo un periodo di uso esteso, oppure quando il sistema si trova a dover gestire carichi elevati. Sul mercato sono disponibili molti ***benchmark***, decisamente troppi per elencarli tutti.

Alcuni sono di livello commerciale, altri sono *shareware*; alcuni sono particolarmente interessanti, mentre altri non hanno nessuna utilità pratica.

Burn-in di sistema e Benchmark

Attualmente, l'uso più comune dei ***benchmark*** è quello di misurare le prestazioni. I ***benchmark*** si possono tuttavia usare anche per il burn-in di un elemento *hardware*, oppure di una nuova **applicazione**. Nel corso di questa procedura, al ***server*** vengono imposti elevati livelli di sollecitazione per lunghi periodi di tempo, in modo da vedere se qualche parte del sistema si guasta.

I guasti del sistema possono dipendere dai *driver software* applicativi o a livello di sistema, da una configurazione errata, oppure dalla presenza di *hardware* difettoso. Quando si effettua il *burn-in*, spesso si possono rilevare dei problemi che altrimenti si manifesterebbero soltanto dopo un periodo di uso esteso, oppure quando il sistema si trova a dover gestire carichi elevati. Sul mercato sono disponibili molti ***benchmark***, decisamente troppi per elencarli tutti.

Transaction processing performance council. Chi usa grossi sistemi orientati alle transazioni ha

probabilmente già sentito parlare dei ***benchmark*** *TPC* (*Transaction Processing Performance Council*). Questi ultimi sono diventati lo *standard* di fatto per la misurazione delle prestazioni dei sistemi basati sulle transazioni. I test *TPC* sono piuttosto complessi e difficili da configurare ed eseguire. Alcuni produttori, tra cui *Sybase* e *Compaq* hanno *team* dedicati all'esecuzione dei test *TPC* allo scopo di ottimizzare i prodotti per ottenere i migliori risultati. I risultati offerti da questi test vengono verificati da *TPC*: non esistono quindi risultati *TPC* non ufficiali.

Spec (*Standard Performance Evaluation Corporation*). È un ***benchmark*** sintetico che misura le capacità di elaborazione del ***server*** sugli interi e a virgola mobile. Questo ***benchmark*** viene usato spesso dai produttori di *CPU*, come *Intel* e AMD, per indicare le prestazioni relative dei microprocessori. Il ***benchmark*** SPEC è diffuso anche nel campo dell'informatica scientifica e tecnica. Esistono cinque tipologie principali di ***benchmark*** SPEC:

- SPEC CINT92: misura le prestazioni dell'elaborazione sugli interi.
- SPEC CFP92: misura le prestazioni dell'elaborazione a virgola mobile.
- SPEC CINT95: nuova versione del ***benchmark*** sull'elaborazione degli interi.
- SPEC CFP95: nuova versione del ***benchmark*** sull'elaborazione a virgola mobile.
- SPEC SDM: *Benchmark Software Development Multitasking*.

I fattori da prendere in considerazione nei benchmark

Quando ci si appresta a effettuare un ***benchmark*** è necessario prendere in considerazione i seguenti fattori.

- Verificare che ogni componente sia aggiornato.
- Mantenere una base uniforme.
- Rendere coerente il test.
- Ripetere il test.
- Tenere sotto controllo il ***server***, la rete e il ***benchmark***.
- Capire innanzitutto cosa fa il ***benchmark***.
- Controllare i risultati campione prima di iniziare.

I sistemi informatici sono complessi; anche il più piccolo ***server*** con processore singolo è formato da molti componenti, che devono lavorare all'unisono per assicurare l'operatività. Quando si esegue un ***benchmark***, bisogna essere certi che il sistema operativo, i *driver* delle applicazioni, il *firmware* specifico al ***server*** e i *driver* siano aggiornati. È possibile che il solo aggiornamento di un *driver* consenta di aumentare fino al 25 per cento le prestazioni del sistema.

Lo scopo del ***benchmark*** è quello di consentire un raffronto tra le prestazioni di due o più sistemi. A questo scopo, è necessario mantenere una base uniforme. Anche le più piccole variazioni nei sistemi possono distorcere i risultati. Per esempio, se si ha un sistema con quattro *hard disk* in una batteria RAID 0 e un altro sistema con sei dischi in una batteria RAID 0, il secondo godrà di un grosso vantaggio in lettura/scrittura, dal momento che usa più dischi per compiere la stessa quantità di lavoro.

Durante il ***benchmark***, la configurazione di ciascun ***server*** che viene sottoposto a test deve essere uniforme. Molti ***benchmark*** consentono una messa a punto secondo le proprie esigenze. Sebbene questa possibilità permetta di ottenere un'enorme flessibilità, esiste il pericolo di compiere un test incoerente. È quindi opportuno documentare tutte le modifiche che sono state apportate al ***benchmark*** e ripeterle per tutti i ***server*** sottoposti a test. La cosa migliore è compiere il test su ogni singolo ***server*** almeno per tre volte, osservando i dati statistici relativi alle prestazioni. Per esempio, se viene eseguito un test su un *database* caratterizzato da un elevato livello di *I/O*, è opportuno verificare che il sottosistema disco non rappresenti un collo di bottiglia.

Quando si esegue il ***benchmark***, bisogna tenere sotto controllo non soltanto il sistema operativo e le applicazioni eseguite sul ***server***, ma anche il *tool* di ***benchmark*** e i vari componenti del ***server***.

Dopo aver terminato ciascun test, è opportuno controllare i *log* degli errori del sistema operativo e dell'**applicazione**, in modo da verificare che non si sia verificato nessun problema. Capire che cosa fa il ***benchmark*** è la chiave per valutare con successo il ***server*** sottoposto al test.

Non è una buona idea quella di sottoporre a *benchmarking* un ***server*** multiprocessore con un solo *hard disk*, dal momento che non si potranno mai valutare le vere prestazioni del ***server***.

Dopo avere eseguito il ***benchmark*** sul ***server*** di test, bisogna scrutinare i risultati. Se viene usato un *tool* di *monitoring* del ***server*** in grado di registrare alcuni dati statistici a livello di sistema, come l'utilizzo della *CPU*, l'*I/O* su disco e la *cache* di memoria, controllare che il ***benchmark*** solleciti correttamente il sistema e che non esistano dei colli di bottiglia imprevisti. Al termine del ***benchmark*** è necessario valutare come si è comportato il sistema, usando i *tool* di *monitoring*. Bisogna essere certi che il ***server*** sia configurato appropriatamente allo scopo di ottenere i risultati che si stanno cercando. Diversamente si potrebbero trarre conclusioni errate.

I colli di bottiglia

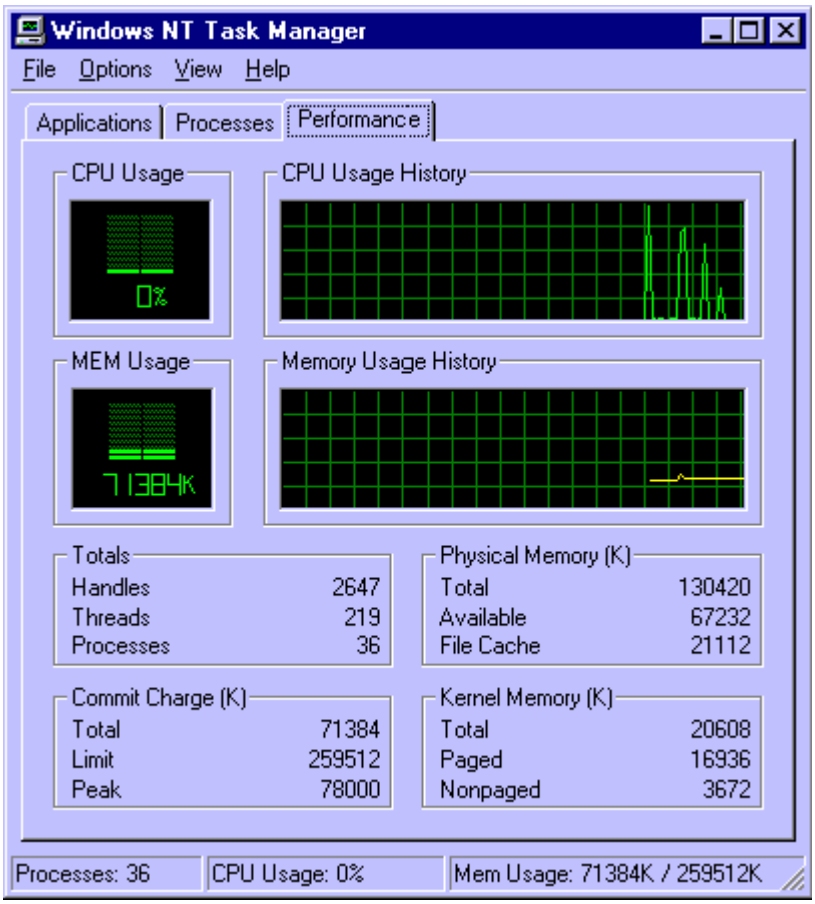
I colli di bottiglia del sistema possono esistere ovunque. Quando viene eliminato un collo di bottiglia, se ne manifesta subito un altro. Spesso il ***server*** è veloce soltanto quanto il collegamento o sub-componente più lento. Vi sono dei ***server*** equipaggiati molto bene, con processori multipli, centinaia di *Mbyte* di memoria e batterie di dischi ad alte prestazioni collegato alla rete attraverso un singolo segmento *Ethernet* da 10 Mbps. In casi come questo, spesso gli amministratori si domandano il perché le prestazioni per gli utenti sembrino così limitate se il ***server*** viene utilizzato soltanto al 5 per cento. Quando si cerca di identificare i colli di bottiglia, è opportuno esaminare i singoli componenti del sistema e il modo in cui questi componenti sono collegati.

La *CPU* e la memoria

Grazie ai nuovi processori, quasi sempre la *CPU* non costituisce più un collo di bottiglia. È tuttavia opportuno verificare attentamente il modello di *CPU* che viene utilizzato. Le dimensioni della *cache* di secondo livello possono avere effetti significativi sulle prestazioni. Se i *computer* vengono utilizzati come ***server*** di applicazioni, è opportuno scegliere una *cache* di secondo livello più capiente possibile.

Un altro importante componente del sistema è la memoria. Quando l'**applicazione** cerca i dati e questi ultimi non sono presenti nella *cache*, la *CPU* deve usare la memoria principale. Sebbene la memoria sia economica e molto veloce, la *cache* della *CPU* è ancora più veloce ma più costosa. La memoria virtuale consente di ingannare il sistema operativo, facendogli credere che sia disponibile più memoria di quanta ne esista effettivamente, ma questa memoria aggiuntiva viene archiviata sul disco. Quando il sistema operativo ha bisogno di dati che non sono presenti nella *cache* della *CPU* o nella memoria principale, legge il *file* della memoria virtuale su disco. Se il sistema operativo deve leggere in continuazione i contenuti del disco per ottenere i dati che gli servono, le prestazioni ne soffrono. Questa condizione viene chiamata **paginazione** e indica che è giunto il momento di aggiungere RAM al sistema.

Sono disponibili alcune linee guida generiche per stabilire quanta memoria dovrebbe essere installata su di un sistema. Una regola generale è quella di analizzare, in condizioni di carico il *task manager* ed osservare quanto la memoria usata (RAM + virtuale) sia lontana dalla RAM effettivamente installata. Anche il carico della *CPU* fornisce un primo sistema di analisi dello stato di salute del sistema.



I fattori che possono determinare la quantità di memoria aggiuntiva che potrebbe rendersi necessaria sono diversi. Bisogna chiedersi quanti sono gli utenti che dovranno collegarsi, quanto spazio su disco occorre, se il ***server*** dovrà eseguire qualche **applicazione** (come un *database*) e quali sono le caratteristiche dell'uso del ***server***.

I dischi

I fattori che possono influenzare le prestazioni del disco sono molteplici. Le stesse caratteristiche dei dischi sono molto importanti. La velocità di rotazione è quella con cui ruotano i singoli piatti del disco. Una velocità di rotazione elevata è opportuna quando si usano applicazioni che richiedono l'uso di dati a flusso continuo, come quelli audio o video.

È opportuno tenere presente anche il numero di dischi che viene utilizzato. Se servono 8 *Gbyte* di spazio su disco, l'acquisto di un singolo *drive* di questa capacità non è la soluzione migliore in termini di prestazioni e di *fault tolerance*. Una soluzione migliore potrebbe essere quella di usare quattro dischi da 2 *Gbyte*; con un numero maggiore di dischi ci sono più piatti in rotazione ed è quindi possibile scrivere una maggiore quantità di dati allo stesso tempo. È importante anche il modo in cui vengono organizzati i vari dischi. Lo schema RAID 0 non offre la *fault tolerance*, ma permette di ottenere prestazioni molto valide. Lo schema RAID 1 offre la *fault tolerance* e buone prestazioni, ma è caratterizzato da un costo più elevato.

Il *controller* è un ulteriore componente del sottosistema a disco. L'implementazione più comune è costituita da un *controller* posto all'interno del ***server*** e collegato ai dischi tramite un *bus* SCSI. Il *controller* usa tipicamente una *cache on-board*, che aiuta ad aumentare le prestazioni mantenendo nella *cache* i dati usati più di recente, eliminando così la necessità di usare le unità a disco per recuperare i dati. Le dimensioni della *cache* possono variare da meno di 1 *Mbyte* a più di 64 *Mbyte*.

Strumenti di *monitoring*

Sul mercato sono disponibili molti strumenti per identificare i colli di bottiglia.

La procedura di identificazione dei colli di bottiglia è più un'arte che una scienza esatta. Tutti i sistemi operativi commerciali forniscono qualche serie di *tool*: NT offre il suo ***Performance Monitor***, mentre *NetWare* usa il *Monitor* NLM. Quasi tutti questi *tool* che accompagnano i sistemi operativi offrono funzionalità a livello di base. Per ottenere funzioni e informazioni più dettagliate, bisogna tuttavia rivolgersi ai *tool* di terze parti.

Negli ambienti *Unix*, molti amministratori usano i *tool* forniti con il sistema operativo centrale. *Utility* come `ps` e `vmstat` vengono usate piuttosto comunemente. Per *Sun Solaris* è disponibile una popolare *utility* chiamata `iostat`, che offre valide informazioni sulle prestazioni di *I/O*.

Tutti i *tool* dedicati alle prestazioni dovrebbero consentire di misurare le prestazioni del ***server*** con l'andare del tempo. Non è sufficiente prendere soltanto qualche campione, ma è necessario misurare le prestazioni entro un certo periodo di tempo, in modo da disporre di una base di misura.