졸업연구 보고서

침입탐지 및 관리 시스템

팀명 : M.O.S

팀장 : 최재현(10)
팀원 : 박은혜(12)
이상섭(10)
손민수(10)
황하림(10)

담당 교수: 유승재 교수님

2015. 6

중부대학교 정보보호학과

1. 서론
1.1 IDS란 ?……4
1.2 IDS의 구성요소 및 주요기능과 구조 ……4
1.3 IDS의 종류 및 탐지영역에 따른 분류……5
1.4 왜 IDS인가?……6
1.5 IDS는 어떻게 발전 할 것인가?……6

2. 본론……7
2.1 구축환경-IDS를 구축하기 위한 환경 ……7
2.1.1. 운영체제 ……7

2.2. 구축 설계……7

2.3. Snort를 이용한 IDS 구축……8
2.3.1 웹 서버 구축 ……16
2.3.2 MYSQL계정 및 DB추가……20
2.3.3 그누보드 설치……23

2.4. 쉘 프로그래밍 ……25
2.4.1. 쉘 스크립트 란?……25
2.4.2. 쉘 프로그래밍의 기본 구조 ……26
2.4.3. Crontab……26

2.5. Rsync ……27
2.5.1 Rsync의 기본구조 ……28
2.5.2 Rsync의 미러서버 동기화 ……28

2.6. 동기화에 따른 백업결과 ……31

3. 결론……31

4. PPT 자료……32

요 약

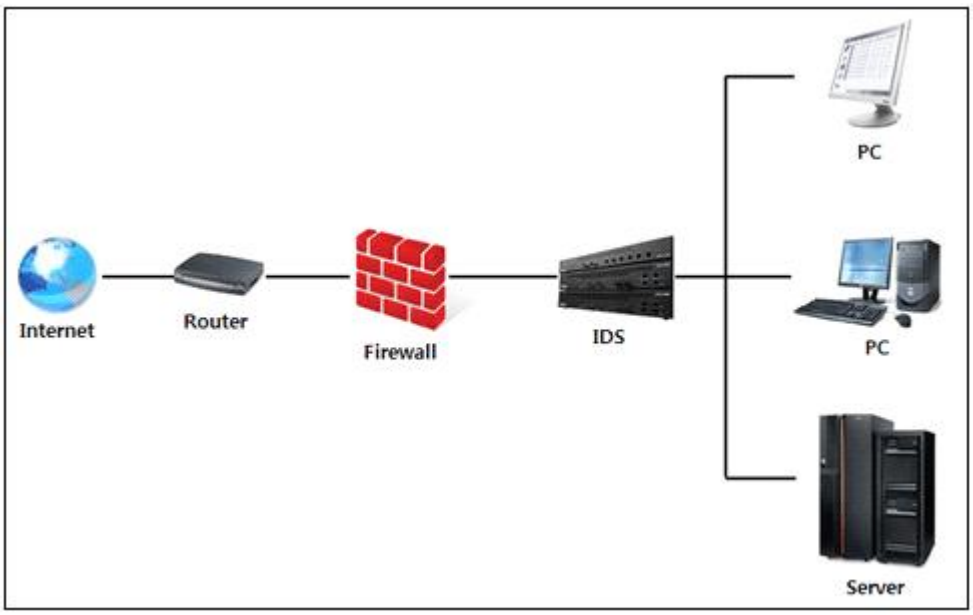
Internet이 대중화 되고 그로 인한 network시장이 급변하면서 네트워크 사용자를 관리하기 위해서 업체에서는 고객관리 데이터베이스 서버와 웹 서버를 따로 분리해서 관리한다. 하지만 사실 데이터베이스 서버를 관리하려면 DB의 내용 뿐 만 아니라 데이터베이스 시스템의 취약점까지 상세히 점검하여 외부로부터의 공격에 대비해야 한다.
일반적으로 데이터베이스와 관련한 모든 업무는 데이터베이스 관리자의 몫이며 전체의 보안 정책을 관리하는 시스템 관리자와는 별도인 경우가 많다.
데이터베이스 관리자 역시 데이터베이스의 관리가 중요하다고 인식하지만 사고를 경험하지 않은 경우 현재 시스템의 취약점을 발견하지 못하고 보안에 충분히 대비하지 못하는 것이 현실이다. 웹을 관리하고 보안을 강화하기 위해선 DB의 정보를 최우선으로 보호해야 하며 사용자들의 원활한 웹사이트 사용을 위해선 DB정보를 보호하면서도 웹사이트를 원활하게 사용 할 수 있어야 한다. 하지만 국내 시장은 DB에 대한 이해도가 그리 높지 않아 DB보안 또한 활성화 되지 못한 것이 사실이다 그렇기 때문에 최근 개인정보 유출 등의 사건사고가 끊이지 않고 있으며 여러 사용자들도 쉽게 접하고 보안상 웹 사이트에서 요구하는 주기적인 비밀번호 변경 등의 간단한 요구사항 조차도 이루어지지 않고 있다.
이러한 요구사항이 지켜지지 않았음에도 웹 사이트 에서는 회원들의 정보를 보관하고 관리해야하는 입장이기 때문에 웹 사이트 자체에서 별도로 DB를 관리하고 효율적으로 보안도 이루어져야 한다 웹 사이트 에서 원하는 요구사항을 지키지 않았더라도 관리자는 회원들의 정보유출을 막아야하고 만일에 사태에 대비 할 수 있는 능력을 가지고 있어야 한다. 실제로 데이터베이스(DB)가 정보기술 시장을 뜨겁게 달구고 있으며 빅 데이터가 실생활에 스며들기 시작했고 국산 데이터베이스관리시스템(DBMS)가 외산을 대체하는 사례가 늘고 있으며 지난해 국내 DB 시장은 전년 대비 7.5% 성장한 11조6517억원을 기록했다 업계는 DB시장이 계속 성장한 전망으로 향후 DB를 다루는 능력이 기업 성장을 결정지을 것으로 내다 보고 있다 한국DB진흥원에 따르면 국내 DB 시장은 최근 5년 동안 연평균 성장률 9%를 기록했고 지난 2009년 8조2514억원 이었던 DB산업은 지난해 11조6517억원으로 커졌다. 앞으로도 성장률 5.3%를 유지해 2018년 약15조원을 기록할 전망이다. 부문별로는 DB서비스와 DB구축이 가장 큰 비중을 차지할 것으로 보이며 2018년 DB서비스와 구축 시장이 각각 약6조원 규모로 성장할 것으로 예측된다. DB 산업 종사자도 꾸준히 늘어나는 추세다. 2013년 DB 산업 종사자는 전년 대비 1만2082명 늘어난 25만789명을 기록했다. DB진흥원 조사 결과 새해 DB 직무 종사자 신규 채용예정 인력은 총 4443명으로 나타났다. 이 중 DB 개발·구축 직무 수요가 1932명으로 가장 많으며, DB 운영·관리 직무(1253명), DB 관련 기획·컨설팅 직무(794명)가 뒤를 잇는 것으로 조사됐다.
이처럼 DB는 현재 가장 중요시 되고 있는 정보로서 이번에 저희 졸업 작품에는 IDS와 RSYNC CRONTAB을 이용한 미러 서버를 구축하여 웹 사이트를 원활하게 이용하면서 비정상적인 로그 발견 시 서버를 다운시키고 RSYNC를 이용한 서버 동기화로 원활한 웹 사이트 이용이 가능하면서도 DB정보는 백업된 미러 서버에 동기화 되어 안전하게 보안이 이루어지는 것을 보여줄 계획이다.

1. 서론

1.1. IDS란 ?

IDS (Intrusion Detection System) 침입탐지 시스템으로 탐지 대상 시스템이나 네트워크를 감시하여 비인가 되거나 비정상적인 행동을 탐지하여 구별하기 때문에 기본으로 탑제 된 방화벽이 내부망 보안을 수행하는데 있어 그 적용의 한계가 드러나 이를 보완 해줄 시스템이 필요하기 때문에 IDS 침입방지 시스템으로 잘 활용 될 수 있다.

1.2 IDS의 구성요소 및 주요기능과 구조



<그림 1> IDS구성요소

Snort의 기능

패킷 스니퍼 기능 : 네트워크의 패킷을 읽어 보여주는 기능을 한다.
패킷 로그 남기는 기능 : 모니터링한 패킷을 저장하고 로그를 기록하는 기능을 한다.
Network IPS기능 : 네트워크 트래픽을 분석하고 공격을 탐지한다.
이외에도 프로토콜 분석 콘텐츠 검색 등의 다양한 기능을 가지고 있다.

Snort의 구조

○Snort의 기본 구조는 sniffer,전처리기,탐지 엔진, 로깅/경고의 네가지 구성요소로 이루어져있다.

○sniffer는 이더넷 인터페이스를 Promiscuous모드로 동작하게 하여 Snort를 통과하는 모든 패킷을 수집하는 역할을 한다.

○ 전저 처리기는 sniffer에서 캡쳐한 네트워크 패킷을 탐지 엔진에서 비교하기 전에 사전 처리작업을 해주는 역활을 한다.

○탐지 엔진은 Snort의 핵심 구성요소로서 패킷과 룰을 비교하여 룰에 해당하는 패킷이 있을 경우 경고를 발생한다.

1.3 IDS의 종류 및 탐지영역에 따른 분류

구분	Firewall	IDS	IPS	UTM
주요역할	차단	Detection	특정 패턴 차단	다양한 기능 동시 포함
물리적 위치	외부와 내부 경계 네트워크	Host, Network	Host, Network	외부와 내부 경계 네트워크
설 계 정책	명백하게 허용하는 것만 허용	명백하게 금지하는 것만 금지	명백하게 금지하는 것만 금지	
장애시 상태	네트워크 사용 불능	네트워크 오픈 사용 가능	네트워크 오픈 사용 가능	네트워크 사용 불능
관 리 비용	관리자 역할 중요 (정책설정 등)	관리자 부담 없음	자동 차단	관리자 역할 중요
네트워크 부하	트래픽 병목 현상 발생	적음	적음	트래픽 병목 현상 발생
특징	정책 기반 대응	오탐, 미탐 문제 발생 가능, 지속 모니터링 필요	IDS의 사후 처리 문제로 대두 Zeroday attack 대응 가능	관리비용 절감, 장비 부하 가중

<표1> IDS의 종류 및 탐지영역에 따른 분류표

탐지영역에 따른 IDS 분류

H-IDS(Host based IDS)

-개별 호스트의 O/S가 제공하는 보안감사 로그, 시스템 로그, 사용자 계정 등의 정보를 이용해서 호스트에 대한 공격을 탐지

-각 호스트에 상주하는 Agent 와 이들을 관리하는 Agent Manager로 구성
중요한 시스템 파일이나 실행코드에 대한 무결성 검사 기능이나 시스템의 취약점들을 탐지해주는 취약성 스캐너(Vulerability Scanner) 등과 결합되어 사용

-특정 시스템과의 O/S와 밀접히 결합되어 각종 행위를 분석하므로 정교한 모니터링과 로깅이 가능

-IDS 문제 발생 시 해당 호스트에 영향을 미치며 IDS로 인해 시스템에 부하를 크게 한다.

N-IDS(Network based IDS)

-네트워크 기반의 공격을 탐지하여 네트워크 기반 구조를 보호하는 것을 목적으로 함

-호스트 기반의 IDS처럼 호스트에 대한 공격을 탐지하거나 상세한 기록을 남길 수는 없으며, 네트워크가 분할되어 있는 경우 제 기능을 발휘하지 못하거나 적용 범위가 제한되어 실용성이 없는 경우도 있음

-NIC를 통해 패킷을 수집하여 수동적인 분석을 하기 때문에 기존 네트워크에 영향을 주지 않고 설치가 편리하다.

1.4 왜 IDS인가?

Internet 의 이용이 보편화되고 홈 네트워킹 모바일 인터넷 전화 등의 새로운 네트워크 인프라와 기술이 지속적으로 발전하고 있는 가운데 국내외 정보보호 침해사고가 점점 진화하고 있다 과거에는 단일 시스템을 대상으로 하는 단순한 웜바이러스 등으로 공격 하였지만 점점 지능화 되어 스파이웨어처럼 네트워크 서비스 전체의 사용권을 침해하는 공격으로 진화되었고 이에 따라 IDS 등의 보안시스템이 도입되어 보안강화에 노력하는 모습을 보여 왔다.

우리가 하려는 rsync를 통한 서버 동기화 또한 ssh를 클라이언트로 리눅스방화벽(iptaples)에 룰을 적용하여 ssh 접근시 기본적인 보안이 가능하지만 침입 여부를 실시간으로 탐지 할 수 없고 log 기반의 매칭 등이 불가능하기 때문에 침입 방지 시스템인 IDS를 기반으로 구축 하게 되었으며 변형된 패턴 등에 대한 탐지가 어려운 IDS에 취약점을 주기적인 백업과 미러 서버 동기화로 좀 더 높은 수준의 정보보호를 가능하게 해주기 때문에 IDS를 사용하게 되었고 IDS 적용 범위 또한 일반 방화벽의 경우 룰은 TCP/IP 2개 계층에 적용되는데 반해 IDS의 룰은 3개의 계층에 적용 된다 그렇기 때문에 IDS는 IP주소와 Port번호는 물론 TCP/IP 응용계층의 데이터까지 패턴 매칭 기법으로 검사가 가능하다.

또한 해킹사고 발생 시 어느 정도의 근원지를 추적 할 수 있고 내부 사용자의 오/남용을 탐지 및 방어가 가능하며 관리비용 또한 부담이 없다.

하지만 IDS의 자동대응 기능은 TCP를 이용한 해킹 공격으로 제한되며 UDP나 ICPM 등의 프로토콜에 대한 자동대응 기능이 적용되지 않고 IDS가 일반화 되면서 탐지 기능 등을 우회 할 수 있는 공격기법등이 다량으로 발표 되고 IDS 특성상 지속적인 모니터링이 필요하여 오탐미탐 등의 문제가 발생 할 가능성이 높다는 단점이 있다.

1.5 IDS는 어떻게 발전 할 것인가?

IDS는 일반적으로 네트워크 기반 침입탐지시스템은 실시간으로 네트워크를 모니터링하고 패킷을 분석하기 위한 처리 능력과 저장의 한계를 극복해야하고 네트워크 트래픽 증가에 따른 별도의 하드웨어 및 소프트웨어가 필요할 것이다.

현재 많은 연구들이 비정상적인 탐지기술을 중심으로 이루어지고 있으나 아직 많은 솔루션들은 단순한 오-남용 등의 비정상 행위들에 대한 탐지를 주로 사용하고 있음에 개별적인 사용에 대한 탐지보다는 일반적인 사용시간, 네트워크 접속 수 , 인증 실패 회수 등의 탐지를 위한 척도로 사용되는 것으로 제한하고 있다.

침입탐지시스템이 활성화되기 위해서는 거짓탐지율의 제거탐지실패율의 제거 보안관련 이벤트는 정의와 보고방법 침입탐지시스템의 테스트 방법의 고안 탐지된 공격의 피해정도의 결정과 피해의 최소화 방법 제공 네트워크에서 요구하는 크기에 변화할 수 있는 시스템 등과 같은 기술적인 논점들이 해결되어야 한다.

2. 본론

2.1 구축환경-IDS를 구축하기 위한 환경

IDS를 구축하기 위하여 우리는 CentOS6.5 버전을 사용하였다 리눅스의 배포판 중 하나인 CentOs는 현재까지 7.1버전까지 나와 있고 페도라가 선구적인 기술을 도입하다 못해 RHEL을 제대로 반영하지 못하게 되자 RHEL를 완벽에 가까운 반영을 목적으로 만든 배포판 이다. CentOS는 리눅스로 RHEL의 소스를 기반으로 만들어지며 철저하게 최신 버전의 RHEL을 미러링 하는데 중점을 둔다. 단 상표권은 회사가 가져가는 GPL의 특성상 레드햇의 트레이트 마크와 로고를 그대로 쓸 경우 상표권 분쟁이 있을 수 있기 때문에 레드햇이 소유하고 있는 레드햇 트레이드마크와 로고는 제거 그리고 그 자리에 CentOS 고유의 로고를 대신 넣으면 완성된다 이 때문에 버전도 RHEL과 똑같이 나간다. 덤으로 CentOS에서 말하는 북미 엔터프라이즈 소프트웨어 벤더는 레드햇을 지칭한다.

CentOS의 장점은 서버 리눅스 시장의 1인자인 레드햇 리눅스를 무료로 체험할 수 있고 실제로 많은 서버등이 CentOS로 운영되고 있고 대기업인 NHN 이나 daum에서도 사용 되고 있다는 것이다.

한국의 수 많은 웹 호스팅 업체도 이걸로 리눅스 서버를 운영하는 경우가 많고 충분한 자체 유지보수 인력이 있고 책임을 자신들이 진다면 이만한 서버 운영체제가 없다.

그렇기 때문에 대형 서점에 가보면 CentOS를 활용한 실무나 서버 관련 책들이 많이 있다.

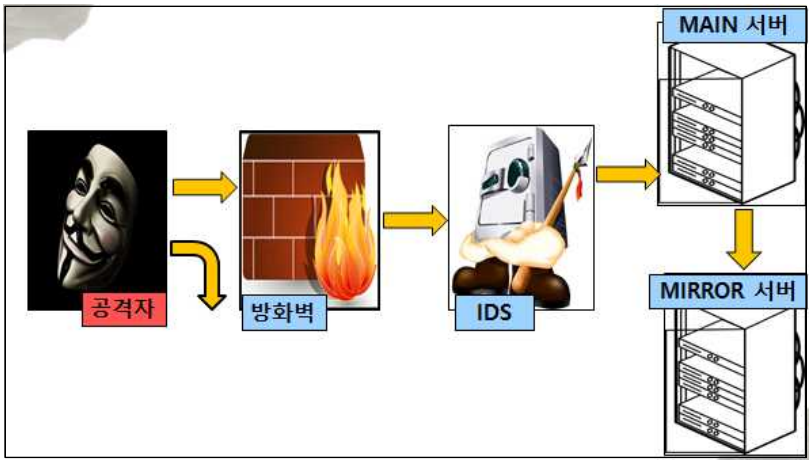
개인의 리눅스 서버 공부나 중소규모 프로젝트에는 훌륭한 배포판 이지만 안정성과 실용성이 중요하다면 RHEL이 더 나을 것 이다 RHEL과 CentOS는 서로 패키지가 호환되기 때문에 리눅스를 처음 배우는 경우에는 RHEL에 앞서 CentOS를 먼저 공부하는 것이 바람직하고 RHEL과 크게 다르지 않으면서 상위판과 100%의 가까운 호환성을 유지한다 주로 Yum을 통한 소프트웨어 업데이트를 하며 up2date도 지원 한다 또한 레드햇 엔터프라이즈 리눅스와 완벽하게 호환되면서 무료로 사용할 수 있기 때문에 우리가 안정적인 서버 구축과 공부를 할 수 있는 최적의 운영체제라 생각 되어 CentOS를 기본 OS로 서버구축을 진행하게 되었다.

2.2. 구축 설계

우리가 이번 졸업 작품을 통해 보여주고 싶은 것은 IDS 침입방지와 만약 침입이 뚫렸을 경우 서버를 Shutdown 시키고 미러 서버로 미리 동기화된 정보를 가지고 문제없이 웹 사이트를 이용 할 수 있게 하는 것이 목적으로 DB에 저장된 회원정보 보호를 최우선으로 하였다.

공격자가 방화벽을 뚫고 IDS에 침입하면 로그기록을 실시간으로 체크하여 이상 로그가 발견되면 Main서버를 Shutdown 시키고 Crontab으로 백업된 DB의 회원정보를 Rsync로 미러서버에 동기화 하여 공격자의 침입과 동시에 최대한 DB정보를 보호하기 위해 노력하였다.

침입 유무에 따라 위에서 설명한 것들이 실행되게 하였으나 웹 서버 사용에는 지장이 없도록 하였고 Rsync를 통해 주기적으로 백업과 동기화를 하여 관리자가 일일이 DB에 회원정보 등을 미러 서버나 Main서버를 대체하는 서버로 동기화 시켜줘야 하는 번거로운 과정을 없애고 IDS 특성상 실시간으로 모니터링 하며 이상 패킷과 로그 등을 직접 잡아내기에 오탐과 미탐의 문제를 쉘 스크립트를 이용한 프로그램과 Corntab을 이용한 주기적인 백업으로 정보유출을 방지 하였다.



<그림 2> IDS 구축설계도

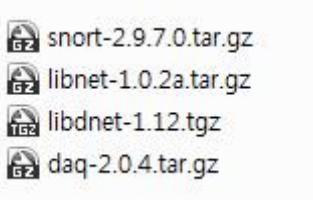
2.3 Snort를 이용한 IDS 구축

```
[root@localhost Snort]# ll
total 7764
-rw-r--r--. 1 root root  495316 May 31 03:44 daq-2.0.4.tar.gz
-rw-r--r--. 1 root root  970125 May 31 03:44 libdnet-1.12.tgz
-rw-r--r--. 1 root root  140191 May 31 03:44 libnet-1.0.2a.tar.gz
-rw-r--r--. 1 root root 6340553 May 31 03:44 snort-2.9.7.0.tar.gz
```

gcc version (4.4.6 including libraries),

flex(2.5.35),
bison(2.4.1),
zlib(1.2.3includingzlib-devel),
libpcap(1.0.0includinglibpcap-devel),
pcre(7.84includingpcre-devel),
libdnet(1.11or1.12includinglibdnet-devel)

설치해야할 선패키지(CentOS)



준비 tar 압축파일

```
[root@localhost Snort]# tar xvzf libnet-1.0.2a.tar.gz
Libnet-1.0.2a/
Libnet-1.0.2a/Makefile.in
Libnet-1.0.2a/README
Libnet-1.0.2a/VERSION
Libnet-1.0.2a/acconfig.h
Libnet-1.0.2a/acinclude.m4
Libnet-1.0.2a/aclocal.m4
Libnet-1.0.2a/config.guess
Libnet-1.0.2a/config.sub
Libnet-1.0.2a/configure
Libnet-1.0.2a/configure.in
Libnet-1.0.2a/ensure-dir.sh
Libnet-1.0.2a/install-sh
Libnet-1.0.2a/libnet-config.in
Libnet-1.0.2a/doc/
Libnet-1.0.2a/doc/CHANGELOG
Libnet-1.0.2a/doc/CHANGELOG-NEWFUNCTIONS
Libnet-1.0.2a/doc/COPYING
Libnet-1.0.2a/doc/README
Libnet-1.0.2a/doc/TODO-1.0
Libnet-1.0.2a/doc/TODO-1.1
```

Snort 의 압축을 풀어준다

```
[root@localhost Snort]# cd Libnet-1.0.2a
[root@localhost Libnet-1.0.2a]# ll
total 244
-rw-------. 1 sangsup sangsup   660 Dec 12  2000 acconfig.h
-rw-------. 1 sangsup sangsup  8895 Dec 12  2000 acinclude.m4
-rw-------. 1 sangsup sangsup  9463 Jan  7  2001 aclocal.m4
-rwx------. 1 sangsup sangsup 20370 May 24  2000 config.guess
-rwx------. 1 sangsup sangsup 19235 Feb  6  2001 config.sub
-rw-------. 1 root    root    19234 Feb  6  2001 config.sub-new
-rwx--x--x. 1 sangsup sangsup 75116 Jan 17  2001 configure
-rw-------. 1 sangsup sangsup  7047 Jan 17  2001 configure.in
drwx------. 3 sangsup sangsup  4096 Dec 17  2000 doc
-rwx------. 1 sangsup sangsup   522 May 24  2000 ensure-dir.sh
drwx------. 2 sangsup sangsup  4096 Feb  6  2001 example
drwx------. 3 sangsup sangsup  4096 Feb  6  2001 include
-rwx------. 1 sangsup sangsup  5585 Feb  5  2001 install-sh
drwx------. 2 sangsup sangsup  4096 Feb  6  2001 lib
-rw-------. 1 sangsup sangsup  1068 Dec 12  2000 libnet-config.in
-rw-------. 1 sangsup sangsup  4198 Feb  5  2001 Makefile.in
drwx------. 2 sangsup sangsup  4096 Dec 12  2000 misc
drwx------. 4 sangsup sangsup  4096 Dec 12  2000 ports
-rw-------. 1 sangsup sangsup   513 Dec 12  2000 README
```

압축이 풀린 것을 확인.

```
[root@localhost Libnet-1.0.2a]# ./cofigure
-bash: ./cofigure: No such file or directory
[root@localhost Libnet-1.0.2a]# ./configure
creating cache ./config.cache
Beginning autoconfiguration process for libnet-1.0.2a...
checking host system type... i686-pc-linux-gnu
checking target system type... i686-pc-linux-gnu
checking build system type... i686-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler (gcc  ) works... yes
checking whether the C compiler (gcc  ) is a cross-compiler... no
checking whether we are using GNU C... yes
checking whether gcc accepts -g... yes
checking for working const... yes
checking for a BSD compatible install... /usr/bin/install -c
checking whether make sets ${MAKE}... yes
checking for ranlib... ranlib
checking for ar... ar
checking for ln... ln
checking for strerror... yes
checking for pcap_open_live in -lpcap... yes
```

```
[root@localhost Libnet-1.0.2a]# make && make install
sed -e 's/.*/#define VERSION "&"/' ./VERSION > version.h
gcc -O2 -funroll-loops -fomit-frame-pointer -Wall   -DHAVE_CONFIG_H -c src/libne
t_resolve.c -o src/libnet_resolve.o
In file included from src/libnet_resolve.c:36:
src/../include/libnet.h:87:8: warning: missing terminating " character
src/../include/libnet.h:89:50: warning: missing terminating " character
src/libnet_resolve.c: In function ‘libnet_host_lookup’:
src/libnet_resolve.c:67: warning: pointer targets in passing argument 1 of ‘spri
ntf’ differ in signedness
/usr/include/stdio.h:363: note: expected ‘char * __restrict__’ but argument is o
f type ‘u_char *’
src/libnet_resolve.c:71: warning: pointer targets in passing argument 1 of ‘__bu
iltin_strncpy’ differ in signedness
src/libnet_resolve.c:71: note: expected ‘char *’ but argument is of type ‘u_char
 *’
src/libnet_resolve.c: In function ‘libnet_host_lookup_r’:
src/libnet_resolve.c:95: warning: pointer targets in passing argument 1 of ‘spri
ntf’ differ in signedness
/usr/include/stdio.h:363: note: expected ‘char * __restrict__’ but argument is o
f type ‘u_char *’
src/libnet_resolve.c:100: warning: pointer targets in passing argument 1 of ‘__b
uiltin_strncpy’ differ in signedness
```

make 로 libnet 패키지 파일을 설치해준다.

```
./install-sh include/libnet/libnet-ospf.h /usr/include/libnet
./install-sh doc/libnet.3
install:        no destination specified
make: *** [install] 오류 1
[root@localhost Libnet-1.0.2a]#
```

오류가 발생하면

```
[root@localhost Libnet-1.0.2a]# vi ./Makefile
   17 BIN_PREFIX  =   ${exec_prefix}/bin/
   18 INC_PREFIX  =   ${prefix}/include/
   19 LIB_PREFIX  =   ${exec_prefix}/lib/
   20 MAN_PREFIX  =   /usr/share/doc/
   21
```

vi 로 ./Makefile 을 열어 20 행쯤 MAN-PREFIX 에 /usr/share/doc/ 추가한 뒤

```
[root@localhost Libnet-1.0.2a]# make install
```

install 만 다시 해주면 된다.

```
[root@localhost install]# tar xvzf libdnet-1.12.tgz
[root@localhost install]# cd ./libdnet-1.12
[root@localhost libdnet-1.12]# ./configure
[root@localhost libdnet-1.12]# make && make install
```

같은 방법으로 libdnet 압축을 풀고 make 로 패키지를 설치해준다.

```
[root@localhost install]# tar xvzf snort-2.9.7.0.tar.gz
[root@localhost install]# cd ./snort-2.9.7.0
[root@localhost snort-2.9.7.0]# ./configure
```

snort make 파일 생성한다.

```
ERROR!  Libpcre header not found.
Get it from http://www.pcre.org
```

다음과 같은 에러 발생시 yum으로 pcre패키지를 설치 해준다.

```
[root@localhost snort-2.9.7.0]# yum install -y pcre-devel
```

pcre-devel설치

```
ERROR! Libpcap library/headers (libpcap.a (or .so)/pcap.h)
```

다음과 같은 에러 발생시 libpcap 설치해줘야 한다.

```
[root@localhost install]# yum install -y libpcap-devel
```

yum사용 하여 libpcap-devel 설치한다.

```
ERROR!  daq_static library not found, go get it from
http://www.snort.org/.
```

다음과 같은 에러 발생시 daq 패키지 파일을 설치 해줘야 한다.

```
[root@localhost install]# tar xvzf daq-2.0.4.tar.gz
[root@localhost install]# cd ./daq-2.0.4
[root@localhost daq-2.0.4]# ./configure
```

daq 압축 풀고 make 한다.

```
configure: error: Your operating system's lex is insufficient to compile
        libsfbpf.  You should install both bison and flex.
        flex is a lex replacement that has many advantages,
        including being able to compile libsfbpf.  For more
        information, see http://www.gnu.org/software/flex/flex.html .
```

다음과 같은 에러 발생시 flex와 bison 설치 해준다.

```
ERROR!  zlib header not found, go get it from
http://www.zlib.net
```

다음과 같은 에러 발생시 zlib-devel을 설치해주면 된다.

```
[root@localhost ~]# snort -V

  ,,_     -*> Snort! <*-
 o"  )~   Version 2.9.7.0 GRE (Build 149)
  ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
          Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
          Copyright (C) 1998-2013 Sourcefire, Inc., et al.
          Using libpcap version 1.7.3
          Using PCRE version: 7.8 2008-09-05
          Using ZLIB version: 1.2.3
```

make make && make install 이 완료 된 후 snort의 정보 및 구동확인 할 수 있다.

https://www.snort.org 에서 메일 인증을 하고 다운 받을 수 있다.

```
[root@localhost /]# mkdir /var/log/snort
[root@localhost /]# mkdir /etc/snort
```

mkdir /etc/snort -> 환경설정을 할 폴더

mkdir /var/log/snort -> log 파일이 저장될 폴더를 생성해준다.

```
[root@localhost snort]# tar xvzf /install/snortrules-snapshot-2956.tar.gz
-C /etc/snort
```

tar xvzf /다운로드 폴더/snortrules-snapshot-2907.tar.gz -C /etc/snort -> 룰 셋 파일을 스노트에 설정 해준다.

```
[root@localhost snort]# ls
etc  preproc_rules  rules  so_rules
[root@localhost snort]# ls -l ./etc/
합계 4480
-rw-r--r--. 1 1210 1210    3854 2014-12-12 01:38 classification.config
-rw-r--r--. 1 1210 1210     746 2014-12-12 01:38 reference.config
-rw-r--r--. 1 1210 1210 4483836 2014-12-12 01:41 sid-msg.map
-rw-r--r--. 1 1210 1210   29395 2014-12-12 01:38 snort.conf
-rw-r--r--. 1 1210 1210    2556 2014-12-12 01:38 threshold.conf
-rw-r--r--. 1 1210 1210   53841 2014-12-12 01:38 unicode.map
[root@localhost snort]# cp etc/* /etc/snort
```

/etc/snort/etc 안의 설정파일을 /etc/snort 로 복사한다.

[root@localhost snort]# groupadd snort -> snort 그룹 생성
[root@localhostsnort]#useradd-gsnortsnort -> snort 그룹에 snort 계정 생성
[root@localhostsnort]#chownsnort:snort/var/log/snort -> 폴더의 소유권자와 소유그룹을 snort 로 변경
[root@localhostsnort]#touch/var/log/snort/alert -> alert 빈 파일 생성
[root@localhostsnort]#chownsnort:snort/var/log/snort/alert -> 파일의 소유권자와 소유그룹을 snort 로 변경
[root@localhostsnort]#chmod600/var/log/snort/alert -> 파일의 권한변경
[root@localhost snort]# mkdir /usr/local/lib/snort_dynamicrules
[root@localhostsnort]#cp/etc/snort/so_rules/precompiled/Centos-5-4/i386/2.9.5.6/*.so/usr/local/lib/snort_dynamicrules
-> snort 에 사용되는 라이브러리를 복사
[root@localhost snort]# cat /etc/snort/so_rules/*.rules >> /etc/snort/rules/so-rules.rules
->so_rules 의 rules 파일을 스노트가 사용할 경로 /etc/snort/rules/so-rules.rules 파일에 덮어 쓴다.

```
[root@localhost snort]# vi /etc/snort/snort.conf
```

환경설정 파일을 vi 편집기로 열어본다.

```
104 var RULE_PATH ../rules
105 var SO_RULE_PATH ../so_rules
106 var PREPROC_RULE_PATH ../preproc_rules
107
108 # If you are using reputation preprocessor set these
109 var WHITE_LIST_PATH ../rules
110 var BLACK_LIST_PATH ../rules
```

경로가 맞는지 확인한다.

```
[root@localhost rules]# pwd
/etc/snort/rules
[root@localhost rules]# ls
app-detect.rules              local.rules              protocol-voip.rules
attack-responses.rules        malware-backdoor.rules   pua-adware.rules
backdoor.rules                malware-cnc.rules        pua-other.rules
bad-traffic.rules             malware-other.rules      pua-p2p.rules
blacklist.rules               malware-tools.rules      pua-toolbars.rules
botnet-cnc.rules              misc.rules               rpc.rules
browser-chrome.rules          multimedia.rules         rservices.rules
browser-firefox.rules         mysql.rules              scada.rules
browser-ie.rules              netbios.rules            scan.rules
browser-other.rules           nntp.rules               server-apache.rules
browser-plugins.rules         oracle.rules             server-iis.rules
browser-webkit.rules          os-linux.rules           server-mail.rules
chat.rules                    os-mobile.rules          server-mssql.rules
content-replace.rules         os-other.rules           server-mysql.rules
ddos.rules                    os-solaris.rules         server-oracle.rules
deleted.rules                 os-windows.rules         server-other.rules
dns.rules                     other-ids.rules          server-samba.rules
dos.rules                     p2p.rules                server-webapp.rules
experimental.rules            phishing-spam.rules      shellcode.rules
exploit-kit.rules             policy-multimedia.rules  smtp.rules
```

/etc/snort/rules -> snort 룰 설정 경로를 확인한다.

```
#------------
# LOCAL RULES
#------------

alert icmp any any -> any any (msg:"ICMP ping Test"; sid:1000001;)
```

vi 로 local.rules 를 열어 룰을 설정한다.

```
[root@localhost rules]# snort -c /etc/snort/rules/local.rules
Running in IDS mode

        --== Initializing Snort ==--
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file "/etc/snort/rules/local.rules"
Tagged Packet Limit: 256
Log directory = /var/log/snort

+++++++++++++++++++++++++++++++++++++++++++++++++++
Initializing rule chains...
1 Snort rules read
    1 detection rules
    0 decoder rules
    0 preprocessor rules
1 Option Chains linked into 1 Chain Headers
0 Dynamic rules
+++++++++++++++++++++++++++++++++++++++++++++++++++

+-------------------[Rule Port Counts]-------------------------
|              tcp     udp    icmp      ip
|     src        0       0       0       0
|     dst        0       0       0       0
|     any        0       0       1       0
|      nc        0       0       1       0
|     s+d        0       0       0       0
```

```
       --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.7.0 GRE (Build 149)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.7.3
           Using PCRE version: 7.8 2008-09-05
           Using ZLIB version: 1.2.3

Commencing packet processing (pid=3430)
```

snort -c 옵션으로 /etc/snort/rules/local.rules 에 설정한 룰 구동시킨다.

```
C:₩Users₩609>ping -l 65000 192.168.176.132 -t

Ping 192.168.176.132 65000바이트 데이터 사용:
192.168.176.132의 응답: 바이트=65000 시간=6ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=6ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=5ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=3ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=3ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=3ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=3ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=4ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=5ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=6ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=4ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=5ms TTL=64
```

공격핑을 보내본다.

```
[root@localhost snort]# pwd
/var/log/snort
[root@localhost snort]# ll
total 52
-rw-------. 1 snort snort 29693 May 28 00:20 alert
-rw-------. 1 root  root   6724 May 13 10:34 snort.log.1431538458
-rw-------. 1 root  root      0 May 14 22:34 snort.log.1431668050
-rw-------. 1 root  root      0 May 14 22:52 snort.log.1431669179
-rw-------. 1 root  root      0 May 14 23:17 snort.log.1431670650
-rw-------. 1 root  root      0 May 14 23:22 snort.log.1431670955
-rw-------. 1 root  root    906 May 14 23:32 snort.log.1431671431
-rw-------. 1 root  root    906 May 27 23:50 snort.log.1432795277
-rw-------. 1 root  root    318 May 28 00:20 snort.log.1432797609
```

snort 에 적용한 icmp 탐지 룰이 공격 icmp 패킷을 잡아 탐지 로그가 생성된다.

2.3.1 웹 서버 구축

```
[root@livecd ~]# yum install httpd mysql-server mysql php php-devel php-pear php
-mysql php-mbstring php-gd
```

yum 으로 Apache + PHP + MySQL 패키지를 설치합니다.

```
Installed:
  httpd.x86_64 0:2.2.15-39.el6.centos      mysql.x86_64 0:5.1.73-3.el6_5
  mysql-server.x86_64 0:5.1.73-3.el6_5     php.x86_64 0:5.3.3-40.el6_6
  php-devel.x86_64 0:5.3.3-40.el6_6        php-gd.x86_64 0:5.3.3-40.el6_6
  php-mbstring.x86_64 0:5.3.3-40.el6_6     php-mysql.x86_64 0:5.3.3-40.el6_6
  php-pear.noarch 1:1.9.4-4.el6

Dependency Installed:
  apr.x86_64 0:1.3.9-5.el6_2
  apr-util.x86_64 0:1.3.9-3.el6_0.1
  apr-util-ldap.x86_64 0:1.3.9-3.el6_0.1
  autoconf.noarch 0:2.63-5.1.el6
  automake.noarch 0:1.11.1-4.el6
  httpd-tools.x86_64 0:2.2.15-39.el6.centos
  libXpm.x86_64 0:3.5.10-2.el6
  mailcap.noarch 0:2.1.31-2.el6
  perl-DBD-MySQL.x86_64 0:4.013-3.el6
  perl-DBI.x86_64 0:1.609-4.el6
  php-cli.x86_64 0:5.3.3-40.el6_6
  php-common.x86_64 0:5.3.3-40.el6_6
  php-pdo.x86_64 0:5.3.3-40.el6_6

Dependency Updated:
  mysql-libs.x86_64 0:5.1.73-3.el6_5

Complete!
```

Apache + PHP + MySQL 패키지를 설치완료

```
[root@livecd ~]# cp -av /etc/httpd/conf/httpd.conf /etc/httpd/conf/httpd.conf.or
i
`/etc/httpd/conf/httpd.conf' -> `/etc/httpd/conf/httpd.conf.ori'
```

httpd 설정파일을 백업한다.

```
[root@livecd ~]# sed -i 's/KeepAlive off/KeepAlive on/' /etc/httpd/conf/httpd.conf
```

KeepAlive 항목을 on으로 변경해준다 .

```
[root@livecd ~]# sed -i 's/#ServerName www.example.com:80/ServerName 127.0.0.1:80/' /etc/httpd/c
onf/httpd.conf
```

아파치 실행시 경고문구가 뜨지 않도록 서버 아이피를 127.0.0.1 설정해 준다.

```
[root@livecd ~]# sed -i 's/AddDefaultCharset UTF-8/#AddDefaultCharset UTF-8/' /etc/ht
tpd/conf/httpd.conf
```

AddDefaultCharset 항목은 UTF-8이 아닌 페이지에서 문제가 될 수 있으므로 주석 처리합니다.

```
[root@livecd ~]# sed -i 's/short_open_tag = Off/short_open_tag = On/' /etc/php.ini
[root@livecd ~]# sed -i 's/;date.timezone =/date.timezone = "Asia\/Seoul"/' /etc/php.ini
[root@livecd ~]# sed -i 's/allow_call_time_pass_reference = Off/allow_call_time_pass_reference =
 On/' /etc/php.ini
[root@livecd ~]# grep -P 'short_open_tag =|expose_php =|date.timezone =|register_globals =|regis
ter_long_arrays =|magic_quotes_gpc =|allow_call_time_pass_reference =|error_reporting =|display_
errors =|display_startup_errors =' /etc/php.ini
short_open_tag = On
allow_call_time_pass_reference = On
expose_php = On
error_reporting = E_ALL & ~E_DEPRECATED
display_errors = Off
display_startup_errors = Off
register_globals = Off
register_long_arrays = Off
magic_quotes_gpc = Off
date.timezone = "Asia/Seoul"
```

일반적인 설정을 적용해준다.

```
short_open_tag = On
allow_call_time_pass_reference = On
expose_php = Off
error_reporting = E_ALL & ~E_NOTICE & ~E_DEPRECATED & ~E_USER_DEPRECATED
display_errors = On
register_globals = On
register_long_arrays = Off
magic_quotes_gpc = On
date.timezone = "Asia/Seoul"
```

수정된 결과가 잘 적용 되었는지 확인해준다 .

```
[root@livecd ~]# /etc/init.d/httpd start
Starting httpd:                                            [  OK  ]
[root@livecd ~]# /sbin/chkconfig httpd on
```

웹서버를 재시작하고 웹서버 부팅시 자동으로 켜지도록 설정해준다.

```
[root@livecd ~]# /etc/init.d/mysqld start
Initializing MySQL database:  WARNING: The host 'livecd.centos' could not be looked up with reso
lveip.
This probably means that your libc libraries are not 100 % compatible
with this binary MySQL version. The MySQL daemon, mysqld, should work
normally with the exception that host name resolving will not work.
This means that you should use IP addresses instead of hostnames
when specifying MySQL privileges !
Installing MySQL system tables...
OK
Filling help tables...
OK

To start mysqld at boot time you have to copy
support-files/mysql.server to the right place for your system

PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !
To do so, start the server, then issue the following commands:

/usr/bin/mysqladmin -u root password 'new-password'
/usr/bin/mysqladmin -u root -h livecd.centos password 'new-password'

Alternatively you can run:
/usr/bin/mysql_secure_installation

which will also give you the option of removing the test
databases and anonymous user created by default.  This is
```

mysqld를 start 시킨다.

```
---> Package libmcrypt.x86_64 0:2.5.8-9.el6 will be installed
---> Package php-bcmath.x86_64 0:5.3.3-40.el6_6 will be installed
---> Package php-tidy.x86_64 0:5.3.3-40.el6_6 will be installed
--> Processing Dependency: libtidy-0.99.so.0()(64bit) for package: php-tidy-5.3.3-40.el6_6.x86_64
--> Running transaction check
---> Package libtidy.x86_64 0:0.99.0-19.20070615.1.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                    Arch          Version                    Repository      Size
================================================================================
Installing:
 phpMyAdmin                 noarch        4.0.10.9-1.el6             epel           4.1 M
Installing for dependencies:
 libmcrypt                  x86_64        2.5.8-9.el6                epel            96 k
 libtidy                    x86_64        0.99.0-19.20070615.1.el6   base           127 k
 php-bcmath                 x86_64        5.3.3-40.el6_6             updates         37 k
 php-mcrypt                 x86_64        5.3.3-3.el6                epel            19 k
 php-php-gettext            noarch        1.0.11-3.el6               epel            21 k
 php-tcpdf                  noarch        6.2.4-1.el6                epel           2.8 M
 php-tcpdf-dejavu-sans-fonts noarch       6.2.4-1.el6                epel           304 k
 php-tidy                   x86_64        5.3.3-40.el6_6             updates         39 k
 php-xml                    x86_64        5.3.3-40.el6_6             updates        106 k

Transaction Summary
================================================================================
```

mysql db관리 툴을 설치한다

```
Install      10 Package(s)

Total download size: 7.6 M
Installed size: 33 M
Is this ok [y/N]: y
Downloading Packages:
(1/10): libmcrypt-2.5.8-9.el6.x86_64.rpm                   |  96 kB     00:00
(2/10): libtidy-0.99.0-19.20070615.1.el6.x86_64.rpm        | 127 kB     00:00
(3/10): php-bcmath-5.3.3-40.el6_6.x86_64.rpm               |  37 kB     00:00
(4/10): php-mcrypt-5.3.3-3.el6.x86_64.rpm                  |  19 kB     00:00
(5/10): php-php-gettext-1.0.11-3.el6.noarch.rpm            |  21 kB     00:00
(6/10): php-tcpdf-6.2.4-1.el6.noarch.rpm                   | 2.8 MB     00:00
(7/10): php-tcpdf-dejavu-sans-fonts-6.2.4-1.el6.noarch.rpm | 304 kB     00:00
(8/10): php-tidy-5.3.3-40.el6_6.x86_64.rpm                 |  39 kB     00:00
(9/10): php-xml-5.3.3-40.el6_6.x86_64.rpm                  | 106 kB     00:00
(10/10): phpMyAdmin-4.0.10.9-1.el6.noarch.rpm              | 4.1 MB     00:00
--------------------------------------------------------------------------------
Total                                              3.0 MB/s | 7.6 MB     00:02
warning: rpmts_HdrFromFdno: Header V3 RSA/SHA256 Signature, key ID 0608b895: NOKEY
Retrieving key from file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-6
Importing GPG key 0x0608B895:
 Userid : EPEL (6) <epel@fedoraproject.org>
 Package: epel-release-6-8.noarch (@extras)
 From   : /etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-6
Is this ok [y/N]: y
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
```

y를 누르면 설치가 완료된다.

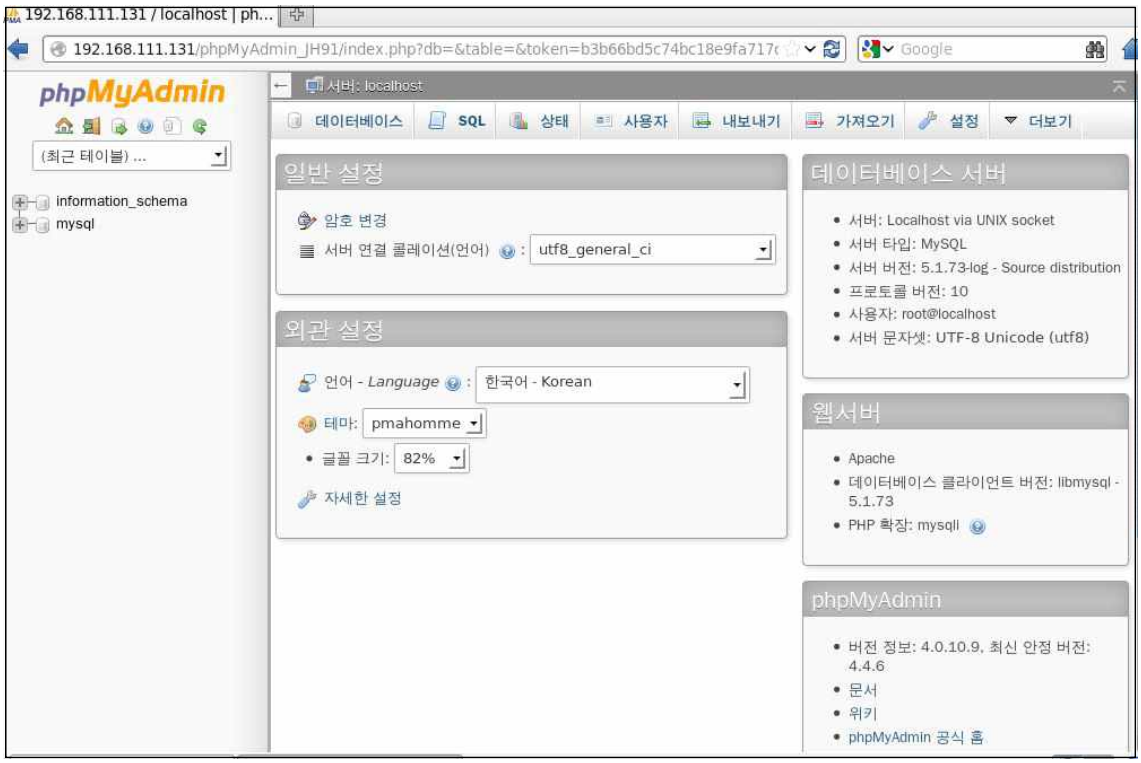
```
Dependency Installed:
  libmcrypt.x86_64 0:2.5.8-9.el6                    libtidy.x86_64 0:0.99.0-19.20070615.1.el6
  php-bcmath.x86_64 0:5.3.3-40.el6_6                php-mcrypt.x86_64 0:5.3.3-3.el6
  php-php-gettext.noarch 0:1.0.11-3.el6             php-tcpdf.noarch 0:6.2.4-1.el6
  php-tcpdf-dejavu-sans-fonts.noarch 0:6.2.4-1.el6  php-tidy.x86_64 0:5.3.3-40.el6_6
  php-xml.x86_64 0:5.3.3-40.el6_6

Complete!
```

설치가 모두 완료된 결과창이다.

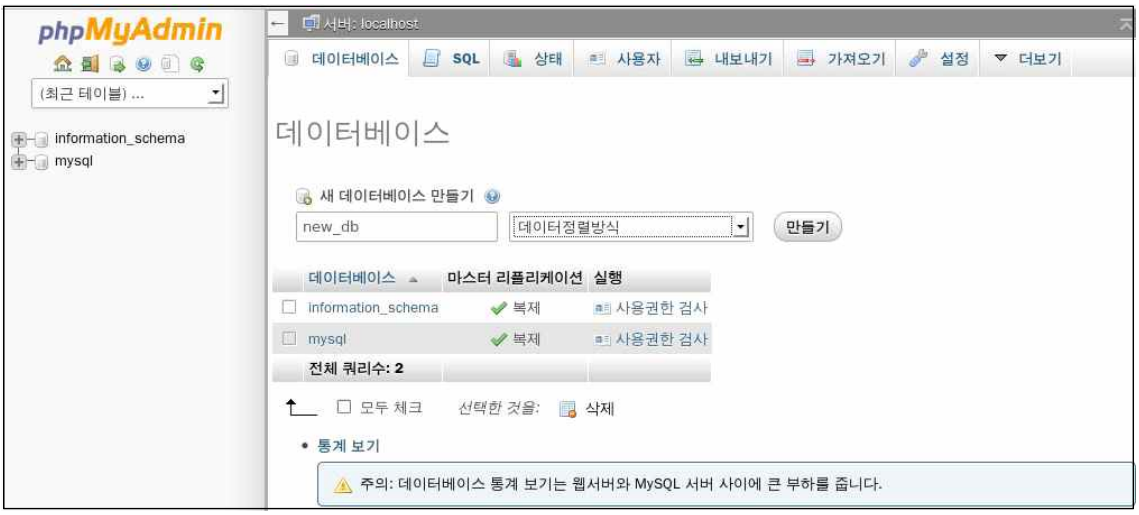
```
[root@livecd ~]# /etc/init.d/httpd restart
Stopping httpd:                                            [  OK  ]
Starting httpd:                                            [  OK  ]
```

httpd를 재시작 하면 정상적으로 구동되고 있는 것을 확인 할 수 있다.

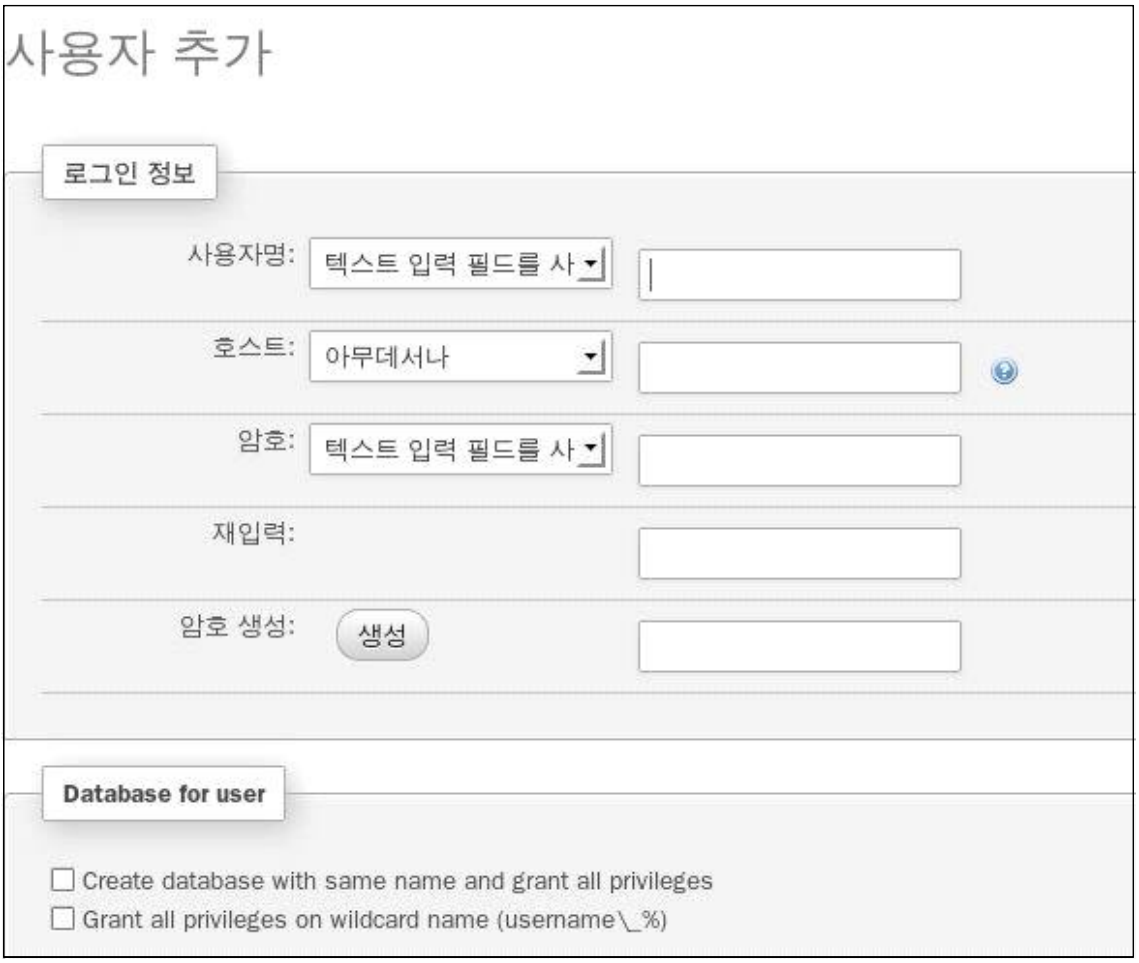
2.3.2 MYSQL계정 및 DB추가



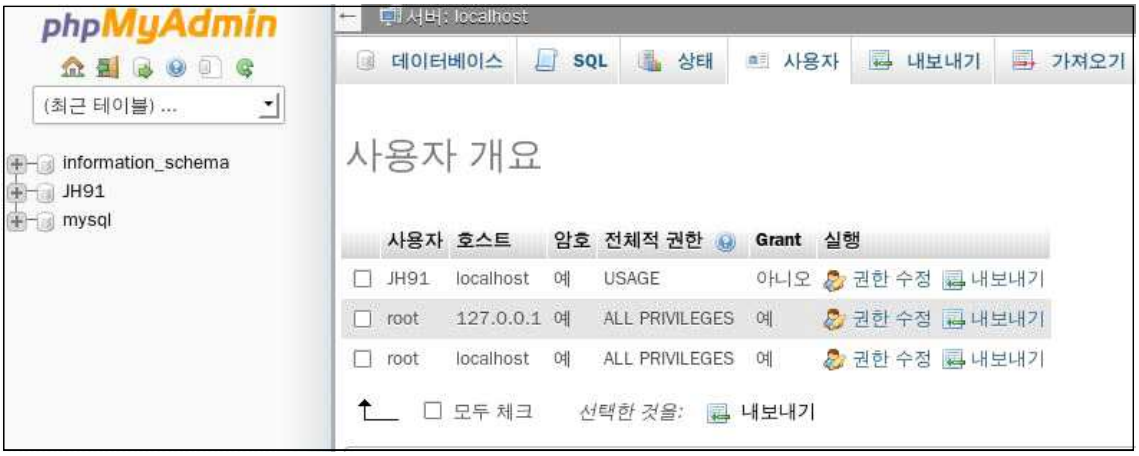
서버 연결 콜레이션을 UTF8으로 하고 한국어로 적용하여 설정을 한다.



새로운 데이터베이스를 만든다.



데이터베이스 사용자계정 정보를 입력하고 사용자를 추가한다.



추가된 사용자 목록을 확인 할 수 있다.

```
[root@livecd ~]# useradd JH91
[root@livecd ~]# passwd jh1234
```

시스템 계정을 추가한다.

```
[root@livecd ~]# ls -l /home/ |grep JH91
drwx------.  5 JH91       JH91       4096 May 12 15:42 JH91
[root@livecd ~]# chmod 710 /home/JH91
[root@livecd ~]# chgrp apache /home/JH91/
[root@livecd ~]# ls -l /home/ |grep JH91
drwx--x---.  5 JH91       apache     4096 May 12 15:42 JH91
```

계정에 웹서버 접근권을 부여한다.

```
[root@livecd ~]# echo '
> <VirtualHost *:80>
> DocumentRoot /home/JH91/public_html
> ServerName www.JH91.com
> ServerAlias JH91.com
> ErrorLog logs/JH91.com-error_log
> CustomLg logs/JH91.com-access_log combined
> </VirtualHost>
> ' >> /etc/httpd/conf/httpd.conf
```

가상호스트를 추가한다

```
[root@livecd ~]# tail -n9 /etc/httpd/conf/httpd.conf

<VirtualHost *:80>
DocumentRoot /home/JH91/public_html
ServerName www.JH91.com
ServerAlias JH91.com
ErrorLog logs/JH91.com-error_log
CustomLg logs/JH91.com-access_log combined
</VirtualHost>
```

마지막에 적용된 호스트 설정을 확인한다.

2.3.3 그누보드 설치

```
[JH91@livecd ~]$ ls -l
total 1380
-rw-r--r--. 1 root root 1405796 May 12 17:09 gnuboard4.utf8.tgz
drwxr-xr-x. 2 JH91 JH91    4096 May 12 16:21 public_html
```

그누보드4 압축파일을 다운로드 한 후 Is -l 명령으로 다운된 압축 파일을 확인해 보았다.

```
[JH91@livecd public_html]$ tar ztvfp ../gnuboard4.utf8.tgz |head
drwxrwxr-x kagla/kagla       0 2015-03-15 22:05 gnuboard4/
-rw-r--r-- kagla/kagla     717 2011-02-18 21:01 gnuboard4/tail.php
-rw-r--r-- kagla/kagla   37694 2011-02-18 21:01 gnuboard4/LICENSE
drwxr-xr-x kagla/kagla       0 2014-07-07 20:48 gnuboard4/lib/
-rw-r--r-- kagla/kagla    1045 2011-02-18 21:01 gnuboard4/lib/popular.lib.php
-rw-r--r-- kagla/kagla    2431 2011-02-18 21:01 gnuboard4/lib/trackback.lib.php
-rw-r--r-- kagla/kagla    3497 2011-02-18 21:01 gnuboard4/lib/visit.lib.php
-rw-r--r-- kagla/kagla    1406 2011-04-28 21:12 gnuboard4/lib/cheditor4.lib.php
-rw-r--r-- kagla/kagla   50625 2014-03-28 00:27 gnuboard4/lib/common.lib.php
-rw-r--r-- kagla/kagla     357 2011-02-18 21:01 gnuboard4/lib/constant.php
```

ztvfp 로 그누보드4 압축을 풀었다.

```
[JH91@livecd gnuboard4]$ mv * ../
[JH91@livecd gnuboard4]$ cd ..
[JH91@livecd public_html]$ rmdir gnuboard4/
[JH91@livecd public_html]$ ls -l
total 348
drwxr-xr-x.   3 JH91 JH91   4096 Sep  6  2013 adm
drwxr-xr-x.   5 JH91 JH91   4096 Jan 15 00:58 bbs
drwxrwxr-x.   7 JH91 JH91   4096 Apr  9  2014 cheditor5
-rw-r--r--.   1 JH91 JH91     90 Feb 18  2011 _common.php
-rw-r--r--.   1 JH91 JH91  20778 Jul 31  2014 common.php
-rw-r--r--.   1 JH91 JH91   4462 Jul  7  2014 config.php
drwxrwxr-x.   2 JH91 JH91   4096 Apr  6  2005 extend
-rw-r--r--.   1 JH91 JH91    110 Feb 18  2011 _head.php
-rw-rw-r--.   1 JH91 JH91   6383 Jul  4  2011 head.php
-rw-r--r--.   1 JH91 JH91   3003 Dec 22 20:58 head.sub.php
-rw-r--r--.   1 JH91 JH91 195858 Mar 15 22:05 HISTORY
drwxr-xr-x.   2 JH91 JH91   4096 Aug 12  2010 img
-rw-r--r--.   1 JH91 JH91    880 Mar 28  2014 index.php
drwxr-xr-x.   3 JH91 JH91   4096 Mar 15 22:05 install
-rw-r--r--.   1 root root     35 May 12 16:21 JH91test.php
drwxr-xr-x.   2 JH91 JH91   4096 Aug 20  2014 js
drwxr-xr-x.   2 JH91 JH91   4096 Jul  7  2014 lib
-rw-r--r--.   1 JH91 JH91  37694 Feb 18  2011 LICENSE
-rw-r--r--.   1 JH91 JH91     30 Feb 18  2011 perms.sh
drwxr-xr-x.  12 JH91 JH91   4096 Aug 15  2010 skin
-rw-r--r--.   1 JH91 JH91   1310 Feb 18  2011 style.css
-rw-r--r--.   1 JH91 JH91    110 Feb 18  2011 _tail.php
-rw-r--r--.   1 JH91 JH91    717 Feb 18  2011 tail.php
-rw-r--r--.   1 JH91 JH91   1513 Dec 16 02:13 tail.sub.php
```

그누보드가 메인이 되는 사이트라면 http://www.php79.com/gnuboard4/index.php 주소 대신, http://www.php79.com/index.php 주소로 설치하고 싶을 것이다 이 경우엔 다음처럼 gnuboard4/ 디렉토리안의 파일을 상위 디렉토리로 이동해줘야 한다.

웹브라우저를 열어 앞서 추가한 가상호스트 주소로 접속한다 .

– 로그인하고 나면, [ADMIN]메뉴를 클릭합니다.

– 이제 관리자화면입니다. 여기에서 각종 환경설정과 게시판관리를 진행하면 됩니다.

모든 설치가 끝난 뒤 관리자 로그인으로 접속한 관리자 화면이다.

2.4. 쉘 프로그래밍

2.4.1. 쉘 스크립트 란?

셸 스크립트(shell script)는 셸이나 명령 줄 인터프리터에서 돌아가도록 작성되었거나 한 운영 체제를 위해 쓰인 스크립트이다. 단순한 도메인 고유 언어로 여기기도 한다.

리눅스에서 쓸 수 있는 모든 쉘들처럼, BASH(**B**ourne **A**gain **Sh**ell)은 뛰어난 명령 라인 쉘이면서, 그 자체로도 하나의 스크립팅 언어이다. 당신은 쉘 스크립팅을 이용해서 쉘이 가진 능력을 충분히 활용할 수 있으며, 쉘 스크립팅이 아니었으면 수많은 명령을 필요로 했을 많은 일들을 자동적으로 처리할 수도 있다. 당신의 리눅스 박스에 놓여 있는 많은 프로그램들은 쉘 스크립트들이다. 만일 쉘 스크립트가 어떻게 작동하는지 배우고 싶거나 당신이 가지고 있는 쉘 스크립트를 수정하고 싶다면, bash 문법을 이해하는 것은 필수적이다. 게다가, bash 언어를 이해하면 정확히 당신이 원하는 방식으로 일을 하는 당신 자신의 프로그램을 작성할 수 있다.

프로그래밍 또는 스크립팅?

프로그래밍을 처음 하는 사람들은 대개 프로그래밍 언어와 스크립팅 언어 사이의 차이를 혼동한다. 프로그래밍 언어는 일반적으로 스크립팅 언어에 비해 보다 강력하고 보다 빠르다. 프로그래밍 언어의 예로는 C, C++, Java가 있다. 프로그래밍 언어는 대개 소스 코드(최종 프로그램이 어떻게 실행될 것인가에 대한 지시문을 담고 있는 텍스트 파일)에서 시작해서 컴파일 과정을 통해 실행 가능 파일로 만들어 진다(built). 이렇게 해서 만들어진 실행 가능 파일은 다른 운영 체제로 쉽게 이식되어지지 않는다. 예를 들어, 당신이 리눅스에서 C 프로그램을 작성했다면, Windows 98시스템에서는 그 프로그램을 실행할 수 없을 것이다. 프로그램을 실행하기 위해서는, Windows 98 시스템 하에서 소스 코드를 다시 컴파일해야만 한다. 스크립팅 언어 역시 소스 코드에서 시작을 하지만, 실행 가능 파일로 만들기 위한 컴파일 과정이 없다. 대신, 번역기(interpreter)가 소스 파일에서 지시문을 읽고 각 지시문을 실행시킨다. 불행히도, 번역기가 각 지시문을 하나 하나 읽어야만 하기 때문에, 일반적으로 번역기를 통해 실행되는 프로그램은 컴파일된 프로그램보다 느리다. 스크립팅 언어의 가장 큰 장점은 소스 파일을 어떤 운영 체제에나 쉽게 이식할 수 있으며 바로 그 자리에서 번역기를 통해 실행할 수 있다는 것이다. 이런 점은 작은 프로그램에서는 장점으로 여겨질 수 있지만, 큰 규모의 어플리케이션을 작성할 것을 계획하고 있다면 프로그래밍 언어를 사용하는 편이 알맞다. 스크립팅 언어의 예로는 Perl, Lisp, Tcl이 있다.

2.4.2. 쉘 프로그래밍의 기본 구조

쉘은 명령어 해석기로 커널과 사용자 중가넹 놓여있다 그렇지만 강력한 프로그래밍 언어이다. 보통 스크립트라 부르는 쉘프로그램은 시스템,콜,툴,유틸,실행파일 등을 묶어 어플리케이션을 쉽게 만들어주는 역할을 한다.

사실상 쉘 스크립트 에서는 온갖 종류의 유닉스 명령어 유틸,툴 등을 쓸 수 있고 test나 loop문 등의 내부 관리자의 시스템 작업이나 반복적인 일등에 적절한 프로그래밍 언어이다.

2.4.3. Crontab

일정시간마다 자동으로 실행시키는 데몬으로 정기적 백업작업을 할 때 많이 이용하는 기능이다.

2. 데몬 시작과 종료

#/etc/rc.d/init.d/crond [start/restart/stop]

3. 옵션 및 형식

#Crontab [옵션]

-l	현재 crontab 내용 출력
-e	crontab 내용을 작성 및 수정
-r	crontab 내용 삭제
-u	root가 해당 사용자의 crontab 파일을 다룰 때 사용

작업 형식

[MM] [HH] [DD] [mm] [d] [command]	
MM	분 (0~59 까지 사용)
HH	시 (0~23 까지 사용)
DD	일 (1~31 까지 사용)
mm	월 (1~12 까지 사용)
d	요일 (0~7 까지 사용)
command	실행할 명령어

1분 단위 실행
*/1 * * * * 명령문

30초 단위 실행 sleep 이용

* * * * * 명령어 & sleep 30; 명령문

30초 단위로 실행

```
[root@localhost /]# crontab -l
*/1 * * * * /Backup.sh
*/1 * * * * rsync -av --delete -e ssh 192.168.111.132:/ /
```

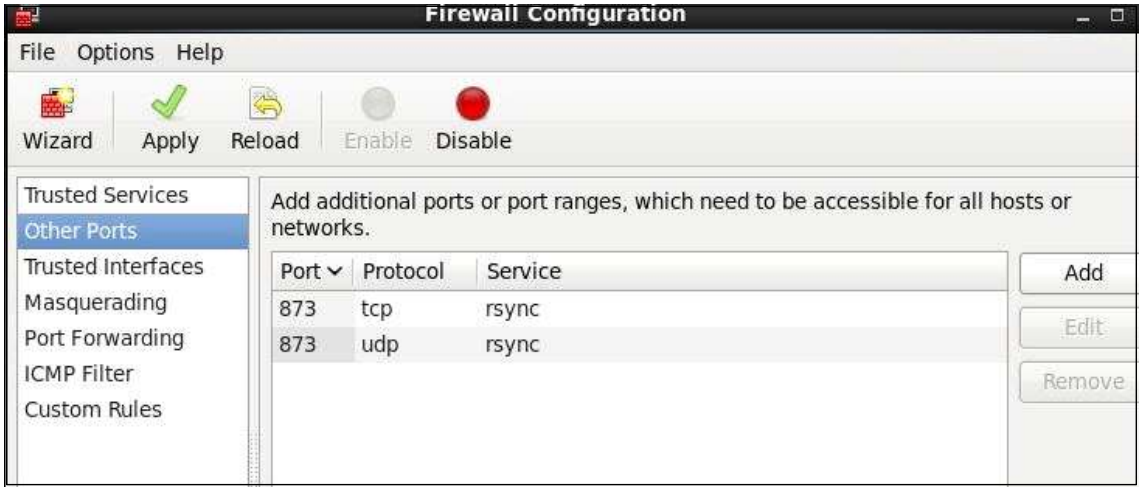
1분단위로 /Backup.sh 쉘 실행
1분단위로 rsync 실행

2.5.1 Rsync

2.5.1 Rsync의 기본구조

Rsync는 파일이 변경된 경우 그 변경이 있었던 파일에 대해서 새롭게 복사해오는 구조이다. 물론 A서버에는 존재하고 B서버에는 존재하지 않는 파일이라면 B서버로 새롭게 복사해오는 것도 가능하다.

2.5.2 Rsync의 미러서버 동기화



system-config-firewall 들어가서 rsync 포트 873 열어 오픈 후 iptables restart 한다.

```
1 # default: off
2 # description: The rsync server is a good addition to an ftp server, as
  it \
3 #       allows crc checksumming etc.
4 service rsync
5 {
6         disable = no
7         flags           = IPv6
8         socket_type     = stream
9         wait            = no
10        user            = root
11        server          = /usr/bin/rsync
12        server_args     = --daemon
13        log_on_failure  += USERID
14 }
```

vi 편집기로 /etc/xinetd.d/rsync 들어가서 6번행 disable 부분을 yes -> no 변경한다.

```
[MOS]
path = /
comment = rsync_test
uid = root
gid = root
use chroot = yes
readonly = yes
hosts allow = 192.168.57.133
max connections = 3
timeout = 600
```

vi 편집기로 /etc/rsyncd.conf 들어가서 설정한다

[MOS]	사용할 rsync 서비스 이름
path = /	데이터 원본 경로
comment = rsync_test	코맨트
uid = root	권한 사용자
gid = root	한 그룹
hosts allow = 192.168.57.133	rsync 클라이언트 IP. 허용할 클라이언트 IP
max connections = 3	최대 연결 수
timeout = 300	타임아웃 값

```
[root@localhost ~]# service xinetd restart
Stopping xinetd:                                           [  OK  ]
Starting xinetd:                                           [  OK  ]
```

Rsync 설정을 적용하기 위해 xinetd 재시작

Rsync 실행

rsync [옵션] [서버IP]:: [서비스명] 저장디렉토리	
-a	-rlptg 와 동일(r옵션 l옵션등 모두 다 포함함. 그래서 이것만 쓰면 됨)
-v	진행과정보임
-z	압축해서 전송 (요즘은 네트워크 속도가 빨라서 이거쓰면 더 느림)
-- progress	진행상태
--stats	작업끝나고 결과정보(파일수, 전송한파일수,전체사이즈,전송한사이즈등등)
--delete	원본에서 지운거 목적지에서도 삭제함. (원본에 없는 파일이 있으면 삭제)

```
[root@localhost ~]# ifconfig
eth2      Link encap:Ethernet  HWaddr 00:0C:29:5E:3D:C5
          inet addr:192.168.111.132  Bcast:192.168.111.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe5e:3dc5/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:26048 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8361 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:36182864 (34.5 MiB)  TX bytes:455167 (444.4 KiB)
          Interrupt:19 Base address:0x2024

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:182 errors:0 dropped:0 overruns:0 frame:0
          TX packets:182 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:38540 (37.6 KiB)  TX bytes:38540 (37.6 KiB)
```

메인서버 IP

```
[root@localhost ~]# ifconfig
eth2      Link encap:Ethernet  HWaddr 00:0C:29:5E:3D:C5
          inet addr:192.168.111.132  Bcast:192.168.111.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe5e:3dc5/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:26048 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8361 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:36182864 (34.5 MiB)  TX bytes:455167 (444.4 KiB)
          Interrupt:19 Base address:0x2024

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:182 errors:0 dropped:0 overruns:0 frame:0
          TX packets:182 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:38540 (37.6 KiB)  TX bytes:38540 (37.6 KiB)
```

미러서버 IP

```
[root@hs /]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:68:96:37
          inet addr:192.168.111.133  Bcast:192.168.111.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe68:9637/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:492 errors:0 dropped:0 overruns:0 frame:0
          TX packets:390 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:420615 (410.7 KiB)  TX bytes:30483 (29.7 KiB)
          Interrupt:67 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:1879 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1879 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2315736 (2.2 MiB)  TX bytes:2315736 (2.2 MiB)
```

미러서버에서 Rsync 실행

메인서버와 동기화

2.6. 동기화에 따른 백업결과

```
[root@localhost /]# rsync -av --delete -e ssh 192.168.111.132:/back_up /backup
The authenticity of host '192.168.111.132 (192.168.111.132)' can't be establishe
d.
RSA key fingerprint is 95:f9:63:26:40:4e:b6:d2:a4:3c:a9:b4:c0:a0:45:20.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.111.132' (RSA) to the list of known hosts.
root@192.168.111.132's password:
receiving incremental file list
created directory /backup
back_up/
back_up/BackUP/
back_up/BackUP/15-05-20/
back_up/BackUP/15-05-20/gnu.tar.gz
back_up/BackUP/15-05-20/gnuboardDB.tar.gz
back_up/BackUP/15-05-20/ib_logfile0.tar.gz
back_up/BackUP/15-05-20/ib_logfile1.tar.gz
back_up/BackUP/15-05-20/ibdata1.tar.gz
back_up/BackUP/15-05-20/mysql.sock.tar.gz
back_up/BackUP/15-05-20/mysql.tar.gz
back_up/BackUP/15-05-20/php79.tar.gz
back_up/BackUP/15-05-27/
back_up/BackUP/15-05-27/Backup.sh.tar.gz
back_up/BackUP/15-05-27/back_up.tar.gz
back_up/BackUP/15-05-27/backup.sh.tar.gz
```

```
sent 673 bytes  received 516191 bytes  79517.54 bytes/sec
total size is 513937  speedup is 0.99
```

동기화에 따른 백업된 파일들과 작업 완료창.

3. 결론

인터넷의 가장 큰 수혜자는 전자상거래 업체가 아니라 보안 업체라는 말이 있다. 최근 보안 시장은 인터넷 관련 사업의 활황과 더불어 급성장을 보이고 있으며 큰 규모를 형성할 것으로 예측되고 있다. 최근 발생한 해킹사건들 또한 국내 보안시장에 힘을 실어주어 시장 수요가 한층 커질 전망이다. 국내 보안시장은 방화벽 및 침입탐지시스템이 대부분을 차지하고 있다.

우리가 이용하는 웹은 이제 우리 실생활 어디든 누구나 사용 할 수 있는 존재가 되었음으로 우리가 사용하는 정보 또한 안전하게 어디서든 볼 수 있고 사용 할 수 있어야 한다는 생각으로 시작하여 서버 동기화 기술을 접목시켜 DB에 회원정보를 안전하고 편리하게 쓸 수 있게 하는 목적으로 프로젝트를 진행하였다 그 결과 안전하게 백업된 DB의 회원정보를 확인 했고 Rsync 동기화 또한 Crontab 예약 작업으로 안정적으로 동기화 되는 것을 확인 할 수 있었다.

침입탐지 및 관리시스템 구현

2015. 6. 9

지도교수 : 유승재 교수님

M. O. S

Index

- 조원 소개 및 역할
- 주제 선정
- 추진 경과
- 시스템 구상도
- 시스템 구축 및 운영
- 결　론

조원 소개 및 역할

이 름		역 할
조장	최재현	졸업작품 총괄 및 미러서버 구축
조원	박은혜	자료조사 및 쉘 스크립트 작성
조원	이상섭	자료조사 및 IDS 구축
조원	손민수	자료조사 및 웹 서버 구축
조원	황하림	자료조사 및 방화벽 구축

주제 선정

♦ 주제 : 침입탐지 및 관리시스템 구현

♦ 선정 사유

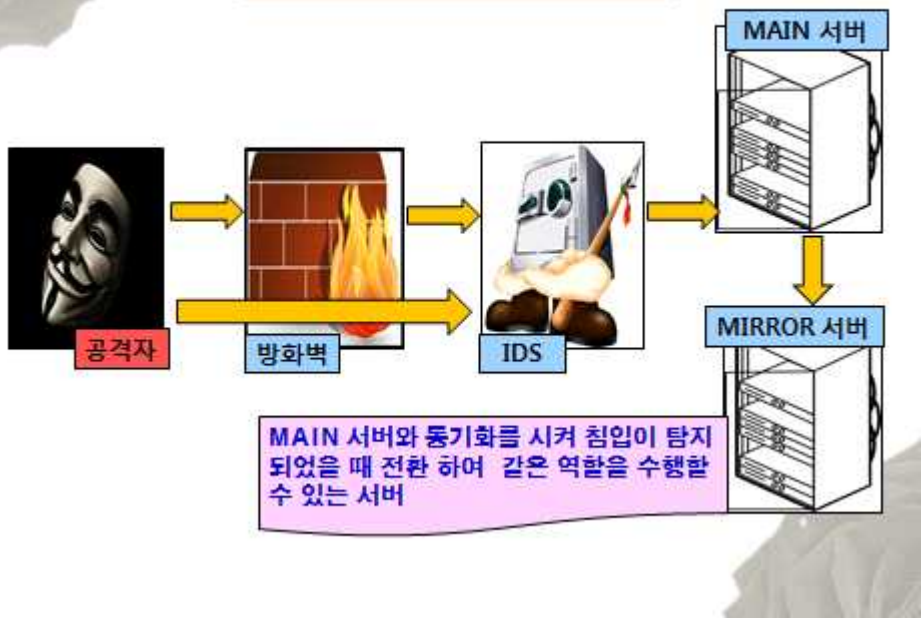
- 방화벽의 종류가 많지만 packet Filtering 방법은 바이러스에 감염된 첨부파일 등을 전송할 경우 차단이 불가능

- packet Filtering의 단점을 보완하기 위해 IDS와 서버 동기화를 이용한 미러서버를 구축, 좀더 높은 수준의 보안을 구현

추진 일정

구 분	2014년				2015년				
	9월	10월	11월	12월	1월	2월	3월	4월	5월
주제 선정	■	■							
자료수집			■	■					
방화벽 구축				■	■				
IDS 구축						■	■		
웹서버 구축						■	■		
Rsync동기화							■	■	
쉘 스크립트							■	■	
시스템 점검								■	■

시스템 구상도



시스템 구축

운영 환경

❍ 운영 체제 : 리눅스(CentOS 6.5)

❍ 개발 언어 : 쉘 스크립트

※ IDS 연동 및 동기화부 자체 개발

시스템 구축

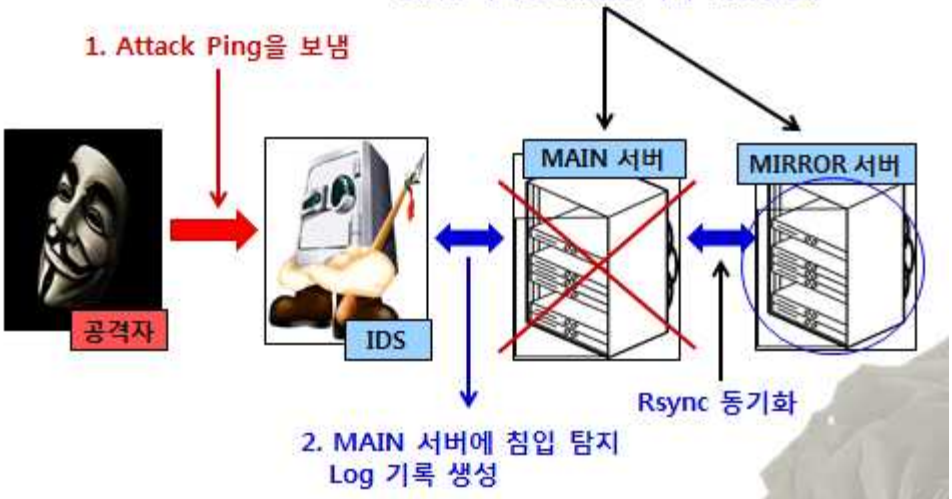
기술 분석

방화벽	IDS
♦ IP와 Port를 통한 보안정책	♦ 접속 IP에 상관없이 침입 차단
♦ 어플리케이션 레벨 방화벽에 비해 적용 및 운용이 편리	♦ 해킹방법 기반으로 빠른 신기술 적용
♦ 바이러스에 감염된 메일과 첨부파일 등 차단 불가	♦ 새로운 침입기법에 적시적 대응 한계
♦ 정책이 많을수록 Delay 증가	♦ 보안사고에 대한 근본적인 해결책이 될 수 없음

방화벽과 IDS의 장점을 취사선택하여 활용

시스템 운영

시스템 구성



시스템 구축

쉘 스크립트 제작

DB 백업쉘

```
#!/bin/bash
export Today="`date '+%y-%m-%d'`"
backup_dir="/back_up/BackUP"

################ 이전 백업 디렉토리 삭제
LDATE=$(date -d yesterday +%y-%m-%d)
rm -rf ${backup_dir}/$LDATE

################ 오늘 날짜로 백업 디렉토리 생성
/bin/mkdir -p ${backup_dir}/$Today
cd ${backup_dir}/$Today

################ / 디렉토리를  압축
dirlists=`/bin/ls -t / 2>/dev/null`
for dir in $dirlists ; do
    tar cvfpz ${backup_dir}/$Today/$dir.tar.gz /$dir
done

################ MySQL 데이터 압축
tar cvfpz ${backup_dir}/$Today/mysql.tar.gz /var/lib/mysql
~
~
~
"Backup.sh" 20L, 581C
```

Log 탐지 쉘

```
#!/bin/sh
cmd='tail -n 1 /var/log/snort'
shutdown='poweroff'
r=`$cmd`

while :
do
    if [ $r != `$cmd` ];
    then
        service network start
        service httpd start
        a=`$shutdown`
        exit
    fi
    sleep 0.1
done
```

Crontab 설정

```
[root@localhost /]# crontab -l
*/1 * * * * /Backup.sh
*/1 * * * * /log.sh
* * * * * rsync -av --delete -e ssh 192.168.111.132:/ /
```

시스템 구축

미러서버 동기화

Rsync 동기화

```
[root@localhost /]# rsync -av --delete -e ssh 192.168.111
The authenticity of host '192.168.111.132 (192.168.111.13
d.
RSA key fingerprint is 95:f9:63:26:40:4e:b6:d2:a4:3c:a9:b4:c0:a0:45:20.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.111.132' (RSA) to the list of known hosts.
root@192.168.111.132's password:
receiving incremental file list
created directory /backup
back_up/
back_up/BackUP/
back_up/BackUP/15-0
back_up/BackUP/15-0
back_up/BackUP/15-0
back_up/BackUP/15-05-20/ib_logfile0.tar.gz
back_up/BackUP/15-05-20/ib_logfile1.tar.gz
back_up/BackUP/15-05-20/ibdata1.tar.gz
back_up/BackUP/15-05-20/mysql.sock.tar.gz
back_up/BackUP/15-05-20/mysql.tar.gz
back_up/BackUP/15-05-20/php79.tar.gz
back_up/BackUP/15-05-27/
back_up/BackUP/15-05-27/Backup.sh.tar.gz
back_up/BackUP/15-05-27/back_up.tar.gz
back_up/BackUP/15-05-27/backup.sh.tar.gz

sent 673 bytes  received 516191 bytes  79517.54 bytes/sec
total size is 513937  speedup is 0.99
```

동기화 완료

시스템 구축

IDS 구축

Snort 설치

```
[root@localhost ~]# snort -V

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.7.0 GRE (Build 149)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.4.0
           Using PCRE version: 8.32 2012-11-30
           Using ZLIB version: 1.2.7
```

Snort rule 적용

```
#-------------
# LOCAL RULES
#-------------

alert icmp any any -> any any (msg:"ICMP ping Test"; sid:1000001;)
```

경로

```
app-detect.rules                local.rules                 protocol-voip.rules
attack-responses.rules          malware-backdoor.rules      pua-adware.rules
backdoor.rules                  malware-cnc.rules           pua-other.rules
bad-traffic.rules               malware-other.rules         pua-p2p.rules
blacklist.rules                 malware-tools.rules         pua-toolbars.rules
botnet-cnc.rules                misc.rules                  rpc.rules
browser-chrome.rules            multimedia.rules            rservices.rules
browser-firefox.rules           mysql.rules                 scada.rules
browser-ie.rules                netbios.rules               scan.rules
```

시스템 운영

웹서버 회원가입

그누보드 5에서 구현한 웹서버 회원가입 창

시스템 운영

웹서버 관리화면

그누보드 5에서 회원가입 후 회원목록을 확인한 결과

시스템 운영

Snort 구동

Snort 구동 확인

```
[root@localhost rules]# snort -c /etc/
Running in IDS mode

       --== Initializing Snort ==--
Initializing Output Plugins!
Initializing Preprocessors!
Initi
Parsi
Tagge
Log d

Initi
1 Snc
    1
    0
    0
1 Opt
0 Dyn

Commencing packet processing (pid=3430)
+-------------------[Rule Port Counts]-------------------+
|             tcp     udp    icmp      ip
|     src       0       0       0       0
|     dst       0       0       0       0
|     any       0       0       1       0
|      nc       0       0       1       0
|     s+d       0       0       0       0
```

Snort 탐지 가동

```
        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.7.0 GRE (Build 149)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.7.3
           Using PCRE version: 7.8 2008-09-05
           Using ZLIB version: 1.2.3
```

시스템 운영

Snort 탐지

공격 Ping 보내기

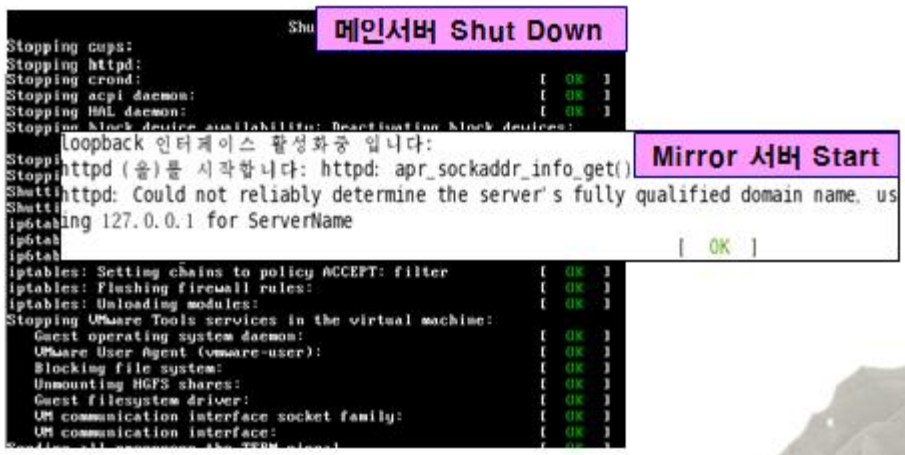
```
C:₩Users₩609>ping -l 65000 192.1

Ping 192.168.176.132 65000바이트 데이터 사용:
192.168.176.132의 응답: 바이트=65000 시간=6ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=6ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=5ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=3ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=3ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=3ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=3ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=3ms TTL=64
192.168.176.132의 응답: 바이트=65000 시간=3ms TTL=64
```

탐지 Log 생성

```
[root@localhost snort]# pwd
/var/log/snort
[root@localhost snort]# ll
total 52
-rw-------. 1 snort snort 29693 May 28 00:20 alert
-rw-------. 1 root  root   6724 May 13 10:34 snort.log.1431538458
-rw-------. 1 root  root      0 May 14 22:34 snort.log.1431668050
-rw-------. 1 root  root      0 May 14 22:52 snort.log.1431669179
-rw-------. 1 root  root      0 May 14 23:17 snort.log.1431670650
-rw-------. 1 root  root      0 May 14 23:22 snort.log.1431670955
-rw-------. 1 root  root    906 May 14 23:32 snort.log.1431671431
-rw-------. 1 root  root    906 May 27 23:50 snort.log.1432795277
-rw-------. 1 root  root    318 May 28 00:20 snort.log.1432797609
```

시스템 운영

미러서버 전환



결 론

♦ 방화벽과 IDS의 장점을 결합하여 침입탐지 및 관리시스템을 구축한 결과

- 방화벽과 IDS의 상호 보완작용을 통해 보안기능이 보강될 수 있음을 확인

- 서버를 동기화 하여 본 서버와 똑같이 구축한 미러서버로 대체하여 추가 피해를 막고 서버의 서비스를 유지

- 다만 IDS의 탐지한계를 벗어나는 새로운 공격기법이나 악성코드 등에 대한 방어 기능을 구현하는 방안을 추가 연구할 필요


Q & A
Thank you