리눅스 보안 하우투

케빈 펜지 (Kevin Fenzi), kevin@tummy.com & 데이브 뢰스키 (Dave Wreski), dave@nic.com v1.1.0 2000년 3월 8일

장 범수, bschang@kldp.org 2000년 5월 9일

이 문서는 리눅스 시스템 관리자가 상대하게 되는 보안 이슈에 대한 일반적 개론을 밝힌다. 일반적인 보안 철학 등과 리눅스 시스템을 침입자로부터 보호할 방법 등의 특정 보기를 몇 가지 적어 놓았다. 보안에 관계된 자료들과 풀그림을 구할 수 있는 곳도 적어 놓았다. 개선 사항, 건설적인 비평, 첨가 사안, 그리고 수정안 등을 감사한 마음으로 수용하겠다. 여러분의 의견은 두 저자 모두에게 "Security HOWTO"를 메일의 제목에 써서 보내 주기 바란다. 번역자의 말: 원문의 순서를 지키기 위해서 머리말에 해당할 16 (번역자의 덧붙이는 말)을 번역의 뒤에 적었습니다. 이 문서의 이전 버전 문서를 읽어보신 분들이나 리눅스에 어느 정도 숙달된 분은 이 항목을 먼저 읽어보시기 바랍니다.

제 1 절 소개

이 문서는 리눅스의 보안에 영향을 주는 몇 가지 중요한 보안 문제들을 다룬다. 일반적인 생각과 인터넷을 통해 등장한 자료들에 대해서 이야기하겠다.

보안 이슈에 대해서는 다른 여러 하우투 문서들에서도 중복 언급이 되고 있으며 중복의 경우는 가능한 밝혀 놓겠다.

이 문서는 *절대로* 최신의 침탈법에 대한 안내 문서로 만든 것이 아니다. 침탈법은 수 없이 새로 밝혀지고 있다. 침탈법의 최신 정보를 구할 수 있는 곳과 침탈법을 막기 위한 일반적인 방법도 언급하겠다. 16.5 ([1 침탈 (Exploit)])

1.1 이 문서의 새 버전

이 문서의 새로운 버전은 정기적으로 *comp.os.linux.answers* 뉴스 그룹에 올려질 것이 다. 새 버전의 문서는 아래를 포함한 여러 익명 FTP 사이트들에도 올려질 것이다.

ftp://sunsite.unc.edu/pub/Linux/docs/HOWTO <ftp://sunsite.unc.edu/pub/Linux/docs/HOWTO>

이 문서는 리눅스 WWW 홈 페이지에서도 찾을 수도 있다.

http://metalab.unc.edu/mdw/linux.html <http://metalab.unc.edu/mdw/linux.html>

끝으로, 이 문서의 최신 버전은 여러 문서의 형식으로 아래에서 찾을 수 있다.

http://scrye.com/~kevin/lsh/ <http://scrye.com/~kevin/lsh/>

1.2 독자의 의견 (Feedback)

모든 평과 오류 지적, 추가 사항과 비판은 아래 두 주소로 보내기 바란다.

kevin@tummy.com <mailto:kevin@tummy.com>

와

dave@nic.com <mailto:dave@nic.com>

주의: 의견은 항시 _두_ 저자 모두에게 보내 주었으면 한다. 메일 제목에 "linux"나 "security", "HOWTO"를 반드시 적어야만 케빈의 스펨 필터를 통과할 수 있다.

1.3 결과적 손해에 대한 책임의 배제

저자들은 이 문서의 내용에서 발생되는 모든 손해에 대하여 어떤 경우에도 책임을 지지 않는다. 이 문서에 적혀 있는 개념과 예제, 그리고 일부 또는 모든 내용의 사용에 대한 책임은 사용자에 게 있다. 붙여서, 이 문서는 이른 버전이기 때문에 부정확한 내용과 오류를 가지고 있을 높은 가능성이 있다.

많은 실례와 서술들이 레드 햇 (tm: 등록 상표) 패키지의 설계와 시스템 구성을 따랐으므로, 여러분의 경우와 다를 수 있다.

필자들이 아는 한, 정해진 조건하에서, 개인적인 목적을 위한 사용이 허락되거나 평가가 허가된 풀그림들만을 서술하였다. 대부분의 풀그림들은 소스를 포함해서, 완전히 자유롭게 *GNU* `<http://www.gnu.org/copyleft/gpl.html>`의 규정 하에서 자유롭게 쓸 수 있을 것이다.

No liability for the contents of this document can be accepted. Use the concepts, examples and other content at your own risk. Additionally, this is an early version, possibly with many inaccuracies or errors.

A number of the examples and descriptions use the RedHat(tm) package layout and system setup. Your mileage may vary.

As far as we know, only programs that, under certain terms may be used or evaluated for personal purposes will be described. Most of the programs will be available, complete with source, under *GNU* `<http://www.gnu.org/copyleft/gpl.html>` terms.

1.4 저작권 정보

- 이 문서의 저작권은 (c)1998-1999 케빈 펜지 (Kevin Fenzi)와 데이브 뢰스키 (Dave Wreski)에게 있으며, 아래의 조건하에서 배포될 수 있다.
- 이 저작권 정보가 포함되는 조건으로, 리눅스 하우투 문서들은 어떠한 물리적, 전자적인 매체를 사용하여 전체 혹은 일부를 복사하고 배포할 수 있다. 상업적인 목적의 배포도 허용되고 장려된다; 하지만 이 경우에는 저자들에게 알려주기 바란다.
- 리눅스 하우투 문서들에 연계된 모든 번역이나 파생적 저술, 총체적 저술들은 이 저작권 안내에 따라야 한다. 이것은 여러분이 하우투에서 파생된 저술을 한 후에 그 문서의 배포에 대하여 추가적인 제한 조건을 달아서는 안 된다는 뜻이다. 이러한 규칙에 대한 예외가 승인될 수도 있다; 아래의 주소를 써서 리눅스 하우투 조정자를 접촉하기 바란다.
- 의문이 있다면 리눅스 하우투 조정자인 팀 바이넌 (Tim Bynun)을 아래의 주소로 접촉하기 바란다.

tjbynum@metalab.unc.edu `<mailto:tjbynum@metalab.unc.edu>`

제 2 절 개요

이 문서에서는 리눅스 시스템의 보안을 위해서 쓰는 몇 가지 절차와 자주 쓰이는 소프트웨어들을 설명하겠다. 시작에 앞서서 기본 개념을 논의하고 보안에 대한 기본적 토대를 만드는 것이 중요할 것이다.

2.1 왜 보안이 필요한가?

항시 변화하는 글로벌 데이타 커뮤니케이션의 세계에서, 그리고 값 싼 인터넷 연결이 가능한 현재에서, 또한 빠르게 움직이는 소프트웨어 개발에 있어서, 보안은 갈수록 중요한 문제로 떠오르고 있다. 컴퓨터라는 것이 개발 초기부터 보안을 염두를 두고 만든 것은 아니었기에, 보안은 근래에 와서야 기본적 필요조건으로 등장하게 되었다. 나쁜 보기를 들자면, 인터넷 상에서 한 데이타가 A 지점에서 B 지점으로 흐르는 중간의 여러 다른 지점에서, 다른 사용자들이 데이타를 가로채거나 심지어 변경해 버릴 수 있는 기회

를 갖게 된다. 심지어 여러분의 시스템에 있는 다른 사용자들이 여러분의 데이타를 – 여러분이 의도하지 않은 다른 어떤 것으로 – 악의로서 변경할 수도 있을 것이다. "크래커"라고도 불리는 침입자들에 의해서 여러분의 시스템이 무단 사용될 수도 있으며, 이들은 뛰어난 지식을 악용함으로서 여러분인 척 신분을 위장하거나, 정보를 훔치거나, 또는 여러분이 여러분 시스템을 사용하고자 함을 거부할 수 있다. 만약 아직도 "핵커"와 "크랙커"의 차이점이 무엇인가 모른다면 *http://www.tuxedo.org/˜esr/faqs/hacker-howto.html* `<http://www.tuxedo.org/~esr/faqs/hacker-howto.html>`에서 에릭 레이몬드(Eric Raymond)가 쓴 "핵커가 되는 법"을 보기 바란다. 16.5 ([2 핵커 하우투 한글판])

2.2 얼마나 안전한 것이 안전한 것인가?

우선 마음에 새겨 두어야 할 것은 어떠한 시스템도 "완벽하게 안전"할 수 없다는 것이다. 여러분이 할 수 있는 최선의 방법은 여러분의 시스템에 침입하는 일을 가능한 어렵게 만드는 것뿐이다. 평균의 가정용 리눅스 사용자로서는 크랙커에 대비하기 위해서 그리 많은 것이 필요하지 않다. 하지만 (은행, 통신 회사 등의) 위치가 노출된 잘 알려져 있는 리눅스의 사용자들 은 보다 많은 작업을 해야 한다.

계산에 두어야 할 사항은 시스템의 보안을 강화하면 할수록 시스템을 쓰기에 불편하게 된다는 것이다. 여러분은 시스템을 사용해야 하는 사용자의 관점에서도 시스템을 봄으로서 보안성과 편의성에 대한 균형을 잡아야 할 것이다. 예로서, (보안의 입장을 고려해서) 여러분의 시스템에 모뎀으로 접속해 들어오는 사용자 모두에게 콜백 모뎀을 쓰도록 할 수도 있을 것이다.16.5 ([3 콜백 모뎀])

비록 이 방법을 쓰면 보안을 보다 강화할 수는 있겠지만, 사용자의 입장에서는 로그인을 하기에 불편하게 만드는 것이 된다. 또는, 네트워크이나 인터넷에 아예 연결되지 않게 리눅스 시스템을 만들 수도 있겠지만, 이것은 유용성을 제한하게 되는 것일 것이다.

만약 중간 규모 이상의 대형 사이트라면, 어떤 수준의 보안이 필요하고 이것을 점검하기 위해서는 어떤 절차 감사 (auditing: 監査)가 필요한 것인가를 밝히는 보안 수칙을 준비하는 것이 좋다. 보안 수칙의 예제는 *www.faqs.org/rfcs/rfc2196.html* `<http://www.faqs.org/rfcs/rfc2196.html>`를 참고하면 될 것이다. 이 문서는 최근에 갱신되었으며, 여러분 회사의 보안 규칙을 성립함에 중요할 뼈대를 담고 있다. 16.5 ([4. 인터닉 문서]) [[[delete this]]]

2.3 무엇을 보호할 것인가?

어떤 위험에 대비해야 할 것인가, 어떤 위험부담을 감수하거나 감수하지 않을 것인가, 그렇다면 결과적으로 시스템이 얼마나 취약하게 되는 것인가 등을 미리 생각해 놓는 것이 좋을 것이다. 무엇을 보호하는가, 왜 보호하는가, 이 보호 대상의 가치는 얼마나 되는가, 그리고 데이타와 자산에 대해서 누가 책임을 질 것인가를 분석하자.

- **위험 요소** *(risk)*란 침입자가 시스템을 성공적으로 침입하는 경우를 암시한다. 여러분에게 중요한 파일을 침입자가 읽거나 쓰고, 심지어는 풀그림을 실행할 수 있는 가? 중요한 데이타를 지울 수 있는가? 여러분이나 여러분의 회사가 중요한 업무를 실행하는 것을 훼방놓을 수 있는가? 또한 여러분의 계정이나 시스템에 엑세스를 가지고 있는 사람이 여러분을 사칭할 수도 있다는 것을 잊지 말아야 한다. 또한, 보안이 취약한 계정 한 개 때문에 전체 네트워크가 침입을 당하는 결과가 생길 수도 있다. 설사 단 한 명의 사용자라 하더라도 리모트 호스트 (`rhost`의 사용을 허락해 주거나, 혹은 `tftp` 등의 보안이 불완전한 서비스의 사용을 허용함으로 침입자에게 "발 디딜 자리"를 주는 결과가 생기는 것이다. 침입자가 여러분 시스템이나 다른 시스템에 사용자 계정을 갖은 순간부터 이 계정은 다른 계정이나 다른 시스템으로의 엑세스를 얻는데 사용될 수 있다.

- **위험 요소** *(threat)*는 여러분 네트워크나 컴퓨터에 불법 엑세스 (unauthorized access)를 얻고자 하는 생각이 있는 사람으로부터 생성된다. 누구를 신임해서 여러분의 시스템에 엑세스를 허락할 것인가, 어느 정도의 위험부담을 그 사람이 발생시키는가 등을 잘 결정해야 한다. 침입자라는 집단은 여러 부류로 나뉘어지며, 보안 작업을 실행 할 때에는 그들 각 각의 특성을 염두에 두는 것이 좋다.

- **궁금증이 많은 사람 (The Curious)**: 이 종류의 침입자는 기본적으로 여러분이 어떤 시스템과 데이타를 가지고 있는가 정도를 알고자 하는 것에 흥미를 둔다.
- **악의가 있는 사람 (The Malicious)**: 이 종류의 침입자는 여러분의 시스템을 다운시키거나 여러분의 웹 페이지를 손상시키거나, 손해를 복구하게 하는 등으로 여러분의 시간과 돈을 낭비하게 만든다.
- **명성을 얻으려는 사람 (The High-Profile Intruder)**: 이 종류의 침입자는 인기와 악명을 얻기 위해서 여러분의 시스템을 쓰려고 한다. 잘 알려진 시스템을 침투함으로서 자신의 능력을 선전하려고 하는 것이다.
- **경쟁자 (The Competition)**: 이 종류의 침입자는 여러분 시스템에 무슨 데이타가 있는가에 흥미를 둔다. 돈이 될 만한 무엇을 여러분이 가지고 있다고 생각하는 불특정인 일 수도 있다.
- **도용자 (The Borrowers)**: 이 종류의 침입자는 그의 목적을 위해서 여러분 시스템을 무단 사용하면서 여러분의 시스템 자원을 훔치는 사람이다. 이들은 일반적으로 챗팅이나 IRC 서버, 포르노 아카이브 사이트 등을 여러분의 컴퓨터에서 돌리고, 심지어는 DNS 서버를 돌리기 까지 한다.
- **건너뛰기 도용자 (The Leapfrogger)**: 이 종류의 침입자는 다른 시스템으로 들어가기 위한 도구로서 여러분의 시스템을 이용하려고 한다. 만약 여러분의 시스템이 많은 수의 호스트에 연결되어 있거나 게이트웨이로 사용되는 경우라면, 이런 류의 침입자가 여러분 시스템을 깨고 들어오려는 노력을 하는 것을 이미 보았을 수도 있을 것이다.

• "취약성 (Vulnerability)"이 있다 함은 여러분의 컴퓨터가 다른 네트워크로부터 보호가 되지 않거나, 누군가가 여러분 컴퓨터에 불법 엑세스를 얻을 가능성이 있는 경우를 뜻한다. 여러분의 시스템에 누군가가 침입했다면 무엇이 상관된 문제일까? 물론 다이내믹 PPP를 사용하는 개인 사용자의 관심은 인터넷이나 다른 큰 네트워크에 연결된 회사의 관심사와는 다르기는 하겠지만 말이다.

  손상된 데이타의 횟수와 복구에 얼마나 시간이 걸릴 것인가? 초기의 조그만 시간 투자는 잃어버린 데이타를 회복하는데 낭비되는 시간의 십분의 일도 안될 수 있다. 근래에 백업 전략을 점검하거나 백업된 데이타를 확인한 적이 있는지?

2.4 보안 수칙의 완성

사용자들이 쉽게 이해하고 따를 수 있는 간단하고 일반적인 수칙을 만들도록 해야 한다. 수칙은 관리자 여러분이 수호하는 데이타를 보호하는 동시에, 사용자의 프라이버시도 지키도록 만들어져야 한다. 숙고해야 할 것들은 누가 시스템에 엑세스를 가질 것인가 (친구들이 내 계정을 써도 될 것인가?), 누가 시스템에 소프트웨어를 설치하도록 허락될 것인가, 누가 어떤 데이타를 소유할 것인가 등과, 재해 시의 복구 대책, 시스템의 적절한 사용 등이다.

일반적으로 이용되고 있는 보안 수칙은 다음의 문장으로 시작된다:

> **허락되어 있지 않은 사항은 금지 사항으로 간주할 것.**

이것은 시스템 관리자 여러분이 허락하지 않은 시스템 서비스를 일반 사용자가 사용을 하면 안된다는 뜻이다. 이 수칙은 관리자 여러분의 일반 계정에조차도 적용이 되도록 해야 할 것이다. "도대체 이것의 허가권은 귀찮구먼. 그냥 루트로 들어가서 해 버리지 뭐" 하는 따위는, 너무도 당연히 알려져 있는 침탈법에 사용됨은 물론이고 아직 발견조차 되지 않은 침탈법에로 까지 발전 사용될 보안 개구멍을 열어 놓는 것과 다름없는 것이다.

rfc1244 `<http://www.faqs.org/rfcs/rfc1244.html>`는 네트워크 보안 수칙을 만드 는 방법을 설명해 주고 있다.

rfc1281 `<http://www.faqs.org/rfcs/rfc1281.html>`은 보안 수칙의 예제를 각 단 계의 자세한 설명과 함께 설명해 주고 있다.

마지막으로, *ftp://coast.cs.purdue.edu/pub/doc/policy* `<ftp://coast.cs.purdue.edu/pub/doc/policy>`에 있는 COAST 아카이브를 가보면, 실제 사용되고 있는 보안 수칙이 어떻게 만들어져 있는 볼 수 있다.

2.5 사이트 보안의 방법

이 문서는 – 기계, 데이타, 사용자, 네트워크, 심지어는 여러분의 명성 등, 여러분이 열심히 일하면서 만들어 모은 – "자산" 보호의 여러 방법에 대해서 논의한다. 만약에 일반 사용자의 데이타를 침입자가 지워 버렸다면 여러분의 명성은 어떻게 될까? 여러분 웹 사이트의 내용이 바꿔져 버렸다면? 여러분 회사가 추진하고 있는 다음 분기의 프로젝트를 침입자가 밝혀 버린다면? 네트워크를 설치할 때는, 단 한 대의 컴퓨터를 더하기 전에도 미리 생각해 두어야 하 는 여러 문제가 있기 마련이다.

여러분이 겨우 하나의 다이얼업 PPP 계정을 가지고 있거나 작은 사이트 하나 만을 가지고 있다고 해서, 침입자가 여러분 시스템에 흥미를 안 가질 이유는 없다. 크고 이름 있는 사이트들만이 침입 대상이라고 생각하면 안된다. 많은 침입자들은 "크기에 상관없이 가능한 많은 사이트를 침입한다"는 목적만으로 침탈을 시행한다. 덧붙여서, 여러분 사이트의 보안 개구멍 (Security hole) 을 사용해서 여러분이 연결되어 있는 다른 사이트로 우회적인 엑세스를 할 수도 있는 것이다.

침입자들은 시간이 남아도는 사람들이며, 설령 여러분이 시스템을 구석에 꽁꽁 감추어 놓는다 해도, 모든 가능한 방법을 시도해 보는 방법 등으로 우회 침탈해서 여러분의 시스템에 들어올 수 있다. 침입자가 여러분의 시스템에 흥미를 가질 이유는 많다. 이에 대해서는 뒤에 좀 더 설명을 하겠다.

2.5.1 호스트 보안

여러 보안 영역 중에서 관리자가 가장 집중적인 관심을 쏟는 부분은 아마도 호스트 보안이 아닐까 한다. 여러분이 소유한 시스템이 안전하도록 철저한 관리를 하는 것과 네트워크의 다른 사람들도 같은 정도로 보안에 만전을 다하기를 바라는 것에서부터 시작을 해야 할 것이다. 좋은 패스워드를 고르는 것, 여러분 호스트의 지역 네트워크 서비스에 대한 보안화 작업을 하는 것, 계정 사용일지를 잘 관리해서 만드는 것, 그리고 보안 개구멍이 있다고 알려진 풀그림을 갱신 교체하는 것 등이 지역 보안 관리자 (local security administrator) 임무의 일부이다. 비록 이런 것들이 철저하게 필요한 일이기는 하지만, 네트워크에 붙여지는 컴퓨터 숫자가 불어나기 시작하면 이러한 작업들은 보안 관리자의 기세가 꺾일 정도의 많은 일이 되기도 한다.

2.5.2 네트워크 보안

지역 호스트 보안처럼 네트워크 보안도 중요하다. 한 네트워크에 수 백 내지 수천의 컴퓨터가 붙어 있는 상황에서, 하나 하나의 모든 시스템이 전부 보안화 작업이 되어 있다고 믿어서는 안 될 것이다. 정식 사용자만이 네트워크를 쓰도록 만들고, 방화벽을 만들고, 강력한 암호 기법을 쓰고, 네트워크 상에 관리자 없는 주인 없는 기계 (rogue machine)따위의 안전관리가 안 되어 있는 기계가 없도록 하는 것 등이 모두 네트워크 관리자의 임무인 것이다.16.5 ([5. 주인 없는 기계])

이 문서는 여러분의 사이트에 대해서 보안화 작업을 실행하는 테크닉을 논의할 것이며, 여러분이 보호하려고 하는 것을 침입자가 엑세스를 못하도록 하는 방지법 등을 보여주려 한다.

2.5.3 은둔 보안 방식 (Security through obscurity)

"구석에 숨는 식의 보안"은 반드시 논의되어야 하는 보안법의 하나이다. 한 예를 들자면, "보안 침탈법이 있다고 알려져 있는 서비스의 포트를 비정규적인 포트로 이동해 놓으면 공격자가 당연히 이 것을 모를 것이므로 당연히 침탈해 들어오지 못할 것"이라는 따위의 생각이다. 이러한 속없는 보안법을 공격자가 어렵잖게 간파해 낼 것임을 믿어 의심치 말라. 이러한 "혼자만 알면서 쓰는 은둔 보안법"은 보안화 방법이 절대 아니다. 단지 여러분이 작은 사이트를 가지고 있다 해서, 혹은 잘 알려져 있지 않다고 해서 여러분이 가지고 있는 것에 침입자가 흥미 를 가지지 않을 것이라는 생각은 금물이다. 무엇을 보호하는 것인가는 다음 항목들에서 다루겠다.

2.6 이 문서의 구성

이 문서는 항목으로 구분해 놓았다. 각 항목은 여러 주제의 보안 이슈를 각각 다룬다. 첫째인 3 (물리적 보안) 항목은 기계에 직접 손을 대는 것을 막는 방법을 다룬다. 둘째인 4 (지역 보안)은 어떻게 시스템을 지역 사용자로부터 보호할 것인가를 다룬다. 셋째인 5 (파일과 파일시스템 보안)은 어떻게 파일시스템을 구성하고 파일의 허가권을 설정해 줄 것인가를 다룬다.6 (패스워드 보안과 암호화)는 암호화 기법을 사용해서 어떻게 여러분의 컴퓨터와 네트워크를 보호할 것인가를 다룬다.7 (커널 보안)은 여러분이 어떤 커널 옵션은 설치해야 하고, 어떤 옵션은 알아두어야 하는가를 논하겠다.8 (네트워크 보안)은 여러분의 리눅스 시스템을 네트워크 공격으로부터 어떻게 보호할 것인가를 다룬다.9 (온라인 전의 보안 준비)는 시스템을 웹에 연결하기 전에 무엇을 준비해야 하는가를 다룬다. 그 다음은 시스템 침입이 목격되고 있는 상황이거나 지난 침입이 발견된 후에는 무엇을 할 것인가를 다룬다.11 (보안 관련 자료)에서는 근본적인 보안 자료들을 열거하겠다. 마지막으로는 13 (FAQ), 그리고 14 (결론)을 적겠다.

이 문서를 읽는 중에 염두에 둘 두 가지는:

- 여러분의 시스템을 파악해 둘 것. `/var/log/messages` 등의 시스템 일지 (log)를 확인하며, 시스템에 주의를 두도록 하고,
- 둘째는 항상 현재 버전의 소프트웨어를 설치하도록 하며, 보안 경고가 나오는 데로 이에 맞게 갱신 설치를 하는 등으로 시스템을 최신의 상태로 유지하도록 할 것이다. 이렇게만 해도 시스템이 눈에 띄게 보안화 되는 것에 도움이 될 것이다.

제 3 절 물리적 보안

염두에 두어야 할 보안의 첫 순서는 컴퓨터의 물리적 보안이다. 누가 컴퓨터 본체에 직접 손을 댈 수 있는가? 그럴 필요가 있는 사람인가? 시스템에 손을 대는 것을 막을 수 있는 가? 그럴 필요가 있는가?

여러분의 시스템에 어느 정도의 물리적 보안이 필요한지는 상황과 예산에 따라 매우 다르다.

집에서 리눅스를 쓴다면 (비록 아이들이나 성가신 친척들이 컴퓨터를 건드리지 못하도록 하기는 해야겠지만) 특별히 물리적 보안에 신경을 쓸 필요는 없을 것이다. 연구실에서 쓴다면 보안에 좀 더 신경을 써야 하지만 사용자들의 작업에 지장이 없도록 되어야 한다. 다음에 나오는 여러 항목이 도움이 될 것이다. 사무실에서라면 여러분이 퇴근을 했거나 자리를 비울 경우 컴퓨터에 접근하지 못하도록 통제해야 하거나 혹은 그렇지 않은 경우도 있다. 어떤 회사에서는 컴퓨터 콘솔을 켠 채로 방치해 두는 것만으로도 해고 사유가 된다.

문에 거는 자물쇠나 쇠사슬, 캐비닛을 잠그는 것, 비디오 감시 장치 같은 당연한 물리적 보안 방법도 분명히 좋은 생각이지만, 이 문서의 범위 밖이다. :)

3.1 컴퓨터 열쇠

많은 신형 컴퓨터 케이스들에는 "잠금" 기능이 있다. 잠금 장치는 대개 열쇠나 본체 전면 열쇠를 이용해서 잠그거나 풀게 되어 있다. 이 장치를 쓰면 다른 사람이 여러분의 pc를 훔쳐 가거나, 케이스를 열고 직접 하드웨어를 조작하거나 훔쳐 가지 못하도록 할 수 있다. 또한 다른 사람이 자신들의 플로피나 다른 하드웨어를 써서 리부트하지 못하도록 할 수도 있다.

이런 케이스 잠금 장치들은 머더보드의 지원과 케이스의 구조에 따라서 다른 기능을 발휘한다. 어떤 PC들은 케이스를 (억지로) 열려고 한다면 케이스를 뜯어내야 만 하도록 만들어져 있기도 하다. 또한 어떤 PC들은 새로운 키보드나 마우스를 꽂을 수 없도록 되어 있다. 여러분의 머더보드와 케이스 구조를 점검하도록 하라. 이런 잠금 장치들은 대개 매우 질이 낮으며 자물쇠 따는 실력이 있는 공격자 앞에서야 무용지물이겠지만, 상황에 따라서는 상대적으로 유용한 것이다.

어떤 기계에는 (주로 스팍과 맥) 뒤편에 구멍(dongle)이 있어서 사용자가 쇠줄로 꿰어 놓을 수 있다. 이렇게 해 놓으면 침입자가 본체 내부에 손을 대려고 할 때, 쇠줄을 끊거나 케이스를 파괴해야 만 한다. 이 구멍에 자물쇠를 채워 놓는 것만으로도 컴퓨터를 훔치려는 사람들에게 부담을 주는 장애물이 될 수 있다.

3.2 바이오스 보안

바이오스(BIOS)는 x86 CPU 기반의 하드웨어들을 제어하는 가장 밑바탕인 소프트웨어다. 릴로나 그 밖의 리눅스 부트 프로그램들은 바이오스를 엑세스해서 리눅스 부팅 방법을 결정한다. 리눅스가 돌아가는 다른 하드웨어에도 바이오스와 비슷한 소프트웨어들이 있다. (맥과 신형 선 컴퓨터의 OpenFirmware, sun boot prom 등). 이러한 바이오스 기능들은 침입자가 컴퓨터를 리부팅하는 방법을 씀으로서 시스템을 조작하려는 것을 막기 위한 방법으로 사용될 수 있다.

많은 경우, PC 바이오스에 부트 패스워드를 설정할 수 있도록 되어 있다. 이런 기능들이 충분한 보안을 제공한다고 할 수는 없지만 (바이오스를 리셋시킬 수도 있고, 케이스를 열 수 있다면 아예 제거해 버릴 수도 있다), 쓸 만한 장애물이 될 수 있을 것이다.(예를 들면, 시간이 걸릴 것이고, 조작의 증거가 남는다) 마찬가지로 S/Linux (Linux for SPARC (tm) 프로세서용 기계)에서는 부팅 시에 패스워드를 묻도록 EEPROM을 설정할 수 있다. 완전한 방법은 아니지만, 침입 속도를 늦출 수는 있을 것이다.

많은 x86 바이오스는 이 밖에도 다양하고 훌륭한 보안 설정 기능들을 제공한다. 여러분이 갖고 있는 바이오스 매뉴얼을 확인하고 다음 부팅 때 바이오스 설정을 자세히 살펴보기 바란다. 예를 들어 플로피 드라이브로 부팅을 하지 못하게 하거나 특정한 바이오스 기능에 접근하기 위해서는 패스워드를 입력하도록 하는 기능 등이 있을 수 있다.

주의: 혹시 서버 머신에 부트 패스워드를 설정했다면, 여러분이 있어야만 부팅이 가능하다는 점을 명심해야 한다. 밤중에 정전이 되면 다시 돌아와서 패스워드를 입력해야 한다. ;(

3.3 Boot loader 보안

여러 가지 리눅스 부트 로더로도 패스워드 보안을 설정할 수 있다. 릴로를 예로 들면, **패스워드** (password)와 **리스트릭티드** (restricted - 부트 제한 기능이 있는 옵션 명령어) 등의 명령어가 있다: `password` 명령어를 쓰면 부팅 때 항상 패스워드를 물어 보도록 만들 수 있으며, `restricted` 명령어를 쓰면 (릴로 프롬프트에서 `single` 등의 특별한 옵션을 사용한 경우에 대비해서) 부팅 패스워드를 조건적으로 물어 보도록 구성하게 된다.16.5 ([6. 릴로 문서])

lolo.conf 맨 페이지에서:

```
password=password
     이미지 당 사용 옵션인 'password=...'는 (아래 참조) 모든 이미지에서 사용할 수 있음.

restricted
     이지지 당 사용 옵션인 'restricted'는 (아래 참조) 모든 이미지에서 사용할 수 있음.

     password=password
          이미지에 패스워드를 설정해 줌.

     restricted
          코맨드 라인에 매개 변수가 설정된 경우에는
          패스워드를 반드시 사용해야만 이미지를 부팅할 수 있도록 설정해 줌.
          (예; single).
```

일단 패스워드들을 설정하고 나면 패스워드들을 모두 기억해 두어야 한다는 점을 명심하라. :) 아울러 패스워드들 또한, 침입하겠다고 굳게 마음을 먹은 침입자가 있는 경우에는 단지 시간을 벌어 주는 역할밖에 못한다는 점을 기억해야 한다. 이 방법으로는 침입자가 플로피로 부팅을 해 서 루트 파티션을 마운트하는 것을 막을 수가 없다. 만약 부트 로더 보안을 이용한다면, 바이오스에서 플로피 부트 옵션을 꺼 버리고 바이오스에 패스워드를 걸어 놓는 방법 등과 섞어서 사용하는 것이 좋을 것이다.

혹시 다른 부트 로더의 보안 관련 정보를 아는 사람이 있다면 필자에게 알려주기 바란다 (`grub`, `silo`, `milo`, `linload` 등).

주의: 만약 여러분이 서버를 관리하고 있고 이 서버에 부트 패스워드를 걸어 놓는다 면, 자동으로 시동이 되도록 설치를 하는 것이 불가능하다는 것을 염두에 두어야 한다. 정전 등의 상황이 벌어지면 여러분이 직접 다시 부트 패스워드를 넣어 주어야 한다.

3.4 xlock과 vlock

여러분이 컴퓨터 앞을 자주 비운다면, 다른 사람이 여러분의 작업을 엿보거나 변조하지 못하도록 콘솔을 "잠글" 수 있는 것이 좋다. 이런 기능이 있는 두 가지 풀그림이 `xlock`과 `vlock`이다.

`xlock`은 X 윈도우 화면을 잠근다. 이 풀그림은 X를 지원하는 모든 리눅스 배포본에 들어 있다. 일반적으로 여러분이 사용하는 단말기의 아무 xterm에서나 `xlock`을 실행시 킬 수 있으며, 일단 실행되면 화면이 잠기게 되고 여러분의 패스워드가 입력되어야 화면을 입력 가능 상태로 되돌릴 수 있게 된다. 더 자세한 옵션은 해당 man 페이지를 찾아보도록 하라.

`vlock`은 리눅스 가상 단말기의 일부나 전부를 잠글 수 있도록 하는 간단한 풀그림이다. 여러 가상 단말기 가운데 본인이 작업 중인 하나 만을 잠글 수 있는데, 이렇게 되면 다른 사람들이 들어와서 다른 단말기는 쓸 수가 있지만, 여러분이 작업 중이던 가상 단말기는 본인이 해제하기 전에는 쓸 수 없게 된다. 이 `vlock`은 레드 햇 리눅스에는 들어 있지만, 여러분이 사용하는 배포본에는 없을 수도 있다.

하지만 단말기를 닫아 두는 정도로 여러분의 작업을 누군가가 조작하는 것 정도야 막을 수는 있겠지만, 침입자가 컴퓨터를 다시 부팅시키거나 작업을 중단시켜 버리는 것은 막을 수 없다. 또한 이 방법만으로는 침입자가 네트워크 상의 다른 컴퓨터를 이용함으로 여러분 컴퓨터에 문제를 일으키는 것을 막을 수 없다.

더 중요한 것은, 이 방법은 침입자가 X 윈도우에서 빠져 나오는 것을 막을 수는 없기 때문에, 침입자가 보통의 버츄얼 콘솔 로그인 프롬프트를 가지게 되는 것과 여러분의 권한을 훔치기 위해서 X11이 시작된 버츄얼 콘솔에 들어가서 X11을 잠정적으로 중지시키는 행위 따위를 완전히 막을 수가 없다는 것이다. 이런 이유로, 이 방법을 꼭 써야 한다면 xdm이 설정된 상황 아래에서 만 조건적으로 쓰기를 권한다.

3.5 물리적인 보안 파손의 감지

늘 점검해야 할 최우선 사항은 여러분의 컴퓨터가 언제 재부팅되었는가 이다. 리눅스는 견고하고 안정적인 운영체제이기 때문에 컴퓨터가 재부팅되어야만 하는 때는 운영체제의 업그레이드나 하드웨어 교체 등을 위해서 재부팅된 경우뿐이다. 만약 여러분이 하지 않았는데도 컴퓨터가 재부팅되었다면, 침입자가 일을 저지른 것이라는 신호일 수 있다. 컴퓨터에 침입하는 방법 중에는 컴퓨터를 재부팅시키거나 전원을 꺼야 하는 경우가 필요한 것이 많으므로.

컴퓨터 케이스와 그 주변에 어떤 조작의 흔적이 있는지 확인하도록 한다. 또한, 많은 경우에 침입자들은 일지 문서에서 자신의 흔적을 지워 버리지만, 그래도 일지 문서 (Log File: 日誌 文 書) 모두를 잘 살펴보고 어긋나는 불일치 점을 살피는 것이 좋다.16.5 ([7 일지 문서])

또한 일지 문서를 — 보호 작업이 잘 된 네트워크에 존재하는 일지 전용 서버 등의 — 안전한 장소에 보관하는 것도 좋은 생각이다. 일단 침입자에 의해서 컴퓨터의 보안이 이미 뚫려진 상황이라면 침입자가 일지 문서도 이미 변경을 해 놓을 수 있다.

시스로그 데몬 (syslog daemon)을 쓰면. 일지 데이타를 한 장소의 중앙 일지 서버로 자동으로 보내도록 만들 수 있다. 하지만, 이 경우에 데이타는 평문 (Unencrypted text)으로 보내지게 되므 로, 데이타가 송신되는 것을 침입자가 중간에서 가로챌 수가 있다. 이러한 상황은 일반 대중이 보아서는 안 될 네트워크 정보가 유출되는 심각한 경우이다. 물론 이런 경우에는 데이타를 암호화해서 보내는 시스로그 데몬을 구해서 쓰면 될 것이다. 16.5 ([8. Cleartext])

또한, 인터넷 등에 이미 구성되어서 배포되어 있는 침탈 풀그림을 사용하면, 침입자가 시스로그 메시지를 가짜로 꾸미는 것이 매우 쉽다는 것을 알아야 한다. 심지어 시스로그는 제의 출처를 밝히지 않은 채로 로칼호스트에서 날아오는 것처럼 날조되어 보내져 오는 네트워크 일지 기록을 받아서 기록하기까지 한다.

여러분 로그에서 확인해야 할 것은:

- 짧거나 불완전한 기록.
- 이상하거나 잘못된 시간 도장 (timestamp)을 가진 기록.
- 잘못된 허가권이나 소유권을 가진 기록.
- 재부팅이나 서비스의 재시작에 대한 기록.
- 없어지거나 지워진 기록.
- su 사용 기록과 이상한 곳으로부터의 접속 기록.

시스템 일지 데이타에 대해서는 9.4 (일지의 조사)에서 좀더 논의하겠다.

제 4 절 지역 보안

다음으로 주목해야 할 것은 지역 사용자(local user)들의 공격에 대한 보안이다. 필자가 방금 ***지역*** 사용자들이라고 말했다는데 주목하기 바란다.

지역 사용자의 사용권을 얻는 것이야말로 시스템에 침입하고자 하는 사람들이 가장 먼저 시도하는 것 가운데 하나다. 지역 사용자들에 대한 보안이 느슨하면, 침입자가 – 여러 가지 버그들과 시스템이 제공하는 서비스 허점을 이용해서 – 그들이 도둑질한 일반 사용자 계정의 사용권을 관리자 (root) 사용권으로 "업그레이드" 해낼 수 있다. 그러므로 지역 사용자들에 대한 보안을 철저히 하면 침입자들이 뛰어넘어야 할 또 하나의 장애물을 만들에 주게 되는 셈이다.

설령 가짜 사용자가 아니라 해도 (특히 진짜 사용자인 경우에도) 지역 사용자들은 여러분의 시스템을 쑥밭으로 만들 수 있다. 여러분과 안면이 없거나 연락 방법을 모르는 사람에게 계정을 주는 것은 매우 좋지 않은 생각이다.

4.1 새로운 계정 만들기

사용자에게 계정을 줄 때에는 작업을 위한 최소한의 권한만을 부여하도록 해야 한다. 열 살 난 아들에게 계정을 준다면, 워드 프로세서와 그림 풀그림 엑세스 정도를 주어야 하며, 그의 것이 아닌 데이타를 지울 수 없도록 권한을 제한해야 한다.

다른 사람들에게 적절한 엑세스 권한을 제공하려 할 때, 염두에 두면 좋을 경험적인 법칙들이 있다.

- 사용자들에게 필요로 하는 최소의 권한만을 준다.
- 사용자들이 언제/어디서 로그인하는지 혹은 로그인해야 하는지 알아야 한다.
- 폐쇄되었거나 사용이 되지 않고 있는 계정은 지운다.
- 개개인의 사용자는 네트워크 전반의 모든 컴퓨터에 대해서 동일한 유저아이디 (userid)를 유지하면서 사용하도록 하는 것이 좋다. 일지 기록을 분석할 때와 개인 계정 관리를 할 때 편하기 때문이다.
- 그룹 유저아이디 (userid)를 만드는 것은 철저하게 금지되어야 한다. 사용자의 개인 계정만 사용하게 한다면 사용자의 사용 책임 인과가 분명하게 되지만, 그룹 계정을 쓴 경우에는 책임 인과의 분명성을 구성하기가 불가능하다.

여러 달이나 여러 해 이상 사용되지 않고 있는 지역 사용자 계정은 종종 침입의 도구로 사용된다. 아무도 사용하고 있지 않기 때문에 이런 계정들은 최상의 공격 도구가 된다. 16.5 ([37. 휴면 계정])

4.2 루트 보안

컴퓨터에서 가장 추구되는 계정은 루트 (수퍼유저 Superuser) 계정이다. 이 계정은 컴퓨터 전체에 대한 권한이 있으며, 네트워크에 있는 다른 컴퓨터에도 권한을 가질 수 있도록 만들어져 있기도 하다. 루트 계정을 사용할 때에는 가능한 짧은 시간 안에 특별한 작업만을 하기 위한 경우에만 써야 하며, 관리자 여러분 자신도 평상시에는 일반 사용자용 계정을 써서 사용하는 것이 좋다는 것을 명심해야 한다. 루트로 로그인해서 저지르는 실수가 아무리 작은 것이라고 할 지라도 이 것은 큰 문제를 일으키게 된다. 루트 권한을 갖는 시간이 짧으면 짧을수록 안전한 것이다. 항상 루트로 작업을 하는 것은 매우, 정말로, **진짜로 나쁜 생각**이다.

루트로 작업하다가 자신의 컴퓨터를 뒤죽박죽으로 만드는 것을 피하기 위한 몇 가지 비결이 있다.

- 복잡한 명령을 써야 할 때는 우선 파괴적이지 않은 방식을 사용하라. 와일드 카드를 쓰는 명령의 경우에 특히 주의해야 한다. 예를 들어 `"rm foo*.bak"`을 실행하기 전에 먼저 `"ls foo*.bak"`을 써서 여러분이 지우려고 생각하는 파일들만을 지우게 되는지 확인해야 한다. 파괴적인 명령 대신 에코를 쓰는 것도 때로는 좋은 방법이다.
- 사용자들이 `rm` 명령어를 쓰면 컴퓨터가 문서 삭제 여부를 재확인 질문하도록, 여러분이 에일리어스 (alias)를 꾸며 주도록 하자.16.5 ([9. Alias])
- 특정한 작업을 하기 위해서만 루트가 되도록 하라. 어떤 일을 하는 방법을 알고 싶다면, 루트 자격으로 수행돼야만 하는 작업이 무엇인지 확신될 때까지 일반 사용자용 계정으로 돌아가도록 하라.
- 루트용의 코맨드 패스 (command path:)는 매우 중요하다. PATH 환경 변수 (PATH environment variable)이라고도 하는 코맨드 패스에는 쉘이 풀그림들을 찾아보는 디렉토리들이 적혀 있다. 루트용의 코맨드 패스를 가능한 짧게 줄이도록 하고, "현재의 디렉토리"를 뜻하는 "." 는 PATH에 **절대로 포함하지 않도록** 해야 한다. 덧붙여서, 이 패스에는 쓰기가 허락된 디렉토리 (writable directory)를 포함되면 안된다. 공격자가 이 디렉토리의 바이너리 이진 파일 을 변경하거나 새로운 이진 파일을 집어넣을 수도 있고 여러분이 이 것을 실행한 경우에는 공격자가 루트 권한을 침탈하게 되기 때문이다. 16.5 ([10. PATH])
- 루트의 권한을 가지고 컴퓨터를 사용할 때에는 (r-유틸리티라고도 불리는) `rlogin / rsh / rexec` 종류의 도구들을 사용하면 안된다. 이것들은 다수의 "공격과 침탈" 방법의 도구 대상이며, 루트가 쓰기에는 절대적으로 위험한 도구들이다. 절대로 루트용 `.rhosts` 파일을 만들지 말아야 한다.
- `/etc/security` 문서에는 루트가 접속할 수 있는 터미널들이 적혀 있다. (레드 햇 리눅스에는) 지역 가상 콘솔 (local virtual consoles: vtys) 만이 기본값으로 적혀져 있다. 다른 것들을 적지 않도록 조심해야 한다. 필요하다면 원격 콘솔에서 일반 사용자 계정으로 접속한 후에 (가능하다면 6.4 ([ssh])나 다른 암호화된 채널을 사용해서) `su`를 실행 해 들어올 수 있으므로, 루트로 직접 접속할 필요는 전혀 없다.
- 루트로서 작업을 할 때에는 언제나 느긋하고 신중하게 행동하라. 여러분이 루트의 자격으로 가지고 하는 행동과 작업은 많은 것들에게 영향을 준다. 충분히 생각한 후에 자판을 두드려라!

(믿을 만한) 누군가에게 수퍼유저 자격으로 접속할 권한을 허용해 주어야 할 절대 명확한 경우가 있을 때 도움이 되는 몇 가지 있다. `sudo`는 사용자가 자신의 패스워드를 써서 루트 권한으로 몇 가지 제한된 명령을 내릴 수 있도록 해준다. 예컨대 리눅스 시스템의 어떤 사용자가 `sudo`를 이용해서 – 루트로서의 별다른 특권을 갖는 일없이 – 시디롬이나 디스켓을 언마운트하거나 마운트하는 것이 가능하도록 할 수가 있다. `sudo`를 쓰면 – 누가 어떤 명령을, 무엇을 하기 위해 사용했는지 추적할 수 있는 – `sudo` 사용 시도와 성공에 대한 기록 등이 (16.4 (일지 문서)에) 적히게 된다. 그렇기 때문에 루트 접근권을 여러 사람들이 가지는 시스템에서는 변경 사항을 추적할 수 있으므로 `sudo`를 쓰도록 하는 것이 좋다.

비록 `sudo`가 특정 사용자에게 특정 작업을 할 수 있는 제한된 특권을 주지만, 이 또한 몇 가지 단점이 있다. 그렇기 때문에 이것은 – 서버를 리스타트하거나 새 사용자를 등록하는 등의 – 극히 제한된 작업을 하는 것에만 사용 되도록 구성되어야 한다. 예를 들면 침입자가 쉘 에스케이프 기회를 주는 풀그림들을 통해

서 `sudo`를 사용함으로서 루트 권한을 침탈할 수가 있다. 대부분의 에디터가 이러한 종류의 풀그림들에 포함된다. 또한, `/bin/cat` 등의 보잘것없는 풀그림을 써서도 – 파일을 겹쳐 쓰는 방법으로 – 루트를 침탈할 수 있다. `sudo`는 책임성을 밝히는 수단 정도로 생각하도록 하고, 루트 사용자를 보호하는 역할로는 기대하지 않도록 하자.

제 5 절 파일과 파일시스템 보안

시스템을 온라인으로 접속시키기 전에, 몇 분 동안이나마 준비와 계획을 하는 것은 여러분의 시스템과 데이타를 보호하는 것에 큰 도움을 준다.

- SUID/SGID를 사용자의 홈 디렉토리에서 쓰게 할 이유가 전혀 없다. 루트가 아닌 다른 사용자들이 자료를 쓸 수 있도록, 쓰기가 허락된 (writable로 되어 있는) 파티션에는 `/etc/fstab`에 `nosuid` 옵션을 적어 놓도록 한다. 이런 방법 등으로 어차피 필요 없어야 하는 – 풀그림의 실행을 금지하며, 블록 디바이스의 형성을 못하도록 – `/var`를 포함해서, 사용자의 홈 파티션에는 `nodev`와 `noexec`을 옵션으로 적어 놓도록 한다.

- 만약 NFS를 써서 파일시스템을 네트워크로 송출 (export)하는 경우라면, `/etc/exports`를 최대 한도로 제한하도록 조정하도록 한다. 이것은 와일드카드를 쓰지 않는 것과, 루트 쓰기 엑세스 (root write access)를 허락하지 않는 것과, 가능하면 읽기 전용의 파일시스템만을 송출하도록 하는 것을 의미한다.

- 사용자의 파일 생성 umask를 가능한 제한된 값으로 조정한다.5.1 ([umask 값])을 참조한다.

- NFS 등의 네트워크 파일시스템을 마운트한다면, /etc/exports를 조정해서 적절한 제한을 주도록 한다. 보통 'nodev'와 'nosuid'를 쓰는 것이 바람직하고, 'noexec'까지도 고려하는 것이 좋다.

- 기본값인 "무제한 (`unlimited`)"이 아닌 값으로 파일시스템의 기본값을 제한한다. 자원 제한 PAM 모듈과 `/etc/pam.d/limits.comf`를 사용함으로서 사용자 각각의 제한치를 조정한다. 예를 들면, `users` 그룹을 위한 제한은 다음과 같을 수 있다.

  ```
  @users     hard  core    0
  @users     hard  nproc   50
  @users     hard  rss     5000
  ```

  이 경우는 코어 파일의 생성을 금하며, 프로세스의 수를 50으로 제한하며, 사용자 한 사람 당 메모리 사용을 5 메가로 제한함을 말한다.

- `/var/log/wtmp`와 `/var/rin/utmp` 파일들은 시스템 모든 사용자의 접속 기록을 가지고 있다. 이들은 사용자가 (혹은 잠재적 침입자가) 언제, 어디서 시스템에 들어왔는가를 조사하기 위한 작업 사용되므로 이 파일들의 보안과 보전은 철저히 유지되어야만 된다. 일반적인 시스템 작동에 영향을 주는 경우가 없도록 함과 동시에 `644` 허가권을 가지고 있어야 한다.

- 보호되어야만 하는 파일들을 실수로 지우거나 덧쓰는 경우가 없도록 하기 위해서 이뮤타블 비트 (immutable bit: 불변의 비트)를 사용한다. 또한 이 방법은 파일에 – `/etc/passwd`나 `/etc/shadow`를 조작하는 방법의 일부가 되는, – 심볼릭 링크를 만드는 것을 방지한다. 이뮤타블 비트에 대한 추가 정보는 `chattr(1)`의 man 페이지를 참조하도록 할 것.

- SUID와 SGID는 잠재적인 보안 위험 요소이기 때문에 철저하게 감시되어야 만 한다. 이 풀그림들은 이 들을 사용하는 사용자들에게 특별 권한을 부여해 주기 때문에, 보안에 불안 요소를 주는 이러한 풀그림들이 설치되는 일이 없도록 해야 한다. 크랙커들이 좋아하는 트릭 중 의 하나는 SUID 루트 프로그램을 침탈하고 그 후에 – 원래의 문제점이 고쳐진 후에라도 – SUID 풀그림을 통해 뒷문의 개구멍으로 들어오는 것이다. 그러므로, 여러분 시스템에 있는 모든 SUDI/SGID를 찾아내서, 그것들이 무엇인지 추적함으로서 –잠재적인 침입자를 의미할 수 있는– 어떠한 변화라도 알 수 있도록 한다. 다음의 명령어를 사용하면 시스템에 있는 모든 SUID/SGID 풀그림을 찾아낼 수 있다.

```
root#  find / -type f \( -perm -04000 -o -perm -02000 \)
```

데비안 디스트리뷰션을 쓰면 어떤 SUID 문서가 존재하는지 매일 밤 확인하는 작업 (Job)을 실행할 수 있고. 그 결과를 전날 밤의 결과와 비교를 할 수가 있다. /var/log/suid*를 보면 이 작업의 일지를 볼 수 있다. 원한다면 의심스러운 SUID나 SGID 허가권을 가진 풀그림을 chmod를 써서 지우거나 바꿀 수 있을 것이다.

chmod를 사용하면 의심쩍은 풀그림의 SUID나 SGID 허가권을 제한적으로 지울 수 있고, 필요함이 나중에라도 확실하게 느껴진다면 다시 복구해 주면 된다.

- 만약 크랙커가 시스템의 사용권을 얻고 – 특히 시스템 파일 등의 – 월드-라이타블(World-writable) 파일들을 마음대로 변경할 수 있게 된다면 그야말로 이 것은 심각한 보안 개구멍이 존재하게 된 것이라고 할 수 있다. 덧붙이면 – 크랙커들이 마음대로 파일을 덧붙이거나 지울 수가 있게 되므로 – 월드-라이타블 디렉토리 또한 위험한 존재인 것이다. 월드-라이타블 파일 모두를 찾기 위해서는 다음의 명령어를 사용한다.

  ```
  root# find / -perm -2 -type l -ls
  ```

  그리고 이 파일들이 왜 "쓰기 가능 (라이타블)"으로 설정되어 있는지 반드시 파악하도록 한다. 정상적인 운영의 경우에 있어서, /dev의 일부와 심볼릭 링크를 포함한 여러 파일들은 월드-라이타블로 되어 있어야 할 것이다. (In the normal course of operation, several files will be writable, including some world-writable, including from /dev, and symbolic links. some from /dev, and symbolic links, thus the ! -type l which excludes these from the previous find command.)

- 주인이 없는 무소속의 파일들 또한 침입자가 시스템에 들어왔다는 징후일 수 있다. 주인이 없거나 그룹에 소속되어 있지 않은 파일들은 다음의 명령어를 쓰면 찾아낼 수 있다.

  ```
  root# find / -nouser -o -nogroup -print
  ```

- 리모트 호스트 (.rhosts) 파일들은 절대로 있으면 안되는 것이기 때문에, 이것들을 찾는 것은 시스템 관리자 임무의 일부가 되어야만 한다. 주지할 것은 크랙커가 여러분 네트워크에 침투하기 위해서는 단 한 개의 불안전한 계정이 필요할 뿐이라는 것이다. 시스템의 모든 리모트 호스트 파일들은 다음의 명령어로 찾을 수 있다.

  ```
  root# find /home -name .rhosts -print
  ```

- 마지막으로, 무턱대고 시스템 파일의 허가권을 바꾸지 말고, 어떤 파일이 무슨 작업을 하도록 되어 있는 가를 정확히 이해하도록 한다. 단순한 작동의 이유만으로 파일의 허가권을 바꾸는 일이 없도록 해야 한다. 허가권을 바꾸기 전에 파일이 왜 이러한 허가권을 가지고 있는지 알도록 해야 한다.ㅌㅊ

5.1 umask 조정

umask 명령어는 시스템 파일이 만들어질 때의 허가권 기본값을 정하기 위해서 사용된다. umask에는 정하려는 파일 모드의 십진 전수 (Octal Complement)를 사용한다. 만약 허가권 기본값을 정하지 않은 상태에서 파일이 형성된 게 된다면, 사용자가 모르는 사이에 허가권을 가지면 안되는 누군가에게 읽기 쓰기 허가권을 주게 될 수가 있다. 일반적으로 umask 값은 022 027, 그리고 (제일 제한적인) 077 등이 있다. umask는 일반적으로 /etc/profile에서 조정되고, 시스템의 모든 사용자에게 적용된다. 문서 생성 기본값 (File creation mask)은 7.7.7.에서 원하는 수를 빼면 나온다. 다시 설명하면, 7.7.7.로 umask값을 정해 준 경우에는 새로 만들어지는 모든 문서는 (소유자를 포함한) 모든 사용자들에 게 읽기, 쓰기, 실행권을 주지 않게 된다. umask값이 666이라면, 새로 만들어지는 모든 문서는 111의 (허가권의) 기본값을 가지게 된다. umask값을 033으로 정해 준 예를 들겠다. 16.5 ([11. 십진 전수])

```
# Set the user's default umask
umask 033
```

특히 루트의 umask 값은 077로 정해서 읽기, 쓰기, 실행을 – 루트가 직접 chmod를 써서 바꿔 주지 않는 한 – 다른 사용자가 못하도록 만드는 것이 좋다. 하여간 위의 예인 033인 경우, 새로 만들어지는 디렉토리들은 – 777에서 033을 뺀 – 744 허가권을 가질 것이다. 루트의 mask 값은 077이 되므로, 다른 사용자가 –chmod를 써서 뚜렷이 명시하며 바꿔 주지 않는 한 – 읽고 쓰고 실행할 수 없도록 만들어 주는 것이다 033 umask를 정해 놓은 후에 만들어지는 문서들은 644 허가권을 가지게 된다.

레드 햇을 쓴다면 – 레드 햇의 사용자의 그룹 ID 구성 방법 (User Private Group rules)을 따른다는 가정 하에 – `umask`는 `002`라도 좋다. 기본 구성은 한 그룹 당 한 사용자로 되어 있기 때문이다.

5.2 파일 허가권 (File Permissions)

시스템 관리를 할 권리가 없는 사용자나 그룹이 시스템 파일을 임의로 편집하는 일이 없도록 하는 것은 당연히 중요한 것이다.

유닉스는 파일과 파일에 대한 엑세스 관리를 owner, group, 그리고 other라는 세 가지 특성으로 구분한다. 언제나 정확히 하나의 소유자 (owner)가 존재하며, 그룹의 멤버 수는 일정하지 않으며, 나머지 사용자들은 other가 된다.

유닉스 허가권에 대한 간단한 설명:

소유권 (Ownership) - 어떤 사용자나 그룹이 노드와 상위 노드의 허가권에 대한 조정을 할 수 있는 권한을 말한다.

허가권 (permission) - 특정 종류의 엑세스가 가능하도록 정해 주거나 변경될 수 있는 비트다. 디렉토리에 대한 허가권은 파일에 대한 허가권과는 다른 의미를 가질 수가 있다.

읽기 허가권 (Read):

- 파일의 내용을 볼 수 있는 것이 가능하다.
- 디렉토리를 읽는 것이 가능하다.

쓰기 허가권 (Write):

- 파일에 만들거나 변경을 하는 것이 가능하다.
- 디렉토리에 있는 파일을 지우거나 이동하는 것이 가능하다.

실행 허가권(Execute):

- 이진 풀그림 (binary)이나 쉘 스크립트를 실행할 수 있다.
- 읽기 허가권이 있다면 디렉토리를 탐색하는 것이 가능하다.

문서 성질의 보존 (Save Text Attribute): (디렉토리의 경우)

"스틱키 비트 (sticky bit)"는 디렉토리에 적용될 경우에는 다른 뜻을 가지게 된다. 디렉토리에 스틱키 비트가 붙을 때에는 사용자는 – 설령 사용자가 디렉토리에 일반적인 쓰기 허가권이 있더라도 – 소유권이 있거나 확실하게 쓰기 허가권이 허락된 파일 만 지울 수 있게 된다. 이것은 `/tmp` 따위의 – 월드-라이타블이면서도 일반 사용자가 무조건 파일을 지우면 좋지 않을 – 디렉토리 등을 위해 쓰여진다. 스틱키 비트는 긴 디렉토리 리스팅 (ls -l)에서 `t`로 표시된다.

SUID의 성질 (파일의 경우)

이것은 파일의 set-user-id 허가권을 정의할 때 사용된다. 소유자 허가권에 set-user-id 엑세스 모드가 붙으면 –그리고 파일이 실행 가능한 파일이라면– 이 파일을 실행하는 프로세스는 프로세스를 만든 사용자가 사용할 수 있는 시스템 리소스를 쓸 수 있는 권한이 부여된다. 이것은 "버퍼 오버플로우 (buffer overflow: 이하 버퍼 범람)"을 사용하는 많은 침탈법의 재료로 쓰여진다.

SGID의 성질 (파일의 경우)

그룹 허가권에 붙은 경우에는 이 비트가 "set-group-id"를 관리하게 된다. 이것은 그룹이 영향을 받는다는 점을 제외한다면 SUID와 같은 역할을 하는 것이다. 영향을 받으려면 역시 파일은 실행 가능하도록 정의되어야 한다.

SGID 어트리뷰트 (디렉토리의 경우)

만약 SGID를 디렉토리에 사용하면 ("chmod g+s 디렉토리"를 씀), 그 디렉토리 안의 파일들은 디렉토리 소유 그룹의 값을 기본 그룹 값으로 가지게 된다.

여러분 - 파일의 소유자 (owner)

그룹 - 여러분이 가입되어 있는 그룹 (group)

나머지 모든 이 - 파일의 소유자나 파일을 소유한 그룹에 속하지 않은 나머지 사용자 (other)

파일의 보기:

```
-rw-r--r--  1 kevin  users         114 Aug 28  1997 .zlogin
    1번 비트  (-) 디렉토리인가?      (아니다)
    2번 비트  (r) 소유자에 읽기권?  (있다.  케빈이 읽을 수 있다)
    3번 비트  (w) 소유자가 쓰기권? (있다.  케빈이 읽을 수 있다)
    4번 비트  (-) 소유자에 실행권?  (없다)
    5번 비트  (r) 그룹에 읽기권?    (있다.  users라는 그룹)
    6번 비트  (-)  그룹에 쓰기권?   (없다)
    7번 비트  (-) 그룹에 실행권?    (없다)
    8번 비트  (r) 모든 이에 읽기권? (있다. 모든 이가 읽을 수 있다)
    9번 비트  (-) 모든 이 쓰기권?   (없다)
    10번 비트 (-) 모든 이에 실행권? (없다)
```

아래에는 필요한 만큼만의 최소한의 허가권을 부여한 보기를 적어 놓았다. 더 큰 허가권을 주는 것이 가능하지만, 설명된 작업 용도에 알맞은 최소 한도로 예를 설정해 놓은 것임을 밝혀 둔다.

```
-r--------  소유자의 읽기 허가권이 파일에 있다.
--w-------  소유자가 파일을 변경하거나 지울 수 있다.
---x------  소유자가 파일 (풀그림)을 실행할 수 있지만, 읽기권도 있어야
            실행되는 쉘 스크립트는 실행하지 못한다.
---s------  실세의 사용자 ID를 가진 개인이라면 실행할 수 있다.
            (setuid 참조)
-------s--  실세의 사용자 ID를 가진 그룹이라면 실행할 수 있다.
            (setgid 참조)
-rw------T  "최근 바뀐 시간 (last modified time)" 정보가 갱신되지 않는다.
            스왑 파일 등에 사용된다.
---t------  상관없음 (전에는 스틱키 비트였음)
```

디렉토리의 보기:

```
drwxr-xr-x  3 kevin  users         512 Sep 19 13:47 .public_html/
1번 비트  (d) 디렉토리인가?   (그렇다.  많은 파일을 가지고 있다)
2번 비트  (r) 소유자의 읽기권?  (있다. 케빈)
3번 비트  (w) 소유자의 쓰기권? (있다. 케빈)
4번 비트  (x) 소유자의 실행권? (있다. 케빈)
5번 비트  (r) 그룹의 읽기권?  (있다. users 그룹)
6번 비트  (-) 그룹의 쓰기권?  (없다)
7번 비트  (x) 그룹의 실행권?  (있다. users 그룹)
```

```
8번 비트 (r) 다른 이의 읽기권? (있다. 아무나 읽을 수 있다)
9번 비트 (-) 다른 이의 쓰기권? (없다)
10번 비트 (x) 다른 이의 실행권? (있다. 아무나 실행할 수 있다)
```

아래는 최소의 허가권을 준 사용 보기이다. 여기에 설명되어 있는 것 보다 허가권을 더 주는 것은 가능하지만, 아래에 설명하는 정도는 최소 한도로 필요하다.

```
dr--------  내용은 보여질 수 있지만, 파일 어트리뷰트는 읽을 수 없게 된다.
d--x------  디렉토리는 실행 패스 (path)에 넘어져서 사용될 수 있다.
dr-x------  파일 어트리뷰트는 이제 소유자에 의해서 읽혀질 수 있다.
d-wx------  디렉토리 현 위치에 있지 않아도 파일은 만들어지고
            지워질 수 있다.
d------x-t  쓰기 엑세스를 가진 다른 사용자들이 파일을 함부로 지우는 것을
            막는다. /tmp 디렉토리에 사용된다.
d---s--s--  아무런 작용을 하지 않는다. (SUID와 SGID 참조)
```

(보통 `/etc` 안에 있는) 시스템 설정 파일 (system configuration files)들은 640 (`-rw-r-----`) 모드이면서 동시에 루트 소유로 되어 있다.16.5 ([12]) 이것은 여러분 사이트의 보안 필요에 따라서 바꾸면 된다. 시스템 파일은 절대로 다른 어떤 그룹이나 누구라도 쓸 수 있도록 하면 안된다. `/etc/shadow`를 포함한 시스템 파일의 일부는 루트만이 읽기 허가권을 가져야 하고, `/etc` 안의 디렉토리들은 다른 이들이 읽지 못하도록 해야 한다.

SUID 쉘 스크립트:

SUID 쉘 스크립트는 심각한 보안 위험 요소이며, 그런 이유 때문에 커널이 받아들이지 않도록 되어 있다. 여러분이 얼마나 쉘 스크립트가 안전하다고 생각을 하던 간에, 이것은 크랙커에게 루 트 쉘을 주는 침탈 도구가 될 수 있다.

5.3 완결성의 검사

트립와이어 (`Tripwire`), 에이드 `Aide`, 오사이리스 (`Osiris`) 등의 완결성 (Integrity) 유지용 검사 도구를 사용하는 것은 지역 사용자가 펼치는 (그리고 네트워크를 통해서 들어오는) 공격을 탐지해 내는 매우 좋은 방법이다. 트립와이어, 에이드, 오사이리스 등은 중요한 이진 파일들과 설정 파일들의 첵섬 (checksum) 값을 검출해서 이전에 만들어 놓은 데이타베이스와 비교한다. 파일에 변화가 있으면 표시가 날 것이다. 이러한 풀그림을 쓰는 경우에는 플로피에 설치하고 쓰기 방지 탭을 사용해서 쓰는 것이 좋다. 이렇게 해 놓으면 침입자는 이러한 검사 풀그림에 손을 대거나 데이타베이스를 바꾸지 못하게 된다. 일단 한 번 설치했으면, 일상적인 보안 관리 임무의 일부분으로서 관례적, 주기적으로 실행하는 것이 좋다.

트립와이어 등의 검사 풀그림을 매일 밤 플로피에서 돌리고 아침에 메일로 결과를 받도록 다음과 같이 크론탭을 설정할 수 있다.

```
# set mailto
MAILTO=kevin
# run Tripwire
15 05 * * * root /usr/local/adm/tcheck/tripwire
```

이와 같이 하면 매일 아침 5:15am에 리포트를 보내 줄 것이다.

이러한 검사기는 여러분이 직접 침입자를 눈치채기 훨씬 전에 미리 자동으로 알려주는 귀중한 존재가 될 수 있다. 하지만 일반적 시스템 안에서는 많은 파일이 항시 바뀌므로 변한 것이 여러분의 일 때문인가, 아니면 크랙커의 행동인가를 파악하는 것에 신중을 기하도록 한다.

오픈 소스 버전으로 만들어진 **트립와이어** (`Tripwire`)는 *http://www.tripwire.org* `<http://www.tripwire.org>`에서 무료로 구할 수 있다. 매뉴얼과 고객 지원은 따로 구입해야 한다.

에이드(Aide)는 *http://www.cs.tut.fi/˜rammer/aide.html* <http://www.cs.tut.fi/˜rammer/aide.html>를 참조할 것.

오사이리스 (Osiris)는 *http://www.shmoo.com/osiris/* <http://www.shmoo.com/osiris/>를 참조할 것. 1086:

5.4 트로이의 목마

트로이의 목마는 호머의 일리어드에 나오는 전설적인 책략에서 비롯된 이름이다. 그럴듯해 보이는 어떤 풀그림이나 이진 파일을 업로드해 놓고, 다른 사람들이 그것을 다운 받아서 루트로서 돌리도록 한다. 그런 뒤에 사용자가 신경을 주지 않는 틈을 타서 시스템에 깨고 들어오는 것이다. 방금 받은 이진 파일이 관리자가 애초에 기대했던 어떤 일을 한다고 생각하는 사이에 –정말로 그런 척 하기도 한다– 한편으로는 보안을 깨고 들어오는 것이다.

여러분의 컴퓨터에 무슨 풀그림을 설치하였는지는 잘 살피도록 해야 한다. 레드 햇은 RPM 파일의 MD5 첵섬과 PGP 시그니춰를 제공하므로, 설치하고 있는 풀그림이 진짜인지 확인할 수 있다. 다른 배포본도 비슷한 방법을 쓴다. 소스도 있거나 잘 알려진 것이 아닌 한, 루트의 권한으로 어떤 이진 파일도 실행되어서는 안 된다! 일반 대중이 정밀한 조사를 할 수 있도록 소스를 공개할 크랙커는 없다시피 하므로.

귀찮을 수도 있지만, 풀그림의 소스를 정품 배포 장소에서 가져왔는지 확인하도록 하라. 풀그림이 루트에서 실행될 상황이라면 여러분이나 믿을 만한 누군가가 소스를 훑어보고 확인하도록 해야 한다.

제 6 절 패스워드 보안과 암호화 (Encryption)

암호는 오늘날 쓰이고 있는 가장 중요한 보안 기능 중 하나다. 탄탄하고 추측할 수 없는 패스워드를 갖는 것은 여러분에게나 여러분의 사용자들에게 중요한 일인 것이다. 요즘의 리눅스 배포본들은 쉽게 추측할 수 있는 패스워드는 설정할 수 없도록 관리하는 `passwd` 풀그림을 포함하고 있다. 여러분의 `passwd` 풀그림이 이런 특성을 가지고 있는 최신판인지 확인하도록 하라.

암호화에 대한 깊은 토론은 이 파일의 범위를 벗어 나는 것이지만, 소개 정도는 해야겠다는 생각이 든다. 암호화는 매우 유용하며, 요즘 같은 시대에는 필수적이기까지 하다. 자료를 암호화하는 방법에는 여러 방법이 있으며, 각각 나름대로의 특성이 있다.

대부분의 (리눅스를 포함한,) 유닉스 계열은 디이에스. (DES Data Encryption Standard 데이타 암호화 표준)라고 하는 단방향 암호화 연산법 (One-way Encryption Algorithm)을 사용해서 패스워드를 암호화한다. 이렇게 암호화된 패스워드는 (흔히) `/etc/passwd`, (쉐도우 패스워드를 쓴 경우에는) `/etc/shadow` 에 저장된다. 여러분이 로그인할 때 입력한 입력한 패스워드는 먼저 암호 처리가 된 후에, 이 처리된 값이 다시 `passwd` 문서에 저장되어 있는 패스워드 처리값과 비교가 되게끔 되어 있다. 둘이 일치하면 같은 패스워드임이 분명하므로 엑세스가 허가된다. 비록 디이에스는 (맞는 키가 사용되었다는 가정 하에서 – 같은 키로 암호화했다가 다시 복호화 하므로) 엄격히 따지면 양방향 암호화 방법 (Two-way Encryption Algorithm)이기는 하지만, 대부분의 유닉스 계열이 쓰는 변종의 디이에스는 단방향 식이다. 이것은 `etc/passwd` (혹은 `/etc/shadow`)안의 암호화된 값을 역산(逆算)해서 원래의 패스워드 값을 얻는 것이 가능하면 안된다는 뜻이다. 16.5 ([38. 단방향 연산])

여러분 패스워드의 랜덤 요소가 충분하지 않다면 크랙 (crack)이나 존 더 립퍼 (John the ripper) 같은 부루트 포스 공격에 (brute force attack) 의해 패스워드를 간파 당하는 경우가 있을 수 있다. (아래의 크랙 참조). 팸 (PAM) 모듈을 쓰면 패스워드에 (MD5 등의) 다른 암호화 방식도 쓸 수 있게 된다 (아래 참조). 또한 여러분에게 크랙이 득이 되도록 쓸 수도 있다. 여러분들이 가지고 있는 패스워드 데이타베이스에서 쉽게 깨질 수 있는 패스워드를 찾아내기 위해서 크랙을 써서 테스트해 낼 수도 있을 것이다. 이러한 약한 패스워드가 발견되면, 패스워드의 주인에게 이 사실을 알려줌과 동시에 패스워드를 쉽게 추측할 수 없도록 만드는 법을 알려주도록 하자.

좋은 패스워드를 고르는 방법에 대해서는 *http://consult.cern.ch/writeup/security/security_3.html* <http://consult.cern.ch/writeup/security/security_3.html>에서 정보를 얻을 수 있다.

6.1 6.1 피지피와 공개 열쇠 암호 기법 (Public Key Cryptography)

피지피 (PGP Pretty Good Privacy) 등에 사용되고 있는, 공개 열쇠 암호 기법은 하나의 열쇠로 암호화하고 또 다른 열쇠로 복호화하는 (두 개의 열쇠를 쓰는) 암호 기법을 쓴다. 전통적인 암호 기법은 동일한 하나의 열쇠로 암호화와 복호화를 둘 다 처리해 왔다. 이 (한 개뿐인) "비밀 열쇠"는 (암호화하는 쪽과 복호화하는 쪽의) 양편이 모두 가지고 있어야 했고, 무슨 수로든 보안을 유지하면서 한 쪽에서 다른 상대방으로 전달되었어야 했다.

이렇게 보안을 유지하면서 열쇠를 전달해 주어야만 되는 어려운 수고를 덜어 주기 위해서, 공개 열쇠 암호법은 두 개의 키를 사용한다. 각 개인의 공개 열쇠는 누구나 암호화에 쓸 수 있도록 배포되고 이에 상응하는 – 복호화에 사용될 – 개인의 비밀 열쇠는 개인이 보관한다.

공용 열쇠 암호 기법과 비밀 열쇠 암호 기법에는 각 장점이 있고, 차이점은 이 항목의 끝 부분에 적어 놓은 *RSA FAQ* <http://www.rsasecurity.com/rsalabs/faq/>를 읽어보기 바란다.

리눅스는 피지피를 잘 지원해 준다. 피지피 2.6.2와 5.0이 잘 작동된다고 알려져 있다. 피지피에 대한 기본 안내문과 사용법을 알고 싶으면 PGP FAQ를 읽기 바란다. *국제판 PGPi: http://www.pgpi.org/doc/faq/* <http://www.pgpi.org/doc/faq/>.16.5 ([13. 국제판 PGP FAQ])

미국 정부는 강력 암호 기법을 군용 무기로 취급하고 있고, 이에 따라서 PGP 등의 강력 암호 기법을 전자적 매체를 통해서 송출하는 것을 "수출 제한 조치"로 금하고 있으므로, 여러분 국가에 맞는 버전을 사용하도록 하라.16.5 ([14. 국제판 PGP]).

http://mercury.chem.pitt.edu/~tiho/LinuxFocus/English/November1997/article7.html <http://mercury.chem.pitt.edu/~tiho/LinuxFocus/English/November1997/article7.html>를 보면 리눅스에 피지피를 설치하는 자세한 설명서가 있다. 새로운 버전의 리눅스에는 패치를 구해서 붙여야 되는데, *ftp://metalab.unc.edu/pub/Linux/apps/crypto* <ftp://metalab.unc.edu/pub/Linux/apps/crypto>에서 구할 수 있다.

또한 피지피를 오픈 소스 형태의 무료판으로 재구성하는 계획이 진행되고 있다. 지엔유피지 (GnuPG)는 무료판 피지피의 완성본이다. 이 것은 IDEA나 RSA를 사용하지 않기 때문에 (수출 제한 조치에 걸리지 않고) 제한 없이 쓸 수 있다. 지엔유피지는 *OpenPGP* <http://www.faqs.org/rfcs/rfc2440.html> 규격에 거의 들어맞게 제작되어 있다. GNU 프라이버시 가드 웹 페이지에 가면 자세한 정보를 얻을 수 있다. *http://www.gnupg.org* <http://www.gnupg.org>

http://www.rsa.com/rsalabs/newfaq/ <http://www.rsa.com/rsalabs/newfaq/>에 있는 RSA FAQ에서 좀 더 정보를 얻을 수 있다. 여기에서 "디피-헬램 (Diffie-Hellamn)", "공용 열쇠 암호 기법 (public-key cryptography)", "전자 인증 (Digital Certificates)" 등의 용어에 대한 정보를 얻을 수 있을 것이다.

6.2 SSL, S-HTTP, HTTPS 그리고 S/MIME

- **SSL**:- SSL, 혹은 시큐어 소켓 레이어 (Secure Sockets Layer)는 인터넷 상에서의 보안을 위해서 넷스케이프 사에서 개발한 것이며 클라이언트/서버 인증용으로 쓰인다. SSL은 트랜스포트 레이어에서 작동되며 – 많은 종류의 데이타들을 사용자가 인지하지 못하는 배경 투명 작업으로 암호화하는 – 안전하며 암호화된 통신로(通信路: channel)를 만들어 준다. SSL의 예제는 넷스케이프 커뮤니케이터 (혹은 나비게이터)로 시큐어 사이트의 파일을 열어 볼 때 쉽게 볼 수 있으며 – 넷스케이프 사의 데이타 인크립션을 비롯한 – 커뮤니케이터를 이용한 보안 통신 (secure communication)의 기초로 쓰인다. *http://www.consensus.com/faqs/ssl-talk-faq.html* <http://www.consensus.com/faqs/ssl-talk-faq.html>에서 추가 정보를 얻을 수 있다. 넷스케이프 회사의 다른 종류의 보안 기술은 *http://home.netscape.com/security/index.html* <http://home.netscape.com/security/index.html>에 가면 있다. 16.5 ([39. 넷스케이프사의 정보])

- **S-HTTP**:- S-HTTP는 인터넷 상에서 보안을 담당하는 또 다른 종류의 프로토콜이다. 다중 열쇠 관리 기법 (multiple key management mechanisms)을 지원하며, 데이타를 주고받는 두 사람이 사용하는 암호 연산 기법 (cryptographic algorithm)의 일치를 옵션 교섭을 통해서 지원하는 동시에, 기밀성 (confidentiality), 인증 (authentication), 보전성 (integrity: 데이타의 무결성), 송신 사실 증명 기능 (non-repudiability) 등을 공급해 준다. S-HTTP는 사용 허가된 특별한 소프트웨어로 만 사용이 되도록 제한되어 있으며, 암호화될 대상 데이타를 부분 부분적으로 잘라서 (블록) 암호화해 준다.16.5 ([36])
- **S**MIME/:- S/MIME (Secure Multipurpose Internet Mail Extension)은 전자 우편과 인터넷 상의 메시지를 암호화하기 위한 인크립션 기준이다. RSA에서 개발한 공개 기준이니 만큼, 언젠가는 리눅스에서도 볼 수 있었으면 한다. S/MIME에 대한 추가 정보는 *http://www.rsasecurity.com/standards/smime/* `<http://www.rsasecurity.com/standards/smime/>`에서 구할 수 있다.16.5 ([15. 새로운 URL])

6.3 리눅스 IPSEC 기술법

CIPE를 포함한 여러 형식의 데이타 인크립션을 포함해서, 리눅스용의 IPSEC 사용 기술법에는 여러 가지가 있다. IPSEC은 IETF가 IP 네트워크 레벨 상에서 암호 기법 상으로 안전한 통신을 하기 위한 목적으로 만들었으며, 인증 (authentication), 보전성 (integrity), 엑세스 관리, 기밀성 등을 제공해 주는 제품이다. IPSEC에 대한 정보와 인터넷 드래프트 문서는 *http://www.ietf.org/html.charters/ipsec-charter.html* `<http://www.ietf.org/html.charters/ipsec-charter.html>`에서 구할 수 있다. 여기에서는 열쇠 관리 기법을 쓰는 다른 프로토콜에 대한 링크와 IPSEC 메일링 리스트, 그리고 메일링 리스트의 아카이브 등을 찾을 수 있다. 여기서는 열쇠 관리 (Key Management)에 관한 여러 프로토콜에 대한 링크와 IPSEC의 메일링 리스트와 아카이브를 볼 수 있다.

애리조나 대학에서 개발하고 있는 "x-커널의 리눅스용 구성본" (x-kernel Linux implementation)은 x-커널이라는 네트워크 프로토콜을 쓰는 오브젝트-베이스 프레임워크이고, *http://www.cs.arizona.edu/xkernel/hpcc-blue/linux.html* `<http://www.cs.arizona.edu/xkernel/hpcc-blue/linux.html>`에서 구할 수 있다. 가장 간단하게 설명을 하자면, x-커널은 커널 차원에서 메시지를 통과시키는 방법이다.

"Linux FreeS/WAN IPSEC"이라는 IPSEC 구성본의 무료 배포판도 있다. 제작자의 웹 페이지에 가보면

> "이러한 서비스는 신뢰할 수 없는 네트워크들 (untrusted networks) 상에서, 신뢰할 수 있는 터널 통로(secure tunnel)를 만들도록 해준다. 신뢰할 수 없는 네트워크를 지나게 되는 모든 통신은 IPSEC 게이트웨이 머신 (gateway machine)을 사용해서 암호화되어서 송신되고, 끝 부분의 수신 지점에서 다시 복호화되게 된다. 결과적으로 버츄얼 프라이비트 네트워크가 (Virtual Private Network: VPN - 가상사설망) 만들어지는 것이다. 비록 이 네트워크가 공개적인 인터넷으로 연결된 여러 사이트를 포함한다 해도, 실질적으로는 통신 보안이 되는 네트워크가 구성되는 것이다."

라고 적혀 있다.

이 풀그림은 *http://www.xs4all.nl/˜freeswan/* `<http://www.xs4all.nl/~freeswan/>`에서 다운로드 받을 수 있고, 이 문서가 만들어 질 당시에 이미 1.0 버전이 만들어져 나와 있다.

다른 류의 암호화 기법은 – 수출 제한 조치 때문에 – 기본적으로 배포본에 포함되지 않는다.

6.4 시큐어 쉘 ssh와 스텔넷 (Stelnet)

ssh와 스텔넷 (Stelnet)은 원격 시스템으로 접속을 하고, 암호화된 커넥션을 유지하기 위한 풀그림 뭉치다.

openssh는 rlogin, rsh, 그리고 rcp의 대체용으로, 이 셋 보다 더 안정적인 풀그림들의 뭉치다. SSH는 두 호스트간의 통신 암호화와 사용자 인증을 위해서 공개 열쇠 암호 기법을 사용한다. 세션 하이젝킹(Session Hijacking)과 DNS 스푸핑을 방지해 주면서, 원격 호스트에 로그인하거나, 호스트끼리 데이타를 복사하기

위해 사용될 수 있다.16.5 ([17. 중계인 공격]) 송수신 시의 데이타 컴프레션을 실행하며, 호스트간의 X11 통신의 통신 보안을 실행해 준다.

이제는 ssh 구성본이 여러 가지 만들어져 있다. 데이터 펠로우스에서 만든 원래의 상업용 구성본은 *http://www.datafellows.com* <http://www.datafellows.com>에서 구할 수 있다.

성능이 뛰어난 openssh는 데이터 펠로우스사의 초기 구성본에 기초를 두었으며 특허권이나 각 회사 전용의 소스를 전혀 사용하지 않도록 완전히 재구성되어 있다. 무료이며 BSD 사용권 (BSD License)에 기초를 두고 배포 사용된다. 이 것은 *http://www.openssh.com* <http://www.openssh.com>에서 구할 수 있다.

ssh를 기초부터 다시 오픈 소스로 구성한 "psst..."도 있다. 자세한 정보는 *http://www.net.lut.ac.uk/psst/* <http://www.net.lut.ac.uk/psst/>에서 구할 수 있다. 16.5 ([18. psst])

윈도우스 워크스테이션 SSH에서 리눅스 SSH로 연결할 수도 있다. 윈도우스 클라이언트용으로 만든 무료 제품이 많은데, *http://guardian.htu.tuwien.ac.at/therapy/ssh/* <http://guardian.htu.tuwien.ac.at/therapy/ssh/> 등이고, 데이타펠로우스 사(社)에서 만드는 유료 제품은 *http://www.datafellows.com/* <http://www.datafellows.com/>에 있다.

SSLeay는 넷스케이프사의 SSL를 무료판으로 구성한 것으로 시큐어 텔넷, 아파치 모듈, 여러 가지 데이타베이스, DES와 IDEA 그리고 불로우피쉬 (Blowfish) 등의 여러 종류 알고리듬을 포함한다.

SSLeasy는 에릭 영 (Eric Young)이 개발한 것으로, 넷스케이프사의 시큐어 소켓 레이어 프로토콜 (Secure Sockets Layer Protocol)의 작동을 무료 구성판으로 만든 것이다. 이 것에는 시큐어 텔넷, 아파치용 모듈, 여러 데이타베이스, 디이에스와 IDEA 불로우피쉬 등을 포함한 알고리듬 등의 포함되어 있다.

텔넷 연결 시에 암호화를 할 수 있는 시큐어 텔넷의 교체품이 이 라이브러리를 써서 만들어져 있다. 스텔넷(Stelnet)은 SSH와는 달리 넷스케이프가 만든 SSL (Secure Sockets Layer)를 사용한다. *http://www.psy.uq.oz.au/˜ftp/Crypto/* <http://www.psy.uq.oz.au/˜ftp/Crypto/>에 있는 SSLeay FAQ를 읽어보면 시큐어 텔넷과 시큐어 FTP에 대한 것을 찾을 수 있다.

SRP는 (Secure Remote Password Protocol) 또 다른 텔넷/ftp 통신 보안용 구성의 하나이다. 제작자의 웹 페이지에 가보면 다음과 같은 설명을 하고 있다.

> "SRP 프로젝트는 통신 보안 목적의 인터넷 풀그림을 무료로 전세계에 배포하는 것이 목적으로 개발되고 있다. 완전한 통신 보안이 되는 텔넷과 FTP 디스트리뷰션을 시작점으로 해 서, (현재의) 빈약한 네트워크 상의 인증 시스템을 사용자가 편하게 쓸 수 있는 강력한 것으로 교체하고자 한다. 보안은 선택적으로 제공되어서는 안되며, 당연히 기본적으로 제공되어야만 하는 것이다."

자세한 정보는 *http://srp.stanford.edu/srp* <http://srp.stanford.edu/srp>에서 구할 수 있다.

6.5 PAM (팸) - 장착 방식 인증 모듈 (Pluggable Authentication Modules)

새로운 버전의 레드 햇에는 "PAM"이라는 통일된 인증 방식이 들어 있다. PAM은 – 사용자 여러분이 이진 파일을 다시 컴파일할 필요가 없이 – 인증법, 제한 사항, 지역 인증법을 쉽게 인캡슐레이션 해 준다. PAM의 인캡슐레이션 처리 방법은 이 파일의 내용 밖의 문제이지만, PAM의 웹 사이트에 가서 꼭 보기를 권한다. *http://www.kernel.org/pub/linux/libs/pam/index.html* <http://www.kernel.org/pub/linux/libs/pam/index.html>

PAM으로 할 수 있는 일 가운데 몇 가지 만 들어보면 아래와 같다.

- 패스워드에 비 (非) DES 암호화 방법을 쓴다. (패스워드를 부루트 포스 공격을 써서 풀어내는 것이 어렵게 된다)
- 사용자들이 쓸 수 있는 (프로세스 수, 메모리의 양 등의) 자원을 제한하는 방법을 써서 서비스 거부식 공격 (Denial of Service: 이하 DoS)을 못하도록 한다.

- 패스워드를 쉐도우 패스워드로 감추는 것을 쉽게 할 수 있도록 한다.6.8 ((참조: 쉐도우 패스워드)
- 특정한 사용자가 특정한 시간에 특정한 장소에서만 로그인할 수 있도록 제한 조정하는 것이 가능하다.

시스템을 설치하고 조정하기 시작한 지 몇 시간 안으로, 공격 시도 시점에서 막을 수 있다. 예를 들면, `.rhosts` 파일을 시스템 전체용으로 사용자 홈 디렉토리에서 사용하는 것을 막기 위해서 다음을 `/etc/pam.d/rlogin`에 PAM을 사용해서 넣을 수 있다.

```
#
# Disable rsh/rlogin/rexec for users
#
login auth required pam_rhosts_auth.so no_rhosts
```

6.6 암호 기술이 적용된 IP 인캡슐레이션 (Cryptographic IP Encapsulation :CIPE)

이 소프트웨어의 일차적 목적은 – 인터넷 등의 – 개방형 패킷 네트워크를 가로 질러가는 서브네트워크를 (가짜 메시지 주입, 트래픽 분석 등의 행위로부터) 보호하기 위한 방법을 제공하는 것이다.

CIPE는 데이타를 네트워크 수준에서 암호화한다. 네트워크의 호스트 사이에서 돌아다니는 패킷이 암호화된다. 암호화 엔진은 패킷들을 주고받는 드라이버 근처에 위치한다.

이것은 –소켓 수준에서 데이타를 연결함으로 암호화를 하는– SSH와는 다른 것이다. CIPE는 –가상사설망 구성하기 위해서– 터널링에 사용될 수 있다. 아래 수준 (Low-level)에서의 암호화는 – 애플리케이션 소프트웨어를 수정할 필요가 없이 – VPN에 연결되어 있는 두 네트워크 사이에서 투명하게 작동되도록 만들어 질 수 있는 이점이 있다.

CIPE 파일에서 요약해 왔음:

> IPSEC 기준은 (다른 일도 하지만) 암호화된 VPN을 만들기 위해서 사용될 수 있는 프로토콜의 집합을 정의한다. 반대적으로, 많은 옵션을 가지고 있는 IPSEC은 상대적으로는 헤비급이면서 복잡하며, 주어진 프로토콜 전부를 사용하는 경우는 아직은 드물면서도, (열쇠 관리 등의) 몇 문제는 아직 완벽히 해결되어 있지 않다. CIPE는 좀 더 간단한 방식을 사용하는데, 초기 설정 시에 매개 변수 형식으로 (정말로 사용하고자 하는 인크립션 방식을 선택하는 등) 많 은 조건에 대한 정해진 선택을 할 수 있다. 이것은 탄력적인 운영을 제한하기는 하지만, 간단한 (그리고, 이 이유로, 쉽게 디버그를 할 수 있는 등으로) 능률적인 설정을 가능하게 해 준다.

정보를 더 원하면 다음을 참조할 것. *http://www.inka.de/˜bigred/devel/cipe.html* `<http://www.inka.de/~bigred/devel/cipe.html>`

다른 크립토그라피의 경우와 마찬가지로 이것도, 수출 제한 조치 때문에, 커널과 함께 배포되지 않는다.16.5 ([19. CIPE 구하기])

6.7 커브로스 (Kerberos)

커브로스는 MIT의 아테나 프로젝트 아래에서 개발된 인증 방식이다. 사용자가 접속해 들어오면, 커브로스는 (패스워드를 사용해서) 사용자를 인증하고, 네트워크 상에 흩어져 존재하는 서버와 호스트들에게 이 사용자의 신분을 증명해 주는 방법을 제공한다.

이 인증법은 리모트 로그인 (`rhost`) 풀그림 등에 의해서 패스워드 없이 사용자가 다른 호스트로 (`.rhost` 파일을 대신해서) 접속을 할 수 있도록 해 준다. 이 인증법은 또한, 보내는 사람 (발송인)이 가짜가 아닌 것을 보증하는 동시에, 메일이 정확한 사람 (수취인)에게 전달이 되도록 보증하기 위해서, 메일 시스템에 의해 사용될 수도 있다.

커브로스와 딸려 있는 많은 풀그림을 사용하는 궁극적인 효과는, 사용자가 시스템을 속여서 다른 사람인 척 "스푸핑"을 할 수 있는 능력을 거의 없애 버리는 데 있다.

커브로스에 대한 추가 정보는 *http://www.cis.ohio-state.edu/hypertext/faq/usenet/kerberos-faq/general/faq.html* <http://www.cis.ohio-state.edu/hypertext/faq/usenet/kerberos-faq/general/faq.html>에서 찾을 수 있고, 코드는 *http://nii.isi.edu/info/kerberos/* <http://nii.isi.edu/info/kerberos/>에 있다.

[인용: 제니퍼 지 스타인, 클리포드 뉴먼, 제프리 엘 쉴러 공저, "커브로스: 오픈 네트워크 시스템용 인증 서비스", 1998년 겨울 미국 텍사스의 달라스에서 열린 유스닉스 발표회 회보, (Stein, Jennifer G., Clifford Neuman, and Jeffrey L. Schiller, "Kerberos: An Authentication Service for Open Network Systems." USENIX Conference Proceedings, Dallas, Texas, Winter 1998)]

커브로스를 호스트의 보안의 정도를 높이가 위한 첫 방법으로 쓰지는 말아야 한다. 이 것은 매우 구성하기 힘들고, SSH처럼 광범위하게 사용되지는 않고 있다.

6.8 쉐도우 패스워드

쉐도우 패스워드는 암호화되어 있는 패스워드 정보를 일반 사용자들로부터 비밀로 유지하기 위한 한 가지 방법이다. 최근에 나온 데비안은 쉐도우 패스워드를 기본적으로 사용하도록 되어 있으며, 다른 리눅스 구성본은 암호화된 패스워드를 /etc/passwd 파일에 누구나 읽을 수 있을 수 있도록 저장한다. 누구라도 이 파일을 패스워드를 추측해 내는 풀그림에 돌려서 패스워드를 알아내려고 할 수 있다. 반면에 쉐도우 패스워드는 특별 권한이 있는 사용자들만 읽을 수 있도록 패스워드에 대한 정보를 /etc/shadow 파일에 저장한다. 쉐도우 패스워드를 사용하려면, 패스워드 정보를 읽어야 하는 모든 유틸리티들이 쉐도우 패스워드를 지원하도록 제대로 리컴파일되었는지 확인해야 한다. 반면에 (위의 설명한) PAM은 실행 풀그림들을 리컴파일할 필요 없이 단지 쉐도우 모듈을 장착시킴으로써 쉐도우 패스워드를 쓸 수 있도록 해준다. 필요하다면 Shadow-Password-HOWTO 파일을 읽으면 된다. 이것은 *http://metalab.unc.edu/LDP/HOWTO/Shadow-Password-HOWTO.html* <http://metalab.unc.edu/LDP/HOWTO/Shadow-Password-HOWTO.html>인데, 지금은 약간 낡았고, PAM을 지원하는 배포본에는 필요가 없다.

6.9 크랙(Crack)과 존 더 립퍼 (John the Ripper)

Passwd 풀그림을 실행할 때, "쉽게 추측할 수 없도록 만든다"는 패스워드 규칙을 어떤 이유가 있어서 실행하지 못하게 된 상황이라면, 여러분 스스로가 패스워드 격파 풀그림을 실행시켜서 실제의 사용자들이 안전한 패스워드를 쓰고 있는지 확인하는 것도 좋은 것이다.

패스워드 격파 풀그림은 간단한 방식으로 작동한다: 사전에 있는 모든 단어와 그 변화형을 패스워드로 시도해 하고, 단어 하나 하나를 암호화하면서 이미 암호화된 패스워드와 비교하는 것이다. 만약에 일치하는 단어를 찾게 되면, 암호를 알아낸 것이다.

원한다면 많은 패스워드 크랙 풀그림들을 구할 수 있을 것이다. 그 중에서 알아두면 좋은 두 개가 바로 "크랙"과 "존 더 립퍼", *http://www.openwall.com/john* <http://www.openwall.com/john/>다. CPU 자원을 엄청나게 소비할 것이지만, 이 풀그림을 여러분이 먼저 사용해 봄으로서 혹시나 공격자가 이런 풀그림을 사용해서 여러분의 시스템에 침입할 가능성이 있는지를 알아보는 동시에, 약한 패스워드를 쓰는 사용자들을 찾아내서 미리 알려줄 수가 있을 것이다. 공격자가 여러분의 passwd (유닉스에서는 /etc/passwd) 파일을 얻으려면 우선은 다른 개구멍을 이용해 먼저 들어와 있어야 하겠지만, 이런 개구멍 허점들이 여러분이 생각하는 것보다 훨씬 흔하다는 점 (즉, passwd가 저장된 파일을 빼내는 것이 어렵지 않다는 점)에 주의해야 한다.

전반적인 네트워크 보안에 관심을 두어야 하는 만큼, 여러분의 네트워크에 마이크로소프트 윈도우스를 쓰는 컴퓨터가 있다면, 윈도우스용으로 제작된 크랙 풀그림인 로프트크랙 (L0phtCrack)도 한 번 보도록 하자. 이것은 *http://www.l0pht.com/* <http://www.l0pht.com/>에서 구할 수 있다.

6.10 CFS와 TCFS - 암호화 파일 시스템과 투명 암호화 파일 시스템

CSF는 디렉토리 전체를 암호화하고, 사용자들이 문서를 암호화해서 저장할 수 있도록 하는 방법의 하나이다. 이것은 NFS 서버를 지역 컴퓨터에서 작동하는 방식으로 실행한다. RPM을 *http://www.zedz.net/redhat/* `<http://www.zedz.net/redhat/>`에서 구할 수 있고, 작동 방식에 대한 정보는 *ftp://ftp.research.att.com/dist/mab/* `<ftp://ftp.research.att.com/dist/mab/>`에 더 있다.

TCSF는 CFS보다 좀 더 완성도를 높여서 (암호화/복호화 작업을 백그라운드에서 투명하게 실행함으로서) 암호화된 파일시스템을 쓰고 있는 사용자 입장에서는 암호화/복호화 작업이 눈에 보이지 않도록 한 것이다. *http://edu-gw.dia.unisa.it/tcfs/* `<http://edu-gw.dia.unisa.it/tcfs/>`에서 정보를 구할 수 있다.

파일시스템 전체에 사용하지 않을 수도 있다. 부분적으로 디렉토리 트리를 암호화하는 것에만 쓰일 수도 있는 것이다.16.5 ([20. 마이크로소프트 OS를 동시에 쓰는 경우])

6.11 X11, SVGA와 디스플레이 보안

6.11.1 X11

디스플레이의 보안은 중요하다. 입력되는 패스워드를 공격자가 가로채거나, 여러분이 모니터로 읽고 있는 문서나 정보를 보거나, 심지어는 수퍼유저의 권한을 얻기 위해 보안 개구멍을 이용하기까지 하는 일들을 막기 위해서다. 도청자(sniffer)들이 여러분과 원격 시스템 사이의 상호작용을 모두 볼 수 있도록 허락하는 셈이라 할 수 있는, 네트워크 상에서의 원격 X 응용 풀그림 수행도 위험 천만한 일이다.

X는 많은 엑세스 통제 장치를 가지고 있다. 가장 간단한 것은 호스트에 기초를 두는 것이다. 여러분의 디스플레이어 접근할 수 있는 호스트들을 `xhost`를 사용해서 지정할 수 있다. 안전한 방법은 아니다. 누군가가 여러분의 컴퓨터에 이미 근접할 수 있다면, 그는 `xhost +그들의 컴퓨터`라는 명령어를 사용해서 쉽게 숨어 들어올 수 있다. 아울러 신임되지 않은 기계 (untrusted machine)의 접속을 허락하면, 그쪽의 누구라도 여러분의 디스플레이를 침탈할 수 있다.

로그인을 위해서 `xdm`을 (xdm: X 디스플레이 매니저, x display manager) 사용하고 있다면, 더 나은 엑세스 방법인 MIT-MAGIC-COOKIE-1을 구해서 사용하는 것을 고려해 보도록 하라. 128 비트 짜리 "쿠키 (cookie)"가 만들어져서 `.Xauthority` 문서에 저장된다. 원격 컴퓨터에서 여러분의 디스플레이에 접근하는 것을 허용할 필요가 있다면, 그 컴퓨터로부터의 접근만을 제공하기 위해 `xauth` 명령과 여러분의 `.Xauthority` 파일에 들어 있는 정보를 적용할 수 있다. *http://metalab.unc.edu/LDP/HOWTO/mini/Remote-X-Apps.html* `<http://metalab.unc.edu/LDP/HOWTO/mini/Remote-X-Apps.html>`에 있는 Remote-X-Apps mini-howto를 보도록 하라.

보안 유지되는 X의 접속을 만들기 위해서 `ssh`를 쓸 수 있다 (위의 6.4 ([ssh])를 참조할 것). 앤드 유저의 시점에서는 투명하게 작동되면서도, 암호화되지 않은 자료가 네트워크 상에 떠다니지 않도록 하는 방법이 되는 장점이 있다.

X 보안에 대해 더 많은 정보가 필요하면 `Xsecurity`의 맨 페이지 (man)를 보기 바란다. 보다 안전한 방법은 `xdm`을 써서 콘솔에 로그인 하도록 하고, ssh를 써서 X 풀그림을 원격 수행하려는 원격 사이트들로 가는 것이다.

6.11.2 SVGA

SVGAlib 풀그림들은 리눅스 컴퓨터에 있는 모든 비디오 하드웨어에 접근할 수 있도록 이것의 SUID가 root로 정해져 있다. 이것은 매우 위험한 것이다. 만일 이 풀그림들이 깨지면, 쓸 수 있는 콘솔을 살리기 위해서 다시 부팅 시켜야 한다. 여러분이 실행시키고 있는 SVGA 풀그림들이 진품인지, 그리고 최소 수준이나마 믿을 수 있는 것들인지 확인하라. 더 나은 방법은 SVGA 풀그림들을 아예 수행시키지 않는 것이다.

6.11.3 GGI (Generic Graphics Interface project)

리눅스 GGI 계획은 여러 가지의 리눅스 비디오 인터페이스 문제들을 해결하고자 노력하고 있다.. GGI는 비디오 코드 일부분을 리눅스 커널 안으로 옮겨 실행하는 방식으로 비디오 시스템에 대한 엑세스를 관리할 것이다. 이것은 GGI가 – 정해 놓은 양호 상태로 – 언제라도 여러분의 콘솔을 복구해 줄 수 있다는 것을 의미한다. 또한 여러분 콘솔에서 트로이 목마식의 로그인 풀그림이 돌지 않도록 하기 위해서, 보안 경계 열쇠(관리)도 허락될 것이다. *http://synergy.caltech.edu/˜ggi/* `<http://synergy.caltech.edu/~ggi/>`

제 7 절 커널 보안

이것은 보안에 관련된 커널 조정 옵션들과 이 것들이 무엇을 하는 지에 대한 설명과 어떻게 사용하는 지에 대한 설명이다.

커널이 여러분 컴퓨터 네트워크 업무를 관리하므로, 커널을 매우 안전하도록 관리하는 것과 커널 자체의 보안이 깨지지 않도록 하는 것은 중요한 것이다. 최신 네트워킹 공격법의 일부를 방지하기 위해서는 커널의 버전을 최신의 것으로 관리해야 한다. 새로운 커널은 *ftp://ftp.kernel.org/* `<ftp://ftp.kernel.org/>`이나 디스트리뷰션 제작자의 사이트에서 찾을 수 있다.

또한 주로 쓰이는 리눅스 커널에 쓸 수 있도록 통일된 암호화 패치를 제공하는 국제 그룹도 있다. 이 패치는 (미국의) 수출 제한 조치 때문에 리눅스 커널에 끼워 넣을 수 없었던 여러 가지 암호화 서브시스템을 지원해 준다. 정보는 *http://www.kerneli.org* `<http://www.kerneli.org>`에서 구할 수 있다.

7.1 "2.0 커널"의 컴파일 옵션

2.0.x 커널에서는 다음의 옵션들을 쓸 수 있고, 커널 구성 단계 (Kernel Configuration Process)에서 볼 수가 있을 것이다. 아래에 설명되어 있는 대부분은 `./linux/Documentation/Configure.help` 문서에 적혀 있고, 이 문서는 커널 컴파일 옵션을 실행할 때 `make config` 단계에서 도움말로 사용되고 있다.

- 네트워크 방화벽 (CONFIG_FIREWALL) 방화벽이나 마스커레이딩을 쓰려면 이 옵션은 켜져 있어야 한다. 만약 단순히 클라이언트로 컴퓨터를 사용하는 경우는 이 옵션을 끄는 것이 좋을 것이다.
- IP: 포워딩/게이트웨이 (CONFIG_IP_FORWARD) 만약 IP 포워딩을 사용한다면, 여러분의 리눅스 박스는 사실상으로 라우터의 역할을 하게 되는 것이다. 만약 사용하는 컴퓨터가 네트워크에 연결되어 있고 데이타를 어떤 네트워크에서 다른 네트워크로 포워딩 (Forwarding 전달)하는 경우라면, 애써서 세운 방화벽을 깨 버리는 결과를 만들 수가 있다. 보통의 다이알업 사용자는 이 기능을 꺼 버렸으면 할 것이지만, 다른 사용자들 은 이 상황에서 생기는 보안 문제의 발생에 주의를 기울일 필요가 생기게 된다. 반면에 방화벽 컴퓨터 자체는 이 기능을 켜서 사용하기를 바랄 것이고, 주로 사용하고 있는 방화벽 풀그림과 섞여서 사용되게 된다.16.5 ([21. KLDP 문서])

  다음의 명령을 써서 다이내믹 IP 포워딩을 실행할 수 있고,

  ```
  root#  echo 1 > /proc/sys/net/ipv4/ip_forward
  ```

  다음의 명령으로 끌 수 있다.

  ```
  root#  echo 0 > /proc/sys/net/ipv4/ip_forward
  ```

  염두에 둘 것은 /proc안에 있는 파일은 가상의 버츄얼 파일이기 때문에, 파일의 실제 크기가 제대로 신고되지 않을 때가 있다.
- IP: syn 쿠키 (CONFIG_SYN_COOKIES) SYN 공격은 모든 사용 가능한 자원을 소비하게 한다는 식의 방법으로, 결과적으로 리부팅을 하게 만드는, 서비스 거부 공격법 (DoS, Denial of Service)의 하나이다. 켜 놓지 않을 이유가 없는 옵션이다. 2.1 커널 시리즈에서는 이 옵션을 켜면 단순히 신 쿠키의 존재를 허용할 뿐, 사용을 허락하지는 않게 된다. 옵션을 켜기 위해서는 다음의 명령어를 쓴다.

```
root# echo 1 > /proc/sys/net/ipv4/tcp_syncookies
```

- IP: 방화벽 처리 (CONFIG_IP_FIREWALL) 만약 여러분의 기계를 방화벽으로 사용하는 경우, 마스커레이딩을 사용하는 경우, 아니면 여러분의 다이얼-업 웍크스테이션에 누군가가 PPP 다이얼-업 인터페이스를 통해서 들어오는 것을 막기 위한 경우에 필요한 옵션이다.

- IP: 방화벽 패켓의 일지 쓰기 (CONFIG_IP_FIREWALL_VERBOSE) 이 옵션은 –발신인, 수신인, 포트 번호 등의– 방화벽이 받은 패켓의 정보를 보여준다.

- IP: 소스에서 라우트된 프레임 떨구기 (CONFIG_IP_NOSR) 이 옵션은 켜져야 한다. 소스에서 라우트돼서 오는 프레임들은 (Source routed frames) 그 패켓 안에 목적지로 가는데 필요한 모든 진로(Path) 정보를 가지고 있고, 이러한 종류의 패켓을 라우터가 받게 되면 라우터는 내용을 검사하지 않고 무조건 통과 처리를 한다. 이러한 상황은 잠재적으로 침탈 수단이 되는 데이타가 여러분 시스템에 들어오는 것으로 발전될 수 있다.

- IP: 마스커레이딩 (CONFIG_IP_MASQUERADE) 여러분의 컴퓨터가 방화벽으로 사용되고 있고 이 방화벽이 보호하고 있는 네트워크 상의 어떤 컴퓨터가 네트워크 바깥의 호스트로 데이타를 보낼 때, 이 방화벽 컴퓨터는 이 호스트인 것처럼 "마스커레이드 (Masquerade, 가장)"하도록 할 수 있다. 보기를 들면, 마스커레이딩을 실행하는 컴퓨터가 데이타를 정해진 목적지로 주고 받으면서도 마치 이런 데이타가 방화벽 컴퓨터에서 발송되는 것인 척 꾸미게 되는 것이다. 이에 대한 정보는 *http://www.indyramp.com/masq* `<http://www.indyramp.com/masq>`에서 구할 수 있다.16.5 ([22. KLDP 문서])

- IP: ICMP 마스커레이딩 (CONFIG_IP_MASQUERADE_ICMP) 위의 마스커레이딩 옵션은 TCP나 UDP 정보의 경우만을 마스커레이딩 해준다. ICMP 마스커레이딩 옵션을 켜면 ICMP 마스커레이딩까지 덧붙여 하게 된다.

- IP: 투명 프락시의 작동 (CONFIG_IP_TRANSPARENT_PROXY) 이 옵션은 여러분의 리눅스 방화벽이 지역 호스트에서 발생되는 네트워크 트래픽과, 지역 호스트에서 외부의 원격 호스트로 보내어지는 네트워크 트래픽을 "트렌스페어런트 프락시 서버"라는 지역 서버로 (투명하게) 보내어 주게 한다. 이 옵션을 쓰면 지역 컴퓨터들을 , 사실상으로는 지역의 프락시로 연결이 되는 것이지만, 마치 외부의 리모트 컴퓨터로 연결이 되는 것처럼 생각하게 만든다. IP-마스커레이드 하우투와 *http://www.indyramp.com/masq* `<http://www.indyramp.com/masq>`를 보면 더 자세한 정보를 얻을 수 있다.

- IP: 데이타를 항상 뭉치로 전송 (CONFIG_IP_ALWAYS_DEFRAG) 이 옵션은 보통 꺼져 있다. 하지만, 만약 여러분이 방화벽이나 마스커레이딩 호스트를 만든다면 이 옵션을 켜 놓는 것이 좋다. 데이타가 한 호스트에서 다른 호스트로 전송될 때, 데이타는 항상 한 뭉치로 보내어 지는 것이 아니라, 대개는 여러 조각으로 쪼개어져서 보내지게 된다. 쪼개어져서 보내어지는 경우의 문제점은 포트 번호가 제일 처음 보내어지는 조각에 만 적혀진다는 것이다. 이 것은 누군가가 나머지 데이타 조각들을 개조해서 그가 원하는 정보를 집어넣을 수가 있다는 뜻이 된다. 또한 이 옵션을 쓰면, 아직 티어드랍 공격에 대한 패치를 붙여 놓지 않은 내부 호스트를 쓸 때 티어드랍 공격이 실행되는 것을 방지할 수 있게 된다.

- 패켓 사인 인증 (CONFIG_NCPFS_PACKET_SIGNING) 이 것은 2.2.x 커널에서 사용할 수 있는 옵션으로, 보다 강한 보안을 위해서 NCP 패켓을 인증 (sign)할 때 사용된다. 보통은 이 옵션이 꺼져 있지만, 여러분이 필요하다면 쓸 수 있다는 것을 염두에 두시기 바란다.

- IP: 방화벽 패켓 넷링크 디바이스 (CONFIG_IP_FIREWALL_NETLINK) 사용자 공간 풀그림 (User-space program) 패켓의 첫 128 바이트를 분석해서 – 이것의 합법성을 분석하는 방법으로 – 패켓을 받거나 혹은 거부해야 하는 지를 결정하게 해주는 깔끔한 옵션이다.

7.1.1 "2.2 커널"의 컴파일 옵션

2.2.x 커널들은 옵션이 대부분 똑같지만, 새로운 옵션이 몇 가지 새로이 개발되어 있다. 아래에 설명되어 있는 대부분은 커널을 컴파일하는 중의 `make config` 단계에서 도움말 (Help facility)로 사용되는 `/linux/Documentation/Configure.help` 문서와 내용이 같다.16.5 ([23]) 새로 추가된 옵션들 만 아래에 적어 놓았다. 다른 필요한 옵션을 알고 싶으면 2.0 설명을 보기 바란다. 아마도 2.2 커널에서 가장 의미 있는 변화는 IP 방화벽 코드일 것이다. IP 방화벽을 설치하기 위해서 2.0 커널에서는 `ipfwadm` 풀그림이 사용되었지만, 2.2 커널에서는 `ipchains` 풀그림이 사용되고 있다.

- 소켓 필터링 (CONFIG_FILTER) 대부분의 사용자들에게는 이 옵션을 끄고 사용하는 것이 안전할 것이다. 이 옵션은 사용자 공간 필터를 소켓에 연결하는 것을 가능하게 해 주고, 패킷 통과가 허락되거나 거부되는 것을 결정하는데 사용한다. 하지만, 여러분이 상당히 독특한 필터를 쓸 필요가 있고 필터를 직접 프로그램 할 수 있는 능력이 있는 사람이 아니라면 이 옵션은 꺼 놓는 것이 좋다. 또한 한 가지 염두에 둘 일은 이 문서가 작성된 시점에서는 TCP를 제외한 모든 프로토콜이 지원된다는 것이다.

- 포트 포워딩 (Forwording 전송 처리 傳送處理) 포트 전송은 IP 마스커레이딩 등과 같이 바깥에서 방화벽 안의 포트들로 들어오는 패킷들을 전달 (Forword)한다. 예를 들어서, 방화벽의 뒤에서나 마스커레이드 하는 위에서 웹 서버를 운영하면서 이 웹 서버를 바깥에서도 접근 가능하게 허락하는 경우 등에 유용하다. 외부의 클라이언트가 방화벽의 80번 포트로 어떤 요청을 보내고, 방화벽은 이 요청 사항을 웹 서버로 전달하고, 그럼 웹 서버는 이 요청 사항을 처리한 후에 결과를 방화벽을 통해서 원래의 클라이언트에게로 다시 보내게 된다. 외부의 클라이언트에게는 마치 방화벽 자체가 웹 서버를 운영하고 있는 듯이 보이게 되는 것이다. 또한 방화벽 뒤에 동일한 모양과 기능의 웹 서버들을 많이 가지고 있다면 부하 조절 (Load Balanceing)을 하는 방법으로 쓰일 수도 있다. 이 기능에 대한 정보는 *http://www.monmouth.demon.co.uk/ipsubs/portforwarding.html* `<http://www.monmouth.demon.co.uk/ipsubs/portforwarding.html>`에서 구할 수 있다. 대략적인 정보는 *ftp://ftp.compsoc.net/users/steve/ipportfw/linux21/* `<ftp://ftp.compsoc.net/users/steve/ipportfw/linux21/>`를 보기 바란다.

- 소켓의 여과 (CONFIG_FILTER) 이 옵션을 써서 사용자 공간 (user-space) 풀그림들이 아무 소켓에다가 필터를 부착시킬 수 있게 되고, 결과적으로 커널이 어떤 종류의 데이타에 대해 소켓 통과를 허가하거나 불허하게 할 수 있게 된다. 리눅스 소켓 여과 (Linux Socket Filtering)는 현재 TCP를 제외한 모든 종류의 소켓에서 작동될 수 있다. `./linux/Documentation/networking/filter.txt`를 읽어보기 바란다.16.5 ([24. 레드 햇])

- IP: 마스커레이딩 2.2 커널의 마스커레이드 기능은 특수한 프로토콜 등의 마스커레이드를 지원하는 등으로 개선되어져 있다. 보다 많은 정보를 원한다면 IPCHINS 하우투를 꼭 읽어보기 바란다.

7.2 커널 디바이스들

리눅스에는 보안에 도움이 되는 몇 개의 블록 디바이스와 문자 디바이스가 있다.

`/dev/random`과 `/dev/urandom`의 두 디바이스는 랜덤 데이타를 언제라도 만들어 낼 수 있도록 커널에서 제공된다.

`/dev/random`과 `/dev/urandom` 둘은 보안상 안전한 (secure) 난수 발생 기능이 필수적인 PGP 열쇠의 제작, `ssh` 수하 도전용 (challenge), 그리고 기타 풀그림들 등에 사용될 수 있을 만큼 충분히 안전해야 한다. 공격자가 – 이 두 기능에서 발생된 숫자들의 조합을 미리 알고 있다고 해서 – 그 다음의 나올 숫자를 알아내는 일이 가능해서는 안된다. 이 둘로부터 생성되는 숫자들이 진정한 의미로서의 난수가 되도록 보장하는 많은 노력들이 쏟아 부어지고 있다.

두 디바이스의 차이점이라면 `/dev/random`은 무작위의 바이트들로 만들어지며, 무작위의 바이트들이 만들어 쌓이는 동안은 대기 상태가 된다는 정도일 것이다. 일부 시스템에서는 새로운 사용자 생성 입력 (user-generated entry)이 시스템에 등록되는 시간이 오래 걸릴 수 있고, 그 동안은 (사용이) 막혀 있을 수 있다는

것을 말하고 싶다. /dev/random을 쓰기 전에는 심사숙고하기를 바란다. (아마도 제일 좋은 방법은 형성이 되고 있는 사이에 – "OK 충분합니다" 하는 메시지가 나올 때까지 – 사용자들이 키보드를 두들기게 끔 하는 것일 것이다)

/dev/random은 – 인터럽트 사이의 시간을 재서 만드는 등의 – 질이 높은 엔트로피이다. 이것도 랜덤 데이타가 충분히 만들어질 때까지 막고 있게 된다.

/dev/urandom은 비슷하지만, – 엔트로피가 낮을 때의 경우 – 암호학 기법 상 강하다고 할 수 있는 헤쉬 값을 만들어 준다. 비록 이것은 (/dev/random으로 만들어지는 값에 비하면) 상대적으로는 덜 안전하지만, 대부분의 풀그림용으로는 충분하다.

다음과 같은 방법 등으로 디바이스로부터 읽어 낼 수 있다.

```
root#  head -c 6 /dev/urandom | mimencode
```

이것은 – 패스워드를 만듦에 적절할 – 8자의 난수를 콘솔로 출력할 것이다. mimencode는 메타 메일 패키지에서 볼 수 있을 것이다.

연산법에 대한 설명은 /usr/src/linux/drivers/char/random.c에 있다. 내가 (데이브) 이 것을 쓰는데 도와준 디어도어 와이 쮜, 존 루이스 그리고 리눅스 커널 팀 여러분에게 감사드린다.

제 8 절 네트워크 보안

사람들이 더 많은 시간을 컴퓨터 접속에 보내게 되면서, 네트워크 보안은 더욱 더 중요해 지고 있다. 네트워크 보안은 물리적이나 지역의 보안을 깨는 것보다 훨씬 쉽다.

네트워크 보안을 도와줄 도구들은 많으며, 갈수록 많은 것들이 리눅스 배포본에 실려 배포되고 있다.

8.1 패킷 스니퍼

침입자가 네트워크의 더 많은 시스템으로 침투하기 위해서 가장 흔하게 쓰는 방법 중의 하나가 이미 깨어진 호스트에서 패킷 스니퍼를 실행하는 것이다. 이 "스니퍼"는 이떠넷 포트를 감청하면서 지나가는 패킷 흐름에서 Password, Login, su 같은 것이 들리면 그 이후의 내용을 녹음해 둔다. 이 방법을 쓰면, 공격자는 침투하려고 시도조차 않았던 시스템으로까지 들어가는 패스워드를 얻게 된다. (암호화가 안된 채로) 평문으로 전송되는 패스워드는 이 공격에 매우 약한 것이다.

예: 호스트 A의 보안이 깨졌다. 공격자는 (여기에) 스니퍼를 설치한다. (잠시 후,) 어떤 관리자의 호스트 C에서 호스트 B로 들어가려는 접속 로그인을 스니퍼가 감지한다. (이제 패킷이 녹음이 되고 있다) 관리자가 B로 로그인을 하는 순간, 이 관리자의 개인 패스워드는 녹음이 된다. 잠시 후 관리자가 – 어떤 문제를 해결하기 위해 – su를 사용한다. 이제 호스트 B의 루트 패스워드까지 얻게 되었다. 잠시 후에, 관리자가 누군가가 자기 계정에서 다른 사이트에 있는 호스트 Z로 텔넷을 하도록 해 두면 공격자는 이제 호스트 Z로 로그인할 패스워드까지 갖게 된다.

요즘에는 공격자가 패킷 스니퍼를 쓰기 위해 시스템의 보안을 깨고 침입할 필요조차 없어져 버렸다. 공격자는 랩탑이나 PC를 건물 안으로 들고 들어와서 네트워크를 감청하면 그만인 것이다.

ssh나 다른 암호화된 패스워드 방식을 사용하면 이 공격을 방해할 수 있다. POP 계정용의 ATOP 등이 이 공격을 방지한다. (네트워크를 통해 평문 패스워드를 전송하는 방법들이 다 그렇듯이, 보통의 POP 로그인은 스니퍼에 대단히 취약하다.)

8.2 시스템 서비스와 tcp_wrapper

어떤 서비스를 제공할 필요가 있는가를 선별하는 것은 네트워크에 리눅스 시스템을 올려놓기 전부터 해야 할 일이다. 제공할 필요가 없는 서비스를 아예 해체해 버리면 걱정거리가 하나 줄고, 공격자가 개구멍을 찾

을 대상을 하나 줄여 버리는 것이 되는 것이므로.

리눅스 시스템에서 서비스를 꺼 버리는 방법은 많이 있다. /etc/inetd.conf 파일을 보면 inetd가 현재 어떤 서비스를 제공하고 있는지 알아볼 수 있다. 필요 없는 서비스는 모두 주석문 (remark) 처리를 해서 막아 버리고 ((#을 줄의 가장 앞에 쓴다), inetd 프로세스에 SIGHUP 신호를 보내도록 조정하라.16.5 ([25. SIGHUP])

아울러 /etc/services 파일에서도 서비스를 주석문 처리를 하거나 삭제할 수 있다. 이것은 지역 사용자들도 또한 서비스를 못쓰게 된다는 뜻이다 (예로서, 만약 여러분이 ftp를 삭제해 난 후, 이 기계에서부터 원격 사이트로 ftp를 사용하려 하면 "unknown service" 메시지가 나오면서 안 받아 줄 것이다) 보안성이 늘어나는 것은 아니므로 꼭 서비스를 /etc/services에서 없애 버릴 가치는 없다. 만약 지역 사용자가 – 여러분이 주석문 처리를 해서 꺼 버린 – ftp를 쓰고 싶어한다면, 그는 간단히 자신의 클라이언트를 사용하면서 공용 ftp 포트를 써서 여전히 일을 할 수 있을 것이다.

켜 놓는 것이 좋을 서비스들은:

- ftp
- telnet (혹은 ssh)
- pop-3 이나 imap 등의 메일
- identd

등이 있다.

어떤 패키지를 쓸 일이 없으리라는 것을 알고 있다면, 그 패키지를 완전히 삭제할 수도 있다. 레드 햇 배포본에서는 rpm -e 명령으로 한 패키지 전체를 지울 수 있다. 데비안에서는 dpkg로 같은 작업을 할 수 있을 것이다.

덧붙여서, (rlogin이 쓰는) login과 (rcp가 쓰는) shell 그리고 (rsh가 쓰는) exec를 /etc/inetd.conf에서 시작되는 것을 막는 것을 포함해서, /rsh/r;pgin/rcp 도구를 꺼 버리는 것이 정말로 필요하다. 이들 프로토콜은 극단적으로 보안이 허술하며 (insecure), 예전부터 침탈 (exploit)의 근원이 되어 왔다.

레드 햇은 /etc/rc.d/rc[0-9].d를 보고, 데비안 경우에는 /etc/rc[0-9].d를 보는 등으로 디렉토리에서 실행되는 서버들 가운데 불필요한 것들이 있는가 확인하라. 이들 파일들은 실제는 /etc/rc.d/init.d (레드에의 경우; 데비안은 /etc/init.d) 디렉토리로 심볼릭 링크 되어 있다. init.d에 있는 파일들의 이름을 바꿔 버리면 심볼릭 링크를 꺼 버리는 효과를 가져온다. 만 약 특정 런 레벨에 맞추어서 적당한 서비스를 꺼 주고 싶으면, 이에 상응하는 심볼릭 링크를 대문자 (Upper-case)에서 소문자 (Lower-case)로 이름을 바꿔 주면 된다. 다음의 경우는 대문자 S를 소문자 s로 바꾼 것이다.

```
root#  cd /etc/rc6.d
root#  mv S45dhcpd s45dhcpd
```

BSD 형식의 rc 파일들을 갖고 있다면 /etc/rc*을 검사해서 필요 없는 풀그림들을 볼 수 있다.

대부분의 리눅스 배포본에는 모든 TCP 서비스들을 "보호해 주는(wrapping)" 티시피 랩퍼 (tcp wrapper)가 들어 있다. tcp_wrapper (tcpd)는 실제 서버를 실행 할 수 있는 것이 아니 고, 대신 inetd가 불러오는 방법으로 실행된다. 그러면 tcpd는 서비스를 요청하는 호스트를 검사해서, 서버를 실행시키거나 그 호스트로부터의 접근을 거부한다. tcpd를 이용해서 tcp 서비스로의 접근을 제한할 수 있는 것이다. /etc/hosts.allow 파일을 만들고, 여러분 컴퓨터의 서비스에 접근할 필요가 있는 호스트들만을 추가하도록 한다.

여러분이 집에서 모뎀을 쓰는 다이얼-업 사용자라면, 필자는 "모든" 서비스에 대한 접근을 거부하도록 조정하기를 권장한다. tcpd는 서비스에 접근하려다가 실패한 시도들을 기록하므로, 공격을 받고 있다는 것을 경고해 줄 수도 있다. TCP를 기반으로 하는 새로운 서비스를 추가로 설치하게 되면, 반드시 tcp wrapper가 이 서비스를 추가 감시하도록 다시 구성하는 것이 좋다. 예를 들면, 가정의 모뎀 사용자 (dial-up)는 외부인

이 자신의 기계에 연결하는 것을 막으면서도, 메일을 받도록 인터넷에 네트워크 연결을 할 수가 있다. 이렇게 만들려면 `/etc/hosts.allow`에 다음을 추가한다.

ALL: 127.

물론 /etc/hosts.deny에도

ALL: ALL

이렇게 해 놓으면 외부에서 들어오는 연결은 막으면서도, 내부에서 인터넷으로 나가는 연결은 할 수 있게 된다.

염두에 둘 것은 tcp_wrapper는 `inetd`와, 선정된 소수의 다른 것들에서부터 실행되는 서비스들만 보호한다는 것이다. 여러분이 쓰는 기계에는 다른 서비스들도 돌아가고 있는 것일 수 있다는 것을 생각해 두자. 여러분 기계에서 돌아가는 모든 서비스를 보려면 `netstat -ta`를 쓰면 된다.

8.3 DNS 정보의 확인

여러분 네트워크의 모든 호스트에 대한 DNS 정보를 최신판으로 유지하는 것으로도 보안이 강화할 수 있다. 만약 불법 호스트가 여러분 네트워크에 연결되는 상황이 벌어지면, DNS 엔트리가 없을 것이므로, 침입을 알아챌 수가 있게 된다. 많은 서비스들은 – 유효한 DNS 엔트리가 없는 호스트는 접속을 거부하는 식으로 – 조정할 수 있게 되어 있다.

8.4 identd

`identd`는 주로 `inetd` 서버에서 수행되는 작은 풀그림이다. 어느 사용자가 어떤 TCP 서비스를 수행시키는지 추적하고, 요구하는 누구에게든 추적 결과를 보고한다.

많은 사람들이 `identd`의 유용성을 오해하고, 이것을 꺼 버리거나 외부 사이트로부터 오는 요청을 거부하도록 막아 둔다. `identd`는 단지 원격 사이트에 도움을 주기 위해서 있는 것이 아니다. 여러분이 원격 `identd`로 얻은 자료가 옳은지 알 방법은 없으므로. `identd` 요청에는 아무런 인증 절차가 없기 때문이다.

그렇다면 왜 `identd`를 수행시켜야 할까? `identd`가 *여러분*을 도와주기 때문이고, 추적 시에는 여러분의 검문소 역할을 하기 때문이다. 여러분의 `identd`가 변조되지 않았다면 TCP 서비스를 쓰고 있는 사람들의 사용자 이름이나 `uid`를 `identd`가 원격 사이트에 말해 줄 수 있는 것을 알 것이다. 만에 하나, 원격 사이트의 관리자가 여러분에게 와서 여러분 컴퓨터의 어느 사용자가 자기의 사이트로 침입하려고 했다고 말한다면, 여러분은 손쉽게 그 사용자에 대해서 행동을 취할 수 있다. `identd`를 실행시키고 있지 않았다면, 누가 그때 있었는지 알아내기 위해서 수많은 기록들을 살펴보아야 하고, 이런 경우 일반적으로 그 사용자를 추적하기 위해서 훨씬 긴 시간이 걸리게 된다.

대부분의 배포판에 들어 있는 `identd`는 대부분의 사람들이 생각하는 것보다 더 다양한 설정이 가능하다. 특정한 사용자용으로 `identd`가 작동하지 않도록 할 수 있고 (이 사용자들은 `.noident` 파일을 만들면 된다), 모든 `identd` 요청을 기록하도록 할 수 있으며 (필자는 이렇게 하기를 권한다) 사용자 이름 대신 `uid`나 NO-USER를 표기하도록 할 수도 있다.

8.5 SATAN, ISS, 그리고 다른 네트워크 스캐너 풀그림들

포트와 서비스를 대상으로 컴퓨터들과 네트워크에 대한 검사 (scan)를 수행하는 많은 소프트웨어 패키지들이 있다. SATAN과 ISS는 그 가운데 비교적 잘 알려진 풀그림이다. 이 소프트웨어들은 표적 컴퓨터의 (혹은 한 네트워크 상의 모든 표적 컴퓨터들의) 가능한 모든 포트에 연결하려고 시도하며, 어떤 서비스가 그곳에서 수행되고 있는지 찾아내고자 한다. 이 정보를 바탕으로 표적 컴퓨터가 어떤 침탈법에 취약한지 찾을 수 있다.

SATAN(Security Administrator's Tool for Analyzing Networks)는 웹 인터페이스를 쓰는 포트 스캐너 풀그림이다. 컴퓨터 한 대나 하나의 네트워크에 대한 검사 강도는 강, 중, 약 등으로 임의 설정할 수 있다.

SATAN을 구해서 여러분의 컴퓨터나 네트워크를 조사해서 발견되는 문제를 고치는 것이 좋다. SATAN을 *메타랩* <http://metalab.unc.edu/pub/packages/security/Satan-for-Linux/> 등의 믿을 만한 FTP, 또는 웹 사이트에서 구하도록 주의해야 한다. SATAN을 가장한 트로이 목마가 인터넷에 떠돌고 있기 때문이다. *http://www.fish.com/satan/* <http://www.fish.com/satan/> 하나 알아두면 좋을 것은, SATAN이 근래에 업데이트되지 않았고, 아래에 적어 놓은 다른 도구들이 검사 작업을 더 잘할 수가 있다는 것이다.

ISS (Internet Security Scanner)는 또 다른 포트형 검사 풀그림이다. SATAN 보다 빠르며, 따라서 대규모의 네트워크를 검사하기에 더 적합할 수 있다. 하지만 SATAN이 더 많은 정보를 제공하는 경향이 있다.

아바커스는 호스트용의 보안과 침입 감지 기능을 제공해 주는 도구들의 뭉치다. 이것의 홈 페이지에 가면 더 많은 정보를 얻을 수 있다. *http://www.psionic.com/abacus* <http://www.psionic.com/abacus>

세인트 (SAINT)는 사탄 (SATAN)을 업데이트한 형식으로 만들어진 것이다. 웹 상에서 돌아가며 SATAN 보다 많은 신형의 테스트를 실행할 수 있다. 이것에 대한 정보는 *http://wwdsilx.wwdsi.com/saint/* <http://wwdsilx.wwdsi.com/saint/>에서 구할 수 있다.

네서스 (Nessus)는 무료 보안 풀그림이다. 쉽게 쓸 수 있도록 GTK 그래픽 인터페이스를 사용한다.16.5 ([26. GTK]) 새로운 포트 스캔 테스트 방법이 나오면 그 부분만을 업데이트를 할 수 있도록 테스트들을 플러그인 형식으로 내려 받아 쓸 수 있게 만들어져 있다. 정보를 원한다면 *http://www.nessus.org* <http://www.nessus.org>에서 구할 수 있다.

8.5.1 포트 스캔 경우의 탐지

SATAN 이나 ISS, 또는 다른 스캐너 풀그림들이 여러분의 컴퓨터를 탐색 (Probe)해 들어오는 것을 경보를 해 주는 여러 도구들이 있기는 하다. 하지만, tcp_wrapper를 여기 저기 많이 쓰면서 여러분의 일지 문서들을 정기적으로 자주 들춰보는 것도 탐색을 알아채는 좋은 방법이다. 가장 강도를 낮추어 조정해도 SATAN은 보통의 레드 햇 시스템의 일지에 자취를 남기게 되므로.

”스텔드 (Stealth)” 포트 스캐너도 염두에 두자. (세션이 연결된 상태에 만들어지는) TCP의 ACK 비트가 적혀 있는 패켓은 패켓 필터링 방화벽도 관통할 경우가 있다. 세션이 연결되지 않은 상태의 포트에서 답으로 보내 주는 RST 패켓은 – 감지자에게는 – 그 포트가 존재한다는 증거로 쓰일 수 있을 것이다. 내 생각에는 tcp_wrapper가 이런 식의 우회 수색을 감지해 내지는 못한다고 생각한다. 16.5 ([27. 네서스 플러그인])

8.6 센드메일, 큐메일과 MTA

여러분이 제공할 수 있는 가장 중요한 서비스들 가운데 하나가 메일 서버이다. 불행히도 메일 서버는 공격에 가장 취약한 서비스 중의 하나인데, 그 까닭은 이것이 수행해야 하는 작업의 숫자와 필요로 하는 권한이 많기 때문이다.

`sendmail`을 쓰고 있다면, 최신 버전을 사용하는 것이 매우 중요하다. `sendmail`은 길고도 긴 침탈의 역사가 있다. 가장 최근의 버전을 항상 사용하도록 유의하라. *http://www.sendmail.org* <http://www.sendmail.org>

염두에 둘 것은 메일을 보내기 위해서 센드메일을 켜 놓을 필요가 없다는 것이다. 만약 여러분이 집에서 혼자 쓰는 홈 유저라면, 센드메일을 아예 끄고서 단순한 메일 클라이언트 풀그림을 써서 메일을 보낼 수 있다. 또한, 센드메일 초기 실행 문서 (Startup File)에서 ”-bd” 플랙을 지움으로서 메일의 수신 요청 (Incoming Request)을 아예 끄는 것도 좋을 것이다. 다른 말로 하면, 여러분의 초기 실행 스크립트에서 다음의 명령어를 사용해서 센드메일을 실행할 수 있을 것이다:

```
# /usr/lib/sendmail -q15m
```

이 명령어는 첫 시도로 전달이 안되어서 메일 큐 (Queue)로 보내진 메시지를 15분 간격으로 다시 보내려는 재 시도를 한다.16.5 ([28])

많은 관리자는 아예 센드메일을 사용하지 않고 다른 메일 전달 에이전트를 사용하기도 한다. 원한다면 qmail로 교체 사용하는 것을 고려하는 것도 좋을 것이다. qmail은 처음부터 보안을 염두에 두고 설계되었다. 이 풀그림은 보다 빠르고 안정적이며 보안상 안전하다. *http://www.qmail.org* `<http://www.qmail.org>`에서 구할 수 있다.16.5 ([29. 센드메일과 큐메일])

qmail과 정면으로 경쟁하고 있는 "postfix"도 있다. 이것은 tcp_wrapper 등의 여러 보안 도구를 만든 윗세 뵈이니마 (Wietse Venena)가 만든 것이다. 이 전에는 vmailer라고 불렸던 것이고, 현재는 IBM이 후원을 하고 있으며, 설계 초반에서부터 보안을 염두에 두고 만들어져 있다. 이에 대한 정보는 *http://postfix.org* `<http://postfix.org>`에서 구할 수 있다. 16.5 ([30. postfix])

8.7 서비스 거부 유도 방식의 공격

8.7.1 (Denial of Service attacks: 이하 DoS)

시스템의 서비스 거부를 유도하게 만드는 방식의 공격 (Denial of Service: DoS)은 공격자가 시스템 자원의 일부를 매우 바쁘게 만들어서 불통시키는 방법을 써서 시스템이 정식 요청에 응답하지 못하게 만들거나 정식 사용자의 시스템 접근을 거부하게 만드는 것이다.

이런 종류의 공격은 근년에 들어 크게 증가해 왔다. 최근의 공격 방법 중 잘 알려진 것들을 아래에 적었다. 새로운 공격 방법들이 항상 나타나고 있으므로 여기 소개된 것들은 그저 몇 가지 사례에 불과하다는 것을 명심해야 한다. 더 새로운 정보를 얻으려면 리눅스 보안 리스트와 벅트랙 (bugtraq) 리스트와 아카이브를 읽도록 하라.16.5 ([34. 벅트랙 주소])

- **SYN 홍수(SYN Flooding)**: SYN 홍수는 네트워크를 통한 서비스 거부 공격이다. 이 방법은 TCP 연결 방식 중에 있는 "허점"을 이용한다. (2.0.30 이후의) 새로운 리눅스 커널들은 SYN 범람 공격 방지하기 위한 조정 옵션들을 가지고 있다. 커널 보호 옵션을 보려면 7 ([커널 보안])을 보도록 하라.

- **펜티움 "FOOF" 버그**: 인텔의 정품 펜티움 프로세서에 일련의 어셈블리 코드를 보낼 경우 컴퓨터가 무조건 리부트하게 된다는 것이 최근에 발견되었다. 이것은 어떤 운영체제인가에 관계없이 (펜티움 클론과 펜티움 프로, 펜티움2를 제외한) 펜티움 프로세서를 사용하는 모든 컴퓨터에 영향을 미친다. 2.0.32 이상의 리눅스 커널에는 이 버그로 인해 컴퓨터가 오류 작동하는 것을 막는 우회법이 포함되어 있다. 2.0.33 커널은 좀 더 개선된 커널 수정안을 가지고 있고, 2.0.32보다 좋은 것으로 인식되고 있다. 펜티움을 사용하고 있다면, 지금 업그레이드를 해야 한다.

- **Ping 홍수 (Ping Flooding)**: Ping 홍수는 쉬운 부루트 포스 DoS 공격의 일종이다. 공격자는 ICMP 패켓을 "홍수처럼" 여러분의 컴퓨터에 보낸다. 공격자가 이 공격을 여러분의 컴퓨터 보다 연결 속도가 빠른 (better bandwidth) 컴퓨터에서 시도한다면, 여러분의 컴퓨터는 네트워크로 아무 것도 전송할 수 없게 될 것이다. 이 공격법의 변종 중 하나인 "스머핑 (Smurfing)"은 – 범인을 찾아내는 것이 더 어렵도록 – ICMP 패켓들의 발신지를 여러분 컴퓨터의 주소인 것처럼 위장해서 다른 호스트에 (패켓 신청을) 보낸다. "스퍼프" 공격에 대해서는 *http://users.quadrunner.com/chuegen/smurf.txt* `<http://users.quadrunner.com/chuegen/smurf.txt>`에서 더 정보를 얻을 수 있다. Ping 범람 공격을 받고 있다면, 어디에서 패켓이 오는지 (혹은 오는 것처럼 보이는지) 알아내기 위해서 tcpdump 같은 도구를 쓰도록 하고, 여러분의 ISP에게 이 사실을 연락하도록 하라. Ping 범람은 라우터 수준에서 차단하거나 방화벽을 쓰는 것이 가장 쉽다.

- **죽음의 핑 (Ping o' death 핑 오브 데스)**: 죽음의 핑 공격은 커널 데이타 구조가 수용할 수 있는 것보다 크기가 큰 IMCP 에코를 요청하는 (IMCP ECHO REQUEST) 패켓이 원인이다. (65510 바이트의) 커다란 "핑"을 시스템에 보내면 시스템이 서 버리거나 죽어 버리기 때문에, "죽음의 핑"이라고 불리게 되었다. 이 문제는 오래 전에 이미 해결책이 나와 있으니 크게 걱정할 필요는 없다.

- **티어드랍 (Teardrop 눈물 방울) / 뉴 티어 (New tear)**: 최근 침탈법의 하나인 데 리눅스와 윈도우스의 IP 프래그멘테이션 코드에 존재하는 버그를 쓴다. 2.0.33 버전의 커널에서부터 고쳐지기 시작했고, 고칠 때 커널 컴파일-타임 옵션을 선택할 필요는 없다. 리눅스는 "뉴 티어" 침탈법에는 영향을 받지 않는다.

대부분의 침탈법 코드와 이 것들이 어떻게 움직이는 지에 대한 깊은 설명이 필요하면 *http://rootshell.com* <http://rootshell.com>에서 그들의 서치 엔진을 써서 구할 수 있다.

8.8 NFS (네트워크 파일 시스템) 보안

NFS는 매우 널리 쓰이는 파일 공유 프로토콜이다. NFS를 이용하면 – 커널에서 nfs 파일시스템을 지원해 주는 (만약 리눅스가 아닌 경우에는 다른 클리이언트가 지원해 주는) 다른 컴퓨터들로 – `nfsd`와 `mountd`를 실행하는 서버가 파일시스템을 "수출" 할 수 있게 해 준다. `Mountd`는 `/etc/mtab`에 마운트된 파일시스템을 관리하며, `showmount`를 쓰면 NFS 내용을 볼 수 있다.

사용자들에게 홈 디렉토리를 제공하기 위해서 NFS를 많은 사이트가 사용하고 있으며, 이렇게 함으로써 사용자들이 어느 컴퓨터에서 로그인하였던 간에 사용자들은 홈 파일들을 가질 수 있게 된다.

파일시스템을 공유할 때 사용할 수 있는 몇 안되는 "보안" 설정이 있다. 여러분은 원격 컴퓨터의 루트 사용자(`uid=0`)를 `nobody` 사용자로 대응시켜서, 공유된 파일시스템 전체 접근 권한을 갖는 것을 거부하도록 nsfd를 설정해야 한다. 그러나 개인 사용자는 각자의 (혹은 최소한 같은 `uid`의) 파일에 대한 접근권이 있기 때문에, 원격지의 루트 유저는 자기 계정으로의 로그인이나 `su` 사용이 가능하며, 자기 파일들에 대해서 완전한 접근권을 가질 수 있다. 이렇게 하는 것은 원격 파일시스템을 마운트할 권한을 가진 공격자에게는 사소한 장애물밖에 되지 못한다.

NFS를 꼭 써야 한다면, 파일시스템이 공유 사용되어야 만 하는 컴퓨터에게로만 전송 되도록 조정하라. 루트 디렉토리 전부를 수출하도록 설정해서는 절대로 안되며, 공유가 필요한 디렉토리만 수출 (Export)하도록 설정해야 한다.

NFS에 대한 더 자세한 정보가 필요하면 NFS 하우투를 보도록 하라. 이 것의 원문은 *http://metalab.unc.edu/mdw/HOWTO/NFS-HOWTO.html* <http://metalab.unc.edu/mdw/HOWTO/NFS-HOWTO.html>이다.

8.9 NIS (네트워크 정보 서비스) (예전의 YP)

네트워크 정보 서비스(Network Information service, 예전의 YP)는 그룹의 컴퓨터들에 정보를 배포하는 한 가지 방식이다. NIS 주 서버는 정보표를 소유하며 그것들을 NIS 대응 (map) 파일들로 변환한다. 이 대응 파일들이 네트워크를 통해 제공됨으로써 NIS 클라이언트 컴퓨터들은 로그인과 패스워드, 홈 디렉토리와 쉘에 대한 정보 (즉 보통의 `/etc/passwd` 파일에 들어 있는 모든 정보)를 얻을 수 있게 된다. NIS를 이용하면 사용자들은 패스워드를 한 번만 바꾸면 그 NIS 영역에 들어 있는 모든 컴퓨터에 (정보가 갱신되도록) 할 수 있다.

NIS는 안전한 것이 아니다. 원래부터 안전을 염두에 두고 만든 것이 아니었다. 단지 간편하고 쓸모 있는 작업 역할을 위해서 만든 것뿐이다. (네트워크 상 어디에 있건) 여러분의 NIS 도메인의 이름을 알아맞힐 수 있는 사람은 여러분의 `passwd` 파일 복사본을 얻을 수 있고, 여러분의 사용자 패스워드를 깨기 위해 "크랙 (Crack)"과 "존 더 립퍼 (John the ripper)"를 쓸 수 있게 된다. NIS를 속여서 (spoof) 온갖 지저분한 일을 하게 할 수도 있다. 꼭 NIS를 써야 겠다면, 이런 위험들을 감수해야 한다.

NIC+라고 불리는 NIC보다 안전한 대체품이 있다. NIC 하우투를 읽어보기 바란다. *http://metalab.unc.edu/mdw/HOWTO/NIS-HOWTO.html* <http://metalab.unc.edu/mdw/HOWTO/NIS-HOWTO.html>

8.10 방화벽

방화벽(firewall)은 여러분의 지역 네트워크 안팎으로 어떤 정보가 출입할 것인가를 조정하는 한 가지 방법이다. 전형적으로 방화벽 호스트는 인터넷과 지역 랜을 간섭해서, 랜에서 인터넷으로의 엑세스는 방화벽을 통해서만 가능하도록 하는 것이다. 이렇게 하면 방화벽을 써서 인터넷과 여러분의 (로칼 호스트) 사이를 오가는 정보를 제어할 수 있다.

방화벽에는 수많은 유형과 구성 방법들이 있다. 리눅스 컴퓨터는 훌륭한 방화벽이 될 수 있다. 2.0 이상의 커널에 컴파일을 써서 방화벽 코드를 바로 삽입될 수 있다. 사용자 공간 도구인 2.0 커널용 ipfwadm, 2.2 커널용 ipchains를 쓰면, 어떤 종류의 네트워크 트래픽을 허락할 것인가를 손쉽게 바꿀 수 있다. 또한 일정한 유형의 네트워크 트래픽을 일지에 적도록 (log) 조정할 수 있다.

방화벽은 네트워크의 보안 작업에 있어서 매우 중요하고도 유용한 기술이다. 하지만 방화벽이 있으니까 방화벽이 보호하고 있는 네트워크의 컴퓨터들 자체의 보안은 필요 없다고 생각해서는 절대로 안 된다. 이런 안이한 생각은 절대적으로 치명적인 실수인 것이다. 방화벽과 리눅스에 대한 정보를 더 얻고 싶으면 선사이트에 가서 최근에 만들어진 방화벽과 하우투를 읽어보도록 하라. *http://metalab.unc.edu/mdw/HOWTO/Firewall-HOWTO.html* `<http://metalab.unc.edu/mdw/HOWTO/Firewall-HOWTO.html>`

추가의 정보가 IP-마스커레이드 미니 하우투 파일에도 있다 *http://metalab.unc.edu/mdw/HOWTO/mini/IP-Masquerade.html* `<http://metalab.unc.edu/mdw/HOWTO/mini/IP-Masquerade.html>`

ipfwadm (방화벽 세팅을 바꾸는 도구)에 대한 추가의 정보는 이 것의 홈 페이지에서 구할 수 있다. *http://www.xos.nl/linux/ipfwadm/* `<http://www.xos.nl/linux/ipfwadm/>`

만약 여러분이 방화벽에 대한 경험이 전혀 없으면서도, 단순 보안용을 넘는 상황에서 방화벽을 쓰기 위해서 방화벽을 세우려는 상황이라면, 오라일리 앤드 어소시에이츠에서 출판한 방화벽 관련 서적이나 온라인 상에서 구할 수 있는 방화벽 문서들을 찾아서 읽어 봐야 하는 것은 필요 조건일 것이다. *http://www.ora.com* `<http://www.ora.com>`에서 정보를 구할 수 있다.16.5 ([책1]) 미국 국립 표준 기술 연구소 (National Institure of Standards and Technology: NIST)가 구성한 방화벽에 대한 우수한 문서도 있다. 비록 1995년에 만들어진 문서이기는 해도 아직도 읽을 만한 문서다. 이 것은 *http://csrc.nist.gov/nistpubs/800-10/main.html* `<http://csrc.nist.gov/nistpubs/800-10/main.html>`에서 구할 수 있다. 또 다른 읽을 만한 문서들은:

- 프리화이어 프로젝트 – 무료로 사용할 수 있는 방화벽 및 도구들의 목록. *http://sites.inka.de/sites/lina/freefire-l/index_en.html* `<http://sites.inka.de/sites/lina/freefire-l/index_en.html>`에서 구할 수 있다.
- 선월드 방화벽 디자인 – 오라일리에서 출판한 책의 저자들이 쓴 문서다. 이 문서는 다른 종류들의 방화벽에 대한 대강의 소개를 한다. 이 것은 *http://www.sunworld.com/swol-01-1996/swol-01-firewall.html* `<http://www.sunworld.com/swol-01-1996/swol-01-firewall.html>`에서 구할 수 있다.
- 메이슨 (Mason) - 리눅스 용의 자동화 방화벽 구축 풀그림. 네트워크에 방화벽을 구축하는 과정에서 자동으로 배우면서 사용할 수 있는 도구이다! 정보는 *http://www.pobox.com/˜wstearns/mason/* `<http://www.pobox.com/˜wstearns/mason/>`에서 구할 수 있다.

8.11 IP 사슬 - 리눅스 커널 2.2.x 방화벽 작업 16.5 ([31])

리눅스 IP 방화벽 사슬 (IP Firewalling Chains)은 2.0 리눅스 방화벽 작업용 코드를 2.2 커널용으로 업데이트한 것이다 이 업데이트된 코드는 예전의 설치 코드에 비하면 매우 많은 기능을 제공한다. 새 기능의 예로서

- 보다 융통성 있는 패켓의 조작.
- 보다 자세한 계정/책임의 관리.
- 국소적이며 쉬운 보안 정책 (policy) 수정이 가능.
- (패켓) 조각들의 철저한 봉쇄, 거부 등이 가능.
- 수상한 패켓을 일지에 기록.
- ICMP/TCP/UDP 이외의 프로토콜도 취급 가능.

만약 여러분이 현재 2.0 커널에서 `ipfwadm`을 사용하고 있다면, `ipfwadm` 명령 형식을 `ipchains`가 사용하는 것으로 바꿔 주는 스크립트가 있다.

보다 많은 정보를 위해서는 IP 사슬 하우투 (IP Chains Howto)를 반드시 읽어보기 바란다. 이 것은 *http://www.rustcorp.com/linux/ipchains/HOWTO.html* `<http://www.rustcorp.com/linux/ipchains/HOWTO.html>`에서 구할 수 있다.

8.12 VPN - 가상사설망

VPN (Virtual Private Network: 가상사설망 혹은 가상 사설 네트워크)은 이미 존재하는 네트워크를 이용해서 "가상 존재"의 네트워크를 설립하는 방법 중의 하나이다. 이 가상 네트워크는 일반적으로 암호화가 되어 있도록 만들어져 있고, 네트워크에 가입되어 있으며 서로가 알고 있는 개체들 (Entities) 간에서만 정보를 송수신 하도록 되어 있다. VPN은 재택근무하는 사람이 암호화된 가상사설망을 사용함으로 — 공공 인터넷을 통해서 — 회사 내부 네트워크에 연결하는 등으로 자주 사용되고 있다.

만약 여러분이 리눅스 마스커레이드를 쓰는 방화벽을 쓰고 있는데 MS PPTP (마이크로소프트의 VPN 포인트 투 포인트 연결용 제품) 패킷을 통과해야 하는 경우에 처한다면, 이 경우에 쓸 수 있는 리눅스 커널 패치가 있다. IP-masq-VPN을 보기 바란다.16.5 ([32. VPN 한글 정보])

리눅스용의 여러 VPN 솔루션이 있다.

-
- vpnd. *http://sunsite.auc.dk/vpnd/* `<http://sunsite.auc.dk/vpnd/>`.
- Free S/Wan. *http://www.xs4all.nl/~freeswan/* `<http://www.xs4all.nl/~freeswan/>`
- (점) ssh를 VPN의 구성으로 쓸 수 있다. VPN 미니-하우투를 보기 바란다.
- vps (Virtual Private Server: 가상 사설 서버). *http://www.strongcrypto.com* `<http://www.strongcrypto.com>`

여러 정보에 대한 사항을 담고 있는 6.3 (IPSEC 항목)도 참조하기 바란다.

제 9 절 온라인 전의 보안 준비 (접속에 앞서서)

좋다. 시스템에 대한 전반적인 검사가 끝났고 가능한 한 시스템을 안전하게 만들었다는 판단이 되었으면, 이제는 접속을 할 순서다. 하지만, 침입이 생길 경우 빠르게 침입자를 무능력화 시키고, 원상 복구를 하려면 준비를 해야 할 것이 아직도 몇 가지 있다.

9.1 완벽한 백업을 만들 것

백업 방법과 저장은 이 문서의 주제에서 벗어나는 일이지만 백업과 보안에 관하여서 몇 마디는 적어 놓겠다. 만약에 하나의 파티션에 650 MB 이하의 데이타를 가지고 있다면 CD-R을 쓰는 것 이 좋다. (내용을 변경하기가 쉽지 않고, 데이타의 장기간 저장이 가능하기 때문이다) 내용을 변경 당하는 경우가 없도록, 테이프와 다른 재사용이 가능한 것들은 백업이 끝나는 대로 쓰기 방지를 하도록 하고, 이에 대한 확인을 하는 것이 좋다. 백업은 안전한 제 2의 장소에 저장하도록 하라. 제대로 된 백업은 시스템을 다시 복구하는 시발점이 될 것이므로.

9.2 좋은 백업 스케줄을 고르기

예로서, 6개의 테이프를 1 주기용 한 묶음으로 쓰는 것이 관리하기에 편하다. 4개의 테이프를 매일 하나씩 주중에 사용하고, 한 개를 격주로 짝수 금요일에, 남은 한 개를 격주로 홀수 금요일에 사용하는 것을 방법이다. 매일 부분 백업 (incremental backup)을 만들고, 전체 백업을 적절한 (격주로 사용되는) 금요일 용 테이프에 만든다. 만약 특별히 중요한 변경을 했거나, 어떤 중요한 데이타를 추가했을 경우에는 특별히 백업을 만드는 것이 좋을 것이다. 16.5 ([33. 백업 스케줄])

9.3 RPM과 데비안 파일 데이타베이스의 백업

침입의 경우에는 RPM 데이타베이스를 트립와이어처럼 사용할 수 있겠지만, 이 경우에는 데이타베이스까지도 변경되지 않도록 확인을 하고 써야 할 것이다. RPM 데이타베이스를 플로피에 카피하고, 이 것을 항상 제 2의 장소에 보관한다. 데비안 배포본도 비슷한 것을 가지고 있다.

`/var/lib/rpm/fileindex.rpm`과 `/var/lib/rpm/packages.rpm`은 디스켓 한 장에 들어가지 않을 수 있다. 하지만 압축하면 각각 한 장씩에 들어갈 것이다.

만약 시스템이 침입을 당하면 다음의 명령어를 사용할 수 있다.

```
root#  rpm -Va
```

이것을 사용해서 시스템의 각 파일을 확인한다. 이 명령어는 다소 말이 많은 셈이니 (짧은 결과가 되도록 하는) 옵션들을 보기 위해서, rpm man 페이지를 보도록 하자. RPM 이진 파일들 또한 침입 당하지 않도록 확인을 해 두도록 하자.

이 방법을 쓰면 새로운 RPM을 더할 때마다 RPM 데이타베이스를 다시 복사해 두어야 한다. 장단점은 여러분이 결정해서 써야 할 것이다.

9.4 시스템 사용 정보 (account data)의 조사

시스로그 (시스템일지: syslog)의 정보가 침해를 받지 않도록 하는 것은 중요한 것이다. `/var/log` 파일을 오직 제한된 사용자들만이 읽고 쓸 수 있도록 하는 것은 좋은 시작이다.

`auth`항에 특히 주의를 두면서, 일지에 어떤 것들이 쓰여지고 있는 지를 감시하는 것이 좋다. 예를 들어서, 여러 번의 (연속된) 접속 실패는 침입 시도를 의미하는 것일 수 있다.

로그 문서 (log file: 일지 문서)가 어디에 위치하는 지는 여러분 배포본의 종류에 따라서 다르다. 레드 햇과 같이 "리눅스 파일시스템 규격 (Linux Filesystem Standard)을 따르는 배포본이라면, `/var/log`을 보면서 메시지를 확인하고, `mail.log`과 다른 것들을 보는 것이 좋다.

`/etc/syslog.conf`를 보면 여러분이 쓰는 배포본이 어느 장소에 무엇에 관한 일지 (log)를 적어 놓는지 볼 수 있다. 이 파일은 `syslogd` (시스템일지 데몬)에게 어떤 내용의 일지를 어떤 장소에 적어 놓아야 한다는 것을 알려주는 것이다. 시간이 날 때 검사할 있도록 – 여러분의 일지 보관 스크립트나 데몬을 조정해서 – 일지들을 오래 보관하도록 하는 것도 좋다. 최근의 레드 햇 배포본의 `logrotate` 패키지를 한 번 보라. 다른 배포본도 비슷할 것이 있을 가능성이 있다.

만약 여러분의 일지 문서가 이미 변경된 것처럼 보이면, 언제 변경이 시작되었는지, 어떤 내용이 변경되었는지 결정을 내리도록 노력해 보라. 일지가 오랜 시간 동안 비어 있는가? (만약 있다면) 백업 테이프에 있을 변경되지 않은 일지를 뒤져보는 것도 좋은 생각이다.

보통은 침입자가 침입 흔적을 없애기 위해서 바꿔 버리지만, 그래도 이상한 일들을 검사하기 위해서 일지를 확인하는 것이 좋다. 어쩌면 침입자가 침입을 시도하는 것이나, 루트 계정을 얻으려고 풀그림의 침탈을 (exploit a program) 시도하는 것을 알아챌 수 있다. 침입자가 채 바꾸기 전에, 일지를 볼 수도 있는 것이다.

`su`를 써서 사용자를 바꾸려는 시도나, 접속 시도나, 다른 사용자 계정 정보 등의 다른 일지 데이타 등에서 `auth`항은 별개로 하는 것이 좋다.

가능하다면, 가장 중요한 데이타는 보안이 철저한 시스템으로 복사본을 보내도록 `syslog`을 조정하라. 이 방법은 `/login/sy/ftp/etc` 시도를 지움으로서 흔적을 지우려는 침입자를 막을 것이다. `syslog.conf`의 man 페이지를 보고, `@` 옵션을 참조하도록.

syslogd보다 진보되어 있는 풀그림들도 있다. *http://www.core-sdi.com/english/freesoft.html* `<http://www.core-sdi.com/english/freesoft.html>`에서 시큐어 시스로그 (Secure Syslog) 풀그림을 한 번 보기 바란다. 시큐어 시스로그는 여러분의 시스로그 값을 암호화함으로서 이것을 누군가가 임의로 조작하는 일이 없도록 해준다.

많은 기능을 가진 또 다른 syslogd에는 *http://www.balabit.hu/products/syslog-ng.html* `<http://www.balabit.hu/products/syslog-ng.html>`에서 구할 수 있는 syslog-ng가 있다. 이 것은 여러분이 일지를 쓸 때에 보다 높은 융통성을 발휘해 주며, 원격 시스로그 스트림을 사용하여 침입을 막는다.

마지막으로, 아무도 읽지 않는 다면, 일지(日誌: Logs)라는 것은 무용지물이다. 시간을 만들어서 일지를 읽도록 하고, 평일 평상시의 작업이 일지에는 어떻게 적히는가를 이해하는 것이 좋다. 이것을 알아두는 것이 이상한 일을 알아채는 데 도움이 된다.

9.5 새로운 시스템 업데이트의 설치

대부분의 리눅스 사용자는 CD-ROM에서 설치를 한다. 보안 구멍 막이 작업은 그 주기가 빠르므로, 새로이 (고쳐진) 풀그림은 항상 나오고 있다. 네트워크에 기계를 연결하기에 앞서서, 배포본의 (ftp.redhat.com 등의) `ftp`에 가서 업데이트된 패키지를 받아서 새로 설치를 먼저 하는 것이 좋다. 이런 패키지는 중요한 보안 개선책을 자주 담고 있으므로, 설치를 반드시 하는 것이 좋을 것이다.

제 10 절 침입 도중이나 후에 할 일들

드디어 여기에 적혀 있는 (혹은 다른 곳의) 조언을 따른 결과로 침입을 감지해 내었다면? 첫 번째로 할 일은 침착성을 유지하는 것이다. 성급한 행동은 공격자가 저지를 수 있는 것보다 더 큰 해를 끼칠 수 있다.

10.1 보안 공격 진행 중!

진행 중인 보안 공격을 알아차리는 것은 긴장되는 일일 수 있다. 여러분이 어떻게 대응하는가에 따라 중요한 결과를 가져올 수 있다.

공격이 물리적인 것이라면, 누군가 여러분의 집이나 사무실, 연구실에 침입한 것을 감지한 경우일 것이다. 경찰에 알려야 한다. 연구실 환경에서 누군가가 케이스를 열려고 하거나 컴퓨터를 재부팅 하려고 하는 것을 여러분이 볼 수도 있다. 여러분의 권한과 절차에 따라, 여러분은 그들에게 중지하도록 요구하거나 지역 보안 책임자에게 연락할 수 있다.

지역 사용자가 보안을 훼손하고자 하는 것을 감지했을 경우, 가장 먼저 해야 할 일은 그 사용자가 실제 본인인지 확인하는 것이다. 그 사용자가 어디에서 로그인하려고 하고 있는지 확인해 보도록 하라. 그 곳이 평상시에 로그인해서 들어오는 곳인가? 그렇지 않은가? 다음에는 (이 메일을 쓰는 등으로) 컴퓨터를 통하지 않은, 직접적인 연락을 취해 보도록 하라. 예를 들어 전화를 걸거나 그 사용자의 집 혹은 사무실로 직접 찾아가서 이야기를 나눌 수 있다. 만일 그 사용자가 자기의 행위를 시인한다면, 그의 행위에 대해서 설명하도록 요구할 수 있고 그런 행위를 중지하라고 말할 수도 있다. 그가 부인하거나, 여러분이 말하는 사건에 대해서 모른다면 좀 더 조사를 해야 한다. 비슷한 사건들을 알아보고 고발이나 비난을 하기 전에 많은 정보를 확보하도록 하라.

네트워크를 통한 침투를 감지했다면, 처음 할 일은 (할 수 있다면) 네트워크 연결을 끊는 것이다. 침입자가 모뎀으로 접속했다면 모뎀 선을 뽑아 버리도록 하고, 이더넷을 통해 접속해 들어온다면 이더넷 선을 뽑아라. 이렇게 하면 침입자가 더 큰 피해를 입히는 것을 막을 수 있고, 침입자는 아마 자신이 들통났다고 생각하기보다는 네트워크에 문제가 생긴 모양이라고 여길 것이다.

여러분이 네트워크 연결을 끊을 수 없다면 (접속이 빈번한 사이트이거나, 컴퓨터에 대한 물리적 관리 권한이 없다면), 차선책은 침입자의 사이트로부터 접속해 들어오는 것을 막기 위해 tcp_wrapper나 `ipfwadm` 같은 풀그림을 사용하는 것이다.

침입자의 사이트에서 들어오는 모든 사람들의 접근을 거부할 수 없는 입장이라면, 사용자들의 계정을 폐쇄하여야 한다. 하나의 계정을 폐쇄하는 것은 쉬운 일이 아니라는 점에 주의하라. `.rhosts` 파일과 `ftp`를 통한 접근, 매우 많은 뒷문 (backdoor)의 가능성을 염두에 두어야 한다.

위의 조치들 (네트워크 절단, 공격자의 사이트로부터 오는 접근 시도 거부, 그리고/혹은 그들의 계정 폐쇄) 가운데 한 가지를 하고 나면, 공격자의 모든 사용자 프로세스를 죽이고 그들을 로그오프 시켜야 한다.

공격자는 다시 들어오려고 시도할 것이므로, 다음 몇 분 동안은 여러분의 사이트를 자세히 감시해야 한다. 공격자는 아마도 다른 계정을 쓸 것이고, 다른 네트워크 주소를 쓸 수도 있다.

10.2 보안 훼손이 이미 일어난 경우

이미 일어난 사고를 뒤늦게 겨우 알아차렸거나, (바라기로는) 감지된 공격자를 여러분의 시스템에서 잠가서 쫓아내 버렸다.. 이제는 무엇을 해야 할까?

10.2.1 개구멍 막아내기

공격자가 여러분의 시스템에 들어오기 위해 사용한 방법이 무엇인지 알 수 있다면, 그 구멍을 막도록 해야 한다. 예를 들어서, 어쩌면 침입자가 들어오기 바로 직전에 FTP 사용이 여러 번 보이는 것을 보았다고 하자. 이런 경우에는 FTP 서비스를 중지하고 개정 버전이나 알려진 교정 사항 목록이 있는지 찾아봐야 한다.

일지 문서들을 확인해 보고, 여러분의 보안 리스트와 페이지 사이트에 가서, 고쳐야 하는 새롭거나 잘 알려지고 있는 침탈법이 올려져 있는지 목록을 살펴보도록 하라. 칼데라 보안 수정안은 *http://www.calderasystems.com/support/security/* `<http://www.calderasystems.com/support/security/>`에서 구할 수 있다. 레드 햇은 아직 보안 수정안 목록을 버그 수정안에서 구분하지 않고 있지만, 배포본의 수정안 정보는 *http://www.redhat.com/corp/support/errata/* `<http://www.redhat.com/corp/support/errata/>`에서 구할 수 있다.

데비안은 웹에 보안 관련 메일링 리스트를 구성해 놓았다. *http://www.debian.org/security/* `<http://www.debian.org/security/>`

또한 한 제작자가 보안 수정안을 내면, 다른 대부분의 제작자 또한 곧 수정본을 내놓을 수 있다.

리눅스 보안 감사 프로젝트 (保安 監査: Linux Security Auditing Project)도 있다. 그들은 사용자 공간 도구들에서 보안 침탈법과 오버플로우들을 찾기 위해서 하나씩 철저히 검사하고 있는 작업을 하고 잇다. 그들의 공고문을 보면:

> "저희는 OpenBSD 정도로 보안이 잘 되도록 만들기 위해서 리눅스 소스를 체계적으로 감사하려는 시도를 하고 있습니다. 이미 여러 개의 문제점을 발견했고 (그리고 고쳐 놓은) 상황입니다만, 많은 도움이 필요합니다. 이 메일링 리스트는 조정자가 없고, 일반적인 보안 관련 토론에 있어서 쓸모 있는 수단이 되고 있습니다. 리스트의 주소는 security-audit@freet.lm.h.ox.ac.uk입니다. 가입하시고 싶으시면 security-audit-subscribe@ferret.lm.h.ox.ac.uk로 메일 주시기 바랍니다".

공격자를 완전히 차단하지 않으면, 그는 대개 다시 돌아온다. 여러분의 컴퓨터뿐 아니라, 여러분의 네트워크 안의 어딘 가로 말이다. 공격자가 패킷 스니퍼를 작동시키고 있었다면, 그는 지역 내의 다른 컴퓨터로 접근할 수 있다.

10.2.2 피해 평가

첫 번째 할 일은 피해를 평가하는 것이다. 무엇이 훼손되었는가? 트립와이어 같은 완전성 검사 툴그림을 사용하고 있다면, 트립와이어를 실행시켜서 무결성의 검사를 실시할 수 있다. 이런 툴그림이 없다면, 모든 중요한 자료들을 일일이 살펴보아야 한다.

리눅스 시스템은 갈수록 설치하기 쉬워지고 있으므로, 중요한 설정 파일들을 따로 저장해 두고 디스크를 아예 지워 버린 다음 리눅스를 처음부터 다시 설치한 뒤, 백업으로부터 사용자 파일과 설정 파일들을 복구하는 것을 고려해 볼 수도 있다. 이렇게 하면 깨끗한 시스템을 새로 갖게 된다. 만약 보안이 깨진 컴퓨터의 백업을 만든다면 – 침입자가 심어 둔 트로이의 목마일 수 있으므로 – 어떠한 이진 파일이라도 주의해서 보아야 할 것이다.

침입자가 루트 권한을 가지게 된다면, 재설치 작업은 반드시 실행되어야 할 작업 중의 하나로 인지되어야 한다. 덧붙여서, 여러분이 증거를 확보하려 한다면 여분의 디스크를 금고에 보관하는 것도 나쁜 것은 아닐 것이다.

그 뒷일로는 언제 침탈이 실행되었는지를 걱정해야 할 것이고, 그에 따라서 백업이 어느 정도나 손상된 데이타를 소지하고 있는가를 봐야 할 것이다. 백업에 대해서는 뒤에 더 설명하겠다.

10.2.3 백업, 백업, 그리고 또 백업!

정기적으로 백업을 해 두는 습관은 보안 문제에 있어서는 정말 중요하다. 시스템의 보안이 깨졌 을 때, 필요한 자료를 백업으로부터 복구할 수 있기 때문이다. 공격자에게 가치 있게 생각하는 자료는, 훔쳐서 자기의 사본을 만들어 둔 다음에 원본을 파괴하려 하겠지만, 최소한 원래의 자료를 도난 당할지언정 아주 잃어버리지는 않게 되는 것이다.

변조된 파일을 백업된 것으로 복구하기 전에, 이전의 여러 백업본들을 확인해 보아야 한다. 침입자가 파일을 오래 전에 망쳐 놓았다면, 이미 변조된 파일들만 잔뜩 백업해 놓았을 수도 있기 때문이다.

물론 백업본들에 대해서도 보안 문제가 있다. 백업본들을 안전한 장소에 두었는지 확인하여야 하고, 누가 거기 접근할 수 있는지 알고 있어야 한다. (공격자가 백업본을 얻을 수 있다면, 여러분이 모르는 사이에 여러분의 모든 자료에 접근할 수 있게 되는 것이다.)

10.2.4 침입자 추적

침입자를 몰아내고, 시스템을 복구했다고 해서 모든 일이 끝난 것은 아니다. 대개 침입자들은 잡히지 않지만, 그래도 공격 사건을 보고해야 한다.

공격자가 여러분 시스템을 공격하던 사이트의 관리자에게 그 사건을 알려주어야 한다. 연락처는 `whois`나 인터닉 데이타베이스를 이용해서 찾아볼 수 있다. 상관된 모든 일지 내용과 날짜 및 시간을 첨부해서 저쪽 관리자에게 메일을 보내는 것도 좋다. 침입자에 대해서 어떤 특이한 점을 발견했다면 그것도 함께 알려주도록 하라. 메일을 보낸 뒤에 (하고 싶다면) 전화를 써서 연락을 계속하도록 하라. 저쪽 관리자가 그 공격자를 찾아냈다면, 그 관리자가 다시 공격자가 들어온 사이트의 관리자에게 말하고 뭐 그렇다.

뛰어난 크랙커는 대개 많은 건너뛰기용 중간 시스템들을 사용한다. 이 시스템들 중의 어떤 (혹은 여러분의) 장소에서는 침입 당했다는 사실조차 모를 수도 있다. 크래커의 원래 시스템까지 좇아가는 것은 어려운 일이다. 여러분이 이야기하게 되는 관리자들에게 공손하게 대하는 것은 그들로부터 도움을 얻어내는데 좋다.

여러분이 관계되는 (*CERT* `<http://www.cert.org/>`나, 이와 비슷한) 모든 보안 조직들에도 알려주어야 한다.

제 11 절 보안 관련 자료

유닉스 보안 일반에 대한 혹은 특별히 리눅스 보안에 대한 훌륭한 사이트들이 정말 많이 있다. 하나 이상의

보안 관련 메일링 리스트에 가입해서 최신의 보안 수정 사항들을 따라가는 것은 매우 중요하다. 이런 리스트들은 대개 매우 분량이 적으면서도 유익하다.

11.1 FTP 사이트들

CERT는 컴퓨터 응급 대응 팀(Computer Emergency Response Team)의 약자다. 이들은 최근의 공격 사건과 수정 사항들에 대한 경보를 자주 발송한다. *ftp://ftp.cert.org/* <ftp://ftp.cert.org/>

http://www.zedz.net/ <http://www.zedz.net/>는 많은 보안 풀그림들을 저장하고 있다. 리플레이는 미국 안에 있지 않기 때문에 미국의 수출 제한 규정에 따르지 않는다.

Matt Blaze는 CFS의 저자이며 탁월한 보안 전문가이다. 매트 블레이즈의 아카이브는 *ftp://ftp.research.att.com/pub/mab* <ftp://ftp.research.att.com/pub/mab>에서 구할 수 있다.

tue.nl은 네덜란드에 있는 훌륭한 보안 관련 ftp 사이트이다. *ftp://ftp.win.tue.nl/pub/security/* <ftp://ftp.win.tue.nl/pub/security/>

11.2 기타 웹 사이트

- **핵커 FAQ**는 핵커들에 대한 FAQ이다. 핵커 FAQ: *http://www.tuxedo.org/~esr/faqs/* <http://www.tuxedo.org/~esr/faqs/>
- **COAST** 아카이브는 많은 유닉스 보안 풀그림과 정보를 가지고 있다. 코스트: *http://www.cs.purdue.edu/coast/* <http://www.cs.purdue.edu/coast/>
- **Suse** 보안 페이지: *http://www.suse.de/de/support/security/* <http://www.suse.de/de/support/security/>
- **Rootshell.com**은 크래커들이 요즘 쓰는 침투 방법에 대해 알아보기에 좋은 사이트이다. *http://rootshell.com/* <http://rootshell.com/>
- **BUGTRAQ**은 보안 관련 문제에 대한 상황 보고를 발표한다. BUGTRAQ archive: *http://www.securityfocus.com/* <http://www.securityfocus.com/>
- 컴퓨터 응급 대응 팀, **CERT**는 유닉스 시스템에 대해 흔히 가해지는 공격을 보고한다. *http://www.cert.org/* <http://www.cert.org/>
- **댄 파머 (Dan Farmer)**는 SATAN과 많은 다른 보안 도구들의 저자이며, 그의 홈 사이트에는 보안 도구들 뿐 아니라 보안에 대한 포괄적이며 흥미로운 글들이 있다. *http://www.fish.com* <http://www.fish.com>
- **리눅스 보안 WWW**는 리눅스 시스템의 보안에 대한 좋은 자료원이다. *http://www.aoy.com/Linux/Security/* <http://www.aoy.com/Linux/Security/>
- **Infilsec**에는 어떤 취약점이 특정한 플랫폼에 영향을 주는지 알려주는 취약점 엔진(vulnerability engine)이 있다. *http://www.infilsec.com/vulnerabilities/* <http://www.infilsec.com/vulnerabilities/>
- **CIAC**는 흔한 침입 사건들에 대해 정기적인 보안 보고서들을 보내 준다. *http://ciac.llnl.gov/cgi-bin/index/bulletins* <http://ciac.llnl.gov/cgi-bin/index/bulletins>
- 리눅스 **PAM** (장착 인증 모듈: Pluggable Authentication Modules) *http://www.kernel.org/pub/linux/libs/pam/* <http://www.kernel.org/pub/linux/libs/pam/>
- **데비안 프로젝트**는 데비안용의 보안 수정안과 정보를 가지고 있다. *http://www.debian.org/security/* <http://www.debian.org/security/>

- 링컨 스타인 (Lincoln Stein)이 쓴 **WWW 보안 FAQ**는 좋은 웹 보안 색인이다. *http://www.w3.org/Security/Faq/www-security-faq.htm* `<http://www.w3.org/Security/Faq/www-security-faq.htm>`

11.3 메일링 리스트

- **벅트랙 (Bugtraq)**: 벅트랙을 구독하려면, 본문에 *subscribe bugtraq*이라고 써서 *listserv@netspace.org* `<mailto:listserv@netspace.org>`로 메일을 보내면 된다. (아카이브)
- **CIAC**: 본문에 (제목에 말고) *subscribe ciac-bulletin*라고 써서 *majordomo@tholia.llnl.gov* `<mailto:majordomo@tholia.llnl.gov>`에 메일을 보내라.
- 레드 햇도 많은 메일링 리스트가 있고, 그 중 가장 중요한 것은 **레드 햇-어나운스** 리스트일 것이다. 이 곳을 통해서 보안과 (다른 것들도 포함해서) 관련이 있는 수정이 나오는 곧바로 발표된다. *redhat-announce-list-request@redhat.com* `<redhat-announce-list-request@redhat.com>`로 *subscribe redhat-announce*라고 써서 가입할 수 있다. *http://www.redhat.com/mailing-lists/redhat-announce-list/* `<http://www.redhat.com/mailing-lists/redhat-announce-list/>`에서 아카이브와 정보를 볼 수 있다.
- 데비안 프로젝트도 **보안 메일링 리스트**를 가지고 있다. *http://www.debian.org/security/* `<http://www.debian.org/security/>`

11.4 도서 목록

보안 관련 서적들은 정말 많이 있다. 이 항에서는 이런 책들 가운데 조금만 나열하고자 한다. 보안을 전문적으로 다룬 책들 뿐 아니라, 시스템 관리에 대한 많은 책들이 보안에 대해서 다루고 있다.

- Building Internet Firewalls By D. Brent Chapman and Elizabeth D. Zwicky 1st Edition September 1995 ISBN: 1-56592-124-0
- Practical UNIX and Internet Security, 2nd Edition By Simson Garfinkel and Gene Spafford 2nd Edition April 1996 ISBN: 1-56592-148-8
- Computer Security Basics By Deborah Russell and G.T. Gangemi, Sr. 1st Edition July 1991 ISBN: 0-937175-71-4
- Linux Network Administrator's Guide By Olaf Kirch 1st Edition January 1995 ISBN: 1-56592-087-2
- PGP: Pretty Good Privacy By Simson Garfinkel 1st Edition December 1994 ISBN: 1-56592-098-8
- Computer Crime A Crimefighter's Handbook By David Icove, Karl Seger and William VonStorch (Consulting Editor Eugene H. Spafford) 1st Edition August 1995 ISBN: 1-56592-086-4
- Linux Security By John S. Flowers New Riders; March 1999 ISBN: 0735700354
- Maximum Linux Security : A Hacker's Guide to Protecting Your Linux Server and Network Anonymous; Paperback - 829 pages Sams; July 1999 ISBN: 0672313413
- Intrusion Detection By Terry Escamilla Paperback - 416 pages (September 1998); John Wiley and Sons; ISBN: 0471290009
- Fighting Computer Crime Donn Parker; Paperback - 526 pages (September 1998); John Wiley and Sons ISBN: 0471163783

제 12 절 색인

- **인증 (Authentication)**: 받은 데이타가 실제의 보내진 데이타이며, 전송자라고 주장하는 사람이 실제의 전송자임을 확인하는 방법
- **보루 호스트 (Bastion Host)**: 인터넷 등에 접속되어 있는 이유 때문에 내부 네트워크 사용자의 주 접점이 되는 이유 등으로 – 그리고 공격에 노출되어 있기 때문에 – 보안을 특별히 강하게 만들어야 하는 컴퓨터. 보루 (bastion)는 성채 외곽 수비 목적으로 만든 (원형 모양의 수비용 탑 따위의) 돌출 수비 부분을 뜻한다. 보루는 중요한 지역을 내려다보면서, 강하고 두꺼운 수비용 벽 뒤에, 예비 병력이 있을 수 있는 공간과, 쓸모가 많던 끓는 기름통 등을 준비하고 있었다.
- **버퍼의 범람 (Buffer Overflow)**: 풀그림을 코딩할 때 버퍼가 "충분히 크도록" 만들지도 않고, 이러한 상태의 버퍼가 넘치는 것도 확인하지 않고 두는 경향이 있다. 이러한 상태의 버퍼가 범람을 하게 되면 (데몬이나 set-uid 등의) 실행되고 있는 풀그림이 다른 일을 하게끔 되게 된다. 보통 스택 함수의 리턴 주소를 다른 장소로 겹쳐 쓰게 하는 방법 등으로 작동된다.
- **서비스 거부 (Denial of Service)**: 서비스 거부는 공격자가 – 원래의 의도에 어긋나는 방법을 동원하는 등으로 컴퓨터가 자원을 소모하도록 유도해서 – 일반의 사용자가 합법적인 네트워크 자원을 쓰지 못하게 막는다.
- **듀얼-홈드 호스트 (Dual-homed Host)**: 최소한 두 개의 네트워크 인터페이스를 가지고 있는 일반 목적의 컴퓨터 시스템.
- **방화벽 (Firewall)**: 보호된 (혹은 보호하는) 네트워크와 인터넷 사이, 혹은 다른 종류의 네트워크들 사이의 엑세스를 관리하는 구성 요소.
- **호스트 (Host)**: 네트워크에 연결된 컴퓨터.
- **IP 스푸핑 (IP Spoofing)**: IP 스푸핑은 여러 요소를 구성해서 쓰는, 복잡하면서도 기술적인 공격법이다. 이것은 상호 신용 관계 (trust-relationship)에 있는 컴퓨터들을 속이는 침탈법이다. *프랙 매거진 (Phrack Magazine)* `<http://www.phrack.com>` 제 7호 48권을 보면 데몬9, 라우트, 그리고 인피니티라는 이름을 가진 핵커들이 쓴 자세한 보고서가 있다.
- **논-레퓨디에이션 (Non-repudiation)**: 데이타를 보낸 전송자가 – 나중에 데이타를 전송했다는 사실을 부인하는 경우 등을 대비해서 – 데이타 수신자가 실제 전송자 인증을 실행할 수 있도록 해주는 기능.
- **패킷 (Packet)**:인터넷 통신의 최소 기본 단위.
- **패킷 필터링 (Packet Filtering)**:네트워크 상에서 오가는 데이타의 흐름을 선택적으로 조정하는 어떤 기기의 동작. 패킷 필터는 패킷을 흐르게 하거나 막는데, 보통 한 네트워크에서 다른 네트워크로 라우팅을 하는 과정에서 (주로 인터넷과 내부 네트워크 사이) 작업을 실시한다. (특정된 IP 주소나 포트에서 들어오는 등의) 정해 놓은 종류의 패킷은 허용되고 어떤 다른 종류의 패킷은 통제하는 등의 규칙을 만들어 놓게 된다.
- **외곽 수비 네트워크 (Perimeter Network)**: 보호하고 있는 (보호된) 네트워크와 외부 네트워크 사이에 –보안 안전의 층을 두껍게 하기 위해서 덧붙여져 만들어진 네트워크. 외곽 수비 네트워크는 DMZ 이라고도 불린다.
- **프락시 서버 (Proxy Server)**: 내부 사용자를 위해서 외부 서버와 관여하는 풀그림. 프락시 클라이언트는 프락시 서버와 통신을 하게 되고, 프락시 서버는 허가 받은 클라이언트 (고객 혹은 사용자)를 실제의 서버에 중계해 준다.
- **수퍼유저 (Super User)**: root를 일반적으로 부르는 말.

제 13 절 FAQ

1. 드라이버 지원을 모듈로 만드는 대신 커널에서 직접 하도록 하는 것이 더 안전 한가요? 어떤 사람들은 – 침입자가 트로이의 목마형 모듈이나 시스템에 영향을 끼칠 수 있는 모듈을 로드할 수 있다고 해서 – 모듈을 써서 디바이스 드라이버를 로드하도록 하는 기능은 꺼 놓는 것이 좋다고 생각하기도 합니다.

   하지만 모듈을 로드하기 위해서는 사용자는 반드시 루트가 되어야 합니다. 모듈 오브젝트 파일 또한 루트만이 적을 수 있도록 되어 있습니다. 만약 침입자가 루트 사용권을 이미 얻었다면, 그가 모듈을 로드할 것인가를 걱정하는 것 말고도 중요한 다른 걱정거리가 많겠지요.

   모듈은 자주 사용되지 않기도 하는 특정한 디바이스를 지원하기 위해서 로드되게 됩니다. 서버 기계에서나 – 방화벽 등의 경우– 자주 일어나지는 않는 일입니다. 이런 이유로 – 서버로 일하고 있는 기계에는 – 커널에 직접 지원을 컴파일해 주는 것이 타당합니다. 또한 모듈은 직접 커널에 컴파일된 지원보다 느립니다.

2. 왜 원격 기계에서 루트로 접속하는 것이 항시 안됩니까? 4.2 ([루트 보안])에 대한 항목을 보십시오. 원격 사용자가 텔넷을 써서 루트로 접속하는 것은 루트의 암호가 평문으로 전송되는 등의 심각한 보안 취약 요건이기에 이런 경우를 막고자 일부러 이렇게 해 놓습니다. 잠재적 침입자에게는 시간이 충분하고, 여러분의 패스워드를 찾아내기 위한 자동 풀그림을 쓸 수도 있다는 것을 언제나 잊지 마십시오.

3. 제가 쓰고 있는 레드 햇 4.2나 5.0 컴퓨터에서 어떻게 쉐도우 패스워드를 씁니까? 쉐도우 패스워드를 쓰려면, 우선 루트의 권한을 가지고서 `pwconv`을 먼저 쓰십시오, 그러면 `/etc/shadow`가 생기게 됩니다. 만약 레드 햇 4.2 이상을 쓰는 경우에는 PAM 모듈이 `/etc/passwd`에서 쉐도우 패스워드로 변환되는 것을 자동으로 감지해서 사용하게 해 줄 것입니다.

   쉐도우 패스워드는 일반적으로 사용되는 /etc/passwd 파일이 아닌 다른 장소에 패스워드를 적어 놓는 방식입니다. 여러 장점이 있는데, 우선은 /etc/shadow 등의 쉐도우 파일은 – 아무나 읽을 수 있도록 되어 있는 /etc/passwd와는 달리 – 루트만이 읽을 수 있도록 되어 있다는 것입니다. 다른 장점은 관리자로서 – 사용자들이 다른 계정의 상태를 아는 일없이 – 계정을 주거나 꺼 버리는 것이 가능하다는 것입니다.

   이 경우에 /etc/passwd 파일은 – /bin/ls 등의 풀그림 등이 디렉토리를 보여줄 때의 경우처럼 사용자 ID와 사용자 이름 (username)을 연결해서 사용할 수 있도록 – 사용자와 그룹 이름을 기록하는 것에만 사용됩니다.

   그리고 /etc/shadow 파일은 사용자 이름 (username)과 패스워드 – 그리고 어쩌면 계정이 언제 끝나는지 등의 – 계정 정보만을 담게 됩니다.

   쉐도우 패스워드를 켜 두려면 루트 권한으로 pwocnv를 실행하시면 /etc/shadow가 만들어지면서 다른 풀그림들에 사용되게 됩니다. 레드 햇 4.2 이상을 사용하신다 했으므로, 이 경우에는 특별히 무엇을 바꾸지 않아도 PAM 모듈이 보통의 /etc/passwd에서 쉐도우 패스워드로 전환 사용되는 것을 자동으로 인식할 것입니다.

   일단 패스워드를 안전하게 만드는 것에 관심을 두셨으니, 아예 처음부터 좋은 패스워드를 만드는 것에 관심을 두는 것도 좋을 것입니다. 이 것을 위해서는 PAM의 일부분인 pam_cracklib을 쓰실 수 있습니다. 이것은 여러분의 패스워드를 크랙 라이브러리를 써서 돌려봄으로서 여러분의 패스워드가 패스워드 크랙 풀그림에 의해서 쉽게 깨질 수 있는 가를 알도록 해줍니다.

4. 어떻게 아파치 SSL 익스텐션을 사용할 수 있지요?

   (a) 우선 *ftp://ftp.psy.uq.oz.au/pub/Crypto/SSL* `<ftp://ftp.psy.uq.oz.au/pub/Crypto/SSL>`에서 SSLeasy 08.0 이상의 것을 구합니다.

   (b) 빌드하고, 실험해 본 후에 설치를 합니다.

   (c) 아파치 1.2.5의 소스를 구합니다.

   (d) 아파치 SSLeay 익스텐션을 구합니다. *ftp://ftp.ox.ac.uk/pub/crypto/SSL/apache_1.2.5+ssl_1.13.tar.gz* `<ftp://ftp.ox.ac.uk/pub/crypto/SSL/apache_1.2.5+ssl_1.13.tar.gz>` 16.5 ([35])

(e) 아파치 1.2.5 소스 디렉토리에 압축된 것을 풀어서 README에서 말하는 대로 수정해 줍니다. (README를 먼저 읽고 나서 실시하세요)

(f) 조정을 하고 빌드하면 끝. 미국의 밖에 소재하고 있는 *예전의 리플레이 어소시에이츠 (Replay Associates): http://www.zedz.net/* `<http://www.zedz.net/>`에서 미리 만들어진 패키지를 구할 수도 있습니다.<<

>>

(g) 보안 수준을 유지하면서 어떻게 사용자 계정을 바꾸죠? 특히 RH 5.0 등의, 레드 햇 배포본은 사용자 계정의 특성을 바꾸는데 사용할 수 있는 많은 도구를 가지고 있습니다.

- pwconv와 unpwconv 풀그림을 사용하시면 쉐도우 패스워드와 비(非) 쉐도우 패스워드로 원하는 대로 바꾸면서 사용할 수 있습니다.
- pwck과 grpck은 passwd와 그룹 파일간의 적절한 구성을 확인하는 것에 사용될 수 있습니다.
- useradd, usermod, 그리고 userdel는 사용자 계정을 더하고 바꾸고, 지우는 데에 사용됩니다. groupadd, groupmod, 그리고 groupdel은 그룹에 적용되는 명령어입니다.
- 그룹 패스워드는 gpasswd 명령어를 써서 만들 수 있습니다.

여기 적혀 있는 모든 풀그림들을 "쉐도우 인지 (認知)"를 합니다. 만약 쉐도우를 켜 놓으면, /etc/shadow를 패스워드 정보를 보기 위해서 사용할 것이고 아니라면, 단순히 사용하지 않는다는 것을 뜻합니다. 더 정보를 원하시면 상관되는 매뉴얼 페이지 man을 보십시오.

(h) 특정한 HTML 파일을 어떻게 아파치를 써서 보호합니까? *http://www.apacheweek.com/* `<http://www.apacheweek.com/>`이라는 곳을 아시는지?

사용자 인증에 대해서는 *http://www.apacheweek.com/features/userauth* `<http://www.apacheweek.com/features/userauth>`를 보고, 다른 웹 서버 보안 팁은 *http://www.apache.org/docs/misc/security_tips.html* `<http://www.apache.org/docs/misc/security_tips.html>`을 보십시오.

제 14 절 결론

보안 경보 메일링 리스트에 가입하고 최신 동향에 따라감으로써 여러분은 여러분의 컴퓨터를 안전하게 하기 위해 많은 일을 할 수 있다. 기록 파일들에 주의를 기울이고, 트립와이어 같은 풀그림들을 정기적으로 수행하면 더 많은 일들을 할 수 있다.

집안의 컴퓨터에서는 적절한 수준의 컴퓨터 보안을 유지하기가 어렵지 않다. 사업용 컴퓨터에서는 더 많은 노력이 필요하지만, 리눅스는 실로 안전한 플랫폼이 될 수 있다. 리눅스의 개발 특성상, 상업용 운영체제보다 더 빨리 보안 관련 수정 사항들이 나온다. 따라서 보안이 필요하다면 리눅스야말로 이상적인 플랫폼이다.

제 15 절 감사의 글

이 글에서 소개된 정보들은 많은 자료원들로부터 모은 것이다. 직접적으로 혹은 간접적으로 도움을 준 아래의 사람들에게 감사한다.

Information here is collected from many sources. Thanks to the following that either indirectly or directly have contributed: following who either indirectly or directly have contributed:

Rob Riggs *rob@DevilsThumb.com* <mailto:rob@DevilsThumb.com>

S. Coffin *scoffin@netcom.com* <mailto:scoffin@netcom.com>

Viktor Przebinda *viktor@CRYSTAL.MATH.ou.edu* <mailto:viktor@CRYSTAL.MATH.ou.edu>

Roelof Osinga *roelof@eboa.com* <mailto:roelof@eboa.com>

Kyle Hasselbacher *kyle@carefree.quux.soltc.net* <mailto:kyle@carefree.quux.soltec.net>

David S. Jackson *dsj@dsj.net* <mailto:dsj@dsj.net>

Todd G. Ruskell *ruskell@boulder.nist.gov* <mailto:ruskell@boulder.nist.gov>

Rogier Wolff *R.E.Wolff@BitWizard.nl* <mailto:R.E.Wolff@BitWizard.nl>

Antonomasia *ant@notatla.demon.co.uk* <mailto:ant@notatla.demon.co.uk>

Nic Bellamy *sky@wibble.net* <mailto:sky@wibble.net>

Eric Hanchrow *offby1@blarg.net* <mailto:offby1@blarg.net>

Robert J. Berger*rberger@ibd.com* <mailto:rberger@ibd.com>

Ulrich Alpers *lurchi@cdrom.uni-stuttgart.de* <mailto:lurchi@cdrom.uni-stuttgart.de>

David Noha *dave@c-c-s.com* <mailto:dave@c-c-s.com>

Pavel Epifanov. *epv@ibm.net* <mailto:epv@ibm.net>

Joe Germuska. *joe@germuska.com* <mailto:joe@germuska.com>

Franklin S. Werren *fswerren@bagpipes.net* <mailto:fswerren@bagpipes.net>

Paul Rusty Russell <*Paul.Russell@rustcorp.com.au*> <mailto:Paul.Russell@rustcorp.com.au>

Christine Gaunt <*cgaunt@umich.edu*> <mailto:cgaunt@umich.edu>

lin *bhewitt@refmntutl01.afsc.noaa.gov* <mailto:bhewitt@refmntutl01.afsc.noaa.gov>

A. Steinmetz *astmail@yahoo.com* <mailto:astmail@yahoo.com>

Jun Morimoto *morimoto@xantia.citroen.org* <mailto:morimoto@xantia.citroen.org>

Xiaotian Sun *sunx@newton.me.berkeley.edu* <mailto:sunx@newton.me.berkeley.edu>

Eric Hanchrow *offby1@blarg.net* <mailto:offby1@blarg.net>

아래의 사람들은 이 하우투 문서를 여러 다른 말로 번역을 해 주었다! 리눅스의 복음을 전파해 준 여러분께 특별한 감사를 드린다.

The following have translated this HOWTO into various other languages! A special thank you to all of them for help spreading the linux word...

Polish: Ziemek Borowski *ziembor@FAQ-bot.ZiemBor.Waw.PL* <mailto:ziembor@FAQ-bot.ZiemBor.Waw.PL>

Japanese: FUJIWARA Teruyoshi *fjwr@mtj.biglobe.ne.jp* <mailto:fjwr@mtj.biglobe.ne.jp>

Indonesian: Tedi Heriyanto *22941219@students.ukdw.ac.id* <mailto:22941219@students.ukdw.ac.id>

Korean: Bume Chang *bschang@kldp.org* <mailto:bschang@kldp.org>

Spanish: Juan Carlos Fernandez *piwiman@visionnetware.com* <mailto:piwiman@visionnetware.com>

Dutch: R. Ekkebus *reggy@zeelandnet.nl* <mailto:reggy@zeelandnet.nl>

제 16 절 번역자의 덧붙이는 말

16.1 원문에 대한 설명

이 보안 하우투 v1.1.0의 원문은 작년에 번역을 올렸던 v1.0.8의 원문과 전혀 다르지 않습니다. 문장을 좀 더 매끄럽게 고치고, 철자와 문법을 수정해 놓고, URL 등의 부분적인 수정을 한 것을 제외하면 말입니다.

16.2 읽으실 때에

원문에 참조하라고 적혀 있는 문서들은 모두 영어로 되어 있고, 주소도 바뀐 경우도 많습니다. 번역을 하면서 원문에 나와 있는 주소를 그대로 사용했기 때문에, 본문에 URL로 나와 있는 문서들은 모두 영문으로 되어 있습니다. 한글로 되어 있는 문서도 이미 많이 있으니 원하시면 한글 검색 엔진을 사용해서 찾아보십시오. 또한 이 글에 나와 있는 대부분의 하우투와 미니-하우투 문서들은 KLDP에서 한글로 번역된 것을 구하실 수 있습니다. *http://kldp.org* `<http://kldp.org>`

중요하거나 의미가 여럿인 단어의 번역에 대한 설명을 아래에 써 놓았습니다.

16.3 메일링 리스트

- **보안 관련 메일링 리스트**: 한국 정보 보호 센터 *http://www/certcc.or.kr* `<http://www.certcc.or.kr>` 웹 사이트의 "기술 권고문 (Korean Advisory)" 항목에서 우리 나라 KISA 및 호주 CERT와 미국 에너지 성에서 발행한 보 안 경고문서 (Security Advisory)의 아카이브를 읽어보실 수 있습니다. 또한 새로운 보안 경고 및 기술 권고가 발행될 때마다 메일로 받아 볼 수 있도록 *http://www.certcc.or.kr/certcc/index.html#mail* `<http://www.certcc.or.kr/certcc/index.html##mail>`에서 여러분의 메일 주소를 등록을 하면 앞으로 발행될 문서를 메일로 받아 볼 수 있을 것입니다.

- **KISA**: 보안 권고문 이외의 보다 넓은 영역의 보안 정보를 원한다면 한국 정보 보호 센터의 또 다른 도메인인 *http://www.kisa.or.kr* `<http://www.kisa.or.kr>`도 참고하시기 바랍니다.

- **NIST 보안 문서의 한글판**: 특히, KISA의 웹 사이트에 가 보시면 *htttp://www.kisa.or.kr/edu/nist/nist-목차.htm* `<http://www.kisa.or.kr/edu/nist/nist-.htm>`에서 미국 NIST가 특별 배포한 문서 중의 하나인 "컴퓨터 보안을 위한 NIST 핸드북 (800-12)"의 한글판 문서를 보실 수 있습니다. 이 문서는 여러분이 지금 읽고 있는 이 "(리눅스) 보안 하우투 문서"와 내용이 어느 정도 중복되어 있기는 하지만, 사용자와 관리자 모두가 읽도록 저술이 되어 있음과 동시에 보안을 매우 중요시하는 한 정부 기관에서 가이드용으로 제작된 문서인 만큼, 읽어볼 가치가 매우 높습니다. 위의 방화벽 항목에서 언급되었던 NIST 방화벽 문서를 비롯한 기타 NIST 원문들은 *http://csrc.nist.gov/publications/welcome.html* `<http://csrc.nist.gov/publications/welcome.html>`의 Special Publication 항목에서 보실 수 있습니다.

- **UGU 메일링 리스트**: 이 메일링 리스트에 가입을 하시면 유닉스의 운용에 관한 팁을 하루에 하나씩 메일로 받아 보실 수 있습니다. 영문이고 보안에 관련된 메일링 리스트라기 보다는 유닉스의 운용에 대한 메일링 리스트지만, 보안에 관련된 팁이 종종 등장하기도 하고, 매일 하나씩 팁을 받아볼 수 있다는 재미가 있습니다. 또한 이 웹 사이트에서 일반적인 보안 이슈를 비롯한 다른 모든 유닉스에 대한 전반적인 정보를 구하실 수 있습니다. *http://www.ugu.com/sui/ugu/show?ugu* `<http://www.ugu.com/sui/ugu/show?ugu>`의 "Help Me" 섹션에서 "Daily Unix Tip"을 선택하시고, 구독을 신청하시면 됩니다.

16.4 번역 단어

- **File**: 의미에 따라서 크게 두 가지의 뜻이 있었습니다. 하나는 이진 파일 (바이너리) 등의 컴퓨터 "파일"이라는 뜻이었고, 다른 하나는 에디터 등으로 읽고 이해할 수 있는 (하우투 문서 등의) "문서"를 뜻했습니다. 일반적으로 전자의 경우는 "파일"로 번역했고 후자의 경우는 "문서"로 번역했습니다. 하지만, `lilo.conf` 등의 시스템 파일이면서도 에디터로 읽는 경우를 설명하는 때에는 편의상 "문서"나 "파일"을 섞어 가면서 번역했습니다.

- **Implementation**: 6.3의 IPSEC "Implementation"을 "구성"이라고 번역했으며, 구성된 풀그림은 "구성본"이라고 번역했습니다.

- **Cleartext**: 암호화되기 전의 파일이나 문서를 뜻하는 Cleartext는 "보통 문장"이라는 뜻으로 평문이라고 번역했습니다. 반대말로는 "암호문"을 썼습니다.
- **Encryption**: 일반적으로 인크립션 (Encryption) 작업을 암호화라고 썼습니다만, 경우에 따라서는 "패스워드를 만드는 일"도 "암호화"라고 썼습니다.
- **Decryption**: 인크립션된 파일을 푸는 디크립션 (Decryption) 작업을 "복호화"라고 번역했습니다.
- **Secure**: "보안", "통신 보안", "안전"으로 등으로 문맥에 따라서 번역했습니다.

16.5 번역자 색인

- 1 **Exploit**: "침탈" 이라고 번역했다
- 2 **핵커 하우투**: *http://kldp.org/~kabin/doc/hacker-howto.htm* `<http://kldp.org/~kabin/doc/hacker-howto.htm>`에 안창선 님이 번역한 한글판이 있음.
- 3 **콜백 모뎀**: 모뎀 접속 시에 사용하는 경우의 서버 보안법의 하나.
  - **방법의 예**: 사용자는 자신이 사용하고 있는 모뎀의 전화번호를 로그인 값과 함께 서버로 보내 주고 일단 일차 접속을 끝낸다.. 서버는 그 전화번호를 사용자의 계정 정보에 있는 전화번호와 대조를 한 후에, 일치하면 그 전화번호로 (가능하면 발신 전용 모뎀과 발신 전용선 등을 사용해서) 전화를 되 걸어서 (= 콜백을 해서) 이차적 접속을 한다. 위의 예처럼 콜백 모뎀 서버는 사용자의 전화번호를 고정시켜서 처리할 수도 있고, 사용자가 아무 장소에서나 연결할 수 있도록 자유롭게 콜백 수신 번호를 풀어놓을 수도 있는 옵션 등을 지정해서 편의에 어울리게 구성할 수 있다.
  - **장점**: 사용자가 전화비를 절감함과 동시에 사용자 인증의 기초적인 부분으로 쓸 수 있다.
  - **단점**: 서버가 접속 전화 비용을 부담한다. 그리고 콜백 모뎀 방법에도 제한적이나마 침탈법이 존재한다.
    * 영문 링크: *리눅스용 콜백 셋업: http://www.icce.rug.nl/docs/programs/callback/callback.html* `<http://www.icce.rug.nl/docs/programs/callback/callback.html>`
    * 영문 링크: 모뎀 하우투 v.0.10 *http://ftp.dei.uc.pt/LDP/HOWTO/Modem-HOWTO.html* `<http://ftp.dei.uc.pt/LDP/HOWTO/Modem-HOWTO.html>`
- 4 **www.internic.net**: 원래 이러한 문서를 저장 관리해 오던 인 터닉.넷은 네트워크 솔루션스로 본격 상업화 된 얼마 후부터 더 이상 RFC, ID, FYI 문서를 다루 지 않기로 했다. *http://www.isi.edu/in-notes/rfc2196.txt* `<http://www.isi.edu/in-notes/rfc2196.txt>`에서도 구할 수 있다.
- 5 주인 없는 기계 (**rogue** machine): 관리자가 아예 없거나 관리가 전혀 안되어 있으면서도 네트워크에 연결되어 있는 기계. (홀로 서기?)
- 6 **linux single**: 만약 리눅스가 부팅할 때 부트 프롬프트에서 "lilo: single"로 부팅을 하면 1인용 싱글 유저 환경으로 들어가게 된다. 이 상황에서의 보안상의 문제는 컴퓨터가 패스워드를 안 물어 보면서 부팅이 된다는 것이다. 비록 이 싱글 유저 환경에서는 파일시스템들이 "읽기 전용"으로 마운트되지만, 일단 침입자가 여기까지 들어왔으면 읽기 전용의 파일시스템들을 "쓰기 OK"로 고쳐서 다시 마운트하는 것은 단순한 실력 문제일 것이다. 이러한 상황을 막으려면 다음과 같은 방법을 쓴다.
  - **Password**: 예로서, /etc/lilo.conf 문서에 password=newpass라고 쓴 후에 릴로를 다시 설치하면, 부트를 할 때마다 항시 릴로에서 newpass라는 암호를 넣어 주어야 만 된다. 물론 이 패스워드는 여러분이 원하는 것으로 정하도록 한다.
  - 만약 부트 때마다 패스워드를 넣어야 하는 것이 싫다면, 위의 방법 대신에 릴로의 막고자 하는 부트 옵션의 라벨 (Label) 앞에 restricted라고 써 놓으면 된다.
  - 덧붙여서, 패스워드가 적혀 있게 되는 lilo.conf 문서를 절대로 "모두 볼 수 있도록 (world readable)" 설정하면 안된다는 것일 것이다. 만약 모두 볼 수 있도록 한다면 애써 적어 놓은 패스워드를 아무나 볼 수 있을 것이므로.

- 7 **로그 파일**: log file: 日誌 文書.
- 8 **cleartext**: 평문: 암호화되기 전의 보통 문장. x
- 9 **비파괴적 명령어와 에일리어스의 구성**: 본인의 경우에는 rm에 대한 에일리어스를 "alias rm='rm -i'로 만들어 놓았다. 물론 사용자가 일부러 "rm -f"를 쓰는 경우를 막을 수는 없겠지만 말이다.
  - 사용자들이 파괴적이면서도 복잡한 명령어를 쓸 때, 비파괴적인 (non-destructive) 명령어를 먼저 사용할 수 있도록 컴퓨터를 구성할 수 있을 것이다. 이것은 특히 별표를 써서 쓰는 명령어를 쓸 때의 실수를 방지하기 위한 방법이 될 수 있다. 예를 들면, "rm foo*.bak:을 쓰기에 앞서서 "ls foo*.bak"의 디렉토리 목록을 프롬프트 하도록 만듦으로서 사용자가 지우고자 하는 문서만을 지우게 되는 것인가를 확인하도록 만드는 것 따위이다. echo 명령어를 파괴적 명령어의 앞부분에 써서 확인을 하는 것도 도움이 된다.
- 10 **트로이의 목마**: 만약 누군가가 현재의 디렉토리 "."에 "su"나 "ls"라는 이름을 가진 트로이의 목마를 심어 놓는다면?
- 11 **umask와 십진 전수 (Octal complement)**: 유닉스 파일에는 "소유자(user), 그룹(group), 그 외(other)"라는 세 종류의 사용자가 항시 존재한다. 각 사용자 유형에는 다시 "읽기 허가권, 쓰기 허가권, 실행 허가권" 의 세 가지 허가권이 항시 정의되어 있다. (5.2 (파일 허가권 참조)). 각 사용자 유형의 세 가지 허가권은 이진수로 표현되어 있는데, 최소 값인 이진수 000 (읽기, 쓰기, 실행권 없음)에서부터 최대 값인 이진수 111 (세 가지 모두 있음)까지다. (최고 값인 이진수 111은 십진수로 7이므로 어떤 파일에 대한 허가권이 "111.111.111" 이라면 이것은 일반적으로 십진수 "777"로 부른다). umask는 새로 만들어질 파일과 디렉토리들의 디폴트 소유권 값을 정할 때 사용한다. umask에는 새로 만들어지는 모든 파일의 기준 값에 반한 십진 전수 (十進 全數)를 사용한다. 예로서 한 사용자의 umask 값을 022에서 077로 바꿔 주면, 이 순간부터 앞으로 이 사용자가 만드는 파일과 디렉토리는 111.000.000의 가장 제한적인 허가권을 자동으로 가지게 된다.

```
umask와 십진 보수의 예:

파일 값            십진수       십진 전수
-.uuu.ggg.ooo       u.g.o       u.g.o         umask
-.rwx.rwx.rwx
0.111.111.111       7.7.7       0.0.0         umask 0
0.111.111.100       7.7.4       0.0.3         umask 3
0.111.101.101       7.5.5       0.2.2         umask 22
0.111.000.000       7.0.0       0.7.7         umask 77
```

- 12 **파일 허가권의 십진수 사용**: 예를 들어서, 이진수 "110.100.000"은 십진수 "640"으로 표시된다. (이것은 "rw-.r–.—"이다). 이진수인 파일 허가권을 "640" 등의 십진수로 부르는 것은 초보자로서는 헷갈리는 것이겠지만, "110.100.000" 등으로 부르는 것보다 "640"으로 부르는 것이 훨씬 편하지 않은가.
- 13 **국제판 PGP**: 원문의 URL은 죽어 있었다. 어차피 미국의 수출 제한 조치 때문에 미국판 PGP를 쓰지 못하므로, 국제판인 PGPi의 FAQ를 적어 놓았다. *http://www.pgpi.org/doc/faq/* `<http://www.pgpi.org/doc/faq/>`. 미국판 PGP는 *http://www.pgp.com/* `<http://www.pgp.com/>`에서 정보를 구할 수 있다,
- 14 **PGP의 국제판**: 미국 밖에서 사용을 하려면 인터내셔널 버전을 쓰면 된다. 최신의 버전은 *http://www.pgpi.com* `<http://www.pgpi.com>`이나 *http://www.pgpi.org/* `<http://www.pgpi.org/>`에서 구할 수 있다.
- 15 **Dead URL**: 원문의 URL인 http://home.netscape.com/assist/security/smime/overview.html은 죽어 있어서, 대신 http://www.rsasecurity.com/standards/smime/을 넣었다.
- 16 **최신판**: 1999/10/15일에 1.1판이 나와 있었다. *ftp://ftp.xs4all.nl/pub/crypto/freeswan/freeswan-1.1.tar.gz* `<ftp://ftp.xs4all.nl/pub/crypto/freeswan/freeswan-1.1.tar.gz>`

- 17 **Session Hijacking의 예**: 중계인 공격 (man-in-the middle attach): A가 B로 보내는 정보를 M이 눈에 보이지 않게 중간인 역할을 해서 정보를 가로채는 방법. A는 B로 정보를 보낸다고 생각하고, B는 A가 정보를 보낸다고 생각하지만, 실제로는 M이 중간에서 A에게는 B인척 위장을 하고, B에게는 A인 척 위장을 하는 방법으로 데이터를 가로챈다.

- 18 **psst**: psst 풀그림은 우리말로 조용히 하라고 할 때 쓰는 "쉬!"에 해당하는 의성어 "Psst!"에서 이름을 따온 듯하다. 현재에는 lsh라고 다시 이름이 지어져 서 개발되고 있다.

- 19 **URL**: (*http://www.consensus.com/faqs/tls_ssl_faq.txt* `<http://www.consensus.com/faqs/tls_ssl_faq.txt>`)을 보면 알겠지만, 유럽판을 구해서 쓸 수 있다.

- 20 **흡사한 작동을 하는 풀그림과 프로토콜**:

  - **NT 사용자**: 여러분이 네트워크 시스템 관리자이며 NT 기계도 같이 관리하고 있다면, 새로운 윈도우스 2000의 EFS (Encrypting File System)를 참조하기 바란다. EFS의 내부 작동 방법은 CFS와 TCFS의 방법과 매우 흡사하다. 비록 영문이지만 EFS에 대한 좋은 정보는 *http://www.ntmag.com/Articles/* `<http://www.ntmag.com/Articles/>`에서 **EFS**를 **키워드**로 검색을 하거나, EFS의 전문가인 **주베어 아메드 (Ahmed)**나 **마크 러시노비치 (Russinovich)** 등을 **Author**에서 골라서 검색을 하면 얻을 수 있다. 앞으로 쓰게 될 (윈도우스 2000의) EFS의 기술적인 내용은 위에서 얻을 수 있을 것이며, NT 3.51과 NT 4.0 (sp3 이상)을 쓰는 경우에는 제3의 장소에서 파일과 디렉토리 암호화 풀그림을 따로 구입해서 (돈주고... -_-;) 써야 한다. (비록 www.winfiles.com 등에서 무료로 사용할 수 있는 NT용 암호화 풀그림을 구할 수 있겠지만, 그 것이 XOR 등을 쓰는 황당한 것인지는 여러분이 책임을 져야 할 문제일 것이다).
  - **윈도우스 95/98 사용자**: 하드 드라이브의 일부를 파티션 파일로 만들어서 암호화해 주는 **스크램디스크 (Scramdisk)**를 권장하고 싶다. RSA의 IDEA를 쓰기 때문에 상업용 사용은 금지되어 있지만, 개인적 사용의 경우에는 무료로 쓸 수 있다. 불로우피쉬 등의 일곱 가지 암호법으로 암호화할 수 있으며 암호 기법에 관심이 있는 사람들은 소스도 같이 구할 수 있다. *http://www.scramdisk.clara.net* `<http://www.scramdisk.clara.net>`

- 21 **KLDP**: 아이피 마스커레이딩 하우투 문서를 참조할 것.

- 22**KLDP**: 마스커레이딩 하우투를 참조할 것.

- 23 **디렉토리**: 레드 햇 6.0 디폴트 인스톨레이션의 경우에는 */usr/src/linux-2.2.5/Documentation/Configure.help* `</usr/src/linux-2.2.5/Documentation/Configure.help>`을 보면 된다. *http://kldp.org/HOWTO/html/Kernel/Kernel-HOWTO.html* `<http://kldp.org/HOWTO/html/Kernel/Kernel-HOWTO.html>`도 참조하기 바란다. *http://math-www.uni-paderborn.de/˜axel/config_help.html* `<http://math-www.uni-paderborn.de/˜axel/config_help.html>`에 2.2의 Configuration.help 최신판이 있다

- 24 **디렉토리**: 레드 햇 6.0에서는 /usr/src/linux-2.2.5/Documentation/Configure.help/Networking/filter.txt이다.

- 25 **SIGHUP**: SIGHUP을 좀 더 심도 있게 쓰고 싶으면 영문이지만 FSF에서 관리하는 배쉬 레퍼런스 매뉴얼 *http://www.fsf.org/manual/bash-2.02/bashref.html* `<http://www.fsf.org/manual/bash-2.02/bashref.html>`을 보라.

- 26 **GTK**: Gimp ToolKit. *http://www.gtk.org* `<http://www.gtk.org>`

- 27 **네서스**: 네서스 플러그인 중에 보면, 이러한 스텔드 포트 스캔을 해주는 스크립트가 이미 구성되어 있다. 방비를 철저히... TCP FIN을 쓰면 서비스를 탐지하는 것이 가능하다.

- 28 **슬렉웨어나 FreeBSD 사용자**: 슬렉웨어나 FreeBSD를 쓰는 사용자는 *http://www.telepath.com/support/unix/security.html* `<http://www.telepath.com/support/unix/security.html>` (영문)을 읽어보기 바란다.

29 **센드메일과 큐메일의 장단점**: 센드메일은 1999/2/4에 8.9.3 버전이 나와 있고, 작동이 불안정하다는 과거의 평과는 달리, 현재 버전은 비교적 안정적인 작동을 하고 있다. 1999/10/30 현재에는 8.10.0.beta6을 테스트하고 있었다.

- 센드메일의 특징은 그 규모가 크다는 것일 것이다. 스펨 차단과 지우기 등의 많은 기능을 지원해주는 등의 장점이 있지만, 규모가 크기 때문에 설치와 관리가 매우 복잡하다. 그렇기 때문에, 도사가 아니라면 소스로 설치하는 것은 거의 불가능하다고 생각한다. 설령 설치 후에도, 모든 옵션을 다 알아서 쓴다는 것도 불가능한 일이라고 생각하기도 한다. 큰 규모 탓에 옵션을 설정하는 "올바른 방법"이라는 것을 알기가 힘들며, 보안 개구멍이 자주 발견되는 것의 주된 이유는 이 큰 규모와 "자칫 잘못 구성하기 쉬운" 많은 옵션의 존재에 있다고 개인적으로 생각한다. (이런 저런 이유로, 센드메일 잘 쓰는 사람들은 책을 써서 돈도 많이 벌었다)
- 반면에 qmail은 보안이 중요한 경우에 권장을 할 수 있는 "보안상 안전"한 풀그림이지만, MTA의 중요한 기능 중의 하나인 스펨 컨트롤이 센드메일보다 힘들다. (스펨을 보내는 사람의 주소를 기준으로 메일을 차단하는 기능은 패치로 나와 있지만, 키워드로 스팸을 필터 하는 기능 은 센드메일을 따라잡지 못하고 있다) 반면에, qmail은 풀그림의 크기가 작고, 느린 컴퓨터 (486/16mb 정도)에서도 큰 규모와 많은 수의 메일을 잘 받아 준다.
  - 센드메일의 기초 사용 (영문): *http://ist.uwaterloo.ca/˜reggers/sendmail/* `<http://ist.uwaterloo.ca/~reggers/sendmail/>`
  - 센드메일의 제거 (영문): *http://www.qmail.org/man/misc/REMOVE.sendmail.txt* `<http://www.qmail.org/man/misc/REMOVE.sendmail.txt>`
  - 센드메일 도움말: *http://trade.chonbuk.ac.kr/˜leesl/mail/* `<http://trade.chonbuk.ac.kr/~leesl/mail/>`
  - KLDP 센드메일 도움말: *http://kldp.org/KoreanDoc/html/Sendmail-KLDP/Sendmail-KLDP.html* `<http://kldp.org/KoreanDoc/html/Sendmail-KLDP/Sendmail-KLDP.html>`
  - 큐메일 하우투: *http://kldp.org/Translations/Qmail-KLDP* `<http://kldp.org/Translations/Qmail-KLDP>`

- 30 **윗세 뵈이니마 홈 페이지**: *http://www.porcupine.org/wietse/* `<http://www.porcupine.org/wietse/>` 와 포스트픽스 및 MTA 리뷰 기사 (영문): *http://www.sunworld.com/swol-03-1999/swol-03-mailtools.html* `<http://www.sunworld.com/swol-03-1999/swol-03-mailtools.html>`

- 31 **IP Firewalling Chians**: "IP Firewalling Chians"를 어떤 말로 번역을 해주어야 할까 하다가 "방화벽 사슬"이란 용어를 이 만용 님의 IPchains-HOWTO 번역본(KLDP_URL)에서 따왔다.
  - 참조했던 이 만용 님의 번역본은 당연히 KLDP_URL에서 쉽게 구할 수 있었지만, 한가지 아쉬웠던 것은 이 문서가 98년 3월에 나온 v0.6의 번역인지라, 2.2.x 커널을 다루고 있지는 않다는 것이었다. 이 문서보다 좀 새로운 정보를 원한다면 1999년 3월에 V1.0.7 판으로 발표된 새로운 버전의 영문 문서를 *http://www.rustcorp.com/linux/ipchains* `<http://www.rustcorp.com/linux/ipchains>`에서 구할 수 있겠지만, 이 원문도 나온지가 오래되어서 (?) 아직 2.2.x를 다루고 있지는 않다.
  - 여러분 중에서 2.2.x에서의 방화벽 사슬 구성에 대한 특별한 성격을 아시는 분이 있으면 연락 주시기를 바란다.

- 32 **VPN**: 한글로 된 VPN 기술 정보는 그리 많지 않았다. 많이 부족하지만, 우선 한국 인터넷 정보 센터에서 만들어 놓은 색인을 보기 바란다. *http://www.nic.or.kr/data/report/Inds/inds-1.2.4.html* `<http://www.nic.or.kr/data/report/Inds/inds-1.2.4.html>`
  - 한글 알타비스타와 한글 야후 등을 써서 VPN에 대한 정보를 얻으려 했지만, 별 소득이 없었다. 전자 신문의 기사 검색을 쓰는 것이 훨씬 생산적이었다. *http://www.etnews.co.kr* `<http://www.etnews.co.kr>`

- 원문 문서 링크: *ftp://ftp.rubyriver.com/pub/jhardin/masquerade/ip_masq_vpn.html* `<ftp://ftp.rubyriver.com/pub/jhardin/masquerade/ip_masq_vpn.html>` 와 *http://www.wolfenet.com/˜jhardin/ip_masq_vpn.html* `<http://www.wolfenet.com/~jhardin/ip_masq_vpn.html>`

- 33 **테이프 백업의 스케줄 관리**: 원문은 일주일에 5일을 근무하는 미국의 경우에 맞게 설명이 되어 있다. 6일을 근무하는 회사가 많은 우리 나라의 경우에는 7 개의 백업 테이프를 쓰는 것이 알맞을 것이다. 또한 설명되어 있는 예는 (일주일 짜리의) 짧은 기간의 백업만을 고려하고 있다. 실제의 경우에는 장기간의 백업도 스케줄에 넣어 놓는 것이 좋다. 백업 스케줄의 테크닉에 관한 문서는 웹에서 쉽게 구할 수 있다.
  - 백업 스케줄의 관리 예제: *http://electron.lbl.gov/ychen/tapeback.html* `<http://electron.lbl.gov/ychen/tapeback.html>` 위의 경우에는 미국 버클리 대학의 로렌스 연구실의 한 분야에서 쓰는 컴퓨터 백업 스케줄을 보여준다. 시스템 관리자가 백업 스케줄을 공표하고 이에 철저히 따르는 것을 볼 수 있으며 백업도 1년, 반 년, 한 달, 각 주 간격으로 정기적으로 실행하고 있는 것을 볼 수 있다.
  - 백업에 앞서서 읽을 만한 문서: *http://kldp.org/KoreanDoc/html/Basic-KLDP/Basic-KLDP-2.html* `<http://kldp.org/KoreanDoc/html/Basic-KLDP/Basic-KLDP-2.html>`

- 34 **벅트랙**: 벅트랙은 *http://www.securityfocus.com/* `<http://www.securityfocus.com/>`로 자리를 옮겼다. 메인 페이지에서 Forum으로 들어가면 벅트랙이 보일 것이고, 그 아래에서 아카이브를 찾을 수 있다.

- 35 **새 버전**: *ftp://ftp.ox.ac.uk/pub/crypto/SSL* `<ftp://ftp.ox.ac.uk/pub/crypto/SSL>` 이미 업데이트가 나와 있었고, 사이트의 디렉토리 구조도 좀 바뀌져서 있었다.

- 36 **SSL과 헤더**: SSL 등의 보안 프로토콜을 쓰는 웹 사이트는 "https://..." 로 URL이 시작된다. 참조로, SSL은 통신 연결망의 보안에 사용되고 s-http는 메시지의 보안에 사용된다. "s-http" 와 "https://" 를 구분해서 보시기 바란다.

- 37 **휴면 계정**: 보안 수칙과 사용자 수칙에 오랜동안 사용되지 않는 계정에 대한 행동 조치 사항을 두어서 관리를 한다. 휴면 계정이 만들어지지 않도록 미리 조치를 취하는 것이 좋다. 예를 들면 일정한 시간이 지나면 패스워드가 자동으로 만기되도록 계정을 꾸미고, 필요 이상으로 오랜 시간 동안 사용이 안된 계정은 일단 잠정 폐쇄해 버린다. 사용자가 이런 계정을 다시 사용하려 한다면 반드시 관리자를 접촉하도록 해서 패스워드를 다시 받도록 할 수도 있을 것이다.

- 38 **단방향식 암호**: 단방향식 암호 (One-way password)는 일방 통행로처럼 생각을 하면 된다. (패스워드의 원래 값을 패스워드 기능으로 처리를 한) 처리 값을 가지고 원래의 패스워드를 역산해 내거나 심지어 패스워드의 처리 기능 까지도 역산해 내는 것이 가능하다면 안될 것이다. 유닉스용의 DES는 단방향식이라 하였는데, "단방향" 이라는 뜻은 한 번 처리가 된 처리값을 다시 패스워드 기능으로 처리를 했을 경우 원래의 기본값이 나오지 않는 경우를 말한다.

- 39 **넷스케이프사의 암호 관련 기법 정보**: 99년 10월에 번역을 하면서 원문에 나와 있는 URL에 가보니 URL이 죽어 있었다. 넷스케이프사가 얼마 전 웹 사이 트를 단장하면서 주소가 바뀐 것 정도로 생각을 했는데, 알고 보니 그 뿐 아니라 대부분의 암호에 관한 정보를 아예 웹 사이트에서 전부 내려 버린 것이었다. 넷스케이프의 암호 정보와 기술법을 구하려면 다른 제 3의 장소에서 구하거나, 넷스케이프의 사업 동반자가 되는 등으로 소스를 사서 쓰는 수밖에는 없을 것 같다. 참고로 – 비록 넷스케이프 브라우저의 소스가 리눅스용으로 공개되어 있기는 하지만 – 암호 기술법에 관한 문서는 관련법과 지적 소유권 등의 이유 때문에 공개를 못하게 되어 있으니_ 혹시 여러분 중에 암호 코드와 기술법에 대한 자세한 자료를 가진 사람이 있더라도 함부로 웹에 올리지 않도록 주의하시기를 바란다. (관계 기관에 여러분의 컴퓨터를 압수 당하거나, 더 무섭게는 관계 회사의 변호사에게서 오는 무서운 편지를 받는 일이 없도록 하자 ^_^;).

- 책1 **한글 번역판**: "인터넷 방화벽 구축하기, D. Brent Chapman 와 Elizabeth D. Zwicky 작, 한빛미디어 출판, ISBN 89-79140-22-3". 알라딘에서 2만 3천원정도에 판매한다는 정보를 볼 수 있었다. 원서 정보는 오렐리 웹 사이트에서 구할 수 있었다. *http://www.oreilly.com/catalog/fire/index.html* `<http://www.oreilly.com/catalog/fire/index.html>`.
- **기타**: 번역을 하다 보니 다른 문서를 참조할 때가 있었고, 이 문서 중에는 "번역이 되어 있었다면 좋았을 것"이라는 생각이 들던 쓸 만한 문서들이 있었습니다. 여러분 중에서 KLDP의 번역 작업에 관심이 있는 분이 있으시면 한 번 이 들 문서의 번역을 고려해 볼 만하다고 생각합니다. 일부 문서는 하우투 문서가 아니므로, 번역에 앞서서 저자의 허가를 꼭 받으시도록.
  - 좋은 패스워드를 고르는 방법: *http://consult.cern.ch/writeup/security/security_3.html* `<http://consult.cern.ch/writeup/security/security_3.html>`
  - VPN: *http://www.wolfenet.com/~jhardin/ip_masq_vpn.html* `<http://www.wolfenet.com/~jhardin/ip_masq_vpn.html>`
  - IP-masq-vpn HOWTO
  - VPN: *ftp://ftp.rubyriver.com/pub/jhardin/masquerade/ip_masq_vpn.html* `<ftp://ftp.rubyriver.com/pub/jhardin/masquerade/ip_masq_vpn.html>`

16.6 감사의 글 (Acknowledgements)

- **박 민석님 (dolman@correl1.snu.ac.kr)**: 1.1.0과 1.0.2는 전혀 다르지 않습니다. 그리고 1.0.2에 추가된 두 항목과 "번역자의 추가 색인"을 제외한다면, 1.0.2는 0.9.11과 다른 것이 거의 없었습니다. 0.9.11 번역에 크게 도움이 되었던 이전 문서를 번역하신 박 민석님의 손길이 1.1.0에도 큰 변화 없이 남아 있습니다. 박 민석님에게 여전 한 감사의 마음을 전하고 싶습니다. 꾸벅
- **유 성태님(sreki@bomun.kaist.ac.kr)**: 예전에 번역을 올리면서 유 성태님께서 SGML을 편집해 주셨습니다. 제가 SGML에는 무지인지라, 참 난감했는데 말입니다. 감사의 마음을 전하고 싶습니다. 꾸벅
- **권 순선님**: 여러 말 할 것 없이 국내 리눅서들에게는 소중한 분입니다. 님의 kldp.org가 없었으면 우리들은 모두 영어로 만 만들어진 하우투를 읽었어야 하는 형편입니다. 우리말로 번역된 문서들을 그냥 넙죽 읽어만 보던 여러 독자님들은 빨리 kldp.org에 가서 슬쩍 감사의 말이라도 한 줄 남겨 보세 요. 자유 게시판에 말입니다. 꾸우우벅
- **황 현욱님**: 1.1.0판을 올리면서 1.0.8판을 둘러보니 틀린 부분이 상당히 많더군요. URL에 오타도 내고, 부분 적으로 틀린 것도 많고, 문법도 엉망이고 말입니다. 그래도 수정하라고 메일 주신 분은 이 분이 유일하게 한 분... (아 아무도 보안 하우투는 안 읽어 주나부다... 흐흑!) 황 현욱님 감사합니다.
- **여러분**: 보잘 것 없는 번역이나마 읽어주셔서 감사합니다. 그리고, 제딴에는 고친다고 고쳤는데 아무래도 어디엔가에는 또 실수가 있을 것 같습니다. 연락주시면 감사하겠습니다.