디지털산업정보학회 논문지 제5권 제3호-2009년 9월

K증권 홈트레이닝 시스템 분석을 통한 성능개선에 관한 연구

김 현 호* · 박 용 덕**

A study for performance improvement by system analysis of HTS running K Securities

Kim, Hyun Ho · Park, Yong Duck

〈Abstract〉

Computer system performance has always had the possibility of affecting business profitability, but with the advent of the World Wide Web where customers interact directly with Web servers, response time can have a direct and dramatic impact on business revenue.

This paper is written in the operation environment and system analysis of HTS(Home Trading System) running K Securities. This paper also shows the method for performance improvement through investigation and analysis for the overall systems resources whether HTS has an appropriate performance or not. Performance analysis includes specially CPU analysis, Memory analysis, Disk Input/Output analysis and application analysis. Besides providing more detailed server specification for expansion from now on, system performance can be maintained with effect in the future. Through this study it is possible to manage the performance of HTS more easily and to solve problems such as a bottleneck more quickly.

Key Words : HTS, System Tuning, Performance Evaluation

Ⅰ. 서론

UNIX 시스템은 1990년대 후반에 들어서면서 거의 컴퓨터 업계의 표준 운영체제가 된 것처럼 많이 개발 및 보급되어 사용하고 있다. 1980년대까지 UNIX 시스템이 워크스테이션 전용으로 사용될 때는 IT업무에 미치는 영향이 상대적으로 적어 시스템 관리 자체가 중요하지 않았지만 요즘의 UNIX 시스템은 거의 모든 업무에 사용됨에 따라 회사 전체의 생산성에 영향을 미치게 되었다. 이런 환경변화에 따라서 시스템 관리자는 도구(Tool)를 이용하여 시스템을 보다 효율적으로 관리하며 생산성을 향상시킬 필요성이 생겼다[1].

IT 구성요소에는 성능관리, 구성관리, 보안관리, 계정관리 및 운영관리 등이 있다. 이들 중에서 시스템의 성능관리 부분이 가장 중요하고 어렵다. 성능관리는 일반적으로 애플리케이션의 개발측면과 시스템의 관리측면으

* 명지전문대학 컴퓨터정보과 부교수
** 한국 썬마이크로시스템즈 시스템전략사업본부 이사

로 나눌 수 있다. 애플리케이션의 개발에는 애플리케이션의 정의, 분석, 설계, 코딩, 테스트, 디버깅, 유지보수 등이 있는데 이런 각 과정의 성능관리가 필요하며 이들 과정의 성능을 어떻게 관리하느냐에 따라 전체적인 애플리케이션의 성능 차이가 발생한다. 시스템의 관리는 시스템의 가용성, 사용율의 감시, 일정 수준의 성능유지, 향후 부하증가에 대비한 용량계획 등이 포함된다. 특히 시스템 용량 관리는 현재 사용하고 있는 컴퓨터 시스템의 용량이 얼마나 어떻게 사용되고 있는지, 업무의 증가로 인한 용량의 사용을 어떻게 할지가 관심사이다.

일반적으로 시스템 분석을 통한 성능개선에 대한 연구는 시스템에 무한 루핑되는 프로그램을 실행시켜 일정한 부하를 주거나, Load Runner와 같은 부하를 주는 프로그램을 실행시켜 시스템 분석을 하지만 본 연구에서는 실제 운영하고 있는 K증권의 HTS(Home Trading System)를 분석함으로써 실무적이고 보다 현실성 있는 연구이다.

본 논문에서는 실제로 운영하고 있는 K증권의 HTS(Home Trading System)의 현재 운영환경을 점검하고 향후 시스템 확장 시 필요한 자원을 산출하고 또한 애플리케이션의 사용 프로세스를 모니터링 함으로써 응답속도 저하 등 문제가 될 수 있는 모듈부분을 찾아내어 성능 개선의 효과를 가져 올 수 있는 부분에 대해 결론 및 제안을 한다. 제2장에서는 관련 연구를 설명하며 3장에서는 HTS 시스템 운영환경 및 분석대상을 살펴본다. 4장에서는 시스템 분석 및 결과를 보이고, 끝으로 결론 및 제안을 한다.

II. 관련연구

컴퓨터 시스템의 성능을 측정, 분석하는 중요한 이유는 시스템에서 사용되는 각 자원의 현황을 측정하여 이를 바탕으로 시스템이 효율적으로 운용되도록 지원하는 것이다[2]. 또한 시스템 성능을 전체적으로 향상시키며 측정된 자료를 토대로 추가적인 애플리케이션 개발 및 운영, 사용자의 증가 등으로 인해 향후 시스템 확장 시 이를 반영하기 위함이다[3].

컴퓨터 시스템의 성능평가 방법에는 CPU 평가방법, Memory 평가방법, 디스크 입출력 평가방법, 네트워크 평가방법, 커널 파라미터 평가방법, 애플리케이션 평가방법 등이 있다. 성능평가 시 필요로 하는 요구량이 가용한 자원의 크기를 초과할 때 병목 현상이 일어나는데 이런 현상을 해결하는 방법에는 두 가지가 있다. 첫째는 가용한 자원의 크기를 증가시키는 방법이고 둘째는 요구량의 크기를 줄이는 방법이 있다. 자원의 크기를 증가시키는 방법은 시스템 튜닝 또는 증설(Upgrading)을 의미하고 요구량의 크기를 감소시키는 방법은 시스템 및 부하의 균형을 효율적으로 맞추는 것을 말한다[4]. CPU 병목현상이 발생할 경우 하드웨어를 통한 해결방법은 더 빠른 프로세서로 증설하거나 프로세서 내의 캐쉬 크기를 증가시킨다. 또는 단일 프로세서에서 다중 프로세서로 바꾸거나 특정 애플리케이션을 여러 시스템에 분산하여 실행시킨다. 소프트웨어를 통한 해결방법은 많은 시간이 필요한 일괄처리 작업은 절정(peak)시간이 아닌 때에 실행하거나, 중요하지 않은 애플리케이션은 낮은 우선순위를 설정하여 CPU 자원이 많을 때 실행시킨다. 또는 시스템 계수(accounting) 작업을 하지 않거나 실행하는 애플리케이션을 최적화시킨다[5]. Memory 병목현상이 발생할 경우 하드웨어를 통한 해결방법은 주기억장치를 추가로 증설하거나 그래픽을 많이 사용할 때는 x-터미널보다 워크스테이션을 사용해서 서버 자체의 부하를 감소시킬 수 있다. 소프트웨어를 통한 해결방법은 주기억장치 잠금(locking)의 사용을 줄이거나 불필요한 윈도우를 삭제하고, 주기억장치를 많이 필요로 하는 프로그램을 찾은 후 실행 시 절정시간(peak time)을 피한다. 또는 문제가 되는 애플리케이션을 재설계하거나 커널의 크기를 감소시키고 시스템 테이블의 크기를 가능한 작게 만들어 사용한다. 디스크의 병목현상이 발생할 경우 하드웨어를 통한 해결방법은 디스크 드라이브를 추가 또는 디스크 채

널을 추가시키거나 처리속도가 더 빠른 디스크로 교체한다. 더 나아가 디스크 스트라이핑을 구현하거나 디스크 미러링을 구현한다. 소프트웨어를 통한 해결방법은 여러 디스크에 스왑영역을 분산 재배치하여 디스크의 입출력 집중을 막거나 여러 디스크의 입출력 균형이 잡히도록 파일 시스템을 재구성한다[6].

증권 시스템의 성능측정 방법도 일반적인 컴퓨터 시스템의 성능측정 방법을 특별히 벗어나지는 않는다. 본 연구에서도 위에서 언급한 바와 같이 일반적인 UNIX 시스템의 성능측정 방법을 이용하여 성능상의 문제점을 발견하고, K증권사의 HTS를 운영하는데 있어 최적의 컴퓨터 자원용량을 산출하여 사용자 증가로 인한 서버의 증설 계획을 세운다. 아울러 시스템 장애 시에도 HTS가 정상적으로 동작할 수 있는 시스템 아키텍처 및 서버 용량에 대해 미리 대비한다. 그리고 본 연구에서는 단위 시스템만의 성능을 측정 및 분석뿐만 아니라 HTS의 한 개의 그룹 관점에서 성능을 측정 및 분석함으로써 HTS를 운영하기 위한 최적의 서버 환경을 도출할 수 있다.

기존 연구에서의 성능평가는 컴퓨터 시스템에 병목현상이 발생하면 주로 CPU, Memory, Disk 입출력을 위주로 분석을 했는데 본 연구에서는 CPU, Memory, Disk 입출력을 분석하였고, 추가적으로 애플리케이션을 분석하였다.

Ⅲ. 시스템 운영환경 및 분석대상

3.1 HTS 시스템 운영환경

K증권사에서는 HTS(Home Trading System) 서버로 SUN E5500 모델 9대가 운영 중에 있으며, 사용하는 애플리케이션은 고객들에게 증권정보 및 매매 등을 제공하는 시스템으로 CIVIC에서 개발한 애플리케이션이다. HTS 시스템 하루 사용자는 약 12,000명에서 14,000명 정도이다. 증권 시스템에서 특히 HTS는 증권사의 수익과 직접적인 연관이 있는데 장애가 발생하여 주식 주문체결이 지연되면 수익률이 떨어질 뿐만 아니라, 고객이 다른 증권사로 이동할 수 있기 때문에 안정적인 운영이 매우 중대한 시스템이다.

HTS를 분석하기 전에는 잘 운영되고 있는 시스템은 아니므로 실제로 K증권 담당자는 HTS에서 시스템 성능이 저하되어 시스템 분석을 요청하였고, 아울러 증시 활황기에 사용자 수가 급격히 늘어날 경우 HTS의 서버증설이 필요한데 어떤 용량의 서버를 서버 풀에 넣어야 하는지를 검토요청을 하였다. HTS 서버 9대중에서 1, 2호기는 CPU 개수가 12개로 같아서 이들 중에서 1호기를 설정했으며, 3호기부터 9호기는 CPU 개수가 8개로 같아서 이들 중에서 3호기를 설정했다. 그리고 9호기는 CPU 개수의 변화에 따르는 성능 분석을 위해서 설정했다. 그러므로 2, 4, 5, 6, 7, 8호기는 분석대상에서 제외되었다.

본 연구에서 홈트레이닝 시스템의 성능개선을 하기 위해 sar 명령어를 이용하여 CPU 사용현황을, top 명령어를 이용하여 Memory 사용현황을, df 명령어를 이용하여 Disk 사용현황을 분석하며 마지막으로 ps 명령어를 이용하여 CPU 점유율이 높은 프로세스를 찾아낸 후 소스 코드를 분석하여 시스템 성능을 개선하고자 한다. 이러한 분석들을 통해 HTS의 응답속도 개선으로 인한 성능 개선 및 방향제시 그리고 효율적이고 알맞은 서버의 용량을 제안하여 향후 증시 활황기에 HTS의 사용자가 증가하였을 경우에도 적적한 시스템의 증설을 통해 안정적인 응답시간을 사용자에게 제공함으로써 지속적이고 안정적인 시스템을 운영을 하고자 한다.

실험에 대한 목표는 CPU, Memory, Disk의 사용현황을 분석하여 이를 토대로 성능개선의 효과를 보일 수 있는 컴퓨팅 자원에 대한 증설 등 방안 제시뿐만 아니라 HTS의 애플리케이션이 운영되는 동안, 높은 CPU 점유율을 보이는 프로세스에 대한 소스 코드의 개선점을 찾아내어 수정함으로써 애플리케이션의 잘못된 설계로 인한 컴퓨터 자원의 낭비를 없애는 것이다.

3.1.1 HTS 서버 환경

HTS의 서버로 SUN E5500 모델 9대가 운영 중에 있으며 HTS 서버 환경은 <표 1>과 같다.

<표 1> HTS 서버 사양

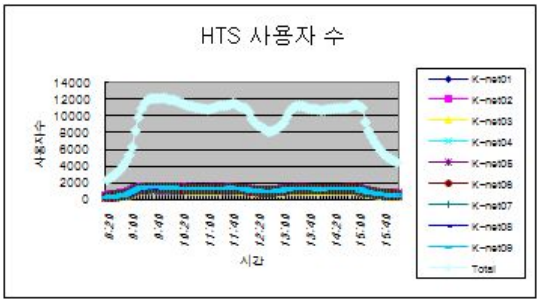
hostname	Model	CPU (400Mhz)	Memory	Disk
K-net01	Sun E5500	12	3GB	I:4GB*2, E:8GB*4
K-net02	Sun E5500	12	3GB	I:4GB*2, E:8GB*4
K-net03	Sun E5500	8	2GB	I:4GB*2, E:8GB*4
K-net04	Sun E5500	8	2GB	I:4GB*2, E:8GB*4
K-net05	Sun E5500	8	2GB	I:4GB*2, E:8GB*4
K-net06	Sun E5500	8	2GB	I:4GB*2, E:8GB*4
K-net07	Sun E5500	8	2GB	I:4GB*2, E:8GB*4
K-net08	Sun E5500	8	2GB	I:4GB*2, E:8GB*4
K-net09	Sun E5500	8	2GB	I:4GB*2, E:8GB*4

3.1.2 사용 중인 애플리케이션 현황

K-net은 K증권에서 제공하는 HTS(Home Trading System)으로 고객들에게 증권정보 및 매매 등을 제공하는 시스템으로 CIVIC에서 개발한 애플리케이션을 사용하고 있다. DBMS는 사용하지 않고 있으며 ISAM 파일 형태의 데이터를 가지고 있고, TCP/IP프로토콜의 TCP을 이용한 전형적인 클라이언트-서버 프로그램이다.

3.1.3 HTS 시스템의 사용자 현황

매일 HTS 시스템을 약 12,000~14,000명의 사용자가 이용하고 있다.



<그림 1> HTS 시스템의 사용자 수

HTS 시스템의 분석을 위해 임의로 7월 6일을 선택하였고, 절정 시간(peak time)대인 오전 9시 50분 호스트별 최대 사용자 현황은 <표 2>와 같다.

<표 2> 호스트별 최대 사용자 현황

K-net	01	02	03	04	05	06	07	08	09	전체
사용자수	1354	1351	1366	1357	1286	1372	1368	1376	1378	12208

3.2 분석 대상

CPU의 개수 4, 8, 12의 환경에서 실제 사용자 수에 따른 시스템 자원을 모니터링한다.

(1) K-net01 호기(CPU 12, Memory 3GB)에서 사용자 수의 변화에 따른 CPU 사용률 및 Memory상황을 분석한다.

(2) K-net03 호기(CPU 8, Memory 2GB)에서 사용자 수의 변화에 따른 CPU 사용률 및 Memory상황을 분석한다.

(3) K-net09 호기에서 프로세서 운영 상태를 바꾸는 psradm 명령어[7]를 이용하여 4개의 CPU를 오프라인시킴으로써 CPU 4개의 운영환경을 만들어 실제 운영환경에서 사용자 수와 CPU 사용률을 분석한다.

Ⅳ. 시스템 분석 및 결과

sar, swap, prtconf, sysdef, vmstat, ps, top 등 시스템에서 제공되는 명령어와 Public으로 제공되는 툴(Tool)을 사용하였다.

(1) 시스템 환경 관련 데이터는 다음 명령어들을 활용하였다.
- 시스템 환경 : sysdef 명령어
- 네트워크 환경 : netstat, ifconfig 명령어
- IPC 자원 : sysdef, ipcs 명령어
- 디스크 환경 : df 명령어
- Swap 환경 : swap 명령어

(2) 모니터링 데이터는 다음 명령어들을 활용하였다.
- CPU : sar, vmstat, mpstat 명령어
- Memory : sar, vmstat, top 명령어
- Process : ps, top 명령어

4.1 CPU 분석 및 결과

CPU 분석방법은 HTS 운영 중에 시스템 활동 레포터(activity reporter)인 sar명령어[8]를 이용하여 시스템의 사용자 증가에 따른 CPU 사용률을 조사함으로써 현 시스템의 CPU의 필요 용량을 찾아낸다. 다음 1호기, 3호기, 9호기만 선택해서 튜닝을 한 이유는 다음과 같다. 1호기, 3호기는 서버 사양이 서로 다르며, 9호기는 CPU 개수의 변화에 따른 성능분석을 위해서 선택했다.

(1) 1호기(CPU: 12, Memory: 3GB)

<표 3> 1호기 CPU 사용률 데이터

시간	사용자 수	CPU Util.	CPU Idle
09:15	1878	82	18
09:20	1926	85	15
09:25	1964	82	18
09:50	1924	71	29
10:15	1875	70	30
11:10	1651	62	38
12:10	1290	47	53
13:10	1517	61	39
14:00	1491	56	44
15:00	1442	56	44
15:10	1080	42	58
15:30	823	37	63
15:50	600	26	74

(2) 3호기(CPU: 8, Memory: 2GB)

<표 4> 3호기 CPU 사용률 데이터

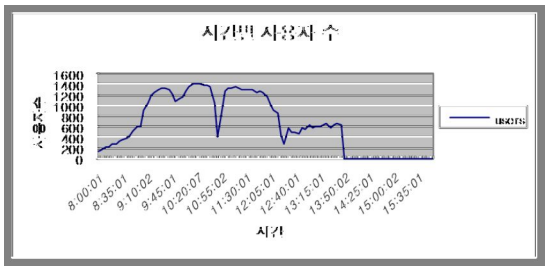
시간	사용자 수	CPU Util.	CPU Idle
09:15	1843	97	3
09:20	1826	89	11
09:25	1874	92	8
09:50	1944	83	17
10:15	1810	68	32
11:10	1676	63	37
12:10	1295	41	59
13:10	1585	61	39
14:00	1560	61	39
15:00	1494	54	46
15:10	1131	42	58
15:30	791	29	71
15:50	598	21	79

(3) 9호기(CPU: 4, Memory: 2GB)

psradm명령어를 이용해 4 CPU를 오프라인 시킨 후 테스트를 실시한다. 왜냐하면 CPU 개수에 따른 최적 사용자 수, 적정 사용자 수, 최대 사용자 수를 분석하기 위함이다.

<표 5> 9호기 CPU 사용률 데이터

시간	사용자 수	CPU Util.	CPU Idle
09:15	1184	100	0
09:20	1265	100	0
09:25	1302	100	0
09:50	1094	100	0
10:15	1407	100	0
11:10	1350	91	9
12:10	927	74	26
13:10	618	66	34
14:00	0	52	48
15:00	0	52	48
15:10	0	47	53
15:30	0	47	53
15:50	0	50	50



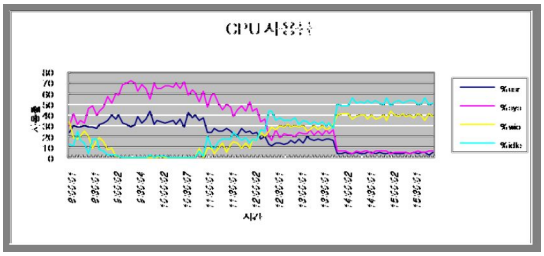
<그림 2> 9호기 시간별 사용자 수

(4) 사용자 수에 따른 시스템 자원

7월 8일 9시 35분 최대 사용자 현황은 <표 6>과 같다. 6호기, 9호기는 실제 장애 상태에서 분석을 하였고, HTS은 서버의 장애가 있더라도 계속해서 운영이 되도록 시스템 아키텍처가 설계되어 있다.

<표 6> 7월 8일 최대 사용자 현황

K-net	01	02	03	04	05	06	07	08	09	Total
Users	1947	1961	1911	1891	1982	0	1685	1908	0	13285
CPU Util	64%		75%							



<그림 3> 9호기 CPU 사용률

7월 9일 9시 45분 최대 사용자 현황은 <표 7>과 같다. 6호기, 9호기는 실제 장애 상태에서 분석을 하였다.

<표 7> 7월 9일 최대 사용자 현황

K-net	01	02	03	04	05	06	07	08	09	Total
Users	1924	1924	1956	1863	1968	0	2000	2096	0	13481
CPU Util	73%		80%							

CPU 분석결과는 다음과 같다. 클라이언트 프로그램에서 최초 접속 시 CPU 사용률이 높으며 3호기, 9호기 CPU의 사용률을 보면 HTS의 절정시간대인 오전 9시부터 10시까지 사용자 수가 증가함에 따라 CPU Util. 은 모두 한계치를 초과해 사용함으로써 시스템 성능이 저하돼 있었으며 이로 인해 CPU 증설이 필요로 하며, 결과적으로 CPU 개수에 따른 적정 사용자 수 및 최대 사용자 수를 <표 8>에서 산출하였다.

<표 8> CPU 개수에 따른 사용자 수

CPU개수	최적 사용자 수	적정 사용자 수	최대 사용자 수
4	600	1000	1400
8	1600	1800	2500
12	2100	2400	3400

현재 HTS에서 운영 중인 프로그램 기준으로 산출하였으며 향후 확장 시 1800 사용자당 CPU 8개의 시스템 단위로 확장하는 것이 바람직하다. 또한 중계 게이트웨

이에서 HTS 서버로의 최초 연결 시 동시 접속의 수를 줄임으로써 로그인으로 인한 CPU 부하를 줄일 수 있다.

4.2 Memory 분석 및 결과

HTS 운영 중에 top명령어를 이용하여 시스템의 사용자 및 프로세스의 증가에 따른 Memory 사용률을 조사함으로써 현 시스템의 Memory 필요 용량을 찾아낸다. 1호기, 3호기는 튜닝을 했는데 9호기가 빠진 이유는 9호기는 3호기와 서버 사양이 같기 때문이다.

(1) 1호기(CPU: 12, Memory: 3GB)

<표 9> 1호기 Memory 사용률 데이터

시간	사용자 수	Process 수	Free Memory	Free Swap
09:20	1926	2688	49 MB	1998 MB
10:20	1875	2569	74 MB	2002 MB
11:10	1651	2390	91 MB	2005 MB
13:10	1517	2280	55 MB	2009 MB
15:30	823	1512	291 MB	2007 MB

(2) 3호기(CPU: 8, Memory: 2GB)

<표 10> 3호기 Memory 사용률 데이터

시간	사용자 수	Process 수	Free Memory	Free Swap
09:20	1826	2544	112 MB	2024 MB
10:20	1810	2400	76 MB	2024 MB
11:20	1676	2326	61 MB	2024 MB
13:10	1585	2266	41 MB	2024 MB
15:30	791	1399	315 MB	2022 MB

Memory 분석결과는 다음과 같다. Memory는 swapping이 없이 적절히 구성되어 있으며 일반적으로 CPU 처리용량과 개수에 따라 Memory를 결정할 것을 권고한다. (<표 11> 참조)

<표 11> Memory 용량산정 표

CPU Spec	요구 Memory
UltraSPARC 400MHz	256 MB

4.3 Disk 분석 및 결과

df 명령어는 마운트(mount)된 파일 시스템 각각의 정보를 나타내며, -k 옵션의 의미는 사용량은 Kbyte 단위로 나타내고 사용가능한 공간에서 운영체제가 사용하고 있는 공간을 나타낸다. df -k 명령어[9]를 이용하여 현재 분할된 파일 시스템, 디스크 용량에 대해 살펴봄으로써 실제 사용되는 디스크의 크기를 알 수 있다. 1호기, 3호기는 튜닝을 했는데 9호기가 빠진 이유는 9호기는 3호기와 서버 사양이 같기 때문이다.

(1) 1호기(CPU: 12, Memory: 3GB)

<표 12> 1호기 Disk 사용률 데이터

파일 시스템	총용량	사용 용량	남은 용량
/	3.9 GB	약 530 MB	약 3.4 GB
/hts01	8 GB	약 3 GB	약 4.9 GB
/hts02	8 GB	약 2.1 GB	약 5.9 GB
/hts03	8 GB	약 0 5 GB	약 7.5 GB
/hts04	8 GB	약 7 GB	약 0.9 GB
swap	4 GB	약 1 MB	약 4 GB

(2) 3호기(CPU: 8, Memory: 2GB)

<표 13> 3호기 Disk 사용률 데이터

파일 시스템	총용량	사용 용량	남은 용량
/	3.9 GB	약 600 MB	약 3.3 GB
/hts01	8 GB	약 3.6 GB	약 4.4 GB
/hts02	8 GB	약 0 GB	약 8 GB
/hts03	8 GB	약 0 GB	약 8 GB
/hts04	8 GB	약 6.3 GB	약 1.7 GB
swap	3 GB	약 1 MB	약 3 GB

Disk 분석결과는 다음과 같다. Disk 용량은 대부분 여유가 있으며, 1호기의 경우를 제외하고는 /hts01, /hts04만 주로 사용되고 있으며, 향후 Disk에 저장 될 용량이 확장 되지 않는다고 가정할 경우 16GB 정도면 운영이 가능하다.

4.4 애플리케이션 분석 및 결과

HTS의 애플리케이션이 운영되는 동안 높은 CPU 점유율을 보이는 프로세스를 찾아내어 소스 코드의 개선점을 찾아내고 성능 개선의 효과를 보일 수 있는 모듈에 대한 제안을 한다.

(1) 현 프로그램의 수정 사항

logon_chk 프로세스가 비교적 높은 CPU 점유율을 보이고 있다.

```
[K-net02:/hts01/newhts/run/logdir] /usr/ucb/ps -aux | head
 USER       PID %CPU %MEM   SZ  RSS TT       S    START  TIME COMMAND
 w3hts     1411  2.5  0.1 3240 1992 ?        O 00:11:10 72:49 /hts01/newhts/run/
 w3hts     1745  0.6  0.2 7256 6520 ?        S 00:11:12 18:55 /hts01/newhts/run/
 w3hts     1409  0.5  0.1 2240 2184 ?        S 00:11:10 18:22 /hts01/newhts/run/
 w3hts     1718  0.5  0.1 2728 2016 ?        S 00:11:11 13:05 /hts01/newhts/run/
 w3hts     1746  0.5  0.2 6648 6128 ?        S 00:11:12 12:59 /hts01/newhts/run/
 w3hts     2916  0.3  0.1 2272 1296 ?        S 00:16:20 10:38 /hts01/newhts/run/
 w3hts     4155  0.3  0.1 2272 1296 ?        S 00:22:52 10:26 /hts01/newhts/run/
 w3hts     4147  0.3  0.1 2256 2200 ?        S 00:22:47  9:12 /hts01/newhts/run/
 w3hts    19329  0.3  0.1 2584 1672 ?        S 08:38:27  8:39 /hts01/newhts/run/
[K-net02:/hts01/newhts/run/logdir] ps -ef | grep 1411
  w3hts 20644 17212   0 13:27:04 pts/11    0:00 grep 1411
  w3hts  1411     1   3 00:11:10 ?         72:57
  /hts01/newhts/run/csad/obj/logon_chk
```

<그림 4> CPU를 많이 차지하는 프로세스 확인

logon_chk 소스 프로그램을 분석하면 다음과 같다. while 루프 내에서 shmget() 함수와 shmat() 함수를 매번 호출하여 필요 이상의 시스템 호출을 하고 있는데 이 부분을 루프 밖으로 분리시켜 LogonUserCheck() 함수 초기에 단 1회만 수행되도록 프로그램 수정이 필요하다.

```
LogonUserCheck(argv)
Byte     **argv;
{
struct  INFORM_TABLE     *inform;
struct  CLIENT_TABLE     *client, *nowclt;
Byte     *csamt;
struct  LOGONUSERuserbuf;
Long     nowtime;
Long     userptr;
Long     i;
Long     shmid;

while(1) {
        if((shmid = shmget(SMKEY_CSAM, 0, 0666)) == -1){
                        AllUserOutLog();
                      exit(0);
        }
        if((csamt = (Byte *) shmat(shmid, 0, 0)) == (Byte *) -1){
                        AllUserOutLog();
                        exit(0);
        }
 ...........
```

<그림 5> 수정 전 소스 프로그램

```
LogonUserCheck(argv)
Byte     **argv;
{
struct  INFORM_TABLE     *inform;
struct  CLIENT_TABLE     *client, *nowclt;
Byte     *csamt;
struct  LOGONUSERuserbuf;
Long     nowtime;
Long     userptr;
Long     i;
Long     shmid;

if((shmid = shmget(SMKEY_CSAM, 0, 0666)) == -1){
        AllUserOutLog();
        exit(0);
}

if((csamt = (Byte *) shmat(shmid, 0, 0)) == (Byte *) -1){
        AllUserOutLog();
        exit(0);
}

   while(1) {
   ......
```

<그림 6> 수정 후 소스 프로그램

(2) 향후 개선점

같은 내용의 메시지를 여러 개의 프로세스에게 보내기 위해 내용은 같지만 타입이 다른 메시지를 메시지 큐를 통해 보내고 있는데 이러한 방식을 사용할 경우 메시지 큐에 대한 입출력 동작이 여러 프로세스에서 동시에 일어난다. 이러한 부하를 줄이고자 공유메모리 내에 해당 데이터를 한 번만 세팅하고 여러 프로세스에 시그널

을 보내주면 각 프로세스가 공유메모리를 참조하여 데이터를 얻을 수 있도록 IPC방식의 변화를 고려해야 한다.

V. 결론 및 제안

3호기, 9호기 CPU의 사용률을 보면 HTS의 절정시간대인 오전 9시부터 10시까지 사용자 수가 증가함에 따라 CPU Util. 은 한계치를 초과해 사용함으로써 응답속도 저하 등으로 인해 시스템 성능이 저하되었다. 이에 대한 성능개선 방안은 CPU 개수를 각각 2개 더 증설해야 한다. 그리고 CPU 개수에 따른 적정 사용자 수를 살펴보면, CPU 개수를 4개로 운영할 경우 시스템의 적정 사용자수는 1000명 정도이고 초과 시 사용자의 응답시간이 늦어지는 현상이 발생함으로 CPU 4개의 시스템으로 운영할 경우 1000명을 초과해서는 안 된다. 최대 1420 사용자 수일 경우 시스템이 과부하 상태로 클라이언트 프로그램이 요청에 대해 정상적인 응답을 받을 수 없는 상태에 도달한다. 또한 CPU 8개로 확장할 경우 적정 사용자 수는 1800명 정도이고 초과 시 사용자의 응답시간이 늦어지는 현상이 발생한다. 따라서 향후 사용되는 서버의 사양은 8CPU, 2GB Memory, 18GB Disk의 성능을 가진 시스템으로 하는 것이 현재의 K증권의 HTS (Home Trading System)로서 가장 적합하다.

게이트웨이에서 HTS 시스템으로 최초 연결 시 가능한 여러 서버에 분산시키는 것이 초기에 시스템의 부하를 최소화시켜 최적의 응답시간을 기대할 수 있다.

전반적으로 애플리케이션은 대부분 시스템 자원을 적절히 사용하고 있지만, logon_chk 프로세스가 비교적 높은 CPU점유율을 차지하고 있어 시스템 성능이 저하된 상태이다. 이에 대한 성능개선 방안은 logon_chk 소스 프로그램을 분석해 보면, while 루프 내에서 shmget()함수와 shmat()함수를 매번 호출하여 필요 이상의 시스템 호출을 하고 있어 이 부분을 루프 밖으로 분리시켜 LogonUserCheck()함수 초기에 단 1회만 수행되도록 프로그램을 수정하여 실행함으로써 수정전과 수정후의 실행 결과는 똑같지만 CPU점유율을 낮추어 응답시간 단축 등 시스템 성능을 개선하였다.

이 연구를 통해 향후 제안 서버로는 E5500과 성능은 동일하지만 설치 공간을 고려하여 rackmount가 가능한 Sun E4500모델을 제안한다. E4500은 하나의 데이터센터 캐비닛에 4개의 E4500을 장착할 수 있다.

<표 14> 제안 서버 사양

모델	CPU(400MHz)	Memory	Disk
Sun E4500	8	2GB	I:4GB*2, E:8GB*4

우리는 이 연구에서 HTS의 CPU, Memory, Disk, 애플리케이션 등 전반적인 분석을 통해 응답속도 향상 등 시스템 성능이 개선되었을 뿐만 아니라 보다 더 효율적이고 실무적으로 안정적인 시스템 운영을 할 수 있다.

참고문헌

[1] HP Computer Systems Training Course, "HP-UX Performance and Tuning under SAP/R3," Version A, Oct. 1995, pp. 4-7.

[2] T. Bemmerl, "Distributed Monitoring Systems - A Basis for General Purpose Distributed Multiprocessors", Pannel Section, Proc. of the Int'l Conf. Distributed Computing Systems, May 1991, p. 380.

[3] P. C. Bates, "Effective Instrumentation is the Key to Effective Monitoring", Pannel Section, Proc. of the Int'l Conf. on Distributed Computing Systems, May 1991, p. 379.

[4] 유응준, "분산 UNIX 시스템의 시스템 성능 관리 및 튜닝에 관한 연구", 연세대학교 산업대학원 석사학위논문, 1996.

[5] 김현호, "성능평가를 이용한 분산 파일시스템의 구성방안", 연세대학교 산업대학원 석사학위논문, 1996.
[6] 김현호, "네트워크 파일 시스템 환경에서의 성능평가," 한국지식정보기술학회 추계학술대회 제2권 제2호, Dec. 2008, pp. 101~109.
[7] SUN, http://docs.sun.com/app/docs/doc/816-5166/psradm-1m?l=en&a=view&q=psradm+, 2009.
[8] SUN, http://docs.sun.com/app/docs/doc/816-5165/sar-1?l=en&a=view&q=sar, 2009.
[9] SUN, http://docs.sun.com/app/docs/doc/816-5166/df-1m?l=en&a=view&q=df, 2009.

논문접수일 : 2009년 3월 30일
수 정 일 : 2009년 5월 30일
게재확정일 : 2009년 6월 17일

■ 저자소개 ■

김 현 호
Kim, Hyun Ho

2003년 3월~현재 명지전문대학 컴퓨터정보과 부교수
2006년 2월 성균관대학교 전기전자 및 컴퓨터공학과(공학박사)
1997년 2월 연세대학교 전자계산전공(공학석사)
1988년 11월~2000년 3월 LG 전자 컴퓨터사업부 시스템 엔지니어
1987년 2월 성균관대학교 전자공학과(공학사)

관심분야 : Mobile Computing, Ad-Hoc Net-works, Sensor network.
E-mail : kimhh@mail.mjc.ac.kr

박 용 덕
Park, Yong Duck

1999년 2월~현재 한국썬마이크로시스템즈 시스템 전략사업부 이사
1990년 1월~1999년 1월 LG 전자 컴퓨터사업부 시스템 엔지니어
1990년 2월 성균관대학교 전자공학과(공학사)

관심분야 : Server Consolidation / Virtualization, Data Warehouse, Project Management
E-mail : ongduck.park@Sun.com