Solaris 9 운영 환경에서의 인증 기능(1)

PAM을 이용한 Solaris 9의 확장된 인증(1)

이번 호부터 Solaris 9 운영 환경에서 PAM을 구현한 인증에 대해 알아보면서, 보안 정책을 요구하는 사이트에 PAM을 어떻게 구성하는지에 대한 연재를 시작한다. 이번 호에는 기존의 Solaris 운영 체계에서의 인증 방법과 PAM 구성 요소에 대해 살펴본다.

정리 · 김봉환 | 한국 썬 시스템엔지니어링 본부 과장

PAM(Pluggable Authentication Modules : 장착형 인증 모듈)은 Solaris 9 운영 환경에서 인증 메커니즘의 필수적인 부분이다. 이것은 시스템 관리자에게 엔드유저에 대한 인증을 시스템 상에서 수행할 수 있는 모든 인증 서비스를 선택할 수 있도록 하는 능력과 유연성을 제공한다. 다른 형태의 PAM으로는 Linux-PAM과 OpenPAM이 있다. PAM을 사용하면 애플리케이션은 주어진 클라이언트를 위한 시스템 관리자에 의해 정의되는 어떤 인증 방법에 상관없이 인증을 수행할 수 있다.

PAM은 시스템 관리자가 네트웍을 통해 제공되는 각 서비스에 적절한 인증 메커니즘을 전개할 수 있게 해준다. 시스템 관리자는 애플리케이션이나 유틸리티를 수정하지 않고도 하나 이상의 인증 기술을 선택할 수 있다. 뿐만 아니라 애플리케이션 개발자들이 혁신적으로 향상된 인증 기술에 신경쓰지 않아도 되는 환경을 제공해주는 동시에, 전개된 애플리케이션이 이러한 향상된 기술을 사용할 수 있도록 해준다.

PAM은 시스템 엔트리 서비스의 인증을 제공하는 런타임 장착형(pluggable) 모듈을 채택하고 있으며, 이것이 제공하는 혜택은 다음과 같다.

- **각 애플리케이션이나 서비스가 자체 인증 정책을 사용할 수 있도록 해주는 유연한 구성 정책** : PAM은 시스템 관리자가 기본 인증 메커니즘을 선택할 수 있는 능력을 제공한다. 또한 복수의 패스워드가 필요한 PAM 메커니즘을 이용함으로써 시스템의 보안성을 향상시킬 수 있다. 예를 들어 시스템 관리자는 Kerberos나 Digest-MD5 사용자를 모두 인증할 수도 있다.

- **엔드유저를 위한 용이한 사용성** : PAM을 사용하면 패스워드 이용이 더욱 쉬워진다. 즉 사용자가 서로 다른 메커니즘에 대해 동일한 패스워드를 사용할 경우 사용자들은 패스워드를 재입력할 필요가 없는 것이다. 제대로 구현된 PAM은 사용자가 복수의 명령어를 입력하지 않아도 여러 개의 인증 수단을 위한 패스워드를 띄워준다. 텔넷으로 접속하는데 인증(Certificate) 기반의 패스워드 인증을 요구하면서도 Unix 패스워드만으로 콘솔 로긴 세션을 허용하는 사이트가 좋은 예다.
- Solaris 9 운영 환경은 다양한 방법으로 향상된 보안성과 용이한 사용 방법을 제공한다. PAM을 통해 접근할 수 있는 보안 메커니즘은 동적으로 로딩할 수 있도록 구현되었으며, 시스템 관리자가 애플리케이션에 투명한 방식으로 설치할 수 있는 소프트웨어 모듈을 공유하도록 되어 있다. 전체적인 보안성이 개선되어 사용자는 향상된 서비스 수준과 소유 비용의 절감을 만끽할 수 있다.

기존 Solaris 운영 환경에서의 인증 방법

기존 Solaris 운영 환경에서의 인증 방법은 초기 Unix 구현 방식으로 개발된 수단을 기반으로 하고 있다. 이 방식은 crypt(3c)라고 하는 단일 암호화 해싱 알고리즘을 채택하고 있다. 암호화된 패스워드는 파일이나 Solaris 운영 환경의 네이밍 서비스 내에 저장되며, 사용자의 로긴 과정 중에 이곳에서 추출된다. crypt(3c)를 이용하는 Solaris 운영 환경의 인증 방법인 기존의 Unix 방식은 매우 유명하며, 데이터 저장 공간으로 LDAP 디렉토리를 사용할 수 있도록 발전되었다.

인증 방식에 대해 자세히 살펴보기 전에, crypt(3c)가 무엇인지에 대해 먼저 알아볼 필요가 있다. 독자들은 crypt라는 이름의 애플리케이션과 혼동할 수도 있는데, 이것은 Solaris 운영 환경에 탑재된 표준 툴이며, 파일의 내용을 암호화하고 복호화하는 프로그램이다. 이 프로그램은 /usr/bin/cript에 있다.

그러나 crypt라는 명칭 역시 인증과 관련되어 있으므로, 여기서는 crypt(3c)라고 하며, 표준 Unix 패스워드 해싱 알고리즘을 crypt(3c)로 부르는 것이다. 참고로 C 프로그래머들은 이것을 libc.so 라이브러리에서 찾아 사용할 수 있다.

공개키 기술을 기반으로 하는 보다 복잡한 인증 방식은 네트웍 인포메이션 시스템(NIS+)의 네이밍 서비스와 함께 소개되었다. NIS+ 네이밍 서비스 방식은 crypt(3c)를 대체하지는 않았지만 네트웍 패스워드라는 개념을 소개해 부가적인 보안 계층을 제공하고 있다. 사용자가 원격 프로시저 호출(RPC) 메커니즘을 통해 네트웍 서비스에 접근할 때에는 네트웍 패스워드가 필요하다.

처음에 썬에서 개발된 후, 공통 데스크탑 환경(CDE) 및 Motif에 포함시키기 위해 개방형 소프트웨어 재단(OSF)에서 채택한 장착형

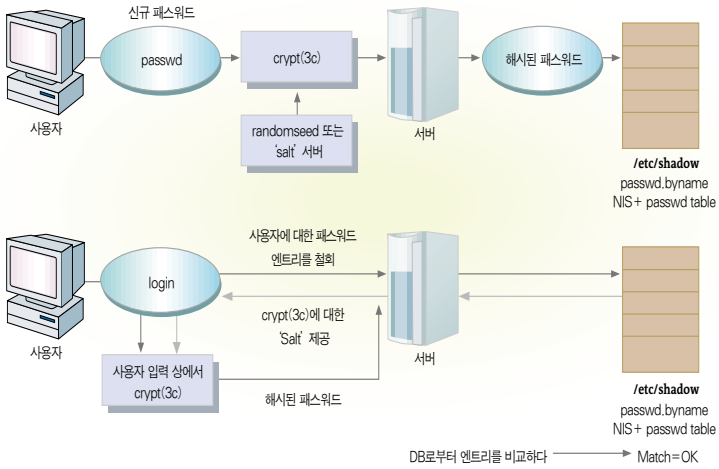

그림 1. Unix 패스워드의 동작 원리

인증 모듈(PAM)은 동적 시스템 인증을 위한 메커니즘을 제공하며, 패스워드와 계정, 세션 관리와 같은 관련 서비스도 제공한다. 현재도 개발이 진행중인 새로운 인증 모델을 발견한 썬은 PAM 구조를 만들어냄으로써 기존의 제품들을 그대로 사용하면서도 부가적인 방식을 덧붙일 수 있도록 해준 것이다. PAM은 Solaris 2.6 운영 환경에서 소개되면서 새로운 인증 메커니즘이 개발되고 소개될 때, login이나 passwd, dtlogin, telnet, rlogin과 같은 시스템 엔트리 서비스를 재코딩할 필요가 없도록 해주었다.

PAM 구조와 기존 Solairs 운영 환경 인증의 대체 방안은 'Solaris 9 운영 환경 PAM 프레임웍'에 소개한다.

Unix 패스워드

패스워드는 Solaris의 passwd 명령어로 생성된다. 이 명령어는 사용자가 (새로운)패스워드를 텍스트 스트링으로 입력하도록 한다. Solaris에서 이 텍스트 스트링은 crypt(3c) 방식을 이용해 해시되거나 단일 방식으로 암호화된다. 이 작업 결과는 /etc/shadow나 passwd.byname과 passwd.byuid NIS 맵에 저장된다. 만일 NIS+ 네이밍 서비스가 사용되면 이 결과는 Passwd 파일에 Cred 테이블 형식으로 저장된다. crypt(3c) 알고리즘은 salt string이라고 기술적으로 알려진 randomseed로 공급되며, 그 결과는 passwd 명령어가 작동될 때마다 달라지므로 동일한 텍스트 스트링이 입력된다 해도 결과는 틀리게 된다.

사용자가 로긴하면 Solaris의 로긴 프로그램은 사용자가 패스워드를 입력하도록 한다. 이 패스워드는 passwd 명령어와 동일한 방식으

로 해시된다. 이 프로세스의 결과치가 패스워드 DB의 결과치와 일치하면 사용자가 인증되는 것이다.

crypt(3c)의 장점과 단점

crypt(3c)의 대표적인 장점 중 하나는 긴밀한 환경에서 구현이 쉽다는 것이다. 즉 인증 과정은 사용자가 로긴하는 호스트에서 이뤄지므로 별도의 인증 서버가 필요하지 않다. 지역 로긴의 경우 명확한 텍스트 형태의 패스워드가 저장되거나 네트웍을 통해서 전송되는 일은 없으므로 누군가 패스워드를 가로챌 우려는 하지 않아도 된다. 그러나 텔넷이나 rlogin을 이용해 네트웍 상에서 인증할 때에는 패스워드가 분명한 텍스트 형태로 전송되게 된다.

crypt(3c)는 단일 인증 알고리즘을 사용하므로, 서버 내에 저장된 패스워드를 복호화하기가 어렵다. 실제 패스워드는 사용자만 알고 있을 뿐이다. 즉 암호화되어 저장된 패스워드는 다른 인증 방식을 필요로 하는 또 다른 형태로 변환할 수 없다는 의미이다.

crypt(3c) 기능이 호출되면 이것은 처음의 8글자를 불러와서 그 계산값을 돌려준다. 이 계산치는 다시 salt라고 하는 불규칙적으로 발생된 값에 넣어진다. 편리한 암호화 방식에서 salt는 처음 2글자가 저장되어 있다. 이 salt값이 더해져 13글자의 순서가 조합된다. 이 결과치는 특정 네이밍 서비스에 저장된 패스워드 스트링의 중요한 부분이 된다.

CPU와 스토리지 능력이 향상되면서 crypt(3c) 알고리즘 역시 점차 공격에 취약해졌다. PAM 인증 방식을 따르는 crypt(3c) 메커니즘은 Solaris 9에 탑재되어 공급된다. 이것은 수년간 Solaris 운영 환경에서 구현해온 방식과 동일하며, 그로 인해 Solaris 9의 발표와 함께 획기적인 인증 방식의 변화를 시도한 이유이기도 하다.

Solaris의 crypt(3c) 메커니즘은 Solaris 운영 환경 로컬 사용자들을 인증할 때에도 물론 잘 동작하지만 Solaris 운영 환경 내에서만, 동작하는 애플리케이션이나 서비스만 사용할 수 있는 방식은 아니다. 이렇게 제한적으로만 사용할 수 있다면 여러 개의 패스워드 시스템으로 작업해야 하는 시스템 개발자들이나 시스템 관리자들, 여러 개의 패스워드를 기억해야 하는 사용자들은 큰 어려움에 처하게 될 것이다.

Solaris 9 내의 PAM 인터페이스는 시스템 개발자들이 로긴이나 텔넷 등과 같은 관리 명령어를 수정하지 않고도 다른 인증 기술을 쉽게 전개할 수 있도록 해준다. 관리자들은 애플리케이션이나 유틸리티를 수정하지 않고도 하나 이상의 인증 기술을 선택할 수 있다. PAM은 싱글 사인온 시스템의 전체가 될 수도 있다. PAM API는 전반적인 시스템 보안성을 향상시키는 유연한 메커니즘을 제공한다. PAM API에 대한 자세한 사항은 다음에 다루겠다.

PAM 구성 요소

PAM을 구성하는 요소들은 다음과 같다.

- Solaris 9의 PAM 프레임웍
- PAM 모듈 형태
- PAM 서비스 모듈 수정
- PAM 구성 파일 수정
- PAM 패스워드 관리 확장자
- pam_ldap 패스워드 관리(Solaris 9 12/02일자 운영 체계에서 사용 가능)

Solaris 9 PAM 프레임웍

PAM 프레임웍은 login이나 dtlogin, rsh, su, ftp, telnetd와 같은 명령어를 수정하지 않고도 장착할 수 있는 새로운 인증 기술을 가능하게 해준다. PAM은 Unix 로그인을 Kerberos나 LDAP 인증과 같은 다른 보안 메커니즘으로 대체할 때에도 사용할 수 있다. 계정과 세션, 패스워드 관리를 위한 메커니즘은 이 프레임웍을 통해 장착할 수 있다.

이 프레임웍은 다음 네 가지 특정 구성 요소로 이뤄져 있다.

- PAM API는 애플리케이션 프로그램에게 공급된다.
- PAM 프레임웍은 API를 구현하는 임무를 담당한다.
- PAM 서비스 공급자 인터페이스(SPI)는 PAM API를 위한 백엔드 기능을 구현한다.
- 구성 파일인 pam.conf는 어떤 서비스 공급자가 여러 가지 프로그램에 사용될지를 지정하게 된다.

PAM은 시스템 관리자가 인증을 제공하는 서비스를 임의대로 조합할 수 있도록 한다. 여기에는 개별 애플리케이션 인증 정책과 지정되지 않은 애플리케이션에 대한 기본 인증 메커니즘 선택, 높은 보안성을 요구하는 시스템의 복수 패스워드 등을 가능하게 하는 유연한 구성 정책 등이 총망라된다. 또 다른 가치 있는 서비스로서 엔드유저가 쉽게 사용할 수 있는 점을 들 수 있는데, 이것은 패스워드가 동일할 경우 사용자 패스워드를 재입력할 필요가 없다는 것과 옵션 패러미터가 서비스를 통과한다는 점 등이다.

Solaris 9의 새로운 PAM 프레임웍이 발표됨으로써 PAM LDAP 서비스 모듈도 계정 서비스를 지원할 수 있도록 확장되었는데, 이것은 디렉토리(LDAP) 서버와 결합해 사용자의 패스워드와 계정 상태를 체크할 수 있는 기능이다. 디렉토리 서버는 패스워드 상태를 pam_ldap으로 되돌려주며, 여기서 패스워드 상태가 PAM 에러 코드로 변환된다. 사용 기간이 만료된 패스워드로 로그인하는 사용자는 거부될 것이며, 패스워드가 만료 시점에 이른 사용자는 경고 메시지를 보게 될 것이다.

pam_ldap 모듈은 패스워드 구문 체크를 지원할 수 있도록 개선되었는데, 이것은 Sun ONE Directory Server(구 iPlanet Directory Server)의 패스워드 정책 엔진을 통해 수행된다. 패스워드 변경시(passwd 명령어를 이용해) 사용자는 '패스워드가 너무 짧다' 혹은 '패스워드를 이미 사용했다' 등의 에러 메시지를 발견하게 될 것이다.

장착형 인증 서비스 모듈 형태

PAM 프레임웍은 현재 네 가지 형태의 서로 다른 서비스 모듈을 제공하는데, 이것은 인증 관련 서비스를 제공하는 동적 적재형(loadable) 모듈 형태로 구현되고 있다. 이 모듈들은 각 모듈이 수행하는 기능에 따라 다음과 같이 구분된다.

- **인증(auth)** : 사용자를 인증해주며 인증서의 설정과 리프레시, 삭제를 가능하게 한다.
- **계정 관리(account)** : 패스워드 에이징(aging), 계정 만료, 접근 시간 제한 등을 점검한다. 인증 모듈이 일단 사용자를 식별해주면 계정 관리 모듈은 사용자의 접근 가능 여부를 결정하게 된다.
- **세션 관리(sesion)** : 세션을 열고 닫는 기능을 관리한다. 이 모듈은 모든 활동 내용을 기록하며 세션이 종료된 후에 사라진다. 예를 들어 unix_session 모듈은 lastlog 파일을 업데이트해준다.
- **패스워드 관리(password)** : 사용자가 인증 토큰(보통 패스워드)을 변경할 수 있도록 해주는 기능을 제공한다.

스태킹(Stacking : 여러 단계를 거쳐 인증하는 것)

PAM은 스태킹(Stacking)을 통해 여러 가지 방법으로 인증할 수 있게 한다. 사용자가 PAM을 통해 인증되면, 사용자를 완전히 식별하기 위해 여러 가지 방법이 선택된다. 구성 방식에 따라 사용자는 각 인증 수단에 대한 패스워드를 알려주게 된다. 즉 사용자는 완벽한 인증을 위해 다른 명령어를 실행할 필요가 없다는 뜻이다. 사용되는 인증 수단의 순서는 /etc/pam.conf 구성 파일을 통해 결정된다.

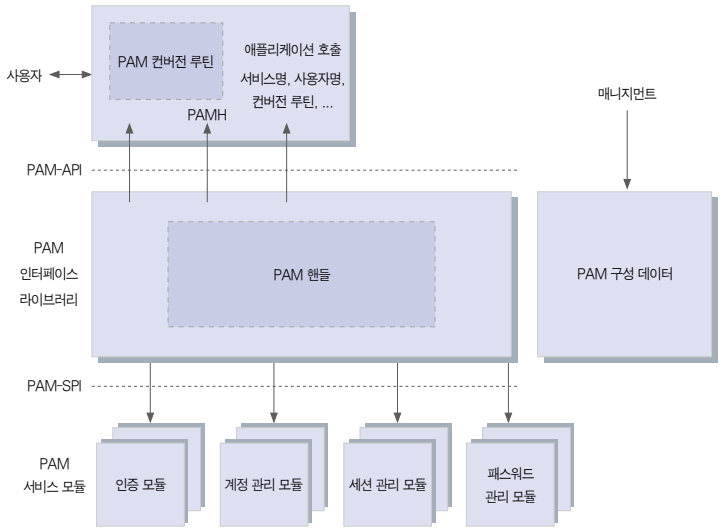

그림 2. PAM 프레임웍 아키텍처

단 스태킹은 보안에 대한 위험성을 증가시키는데, 이는 각 메커니즘의 보안성이 스택에서 사용되는 최소한의 보안 패스워드로 제한되기 때문이다. 예를 들어 Solaris의 LDAP 클라이언트 구현의 pam_kerb5(PAM의 Kerberos V5 서비스 모듈)과 같은 최강의 PAM 메커니즘을 디렉토리 서버와 사용할 수는 없는데, 그 이유는 현재 가용한 디렉토리 서버가 Kerberos를 지원하지 않기 때문이다.

이제 PAM의 기초에 대한 전반적인 사항은 모두 살펴보았다. 다음에는 PAM 프레임웍의 구조 전반을 알아보겠다. 그림 2는 PAM 프레임웍을 나타낸 것이다.

· PAM 작동

PAM 소프트웨어는 라이브러리와 몇 가지 모듈, 구성 파일로 이뤄져 있다. PAM 라이브러리인 /usr/lib/libpam.so는 적절한 모듈을 적재하고 스태킹을 관리하는 프레임웍을 제공한다. 이것은 장착될 모든 모듈의 포괄적인 구조를 제공한다.

그림 3은 애플리케이션과 라이브러리, 모듈 간의 관계를 나타낸다.

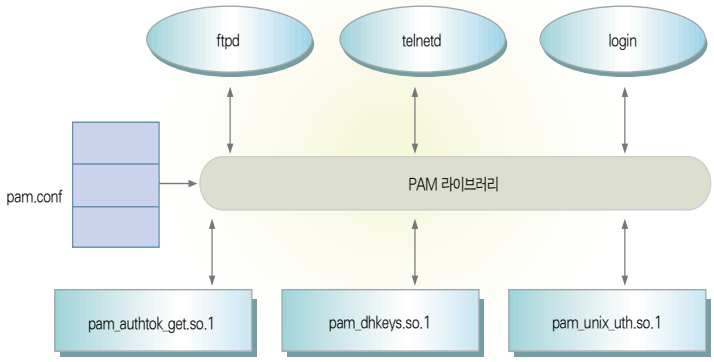

그림 3. PAM 애플리케이션 및 라이브러리, 모듈 간의 관계

애플리케이션의 login과 passwd su는 적절한 모듈에 접근할 때 PAM 라이브러리를 사용한다. pam.conf 파일은 각 애플리케이션과 어떠한 모듈을 사용할지를 규정짓는다. 모듈로부터 나온 응답은 라이브러리를 거쳐서 애플리케이션으로 전송된다.

장착형 인증 서비스 모듈

각 모듈은 특정 메커니즘의 구현을 제공한다. 하나 이상의 모듈 형태(auth나 account, session, password)는 각 모듈과 연결될 수 있으나 각 모듈은 하나 이상의 모듈 형태를 관리해야 한다. 다음은 Solaris 9의 한 부분인 모듈들을 설명한 것이다.

보안을 위해 이러한 파일들은 루트가 소유해야 하며, group이나 other 권한을 이용해 사용하지 못하도록 설정해야 한다. 만일 이 파일들의 소유자가 루트가 아니면 PAM은 모듈을 로드하지 않을 것이다. 모듈에 대한 권한과 소유권에 대한 요구 사항은 어떤 문서에도 나와 있지 않지만 차기 버전에서는 제공될 것이다.

- **pam_authok_get** : 인증과 패스워드 관리를 지원한다. 이 모듈이 사용자로부터 패스워드를 획득하는 일을 관리하므로, 스택상의 다른 모듈은 자신들의 임무에만 전념할 수 있고 사용자로부터 정보를 획득하는 사항에는 관여하지 않게 된다.
- **pam_authok_check** : 이 모듈은 패스워드 관리 스택 기능을 제공한다. 특히 이것은 새로 입력된 패스워드의 작성을 점검한다. 이것이 수행하는 점검 사항은 pam_authok_check의 man 페이지에 자세히 기록되어 있다.
- **pam_authok_store** : PAM 패스워드 관리 스택 기능을 제공한다. pam_authok_store를 구성하는 플래그가 발생하면, 이 모듈은 pam_user가 지정한 사용자를 위한 인증 토큰을 업데이트한다.
- **pam_dhkeys** : 인증과 패스워드 관리를 지원한다. 이 모듈은 특히 Secure RPC 호출(NIS+와 Secure NFS)에 이용되는 디피-헬만(Diffe-Hellman) 키의 생성과 수정에 관련되어 있다.
- **pam_passwd_auth** : passwd가 구현하는 것과 같은 패스워드 서비스 인증 기능을 제공한다. 이것은 표준 PAM 인증 모듈과는 차이가 있다.
- **pam_unix_account** : Unix의 PAM 계정 관리 모듈처럼 PAM 계정 관리 스택 기능을 제공한다. pam_acct_mgmt(3PAM) 기능은 nsswitch.conf(4)에 지정되어 있는 저장소로부터 패스워드 에이징 정보를 추출해 사용자의 계정과 패스워드의 만료 여부를 검사한다.
- **pam_uinix_auth** : 일반적인 Unix crypt(3c) 스타일의 패스워드 암호화 방식을 이용해 nsswitch.conf에서 지정한 패스워드 저장소에서 사용자가 입력한 패스워드를 점검한다. 단 인증에서만 이용할 수 있다.
- **pam_unix_session** : Unix의 세션 관리 PAM 모듈과 같이 세션을 초기화하고 종료하는 기능을 제공한다.

보안을 위해 이러한 파일들은 루트가 소유해야 하며, group이나 other 권한을 이용해 사용하지 못하도록 설정해야 한다. 만일 이 파일들의 소유자가 루트가 아니면 PAM은 모듈을 로드하지 않을 것이다. 모듈에 대한 권한과 소유권에 대한 요구 사항은 어떤 문서에도 나와 있지 않지만 차기 버전에서는 제공될 것이다.

참고로 그림 3에서 pam_unix는 LDAP 서버 상에서 전체적으로 계층화된 것이 아니다. pam_unix 모듈은 네임 서버 스위치(NSS)와 NSS 백엔드 계층 상에 있으며 NIS와 NIS+, LDAP과 같은 파일 형태로 되어 있다.

PAM 구성 파일 업데이트

PAM 구성 파일인 /etc/pam.conf는 어떠한 인증 서비스가 어떠한 순서로 사용될지를 결정한다. 각 시스템 엔트리 애플리케이션에서 원하는 인증 메커니즘을 선택하려면 이 파일을 수정해야 한다.

· 구성 파일 문법

PAM 구성 파일은 다음과 같은 문법 구조로 이뤄져 있으며, 표 1은 문법의 기능을 설명하고 있다.

service_name module_type control_flag module_path module_option

표 1. 구성 파일 문법

문법	기능
service_name	서비스의 명칭(예 ftp, login, telnet)
module_type	서비스를 위한 모듈 형태(auth, account, session, password)
control_flag	모듈의 지속 혹은 중지를 결정
module_option	서비스 모듈로 주어질 옵션을 지정

파운드 기호(#)를 각 줄의 첫 번째에 붙이면 주석을 달 수 있다.

단 PAM 구성 파일을 입력할 때 다음과 같은 사항이 하나라도 존재하면 구성 파일은 무시된다는 것에 주의하자.

- 라인이 4개의 필드보다 적다.
- 유효하지 않은 값이 module_type이나 control_flag에 주어졌다.
- 지정된 모듈을 찾을 수 없다.

표 2는 PAM 구성을 요약한 것이다.

표 2. PAM 구성

서비스	명칭 대몬 혹은	명령어 모듈 형태
cron	/usr/sbin/cron	account
dtlogin	/usr/dt/bin/dtlogin	auth, account, session
ftp	/usr/sbin/in.ftpd	auth, account, session
init	/usr/sbin/init	session
login	/usr/bin/login	auth, account, session, password
passwd	/usr/bin/passwd	auth, account, password
ppp	/usr/bin/pppd	auth, account, session
rexecd	/usr/sbin/in.rexecd	auth, account
rexd	/usr/sbin/rpc.rexd	account, session
rlogin	/usr/sbin/in.rlogind	auth, account, session, password
rsh	/usr/sbin/in.rshd	auth, account
sac	/usr/lib/saf/sac	session
sshd	/usr/lib/ssh/sshd	auth, account, session, password
su	/usr/bin/su	auth, account
telnet	/usr/sbin/in.telnetd	auth, account, session, password
ttymon	/usr/lib/saf/ttymon	session
uucp	/usr/sbin/in.uucpd	auth, account

· 컨트롤 플래그

인증 과정중 모듈로부터 지속 혹은 실패의 성격을 결정하려면, 각 엔트리에 대한 4개의 컨트롤 플래그 중 하나를 선택해야 한다. 성공적인 시도인지 실패한 시도인지는 컨트롤 플래그를 통해 지시된다. 이 플래그는 모든 형태의 모듈에 적용되지만 다음 설명은 이 플래그들이 인증 모듈에 사용된다고 가정해 기술한 것이다. 컨트롤 플래그는 다음과 같다.

- **required** : 이 모듈은 전체의 성공적인 결과를 가져오기 위해 반드시 success값을 되돌려줘야 한다. 만일 모든 모듈에 required라는 레이블이 붙는다면, 모든 모듈을 통한 인증은 사용자를 인증하기 위해 성공돼야 한다. 만일 어떤 모듈이 실패하면 첫 번째 실패한 모듈로부터 에러값이 보고된다. 만일 required라는 플래그가 붙은 모듈에서 실패가 나오면 스택 내의 모든 모듈이 동작을 시도하더라도 실패값이 나온다. 만일 required라는 플래그가 붙은 모듈이 없으면 서비스에 대한 엔트리가 적어도 하나는 사용자를 인증하기 위해 성공해야 한다.
- **requisite** : 이 모듈은 부가적인 인증을 위해 반드시 success값을 돌려줘야 한다. 만일 requisite라는 플래그가 붙은 모듈이 실패하면 즉각적으로 애플리케이션에 에러가 보고되며, 더이상의 인증을 시도하지 않게 된다. 만일 스택이 required라는 레이블의 실패한 우선 모듈을 포함하지 않으면 이 모듈에서 에러가 발생한다. 만일 required라는 레이블의 초기 모듈이 실패하면 필요한 모듈로부터 에러 메시지가 나타난다.
- **optional** : 만일 이 모듈이 실패하고 스택 내의 다른 모듈이 success값을 보낸다면 전체적인 결과는 성공적인 것이 된다. optional 플래그는 스택 내에 하나의 success만으로도 사용자를 인증할 수 있는 경우에 사용된다. 이 플래그는 특별한 메커니즘을 성공시키는 것이 그다지 중요하지 않을 경우에만 사용돼야 한다. 사용자가 작업하기 위해 특정 메커니즘과 연관된 권한을 가져야 할 경우에는 optional이라는 레이블을 붙이면 안된다.
- **sufficient** : 이 모듈이 성공하면 스택 내의 다른 모듈들은 required라는 레이블이 붙었다 해도 그냥 스킵하게 된다. sufficient 플래그는 사용자가 접근하는 데에 하나의 성공적인 인증만으로도 충분하다는 것을 나타낸다.

참고로 Solaris 9의 12월 2일자 버전에는 새로운 컨트롤 플래그가 PAM 프레임웍에 추가되었다. 컨트롤 플래그 binding은 성공시에는 프로세싱을 종료하고 실패할 경우에는 실패를 보고한다는 의미다.

마치며

이번 호에는 Solaris 9에서 필수적인 인증 메커니즘인 PAM의 구성 요소와 프레임웍 아키텍처 등을 살펴보았다. 다음 호에는 PAM 모듈을 추가시키는 방법과 PAM LDAP 모듈 등에 대해 알아본다.

참고로 이 글은 Sun ONE Directory Server 그룹의 Michale Haines 씨가 쓴 'Extending Authentication in the Solaris 9 Operating Environment Using Pluggable Authentication Modules(PAM)'을 정리한 것이다.