Goodus 기술노트 [24 회]
Solaris System Performance Monitoring

Author	**유광복, 조연철**
Creation Date	**2007-09-30**
Last Updated	**2007-10-04**
Version	**1.0**

Version	변경일자	변경자(작성자)	주요내용
1	2007-10-04	유광복, 조연철	문서 최초 작성
2			
3			

Goodus
Breathing Life into Technology

Contents

1.1. System Performance Analysis 란? 3
1.1.1. 시스템 관리자의 역할 3
1.1.2. 시스템 성능측정의 목적 3
1.1.3. 시스템 성능측정 요소 4
1.2. System Performance Analysis Tools 6
1.2.1. sar(System Activity Report) 6
1.2.2. vmstat(Virtual Memory Statistics) 8
1.2.3. prstat 10
1.2.4. iostat(Input Output Statistics) 13
1.2.5. netstat(Network Statistics) 15
1.2.6. top Utility 17
1.2.7. Parm Tool 19
1.3. 부 록 25
1.3.1. Performance Monitoring 성능 측정 기준 25
● Memory Rule 25
● CPU Rule 26
● DISK I/O Rule 26
● Network Rule(Ethernet collision) 26
1.3.2. Solaris News 27
◆ ZFS 기능 28
◆ ZFS volume 지원 28
◆ zpool 구성(raid 0, raid 1, raid 0*1, raid 5) 28
◆ ZFS 구성(Zeta File System 구성) 29
◆ ZFS mount, umount 30
◆ ZFS mount point 변경 30
◆ ZFS 속성 변경 30
◆ zpool status check 30
◆ zfs, zpool 제거 31
◆ zfs iostat check 31
◆ 기타 명령 31

2. Pro-Active Tuning Service 32
2.1. 실제 사용자(End-User) 관점의 응답시간 튜닝 32
2.2. 최상의 성능 상태로 비즈니스 고가용성을 유지 32
2.3. Knowledge Transfer 33
2.4. Tuning 범위 확대 33
2.5. 기대효과 34
2.5.1. 재무적 관점 34
2.5.2. 서비스 관점 34
2.5.3. 사용자 관점 35
2.5.4. 혁신적 관점 35
2.6. 제안 Package 별 가격 36
2.6.1. DBMS Tuning 36
2.6.2. DBMS Tuning+ APM + NA 36
2.6.3. Server (H/W, OS) Tuning + Storage 재구성 컨설팅 + DBMS Tuning 36
2.7. Package 설명 37
2.8. Promotion 38

1. Solaris System Performance Analysis

1.1. System Performance Analysis 란?

시스템 성능측정은 시스템 상태를 분석하는 전체 과정을 포함한다. 보통 시스템 성능측정은 CPU, 메모리, 디스크와 네트워크의 사용량에 국한하는 경향이 많으나, 진정한 성능측정이란 정확한 목적과 계획을 가지고, 그에 맞는 절차를 정확히 이해하고 수행하여야 하며, 그렇게 이루어질 때 효과적인 결과를 도출할 수 있다.

이 문서에서는 효율적이고 목적에 맞는 성능측정을 할 수 있는 유용한 명령어 / tools 의 소개와 사용방법 및 결과 분석 방법에 대해 알아보고자 한다.

1.1.1. 시스템 관리자의 역할

시스템 성능측정의 주요 목적은 시스템을 얼마나 효율적으로 사용하는가에 있다. 시스템의 효율적 사용과 최적화는 시스템 관리자의 몫이다. 그래서, 시스템 관리자는 시스템을 효율적으로 관리하고, 이를 적절히 유지하기 위해 운영체제 설치, 새로운 사용자 등록, 시스템 백업과 보안에 대한 정책, 네트워크 서비스, 기타 다른 하드웨어의 정상동작 여부 확인, 장애의 신속한 대처 및 방안 등을 수행한다.

시스템 관리자는 root 라는 사용자 계정을 가지며 시스템 하드웨어에 대해 절대적인 권한을 가진다. root 권한을 갖는다는 것은 많은 책임이 따르고, 시스템에 대해 다음과 같은 주요 업무를 수행한다.

- **시스템 시작(Start)**
- **사용자 어카운트 등록 및 삭제, 수정 등의 작업**
- **다른 사용자 메일이나 파일들을 열람 가능**
- **효율적인 운영에 대한 방안 모색**
- **하드웨어 유지 보수(신규, 추가, 장애시 교체 등)**
- **시스템 백업 및 필요에 의한 복구(restore)**
- **시스템 상태 감시(monitoring) : 성능 측정**
- **프로세스들의 상태**
- **신규 업그레이드 된 패치 적용**
- **사용자가 원하는 소프트웨어 설치 및 업그레이드**
- **시스템 셧다운**
- **서비스를 위한 환경 구성 및 유지**

시스템 관리자는 시스템 처리 능력을 향상시키는 것을 목표로하고 있으며, 최종적으로는 시스템의 성능 향상으로 ROI(Returen on Investment : 투자 수익)를 극대화 하는데 중점을 두고 시스템을 사용하는 다양한 사용자들에게 같은 성능을 동시에 느끼게 해야 한다.

1.1.2. 시스템 성능측정의 목적

일반적으로 시스템 성능측정은 효율적인 시스템구성과, 성능향상이라는 2 가지 목적을 가지고 실시된다.

◆ 효율적인 시스템 구성

현재 시스템에서 운영중인 여러 작업(task) 중에서 불필요한 작업들을 제거하여 현재의 성능 보다 더 나은 시스템을 구성하기 위함이다. 만약, 시스템이 어느 특정한 업무를 처리하는데 시간이 오래 걸려 결과가 늦게 나온다면 빠른 응답을 요구하는 사용자의 기대를 충족시키지 못한다. 그

래서, 성능저하의 요인을 제거하여 안정적이고 효율적인 시스템을 구현하게 된다.

◆ 성능 향상

하드웨어적인 업그레이드 없이 시스템 성능이 월등히 향상된다면, 이것은 적은 비용으로 최대의 효과를 보는 것과 같다. 하나의 프로세스를 통하여 결과를 보기까지 10 분이나 걸리는 작업이 2 분으로 단축 되었다면, 이거은 5 배의 성능 향상이 아니라 그 이상의 효과를 볼 수 있을 것이다. 즉, 불필요하게 증설을 해야 하는 것을 해소 할 수 있다. 그리고, 시스템 가용성에 대한 수치 이해는 향후 시스템 가용성이 어느 정도 되는지 평가할 수 있다.

저 비용을 통한 최대 효과, 시스템 자원의 사용도 향상, 시스템 사용자에 대한 신뢰도 향상, 시스템 사용량 증가 등을 통해 시스템이 최적의 상태에서 최상의 성능을 발휘하여 효율적 시스템으로 운영하는 것이 시스템 성능측정의 목적이라 할 수 있다.

훌륭히 수행된 성능관리는 시스템 자원의 효율적인 관리와 안정적 서비스가 가능해진다. 자원관리측면으로는 응답 시간을 가지는 요소에 대해 낮은 운영 비용으로 적절한 조치를 취할 수 있으며, 더 많은 처리의 크기를 가질 수 있다. 서비스 측면으로는 서비스 증가와 늘어나는 사용량에 대해 적절한 계획수립을 할 수 있으며, 업무마비로 인한 수익 감소 등의 기업의 경쟁력에 기여를 하여 보다 많은 수익을 창출 할 수 있도록 해준다. 그리고, 장애 발생에 따른 사후처리보다 수요 증가에 따른 장애방지를 위한 예측이 가능하게 된다.

1.1.3. 시스템 성능측정 요소

성능 분석 대상으로는 크게 소프트웨어와 하드웨어 요소로 구분한다. 소프트웨어 요소로는 운영체제, 사용자, 사용자 프로그램(=어플리케이션)이 있으며, 작업부하를 발생하는 어플리케이션 특성에 관한 정보가 여기에 포함된다. 하드웨어 요소로는 물리적인 3 대 요소인 CPU, 메모리, I/O 장치가 있으며, 시스템 성능측정은 사용자가 운영하는 어플리케이션이 적용된 시스템의 하드웨어 요소들을 분석하는 것을 말할 수 있다.

다음은 소프트웨어 요소들을 구분한 것으로 크게 4 개의 부분으로 나눈다.

◆ OS 부분(운영체제 부분)

OS 는 하드웨어를 움직이게 하는 가장 기본적인 소프트웨어 요소이다. 이 기본적인 요소가 적절히 구성이 안되어 있다면, 다음의 DBMS, 어플리케이션, 네트워크 부분 등이 정상적인 성능을 발휘할 수 없을 것이다. 사용자 시스템의 서비스 환경과 운영체제의 충돌에 의해 생긴 문제는 대부분의 운영체제 제조회사가 철저한 코어 분석(Core Analysis)를 통해 수정 또는 업그레이드, 패치 등을 통해 배포하고 있다.

◆ DBMS(Database Management System) 부분

데이터베이스는 운영체제에 맞게 설계되어야 하며, 튜닝은 DBMS 에서 제공하는 튜닝 툴 등을 사용하여 측정한다.

◆ 어플리케이션(Application) 부분

운영체제를 근간으로 하여 서비스를 하기 위한 다양한 프로그램이 될 수 있다. 현재의 시스템에 맞지 않게 구성된 프로그램은 시스템 자원을 낭비하여 성능을 저하시키는 요인이 된다. 거의 모든 시스템 튜닝이라면 어플리케이션이 실행된 시스템의 전체 상태를 분석하는 것이라 할 수 있다. 그리고, 개발공학과 사용자 환경을 충분히 검토하고 적용되어야 한다.

◆ 서버와 네트워크 부분

시스템의 능력은 한정적이다. 그러나, 이 제한적인 능력에 비하여 더 많은 서비스를 요구하거나

업무를 요청하게 된다면 시스템은 너무 지쳐버려 요청에 대하여 미처 응답을 하지 못하거나, 설사 응답을 하더라도 아주 느리게 되며, 심할 경우 시스템은 동작을 멈춰 버릴 수도 있다. 그래서, 상호 작용과 수용 능력 등이 고려되어야 한다.

하드웨어 부분은 주로 시스템을 구성하고 있는 물리 요소들에 대한 측정이라 할 수 있다. 하드웨어 요소로는 크게 디스크, 메모리, CPU 로 본다.

◆ CPU(Central Processing Unit : 중앙처리장치)

CPU 는 명령어나 프로그램을 실행한다. 사용자의 요청이나 작업에 대해 CPU 는 처리하게끔 되어 있다. 빠른 CPU 를 장착하고 있다면 빠른 처리를 하게 된다. CPU 에 대한 튜닝을 하려면 프로세스, 쓰레드, CPU 스케쥴링 등의 요소들을 꼼꼼히 따져보아야 할 것이다. “sar –u” 나 “vmstat”를 사용하여 CPU 사용을 측정할 수 있다. “uptime” 은 시스템 부팅 이후로 현재까지의 평균적인 CPU 부하를 측정할 수 있다. 그리고, “/usr/bin/ps –elf”, “/usr/ucb/ps –aux”는 CPU 를 가장 많이 사용하고 있는 프로세스를 추적할 수 있다. 실시간으로 CPU 를 가장 많이 사용하고 있는 프로세스를 추적할 수 있다. 실시간으로 CPU 를 가장 많이 사용하는 것을 추적하기 위해 top 이라는 툴을 사용하기도 한다. ps 명령어의 확장형인 prstat 를 사용할 수도 있다.

◆ 메모리(Memory)

명령어와 프로그램이 CPU 에서 실행되기 위해서는 메모리를 거쳐야 한다. 메모리는 CPU 에서 실행될 프로그램이 매번 이용할 뿐만 아니라 실행된 프로세스들을 저장해두기도 한다. 이것은 해당 프로세스가 재 사용될 때 디스크에서 메모리로 적해 할 필요 없이 바로 싱행을 할 수 있어 시스템의 성능을 향상시키는 아주 중요한 요소라 할 수 있다. 시스템에 많은 메모리가 있다면 이것 또한 시스템의 성능을 향상 시킬 수 있다. 그러나, 메모리는 프로그램에 의해 낭비되며, 가격 또한 고가인 요소이어서 사용자에게 상당한 부담이 된다. 시스템 튜닝에 있어서 가장 복잡하고 어려운 요소라 할 수 있다.

◆ Disk I/O(Input/Output)

Disk I/O 는 디스크라는 저장장치를 사용하기 때문에 시스템에서 가장 많은 병목이 발생하는 부분이다. 디스크 운영 처리속도는 메모리에 비해 현저히 떨어지기 때문에 대단히 큰 입출력 작업은 입출력 작업이 완료 될 때까지 많은 시간을 기다려야 한다. 입출력 튜닝은 iostat 라는 명령어를 사용하며, “iostat –D”는 각 디스크 사용에 대해 퍼센트(%)로서 보고한다.

Disk I/O 는 데이터의 속도뿐만 아니라 저장되는 데이터의 안정성과 신뢰성에 영향을 미친다. 실패한 CPU 나 메모리는 교체하면 그만이지만 디스크 장애는 저장된 내용을 잃어 버릴 수 있어, 오히려 CPU 나 메모리보다 중요도는 더 높다고 할 수 있다. 그래서, 대부분의 중요한 데이터들은 RAID 라는 기법을 사용하여 데이터의 신뢰성과 안정성을 향상시키고 있다.

◆ 네트워크(Network)

네트워크 구성은 어렵고 복잡하다. 그러나, 시스템 측면에서의 네트워크 튜닝은 오히려 간단하다. 네트워크 튜닝은 인터페이스의 에러유무, 대역폭(Bandwidth)이 충분한지, 또는 패킷(packet)의 충돌 등에 관한 내용이 된다. CPU, 메모리, 디스크 I/O 가 아무리 훌륭하여도 네트워크 속도가 느리면 서비스 이용자는 답답할 것이다. 이 경우 네트워크 환경만 개선하면 훌륭한 서비스를 할 수 있을 것이다. 네트워크 튜닝은 netstat 를 사용하며, ndd, ifconfig, ping, route, tracroute 등의 명령어를 사용하여 네트워크 관련 각종 상태들을 확인하고 측정할 수 있다.

성능측정은 하드웨어 요소들의 사용과 부족 여부, 병목 현상이 발생하는 부분을 찾아 적절한 조치하기까지의 모든 행위가 포함된다. 시스템 운영환경에 따라 분석결과는 달라지기 때문에 정확한 분석은 어렵다. 예를 들어, CPU 사용이 많아서 프로세싱(processing)을 정상적으로 할 수 없는 경우가 있는데, 아래의 원인들이 있을 수 있다.

- CPU 자원의 부족

- 메모리의 부족
- 많은 디스크 입출력
- 많은 네트워크 패킷의 처리
- 프로세스 불필요한 CPU 자원 낭비
- 기타

이것은 CPU 성능저하의 원인이 여거 가지가 있을 수 있음을 의미한다. 이 중에서 어떤 부분이 CPU 의 자원을 낭비하는지 시스템 관리자는 찾아야 한다. CPU 낭비를 가장 많이 일으키는 요인을 메모리, 디스크, 네트워크, 프로세스 등 전 영역에 걸쳐 튜닝해야 하는 것이다. 이것은 CPU 가 성능측정의 최상위에 있는 부분이어서, 그 아래에 있는 모든 요소들은 CPU 의 영향을 받는다 할 수 있다. 반대로, 하위에 있는 요소들은 CPU 에 직접적인 영향을 주지는 않지만 관련 프로세스들은 하위 요소들의 작업이 완료 될 때까지 기다려야 한다. 특히, 대형 데이터베이스 시스템이나 많은 입출력을 처리하는 시스템에서는 응답속도가 느리게 나올 수 있다. 그래서, RAID 같은 메커니즘을 사용한다.

1.2. System Performance Analysis Tools

Solaris 는 기본적인 튜닝 툴들을 제공하고 있다. 이들 툴들은 수많은 옵션과 기능들이 있다. 옵션들에 따라 결과가 달라지며, 사용자들은 수많은 옵션의 사용과 결과에 대한 이해가 필요하다. 그래야만 시스템 성능을 정확하게 분석할 수 있기 때문이다. 그래서, Solaris 에서 제공하는 성능측정 툴들에 대해 설명을 하고, 성능과 기능적인 면에서 뛰어난 일부 소프트웨어에 대한 개략적인 설명을 하고자 한다.

1.2.1. sar(System Activity Report)

sar 는 system Activity Report 의 약자로서 시스템 활동에 대해 16 가지의 다양한 형식으로 측정할 수 있다. 시스템 활동 정보는 사용자가 요청 할 경우 응답하며, 운영체제는 시스템에서 발생하는 행위를 지속적으로 확인하기 위한 여러 개의 카운터(counter)를 포함한다. 카운터는 CPU 사용의 효율성, 버퍼(buffer) 가용성, 디스크나 테이프 I/O 활동성, TTY 디바이스 사용, 시스템 요청의 스위칭(switching)과 활용성, 파일 접근, 큐(queue), 프로세스간 통신과 페이징에 대한 것을 포함한다.

다음은 sar 명령어의 일반적인 형식이다.

```
sar [-ubdycwaqvmpgrkA][-o filename] t [n]
sar [-ubdycwaqvmpgrkA] [-s hh:mm][-e hh:mm][-i sec][-f file]
```

t 는 측정시간 간격(초(second) 단위), n 은 반복 횟수를 의미한다. sar 는 시스템에서 발생하는 다양한 활동들에 대한 데이터를 주어진 시간 간격과 횟수만큼 표준출력 하고, 마지막에는 출력 횟수에 대한 전체 평균을 출력한다. -o 옵션을 사용하면 사용자가 임의의 파일로 저장할 수도 있으며, -f 옵션을 사용하면 저장된 파일의 내용을 확인 가능하다.

예제) 아무런 옵션 없이 sar 를 사용하면

```
sf-a [/]# sar
sar: can't open /var/adm/sa/sa02
No such file or directory
```

라는 에러메시지가 출력되는데, 원본 데이터를 입력받을 /var/adm/sa 라는 디렉터리에 "sa*dd*" 라는

파일이 없기 때문이다.

다음과 같이 sar 데이터의 출력을 /var/adm/sa/sa02 라는 파일로 저장을 한 다음 옵션 없이 sar 를 실행하면 /var/adm/sa/sa02 라는 파일을 입력 데이터로 사용한다. 숫자가 02 라고 표시된 것은 현재 일이 2 일이기 때문이다.

```
sf-a [/]# sar -o /var/adm/sa/sa02 1 5
SunOS sf-a 5.10 Generic_118833-17 sun4u      10/02/2007
00:03:42    %usr    %sys    %wio   %idle
00:03:43      35      24       0      41
00:03:44      12      28       0      60
00:03:45       6      16       0      78
00:03:46       8      15       0      77
00:03:47       4      10       0      86

Average       13      18       0      69

sf-a [/]# sar
SunOS sf-a 5.10 Generic_118833-17 sun4u      10/02/2007

00:03:42    %usr    %sys    %wio   %idle
00:03:43      35      24       0      41
00:03:44      12      28       0      60
00:03:45       6      16       0      78
00:03:46       8      15       0      77
00:03:47       4      10       0      86

Average       13      18       0      69
```

만약, /var/spool/cron/sys 에서 sa1, sa2 가 동작하고 있다면 /var/adm/sa/sa*dd* 형식의 파일이 존재할 것이다. 이때 sar 를 옵션 없이 실행하면 하루 동안의 시스템 CPU 활동에 대한 내용을 확인 할 수 있다. 이것은 하루 동안 시스템 사용에 대한 결과와 평균을 확인 할 경우 매우 유용하다.

예제) CPU 활동에 대하여 1 분 간격으로 10 분 동안 측정. 이때 측정치는 /tmp/cpu_sar 라는 파일로 출력한다.

```
sf-a [/]# sar -o /tmp/cpu_sar 60 10
```

60 초 간격으로 10 회 측정하라는 의미이며, 그 결과는 /tmp/cpu_sar 라는 파일로 저장된다. cpu_sar 라는 파일은 텍스트가 아닌 바이너리(binary) 파일로서 vi 같은 편집기로 내용을 볼 수 없다. 이 파일의 내용을 보기 위해서는 sar 의 -f 옵션을 사용한다.

```
sf-a [/]# sar -f /tmp/cpu_sar
```

◆ sar 에서 사용하는 주요 옵션

옵 션	기 능
-A	sar 에서 주어지는 모든 옵션의 결과를 보고 – abcdgkmpqruvwy

-f *filename*	sar 의 측정 내용이 파일로 저장될 경우 저장된 내용을 읽기 위해 사용하는 옵션이다. 일일(daily) 데이터는 /var/adm/sa/sadd 라는 파일로 저장되며, sar 에서 -o 옵션을 사용하면 sar 결과를 저장할 수 있는데, 이들 파일들을 읽으려고 할 경우 이 옵션을 사용한다.
-o *filename*	sar 결과를 파일로 저장하며, 파일은 2 진 바이너리(binary) 형식으로 저장된다. 레코드 형식으로 저장되며, 각 레코드에는 시간을 확인하는 태그가 포함된다.
-u	CPU 활동성에 대한 보고, sar 에서 아무런 옵션 없이 사용할 경우와 결과가 같다. 결과는 %로서 출력되며, 각 CPU 의 평균적인 값이 된다. 입출력 대기는 CPU 보다는 시스템 입출력에 따라 정의된다. %usr : CPU 가 사용자 레벨(또는 프로그램)을 실행하는 CPU 사용 시간의 % %sys : CPU 가 시스템(또는 커널)레벨을 실행하는 CPU 사용 시간의 % %wio : CPU 가 I/O 를 완료할 때까지 기다리는 유휴시간의 % %idle : 과도한 입출력 없이 순수하게 CPU 가 유휴 시간의 %

1.2.2. vmstat(Virtual Memory Statistics)

vmstat 는 Virtual Memory Statistics 의 약어이며, 커널 스레드(kernel thread), VM(가상 메모리 : Virtual Memory), 디스크, 트랩(trap), 그리고, CPU 활동과 관련된 가상메모리 상태를 보고한다. 출력 값들은 프로세서 수에 따른 합계와 평균으로서 표시된다. vmstat 는 크게 5 가지 형식으로 출력되며, 이를 통해 프로세스, 인터럽트, CPU 캐시(cache), CPU 활용, VM 과 스왑핑(swapping) 활용, 디스크 활용 등의 내용을 측정할 수 있다. sar 는 한 번에 하나의 요소에 대해서만 측정을 할 수 있지만, vmstat 는 동시에 여러 가지 요소를 볼 수 있는 장점이 있다. 시스템 모든 요소들을 운영하기 위해 가상메모리 영역에서 각 이벤트가 관리된다. 그래서, 가상 메모리에서 발생하는 모든 이벤트들을 종합적으로 분석한다면 현재 시스템 상황을 쉽게 판단할 수 있다. vmstat 는 이러한 요건을 충분히 만족시켜 준다고 보면 될 것이다.

```
vmstat [-cipqsS] [disk ...] [interval [count]]
```

예제) 5 초 간격으로 시스템에서 발생하는 이벤트들을 종합적으로 측정하려고 할 경우

```
# vmstat 5
 procs     memory            page            disk          faults      cpu
 r b w   swap  free  re  mf pi po fr de sr dd f0 s0 --   in   sy   cs us sy id
 0 0 0 1093808 69304  1   4  2  0  0  0  0  0  0  0  0  318   51   48  1  0 99
 0 0 0 1091440 69720  0   1  0  0  0  0  0  0  0  0  0  302    7   36  0  0 100
 0 0 0 1091440 69720  0   0  0  0  0  0  0  1  0  0  0  304    6   37  0  0 100
```

5 초 간격으로 vmstat 결과를 반복 출력한다. 결과는 크게 proc, memory, page, disk, faults, cpu 라는 6 개로 구분되며, vmstat 결과의 각 필드(field)에 대한 의미는 다음 표와 같다.

항 목	내 용
procs(Solaris 9 이상 에서는	각 상황에 따른 커널 쓰레드의 수를 출력하며, 다음 3 가지 상황에 따르는 프로세스

«kthr»로 표시)	수를 출력한다. r : 런큐(run queue)에 쌓인 프로세스의 수를 나타내며, 여기에 숫자가 높다는 것은 CPU 에서 처리되기 위해 기다리는 프로세스가 많음을 의미한다. 즉, 현재 CPU 작업이 많다는 것을 의미한다. 보통 이의 상태에 따라 CPU 의 바쁨 정도를 추적할 수 있다. b : 페이징 등과 같이 I/O 자원에 대해 블록(block)되는 프로세스의 수(waiting)를 말한다. I/O 에 있어 CPU 자원을 할당 받지 못해 블록화 된 프로세스의 수를 의미한다. w : 스왑 아웃(Swap out)되는 프로세스의 수로서, 실행 가능한 실행 큐에는 쌓이지 않고 바로 스왑 아웃되는 프로세스를 의미한다. 이를 통해 물리적 메모리의 부족 여부를 판단할 수 있다.
memory	가상 메모리와 실제 메모리의 가용에 대한 것을 출력 swap : 현재 가용 가능한 스왑 공간의 크기(Kbytes) free : 자유 목록(free list)의 크기(Kbytes)로서 자유 메모리가 현재 운영중인 물리적인 메모리의 6%보다 계속적으로 적게 나오면 가용할 수 있는 메모리가 부족하다는 것을 의미하며, 이것은 시스템 전체 병목(Bottleneck)이 발생 할 수 있음을 의미한다.
page	페이지 실패(fault)와 페이징(paging) 활동에 관한 정보를 출력 re : 페이지 재 사용 - -S 옵셥을 사용하면 이 필드는 si 로 수정 mf : 최소한의 페이지 실패 - -S 옵션을 사용하면 이 필드는 so 로 수정 pi : 페이지 인 하는(paged in) 페이지 수(단위 : Kbytes) po : 페이지 아웃되는(paged out) 페이지 수(단위 : Kbytes) fr : 자유(free) 페이지 수(단위 : Kbytes) de : 메모리 부족한 크기를 미리 예측(단위 : Kbytes)한다. 부족한 메모리에 대해 스왑아웃을 하기 위해 스왑 인을 막게 된다. sr : 클럭 알고리즘(clock algorithm)에 의해 스캔(scan)되는 페이지 수로서, 가용할 수 있는 메모리가 부족할 경우 활성화 되어 있는 페이지의 데몬의 수를 나타낸다. 즉, sr 이 크다는 것은 이에 비례하여 메모리가 부족하다는 것을 의미한다.
disk	디스크 운영에 대한 수를 출력한다(초 단위). 4 개의 디스크에 대한 상태를 각각 보여준다. 디스크는 유형에 맞는 이름이 출력된다(s=SCSI, I=IPI 등)
faults	트랩(trap)/인터럽트(interrupt) 비율을 출력(초 단위)

	in : 인터럽트 수 sy : 시스템 요청(system call) cs : CPU 문맥 교환(Context switches)
cpu	CPU 사용에 대한 것을 %로 구분하여 출력한다. 여러 개의 CPU 로 구성이 될 경우에는 CPU 개수에 따른 평균 값이 출력된다. us : 사용자가 사용한 CPU 시간 비율(user time) sy : 시스템이 사용한 CPU 시간 비율(system time) id : 가용 가능한 cpu 시간 비율(idle time)

vmstat 는 주로 이벤트에 따른 CPU 와 메모리를 측정 할 경우 유용하다. 이벤트는 주로 프로세스의 요청과 응답에 의해 발생하기 때문에 CPU 가용성과 메모리의 가용성을 분석하기 쉽다. vmstat 는 CPU 와 메모리, 가상 메모리 등의 상태를 한번에 볼 수 있어서 sar 보다 더 많이 사용하고 있다. 프로세스의 상태는 다양하지만 실제 CPU 에서 실행될 때는 커널 쓰레드를 통해 이루어지며, Solaris 환경에서는 쓰레드 단위로 처리한다. 멀티 쓰레드에서는 하나의 쓰레드가 하나의 프로세스처럼 동작하여 시스템 가용성을 향상시켰으며, 멀티 쓰레드가 아닌 프로세스는 하나의 쓰레드가 프로세스로 운영된다. 하나의 프로세스에 몇 개의 쓰레드가 있는지는 "ps -efL"을 사용하여 NLWP 필드에서 확인 할 수 있다.

1.2.3. prstat

prstat 는 현재 활동 중인 프로세스들의 상태를 보고한다. ps 명령어는 한번의 실행으로 한번의 상태만 볼 수 있지만, prstat 유틸리티는 지속적으로 현재 프로세스의 상태를 자동으로 업데이트(update)한다. 출력은 /usr/ucb/ps 와 같이 CPU 사용 시간의 비율을 기준으로 내림차순 정렬 출력한다. 업데이트 시간 간격은 5 초이다.

prstat 사용 형식

```
prstat [-acJLmRtTv] [-u euidlist] [-U uidlist]
        [-p pidlist] [-P cpulist] [-C psrsetlist]
        [-j projidlist] [-k taskidlist]
        [-s key | -S key] [-n nprocs[,nusers]]
        [interval [counter]]
```

◆ prstat 에서 사용하는 주요 옵션

옵 션	기 능
-a	프로세스와 사용자와 고나한 정보를 동시에 보고한다. 사용자 보고 내용은 해당 사용자에서 시작된 프로세스들이 CPU 나 메모리를 어느 정도 사용하고 있는지 요약 정보를 아래에 출력한다.
-n ntop[nbottom]	prstat 의 기본 출력에서 상위, 또는 하위 범위를 끊어서 출력한다. ntop 은 상위에서, nbottom 은 하위를 끊어서 출력한다.
-P cpulist	주어진 목록에서 가장 최근에 CPU 에서 실행되는 프로세스 또는 lwp 만을 출력한다. 각

	CPU는 psrinfo 에 의해서 정수형 값으로 ID를 확인
-s key	주어진 키(key)에 의해 내림차순으로 정렬한다. key는 다음과 같다. cpu : 프로세스가 CPU 가용에 의해 정렬(기본) pri : 프로세스 우선순위에 의해 정렬 rss : 메모리를 점유하고 있는 크기대로 정렬 size : 프로세스 이미지 크기에 의해 정렬 time : 프로세스 실행 시간에 의해 정렬
-u euidlist	주어진 목록에서 효과적인 UID와 연계된 프로세스와 관련된 내용을 출력

예제) prstat 명령어 실행화면

```
sf-a [/]# prstat
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
 18919 ora10     154M  103M sleep   29   10   0:07:20 2.1% java/19
 15228 ora10     175M  162M sleep   59  -20   0:00:58 0.6% ocssd.bin/16
  3018 root      138M  130M sleep  111    -   0:40:47 0.4% lkmgr/1
 14983 root       63M   51M sleep   59    0   0:00:39 0.3% crsd.bin/46
 15145 ora10      35M   26M sleep   59    0   0:00:25 0.3% oclsomon.bin/1
 14769 root     1320K 1136K sleep    0    0   0:00:22 0.3% init.cssd/1
 19085 oracle    551M  535M sleep   59    0   0:19:44 0.2% oracle/1
 12449 root     4856K 4744K cpu2     49    0   0:00:00 0.2% prstat/1
 19089 oracle    422M  410M sleep   59    0   0:16:06 0.2% oracle/1
 19087 oracle    553M  537M sleep   59    0   0:15:15 0.1% oracle/2
 15077 ora10     167M  154M sleep   59    0   0:00:09 0.1% oclsvmon.bin/1
     1 root     2384K 1168K sleep   59    0   0:09:43 0.1% init/1
 19093 oracle    423M  409M sleep   59    0   0:08:45 0.1% oracle/1
Total: 117 processes, 1612 lwps, load averages: 0.57, 0.58, 0.64
```

◆ prstat 출력의 각 필드 요약

항 목	내 용
PID	프로세스의 PID(Process ID)
USERNAME	실제 로그인 한 사용자 이름 또는 실제 사용자 ID(UID)

SIZE	프로세스가 차지하는 전체 가상 메모리의 크기(파일과 디바이스에 연결되는 모든 것을 포함)이며, 단위는 kilobyte(K), Megabyte(M), Gigabyte(G)로 표현된다. 프로세스가 실제 차지하는 물리 메모리 크기를 나타내는 RSS 도 마찬가지이다.
STATE	프로세스의 상태를 나타내며, 다음 중 하나를 의미한다. cpu*N*: 프로세스가 *N*번째 CPU 에서 수행 중 sleep : 잠듦(sleeping), 프로세스가 이벤트가 완료하기 위한 기다림 run : 실행 가능 상태(eunnable). 프로세스가 현재 런큐(run queue)에 있음 zombie : 좀비 상태(zombie state) stop : 프로세스가 멈추어 있는 상태(stopped)
PRI	프로세스의 우선 순위를 나타내며, 클수록 우선 순위가 높다
NICE	우선순위를 계산하기 위하여 사용되는 nice 값
TIME	프로세스에 대해 실행한 시간을 계산
CPU	프로세스에 의해 최근에 사용된 CPU 시간의 퍼센트(%)
PROCESS	프로세스의 이름(실행 파일의 이름)
NLWP	프로세스의 lwp 수

예제) CPU1, CPU2 에서 수행중인 프로세스 중에서 슈퍼유저(root)에 의해서 수행되는 상위 5 개의 프로세스에 대하여 1 회만 출력하시오.

```
sf-a [/]# prstat -u root -n 5 -P 1,2 1 1
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
     4 root        0K     0K sleep   60    -   0:07:39 0.0% cluster/1067
   104 root     2280K  760K sleep   59    0   0:02:12 0.0% in.mpathd/1
   699 root     2328K  872K sleep  100    -   0:00:45 0.0% xntpd/1
  2404 root       15M 1376K sleep   59    0   0:00:47 0.0% scdpmd/13
   408 root       14M  616K sleep  100    -   0:00:00 0.0% clexecd/13
Total: 29 processes, 1169 lwps, load averages: 0.55, 0.56, 0.57
```

마지막에 있는 1, 1 은 각각 count 와 interval 을 나타낸다

예제) CPU 를 과도하게 사용하는 상위 5 개의 프로세스를 출력

```
sf-a [/]# prstat -s cpu -n 5
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
 18919 ora10     154M  103M sleep   29   10   0:08:35 2.1% java/19
 15228 ora10     175M  161M sleep   59  -20   0:01:20 0.6% ocssd.bin/16
```

```
 3018 root       138M  130M sleep  111    -   0:41:01 0.4% lkmgr/1
 14983 root       63M   51M sleep   59    0   0:00:51 0.4% crsd.bin/46
 15145 ora10      35M   26M sleep   59    0   0:00:34 0.3% oclsomon.bin/1
Total: 119 processes, 1613 lwps, load averages: 0.65, 0.58, 0.57
```

1.2.4. iostat(Input Output Statistics)

iostat 는 Input Output Statistics 의 약어로서, 시스템 입출력에 대한 상태를 측정하고자 할 경우 사용된다. I/O 장치(device)에 대한 성능측정 툴이라 할 수 있다. isotat 는 13 가지 형식 지원을 지원하며, CPU 사용도, TTY 활동, 디스크 활동, 디바이스의 에러 상태 등을 확인할 수 있다. vmstat 와 마찬가지로 첫 번째 출력 데이터는 시스템이 부팅한 이후로 현재까지의 평균적인 값이므로 무시해도 된다.

vmstat 에서는 가상 메모리에서 발생하는 이벤트를 가지고 시스템에 대한 전반적인 성능을 측정하였다. 이에 비하여 iostat 는 시스템에서 발생하는 I/O 를 가지고 시스템의 상태를 측정한다. 그래서, iostat 는 처리량(throutghput), 가용성, 큐의 길이(queue length), 전송 비율과 서비스 시간에 대한 내용이 중심이 된다.

```
iostat [-cCdDeElmMnpPrstxz]  [-l n] [T d|u] [disk ...] [interval [count]]
```

◆ iostat 에서 사용하는 주요 옵션

옵 션	기 능
-c	사용자 모드, 시스템 모드, I/O 기다림(waiting), 유휴(Idle) 에서 소비(낭비)되는 시스템 시간의 %를 출력. sar –u 와 비슷한 결과
-d	각 디스크에 대해 초 단위로 전송(read/write)의 수를 Kbytes 로 출력하며, 평균 서비스 시간은 1/1000 초(milliseconds)
-e	장치들의 에러 상태를 요약 출력 – 전체 에러(total errors), 하드에어(hard error), 소프트 에러(냈 error), 전송 에러(transport error)로 구분 출력
-E	모든 장치 에러 상태를 출력
-n	장치 이름을 상세한 형식으로 출력
-p	각 디스크에 대한 파티션(partition) 상태와 장치(device) 상태를 출력
-x	모든 디스크들의 상태를 출력하며, 장치를 표현하는 필드에는 장치이름과 내부적인 번호로서 나타난다. 예를 들어 sd6 는 SCSI 장치에 연결되어 있는 장치로서 6 번은 보통 CD-ROM 장치를 의미한다.

예제) 장치들의 상태를 확장되게 측정하여 출력(3 초 간격으로 5 번만 출력)

```
# iostat -xcn 3 5
     cpu
 us sy wt id
```

```
 1  0  0 99
                  extended device statistics
   r/s    w/s   kr/s   kw/s wait actv wsvc_t asvc_t  %w  %b device
   0.0    0.1    2.2    2.0  0.0  0.0    0.6    6.2   0   0 c0t0d0
   0.0    0.0    0.0    0.0  0.0  0.0    0.0    0.0   0   0 c0t2d0
    cpu
us sy wt id
 0  0  0 100
                  extended device statistics
   r/s    w/s   kr/s   kw/s wait actv wsvc_t asvc_t  %w  %b device
   0.0    0.0    0.0    0.0  0.0  0.0    0.0    0.0   0   0 c0t0d0
   0.0    0.0    0.0    0.0  0.0  0.0    0.0    0.0   0   0 c0t2d0
```

◆ iostat 출력의 각 필드 요약

항 목	내 용
device	device 디스크 이름 r/s : 초당 읽는 문자의 수 w/s : 초당 기록하는 문자 수. r/s, w/s 는 커널이 수행한 읽기/기록하는 횟수 Kr/s : 초당 기록하는 문자의 수(Kbyte) Wr/s : 초당 읽기를 하는 문자의 수(Kbyte) Kr/s, Wr/s 은 커널이 수행한 읽기/기록하는 데이터 양이라 할 수 있다. wait : 서비스(큐의 길이)에 대하여 전송을 기다리는 평균 수 actv : 실제적으로 전송 서비스 되는 평균 수 svc_t : 평균 서비스 시간(1/1000 초 – milliseconds) %w : 서비스에 대해 전송을 기다리는 시간의 비율(%) 또는 서비스 시간(service time) %b : 디스크의 바쁨(busy)에 대한 시간의 비율(%)
cpu	us : 사용자 모드의 시스템 사용 시간의 비율(%) sy : 시스템 모드의 시스템 사용 시간의 비율(%) wt : 시스템을 사용하기 위하여 기다리는(wait) 시간의 비율(%) id : 시스템 유휴 시간의 비율(%)

iostat 에서 중요하게 분석되는 부분은 %w, %b 라 할 수 있다. iostat 에서 %w 는 대기 큐에서 입출력

요청이 대기하고 있는 시간에 대한 비율이며, %b 는 디스크가 입출력 요청을 처리하고 있는 시간에 대한 비율을 나타낸다.

예제) 각 파티션, 각 디스크 활동 정보

```
iostat -xnp 3
                    extended device statistics
    r/s    w/s   kr/s   kw/s wait actv wsvc_t asvc_t  %w  %b device
    0.0    0.1    2.2    2.0  0.0  0.0    0.6    6.2   0   0 c0t0d0
    0.0    0.0    0.2    0.1  0.0  0.0    0.9    5.0   0   0 c0t0d0s0
    0.0    0.0    0.0    0.0  0.0  0.0    3.7    9.4   0   0 c0t0d0s1
    0.0    0.0    0.0    0.0  0.0  0.0    0.0    0.0   0   0 c0t0d0s2
    0.0    0.0    2.0    1.8  0.0  0.0    0.2    8.3   0   0 c0t0d0s6
    0.0    0.0    0.0    0.0  0.0  0.0    0.0    0.0   0   0 c0t2d0
```

앞 예제의 내용에서 디스크 부분의 측정 내용과 비슷한 유형이며, wsvc_t, asvc_t 라는 내용이 추가되었다. wsvc_t 는 큐에서 기다리는 평균 서비스 시간을 의미하며(단위 : 1/1000 초-milliseconds), asvc_t 는 실제 전송하는 평균 서비스 시간(단위 : 1/1000 초-milliseconds)을 의미한다. 그래서, 디스크에 대한 응답 시간(response time)을 측정하고자 한다면 wsvc_t 와 asvc_t 를 합하면 된다.

1.2.5. netstat(Network Statistics)

netstat 는 네트워크 관련 상태를 측정하기 위한 도구이다. 네트워크 관련내용을 15 가지 형식으로 출력하며, 주요 측정내용은 소켓 상태(socket statistics), 인터페이스 상태(interface statistics), DHCP 정보, ARPdhk 라우팅 테이블(routing table), 스트림(STREAMS) 상태 등이다. 시스템 성능이 좋아도 네트워크 서비스가 약하다면 시스템은 불쌍하게도 느린 응답속도를 가질 수 밖에 없다. 특히, 온라인(Online) 서비스를 하는 경우라면 네트워크는 아주 중요한 요소라 할 수 있다. 그리고, 네트워크 상태를 통해 현재 시스템의 해킹 여부 등을 판단 할 수 있는 근거가 되기도 한다.

```
netstat [-anv] [-f address_family]
        netstat [-g | -p | -s] [-n] [-f address_family] [-P protocol]
        netstat -m
        netstat -i [-I interface] [-an] [-f address_family] [interval]
        netstat -r [-anv] [-f address_family]
        netstat -M [-ns] [-f address_family]
        netstat -D [-I interface] [-f address_family]
```

netstat 는 시스템 네트워크 상황에 따른 옵션들이 사용된다. 각 사용하는 유형들은 다음과 같이 해석된다.

(1) 유형은 각 프로토콜(protocol)에 대해 활동중인 소켓(socket) 목록을 출력한다.

(2) 유형은 다양한 다른 네트워크 데이터 구조(structure)로부터 하나를 선택한다.

(3) 유형의 -m 옵션은 스트림(STREAM) 메모리 상태들을 출력한다.

(4) 유형의 -I 옵션은 네트워크 인터페이스의 상태를 출력한다.

(5) 유형의 -r 옵션은 라우팅 테이블을 출력한다.

(6) 유형의 -M 은 멀티캐스트(multicast) 라우팅 테이블을 출력한다.

(7) 유형의 -D 옵션은 하나 또는 모든 인터페이스에서 DHCP 상태를 출력한다.

◆ netstat 에서 사용하는 주요 옵션

옵 션	기 능
-a	모든 소켓들의 상태, 모든 라우팅 테이블의 목록, 물리적이고 논리적인 모든 인터페이스를 보여준다. 정상적으로는 서버 프로세스들에 의해 사용되는 리스너(listener) 소켓은 보여지지 않는다.
-i	IP 트래픽에 의해 사용되는 인터페이스의 상태를 보여준다. 일반적으로는 물리 인터페이스의 상태를 보여준다. 인터페이스 마다 패킷 전송, 수신 상태 등을 출력한다. -a 옵션과 같이 사용하면 논리적인 인터페이스에 대한 정보도 출력할 수 있다.
-n	네트워크 주소들을 번호로서 출력한다. netstat 는 주소들을 보통 기호(symbol)로서 출력을 하며, 이 옵션을 사용하면 IP 주소 형식으로 출력한다.
-r	라우팅 테이블을 출력한다. 보통 인터페이스, 호스트, 네트워크, 기본 라우트(default routers)를 출력하지만, -a 옵션과 같이 사용하면 캐시를 포함하는 모든 라우트들을 출력한다.

예제) 시스템의 기본 인터페이스에 대한 상태를 출력

```
sf-a [/]# netstat -i 1
     input   hme0       output       input  (Total)     output
packets errs  packets errs  colls  packets errs  packets errs  colls
7746219 0     14306875 0     0      28775436 0     41919284 0     0
3       0     2        0     0      111      0     106      0     0
29      0     33       0     0      210      0     217      0     0
8       0     8        0     0      116      0     115      0     0
4       0     3        0     0      122      0     118      0     0
4       0     3        0     0      124      0     113      0     0
2       0     2        0     0      121      0     118      0     0
```

"-I" 옵션은 네트워크 인터페이스에 대한 상태를 보여준다. 위에서는 hme0 인터페이스는 시스템의 기

본 인터페이스로 사용하고 있으며, hme0 이더넷 인터페이스의 입출력 크기를 보여주고 있다. 현재 input packets, output packets 이 있음을 볼 수 있다. 네트워크에서 데이터를 주고 받을 때 패킷(packet) 단위로서 데이터를 전송한다. 패킷의 크기는 전송할 데이터의 크기에 따라 가변적으로 변하며 LAN 의 경우 최대 1500byte (이를 MTU=Maximum Transfer Unit 라 한다)까지 가능하다. 패킷 전송은 TCP 또는 UDP 로 이루어지며 만일 TCP 소켓을 통해 데이터를 전송할 경우라면 1byte 의 데이터 전송 시 최소 55byte(헤더(header) 54byte + 데이터 1 byte)를 전송하게 된다. netstat 에서 보여주는 것은 주어진 시간 간격(웨어서는 1 초 간격으로 하였다)의 차이 값이라 할 수 있으며, 초 단위로 몇 개의 패킷이라고 표현할 수는 없다.

예제) 네트워크 라우팅 테이블에 대한 정보 출력

```
sf-a [/]# netstat -rn

Routing Table: IPv4
  Destination           Gateway           Flags  Ref    Use    Interface
-------------------- -------------------- ----- ----- ------ ---------
172.16.0.128         172.16.0.130         U         1    124  ge0
172.16.1.0           172.16.1.2           U         1    129  ge1
61.250.99.0          61.250.99.232        U         1    588  hme0
61.250.99.0          61.250.99.232        U         1      0  hme0:1
172.16.193.0         172.16.193.2         U         1    124  clprivnet0
224.0.0.0            61.250.99.232        U         1      0  hme0
default              61.250.99.1          UG        1     56
127.0.0.1            127.0.0.1            UH        521585926  lo0
```

1.2.6. top Utility

top 유틸리티는 http://sunfreeware.com 에서 OS버전에 맞는 것을 다운로드 하여 사용할 수 있는 프리(free) 소프트웨어이다. 툴은 언번들(Unbundle) 소프트웨어이므로 기본적으로 실행 프로그램이 설치되는 위치는 /usr/local/bin이다. 설치 후 사용자는 PATH 경로를 설정해 두면 어떤 위치에 있는 실행할 수 있다.

아래는 top 유틸리트의 설치 절차이다. 패키지 압축을 해제한 후 pkgadd 명령어로 설치한다.

```
# gunzip top-3.6-sol7-sparc-local.gz
# pkgadd -d ./top-3.6-sol7-sparc-local
다음과 같은 패키지를 사용할 수 있습니다.
  1  SMCtop     top
                (sparc) 3.6
처리할 패키지(들)를 선택하십시오.(또는 모든 패키지를
처리하려면'all'을 입력하십시오.) (default: all) [?,??,q]: y
```

“y”를 입력하면 top 소프트웨어를 설치한다.

다음은 top 유틸리티를 실행했을 경우의 초기 화면이다.

```
last pid:  7255;  load averages:  0.02,  0.03,  0.03                    10:40:36
36 processes:  35 sleeping, 1 on cpu
CPU states:  100% idle,  0.0% user,  0.0% kernel,  0.0% iowait,  0.0% swap
Memory: 128M real, 66M free, 26M swap in use, 1066M swap free

   PID USERNAME LWP PRI NICE  SIZE   RES STATE    TIME    CPU COMMAND
  7255 root       1  46    4 2960K 1776K cpu      0:00  0.29% top
   303 root      12  58    0 2768K 1936K sleep   17:04  0.03% mibiisa
  ...
  ...
    64 root       5  13    0 2944K  856K sleep    0:00  0.00% picld
```

top 유틸리티는 많은 정보를 한 순간에 볼 수 있다. 평균 부하, 현재 시스템의 전체 프로세스의 수, 동작 현황, CPU IDLE 에 대한 정보, 메모리와 스왑(swap)에 대한 정보를 요약하여 보여준다. 그 밑으로는 기본적으로 CPU 를 많이 점유하는 상위 15 개의 프로세스를 보여준다. 이 툴은 다양한 명령어가 있다. 이들에 대한 도움말을 보고 싶다면 키보드에서 “h”를 치면 다음과 같이 사용할 수 있는 명령어와 도움말을 볼 수 있다.

```
Top version 3.5, Copyright (c) 1984 through 2004, William LeFebvre
A top users display for Unix
These single-character commands are available:
^L      - redraw screen
q       - quit
h or ?  - help; show this text
d       - change number of displays to show
e       - list errors generated by last "kill" or "renice" command
i       - toggle the displaying of idle processes
l       - same as 'i'
k       - kill processes; send a signal to a list of processes
n or #  - change number of processes to display
o       - specify sort order (size, res, cpu, time)
r       - renice a process
```

```
s       - change number of seconds to delay between updates
u       - display processes for only one user (+ selects all users)
```

1.2.7. Parm Tool

이 툴은 국내 준 소프트(Jun soft) 회사에서 개발하여 판매하고 있는 유료 소프트웨어이다. 패키지 이름은 JSparm 이며, www.junsoft.com 웹 사이트에서 다운로드 받을 수 있다(라이센스는 데모로서 1 개월 사용 가능하다) 최신버전은Parm V7.1.1 로 Solaris 2.6 ~ 10 에서 실행되며, 시스템 통계정보를 사용자가 쉽게 볼 수 있도록 하는 패키지이다.

parm 에는 Solaris 명령어 vmstat, iostat, netstat, mpstat, sar 를 통해 수집한 데이터를 사용자가 보기 쉽도록 GIF 파일 포맷의 그래프를 만들어 주는 명령어(gvmstat, giostat, gnetstat, gmpstat, gsar)와 프로세스의 상태를 모니터하는 명령어(psinfo)와 실시간으로 시스템의 통계정보를 모니터하는 툴(perfmon)과 vmstat, iostat, netstat 명령어의 출력을 실시간으로 받아서 그래프를 그려주는 vmmon, iomon, netmon, tcpmon 이 포함되어 있다.

그리고 2 개의 웹 어플리케이션 ParmView 와 ParmClient 가 있다. mmonx 은 시스템에서 1 분 간격으로 통계정보를 수집하여 데이터 파일에 저장하고 ParmView 는 저장된 시스템 통계정보를 읽어들여 웹으로 다양한 형태의 그래프로 만들어 준다. ParmClient 는 실시간으로 시스템 통계정보를 자바 애플릿으로 웹을 통하여 보여준다.

- ◆ ParmView

웹 애플리케이션 ParmView는 mmonx 명령어가 수집한 시스템 통계 데이타를 웹 서비스를 통하여 다양한 형태의 그래프로 보여준다. ParmView를 실행하기 위해서는 먼저 mmonx 명령어에 의해 데이타가 수집되어 있어야 하고, 웹 서버가 /opt/JSparm/parmview 디렉토리를 웹 서비스하여야 한다.
이 디렉토리에는 ParmView를 서비스하는 CGI 프로그램 mrepx.cgi가 있다.

JSparm 패키지를 설치한 후에, /opt/JSparm/etc 디렉토리에 있는 parmview.conf 화일을 /etc 디렉토리에 복사하고, 쉘 스크립트 /etc/rc2.d/S90mmonx을 실행하면, ParmView를 서비스하기 위한 mmonx 명령어와 gwserv 명령어가 실행된다

```
# cd /opt/JSparm/etc
# cp parmview.conf /etc
# sh /etc/rc2.d/S90mmonx start
mmonx[2033] is started
gwserv[2035] is startd on port 9900
```

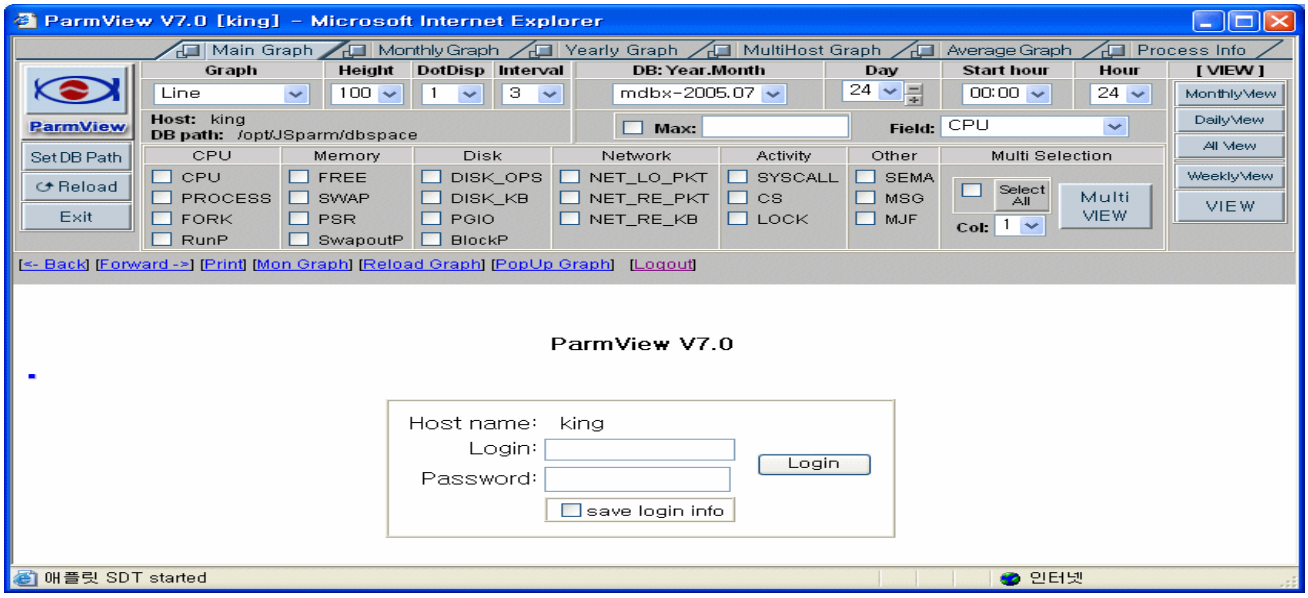
gwserv 명령어가 9900 포트에서 웹서비스를 한다. 포트 번호를 변경하려면, /etc/parmview.conf 화일에서 "port: 9900" 인 라인에서 9900 번 대신 다른 포트 번호를 지정하면 된다.

ParmView 를 실행하기 위해서는 웹 브라우저에서 다음과 같은 URL 를 사용하면 된다.

http://*hostname*:9900

그러면, 다음과 같은 웹 페이지가 화면에 출력된다.

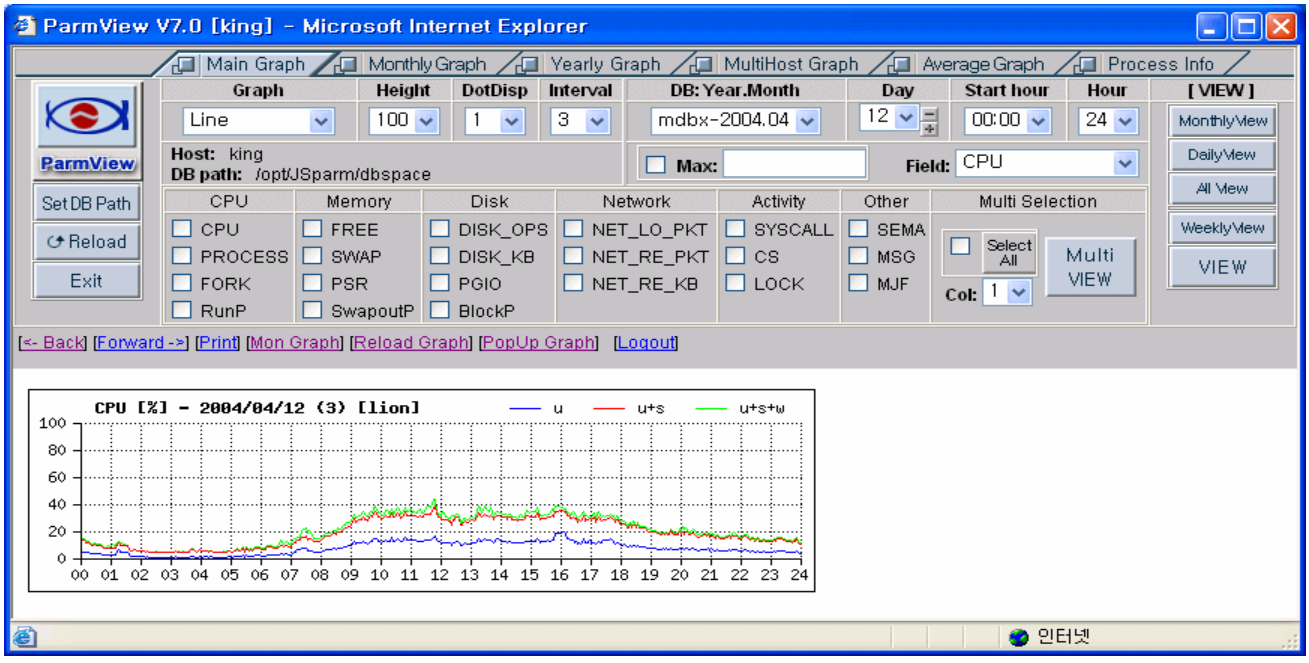
여기서 **[Click this to start ParmView]** 버튼을 클릭하면, 다음과 같은 ParmView 의 로그인 화면이 나타난다.



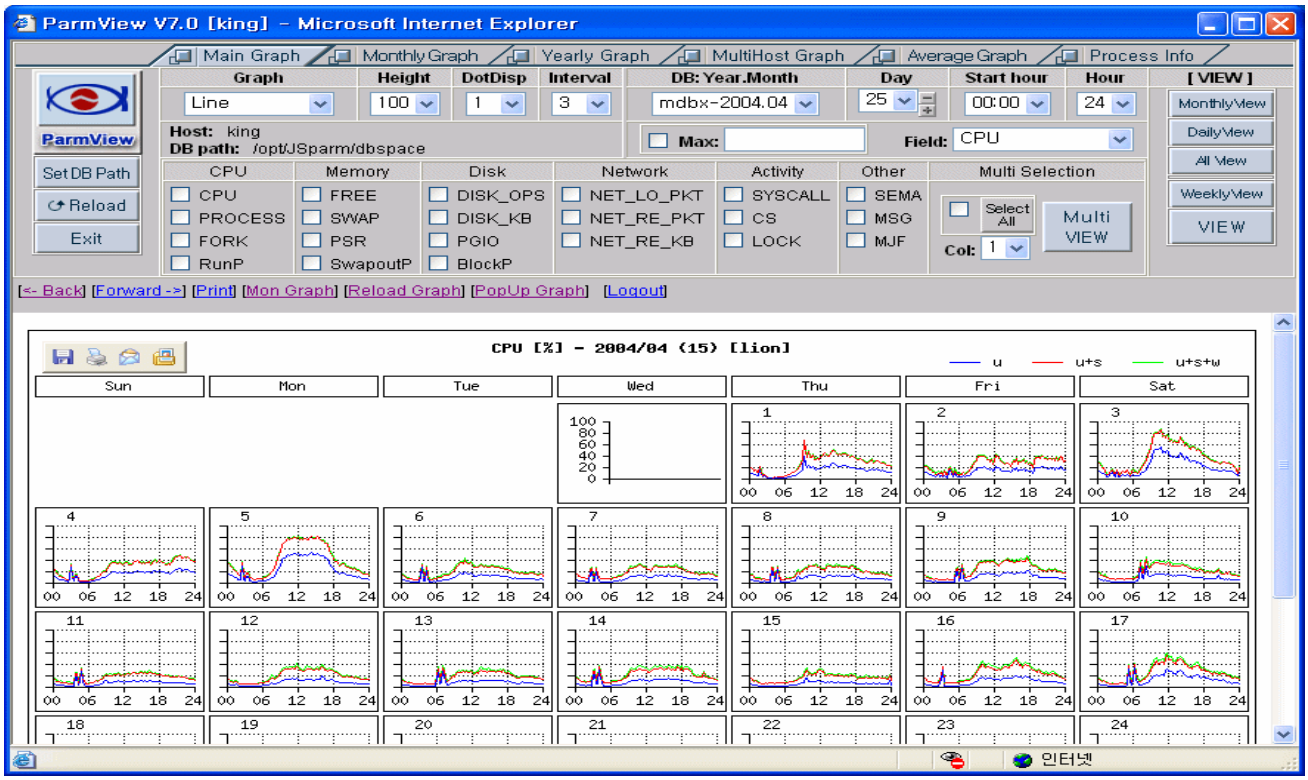
로그인 화면에서 서버 시스템의 사용자명과 패스워드를 넣고 인증을 마치면, ParmView 를 사용할 수 있다. **여기서 사용자명과 패스워드는 암호화되어 서버로 전송된다.**
ParmView 를 **(Exit)** 버튼으로 종료하더라도 로그인된 정보는 웹 브라우저의 쿠키로 남아있다.
웹 브라우저를 완전히 종료시키거나 **[Logout]** 링크를 클릭하여 로그인 쿠키 정보를 제거할 수 있다.

[Save Login Info] 체크박스를 클릭하여 로그인한 후, **[Logout]** 링크를 클릭하지 않고 웹 브라우저를 종료하면, 로그인 쿠키 정보는 시스템에 저장된다. 따라서 시스템을 끄다 껴서 웹 브라우저를 다시 실행하여도 로그인 과정없이 ParmView 를 사용할 수 있다.
로그인 쿠키 정보에는 PC 의 IP 정보가 들어 있기 때문에 PC 의 IP 가 바뀌면 다시 로그인하여야 한다.

다음은 ParmView 를 로그인하여 CPU 사용률 그래프를 본 예제이다.

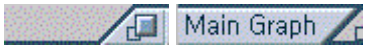


ParmView 에서는 월간 사용률까지 한 화면에 표시하여 체크할 수 있는 기능을 제공한다.



상단의 버튼은 클릭하면 해당 버튼에 맞는 화면으로 변경되거나, 새로운 화면이 팝업된다.

각각의 그래프 버튼은 다음과 같이 두개의 부분으로 나누어져 있다.



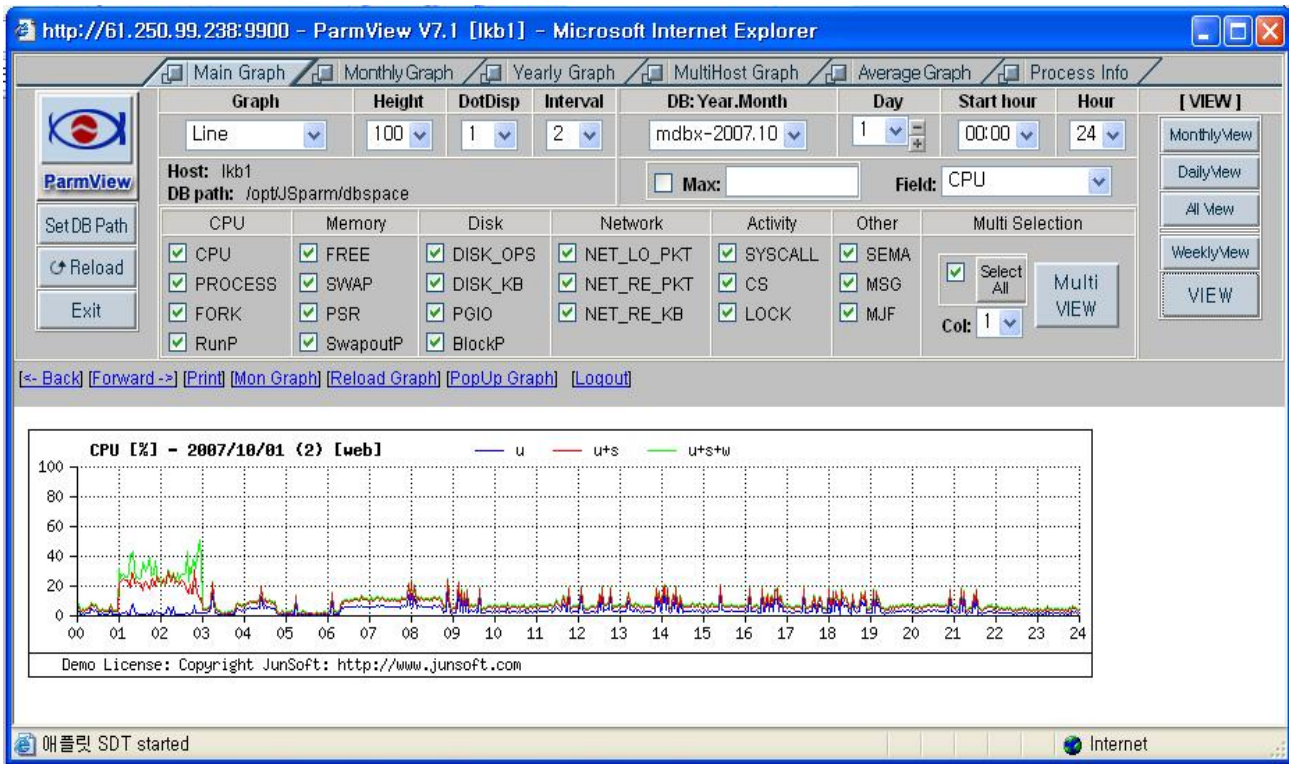
왼쪽 부분을 클릭하면, 해당되는 그래프를 보여주는 화면이 팝업되고, 우측 부분을 클릭하면 현재 화면이 해당되는 그래프 화면으로 변경된다.

[Main Graph]	자주 사용되는 그래프를 만들 수 있는 윈도우를 보여준다.
[Monthly Graph]	월간 그래프를 만들 수 있는 윈도우를 보여준다.
[Yearly Graph]	연간 그래프를 만들 수 있는 윈도우를 보여준다.
[MultiHost Graph]	여러 시스템의 그래프를 하나의 윈도우에서 보여준다.
[Average Graph]	평균 그래프를 만들 수 있는 윈도우를 보여준다.
[Process Info]	mmonx_proc 명령어로 수집된 프로세스 정보를 보여준다.

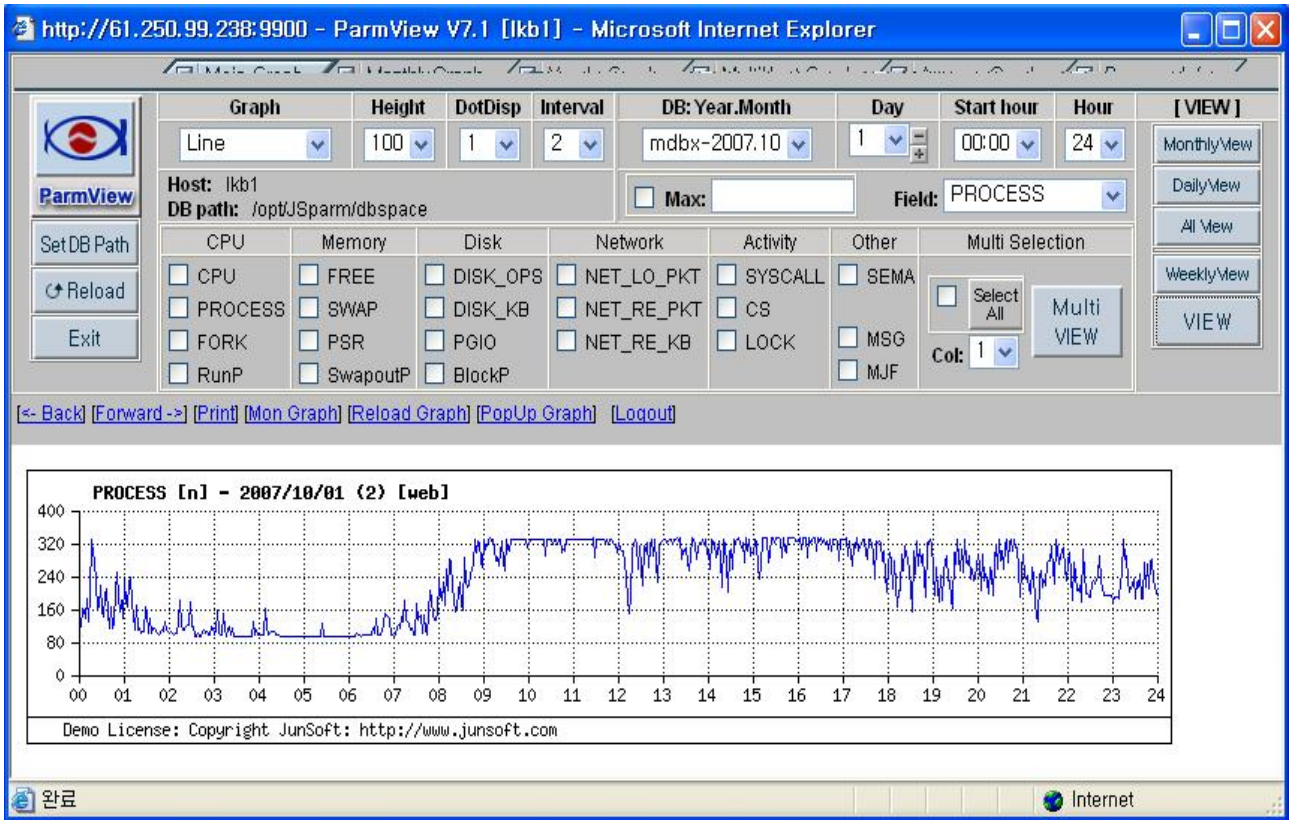
** 자세한 메뉴 설명 및 기타 메뉴는 http://www.junsoft.com/doc/parm 참고

◆ 다음은 00 시청 Web 서버에 대한 모니터링 결과를 ParmView 로 체크한 화면이다.

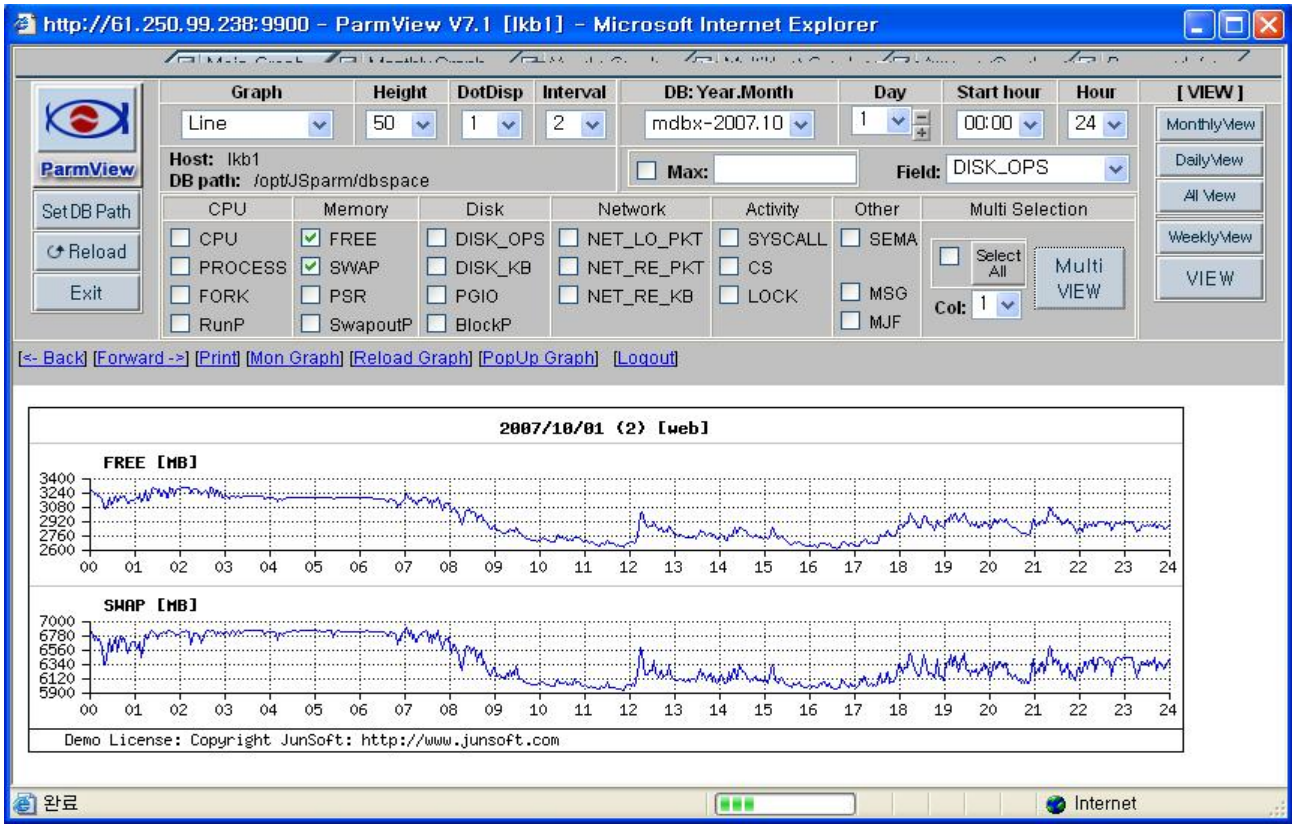
아래는 CPU 사용량 체크 결과로 user 와 system 사용량이 비슷하며, 새벽 01~03 시 사이에 CPU 처리량이 급증하는 것을 볼 수 있다.



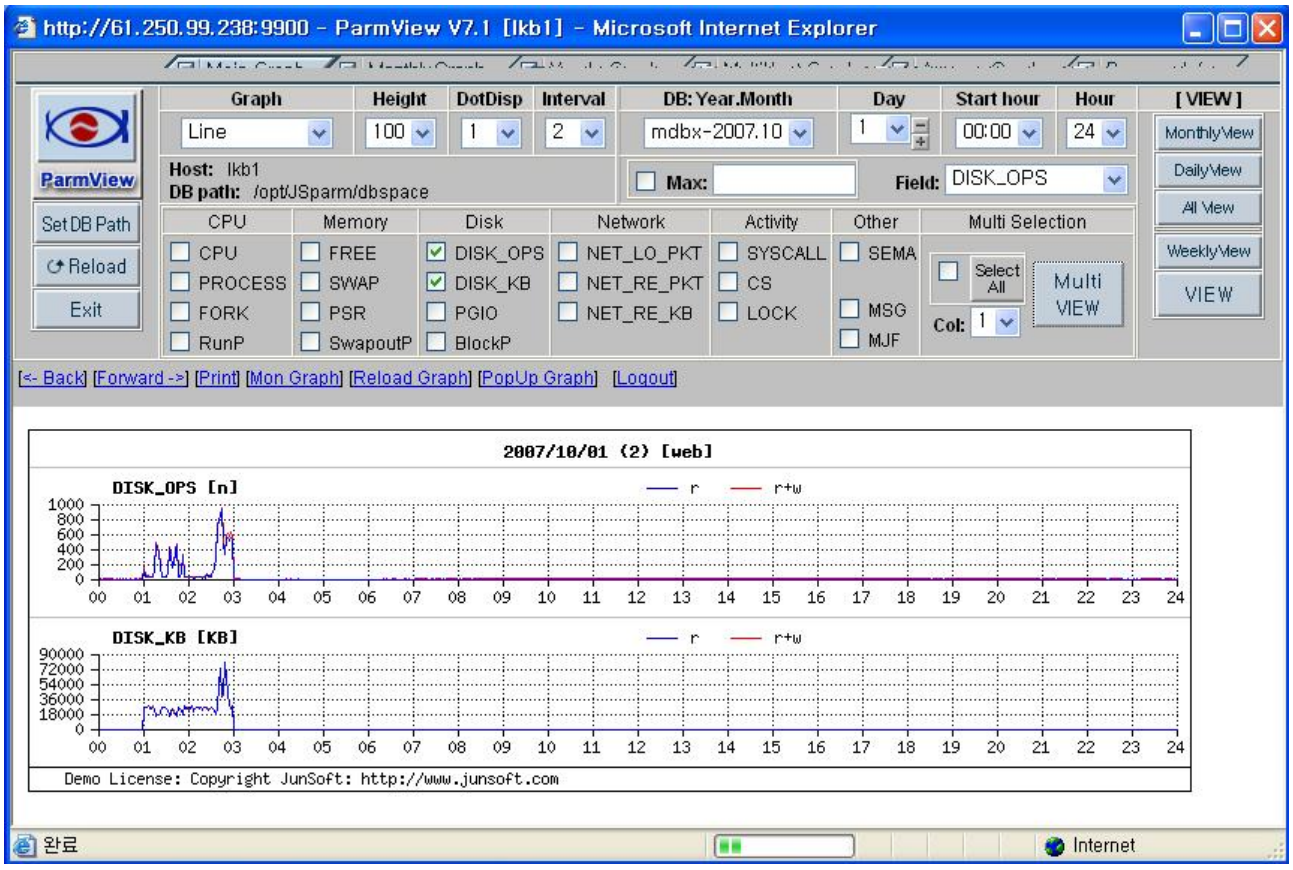
아래는 Process 개수를 체크한 결과로 최대 330 개 정도가 실행되었고, 오전 9 시부터 6 시까지 프로세스의 실행이 많아 지는 것을 볼 수 있다.



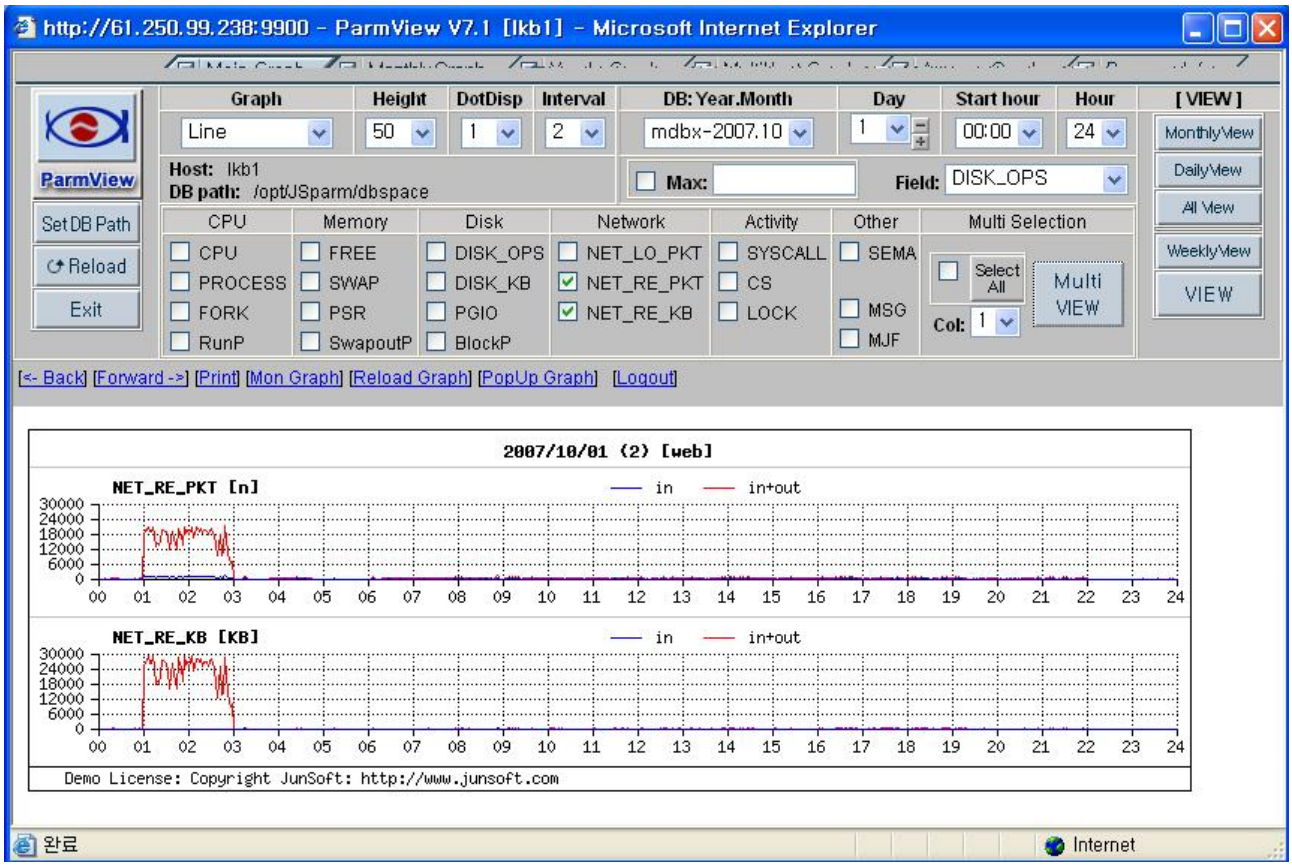
아래는 물리적 메모리와 Swap 메모리의 사용량을 나타낸 것으로 프로세스의 증감에 따라 메모리가 함께 증감하는 것을 볼 수 있다.



아래는 디스크의 Read / Write 횟수와 용량(Kbyte)을 체크한 결과로, 새벽 01~03 시에 I/O 가 활발하게 이루어지는 것을 볼 수 있다.



아래는 Network 패킷 전송 개수와 전송량(Kbyte)를 체크한 결과로 대용량 전송이 새벽 01~03 시에 집중되는 것을 볼 수 있다.



1.3. 부 록

1.3.1. Performance Monitoring 성능 측정 기준

시스템의 원활한 운영을 위해 앞서 설명한 명령어 등을 이용하여 주기적인 점검을 실시해야 하며, 그에 따른 적정한 개선 사항이 꼭 필요하다. 다음은 시스템의 성능을 측정한 데이터를 기초로 하여 시스템 자원의 적정 기준에 대한 설명이다.

◆ Rule : 우선 측정 결과를 나타내기 위해서 명령어의 '이름'과, '.'과 변수명(포맷명)을 조합하여 표시하였다. 즉 예를 들어 vmstat 의 결과 중 사용 가능한 스왑 영역의 크기를 나타내는 것이라면 vmstat.swap 이라고 표기하였으며, 변수들간에는 논리 연산자인 &&, ||, == 등을 사용하였다.

◆ Level : 각 테이블의 레벨에는 상태에 따라 그 심각한 정도를 다음과 같이 표기하였다.

레 벨	설 명
White	사용량이 아주 작다
Blue	사용량이 적다
Green	특이한 문제점 없이 원활히 운영되고 있다.
Amber	Warning Level
Red	심각한 레벨이며 조치가 필요하다
Black	시스템의 상태가 아주 심각하다

◆ Action : 각 테이블의 rule 에서 취해야 할 조치 사항에 관한 사항들이다.

● **Memory Rule**

◆ 가상 메모리

구 분	레 벨	액 션
100000k vmstat.swap	White	Swap Waste
10000k vmstat.swap 100000k	Green	No Problem
4000k vmstat.swap 10000k	Amber	Swap Low
1000k vmstat.swap 4000k	Red	Swap Low
vmstat.swap 1000k	black	No Swap

◆ Physical 메모리

구 분	레 벨	액 션
vmstat.sr == 0	White	RAM Waste
0 < vmstat.sr < 200	Green	No Problem
200 vmstat.sr 300	Amber	Raw Low
300 vmstat.sr	Red	Raw Low

- **CPU Rule**

구 분	레 벨	액 션
vmstat.r == 0	White	CPU Idle
0 < (vmstat.r / ncpus) < 3.0	Green	No Problem
3.0 (vmstat.r / ncpus) 5.0	Amber	CPU busy
5.0 < (vmstat.r / ncpus)	Red	CPU busy
mpstat.smtx < 200	Green	No Problem
200 mpstat.smtx < 400	Amber	Mutex Stall
400 mpstat.smtx	Red	Mutex Stall

- **DISK I/O Rule**

구 분	레 벨	액 션
(iostat –x .b < 5%) && (Other disks white or green)	White	No Problem
(iostat –x .b < 5%) && (Other disks amber or red)	Blue	Idle Disk
(5% iostat –x. b) && (iostat –x.svc_t < 30ms)	Green	No Problem
(20% iostat –x. b) && (30ms iostat –x. svc_t < 50ms)	Amber	Busy Disk
(20% iostat –x. b) && (50ms iostat –x. svc_t)	Red	Busy Disk
(20% iostat –x. b) && (50ms iostat –x. svc_t) && (iostat –x. disk == ”fd0” || iostat –x. disk == ”sd6”)	Amber	Floppy / CD
0% == iostat –x. w	Green	No Problem
0% < iostat –x . w < 5%	Amber	SCSI Busy
5% iostat –x. w	Red	SCSI Busy

- **Network Rule(Ethernet collision)**

구 분	레 벨	액 션
(0 < netstat –i. output.packet < 10) && (100 * netstat –i. output.colls / netstat –i. output.packets < 0.5%) && (Other nets white or green)	White	No Problem
(0 < netstat –i. output.packet < 10) && (100 * netstat –i. output.colls / netstat –i. output.packets < 0.5%) && (Other nets amver or red)	Blue	Inactive Net
(10 netstat –i. output.packet) && (0.5% 100 * netstat –i. output.colls / netstat –i. output.packets < 2.0%)	Green	No Problem
(10 netstat –i. output.packet) && (2.0% 100 * netstat –i. output.colls / netstat –i. output.packets < 5.0%)	Amber	Busy Net
(10 netstat –i. output.packet) && (5.0% 100 * netstat –i. output.colls / netstat –i. output.packets)	Red	Busy Net

1.3.2. Solaris News

: ZFS(Zettabyte File System) - 지구상의 마지막 화일 시스템

썬 마이크로시스템즈가 Solaris 를 오픈하기 이전부터 해결하고자 하려 했던 프로젝트가 하나 있었는데, 그건 바로 Solaris 가 기본적으로 사용하는 화일 시스템인 UFS 를 대체할 만한 화일 시스템을 만드는 것이었다. 이 프로젝트는 오랜 검토와 코딩 그리고 테스트를 거쳐서 나오게 되었는데, 마침 나오자마자 오픈되어 있던 Solaris 와 함께 오픈 소스의 세계에서 모습을 나타내게 되었다.

이러한 ZFS 는 시스템 관리자의 디스크/스토리지 관리에 들어가는 수고를 대폭 줄여주면서도 서비스 장애 및 데이타 발생율을 현격하게 낮춰줌으로써 일주일 24 시간 내내 수행되어야 하는 인터넷 서비스나 중요 서비스 업무에 진정 최적이 아니라 할 수 없다.

또한, ZFS 는 Solaris 의 실시간 장애 관리 오브젝트로 등록이 되어 있어서, 관리자가 언제든지 화일시스템의 장애를 실시간으로 관리할 수 있게 되어 있다. 결과적으로 이 ZFS 는 사용자 관점에 매우 편리한 방식을 제공하게 되는 데, 그런 점이 매력적이었는지 이 ZFS 도 애플에 간택이 되어 미래 Mac OSX 버젼에 나타나게 될 전망이다.

썬마이크로시스템즈는 운영체제를 만드는 입장에서 관리자의 작업의 종류와 어려운 수준, 소요 시간등을 고찰해왔는데, 그 결과는 매우 흥미로운 것이었다. 시스템 관리자는 스토리지 관리와 화일 시스템 구축 및 변경, 복구에 매우 많은 시간을 보내고 있었다는 것이다. 그 이유는 zfs 이전의 모든 화일 시스템들은 스토리지/디스크에 문제나 변경이 생기면, 가장 커다란 문제가 발생하기 때문이었다. 서비스는 모두 내려져야 하고, 기존 데이타는 모두 백업/복구되어져야 하며 새로운 디스크/스토리지와 새로운 구성을 하고 나서야 새로운 화일시스템을 구축할 수 있게 되었기 때문이다.

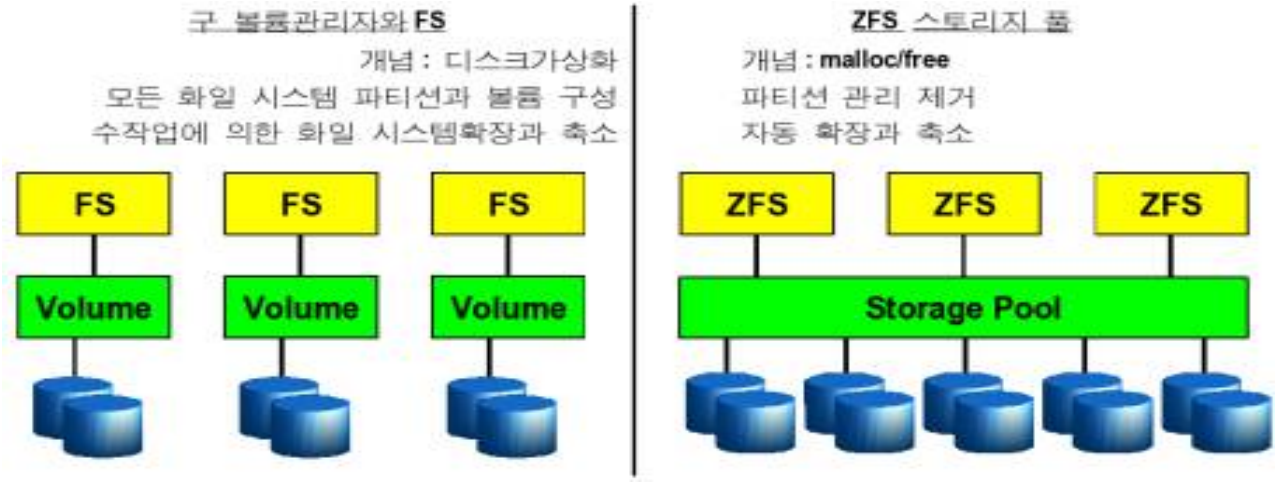
Solaris 화일 시스템 개발자는 관점의 전환을 이루어 디스크/스토리지가 중심이 아닌 '서비스'를 중심인 화일 시스템에 촛점을 두었다. 시스템에 디스크 관련 어떠한 일이 일어나도 서비스는 절대 죽지 않아야 하며, 서비스 가동중에 디스크/스토리지는 얼마든지 추가 및 제거가 가능해야 하며, 얼마든지 이전의 구성으로 되돌아갈 수 있는(undo) 그런 환상적인 화일 시스템을 고려해 보게 되었다. 이런 기능에 가장 근접한 기술이 이미 있었는데 그것은 바로 Solaris 가 메모리를 관리하는 기술이었다.

시스템 사용자들은 메모리를 장착하면서, 메모리를 어떻게 나누어야 할 지, 메모리를 어떻게 구성해야 할 지에 대해서는 전혀 신경쓰지 않는다. 운영체제는 전체의 메모리를 하나의 커다란 풀(pool)로 운영하면서 필요한 요청에 대응한다. 이점에 착안하여 새로이 만들어진 화일 시스템이 바로 zfs 화일 시스템이다.

ZFS 화일 시스템은 모든 스토리지를 하나의 거대한 풀(Pool)로 구성 운영한다. 구성된 풀에서 필요한 만큼의 화일 시스템을 만들고 변경하며 운영할 수 있도록 하고 있다. 따라서, 새로운 디스크를 추가할 때 어떻게 화일 시스템을 나눌지, 파티션을 어떻게 해야할 지에 대해서 고민할 필요가 없어지게 되었다. 풀이 허용하는 한 화일 시스템의 크기는 커졌다 작아졌다를 마음대로 할 수 있으며, 필요에 따라서 특정 상태를 저장(snapshot)을 무한정 해두었다 원하는 상태로 원복(undo)을 할 수 있게 됨에 따라, 이론적으로 풀의 용량이 허용하는 한 백업도 필요가 없게 되었다.

그 뿐만 아니라 ZFS 는 데이타는 물론 메타데이타에 대한 체섬을 유지함으로써 데이타의 장애시 복구

율을 혁신적으로 증가시켜놓았다. 즉, 데이타의 장애로 인한 데이타의 분실 자체를 고민할 필요가 없어졌다는 얘기다. 더불어, ZFS 는 미러로 구성된 환경에서 한 디스크에 장애가 발생하면 문제없는 디스크로부터 정상적인 데이타를 이용하여 장애가 난 디스크를 치료하는 기능까지 제공함에 따라 백업의 필요성도 덜해졌다. 저가의 스토리지에서는 그야말로 환상적인 궁합이다.



◆ ZFS 기능

- mount point 를 생성할 필요 없음
- file system 을 check 할 필요 없음
- /etc/vfstab 를 추가할 필요 없음
- volume 를 알 필요 없음
- nfs share 지원 가능
- 자동 mount 됨

◆ ZFS volume 지원

- 기존 volume manager 처럼 volume (raid 0, raid 1, raid 0+1, raid 5) 를 지원한다.
- ufs, vxfs, raw device 지원한다.
- volume resize 가능

◆ zpool 구성(raid 0, raid 1, raid 0*1, raid 5)

```
- Raid 0
# zpool create zfs-stripe c2t3d0s0

- Raid 1
# zpool create -m /zfs-mirror zfs-mirror mirror c2t3d0s1 c2t3d0s2
```

```
- Raid z (raid 5 에 비해 parity 분배나 제거가 향상됨)
# zpool create zfs-raid raidz c2t3d0s3 c2t3d0s4 c2t3d0s5

- Raid 0+1
# zpool create zfs-stripe-mirror mirror c2t3d0s0 c2t3d0s1 mirror c2t3d0s2 c2t3d0s6

# df -k
파일시스템              K 바이트     사용     가용    용량     설치지점
/dev/dsk/c0t1d0s0     10080200 4476397 5503001    45%    /
/dev/dsk/c0t0d0s5     5161437 3019366 2090457    60%    /data
zone                  17289216  165988 17123096     1%    /zone
zfs-stripe-mirror     10257408      24 10257330     1%    /zfs-stripe-mirror

# zpool list
NAME                   SIZE    USED   AVAIL    CAP  HEALTH     ALTROOT
zfs-stripe-mirror      9.94G    81K   9.94G     0%  온라인     -
zone                   16.8G    162M  16.6G     0%  온라인     -

# zpool status
  풀: zfs-stripe-mirror
 상태: ONLINE
 스크럽: 요청된 항목이 없습니다.
구성:

        NAME          STATE     READ WRITE CKSUM
        zfs-stripe-mirror  온라인       0     0     0
          mirror      온라인       0     0     0
            c2t3d0s0  온라인       0     0     0
            c2t3d0s1  온라인       0     0     0
          mirror      온라인       0     0     0
            c2t3d0s2  온라인       0     0     0
            c2t3d0s6  온라인       0     0     0
```

◆ ZFS 구성(Zeta File System 구성)

```
# zfs create zfs-stripe-mirror/oradata
# zfs create zfs-stripe-mirror/arch
# zfs create -V 50m zfs-stripe-mirror/backup
(raw device:/dev/zvol/rdsk/zfs-stripe-mirror/backup)
```

```
# df -k
zfs-stripe-mirror     10257408       27 10206063      1%    /zfs-stripe-mirror
zfs-stripe-mirror/oradata
      10257408       24 10206063      1%    /zfs-stripe-mirror/oradata
zfs-stripe-mirror/arch
      10257408       24 10206063      1%    /zfs-stripe-mirror/arch

# zfs list
NAME                    USED  AVAIL  REFER  MOUNTPOINT
zfs-stripe-mirror      50.1M  9.73G  27.5K  /zfs-stripe-mirror
zfs-stripe-mirror/arch  24.5K  9.73G  24.5K  /zfs-stripe-mirror/arch
zfs-stripe-mirror/backup  22.5K  9.78G  22.5K  -
zfs-stripe-mirror/oradata  24.5K  9.73G  24.5K  /zfs-stripe-mirror/oradata
```

◆ ZFS mount, umount

```
# zfs umount zfs-stripe-mirror/arch
# zfs mount zfs-stripe-mirror/arch
# zfs mount -a
```

◆ ZFS mount point 변경

```
# zfs set mountpoint=/test1 zfs-stripe-mirror/arch
# df -k
```

◆ ZFS 속성 변경

```
# zfs get all
# zfs set property=value
```

◆ zpool status check

```
# zpool status -x
모든 풀의 상태가 정상입니다.
```

◆ zfs, zpool 제거

```
# zfs destroy zfs-stripe-mirror/backup
# zpool destroy zfs-stripe-mirror
```

◆ zfs iostat check

```
# zpool add zfs-stripe-mirror c2t3d0s5

# zpool iostat 1
capacity     operations    bandwidth
pool         used  avail   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
oracle      14.4G  2.39G      0     21  44.1K   267K
oracle      14.4G  2.39G      0      0      0      0
oracle      14.4G  2.39G      0      0      0      0
#
# zpool iostat -v oracle 1
               capacity     operations    bandwidth
pool         used  avail   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
oracle      14.4G  2.39G      0     21  44.0K   267K
  c1t16d0   7.18G  1.19G      0     10  22.0K   133K
  c1t17d0   7.18G  1.20G      0     10  22.0K   135K
----------  -----  -----  -----  -----  -----  -----

# zpool replace zfs-stripe-mirror c2t3d0s5 c2t3d0s6
```

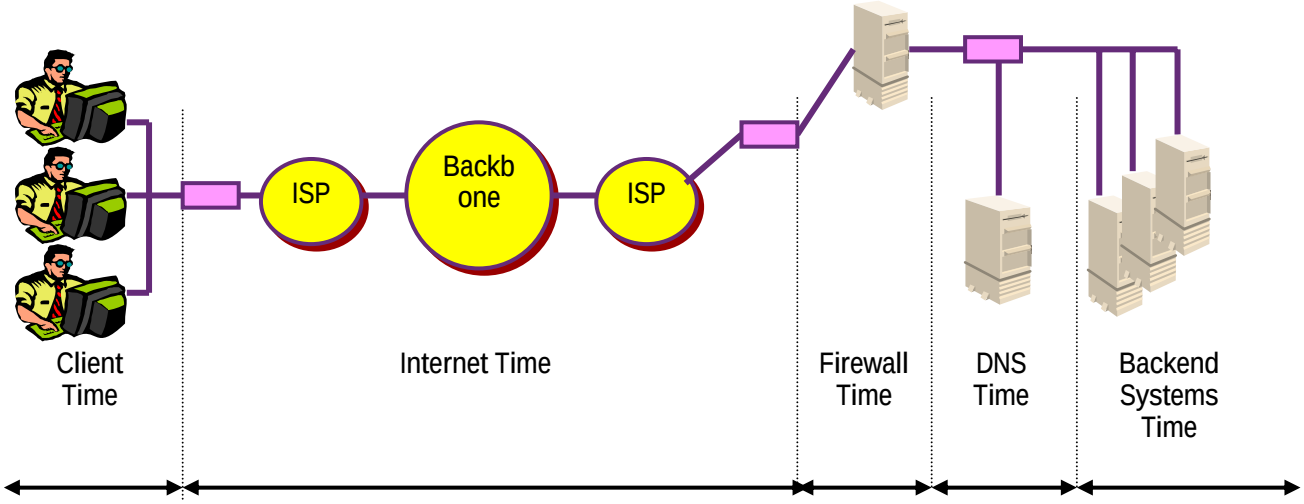
◆ 기타 명령

```
# zpool add zfs-stripe-mirror c2t3d0s5                (disk 추가)
# zpool remove zfs-stripe-mirror c2t3d0s5             (Disk 제거)
# zpool replace zfs-stripe-mirror c2t3d0s5 c2t3d0s6  (Disk 교체)
```

2. Pro-Active Tuning Service

2.1. 실제 사용자(End-User) 관점의 응답시간 튜닝

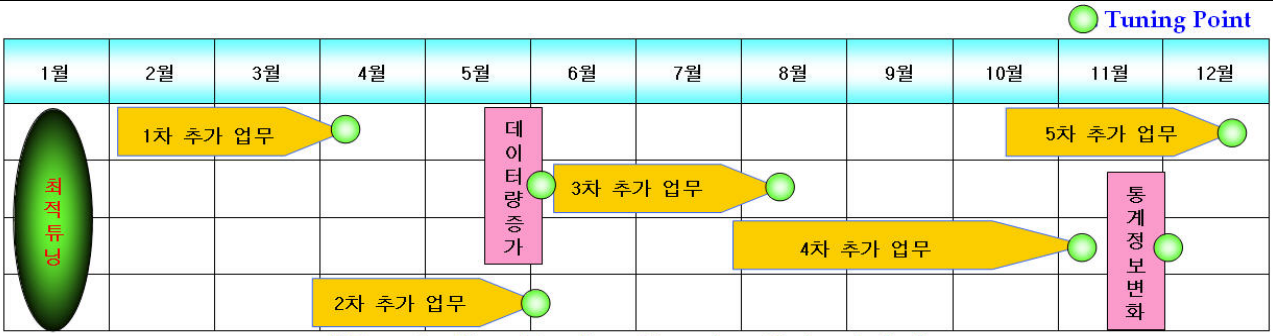
Pro-active tuning service 는 사용자 관점의 모니터링 및 분석을 통하여 실제 End-User 가 느끼는 응답시간(Response Time)을 튜닝합니다. APM(Application Performance Management) 툴을 이용하여 End-User 의 Request 결과를 반환받기까지의 모든 구간(Client PC, Internet 구간, FireWall, DNS, Web Server, WAS, DBMS) 을 분석하여 가장 Delay Time 이 많이 소요된 구간을 찾아 냅니다.



2.2. 최상의 성능 상태로 비즈니스 고가용성을 유지

Pro-Active Tuning Service

- 매 업무 단위 프로젝트 마다 참여하여 업무 적용(Open) 前 문제 요소를 분석하여 튜닝.
- 단위 업무 적용(Open) 후 매 3 개월(데이터량 갱신 주기) 마다 튜닝 포인트를 설정, 성능 둔화 요소를 해결.
- 전사적으로 새롭게 추가되는 업무 단위 프로젝트의 모든 SQL 쿼리를 검토 및 튜닝.
- 다양한 대용량 데이터베이스 관리/튜닝 기법을 도입하여 최적의 DB 상태를 1 년 내내 상시 유지.
- 전략적 튜닝 Factor 를 분석, 투자 대비 효율이 높은 Targeting 기법 적용. (비중도 높은 SQL 을 튜닝함)



[Pro-Active Tuning Service Schedule]

Performance Drop Point 마다 적절한 튜닝을 실시함으로써 항상 최적의 성능을 유지한다. !!

2.3. Knowledge Transfer

Pro-Active Tuning Service 는 고객의 Business Process 를 이해하고 시스템을 분석한 후 튜닝하는 것으로 완료되지 않습니다. 실제로 고객사 환경에서 튜닝한 내용을 그대로 실무자들에게 전수하여 내부 임직원의 역량을 제고시킵니다. 또한, Oracle RDBMS 신 버젼의 New Features 를 교육함으로써, 이용자(관리자 및 개발자)가 스스로 개발 업무의 효율 및 생산성을 향상시킬 수 있도록 지원합니다. 이외에도 DBMS 관리자를 위한 관리 노하우(고급 Trouble-Shooting, 대용량 DB 처리, 병렬 처리 등)를 전수함으로써, 최상의 시스템을 최고의 기술로 유지할 수 있도록 지원합니다.

UAS (User Adapted Seminar) 진행 사례 및 내용 (Contents)

- **개발자를 위한 SQL 튜닝 실무 사례 세미나**

  G 쇼핑몰 업체 튜닝 후 실제 고객사의 튜닝 사례를 개발자들에게 전수하여 개발자들이 성능을 고려한 SQL 을 작성할 수 있도록 내부 역량을 제고시킴.

- **Oracle 10g New Features 세미나**

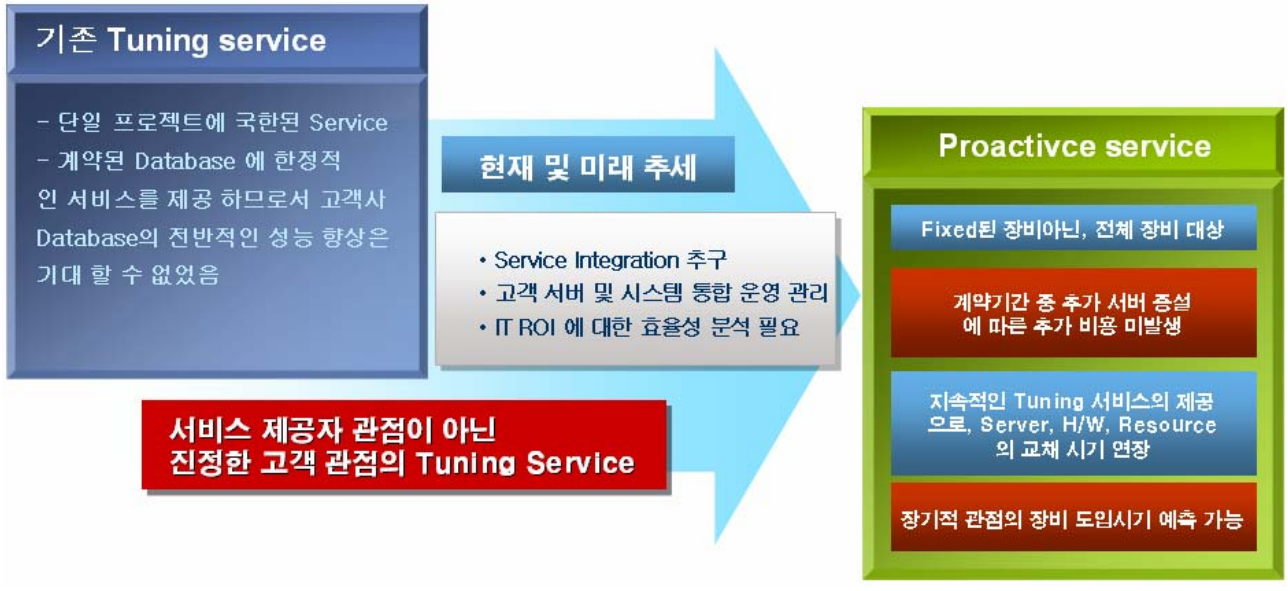
  S Global 전자 기업 : Oracle 10g 버전으로 업그레이드 하기 전, 신 버전의 새로운 기능과 주의 사항을 전파함으로써, 업그레이드 후 발생할 수 있는 문제점의 사전 제거와 개발자들이 새로운 기능을 이용함으로써, 개발 생산성을 향상시킴.

- **K 국가기관 DBMS 관리 노하우 세미나**

  내부 관리자 (DBA,SE)들을 대상으로 DBMS 관리자들이 흔히 겪을 수 있는 상황에 대한 Administration Know-How 와 고급 Trouble-Shooting 사례를 소개함으로써, 관리 기술력을 향상시킴.

2.4. Tuning 범위 확대

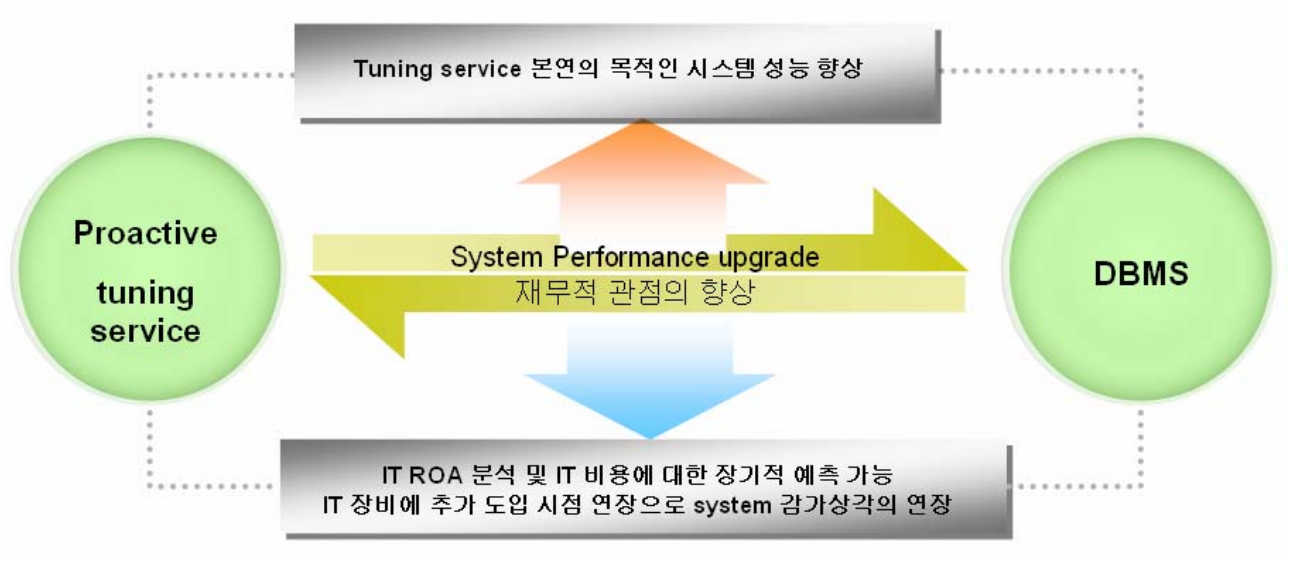
Pro-active tuning service 의 제공 범위는 제한된 Database, 제한된 Server 에 국한 되지 않으며, 고객사 전체의 Server+DB 를 대상으로 그 범위를 확대 함으로써 고객사 전체 Performance 향상에 기여 합니다.



2.5. 기대효과

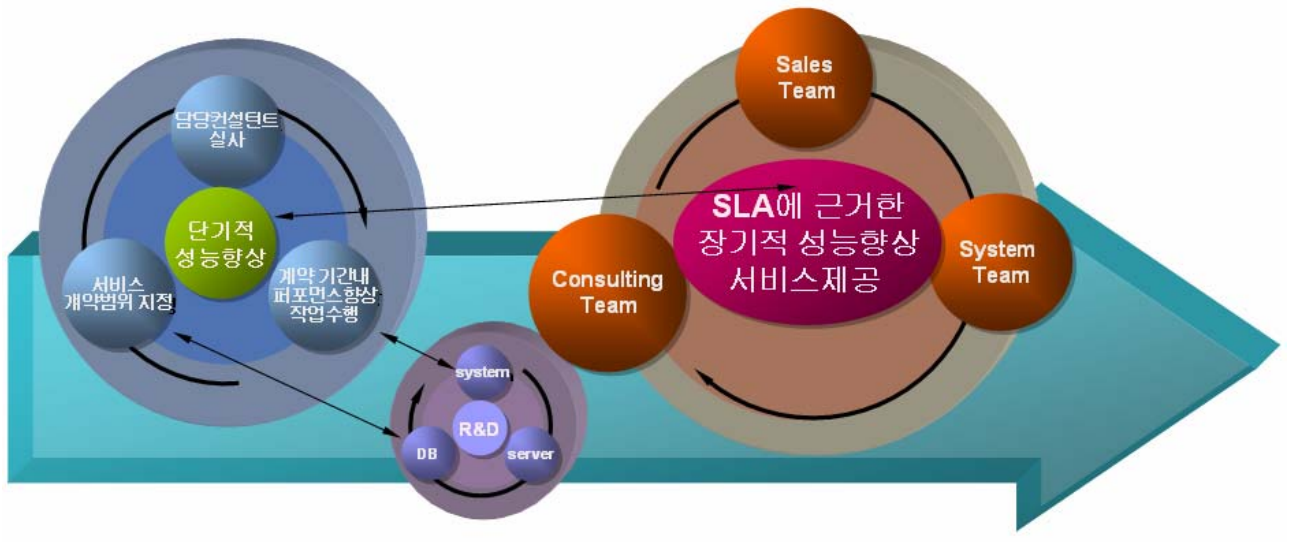
2.5.1. 재무적 관점

기존 Tuning Service 는 주로 System Performance 향상에 따른 업무 트레픽 감소에 초점이 맞춰져 있었습니다. Pro-actvice 서비스는 Tuning 작업을 통한 업무 처리 시간 단축 뿐만 아니라, 업무 처리 시간 단축으로 가져 올 수 있는 재무적 성과를 가능하게 합니다.



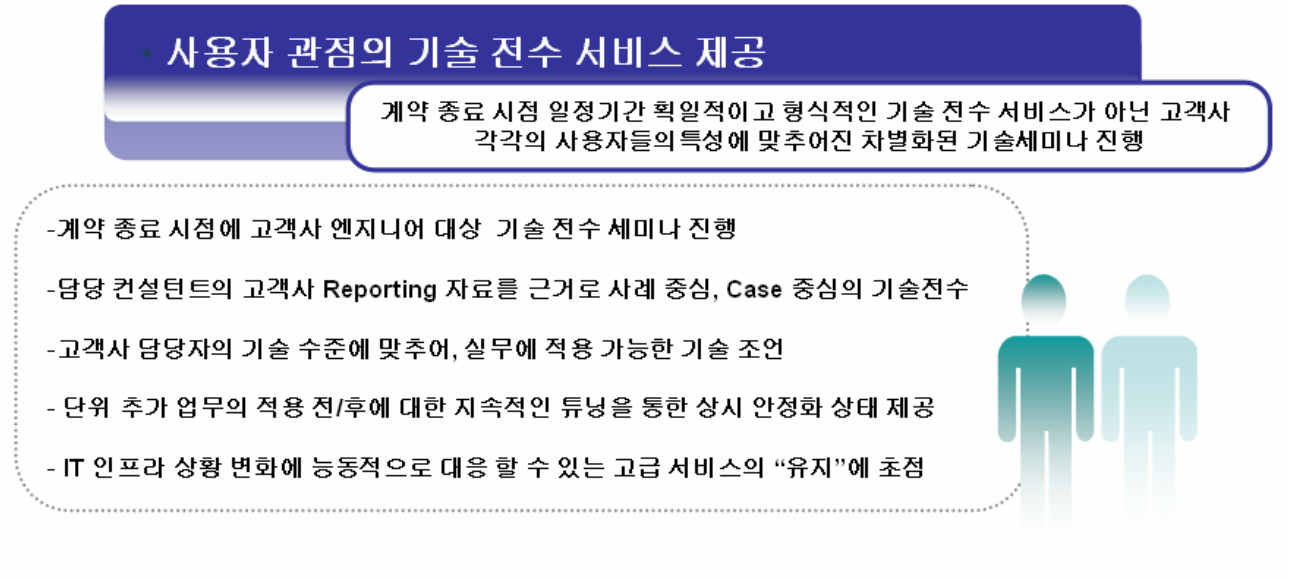
2.5.2. 서비스 관점

단기적 성능 향상에 맞추어진 기존 Tuning 서비스는 계약된 system 및 Database 를 서비스 대상으로 하기 때문에 전사적인 차원의 성능 향상을 기대 하기 어려웠습니다. Proactive tuning service 는 계약 기간 동안 주요 비즈니스 Factor 별로 SLA 를 정하여 Tuning consulting 을 수행함으로써 서비스 자체의 안정성을 제고 할 수 있습니다.

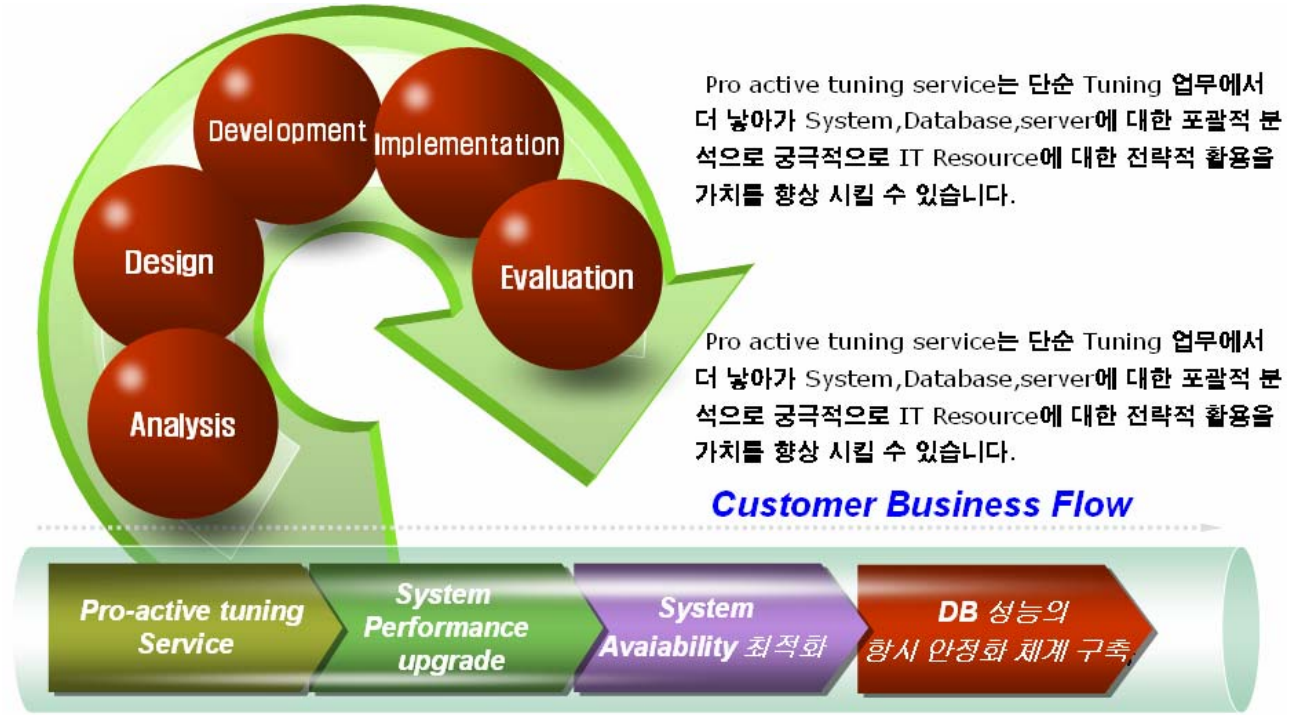


2.5.3. 사용자 관점

Proactive tuning service 는 계약 종료 시점 작업한 Tuning 산출물을 통한 기술전수 세미나를 진행 함으로서 고객사의 사용자가 실무에서 바로 적용 가능한 기술을 전수함과 동시에, 계약기간 종료 후에도 Proactive tuning service 를 유지 할 수 있는 방향을 제시 합니다.



2.5.4. 혁신적 관점



2.6. 제안 Package별 가격

2.6.1. DBMS Tuning

(단위 : 천원)

상 품	집중 Tuning	정기 Tuning	Total 시간	시간당	List Price	할인가	할인률
Package1	1 주(5 일) 8H*5D = 40H	1 일/분기 8H*3Q = 24H	40H + 24H = 64H (8D)	200	12,800	12,000	6.3 %
Package2	2 주(10 일) 8H*10D = 80H	1 일/월 8H*1D*11M = 88H	80H + 88H = 168H (21D)	200	33,600	30,000	10.7 %
Package3	3 주(15 일) 8H*15D = 120H	2 일/월 8H*2D*11M = 176H	120H + 176H = 296H (37D)	200	59,200	50,000	15.5 %
Package4	4 주(1 개월) 8H*20D = 160H	3 일/월 8H*3D*11M = 264H	160H + 264H = 424H (53D)	200	84,800	68,000	19.8 %

※ 집중 Tuning 이나 정기 Tuning 의 시간을 사용자교육(UAS)로 전환 하실 수 있습니다.

※ Tuning 대상 Instance 를 고정하지 않습니다.

2.6.2. DBMS Tuning+ APM + NA

(단위 : 천원)

상 품	집중 Tuning	정기 Tuning	Total 시간	시간당	List Price	할인가	할인률
Package5	1 주(5 일) 8H*5D = 40H	1 일/분기 8H*3Q = 24H	40H + 24H = 64H (8D)	250	16,000	15,000	6.3 %

※ 추가 1 일당 2,000,000 원이 책정 됩니다.

2.6.3. Server (H/W, OS) Tuning + Storage 재구성 컨설팅 + DBMS Tuning (단위 : 천원)

상 품	집중 Tuning	정기 Tuning	Total 시간	시간당	List Price	할인가	할인률
Package6 Athena	1 주(5 일) 8H*5D = 40H	1 일/분기 8H*3Q = 24H	40H + 24H = 64H (8D)	250	16,000	15,000	6.3 %

※ 추가 1 일당 2,000,000 원이 책정 됩니다.

2.7. Package 설명

내 용	설 명
집중 Tuning	Target DB 의 산정된 Tuning 기간에 따라, 성능이 저하된 DB 를 최적의 성능을 발휘 할 수 있도록 집중적인 Tuning 을 실시 합니다. Database Server Tuning, Database Instance Tuning, Database Object Tuning, Database SQL Tuning
정기 Tuning	집중 Tuning 을 통해서 향상된 최적의 성능을 지속적으로 유지하기 위하여, 정기적으로 Tuning 을 실시하여, 업무 추가/변경, Data 증가 등으로 인한 성능저하의 요인을 찾아 최적의 성능을 유지 할 수 있도록 정기적으로 Tuning 을 실시 합니다.
User Adapted Seminar	장기적 관점의 관리 운영 Know-How 전파 합니다. 내부 관리자 (DBA,SE) 대상 DBMS Administration Know-How 와 사례 소개함으로써, 관리 기술력을 향상할 수 있습니다. 개발자 대상 최적의 SQL 을 작성할 수 있는 Skill 을 전파합니다.
Package5	DBMS Tuning 뿐만이 아니라, DB 및 Network 에 대한 Advanced Trouble Shooting 을 실시 합니다. (예를 들어, 고객사의 본사에서는 속도가 빠른데, 지사에서는 느릴 경우, 해결되지 않는 Database 의 복구)
Package6	사용중인 고가의 System 의 최적화 여부를 진단해서 튜닝을 실시 합니다. Server(Hardware, O/S), Storage 구성상태를 진단하여, 최상의 성능을 발휘할 수 있도록 Tuning 을 실시합니다,

2.8. Promotion

Promotion 기간	2007 년 9 월 1 일 ~ 2007 년 11 월 31 일	3 개월간

(단위 : 천원)

상 품	Total 시간	List Price		특전 1		특전 2		특전 3	할인률
		시간당	금액	무상 정밀진단		시간당	할인가	추가할인	
Package1	40H + 24H = 64H (8D)	200	12,800	1 일	무상	150	9,600	9,000	29.7%
Package2	80H + 88H = 168H (21D)	200	33,600	1 일	무상	150	25,200	25,000	25.6%
Package3	120H + 176H = 296H (37D)	200	59,200	1 일	무상	150	44,400	44,000	25.7%
Package4	160H + 264H = 424H (53D)	200	84,800	1 일	무상	150	63,600	63,000	25.7%

Promotion 기간 중 계약하시는 모든 고객에게 아래의 특전이 주어지며, 계약여부와 관계 없이 무상진단을 제공 합니다.

※ Promotion 특전 1 : Database 무상 Analysis (Max 3 Instances)

※ Promotion 특전 2 : Package 구매시, 시간당 15 만원으로 할인 적용

※ Promotion 특전 3 : Package 별 추가 할인 적용

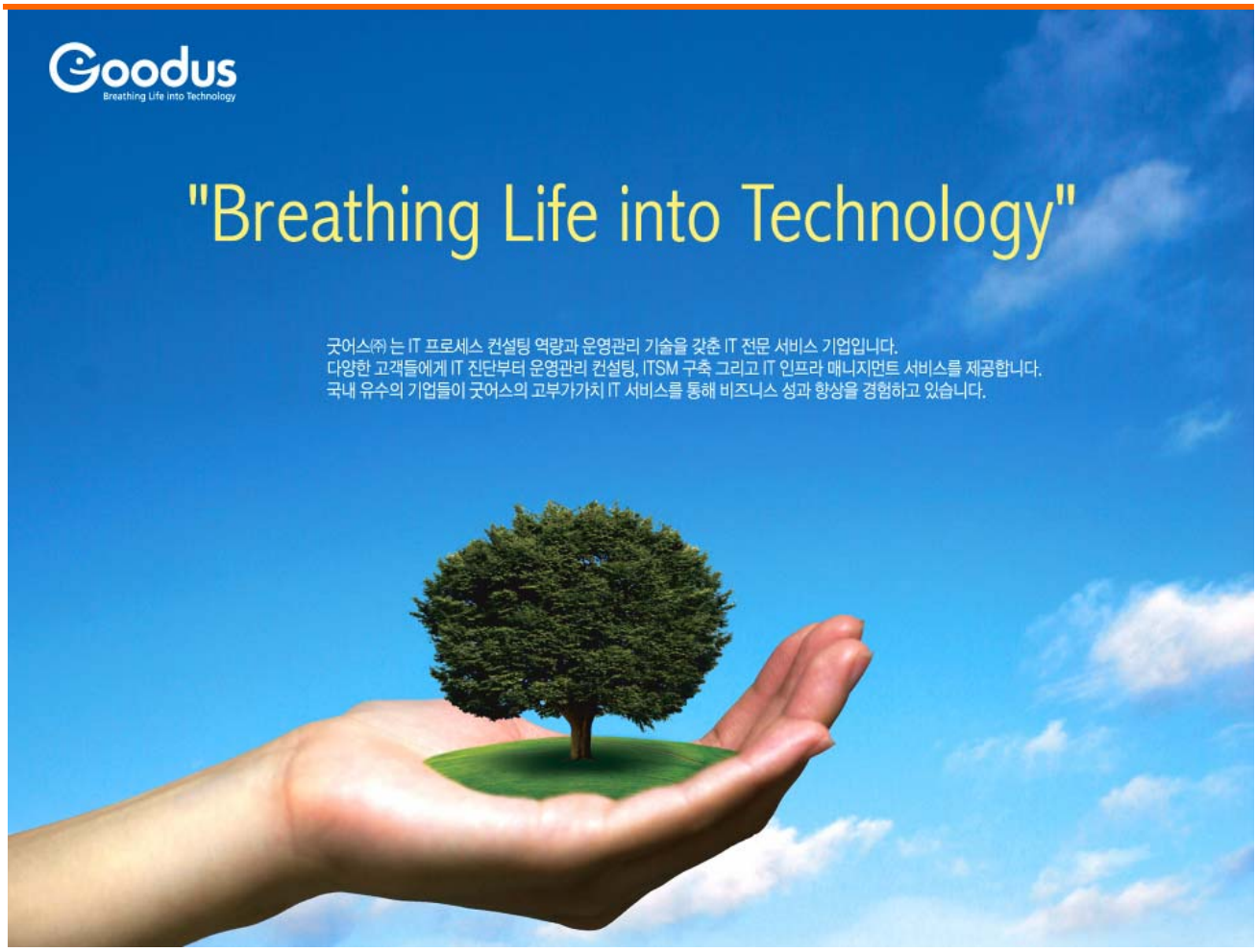
※ 집중 Tuning 이나 정기 Tuning 의 시간을 사용자교육(UAS)로 전환 하실 수 있습니다.

※ Tuning 대상 Instance 를 고정하지 않습니다.

※ 담당자 연락처

권웅원 팀장 011-514-6678 070-7017-4206

김승필 대리 011-9096-4919 070-7017-4129


Goodus
Breathing Life into Technology
"Breathing Life into Technology"
굿어스㈜ 는 IT 프로세스 컨설팅 역량과 운영관리 기술을 갖춘 IT 전문 서비스 기업입니다.
다양한 고객들에게 IT 진단부터 운영관리 컨설팅, ITSM 구축 그리고 IT 인프라 매니지먼트 서비스를 제공합니다.
국내 유수의 기업들이 굿어스의 고부가가치 IT 서비스를 통해 비즈니스 성과 향상을 경험하고 있습니다.