아파치 웹서버에서의 장애시 대처 방법

시스템 관리자는 시스템에 대해 보수적이어야 한다는 말이 있다. 이 “보수적” 이라는 말은 특정한 정치적 성향을 뜻하는 것이 아니라 시스템에 대해 단 1%라도 문제가 될 수 있는 여지를 제거하고, 각 유저에게는 불편함을 주지 않는 한도내에서 가급적 사용 권한을 제한하여 혹시나 있을지 모를 시스템의 침해나 크래쉬등에 대비하는 것을 뜻한다. 아울러 새로운 룰을 적용 또는 변경하거나 프로그램을 새로이 도입할 때에도 이 프로그램으로 인하여 다른 문제가 발생할 수 있는 여지는 없는지 여부를 신중히 검토 후 도입하여야 한다.

따라서 이번 호에서는 아파치 웹서버를 운영해 오면서 서버 관리자라면 누구나 접할 수 있는 여러 가지 장애와 이에 따르는 원인과 해결 방법을 찾아보도록 하겠다.

오늘과내일 넷센터 홍석범 (antihong@tt.co.kr)

장애 1. 접속이 갑자기 느려지거나 아예 접속이 안 될 때

가끔 접속자가 많은 서버를 운영하다 보면 갑자기 웹 접속이 되지 않거나 접속이 너무 느려 아파치 데몬 개수를 확인해 보면 httpd 가 256 개나 떠 있는 경우가 있다. 기본적으로 아파치 웹서버의 경우 MaxClients 가 256 으로 설정되어 있어 동시에 256 개의 데몬이 뜨게 되면 더 이상의 접속을 받아들이지 않고, 기존의 프로세스가 죽을 때까지 대기한 후 접속이 끊기게 되면 그제서야 접속을 받아들이게 된다. 따라서 동시 접속이 많은 경우에는 이전의 웹 접속이 끊길 때까지 대기를 하여야 하므로 접속 속도가 느린 것처럼 느끼게 되는 것이다. 일반적으로 정상적인 접속의 경우에 256 개의 프로세스가 모두 뜨는 경우는 그리 많지 않기에 현재의 상태가 비정상적인 접속인지 여부를 판단하여야 한다. 이를 판단할 수 있는 방법은 netstat –na | grep ES 로 ESTABLISHED 된 연결 상태를 확인하여 클라이언트의 IP 가 정상적인 연결인지 여부를 확인하면 된다. 또는 netstat -na|grep ES|awk '{print $5}'|sort 로 클라이언트의 IP 만 따로 Sort 하여 확인하여 보도록 한다. 통상적으로 HTTP 1.1 규약에서부터 적용되기 시작한 KeepAlive 기능을 지정하였을 경우 한 클라이언트 IP 에서 동시에 3-5 개정도의 http 프로세스를 생성하므로 한 IP 에서 3-5 개 정도의 프로세스를 생성하는 것은 정상적인 현상이다. 비정상적인 접속의 경우에는 아래와 같은 이유가 있을 수 있다.

(1) 서비스 거부 공격(DoS) 의 경우

동시에 서비스할 수 있는 프로세스의 한계가 있다는 점을 악용한 서비스 거부 공격일 가능성이 있다. 이미 한번의 실행으로 100 개나 200 개등 원하는 만큼의 동시 접속을 맺은 후 이 접속을 끊지 않고 유지할 수 있는 공격 코드가 인터넷상에 공개되어 있다. 그러나 이러한 공격의 경우 공격지의 IP 를 속이기가 매우 어려우므로 netstat 으로 확인 후 비정상적인 접속으로 확인시 해당 IP 를 차단하면 된다.

특정 IP 의 라우팅을 차단하는 방법은 아래와 같이 route 를 이용한 방법과 iptables (커널 2.4 이상) 를 이용한 방법 이렇게 두 가지가 있다.

예) 공격지 IP 인 211.40.4.6 으로부터의 라우팅을 차단하는 설정

```
# route add –host 211.40.4.6 reject
# iptables –A INPUT –s 211.40.4.6 –j DROP
```

실제 적용되었는지 확인하는 방법은 각각 route –n 과 iptables –L –n 이다.

참고로 TCP SYN Flooding 공격의 경우 SYN 패킷만 대량으로 발송할 뿐 ESTABLISHED 상태가 되지 않으므로 TCP SYN Flooding 공격과는 무관하다.

(2) include 를 잘못하여 무한 루프가 돌 경우

요즘에는 php 와 mysql 을 연동하여 많이 사용하고 있는데, 프로그래밍 과정에서의 실수로 php 파일에서 같은 php 파일을 include 하는 경우가 있다. 또는 a.php 파일에서 b.php 파일을 include 하고 b.php 파일에서 다시 a.php 파일을 include 하는 경우도 그러한 경우일 것이다. 이러한 경우에는 무한 루프가 돌게 되어 결국은 아파치 데몬이 금새 Maxclients 에서 지정한 개수로 차 버리게 되는데, 어떤 파일에서 무한 루프가 돌고 있는지 찾기가 힘들다.

따라서 임시로 아래와 같이 include 를 하지 못하도록 차단을 하는 방법이 있다.

```
# iptables –A    INPUT -p tcp -i lo -s xxx.xxx.xxx.xxx    --sport 1024:65535 -j DROP
```

이는 같이 서버내에서 include 시에는 lo (Lookback Interface) 를 통해 sport 가 1024 이후의 high port 를 이용하여 통신한다는 특성을 이용한 것이다. 그러나 이 설정을 하였을 경우 로컬 서버에서 클라이언트 포트를 전혀 사용할 수 없게 되므로 다른 서비스에도 장애가 되기 때문에 임시로만 사용하기 바란다.

또는 ps aux | grep http 로 보이는 프로세스에서 ls –la /proc/pid/ 로 각각의 http 프로세스가 어떤 파일을 참조하고 있는지 일일이 추적하는 방법도 있다.

(예:cwd -> /home/user1/public_html/infinite_loop/)

정상적인 접속의 경우에는 아래와 같이 대처한다.

(1) KeepAlive 옵션 변경

기본값으로 설정되어 있는 KeepAlive On 을 KeepAlive Off 로 변경 후 아파치를 재시작한다. KeepAlive 는 HTTP 1.1 규약에서부터 적용된 것으로 접속 속도에 큰 영향을 준다. KeepAlive 를 Off 로 설정시 다소 접속 속도는 떨어지지만 좀 더 많은 동시 접속을 수용할 수 있다. 따라서 MaxClients 에 도달할 정도로 동시 접속자가 많은 경우에는 KeepAlive 를 Off 로 설정하는 것이 다소 임시 방편이기는 하지만 해결 방법이 될 것이다.

KeepAlive 설정에 대해서는 Hit 의 개념과 관련 지어 이해하면 된다. 예를 들어 10 개의 이미지 파일을 링크한 HTML 페이지를 로딩시 웹브라우저는 이 HTML 파일을 다운로드하여 클라이언트에서 파싱(parsing) 을 하면서 이미지 파일등이 링크되어 있을 경우 서버에 접속하여 이미지 파일을 요청하는데, KeepAlive 가 On 일 경우에는 한 번 맺은 TCP 연결에 대해 같은 Client IP 에서 접속이 있을 것이라 가정하고 기존의 프로세스가 대기하고 있다가 이후의 접속을 처리하기 때문에 다시 접속을 맺는 절차가 필요 없이 빨리 서비스가 가능하지만, KeepAlive 가 Off 인 경우에는 이미지 파일을 불러올 때마다 매번 세션을 새로 맺고 끊는 과정을 반복하여야 하기 때문에 속도가 느려질 수 밖에 없다. 아파치 홈페이지의 문서에 의하면 많은 이미지 파일이 있는 HTML 문서를 로딩시 KeepAlive 설정에 따라 최고 50%까지 속도 차이가 날 수 있다고 한다. 그렇다고 해서 모든 사이트에서 KeepAlive 를 On 으로 하는 것이 좋은 것이 아니다. 순간적인 동시 접속자는 많지만 한 두 번 검색 후 검색 결과의 링크를 따라 다른 사이트로 빠져 나가는 검색 엔진의 경우에는 KeepAlive 를 Off 로 하는 것이 유리할 것이다. KeepAlive 를 On 으로 설정하여 그대로 사용할 경우에는 15 초로 설정된 KeepAlive Timeout 을 15 초에서 5 초 정도로 낮게 설정하는 방법도 있으며 이 값은 자신의 시스템 환경에 맞게 적절히 설정하기 바란다.

(2) 아파치의 MaxClients 조절

기본적으로는 256 으로 설정되어 있는 MaxClients 의 한계를 512 나 1024 와 같이 적절히 변경한다. 그러나 이 값을 변경하기 위해서는 아파치의 소스를 수정 후 다시 컴파일 하여야 하므로 아파치의 소스 디렉토리에 있는 src/include/httpd.h 파일에서 HARD_SERVER_LIMIT 256 로 설정된 값을 512 나 1024 로 변경 후 아파치를 재컴파일 하면 된다. 만약 커널 2.2.X 일 경우에는 /usr/src/linux/include/linux/tasks.h 에서 NR_TASKS 와 MAX_TASKS_PER_USER 변수 역시 수정한 후 커널을 재컴파일 해 주어야 하며, 2.4.X 의 경우에는 관련된 커널 제한이 없어졌으므로 아파치만 재컴파일 하면 된다.

그러나 대부분의 사이트에서는 256 정도로 설정되어도 충분히 서비스가 가능하므로 무작정 이 값을 크게 늘려 메모리를 낭비할 필요가 없으니 특별한 경우가 아니라면 이 값을 늘리지 않는 것이 좋다.

(3) 추가 아파치 데몬 설정

만약 여러 도메인중 특정 도메인이나 어떠한 사이트내 특정 컨텐츠의 접속이 특별히 많아 같은 서버에 있는 다른 사이트에까지 피해를 주고 있다면 이 부분을 별도로 데몬을 띄워 서비스하는 방법도 있다. 이를테면 한 사이트에서 게시판의 접속이 매우 많다면 기존의 80 번 포트외에 8080 과 같은 임의의 포트로 작동하는 웹 데몬을 추가로 띄워 이 포트를 통해 접속이 많은 서비스를 담당하게 하는 것이다. 이를 위해서는 기존의 httpd.conf 파일을 httpd8080.conf 와 같이 설정 파일을 복사 후 httpd8080.conf 파일을 아래와 같이 변경하면 된다.

port 80 → port 8080

User nobody → User www

Group nobody → Group www

그리고 /usr/local/apache/bin/httpd –f /usr/local/apache/conf/httpd8080.conf 와 같이 실행하면 8080 포트로 작동하는 웹서버 데몬을 추가로 띄우게 되는 것이다. 물론 이때 www 라는 계정은 서버에 생성되어 있어야 하며 Nobody 가 아닌 www 라는 별도의 계정으로 데몬을 작동하는 이유는 한 유저(nobody) 가 생성할 수 있는 프로세스의 한계가 있기 때문이며 커널 2.4.X 에서는 이 제한이 없으므로 Nobody 로 작동해도 관계 없다.
또는 기존의 웹데몬인 httpd 와 파일 이름을 다르게 하여 서로 구별을 쉽게 하기 위해 httpd 대신 httpd8080 등 다른 이름으로 변경하여 실행하여도 좋다.
웹 접속은 http://domain.com:8080/ 으로 하면 되며 이러한 방식으로 8081, 8082, 8083….등의 여러 포트를 띄울 수 있다. 실제로 얼마 전 필자가 운영하는 호스팅 서버에서 특정 사이트의 게시판의 접속이 폭주하여 모든 웹 접속이 느려진 적이 있었는데, 위와 같이 게시판 부분만 따로 떼어 8080 포트로 분리하여 서비스를 하여 문제를 해결한 적도 있다.

장애. 프로세스가 과도한 메모리를 사용할 경우.

특별히 이상한 프로세스는 없는데, 시스템의 부하가 갑자기 올라가는 경우가 있다.
심할 경우에는 콘솔에서 키 작동이 되지 않아 제어 자체가 불가능한 경우도 있는데, 이러한 경우에는 어쩔 수 없이 시스템 reboot 밖에 방법이 없다. 이는 의도적이든 그렇지 않든 유저가 cgi 나 php 등의 프로그램을 잘못 짜서 과도한 메모리를 소모하는 프로세스를 실행하기 때문인 경우인데, 실제로 시스템의 메모리를 많이 소모하는 프로그램을 짜는 것은 단 몇 줄의 코드로도 가능하다. 아래와 같이 메모리를 소모하는 간단한 C 코드를 작성해 보자.
참고로 스크립트 키드의 악용을 막기위해 코드 중 일부를 변경하였다.

```
/*  memory.c  */

#include <unistd.h>
#include <stdlib.h>

void html_content(void);

int main() {
```

```
    char *some_memory;
    int    size_to_allocate = ONE_K;
    int    megs_obtained = 0;
    int    ks_obtained = 0;

    html_content();
    printf("Program Executed !!!<p>");

    while (1) {
        for (ks_obtained = 0; ks_obtained < 1024; ks_obtained++) {
            some_memory = (char *)malloc(size_to_allocate);
            if (some_memory == NULL) exit(EXIT_FAILURE);
            sprintf(some_memory, "Hello World");
        }
      printf("Now allocated %d Megabytes<br>\n", megs_obtained);
    }
    exit(0);
}

void html_content(void)
{
    printf("Content-type: text/html\n\n");
}
```

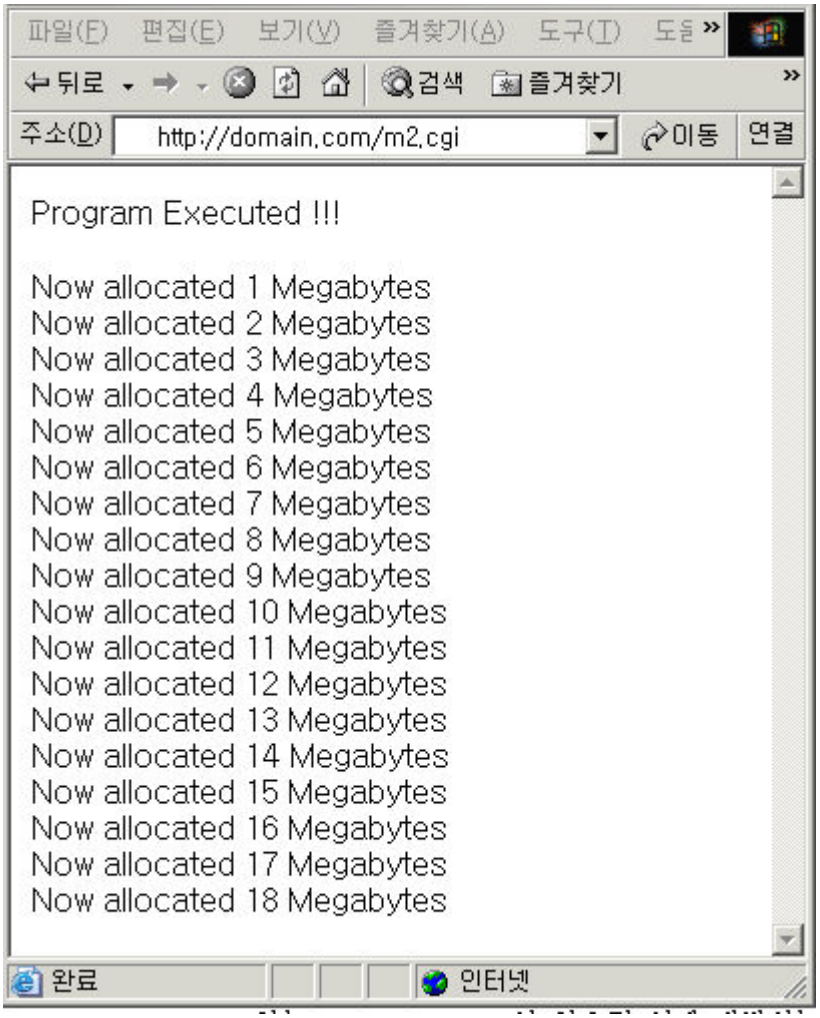
작성후 gcc –o memory.cgi memory.c 로 컴파일 한 후 이 파일을 http://domain.com/memory.cgi 와 같이 웹에서 실행해 보도록 하자. 아무리 메모리와 CPU 가 많은 시스템이라 하더라도 이 프로그램을 실행하자 마자 거의 통제 불능 상태로 되어 결국 시스템 재부팅을 하게 될 것이다. 이렇듯이 메모리를 무한정 소모하는 것을 차단하기 위해서는 아파치의 설정 인자(Directive) 중에서 RLimitMEM 을 사용하여 차단할 수 있다. 이 인자는 아파치 웹 서버에서 생성된 특정 프로세스가 작동시 소요 가능한 최대 메모리의 양을 제한하는 것으로 메모리를 많이 소모하는 CGI 가 작동할 때 이 인자에서 지정된 메모리까지만 실행이 되고 그 이상 소요시에는 더 이상 작동하지 않도록 해 준다.

예를 들어 httpd.conf 에

```
RLimitMEM 20480000 21504000

<Directory /home/special/public_html/*>
 RLimitMEM 51200000 52224000
</Directory>
```

와 같이 설정하였다면 모든 디렉토리에서는 메모리를 20 메가나 최대 21 메가까지만 사용이 가능하고 /home/special/public_html/* 디렉토리 이하에 접근시에는 특별히 50 메가까지 메모리 이용이 가능하게 된다. 실제로 위와 같이 설정 후 memory.cgi 를 웹에서 호출하면 아래와 같이 일정량의 메모리만 사용되고 중단하는 것을 확인할 수 있다.



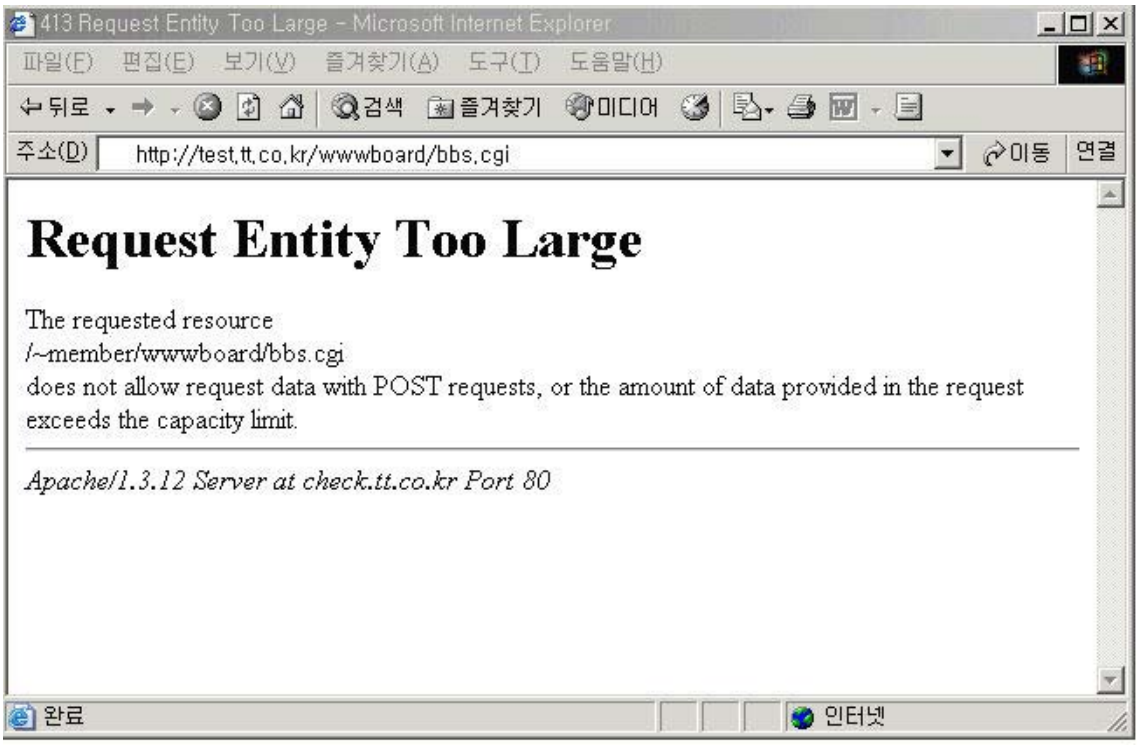
이와 비슷한 인자로 CPU 점유율을 제한하는 RLimitCPU 와 사용자당 프로세스의 개수를 제한할 수 있는 RLimitNPROC 이 있으며 이에 대해서는 http://httpd.apache.org/docs-2.0/mod/core.html 를 참고하기 바란다.

장애. 큰 사이즈의 파일을 업/다운로드하여 부하가 유발될 경우

누구나 업로드나 다운로드가 가능한 자료실을 운영하고 있을 경우 사이즈가 너무 큰 파일을 업로드 또는 다운로드 할 경우 부하가 많이 걸리게 되어 결국 시스템의 성능 저하를 유발하게 된다. 최근 배포되는 게시판/자료실 프로그램에서는 대부분 업로드 할 수 있는 용량등을 제한할 수 있는 기능이 있지만 그렇지 않은 프로그램도 상당수 있어 웹 서버 차원에서 이 제한을 적절히 설정할 필요가 있다. 아파치에서는 이와 관련하여 웹 서버에서 업로드/다운로드 할 수 있는 파일의 사이즈를 제한하는 LimitRequestBody 기능을 이용할 수 있다.
LimitRequestBody 는 클라이언트가 요청시 http 프로토콜을 통해 서버가 제공할 수 있는 메시지의 크기를 byte 단위로 정의하는 것으로 무한대를 의미하는 0 부터 2,147,483,647(2Giga) 까지 설정 가능하며 이 설정으로 대용량의 파일을 업/다운로드 하는 형태의 서비스 거부 공격을 차단할 수 있다. 이를 설정하는 방법은
httpd.conf 를 열어 아래의 라인을 추가하면 된다.

```
<Directory />
  LimitRequestBody 7168000
</Directory>
<Directory /home/special/>
  LimitRequestBody 10240000
</Directory>
```

위와 같이 LimitRequestBody 인자를 설정하면 아파치 웹서버를 이용하여 업/다운로드 하는 모든 파일의 사이즈를 7M 로 제한하고 /home/special/ 이하에 대해서는 10M 로 제한하게 된다. 위와 같이 설정시 지정된 사이즈를 초과하는 파일을 업로드/다운로드 시에는 아래 그림과 같은 에러 메시지가 뜨며 업로드/다운로드가 되지 않는다.



장애. 특정한 이름의 파일을 실행하지 못하도록 하고자 할 때

최근에는 대부분의 사이트들이 대화방(채팅)을 위해 자바로 프로그래밍된 자바 대화방을 사용하는 추세이지만 얼마전 까지만 하더라도 C 나 Perl 로 된 CGI 대화방을 설치하여 사용하는 것이 유행이었다. 그런데, 이 대화방의 경우 대화방에 참여하는 유저 중 한 명이 글을 입력할 경우 모든 유저에게 이 내용이 실시간으로 뿌려 주어야 하는 특성상 시스템의 메모리를 매우 많이 소모하는 대표적인 프로그램중 하나였다. 따라서 채팅 CGI 프로그램을 원천적으로 사용할 수 없도록 하는 방법을 고민하게 되었는데, 해결 방법은 대부분의 채팅 프로그램이 xxxchat.cgi 또는 chatxxx.cgi 라는 파일을 실행 파일 이름으로 한다는 특징을 이용하면 된다.

즉, httpd.conf 를 열어 아래와 같이 설정하면 파일명에 chat 이라는 단어가 포함된 CGI 파일은 작동하지 않게 되는 것이다.

```
<Files   "*chat*.cgi">
    Order allow,deny
    Deny from all
```

</Files>

이러한 방식으로 특정한 파일에 대해 웹에서의 접근을 차단할 수 있다. 따라서 CGI 파일에 아무리 소유권과 실행 권한을 주었다 하더라도 웹서버 자체에서 접근을 거부하였으므로 웹에서 접근하면 Forbidden 에러가 나게 된다. 이러한 식으로 특정 파일 이름을 가진 파일의 실행이나 접근을 차단할 수 있다.

장애. 검색 로봇의 접근을 차단하고자 할 때

갑자기 특정한 IP 주소에서 짧은 시간에 많은 접속을 하여 시스템의 부하가 올라가 웹 접속 로그를 살펴보니 아래와 같이 이해할 수 없는 내용이 남는 경우가 있다.

211.51.63.4 - - [26/Sep/2001:22:19:42 +0900] "GET /robots.txt HTTP/1.0" 404 285
211.51.63.4 - - [26/Sep/2001:22:19:43 +0900] "GET /index.asp HTTP/1.0" 404 284
211.51.63.4 - - [26/Sep/2001:22:19:43 +0900] "GET /index.php HTTP/1.0" 404 284
211.51.63.4 - - [26/Sep/2001:22:19:43 +0900] "GET /index.php3 HTTP/1.0" 404 285
211.51.63.4 - - [26/Sep/2001:22:19:43 +0900] "GET /default.htm HTTP/1.0" 404 286
211.51.63.4 - - [26/Sep/2001:22:19:43 +0900] "GET /default.html HTTP/1.0" 404 287
211.51.63.4 - - [26/Sep/2001:22:19:43 +0900] "GET /default.asp HTTP/1.0" 404 286
211.51.63.4 - - [26/Sep/2001:22:19:43 +0900] "GET /default.php HTTP/1.0" 404 286
211.51.63.4 - - [26/Sep/2001:22:19:43 +0900] "GET /default.php3 HTTP/1.0" 404 287
211.51.63.4 - - [26/Sep/2001:22:19:43 +0900] "GET /main.htm HTTP/1.0" 404 283
211.51.63.4 - - [26/Sep/2001:22:19:43 +0900] "GET /main.html HTTP/1.0" 404 284
211.51.63.4 - - [26/Sep/2001:22:19:43 +0900] "GET /main.asp HTTP/1.0" 404 283
211.51.63.4 - - [26/Sep/2001:22:19:44 +0900] "GET /main.php HTTP/1.0" 404 283
211.51.63.4 - - [26/Sep/2001:22:19:44 +0900] "GET /main.php3 HTTP/1.0" 404 284
211.51.63.4 - - [26/Sep/2001:22:19:44 +0900] "GET /home.htm HTTP/1.0" 404 283
211.51.63.4 - - [26/Sep/2001:22:19:44 +0900] "GET /home.html HTTP/1.0" 404 284
211.51.63.4 - - [26/Sep/2001:22:19:44 +0900] "GET /home.asp HTTP/1.0" 404 283
211.51.63.4 - - [26/Sep/2001:22:19:44 +0900] "GET /home.php HTTP/1.0" 404 283
211.51.63.4 - - [26/Sep/2001:22:19:44 +0900] "GET /home.php3 HTTP/1.0" 404 284

무작위로 index.php index.asp, index.php3, default.html, default.asp 등의 파일을 순서대로 요청하는 것으로 보아 검색 엔진일 가능성이 높다고 가정할 수 있다. 특히 robots.txt 파일을 요청하는 것으로 검색 엔진이라고 장담할 수 있을 것이다.
httpd.conf 에서 Logformat 를 common 대신 {User-agent} 변수를 추가하여 정의하면

서버에 접근하는 agent 정보도 알 수 있는데, UA(User Agent)는 일반적인 웹 브라우저 뿐만 아니라 검색 로봇이나 방랑 로봇등 웹서버에 접속하여 웹 페이지를 가져오거나 해석하는 모든 종류의 프로그램을 뜻한다. 이는 흔히 사용하는 Internet Explorer나 Netscape 등의 브라우저외에도 lycos의 spider 나 AltaVista의 Scooter와 같은 검색 로봇과 Teleport 나 WebZIP, GetRight 등 오프라인 브라우저 모두 UA의 범위에 속한다. 검색 로봇이 어떤 사이트를 방문하여 문서를 인덱싱 하거나 오프라인 브라우저가 페이지를 한꺼번에 요청하여 긁어가는 것은 일반 사용자가 웹 브라우저로 서버에 접속하여 원하는 페이지를 보는 일반적인 경우와 그 성격이 다르다. 여러 페이지를 동시에 요청하는 정도를 벗어나 아예 한 웹 사이트의 모든 페이지를 짧은 시간에 통째로 긁어가기도 하기 때문에 이러한 경우에는 서버에 매우 많은 프로세스를 생성하면서 웹 서버의 로드가 크게 올라가게 되는 것이다. 특히 DB와 연동하는 사이트의 경우에는 심할 경우 정상적인 서비스를 하지 못 할 정도이다.

모든 사이트가 검색 엔진에 등록될 필요는 없거나 또는 허용된 일부 유저만 접근이 가능한 페이지의 경우 로봇의 접근을 차단할 필요가 있으므로 이러한 경우에는 아래와 같이 설정된 robots.txt 파일을 웹 서버의 최상위 / 디렉토리에 두면 모든 검색 로봇이 /secure 디렉터리를 인덱싱하지 않는다.

```
User-agent: *
Disallow: /secure
```

"User-agent: *"는 모든 로봇를 대상으로 한다는 것을 뜻하며 예를 들어 AltaVista Scooter등 특정한 UA 에 대해서만 설정하고 싶다면 다음과 같이 하면 된다.

```
User-agent: scooter
```

검색로봇과 관련된 더 자세한 정보를 얻기 원한다면 아래의 사이트를 참고하기 바란다.

http://info.webcrawler.com/mak/projects/robots/robots.html

http://info.webcrawler.com/mak/projects/robots/norobots.html

아울러 웹서버에서 특정한 User-Agent 의 접근을 차단하고자 한다면 httpd.conf 에 아래와 같이 BrowserMatch 를 사용하여 설정해도 된다.

```
BrowserMatch "WebZIP" go_out
BrowserMatch "Teleport" go_out
BrowserMatch "GetRight" go_out
BrowserMatch "WebCopier" go_out
BrowserMatch "NetZip Downloader 1.0" go_out
BrowserMatch "NetZip-Downloader/1.0.62" go_out
BrowserMatch "Teleport Pro/1.29" go_out
BrowserMatch "Teleport Pro/1.24" go_out
BrowserMatch "Teleport Pro/1.26" go_out
<Directory /home/no-ua/>
   Options Includes ExecCGI
```

```
    AllowOverride None
    Order allow,deny
    Allow from all
    Deny from env=go_out
</Directory>
```

위와 같이 설정시에는 /home/no-ua/ 디렉토리 이하에 대해서는 go_out 이라는 변수에 지정한 WebZip 이나 Teleport 등 UA 프로그램의 접근을 차단하게 된다. 다른 UA 도 차단하고 싶으면 위와 같이 웹서버의 로그를 살펴보아 agent 정보에 남는 UA 를 go_out 으로 추가해 주면 된다.
같은 방식으로 만약 특정 디렉토리 이하에 대해서 MSIE 브라우저로 접근하지 못하도록 설정한다면 어떻게 하면 될까?
아래와 같이 BrowserMacth 를 이용하여 설정하면 agent 정보에 MSIE 라 설정되는 UA 는 차단될 것이다.

```
 BrowserMatch "MSIE" msie
<Directory />
  Options Includes ExecCGI
    AllowOverride None
    Order allow,deny
    Allow from all
    Deny from env=msie
</Directory>
```

최근에는 각종 로봇이 버전을 새롭게 하며 계속적으로 나오고 있으므로 지속적으로 로그를 살펴보아 접근 통제를 하고자 하는 UA 를 설정하는 것이 좋다.

장애. 외부에서 데이터를 무단 링크하여 부하가 유발될 때

최근에 와레즈 사이트를 통해 게임이나 오락, 동영상등 각종 데이터들이 공유되면서 와레즈 사이트에서 관련 없는 임의의 서버에 데이터를 업로드 한 후 무단 링크하여 서비스하는 경우가 많다. 이러한 경우 데이터 전송량이 갑자기 늘어 서버의 부하가 급격히 올라감은 물론 한정된 회선의 대역폭도 소모하게 되어 서버를 관리하는 관리자들에게는 이러한 무단 링크가 큰 골치 거리중에 하나이다.
대부분의 무단 링크가 대용량이기 때문에 이를 차단하기 위해 위와 같이 LimitRequestBody

를 이용하여 임의의 용량 이상의 데이터에 대한 업/다운로드 용량을 제한하는 방법도 있지만 아래와 같이 BrowserMatch 대신 SetEnvIFNoCase Referer 를 이용하는 방법도 있다.

```
SetEnvIFNoCase Referer "warez" link_deny
SetEnvIFNoCase Referer "free" link_deny
SetEnvIFNoCase Referer "home" link_deny

<FilesMatch "\.(avi|mpe?g|zip|asf|exe)$">
Order allow,deny
allow from all
deny from env=link_deny
</FilesMatch>
```

위와 같이 설정시에는 Referer 이 warez 또는 free 나 home 이 포함되었을 경우 이 사이트를 link_deny 라는 변수에 할당하고, 환경 변수가 link_deny 일 때는 확장자가 avi 나 mprg,zip,asf 등인 파일에 대한 access 를 제한하고 있다. 따라서 홈페이지 주소에 warez 나 free 또는 home 이 포함된 주소에서 위의 데이터를 무단 링크하였을 경우에는 이를 차단할 수 있다. Nocase 를 추가로 설정하였을 경우에는 대소문자를 가리지 않아 링크하는 사이트가 대소문자에 관계없이 http://my.warez.org/ 나 http://My.Warez.Org/ http://MY.WAREZ.ORG/ 모두 적용된다.

다른 확장자를 추가하고나 설정을 변경하고자 할 경우에는 정규식(Regurar Expression)을 이용하여 설정하면 된다. 또는 이와는 반대로 아예 httpd.conf 에서 아래와 같이 설정할 수도 있다.

```
<VirtualHost tt.co.kr>
  ServerAdmin webmaster@tt.co.kr
  DocumentRoot /home/tt/public_html
  ServerName tt.co.kr
  ServerAlias www.tt.co.kr
  SetEnvIf Referer tt\.co\.kr link_allow
  SetEnvIf Referer www\.tt\.co\.kr link_allow
  SetEnvIf Referer ^$ link_allow
  <FilesMatch ".(mpg|asf|wmv)$")
   Order Deny,Allow
   Allow from env=link_allow
   Deny from all
   </FilesMatch>
```

</VirtualHost>

위와 같이 설정하였을 경우 외부에서는 mpg, asf, wmv 확장자를 갖는 파일에 대해서는 일체의 무단 자료 링크가 불가능하게 되며 오직 link_allow 에서 지정한 도메인에서만 링크가 가능하게 된다. 위와 같이 설정 후 외부에서 무단 링크를 하였을 때 error_log 를 살펴 보면 아래와 같이 무단으로 링크한 자료의 다운로드가 작동하지 않는 것을 확인할 수 있다.

[Tue Sep 4 15:55:44 2001] [error] [client 211.230.82.78]
client denied by server configuration: /home/tt/public_html/data/3-2.wmv

장애. 아파치 데몬은 떠 있는데, 접속이 되지 않는 경우

분명 데몬은 정상적으로 떠 있고 MaxClients 에 도달하지도 않았는데, 실제로 접속이 되지 않는 경우가 있다. 이러한 경우라면 웹서버가 TCP SYN Flooding 공격을 받고 있을 가능성이 있다. netstat –na | grep SYN 으로 확인하여 많은 SYN_RECEIVED 프로세스가 보인다면 이 공격 때문이며 이러한 경우에는
sysctl –w net.ipv4.tcp_syncookies=1 또는
echo 1 > /proc/sys/net/ipv4/tcp_syncookies 로 syncookies 기능을 enable 하면 된다. Syncookie 기능은 일단 커널에서 지원되어야 하므로 이 설정이 적용되지 않으면 커널에서의 설정여부를 확인해 보아야 한다. 이 공격에 대한 보다 자세한 내용은 본지 7 월호에 실린 “TCP SYN Flooding 공격의 원인과 해결책” 을 참고하기 바란다.

장애. 코드레드나 Nimda 등의 웜공격으로 로그 파일의 크기가 커질 때

최근에 코드레드와 Nimda 등 Windows NT/2000 기반의 IIS 를 공격하는 무차별적인 웜 공격으로 인하여 부하가 유발되고 로그 파일이 불필요한 데이터로 채워지는 경우가 있다. 로그 파일의 크기가 커지는 것은 둘째치고서라도 당장 계속적으로 커지는 로그 파일 때문에 서버에 부하를 유발하는 것이 더욱 큰 문제이다. 서버에서 로그를 남기는 방식은 매번 클라이언트의 요청이 있을 때마다 웹 서버에서 패킷 헤더에 있는 클라이언트의 정보를 받아낸 후 로그 파일을 open 한 후 로그 파일을 읽어 파일의 제일 끝으로 이동하여 로그 정보를 추가한 후 파일을 close 하는 것인데, 불 필요한 요청이 있을 때마다 이 작업을 계속하여야 하

므로 서버에 부하를 유발함은 당연한 현상이다. 따라서 아예 로그를 남기지 않도록 하거나 로그를 남긴다 하더라도 불필요한 정보를 남기지 않도록 하는 것이 미소하게나마 서버의 성능을 높이는 것이 될 것이다. 그럼 실제로 불필요한 정보는 아예 로그를 남기지 않는 방법에 대해 알아보도록 하자.

이를테면 코드레드의 경우 아래와 같이 무차별적인 로그가 남게 되는데

2001/08/01 23:39:50.765446 152.158.99.4:58781 -> 211.233.38.193:80 [AP]

GET/default.ida?NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN

이러한 로그를 막기 위해서는 httpd.conf 파일을 열어

CustomLog 윗 줄에

SetEnvIf Request_URI "/default.ida$" cord-red 와 같이 정의한 후

CustomLog /usr/local/apache/logs/access.log combined 라고 설정되어 있는 부분을

CustomLog /usr/local/apache/logs//access.log combined env=!cord-red 라고 수정을 한다.

! 는 not 의 의미이므로 위 설정은 환경변수 cord-red 로 정의된 요청을 거부하라는 뜻이다.

위와 같이 설정 후 아파치를 재시작하면 코드 레드와 관련된 로그가 남지 않게 된다. 코드레드에 이은 Nimda 웜의 경우도 같은 방식으로 설정하여 로그에 남지 않도록 설정할 수 있다.

또한 같은 원리를 이용하여 iptables 를 이용하여 아예 패킷 자체를 차단할 수도 있는데, 코드 레드의 경우 default.ida 문자열을 요청한다는 특징을 이용하여

iptables –A INPUT -i eth0 -p tcp --tcp-flags ACK ACK --dport 80 \

-m string --string '/default.ida?' -j REJECT --reject-with tcp-reset 와 같이 iptables 의 –strings 를 이용하여 특정 문자열이 포함된 패킷을 차단하는 방법도 있기는 하지만 이 방법은 커널과 iptables 를 다시 컴파일 하여야 하는 문제가 있다. Nimda Worm 역시 이외 cmd.exe 와 root.exe 를 포함하므로 같은 방식으로

iptables -A INPUT -p tcp --tcp-flags ACK ACK --dport 80 -m string --string "cmd.exe" -j REJECT --reject-with tcp-reset

iptables -A INPUT -p tcp --tcp-flags ACK ACK --dport 80 -m string --string "root.exe?" -j REJECT --reject-with tcp-reset 와 같이 차단할 수 있다.

장애. ServerName 인자 관련

아파치 서버를 시작시 자주 만나는 에러와 관련된 설정 중 하나가 바로 아래와 같은 ServerName 이다.

/usr/local/apache/bin/apachectl start
httpd: cannot determine local host name.
Use the ServerName directive to set it manually.
./apachectl startl: httpd could not be started

대부분 ServerName 은 리눅스 설치시 입력한 호스트 이름을 자동으로 가지고 와 설정되나 DNS 상에 존재하지 않은 도메인명이나 설사 존재하더라도 로컬 서버가 아닌 잘못된 도메인을 정의시 이러한 현상이 나타난다. 이러한 경우에는 ServerName 을 실제 로컬 서버의 호스트 이름이나 IP 주소로 설정해 주어야 한다.
또한 ServerName 을 잘못 설정시 나타날 수 있는 현상중 하나가 http://domain.com/~user 와 같이 접속할 때의 문제이다. 즉, 서버내 계정 사용자의 홈페이지를 접속시 http://domain.com/~user/ 와 같이 접속하면 접속이 되나 http://domain.com/~user 와 같이 / 를 붙이지 않으면 접속이 되지 않는 경우이다.
클라이언트가 서버의 디렉토리에 접속시 끝에 / (trailing) 을 하지 않은 경우 서버는 클라이언트에게 / 을 붙여 다시 접속을 하라고 요청한다. 그렇지 않으면 상대 URL 경로를 인식하지 못하는 문제가 있기 때문이다. 만약 DNS 가 정상적으로 세팅되어 작동하고 있을 경우에는 문제가 없지만 그렇지 않은 경우에는 접속이 되지 않는 경우가 생긴다. 또는 위에서처럼 ServerName 에 지정된 호스트네임이 실제로 DNS 상에 리졸빙이 되지 않는 경우도 이러한 현상이 나타나므로 이러한 경우에는 httpd.conf 의 ServerName 옵션에 실제 서비스중인 도메인명으로 입력해 주면 된다.

장애. 특정 파일의 접근 제한이 되지 않을때

가끔 서버를 운영하다보면 특정한 디렉토리 이하에 대해서는 인증된 유저만 접속이 가능하게 한다거나 특정 IP 대역의 유저만 접근하도록 하고자 할 필요가 있을 때가 있다. 특정 디렉토리 이하에 대해서 접근을 제어하고자 할 때에는 .htaccess 를 사용하거나 httpd.conf 에서 <Directory> 를 이용하여 제어를 할 수 있지만, 만약 특정한 파일에 대해서 외부에서의 접근을 제한하고자 한다면 어떻게 하여야 할까?
이때에는 Location 을 사용하면 된다. httpd.conf 파일에 아래와 같이 설정시 모든 디렉토리 이하의 secret.html 파일에 대해서는 192.168.1.1 에서만 접근이 가능하게 된다.

```
<Location /secret.html>
   order deny,allow
   deny from all
   allow from 192.168.1.1
</Location>
```

만약 여러 도메인이 설치되어 있는 호스팅 서버의 경우 secret.tt.co.kr 도메인내 secret.html 에 대해서만 접근을 제어하고자 할 경우에는 아래와 같이 VirtualHost 설정에서 하면 된다.

```
<VirtualHost secret.tt.co.kr>
        ServerAdmin antihong@tt.co.kr
        DocumentRoot /usr/local/apache/htdocs/secret/
        ServerName secret.tt.co.kr
        <Location /secret.html>
          order deny,allow
          deny from all
          allow from 192.168.1.1
        </Location>
</VirtualHost>
```

또는 위와 같이 IP 가 아니라 특정한 ID/PW 를 입력한 유저에 대해서만 특정 파일에 대하여 접근을 허용하고자 할 때가 있다. 이러한 경우에는 httpd.conf 에 아래와 같이 설정하면 된다.

```
 <VirtualHost secret.tt.co.kr>
        ServerAdmin antihong@tt.co.kr
        DocumentRoot /usr/local/apache/htdocs/secret/
        ServerName secret.tt.co.kr
        <Files secret.html>
            AuthName "ID/PW 를 입력하세요."
            AuthType Basic
            AuthUserFile /usr/local/apache/htdocs/.htpasswd
            Require valid-user
        </Files>
</VirtualHost>
```

그리고 htpasswd –c .htpasswd id 로 .htpasswd 파일에 ID/PW 를 생성하여 secret.html 에 접근시 ID/PW 를 정확히 입력한 유저에 대해서만 접근이 가능하게 된다.

장애. 한글 도메인으로 접속이 되지 않을때

한글.com 형식의 한글 도메인 서비스가 1년 가까이 아직 정상적인 서비스가 되고 있지는 않지만 현재는 포워딩 방식으로나마 일부 사용 가능하다. 즉, 그 자체로 한글.com 으로의 사용은 불가능하며 단지 한글.com 을 접속시 다른 도메인으로 포워딩 할 수 있는 것이다. 현재 http://www.gabia.com/ 나 http://www.doregi.com/ 등에서 한글 도메인을 검색해 보면 예를 들어 팬메일.net 의 경우 BQ--3DJSZOSUY56A.NET 와 같이 BQ—형식의 RaceCode 문자로 되어 있는데, 실제로는 bq--3djszosuy56a.mltbd.net 와 같이 mltbd.net(한글.com 일 경우에는 mltbd.com, 한글.org 일 경우에는 mltbd.org) 의 2차 도메인 형식으로 서비스된다. 따라서 위와 같이 WHOIS 검색을 한 후 나온 RaceCode 을 DNS 서버에서 먼저 설정하여야 하는데, DNS 서버의 named.conf 에서 설정하여야 할 내용은 아래와 같다.

```
zone "bq--3djszosuy56a.mltbd.net" {
        type master;
        file "bq--3djszosuy56a.mltbd.net.zone";
};
```

bq--3djszosuy56a.mltbd.net.zone 파일의 형식은 일반 zone 파일 형식과 동일하게 설정하면 된다.

그리고 DNS 서버에서 지정한 해당 웹 서버에서는 아래와 같이 VirtualHost 설정을 하여 원래의 사이트인 http://panmail.net/ 으로 Redirect 하면 된다.
(현재까지는 포워딩 서비스만 제공하므로 Redirect 를 이용하여야 한다.)

```
<VirtualHost bq--3djszosuy56a.mltbd.net>
    ServerAdmin webmaster@bq--3djszosuy56a.mltbd.net
    DocumentRoot /usr/local/apache/htdocs/panmail
    ServerName bq--3djszosuy56a.mltbd.net
    ServerAlias www.bq--3djszosuy56a.mltbd.net
Redirect /      http://panmail.net
</VirtualHost>
```

위와 같이 설정후 아파치를 재시작하면 팬메일.net 으로 접속시 http://panmail.net/ 으로 접속이 되는 것을 확인할 수 있을 것이다.

장애. 기타 장애 문제를 해결하는 방법

리눅스가 윈도우 계열과 다른 가장 큰 특징중 하나는 로그(log)를 철저히 남긴다는 것이다. 남겨진 로그 정보를 이용하여 각종 시스템의 장애나 상태를 점검할 수가 있는데, 아파치 웹 서버 역시 마찬가지이다. 문제나 장애가 발생시에는 반드시 error_log 를 남기게 하여 /usr/local/apache/logs/error_log 의 메시지를 살펴보면 문제의 원인과 해결책을 어렵지 않게 찾을 수 있게 될 것이다. 문제가 발생하였다고 당황하거나 무턱대고 관련 게시판에 질문을 올리지 말고, 에러 로그의 내용을 기초로 차근차근 문제의 원인을 분석하고 해결해 나간다면 조금씩 조금씩 서버 관리자에 가까워지는 자신을 발견할 수 있을 것이다.