CA Access Control

UNIX 용 끝점 관리 안내서

12.6.01

타사 고지 사항

CONTAINS IBM(R) 32-bit Runtime Environment for AIX(TM), Java(TM) 2 Technology Edition, Version 1.4 Modules

샘플 스크립트와 샘플 SDK 코드

CA Access Control 제품에 포함된 샘플 스크립트와 샘플 SDK 코드는 정보 제공 목적으로만 "있는 그대로" 제공됩니다. 이 항목은 특정 환경에 맞게 수정이 필요할 수 있으며, 프로덕션 환경에 사용하려면 프로덕션 시스템에 배포하기 전에 반드시 테스트 및 검사를 수행해야 합니다.

CA Technologies 는 이러한 샘플에 대한 지원을 제공하지 않으며 이 스크립트로 인한 어떠한 오류에도 책임을 지지 않습니다.

CA Technologies 제품 참조

이 문서는 다음 CA Technologies 제품을 참조합니다 :

- CA Access Control Enterprise Edition
- CA Access Control
- CA Single Sign-On(CA SSO)
- CA Top Secret®
- CA ACF2™
- CA Audit
- CA NSM(CA Network and Systems Management, 이전의 Unicenter NSM 및 Unicenter TNG)
- CA Software Delivery(이전의 Unicenter Software Delivery)
- CA Service Desk(이전 이름: Unicenter Service Desk)
- 사용자 활동 보고 (이전 명칭: CA Enterprise Log Manager)
- CA Identity Manager

설명서 규칙

CA Access Control 설명서는 다음과 같은 규칙을 따릅니다.

형식	의미
고정 폭 글꼴	코드 또는 프로그램 출력
기울임꼴	강조 또는 새 용어
굵게	표시된 대로 동일하게 입력해야 하는 텍스트
슬래시(/)	UNIX 및 Windows 경로를 기술하는 데 사용되는 플랫폼 독립적인 디렉터리 구분 기호

이 설명서는 또한 명령 구문과 사용자 입력(고정 폭 글꼴로 표시됨)을 설명할 때 다음과 같은 특별한 규칙을 사용합니다.

형식	의미
기울임꼴	반드시 입력해야 하는 정보
대괄호([ ]) 사이	선택적 피연산자
중괄호({ }) 사이	필수 피연산자 집합
파이프(|)로 구분된 선택 사항	대체 피연산자(하나 선택)를 구분합니다. 예를 들어, 다음은 사용자 이름 *또는* 그룹 이름 중 *하나*라는 의미입니다. {*username*|*groupname*}
...	앞의 항목 또는 항목 그룹이 반복될 수 있음을 나타냅니다.
밑줄	기본값
줄 마지막에 공백 다음의 백슬래시(\)	때때로 이 안내서에서 명령이 한 줄에 모두 표시되지 않는 경우가 있습니다. 이런 경우에는 줄 끝에 공백과 백슬래시(\)를 표시하여 명령이 다음 줄에서 계속됨을 나타냅니다. **참고:** 실제 명령을 입력할 때는 이러한 백슬래시를 포함하지 말고 줄바꿈 없이 명령을 한 줄에 입력하십시오. 백슬래시 및 줄바꿈은 실제 명령 구문에 포함되지 않습니다.

예제: 명령 표기 규칙

다음 코드는 이 안내서에서 명령 규칙이 사용되는 방식을 보여 줍니다.

ruler *className* [props({all|{*propertyName1*[,*propertyName2*]...})]

설명:

- 표시되는 그대로 입력해야 하는 명령 이름(ruler)은 일반 고정 폭 글꼴로 표시됩니다.
- *className* 옵션은 클래스 이름(예: USER)이 들어갈 자리이므로 기울임꼴로 표시됩니다.

- 대괄호로 묶인 두 번째 부분은 선택적 피연산자를 의미하므로 이 부분 없이 명령을 실행할 수도 있습니다.
- 옵션 매개 변수(props)를 사용할 때 키워드 *all* 을 선택하거나 하나 이상의 속성 이름을 쉼표로 구분하여 지정할 수 있습니다.

파일 위치 규칙

CA Access Control 설명서는 다음과 같은 파일 위치 규칙을 따릅니다.

- *ACInstallDir* - 기본 CA Access Control 설치 디렉터리입니다.
  - Windows - C:\Program Files\CA\AccessControl\
  - UNIX - /opt/CA/AccessControl/
- *ACSharedDir* - UNIX 에서 CA Access Control 에 의해 사용되는 기본 디렉터리입니다.
  - UNIX - /opt/CA/AccessControlShared
- *ACServerInstallDir* - 기본 CA Access Control 엔터프라이즈 관리 설치 디렉터리입니다.
  - /opt/CA/AccessControlServer
- *DistServerInstallDir* - 기본 배포 서버 설치 디렉터리입니다.
  - /opt/CA/DistributionServer
- *JBoss_HOME* - 기본 JBoss 설치 디렉터리입니다.
  - /opt/jboss-4.2.3.GA

CA 에 문의

기술 지원팀에 문의

온라인 기술 지원 및 지사 목록, 기본 서비스 시간, 전화 번호에 대해서는 http://www.ca.com/worldwide에서 기술 지원팀에 문의하십시오.

설명서 변경 사항

이 설명서가 마지막으로 릴리스된 이후에 다음과 같이 업데이트되었습니다.

- 파일 및 프로그램 보호 - 다음과 같은 변경 내용을 포함하여 업데이트되었습니다.
  - setuid 및 setgid 프로그램 보호

목차

제 1 장: 소개

이 섹션은 다음 항목을 포함하고 있습니다.

안내서 정보

이 안내서에서는 개방형 시스템에 토탈 보안 솔루션을 제공하는 제품인 UNIX 용 CA Access Control 에서 사용하는 개념에 대해 설명하고 UNIX 끝점 관리 작업과 개념에 대해 설명합니다.

이 안내서는 또한 엔터프라이즈 관리 및 보고 기능과 고급 정책 관리 기능을 제공하는 CA Access Control Enterprise Edition 과 함께 제공됩니다.

용어를 간단히 나타내기 위해 이 안내서에서는 제품을 CA Access Control 이라고 합니다.

본 안내서의 사용자

이 안내서는 CA Access Control 보호 환경을 구현 및 유지 보수하는 보안 관리자와 시스템 관리자를 대상으로 합니다.

제 2 장: 끝점 관리

CA Access Control 은 개방형 시스템을 위한 포괄적인 액티브 보안 소프트웨어 솔루션으로서 운영 체제와 동적으로 연결되어 있습니다. 사용자가 파일 열기, 사용자 ID 대체, 네트워크 서비스 획득 등과 같이 보안상 중요한 작업을 요청할 때마다 CA Access Control 은 실시간으로 이벤트를 차단하고 표준 운영 체제(OS) 기능으로 제어권을 넘겨주기 전에 그 유효성을 평가합니다.

이 섹션은 다음 항목을 포함하고 있습니다.

CA Access Control 란 무엇입니까?

CA Access Control 은 사용자 기본 플랫폼의 보안을 관리하는 강력한 도구를 제공하여 기업의 보안 요구 사항에 완벽하게 맞춰 사용자 지정할 수 있는 보안 정책을 구현할 수 있습니다. CA Access Control 을 사용하면 사용자, 그룹 및 리소스에 대해 기본 운영 체제에서 제공되는 보안 이상의 보안을 제공할 수 있으며, 한 곳에서 조직 전체의 보안을 관리하고 서로 다른 환경의 Windows 및 UNIX 보안 정책을 통합할 수 있습니다.

UNIX 를 보호해야 하는 이유

여러 운영 체제에는 하나 이상의 기술을 사용하는 자체 액세스 제어 기능이 있습니다-. 완성도 높은 메인프레임 운영 체제인 IBM 의 z/OS 에는 사용자의 권한을 확인하기 위해 운영 체제 자체에서 실행하는 호출 집합인 SAF(시스템 권한 부여 기능)가 포함되어 있습니다.

z/OS 환경의 Access Control 소프트웨어는 SAF 호출에 대한 반환 코드를 설정하며 z/OS 는 이 코드에 따라 액세스를 부여하거나 거부합니다. 설정되는 반환 코드는 보안 관리자가 보안 데이터베이스에 정의한 액세스 규칙 및 정책을 기반으로 결정됩니다.

OS/2 와 같은 다른 운영 체제에서는 액세스 제어를 위해 이와 유사한 기술을 제공합니다. SES(보안 활성화 서비스)라는 OS/2 액세스 제어 모듈은 z/OS SAF 와 같은 개념을 기반으로 하고 있습니다.

그러나 UNIX 기반 운영 체제는 이러한 방식으로 디자인되지 않았습니다. 권한 부여는 주로 파일 액세스에 대해 지정되고 파일의 *inode* 항목에서 9 비트(rwx-rwx-rwx)를 사용하여 운영 체제에서 수행됩니다. SAF 와는 달리 이벤트 차단을 위한 종료점을 제공하지 않습니다. 그러므로 이러한 메인프레임 유형의 보안 패키지 기능보다 복잡한 기능을 수행하려면 보다 철저한 보안이 필요합니다.

Access Control 작동 원리

액세스 규칙 데이터베이스, 감사 로그, 관리 도구와 같은 일반 보안 기능을 제공하는 것 외에도 CA Access Control 은 보호해야 하는 운영 체제 이벤트를 차단합니다. CA Access Control 은 다양한 운영 체제에서 작동해야 하므로 이벤트를 메모리에서 차단합니다. 시스템 파일에는 아무 변경 사항이 없으며 운영 체제에도 전혀 영향을 주지 않습니다.

Access Control 이 보호하는 엔티티

CA Access Control 은 다음과 같은 항목을 보호합니다.

- **파일**

  사용자가 특정 파일에 대한 액세스 권한을 부여받았습니까?

  CA Access Control 은 파일에 액세스할 수 있는 사용자의 능력을 제한합니다. 사용자에게 READ, WRITE, EXECUTE, DELETE 및 RENAME 과 같은 하나 이상의 액세스 유형을 지정할 수 있습니다. 개별 파일에 대한 액세스를 지정하거나, 이름이 유사한 파일의 집합에 대한 액세스를 지정할 수 있습니다.

- **터미널**

  사용자에게 특정 터미널을 사용할 권한이 있습니까?

  이 검사는 로그인 프로세스 중에 수행됩니다. 개별 터미널 및 터미널 그룹은 CA Access Control 데이터베이스에서 정의할 수 있으며 터미널이나 터미널 그룹을 사용할 수 있는 사용자 또는 사용자 그룹을 설명하는 액세스 규칙도 함께 정의할 수 있습니다. 터미널 보호를 사용하면 권한 없는 터미널이나 스테이션을 통해 강력한 권한을 가진 사용자 계정에 로그인할 수 없습니다.

- **로그온 시간**

  사용자가 특정 요일의 특정 시간에 로그온할 수 있는 권한을 부여받았습니까?

  대부분의 사용자는 스테이션을 주중과 근무 시간에만 사용합니다. 휴일 제한 뿐만 아니라 시간 및 요일 로그인 제한은 해커와 권한 없는 다른 접근자로부터 보호를 제공합니다.

- **TCP/IP**

  다른 스테이션이 로컬 컴퓨터에서 TCP/IP 서비스를 받을 수 있는 권한을 부여받았습니까? 다른 스테이션이 로컬 컴퓨터에 TCP/IP 서비스를 제공할 권한이 있습니까? 다른 스테이션이 로컬 스테이션의 모든 사용자로부터 서비스를 수신할 수 있습니까?

  컴퓨터와 네트워크가 모두 개방되어 있는 개방형 시스템의 장점은 단점이 되기도 합니다. 컴퓨터를 외부에 연결하면 누가 시스템에 들어오고 외부 사용자가 고의적으로 또는 실수로 어떠한 피해를 입힐 수 있는지 확신할 수 없습니다. CA Access Control 에는 로컬 스테이션과 서버가 알 수 없는 스테이션에 서비스를 제공하는 것을 방지하는 "방화벽"이 있습니다.

- **다중 로그인 권한**

  사용자가 두 번째 터미널에서 로그인할 수 있습니까?

  *동시 로그인*이라는 용어는 두 개 이상의 터미널에서 시스템에 로그인할 수 있는 사용자의 능력을 의미합니다. CA Access Control 에서는 사용자가 두 번 이상 로그인하지 못하게 할 수 있습니다. 이러한 기능을 통해 침입자는 이미 로그인되어 있는 사용자 계정에 로그인할 수 없습니다.

- **사용자-정의된 엔터티**

  일반 항목(예: TCP/IP 서비스 및 터미널)과 *추상* 개체라고도 하는 기능 항목(예: 트랜잭션 수행 및 데이터베이스의 레코드 액세스)을 모두 정의하고 보호할 수 있습니다.

- **관리자 권한 측면**

  CA Access Control 에서는 슈퍼 사용자 권한을 운영자에게 위임하고 슈퍼 사용자 계정의 권한을 제한할 수 있습니다.

- **사용자-대체**

  사용자가 자신의 사용자 ID 를 대체할 권한이 있습니까?

  CA Access Control 은 운영 체제에서 제공하는 가장 중요한 서비스 중 하나인 UNIX *setuid* 시스템 호출을 차단하여 사용자에게 대체를 수행할 권한이 있는지 여부를 확인합니다. 사용자 대체 권한 검사에는 사용자가 특정 프로그램을 통해서만 사용자 ID 를 대체할 수 있도록 하는 프로그램 경로 지정이 포함됩니다. 이 검사는 root 로 대체하여 root 액세스 권한을 얻을 수 있는 사용자를 제어하는 데 특히 중요합니다.

- **그룹-대체**

  사용자가 newgrp(그룹-대체) 명령을 실행할 수 있는 권한을 부여받았습니까?

  그룹-대체 보호는 사용자-대체 보호와 유사합니다.

- **Setuid 및 setgid 프로그램**

  특정 setuid 또는 setgid 프로그램을 트러스트할 수 있습니까? 사용자가 이 프로그램을 호출할 권한이 있습니까?

  보안 관리자는 setuid 또는 setgid 실행 파일로 표시된 프로그램을 테스트하여 이들 프로그램에 권한을 부여받지 못한 접근자가 침입할 수 있는 보안상의 허점은 없는지 확인할 수 있습니다. 이 테스트를 통과하여 안전한 것으로 여겨지는 프로그램은 트러스트된 프로그램으로 정의됩니다. CA Access Control *watchdog* 이라고도 하는 CA Access Control 자체-보호 모듈은 특정 시간에 어떤 프로그램이 제어되고 있는지를 파악하고, 이 프로그램이 트러스트된 것으로 분류된 이후 수정 또는 이동되었는지 여부를 확인합니다. 트러스트된 프로그램이 수정 또는 이동된 경우에는 더 이상 트러스트된 것으로 간주되지 않으며 CA Access Control 은 프로그램의 실행을 허용하지 않습니다.

또한 CA Access Control 은 의도하거나 의도하지 않은 다음과 같은 위협으로부터 보호합니다.

- **중지 시도**

  CA Access Control 을 사용하면 중지 시도로부터 중요한 서버와 서비스 또는 데몬을 보호할 수 있습니다.

- **암호 공격**

  CA Access Control 은 여러 유형의 암호 공격으로부터 보호하며 사이트의 암호-정의 정책을 시행하고 침입-시도를 감지합니다.

- **암호를 이용한 범죄**

  CA Access Control 정책은 사용자가 최적의 암호를 만들어 사용하도록 하는 규칙을 설정합니다. 사용자가 적합한 암호를 만들어 사용할 수 있도록 CA Access Control 은 암호의 최대 및 최소 수명을 설정하고, 특정 단어를 사용하지 못하게 하며, 문자를 반복해서 사용할 수 없도록 하고, 기타 제한 사항을 적용할 수 있습니다. 암호는 너무 오랫동안 사용할 수 없습니다.

- **계정 관리**

  CA Access Control 정책을 통해 유휴 계정이 적절히 처리되도록 합니다.

- **도메인 관리**

  CA Access Control 은 암호 보호를 구현하고 NIS 및 비-NIS 도메인에 모두 보안을 적용할 수 있습니다.

보호 방법

CA Access Control 은 운영 체제에서 초기화가 완료된 후 바로 시작됩니다. CA Access Control 은 보호가 필요한 시스템 서비스에 후크를 배치합니다. 이런 방식으로 서비스가 수행되기 전에 CA Access Control 로 제어가 전달됩니다. CA Access Control 은 해당 사용자에게 서비스를 허용할지 여부를 결정합니다.

예를 들어 사용자는 CA Access Control 에 의해 보호되는 리소스에 액세스하려고 시도할 수 있습니다. 이러한 액세스를 요청하면 리소스를 열기 위해 커널에 대한 시스템 호출이 생성됩니다. CA Access Control 은 해당 시스템 호출을 차단하고 액세스 권한을 부여할지 여부를 결정합니다. 사용 권한이 부여되면, CA Access Control 은 일반 시스템 서비스로 제어를 넘겨줍니다. CA Access Control 이 사용 권한을 거부하면, 시스템 호출을 활성화한 프로그램에 표준 사용 권한-거부 오류 코드를 반환하고 시스템 호출이 종료됩니다.

데이터베이스에 정의된 액세스 규칙 및 정책을 기준으로 결정이 이루어집니다. 데이터베이스는 접근자와 리소스라는 두 가지 개체 유형을 설명합니다. *접근자*는 사용자 및 그룹이며, *리소스*는 파일과 서비스와 같은 보호할 개체입니다. 데이터베이스의 각 레코드는 접근자 또는 리소스에 대해 설명합니다.

각 개체는 동일한 유형의 개체 모음인 클래스에 속합니다. 예를 들어, TERMINAL 은 CA Access Control 로 보호되는 터미널(워크스테이션) 개체를 포함하는 클래스입니다.

클래스 활성화

CA Access Control 은 CLASS 가 데이터베이스에서 활성 상태인지 비활성 상태인지에 대한 정보를 저장합니다. CA Access Control 이 시작되면 활성 클래스 목록을 SEOS_syscall 에 전달하므로 CA Access Control 은 계속 이들 클래스를 차단할 필요가 없습니다. CA Access Control 이 클래스를 차단하는 경우는 사용자가 클래스의 활동 상태를 변경할 때뿐입니다. 클래스가 비활성인 경우 리소스에 대한 액세스는 차단되지 않습니다.

FILE, HOST, TCP, CONNECT 및 PROCESS 클래스에 비활성 클래스 바이패스를 사용할 수 있습니다.

접근자 요소

각 사용자는 데이터베이스에 있는 사용자 레코드가 내부 메모리에 반영된 형태인 *접근자 요소*(ACEE)로 나타납니다. CA Access Control 은 로그인 프로세스 중에 접근자 요소를 작성합니다. 접근자 요소는 사용자 프로세스와 연결되어 있습니다. 프로세스가 CA Access Control 로 보호되는 시스템 서비스를 요청하거나 암시적으로 리소스 액세스를 요청할 때마다 CA Access Control 이 해당 리소스 레코드에 액세스합니다. 그런 다음 사용자의 보안 수준, 모드 및 그룹과 같이 이미 작성된 접근자 요소의 정보로 사용자가 리소스에 액세스할 수 있는지 여부를 결정합니다.

기본 Windows 보안 확장

다음과 같은 CA Access Control 기능은 기본 보안을 확장합니다.

슈퍼 사용자 계정 제한 사항

운영 체제를 관리하는 사용자는 일반적으로 UNIX 시스템의 root 계정 및 Windows 시스템의 Administrator 계정과 같이 시스템 설치 중에 자동으로 생성되는 미리 정의된 계정의 구성원입니다. 미리 정의된 각 계정은 일련의 특정 시스템 기능을 실행하기 위해 존재합니다.

root 또는 Administrator 역할을 수행하는 사용자는 사용자 작성, 삭제 및 수정에서 서버 잠금, 재구성 및 종료에 이르는 광범위한 작업을 수행할 수 있습니다.

이러한 운영 체제에서 가장 큰 보안 위험 중 하나는 권한 없는 사용자가 이러한 계정을 제어할 수 있다는 점입니다. 이런 경우 권한 없는 사용자로 인해 시스템에 엄청난 손상이 발생할 수 있습니다.

CA Access Control 을 통해 이러한 계정에 부여되는 권한을 제한하고 이러한 계정이 구성원으로 포함된 사용자 그룹의 구성원인 사용자 권한을 제한할 수 있습니다. 이 기능을 통해 운영 시스템의 취약성을 보완할 수 있습니다.

CA Access Control 관리자

CA Access Control 을 설치할 때 하나 이상의 CA Access Control 관리자 이름을 지정하라는 요청을 받았습니다. CA Access Control 관리자는 규칙 데이터베이스를 일부 수정하거나 전부 수정할 수 있는 권한을 가지고 있습니다. 모든 권한을 가진 관리자는 한 명 이상이어야 합니다. 관리자는 원하는 대로 액세스 규칙을 수정하거나 작성할 수 있으며, 다른 수준의 관리자를 지정할 수 있습니다.

시스템 사용자를 정의한 후에 다른 사용자에게 ADMIN 특성을 할당하여 관리 권한을 할당할 수 있습니다.

참고: ADMIN 특성을 가진 사용자는 강력한 권한을 보유합니다. 따라서 ADMIN 사용자의 수는 엄격히 제한되어야 합니다. CA Access Control 보안 관리자를 한 명 이상 설정한 후 슈퍼 사용자에서 ADMIN 특성을 제거하여 기본 슈퍼 사용자와 ADMIN 의 역할을 구분하는 것도 좋습니다.

데이터베이스를 관리할 수 있는 권한을 가진 사용자가 항상 한 명 이상 필요하므로, CA Access Control 에서 ADMIN 특성을 가진 마지막 사용자는 삭제할 수 없습니다.

CA Access Control 관리자 중 한 명 이상이 워크스테이션에서 다른 호스트를 관리할 것으로 예상할 경우, 해당 호스트의 데이터베이스에 있는 규칙이 워크스테이션에서 READ 및 WRITE 액세스를 해당 관리자에게 부여하는지 확인하십시오.

하위 관리

CA Access Control 에는 *하위 관리* 기능이 있습니다. 이 기능을 통해 관리자는 일반 사용자가 특정 클래스를 관리하게 하는 특정 권한을 부여할 수 있습니다. 이러한 사용자를 하위 관리자라고 합니다.

예를 들어, 특정 사용자에게 사용자와 그룹만 관리하도록 허용할 수 있습니다.

또한 특정 클래스에 대한 액세스를 허용하면서 해당 클래스의 지정된 레코드에 대한 액세스도 허용함으로써 더 높은 수준의 하위 관리를 지정할 수 있습니다.

일반 사용자를 위한 관리자 권한

CA Access Control 은 일반 사용자(관리자가 아닌 사용자)에게 필요한 권한을 부여하여 해당 사용자가 관리자 그룹의 구성원이 아니더라도 관리 작업을 수행할 수 있습니다. 이런 세분화된 방법으로 관리 권한을 부여하여 작업을 위임하는 기능은 CA Access Control 의 중요한 장점입니다.

- SUDO 클래스의 레코드에는 사용자가 빌린 권한으로 스크립트를 실행할 수 있는 명령 스크립트가 저장되어 있습니다.
- 데이터 속성 값이 명령 스크립트입니다. 값에 선택적인 스크립트 매개 변수 값을 추가함으로써 값을 수정할 수 있습니다.
- SUDO 클래스의 각 레코드는 다른 사용자로부터 사용 권한을 빌려 올 수 있는 명령을 식별합니다.
- SUDO 클래스 레코드의 키는 SUDO 레코드 이름입니다. 이 이름은 사용자가 SUDO 레코드에서 명령을 실행할 때 명령 이름 대신 사용됩니다.

프로그램 경로 지정

*프로그램 경로 지정*은 특정 프로그램을 통해서만 파일에 액세스하도록 하는 파일 관련 액세스 규칙입니다. 프로그램 경로 지정을 통해 중요한 파일의 보안이 상당히 향상됩니다. CA Access Control 에서는 프로그램 경로 지정을 사용하여 시스템의 파일에 대한 보호를 추가로 제공할 수 있습니다.

B1 보안 수준 인증

CA Access Control 에는 보안 수준, 보안 범주, 보안 레이블 등의 B1 "Orange Book" 기능이 포함되어 있습니다.

- 데이터베이스의 접근자와 리소스에 *보안 수준*을 할당할 수 있습니다. 보안 수준은 1 에서 255 사이의 정수입니다. 접근자는 보안 수준이 리소스에 할당된 보안 수준과 같거나 높은 경우에만 리소스에 액세스할 수 있습니다.
- 데이터베이스의 접근자와 리소스는 하나 이상의 *보안 범주*에 속할 수 있습니다. 접근자는 리소스에 할당된 모든 보안 범주에 속해 있을 경우에만 리소스를 액세스할 수 있습니다.

- *보안 레이블*은 특정 보안 수준을 0 개 이상의 보안 범주 집합에 연결하는 이름입니다. 사용자에게 보안 레이블을 할당하면 보안 수준과 보안 레이블에 관련된 보안 범주가 사용자에게 모두 부여됩니다.

참고: B1 Orange Book 기능에 대한 자세한 내용은 *구현 안내서*를 참조하십시오.

끝점 관리

CA Access Control 에서는 기업의 리소스를 관리하고 리소스에 액세스할 수 있는 사용자를 제어하게 하는 두 가지 방법을 제공합니다.

- **selang** - CA Access Control 명령 언어입니다.

  selang 명령 언어를 사용하면 CA Access Control 데이터베이스에서 정의를 만들 수 있습니다. selang 명령 언어는 명령 정의 언어입니다.

  참고: selang 사용에 대한 자세한 내용은 *selang 참조 안내서*를 참조하십시오.

- **CA Access Control 끝점 관리** - 끝점 관리 인터페이스입니다.

  이 웹 기반 인터페이스에서는 중앙 관리 서버를 통해 원격 끝점을 관리할 수 있습니다.

  참고: CA Access Control 끝점 관리 설치에 대한 자세한 내용은 *구현 안내서*를 참조하십시오.

제 3 장: 사용자 및 그룹 관리

이 섹션은 다음 항목을 포함하고 있습니다.

사용자 및 그룹

CA Access Control 에서 모든 작업이나 액세스 시도는 요청을 제출할 책임이 있는 사용자를 위해 수행됩니다. 따라서 시스템의 모든 프로세스는 특정 사용자 이름과 연결되어 있습니다. CA Access Control 은 사용자 이름으로 사용자를 식별합니다.

*사용자*는 배치 또는 데몬 프로그램의 소유자가 될 수 있거나 로그온할 수 있는 개인입니다. CA Access Control 에서 모든 액세스 시도는 사용자가 수행합니다. CA Access Control 은 CA Access Control 데이터베이스 및 엔터프라이즈 사용자 저장소에 있는 사용자 정보를 사용할 수 있습니다. 사용자 정보는 데이터베이스의 USER 레코드나 XUSER 레코드에 저장됩니다.

참고: *엔터프라이즈 사용자 저장소*는 사용자나 그룹을 저장하는 운영 체제의 저장소입니다(예: UNIX 의 /etc/passwd 및 /etc/groups 또는 Windows 의 Active Directory)

*그룹*은 사용자 집합입니다. 그룹은 해당 그룹의 사용자에 대한 공통 액세스 규칙을 정의합니다. 그룹은 중첩될 수 있습니다(다른 그룹에 속함). CA Access Control 은 CA Access Control 데이터베이스와 엔터프라이즈 사용자 저장소에서 그룹 정보를 사용할 수 있습니다. 일반적으로 그룹을 작성한 다음 database_administrators 와 같은 역할을 기준으로 사용자를 이 그룹에 할당합니다.

사용자 레코드는 주요 접근자 레코드입니다. CA Access Control 에서 그룹을 사용하는 주요 목적은 그룹의 모든 사용자에게 액세스 권한을 한 번에 할당하기 위해서입니다. 액세스 권한을 한 번에 할당하는 것이 각 사용자에게 개별적으로 할당하는 것보다 더 간편하고 오류가 덜 발생합니다.

접근자 정보가 저장된 위치

CA Access Control 에서 사용하는 사용자 및 그룹 정보는 CA Access Control 데이터베이스와 호스트 운영 체제에 저장됩니다. 호스트 운영 체제 정보 저장소를 *엔터프라이즈 사용자 저장소* 또는 *엔터프라이즈 저장소*라고 합니다. 기본적으로 CA Access Control 은 엔터프라이즈 저장소를 사용하지 않도록 구성됩니다. 그러나 CA Access Control 이 데이터베이스에 정의된 사용자나 그룹을 찾을 수 없는 경우 엔터프라이즈 저장소에 정의된 사용자 및 그룹 구성원을 검색하고 해당 정보를 사용하도록 구성할 수 있습니다.

참고: CA Access Control 에서는 엔터프라이즈 저장소에서 정보를 사용할 뿐만 아니라 네이티브 환경에서 selang 명령을 사용할 경우 엔터프라이즈 저장소에 쓰기도 합니다.

권한 부여를 확인할 때 CA Access Control 에서는 항상 자체 데이터베이스에 정의된 접근자를 확인한 후 엔터프라이즈 저장소를 확인합니다. CA Access Control 데이터베이스에 정의된 사용자와 동일한 이름을 가진 엔터프라이즈 사용자가 있는 경우에는 엔터프라이즈 사용자는 CA Access Control 에서 무시됩니다.

CA Access Control 이 사용자 레코드를 찾는 방법

사용자가 로그인할 때 CA Access Control 은 사용자와 관련된 레코드를 찾을 때까지 다음 순서대로 검색을 수행합니다.

1. CA Access Control 은 데이터베이스에 정의된 사용자를 검색합니다.
2. CA Access Control 은 캐시에서 해당 이름을 가진 엔터프라이즈 사용자를 검색합니다.

   네트워크의 연결이 끊긴 경우 운영 체제(OS)는 OS 에 캐시된 자격 증명을 사용하여 사용자가 로그인할 수 있게 합니다. CA Access Control 캐시의 목적은 이러한 상황에서 CA Access Control 이 엔터프라이즈 사용자의 레코드를 사용할 수 있게 하는 것입니다.
3. CA Access Control 은 운영 체제를 사용하여 엔터프라이즈 사용자 저장소에서 해당 이름을 가진 사용자를 검색합니다.
4. CA Access Control 이 데이터베이스나 엔터프라이즈 저장소에서 사용자와 관련된 레코드를 찾지 못하면 CA Access Control 은 사용자에게 _undefined USER 레코드에 있는 속성을 할당합니다.

엔터프라이즈 사용자 저장소와 통합

일반적으로 CA Access Control 에서 엔터프라이즈 사용자 저장소에 정의된 그룹과 사용자를 사용하도록 구성합니다.

CA Access Control 을 이와 같이 구성하면 기본적으로 엔터프라이즈 사용자 저장소를 참조하는 액세스 규칙이 작성되거나 사용자가 운영 체제에 로그인할 때 CA Access Control 은 데이터베이스에서 기존 레코드가 없는 경우 해당 사용자나 그룹에 대한 레코드를 생성합니다. 이러한 레코드에는 XUSER(엔터프라이즈 사용자용) 또는 XGROUP(엔터프라이즈 그룹용) 클래스가 있습니다. 레코드에는 CA Access Control 에서 액세스 규칙을 적용하기 위해 필요한 속성이 저장되어 있습니다. CA Access Control 에서 필요에 따라 레코드를 생성하므로 레코드를 관리할 필요가 없습니다.

CA Access Control 이 엔터프라이즈 사용자 저장소에서 가져오는 엔터프라이즈 사용자 또는 그룹의 속성은 이름과 그룹 구성원 속성뿐입니다.

엔터프라이즈 저장소에서 접근자 관리에 대한 지침

엔터프라이즈 사용자 저장소에서 접근자를 관리하려면 다음 절의 지침을 고려해야 합니다.

데이터베이스에 정의해야 하는 사용자 및 그룹

CA Access Control 은 일부 사용자와 그룹을 엔터프라이즈 사용자 저장소가 아니라 데이터베이스에 저장하도록 합니다. 해당되는 정보는 다음과 같습니다.

- 미리 정의된 사용자 (페이지 38)
- 미리 정의된 그룹 (페이지 39)
- CA Access Control 관리자
- 프로필 그룹
- 논리적 사용자

엔터프라이즈 사용자 사용 제한 사항

CA Access Control 에서는 엔터프라이즈 사용자 사용에 대해 다음 제한 사항을 적용합니다.

- 데이터베이스에 동일한 이름을 가진 사용자가 정의되어 있는 경우에는 CA Access Control 에서 엔터프라이즈 사용자를 생성하거나 참조할 수 없습니다.
- selang AC 환경을 사용하여 엔터프라이즈 사용자를 작성, 삭제 또는 수정할 수 없습니다.
- 엔터프라이즈 사용자를 논리적 사용자로 사용할 수 없습니다.
- 기본적으로 엔터프라이즈 사용자 저장소에 사용자가 이미 정의되어 있지 않으면 CA Access Control 에 엔터프라이즈 사용자를 만들 수 없습니다. 그러나 UNIX 시스템에서는 이 동작을 활성화 또는 비활성화할 수 있습니다.

추가 정보:

UNIX 에서 XUSER 레코드를 생성하기 전에 엔터프라이즈 저장소 검사 활성화 또는 비활성화 (페이지 34)

엔터프라이즈 그룹 사용 제한 사항

CA Access Control 에서는 엔터프라이즈 그룹 사용에 대해 다음 제한 사항을 적용합니다.

- selang AC 환경에서는 엔터프라이즈 그룹을 작성하거나 삭제할 수 없습니다.
- selang AC 환경에서는 엔터프라이즈 그룹 구성원을 변경할 수 없습니다.
- 엔터프라이즈 그룹을 프로필 그룹 (페이지 40)으로 사용할 수 없습니다.

엔터프라이즈 사용자 및 그룹 사용 활성화 또는 비활성화

기본적으로 CA Access Control 에서는 엔터프라이즈 사용자 저장소에 정의된 그룹과 사용자를 사용할 수 없지만 사용하도록 CA Access Control 을 활성화할 수 있습니다. 이전 CA Access Control 버전과의 호환성이 필요한 경우가 아니면 이 기능을 활성화하는 것이 좋습니다.

CA Access Control 에서 엔터프라이즈 사용자와 그룹을 사용하게 하려면 구성 설정 osuser_enabled 를 'yes'로 설정하십시오. 이 동작을 비활성화하려면 osuser_enabled 값을 'no'로 설정하십시오.

예: Windows 에서 엔터프라이즈 사용자 및 그룹 사용 활성화

다음 레지스트리 설정은 Windows 에서 엔터프라이즈 사용자 및 그룹의 사용을 활성화합니다.

- 키: HKLM\SOFTWARE\ComputerAssociates\AccessControl\OS_user
- 이름: osuser_enabled
- 유형: REG_DWORD
- 값: yes

예: UNIX 에서 엔터프라이즈 사용자 및 그룹 사용 활성화

다음 명령은 CA Access Control 을 중지하고 UNIX 에서 엔터프라이즈 사용자 및 그룹 사용을 활성화한 다음 CA Access Control 을 다시 시작합니다.

```
secons -s
seini -s OS_User.osuser_enabled yes
seload
```

엔터프라이즈 사용자 로그인 시 XUSER 레코드 생성 활성화 또는 비활성화

CA Access Control 은 엔터프라이즈 사용자를 사용하도록 활성화된 경우 기본적으로 해당 사용자가 로그인할 때 사용자에 대한 레코드(XUSER 클래스에서)를 생성합니다. 매일 같은 시간에 수천 명의 사용자가 로그인하는 경우 이 작업을 수행하지 않을 수 있습니다.

사용자가 로그인할 때 CA Access Control 에서 XUSER 레코드를 생성하지 않게 하려면 구성 설정 create_user_in_db 값을 0(영)으로 변경합니다. 이 동작을 다시 활성화하려면 값을 1(일)로 설정합니다.

예: Windows 에서 엔터프라이즈 사용자 로그인 시 XUSER 레코드의 자동 생성 비활성화

다음 레지스트리 설정은 Windows 에서 CA Access Control 의 엔터프라이즈 사용자 레코드 자동 생성을 비활성화합니다.

- 키: HKLM\Software\ComputerAssociates\AccessControl\OS_user
- 이름: create_user_in_db
- 유형: REG_DWORD
- 값: 0

예: UNIX 에서 엔터프라이즈 사용자 로그인 시 XUSER 레코드의 자동 생성 비활성화

다음 명령은 CA Access Control 을 중지하고 UNIX 에서 XUSER 의 자동 생성을 비활성화한 다음 CA Access Control 을 다시 시작합니다.

```
secons -s
seini -s OS_User.create_user_in_db 0
seload
```

UNIX 에서 XUSER 레코드를 생성하기 전에 엔터프라이즈 저장소 검사 활성화 또는 비활성화

사용자가 엔터프라이즈 사용자 저장소에 정의되어 있지 않으면 CA Access Control 에서 엔터프라이즈 사용자를 생성할 수 없습니다. Windows 의 경우 사용자가 Windows 사용자 저장소에 정의되어 있지 않으면 CA Access Control 에서 엔터프라이즈 사용자를 생성할 수 없습니다. UNIX 의 경우 기본 동작이 Windows 와 반대입니다. 그러나 UNIX 의 경우 이 기본 동작을 활성화하거나 비활성화할 수 있습니다.

검사를 비활성화하여 일치하는 엔터프라이즈 사용자가 없을 경우 CA Access Control 에서 XUSER 레코드를 생성하게 하려면 구성 설정 verify_osuser 값을 0 으로 변경하십시오. 검사를 적용하려면 값을 1 로 설정합니다.

예: 엔터프라이즈 사용자 저장소 검사 없이 XUSER 레코드 생성 활성화

다음 명령 집합은 CA Access Control 을 중지하고 일치하는 엔터프라이즈 저장소 없이 XUSER 레코드 생성을 활성화한 다음 CA Access Control 을 다시 시작합니다.

```
secons -s
seini -s OS_User.verify_osuser 0
seload
```

Windows 에서 재사용된 엔터프라이즈 저장소 계정

*재사용 계정*은 삭제되었으나 같은 이름으로 다시 생성된 엔터프라이즈 저장소 사용자 또는 그룹입니다. 이는 예를 들어 사용자가 사임하는 경우 사용자 저장소에서 사용자를 제거한 다음 이전에 제거된 사용자와 같은 이름을 가진 새 사용자의 계정을 새로 만드는 것과 같습니다.

같은 이름을 가진 이전 계정에 부여했던 것과 동일한 액세스 권한을 새 접근자에게 부여할 필요가 없으므로 재사용 계정은 보안 문제를 일으킬 수 있습니다. 이 문제를 해결하기 위해 CA Access Control 권한 부여는 SID 를 기반으로 합니다. 따라서 기존 액세스 권한을 가진 삭제된 접근자와 동일한 이름을 가진 새 접근자를 만들 때 새 접근자는 이전 접근자의 이전 사용 권한을 자동으로 부여받지 않습니다.

중요! 재사용 계정 접근자는 이전 액세스 권한을 상속하지 *않습니다*. 그러나 데이터베이스 액세스 규칙에는 SID 가 아니라 접근자 이름이 표시되므로 이러한 규칙이 계속 적용되는 것처럼 보일 수 있습니다. 이를 확인하려면 secons -checkSID 명령을 사용합니다.

Windows 에서 재사용된 엔터프라이즈 계정 확인

데이터베이스 규칙이 관련되어 있는 엔터프라이즈 계정(사용자 또는 그룹)이 재사용(삭제된 후 같은 이름으로 생성)될 경우 이전 데이터베이스 규칙이 계속 새 계정에 적용되는 것처럼 보일 수 있습니다. 그러나 CA Access Control 권한 부여는 SID 를 기반으로 하므로 이러한 규칙은 더 이상 적용되지 않으며 새 그룹에 대한 새 규칙을 만들어야 합니다. 새 규칙을 만들려면 먼저 재사용 계정을 확인해야 합니다.

재사용 엔터프라이즈 계정을 확인하려면 명령 프롬프트를 열고 다음 명령을 실행합니다.

```
secons -checkSID -users
secons -checkSID -groups
```

CA Access Control 은 포함되어 있는 모든 엔터프라이즈 사용자 계정(XUSER 레코드)을 통과한 다음 모든 그룹 계정(XGROUP 레코드)을 통과하여 엔터프라이즈 계정 SID 와 다른 SID 를 가진 계정을 식별합니다. 그런 다음 명명 규칙 *SID*(*accountName*)를 사용하여 이러한 계정 이름을 변경합니다.

이제 재사용 계정에 대한 새 규칙을 만들 수 있습니다.

참고: 재사용 사용자 계정은 사용자가 로그인하거나 리소스에 액세스를 시도할 때 이런 방식으로 확인됩니다. 엔터프라이즈 계정을 만들 때 secons -checkSID 명령을 예약 작업으로 실행하는 것이 좋습니다.

예: 재사용 그룹 계정

회사 ABCD 에는 엔터프라이즈 저장소에 *interns* 라는 그룹이 있습니다. 이 그룹에는 productA 작업을 하는 아홉 명의 구성원이 있습니다. 관리자는 그룹을 CA Access Control 에 인식시키고 다음과 같이 그룹 구성원이 액세스하는 데 필요한 파일 액세스 권한을 그룹에 할당합니다.

```
nxg interns owner(msmith)
auth file c:\products\productA\materials\* xgid(interns) access(all)
auth file c:\HR\interns\* xgid(interns) access(read)
```

interns 가 ABCD 보유를 완료할 때 엔터프라이즈 저장소 관리자는 그룹을 삭제합니다. 3 개월 이후 구성원이 여섯 명인 새로운 interns 그룹이 같은 이름으로 엔터프라이즈 저장소에 생성됩니다. CA Access Control 데이터베이스에 이전 규칙이 계속 존재하므로 새 *interns* 그룹이 이전 그룹의 사용 권한을 상속한 것처럼 보입니다. 그러나 이러한 규칙은 이전 interns 그룹에만 적용되므로 CA Access Control 관리자는 새 그룹에 대한 새 규칙을 만들어야 합니다.

이렇게 하려면 관리자가 다음과 같이 재사용 interns 계정을 식별 및 확인해야 합니다.

```
secons -checkSID -groups interns
```

이 명령은 XGROUP 리소스 및 리소스에 대한 액세스 규칙 참조의 이름을 "*SID*(*domain*\interns)"로 변경합니다. 이제 관리자는 productB 작업을 수행하는 새 interns 그룹에 대한 새 규칙을 만들 수 있습니다.

```
nxg interns owner(msmith)
auth file c:\products\productB\materials\* xgid(interns) access(all)
auth file c:\HR\interns\* xgid(interns) access(read)
```

참고: secons 유틸리티에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

데이터베이스 접근자

다음 절에 설명된 대로 사용자 관리 방법에 관계없이 일부 접근자는 CA Access Control 데이터베이스에 정의되어야 합니다.

미리 정의된 사용자

CA Access Control 에서는 삭제할 수 없는 다음 사용자를 미리 정의합니다.

+devcalc

(Windows) CA Access Control 이 위반 계산 프로세스인 devcalc 를 실행하는 데 사용하는 사용자 이름입니다.

_dms

고급 정책 관리 서버 구성 요소 데이터베이스(DMS, DH 구독기 및 DH 작성기)에 설치되는 _dms 사용자는 policyfetcher 및 devcalc 에서 DH 및 DMS 와 통신하는 데 사용됩니다.

nobody

nobody 사용자는 실제 사용자와 일치할 수 없는 사용자 레코드입니다. 이 레코드를 사용하여 사용자에게 관련 사용 권한을 제공하지 않는 규칙을 만듭니다. 예를 들어 *nobody* 를 리소스 소유자로 설정하면 모든 사용자가 레코드 소유와 관련된 사용 권한을 부여받을 수 없습니다.

+reportagent

CA Access Control 이 보고서 에이전트를 실행하는 데 사용하는 사용자 이름입니다.

_seagent

_seagent 는 CA Access Control 이 다음과 같은 내부 프로세스를 실행하는 데 사용하는 사용자 이름입니다.

- PMDB 프로세스, sepmdd
- (UNIX) 위반 계산 프로세스, devcalc
- 사용자 및 그룹 레코드 업데이트 종료 프로세스

_seagent 사용자는 SERVER 특성을 가지고 있습니다.

_sebuildla

(UNIX) _sebuildla 사용자는 CA Access Control 데몬인 seosd 에 대한 참조(lookaside) 데이터베이스를 만들기 위해 CA Access Control 이 sebuildla 유틸리티를 실행하는 데 사용하는 사용자 이름입니다.

_seoswd

(UNIX)_seoswd 는 파일 정보와 데이터베이스에서 신뢰하는 프로그램으로 정의된 프로그램의 디지털 서명을 모니터링하기 위해 seoswd watchdog 데몬을 실행하는 데 사용되는 사용자 이름입니다.

_undefined

_undefined 는 CA Access Control 에 정의되지 않은 모든 사용자를 나타냅니다. _undefined 를 사용하여 정의되지 않은 사용자를 ACL 에 포함할 수 있습니다.

미리 정의된 그룹

CA Access Control 에는 미리 정의된 그룹이 제공됩니다. _interactive 및 _network 그룹을 제외하고, 다른 그룹에 수행하는 것과 동일한 방법으로 이러한 그룹에 사용자를 추가할 수 있습니다.

_abspath

로그인 시 사용자가 _abspath 그룹에 있는 경우 프로그램을 호출하려면 절대 경로 이름을 사용해야 합니다.

_interactive

사용자는 액세스 시도의 목적으로만 _interactive 그룹의 구성원입니다. 사용자는 액세스를 시도하고 있는 리소스와 동일한 호스트에 로그인되어 있는 경우 _interactive 그룹의 구성원입니다. CA Access Control 에서는 _interactive 그룹의 구성원을 동적으로 자동 관리하므로 사용자가 구성원을 변경할 수 없습니다.

_network

이 그룹은 _interactive 의 보조 그룹입니다. 사용자는 액세스 목적으로만 _network 그룹의 구성원입니다. 사용자는 리소스가 속한 호스트와 다른 호스트에서 리소스에 액세스하고 있는 경우 _network 그룹의 구성원입니다. CA Access Control 에서는 _network 그룹의 구성원을 동적으로 자동 관리하므로 사용자가 구성원을 변경할 수 없습니다.

_restricted

_restricted 그룹의 사용자에 대한 모든 파일과 Windows 의 레지스트리 키는 CA Access Control 로 보호됩니다. 파일이나 Windows 레지스트리 키에 명시적으로 정의된 액세스 규칙이 없는 경우에는 액세스 권한이 해당 클래스(FILE 또는 REGKEY)의 _default 레코드로 처리됩니다.

참고: _restricted 그룹의 사용자는 작업을 수행할 충분한 권한을 가질 수 없습니다. _restricted 그룹에 사용자를 추가하려면 초기에 경고 모드를 사용해 보십시오.

_surrogate

사용자가 _surrogate 그룹의 구성원을 대리 사용자로 사용할 경우 CA Access Control 에서는 대리 사용자 작업의 감사 기록에 전체 추적을 기록하고 원래 사용자 이름으로 태그를 지정합니다.

예: selang 을 사용하여 _restricted 그룹에 사용자 추가

다음 selang 명령은 _restricted 그룹에 엔터프라이즈 사용자 john_smith 를 추가합니다.

```
joinx john_smith group(_restricted)
```

프로필 그룹

*프로필 그룹*은 CA Access Control 데이터베이스에 정의되는 사용자 속성 기본값이 포함된 그룹입니다. 사용자에게 프로필 그룹을 할당하면 해당 값이 이미 사용자에 대해 설정된 경우가 아니면 프로필 그룹이 해당 값을 사용자에게 제공합니다.

사용자를 만들 때 사용자의 프로필 그룹을 지정하거나 나중에 프로필 그룹에 사용자를 할당할 수 있습니다.

프로필 그룹을 사용하여 관리자는 해당 그룹에 할당된 새 사용자를 위한 특정 권한을 포함한 표준 설정을 효율적으로 작성할 수 있습니다. 이 설정은 사용자의 홈 디렉터리, 감사 속성, 액세스 권한을 정의하는 PMDB, 프로필 그룹과 연결된 사용자에게 영향을 주는 다양한 암호 규칙과 같은 사항을 정의할 수 있습니다.

CA Access Control 이 프로필 그룹을 사용하여 사용자 속성을 파악하는 방법

다음 프로세스는 CA Access Control 이 프로필 그룹을 사용하여 사용자 속성을 파악하는 방법에 대해 설명합니다.

1. CA Access Control 은 USER 또는 XUSER 클래스에 있는 사용자의 레코드에 속성 값이 있는지 확인합니다.

   사용자의 레코드에 속성 값이 있으면 CA Access Control 은 이 값을 사용합니다.

2. CA Access Control 은 사용자가 프로필 그룹에 할당되었는지 여부를 확인합니다.

   사용자가 프로필 그룹에 할당된 경우 프로세스가 계속 진행됩니다. 사용자가 프로필 그룹에 할당되지 않은 경우 CA Access Control 은 기본 속성 값을 이 사용자에게 할당합니다.

3. CA Access Control 은 프로필 그룹에 이 속성 값이 있는지 확인합니다.

   프로필 그룹에 이 속성 값이 있는 경우 CA Access Control 은 이 값을 해당 사용자에게 할당합니다. 프로필 그룹에 속성 값이 없는 경우 CA Access Control 은 기본 속성 값을 이 사용자에게 할당합니다.

   참고: 사용자 또는 프로필 그룹의 감사 속성이 설정되어 있지 않으면 그룹의 감사 속성이 사용자의 감사 속성에 영향을 줄 수 있습니다.

추가 정보:

CA Access Control 이 사용자의 감사 모드를 결정하는 방법 (페이지 197)

접근자 관리

CA Access Control 끝점 관리 또는 selang 을 사용하여 데이터베이스나 엔터프라이즈 사용자 또는 그룹을 생성, 수정 및 삭제할 수 있습니다.

사용자 또는 그룹 관리

특정 접근자의 속성을 표시 또는 수정하거나 접근자를 삭제하려면 먼저 해당 접근자를 찾아야 합니다.

사용자 또는 그룹을 관리하려면

1. CA Access Control 끝점 관리에서 다음을 수행하십시오.
   a. 사용자를 클릭합니다.
   b. "사용자" *또는* "그룹" 하위 탭을 클릭합니다.

   선택에 따라 "사용자" 또는 "그룹" 페이지가 나타납니다.
2. "검색" 섹션에서 다음 필드를 완료합니다.

   사용자/그룹 이름

   찾을 접근자의 마스크를 정의합니다. 나중에 접근자의 전체 이름을 입력하거나 마스크를 사용할 수 있습니다. 예를 들어 이름에 "admin"이 포함된 접근자를 나열하려면 *admin*을 사용합니다.

   모든 접근자를 나열하려면 *(별표)를 사용하고 하나의 문자만 대체하려면 ?(물음표)를 사용하십시오.

   사용자/그룹 리포지토리

   접근자 목록을 가져오려는 소스를 지정합니다. 소스는 다음 중 하나가 될 수 있습니다.

   - **내부 계정** - CA Access Control 데이터베이스에 정의된 접근자입니다.
   - **엔터프라이즈 계정** - 특정 엔터프라이즈 사용자 저장소에 정의된 접근자입니다.

AC 계정/프로필만 표시합니다.

다음과 같이 CA Access Control 데이터베이스에 레코드가 있는 계정만 나열할지 여부를 지정합니다.

- "내부 계정"을 선택한 경우에는 CA Access Control 데이터베이스에 있는 계정(네이티브가 아닌 계정)만 나열합니다.
- 엔터프라이즈 계정을 선택한 경우에는 CA Access Control 엔터프라이즈 프로필이 있는 계정(XUSER 또는 XGROUP 레코드)만 나열합니다.

"실행"을 클릭합니다.

선택한 리포지토리에 있는 접근자 목록이 나타납니다.

3. 다음 작업 중 *하나*를 수행합니다.
   - "보기" 열에서  을 클릭하여 접근자의 속성을 표시합니다.
   - 접근자를 삭제하려면 "삭제" 열에서  을 클릭합니다.
   - 접근자의 속성을 수정하려면 접근자의 이름을 클릭합니다.
   - 삭제할 접근자를 선택하고 "삭제"를 클릭합니다.
   - CA Access Control 데이터베이스에서 사용자 또는 그룹 레코드를 만들려면 "사용자 만들기" 또는 "그룹 만들기"를 클릭합니다.

예: 리포지토리에서 엔터프라이즈 사용자 검색

다음 그림은 ABC-DM1 엔터프라이즈 사용자 저장소에 있는 모든 사용자의 검색 결과를 보여 줍니다.

selang 을 사용한 사용자 관리

엔터프라이즈 사용자 레코드에 다음 selang 명령을 사용합니다.

- **newxusr** 및 **editxusr** - 새 엔터프라이즈 사용자 레코드 정의
- **chxusr** 및 **editxusr** - 엔터프라이즈 사용자의 CA Access Control 속성 변경
- **find xuser** - CA Access Control 레코드가 있는 엔터프라이즈 사용자 나열
- **rmxusr** - 사용자 삭제
- **show xuser** - 엔터프라이즈 사용자의 CA Access Control 속성 표시

CA Access Control 데이터베이스 사용자 레코드에 다음 selang 명령을 사용합니다.

- **newusr** 및 **editusr** - 새 사용자 레코드 정의
- **chusr** 및 **editusr** - 사용자 속성 변경
- **rmusr** - 사용자 삭제
- **find user** - 데이터베이스 사용자 나열
- **show user** - 사용자 속성 표시

예: selang 을 사용하여 데이터베이스에서 사용자 정의

다음 selang 명령은 CA Access Control 데이터베이스에서 보안 수준 100 을 사용하여 새 사용자를 정의합니다.

```
newusr internalUser level(100)
```

예: selang 을 사용하여 엔터프라이즈 사용자 속성 변경

다음 selang 명령은 엔터프라이즈 사용자 Terry 에게 AUDITOR 속성을 제공합니다.

```
chxusr Terry auditor
```

selang 을 사용한 그룹 관리

엔터프라이즈 그룹의 이름과 구성원을 변경할 수 없다는 점을 제외하고 모든 그룹의 모든 속성을 변경할 수 있습니다(CA Access Control 내에서).

그룹 속성을 변경하거나 그룹과 관련된 액세스 권한을 할당하려면 CA Access Control 끝점 관리를 사용하거나 다음 selang 명령을 사용합니다.

- **join[-]** 및 **joinx[-]**

  내부 그룹 구성원 변경

  그룹에 내부 접근자를 추가하려면 join 을 사용합니다. 내부 그룹에 엔터프라이즈 그룹 및 사용자를 추가하려면 joinx 를 사용합니다. 접근자를 제거하려면 명령의 -(빼기) 양식을 사용합니다.

- **editgrp, newgrp, chgrp**

  내부 그룹의 비구성원 속성 변경

- **editxgrp, newxgrp, chxgrp**

  엔터프라이즈 그룹의 비구성원 속성 변경

- **rmgrp, rmxgrp**

  사용자 그룹 삭제

예: selang 을 사용하여 데이터베이스에서 그룹 정의

다음 selang 명령은 데이터베이스에서 새 그룹 "sales"를 정의합니다. 그룹의 전체 이름은 "Sales Department"입니다.

```
newgrp sales name('Sales Department')
```

예: selang 을 사용하여 데이터베이스에 정의된 그룹 속성 변경

다음 selang 명령은 CA Access Control 이 그룹 AC_admins 의 구성원에 대한 모든 이벤트를 감사하게 합니다.

```
chgrp AC_admins audit(all)
```

예: selang 을 사용하여 ACL 에 엔터프라이즈 그룹 추가

다음 selang 명령은 myfile 의 ACL 에 엔터프라이즈 그룹 mygroup 을 추가합니다.

```
Authorize FILE (myfile) xgid(mygroup)
```

예: selang 을 사용하여 데이터베이스에 정의된 그룹에 엔터프라이즈 사용자 추가

다음 selang 명령은 데이터베이스에 정의된 그룹 AC_admins 에 엔터프라이즈 사용자 mydomain\administrator 를 추가합니다.

```
joinx mydomain\administrator group(AC_admins)
```

예: selang 을 사용하여 데이터베이스에 정의된 그룹에 엔터프라이즈 그룹 추가

다음 selang 명령은 _restricted 그룹에 엔터프라이즈 그룹 Guests 를 추가합니다.

```
joinx Guests group(_restricted)
```

제 4 장: 리소스 관리

이 섹션은 다음 항목을 포함하고 있습니다.

리소스

*리소스*는 접근자가 액세스하고 액세스 규칙으로 보호할 수 있는 엔터티이거나 이 엔터티에 해당하는 CA Access Control 데이터베이스 레코드입니다. 리소스의 예로는 파일, 프로그램, 호스트 및 터미널이 있습니다.

CA Access Control 에서 리소스 레코드를 생성하는 주요 목적은 리소스 레코드와 일치하는 리소스의 액세스 사용 권한을 정의하기 위해서입니다. 리소스 액세스에 필요한 액세스 사용 권한은 리소스 레코드의 액세스 제어 목록에 지정됩니다.

리소스 그룹

*리소스 그룹*은 다른 리소스 목록을 포함하는 리소스입니다. 리소스 그룹은 CONTAINER, GFILE, GSUDO, GTERMINAL 또는 GHOST 클래스 중 하나의 구성원입니다.

리소스 그룹 자체가 리소스이기 때문에 동일한 속성을 구성원 리소스로 포함합니다. 따라서 리소스 그룹을 사용하면 관리를 간소화할 수 있습니다. 리소스 그룹의 속성을 변경하면 모든 구성원 리소스의 속성을 변경할 수 있습니다.

참고: Windows 에서 CA Access Control 은 리소스에 대한 사용자 권한 부여를 확인할 때 리소스 그룹 소유권을 고려합니다. 이 동작은 r12.0 에서 처음 도입되었습니다. 이전 릴리스에서는 권한 부여 프로세스에서 리소스의 소유자만 고려했습니다.

예를 들어, 기본 액세스와 소유자 없이 FILE 리소스를 정의합니다. FILE 리소스는 명명된 소유자가 있는 GFILE 리소스의 구성원입니다. CA Access Control r12.0 이상에서는 명명된 그룹 소유자가 파일에 대한 모든 액세스 권한을 갖습니다. 이전 릴리스에서는 아무도 파일에 대한 액세스 권한을 갖지 않습니다.

클래스

CA Access Control 에서 레코드 *클래스*는 레코드가 가질 수 있는 속성을 정의합니다. 클래스의 모든 레코드는 속성 값은 다르지만 동일한 속성을 가집니다.

다음은 클래스의 예입니다.

- TERMINAL 클래스. tty1, tty 와 같은 터미널 레코드를 포함합니다.
- FILE 클래스. 파일 레코드를 포함합니다.
- PROGRAM 클래스. 프로그램 레코드를 포함합니다.

각 레코드에는 레코드 클래스에 해당하는 속성 값이 포함됩니다. 예를 들어 XUSER 클래스의 레코드에는 엔터프라이즈 사용자의 위치 및 근무 시간과 같은 속성이 포함되고, HOSTNET 클래스의 레코드에는 네트워크 서비스 및 IP 주소 데이터와 같은 속성이 포함됩니다.

CA Access Control 에는 미리 정의된 클래스가 있습니다. 사용자 정의 클래스라는 새 클래스를 정의할 수도 있습니다.

클래스의 기본 레코드

대부분의 클래스에는 자체 데이터베이스에 정의되지 않은 해당 클래스의 리소스에 대한 액세스 유형을 지정하는 기본 레코드(_default)가 포함될 수 있습니다.

다른 리소스 레코드와 마찬가지로 _default 레코드에는 ACL 및 defaces 필드가 포함될 수 있습니다. USER, GROUP, CATEGORY, SECLABEL 및 SEOS 를 제외한 모든 클래스에 대해 _default 레코드를 만들 수 있습니다.

UACC 클래스(폐기됨)

UACC 클래스는 더 이상 권장되지 않습니다. 클래스에서 레코드 기본값을 지정하려면 _default 레코드를 사용합니다.

몇몇 CA Access Control 의 초기 버전에서는 다른 클래스의 _default 레코드와 유사한 레코드에 대해 UACC 라는 별도의 클래스를 사용했습니다. UACC 클래스는 더 이상 권장되지 않으며 _default 레코드를 사용하는 경우 UACC 클래스에서 동일한 레코드를 확인하지 않습니다. 향후 버전에서는 UACC 클래스를 더 이상 지원하지 않을 수도 있습니다.

예를 들어 Henderson 이라는 사용자가 store_log 프로세스 중지를 시도한다고 가정하면 CA Access Control 은 다음 순서에 따라 권한 부여를 검사합니다. 첫 번째 질문은 다음과 같습니다. store_log 프로세스가 데이터베이스에 정의되어 있습니까? CA Access Control 은 데이터베이스를 검색하여 PROCESS 클래스에서 이름이 store_log 인 레코드를 찾습니다.

- 그런 레코드가 없으면 해당 프로세스는 CA Access Control 에 정의되어 있지 않습니다. 이 경우 CA Access Control 은 PROCESS 클래스의 _default 레코드나 UACC 클래스의 PROCESS 레코드를 사용하여 Henderson 이 store_log 를 종료할 수 있는지 여부를 확인합니다.
  - Henderson 이라는 사용자가 _default 레코드의 ACL 에 나타날 경우 ACL 에 지정된 권한이 적용됩니다.
  - Henderson 이 _default 레코드의 ACL 에 나타나지 *않을* 경우 _default 레코드의 defaccess 속성에 지정된 권한이 적용됩니다. 이 권한은 _default ACL 에 나타나지 않는 모든 사용자에게 적용됩니다.
- store_log 프로세스가 데이터베이스에 정의되어 있을 경우 문제는 Henderson 이라는 사용자가 데이터베이스의 store_log 프로세스 ACL 에 나타나는지 여부입니다.
  - 사용자 Henderson 이 store_log 프로세스에 대한 ACL 에 있는 경우 이 목록에 지정되어 있는 권한이 적용됩니다.
  - Henderson 이 ACL 에 나타나지 *않을* 경우 CA Access Control 은 store_log 리소스의 기본 액세스 속성에 지정된 권한을 적용합니다. 이러한 권한을 리소스의 기본 액세스 권한이라고 합니다.

참고: _default 의 기본 액세스 권한(defaccess)을 NONE 으로 설정하거나 _default 를 지정하지 않고 UACC 클래스의 해당 리소스 기본값이 NONE 일 경우, 클래스에 정의되지 않은 리소스에 액세스하려는 모든 접근자는 리소스에 대한 액세스가 거부됩니다.

_default(또는 UACC)의 기본 액세스 권한을 최상위 권한(ALL 또는 경우에 따라 READ 또는 EXECUTE)으로 설정할 경우 모든 사용자는 명시적으로 보호되지 않은 모든 리소스에 액세스할 수 있습니다.

미리 정의된 클래스

미리 정의된 클래스는 다음 유형으로 분류됩니다.

클래스 유형	목적
접근자	사용자 및 그룹과 같은 리소스에 액세스하는 개체를 정의합니다.
정의	보안 레이블 및 범주와 같은 보안 항목을 정의하는 개체를 정의합니다.
설치	CA Access Control 의 동작을 제어하는 객체를 정의합니다.
리소스	액세스 규칙에 의해 보호되는 개체를 정의합니다.

다음 표에서는 미리 정의된 클래스의 전체 목록을 보여 줍니다.

클래스	클래스 유형	설명
ADMIN	정의	ADMIN 특성이 없는 사용자에게 관리 업무를 위임하게 합니다. 이러한 사용자에게 전역 권한 부여 특성을 제공하고 관리 권한 범위를 제한합니다.
AGENT	리소스	CA Access Control 에 적용되지 않습니다.
AGENT_TYPE	리소스	CA Access Control 에 적용되지 않습니다.
APPL	리소스	CA Access Control 에 적용되지 않습니다.
AUTHHOST	접근자	CA Access Control 에 적용되지 않습니다.
CALENDAR	리소스	사용자, 그룹 및 리소스에 적용된 시간 제한에 대한 Unicenter TNG 달력 개체를 정의하게 합니다.
CATEGORY	정의	보안 범주를 정의하게 합니다.

클래스	클래스 유형	설명
CONNECT	리소스	나가는 연결을 보호하게 합니다. 이 클래스의 레코드는 어떤 사용자가 어떤 인터넷 호스트에 액세스할 수 있는지 정의합니다. CONNECT 클래스를 활성화하려면 먼저 스트림 모듈이 활성화되어 있는지 확인하십시오.
CONTAINER	리소스	다른 리소스 클래스의 개체 그룹을 정의하게 하므로 개체의 여러 다른 클래스에 규칙을 적용할 때 액세스 규칙을 간단하게 정의할 수 있습니다.
FILE	리소스	파일, 디렉터리 또는 파일 이름 마스크를 보호하게 합니다.
GAPPL	리소스	CA Access Control 에 적용되지 않습니다.
GAUTHHOST	정의	CA Access Control 에 적용되지 않습니다.
GFILE	리소스	GFILE 클래스의 각 레코드는 파일 또는 디렉터리 그룹을 정의합니다. 그룹화는 사용자를 그룹에 연결하는 것과 같은 방식으로 파일 또는 디렉터리(FILE 클래스의 리소스)를 GFILE 리소스에 명시적으로 연결하여 수행됩니다.
GHOST	리소스	GHOST 클래스의 각 레코드는 호스트 그룹을 정의합니다. 그룹화는 사용자를 그룹에 연결하는 것과 같은 방식으로 호스트(HOST 클래스의 리소스)를 GHOST 리소스에 명시적으로 연결하여 수행됩니다.
GROUP	접근자	이 클래스의 각 레코드는 내부 그룹을 정의합니다.
GSUDO	리소스	이 클래스에 있는 각 레코드는 명령 그룹을 정의합니다. 이 명령은 마치 다른 사용자가 명령을 실행하고 있는 것처럼 실행할 수 있습니다. sesudo 명령은 이 클래스를 사용합니다.
GTERMINAL	리소스	GTERMINAL 클래스의 각 레코드는 터미널 그룹을 정의합니다.
HNODE	정의	HNODE 클래스에는 조직의 CA Access Control 호스트에 대한 정보가 포함되어 있습니다. 클래스의 각 레코드는 엔터프라이즈의 노드를 나타냅니다.
HOLIDAY	정의	HOLIDAY 클래스의 각 레코드는 사용자가 로그인할 때 추가 권한이 필요한 하나 이상의 기간을 정의합니다.

클래스	클래스 유형	설명
HOST	리소스	HOST 클래스의 각 레코드는 호스트를 정의합니다. 호스트는 이름이나 IP 주소로 식별합니다. 개체에는 로컬 호스트가 이 호스트로부터 서비스를 받을 수 있는지 여부를 결정하는 액세스 규칙이 포함됩니다. HOST 클래스를 활성화하려면 먼저 스트림 모듈이 활성화되어 있는지 확인하십시오.
HOSTNET	리소스	HOSTNET 클래스의 각 레코드는 IP 주소 마크스로 식별하고 액세스 규칙이 포함되어 있습니다.
HOSTNP	리소스	이 클래스에 있는 각 레코드는 호스트 그룹을 정의합니다. 이 그룹에 속하는 호스트는 모두 동일한 이름 패턴을 가집니다. 각 HOSTNP 개체의 이름에는 와일드카드가 포함됩니다.
LOGINAPPL	정의	LOGINAPPL 클래스의 각 레코드는 로그인 응용 프로그램을 정의하고, 프로그램을 사용하여 로그인할 수 있는 사용자를 식별하며, 로그인 프로그램이 사용되는 방식을 제어합니다.
MFTERMINAL	정의	MFTERMINAL 클래스의 각 레코드는 메인프레임 CA Access Control 관리 컴퓨터를 정의합니다.
POLICY	리소스	POLICY 클래스의 각 레코드는 정책을 배포하고 제거하는 데 필요한 정보를 정의합니다. 여기에는 정책을 배포하고 제거하는 데 사용되는 selang 명령이 있는 RULESET 개체에 연결하는 링크가 포함되어 있습니다.
PROCESS	리소스	PROCESS 클래스의 각 레코드는 실행 파일을 정의합니다.
PROGRAM	리소스	PROGRAM 클래스의 각 레코드는 조건부 액세스 규칙과 함께 사용할 수 있는 트러스트된 프로그램을 정의합니다. 트러스트된 프로그램은 프로그램이 손상되지 않도록 Watchdog 에서 모니터링하는 setuid/setgid 프로그램입니다.
PWPOLICY	정의	PWPOLICY 클래스의 각 레코드는 암호 정책을 정의합니다.
RESOURCE_DESC	정의	CA Access Control 에 적용되지 않습니다.
RESPONSE_TAB	정의	CA Access Control 에 적용되지 않습니다.
RULESET	리소스	RULESET 클래스의 각 레코드는 정책을 정의하는 규칙 집합을 나타냅니다.
SECFILE	정의	SECFILE 클래스의 각 레코드는 변경해서는 안 되는 파일을 정의합니다.

클래스	클래스 유형	설명
SECLABEL	정의	SECLABEL 클래스의 각 레코드는 보안 레이블을 정의합니다.
SEOS	설치	SEOS 클래스의 한 레코드는 활성 클래스 및 암호 규칙을 지정합니다.
SPECIALPGM	설치	SPECIALPGM 클래스의 각 레코드는 Windows 의 백업, DCM, PBF 및 PBN 기능이나 UNIX 의 xdm, 백업, 메일, DCM, PBF 및 PBN 프로그램을 등록하거나 특수 CA Access Control 권한 보호가 필요한 응용 프로그램을 논리적 사용자 ID 에 연결합니다. 이렇게 하면 수행하는 사람이 아니라 작업의 내용에 따라 액세스 권한을 설정할 수 있습니다.
SUDO	리소스	sesudo 명령에서 사용하는 이 클래스는 root 같은 다른 사용자가 명령을 실행하는 것처럼 일반 사용자 같은 한 사용자가 실행할 수 있는 명령을 정의합니다.
SURROGATE	리소스	이 클래스의 각 레코드에는 접근자를 대리 사용자로 사용할 수 있는 사용자를 정의하는 해당 접근자에 대한 액세스 규칙이 포함되어 있습니다.
TCP	리소스	이 클래스의 각 레코드는 메일, http 또는 ftp 와 같은 TCP/IP 서비스를 정의합니다.
TERMINAL	리소스	TERMINAL 클래스의 각 레코드는 사용자가 로그인할 수 있는 장치인 터미널을 정의합니다.
UACC	리소스	각 리소스 클래스에 대한 기본 액세스 규칙을 정의합니다.
USER	접근자	이 클래스의 각 레코드는 내부 사용자를 정의합니다.
USER_ATTR	정의	CA Access Control 에 적용되지 않습니다.
USER_DIR	리소스	CA Access Control 에 적용되지 않습니다.
XGROUP	리소스	이 클래스의 각 레코드는 CA Access Control 에 대해 엔터프라이즈 그룹을 정의합니다.
XUSER	리소스	이 클래스의 각 레코드는 CA Access Control 에 대해 엔터프라이즈 사용자를 정의합니다.

참고: CA Access Control 데이터베이스 클래스 TCP 및 SURROGATE 는 기본적으로 활성화되지 않습니다.

TCP 클래스가 활성화되어 있지만 어떠한 TCP 레코드도 없고 _default TCP 리소스가 변경되지 않은 이전 릴리스에서 업그레이드하는 경우 CA Access Control 은 업그레이드 중에 이 클래스를 비활성화합니다. 이 사항은 SURROGATE 클래스에 대해서도 동일합니다.

SURROGATE 클래스가 활성화되어 있는 이전 릴리스에서 업그레이드하고, SURROGATE 레코드를 정의했거나 임의의 SURROGATE 레코드의 값을 기본값에서 변경한 경우 CA Access Control 은 업그레이드 후 SURROGATE 클래스 구성을 유지합니다. 클래스와 커널 모드 차단이 활성화된 상태로 유지됩니다.

참고: CA Access Control 클래스에 대한 자세한 내용은 *selang 참조 안내서*를 참조하십시오.

사용자 정의 클래스

CA Access Control 을 통해 새 클래스를 정의하면 추상 개체에 적합한 레코드를 작성하여 추상 개체를 보호할 수 있습니다.

예: 데이터베이스 보기에 대한 사용자 정의 클래스

사이트는 데이터베이스를 사용하여 독점 데이터를 저장 및 표시할 수 있습니다.

사용자 정의 클래스 DATABASE_VIEWS 를 정의하고 각 데이터베이스 보기가 해당 클래스의 리소스 구성원이 되도록 정의할 수 있습니다. 해당 데이터베이스 보기를 작성하는 데 필요한 액세스 권한을 정의하는 ACL 을 리소스에 제공합니다. 사용자가 데이터베이스 보기를 만들려고 하면 CA Access Control 이 사용자의 액세스 권한을 확인하고 ACL 에 따라 작성을 허용하거나 거부합니다.

사용자 정의 클래스 리소스의 와일드카드

사용자 정의 클래스의 리소스 이름에 와일드카드를 사용하면 여러 실제 리소스와 일치하는 리소스 레코드를 작성할 수 있습니다. 와일드카드 패턴과 일치하는 이름을 가진 실제 리소스는 리소스 레코드와 관련된 액세스 권한으로 보호됩니다.

사용할 수 있는 와일드카드는 다음과 같습니다.

- * - 수 제한 없는 모든 문자
- ? - 문자 한 개

실제 리소스 이름이 리소스 레코드 이름 여러 개와 일치할 경우에는 가장 긴 와일드카드가 아닌 일치 항목이 해당 리소스에 사용됩니다.

CA Access Control 은 다음 와일드카드 패턴을 리소스 이름으로 허용하지 *않습니다*.

- *
- /*
- /tmp/*
- /etc/*

사용자 정의 클래스 - 예

시스템에서 은행 서비스를 제공하고 계좌 간 거액 이체를 보호하려고 합니다. 다음 개요를 사용하여 이 보안을 설정할 수 있습니다.

1. 예를 들어 TRANSFERS 라는 이체를 설명하는 레코드가 포함된 클래스를 정의합니다.
2. 보호할 각 이체 금액 수준마다 TRANSFERS 클래스에 레코드를 정의합니다.

   예를 들어 Upto.$1K, Upto.$1M, Upto.$10M 및 Over.$10M 이라는 레코드를 정의할 수 있습니다.

   이체를 제어하는 데 필요한 다른 리소스를 TRANSFERS 클래스의 구성원으로 정의합니다.
3. 서로 다른 사용자에게 서로 다른 최대 이체 권한을 부여하려면 TRANSFERS 클래스의 다양한 레코드에 대한 액세스 권한을 부여하거나 거부합니다.
4. 또한 프로그래밍 방식의 전송을 처리하려면 은행의 송금 프로그램에 CA Access Control API 에 대한 호출을 삽입하여 송금을 허용하기 전에 사용자의 권한을 확인하도록 합니다.

제 5 장: 권한 부여 관리

이 섹션은 다음 항목을 포함하고 있습니다.

액세스 권한

CA Access Control 의 주요 목적은 액세스 권한을 할당 및 적용하는 것입니다.

액세스 권한은 항상 다음 구성 요소로 구성됩니다.

- 액세스 권한이 적용되는 리소스(예: 파일, 호스트 또는 터미널)
- 액세스 유형(예: 읽기, 쓰기, 삭제, 로그인, 실행)
- 접근자(사용자 또는 그룹)

사용자는 다음 중 하나 이상에 해당되기 때문에 특정 방법으로 리소스에 액세스하는 권한을 가집니다.

- 사용자가 리소스 ACL 에서 부여하는 액세스 권한을 가집니다.
- 사용자가 액세스 권한을 가진 그룹의 구성원입니다.
- 사용자가 액세스 권한을 가진 프로그램을 실행하고 있습니다. 예를 들어 사용자가 SPECIALPGM 클래스의 프로그램을 실행하거나 SUDO 클래스의 명령을 실행하는 권한을 가집니다.

참고: 클래스별 액세스 권한에 대한 자세한 내용은 *selang 참조 안내서*를 참조하십시오.

액세스 권한 설정 - 예

예: 내부 사용자에게 읽기 액세스 권한 제공

다음 selang 명령은 터미널 tty30 의 ACL 에 내부 사용자 internal_user 를 추가하여 터미널에 대한 읽기 액세스 권한을 제공합니다.

```
authorize TERMINAL tty30 access(READ) uid(internal_user)
```

예: 엔터프라이즈 사용자에게 읽기 액세스 권한 제공

다음 selang 명령은 터미널 tty30 의 ACL 에 엔터프라이즈 사용자 Terry 를 추가하여 터미널에 대한 읽기 액세스 권한을 제공합니다.

```
authorize TERMINAL tty30 access(READ) xuid(Terry)
```

예: 엔터프라이즈 사용자의 액세스 권한을 리소스로 변경

다음 selang 명령은 Terry 의 터미널 tty30 액세스 권한을 없음으로 설정하므로 Terry 의 액세스를 거부합니다.

```
authorize TERMINAL tty30 access(NONE) xuid(Terry)
```

예: 리소스에서 엔터프라이즈 사용자의 액세스 권한 제거

다음 selang 명령은 터미널 tty30 의 ACL 에서 Terry 를 제거합니다.

```
authorize- TERMINAL tty30 xuid(Terry) access-
```

이제 Terry 는 터미널에 대한 기본 액세스 권한을 가집니다.

예: 엔터프라이즈 사용자에게 하위 관리자 액세스 권한 제공

다음 selang 명령은 엔터프라이즈 사용자 Terry 를 사용자와 파일을 관리하는 권한을 가진 하위 관리자로 설정합니다.

```
authorize ADMIN USER xuid(Terry)
authorize ADMIN FILE xuid(Terry)
```

액세스 제어 목록

리소스 액세스 권한은 액세스 제어 목록에 지정됩니다. 모든 리소스 레코드에는 액세스 제어 목록이 여러 개 있습니다.

ACL

리소스 액세스 권한을 부여받은 접근자와 부여받은 액세스 유형을 함께 지정합니다.

NACL

리소스 권한 부여가 거부된 접근자와 거부된 액세스 유형을 함께 지정합니다.

액세스 권한은 사용자가 로컬에서 로그인했는지 여부와 같은 액세스 관련 상황에 따라 달라질 수도 있습니다.

조건부 액세스 제어 목록

CACL(조건부 액세스 제어 목록)은 ACL 에 대한 확장을 제공합니다. 접근자가 리소스에 액세스하려고 할 때 리소스의 ACL 및 NACL 이 사용자의 액세스 권한을 정의하지 않을 경우 CA Access Control 에서는 조건부 액세스 제어 목록을 검사합니다.

조건부 제어 목록은 지정된 프로그램 사용과 같은 특정 방법으로 액세스하는 리소스에 대한 액세스 권한을 지정합니다.

예를 들어 조건부 액세스 제어 목록을 사용하여 프로그램 경로 지정 규칙을 정의할 수 있습니다.

CA Access Control 에서는 다음과 같은 조건부 액세스 제어 목록을 허용합니다.

- 프로그램 액세스 제어 목록(PACL)
- TCP 클래스 액세스 제어 목록
- CALENDAR 클래스 액세스 제어 목록

조건부 액세스 제어 목록 항목에서 항목을 정의하려면 selang authorize 명령의 via 옵션을 사용할 수 있습니다.

다른 액세스 제어 목록과 같이 조건부 액세스 제어 목록의 각 항목은 리소스 액세스 권한을 부여받은 접근자와 부여받은 액세스 유형을 함께 지정합니다. 또한 조건부 액세스 제어 목록의 항목은 권한 할당에 사용되는 조건을 지정합니다. PACL 의 경우 조건은 접근자가 액세스 권한을 갖기 위해 실행해야 하는 프로그램 이름입니다.

예: PACL 사용

엔터프라이즈 사용자 sysadm1 이 프로그램 secured_su 실행을 통해서만 슈퍼 사용자가 되게 하려면 다음 selang 명령을 사용하여 해당하는 조건부 액세스 규칙을 지정합니다.

```
authorize SURROGATE user.root xuid(sysadm1) via(pgm(secured_su))
```

defaccess - 기본 액세스 필드

리소스 레코드에는 기본 액세스 필드인 defaccess 가 포함될 수 있습니다. defaccess 필드 값은 리소스 액세스 제어 목록이 적용되지 않는 접근자에게 허용되는 액세스 권한을 지정합니다.

리소스 액세스 권한을 확인하는 방법

접근자가 리소스에 액세스하려고 하면 CA Access Control 에서는 결과를 얻을 때까지 미리 결정된 순서에 따라 하나 이상의 검사를 실행하여 액세스 권한을 확인합니다. 검사에서 액세스 결과(액세스 거부 또는 허용)가 생성되면 CA Access Control 에서는 추가로 검사하지 않고 결과를 반환합니다.

이러한 검사를 실행하는 순서가 중요합니다. 각 리소스에 대해 CA Access Control 에서는 기본적으로 다음 순서에 따라 액세스 레코드를 검사합니다.

1. 리소스의 시간 기반 제한 사항
2. 리소스 소유권(소유자에게 액세스가 허용됨)
3. B1 검사
4. 리소스 NACL
5. 리소스 ACL

6. 리소스 PACL
7. 리소스 defaccess 필드

마지막 검사 두 개의 순서는 accpacl 옵션에 의해 결정됩니다. selang 명령 setoptions setpacl-를 사용하여 리소스 PACL 사용을 비활성화할 수 있습니다.

하나의 액세스 제어 목록에는 사용자에게 영향을 미치는 항목이 여러 개 포함될 수 있습니다. 예를 들어 사용자를 명시적으로 언급하는 한 항목과 사용자가 속한 각 그룹에 대한 여러 항목을 포함할 수 있습니다. CA Access Control 에서는 다음 수준으로 이동하기 전에 각 수준에서 가능한 항목을 모두 확인합니다. 각 수준에서 충돌하는 규칙을 확인하는 방법에 대한 자세한 내용은 사용자 및 그룹 액세스 권한 간 상호 작용 (페이지 64)을 참조하십시오.

예: 파일에 대한 결과 사용 권한

다음 표에서는 user1 이라는 접근자가 리소스 file1 을 읽으려고 한다고 가정합니다.

다음 표에서 CA Access Control 은 PACL 을 사용하는 accpacl 옵션의 기본 설정을 따르고 있습니다.

user1 에 대한 NACL 의 항목	user1 에 대한 ACL 의 항목	user1 에 대한 PACL 의 항목	defaccess 의 항목	결과 사용 권한
읽기	*(모두)*	*(모두)*	*(모두)*	읽기 거부됨
(정의되지 않음)	없음	*(모두)*	*(모두)*	읽기 거부됨
(정의되지 않음)	읽기	*(모두)*	*(모두)*	읽기 허용됨
(정의되지 않음)	*(정의되지 않음)*	via pgm securereader	*(모두)*	securereader 프로그램을 통해 읽기 허용됨
(정의되지 않음)	*(정의되지 않음)*	*(정의되지 않음)*	읽기	읽기 허용됨

항목이 *(정의되지 않음)*으로 표시되면 해당 액세스 제어 목록에 user1 항목이 없음을 의미합니다.

항목이 *(모두)*로 표시되면 CA Access Control 에서 액세스 제어 목록을 확인하지 않으므로 해당 액세스 제어 목록의 항목이 문제가 없음을 의미합니다.

CA Access Control 에서 확인하는 순서는 왼쪽에서 오른쪽입니다. 모든 행의 경우 거부된 액세스 권한이 포함된 셀의 오른쪽에 있는 셀 값이 *(모두)*임을 알 수 있습니다. 반대로 거부된 액세스 권한이 포함된 셀의 왼쪽에 있는 모든 셀 값은 *(정의되지 않음)*입니다.

사용자 및 그룹 액세스 권한 간 상호 작용

액세스 권한을 사용자 및 사용자가 속한 그룹에 명시적으로 부여하거나 거부할 수 있습니다. 경우에 따라 이러한 권한이 충돌할 수 있습니다. 다음 예에서는 사용자가 그룹 두 개(Group 1 및 Group 2)의 구성원일 때 충돌하는 액세스 권한이 동일한 리소스에 할당될 경우 발생하는 결과를 보여 줍니다.

이 예에서는 누적된 그룹 권한 (페이지 65) 옵션이 설정되었다고 가정합니다(기본 설정).

사용자에 대한 액세스 권한	Group 1 에 대한 액세스 권한	Group 2 에 대한 액세스 권한	결과 액세스 권한
거부된 액세스	*(모두)*	*(모두)*	거부된 액세스
액세스 허용됨	*(모두)*	*(모두)*	액세스 허용됨
(정의되지 않음)	액세스 허용됨	*(정의되지 않음)*	액세스 허용됨
(정의되지 않음)	*(정의되지 않음)*	액세스 허용됨	액세스 허용됨
(정의되지 않음)	액세스 허용됨	액세스 허용됨	액세스 허용됨
(정의되지 않음)	거부된 액세스	*(모두)*	거부된 액세스
(정의되지 않음)	*(모두)*	거부된 액세스	거부된 액세스

항목이 *(정의되지 않음)*으로 표시되면 거부된 사용자 또는 그룹 항목이 없음을 의미합니다.

항목이 *(모두)*로 표시되면 CA Access Control 에서 액세스 권한을 확인하지 않으므로 해당 액세스 권한이 문제가 없음을 의미합니다.

ACCGRR(누적된 그룹 권한)

ACCGRR(*누적된 그룹 권한*) 옵션은 CA Access Control 이 리소스 ACL 을 검사하는 방법에 영향을 미칩니다. ACCGRR 이 활성화되면 CA Access Control 은 사용자가 속한 모든 그룹에서 부여된 권한의 ACL 을 확인합니다. ACCGRR 이 비활성화되면 CA Access Control 은 ACL 에서 적용 가능한 항목에 값 none 이 포함되어 있는지 확인합니다. 값 none 이 포함되어 있으면 액세스가 거부됩니다. 그렇지 않으면 CA Access Control 은 액세스 제어 목록의 첫 번째 적용 가능한 항목을 제외하고 모든 그룹 항목을 무시합니다. 기본적으로 이 옵션은 활성화됩니다.

ACCGRR 옵션을 활성화하려면 다음 selang 명령을 사용합니다.

```
setoptions accgrr
```

ACCGRR 옵션을 비활성화하려면 다음 selang 명령을 사용합니다.

```
setoptions accgrr-
```

보안 수준, 범주 및 레이블

보안 수준과 보안 범주는 리소스 액세스 권한을 제한하여 액세스 제어 목록 사용을 보완하는 방법을 추가로 제공합니다.

보안 레이블은 보안 수준과 보안 범주를 더욱 쉽게 관리할 수 있도록 함께 번들로 제공하는 수단입니다.

보안 수준

*보안 수준*은 접근자와 리소스에 할당할 수 있는 0 에서 255 사이의 정수입니다. 리소스의 액세스 제어 목록에서 사용자에게 액세스 권한이 부여된 경우에도 접근자의 보안 수준이 리소스에 할당된 보안 수준보다 낮으면 접근자는 리소스에 액세스할 수 없습니다. 리소스에 0 보안 수준이 있으면 보안 수준 검사가 해당 리소스를 확인하지 않습니다.

보안 수준이 0 인 접근자는 0 이 아닌 보안 수준을 가진 리소스에 액세스할 수 없습니다.

보안 범주

*보안 범주*는 CATEGORY 클래스의 레코드 이름입니다. 접근자와 리소스에 보안 범주를 할당할 수 있습니다. 접근자는 리소스에 할당된 모든 보안 범주에 할당된 경우에만 리소스에 액세스할 수 있습니다.

보안 레이블

*보안 레이블*은 SECLABEL 클래스의 레코드 이름입니다. 보안 레이블은 보안 수준과 일련의 보안 범주를 함께 번들로 제공합니다. 접근자나 리소스에 보안 레이블을 할당하면 보안 레이블과 관련된 결합 보안 수준 및 보안 범주가 접근자나 리소스에 제공됩니다. 보안 레이블은 접근자나 리소스의 특정 보안 수준 및 범주 할당보다 우선합니다.

예: 보안 레이블 High_Security 사용

보안 수준 255 와 보안 범주 MANAGEMENT 및 CONFIDENTIAL 이 포함된 보안 레이블이 High_Security 라고 가정합니다.

사용자 user1 에 보안 레이블 High_Security 를 할당하면 user1 의 보안 수준은 255 가 되고 보안 범주는 MANAGEMENT 및 CONFIDENTIAL 이 됩니다.

제 6 장: 계정 보호

이 섹션은 다음 항목을 포함하고 있습니다.

계정을 보호하는 이유

사용자 계정이 주로 암호 공격의 대상이 됩니다. 루트 계정 보호에는 슈퍼 사용자 권한의 문제점을 해결하는 SUDO(Surrogate DO) 기능 사용 및 대체 사용자(su) 요청 모니터가 포함됩니다. CA Access Control 은 serevu(사용자 데몬 해지) 및 PAM(Pluggable Authentication Module)의 2 수준 암호 보호 시스템을 제공합니다. 또한 사용자가 작업하지 않는 일정 기간 후의 자동 잠금을 지정하여 계정을 보호할 수 있습니다.

안전한 사용자 대체

UNIX su 명령은 사용자가 대상 사용자의 암호를 사용하여 다른 사용자로 대체하게 합니다. 사용자 ID 를 대체하려는 사용자는 대상 사용자의 암호를 기억하거나, 기록해 놓거나, 대상 사용자에게 평범한 암호를 사용하도록 요청해야 합니다. 이것은 여러 암호 정책에 위반되며 su 명령은 명령을 실행한 사용자를 기록하지 않으므로 계정 소유자를 가장하는 사용자와 실제 소유자를 구분할 수도 없습니다.

CA Access Control 은 UNIX su 명령의 향상된 버전인 sesu 유틸리티를 포함하고 있습니다. 인증을 위해 대상 사용자의 암호가 아닌 현재 사용자의 암호를 입력하도록 sesu 를 구성할 수 있습니다. 권한 부여 프로세스는 SURROGATE 클래스에 정의된 액세스 규칙 및 (선택적으로) 명령을 실행하는 사용자의 암호에 기반합니다.

su 와 달리 sesu 는 대상 사용자의 암호를 몰라도 해당 사용자로 전환할 수 있습니다. 전환 가능 여부는 데이터베이스에 지정된 권한에 따라 결정되며 사용자는 자신의 로그인 ID 가 기록되기 때문에 이후 작업에 책임을 집니다.

사용자가 _surrogate 그룹 사용자 중 하나로 전환하는 경우 CA Access Control 은 새 사용자로 수행한 동작을 모두 추적하여 감사 기록에 전송합니다.

부주의한 사용을 방지하기 위해 파일 시스템에서 sesu 는 누구도 실행할 수 없도록 설정되어 있습니다. 이 프로그램을 사용하려면 보안 관리자가 프로그램을 실행할 수 있도록 설정하고 setuid 를 root 로 설정해야 합니다.

중요! sesu 유틸리티를 사용하기 전에 모든 사용자를 CA Access Control 데이터베이스에 정의하고 sesu 사전 요구 사항을 설정하십시오. 이렇게 하면 CA Access Control 에 정의되지 않은 사용자에게 전체 시스템이 개방되는 사고를 방지할 수 있습니다.

사용자 ID 대체 규칙 설정

사용자가 다른 사용자를 대체하지 못하게 하거나 대체하게 하려면 사용자 ID 대체 규칙을 설정해야 합니다. 이러한 규칙은 SURROGATE 클래스 리소스를 통해 관리됩니다. 사용자 대체 규칙을 정의하려면 SURROGATE 레코드를 만들어야 합니다.

사용자 ID 대체 규칙을 설정하려면

1. CA Access Control 끝점 관리에서 "사용자" 탭을 클릭한 다음 "권한 부여 및 위임" 하위 탭을 클릭합니다.

   "권한 부여 및 위임" 메뉴 옵션이 왼쪽에 나타납니다.

2. "사용자 ID 대체"를 클릭합니다.

   "사용자 ID 대체" 페이지가 나타납니다.

3. "Create User ID Substitution(사용자 ID 대체 만들기)"을 클릭합니다.

   "Create User ID Substitution(사용자 ID 대체 만들기)" 페이지가 나타납니다.

4. 탭 구분 페이지에서 필드를 완료하고 "저장"을 클릭합니다.

   참고: SURROGATE 클래스 속성에 대한 자세한 내용은 *selang 참조 안내서*를 참조하십시오.

사용자 대체를 위해 sesu 를 설정하는 방법

기본적으로 sesu 유틸리티는 아무도 실행할 수 없도록 파일 시스템에 설정되어 있습니다. 사용자들이 sesu 를 사용하도록 허용하기 전에 안전한 사용을 위해 먼저 데이터베이스 규칙을 설정해야 합니다. 그런 다음 사용자들이 반드시 CA Access Control sesu 유틸리티를 대신 사용하도록 시스템 su 유틸리티를 잠가야 합니다.

sesu 를 설정하려면 다음과 같이 하십시오.

1. 기본 사용자 대체 규칙을 설정합니다 (페이지 69).
2. 시스템 su 유틸리티를 CA Access Control sesu 유틸리티로 바꿉니다 (페이지 69).
3. 사용자가 시스템 su 유틸리티를 실행할 수 없도록 합니다 (페이지 72).

참고: 이 설정을 완료하면 CA Access Control 이 실행되는 동안 시스템 su 유틸리티는 실행되지 않으며, 사용자들은 반드시 보안 sesu 유틸리티를 사용해야만 합니다. CA Access Control 이 실행 중이지 않을 때는 시스템 su 유틸리티를 사용할 수 있습니다.

기본 사용자 전환 규칙 설정

sesu 유틸리티를 사용하기 전에 데이터베이스에 몇 가지 공통적인 사용자 전환 규칙을 설정해야 합니다. 이러한 규칙은 알 수 없는 사용자가 권한 있는 사용자 계정으로 전환하지 못하도록 방지하지만 특정 사용자 및 프로세스가 필요한 사용자 전환 작업을 수행할 수 있도록 허용합니다.

기본 사용자 전환 규칙을 설정하려면

1. 다음 특성을 사용하여 root 사용자(USER.root)에 대한 대리 리소스를 만듭니다.
   - 소유자로서 *nobody*
   - 기본 액세스 *none*
   - 모든 관리자는 모든 권한을 가져야 함

   이렇게 하면 명시적으로 권한 부여되지 않는 한 모든 사용자가 root 를 대체하지 못합니다. 모든 관리자는 root 를 대체하도록 명시적으로 권한 부여됩니다.

   참고: 개별 관리자를 개별적으로 권한 부여하거나 관리자 그룹을 사용하여 모든 관리자를 권한 부여할 수 있습니다.

2. 다음 특성을 사용하여 root 그룹(GROUP.other)에 대한 대리 리소스를 만듭니다.
   - 소유자로서 *nobody*
   - 기본 액세스 *none*
   - 모든 관리자는 모든 권한을 가져야 함

   이렇게 하면 명시적으로 권한 부여되지 않는 한 모든 사용자가 root 그룹을 대체하지 못합니다. 모든 관리자는 root 그룹을 대체하도록 명시적으로 권한 부여됩니다.

   참고: 대부분의 UNIX 시스템에서 root 그룹은 *other* 또는 *sys* 입니다.
3. 다음과 같이 USER._default 에 대한 사용자 대체 규칙을 변경합니다.
   - 소유자로서 *nobody*
   - 기본 액세스 *none*
   - 정의되지 않은 사용자로 대체하도록 root 에 권한 부여
   - 정의되지 않은 사용자로 대체하도록 관리자 그룹에 권한 부여

   이렇게 하면 명시적으로 권한 부여되지 않는 한 모든 사용자가 그룹을 대체하지 못하고 명시적으로 거부되지 않는 한 사용자를 대체하도록 root 및 root 그룹에 권한 부여됩니다.

   참고: dtlogin 과 같은 프로그램이 세션 소유권을 root(uid=0, 기본 X Window 소유자)에서 다른 사용자로 전환하도록 허용하려면 명시적으로 root 에 권한을 부여해야 합니다. 이렇게 하지 않으면 CA Access Control 이 명시적으로 허용되지 않은 모든 사용자 전환 작업을 차단하므로 로그인 시도가 실패합니다.
4. 다음과 같이 GROUP._default 에 대한 그룹 대체 규칙을 변경합니다.
   - 소유자로서 *nobody*
   - 기본 액세스 *none*
   - 정의되지 않은 그룹을 대체하도록 root 에 권한 부여
   - 정의되지 않은 그룹으로 대체하도록 관리자 그룹에 권한 부여

   이렇게 하면 명시적으로 권한 부여되지 않는 한 모든 사용자가 그룹을 대체하지 못하고 명시적으로 거부되지 않는 한 그룹을 대체하도록 root 및 root 그룹에 권한 부여됩니다.

예: selang 으로 기본 사용자 대체 규칙 설정

다음 selang 명령을 사용하여 사용자 환경에서 기존 사용자 대체 규칙을 설정합니다.

```
nr surrogate USER.root defacc(n) own(nobody)
auth surrogate USER.root gid(sys_admin_GID) acc(a)
nr surrogate GROUP.other defacc(n) own(nobody)
auth surrogate GROUP.other gid(sys_admin_GID) acc(a)
cr surrogate USER._default defacc(n) own(nobody)
cr surrogate GROUP._default defacc(n) own(nobody)
auth surrogate USER._default uid(root) acc(a)
auth surrogate GROUP._default uid(root) acc(a)
auth surrogate USER._default gid(sys_admin_GID) acc(a)
auth surrogate GROUP._default gid(sys_admin_GID) acc(a)
```

시스템 su 유틸리티를 CA Access Control sesu 유틸리티로 대체

기본적으로 sesu 유틸리티는 아무도 실행할 수 없도록 파일 시스템에 설정되어 있습니다. 사용자들이 sesu 유틸리티를 사용하여 다른 사용자로 전환할 수 있도록 하려면 우선 sesu 를 활성화한 다음 시스템 su 를 이 유틸리티로 대체해야 합니다.

시스템 su 유틸리티를 CA Access Control sesu 유틸리티로 대체하려면

참고: 다음 단계를 실행하려면 root 또는 다른 권한 있는 사용자여야 합니다.

1. 사용자들이 다음 명령을 사용하여 sesu 유틸리티를 실행하도록 허용합니다.

   ```
   chmod +s /opt/CA/AccessControl//bin/sesu
   ```

2. 다음 명령을 사용하여 시스템 su 유틸리티를 찾습니다.

   ```
   which su
   ```

3. 다음 명령을 사용하여 시스템 su 유틸리티의 이름을 변경합니다.

   ```
   mv su_dir/su su_dir/su.ORIG
   ```

   여기서 *su_dir* 은 su 가 있는 디렉터리입니다.

4. sesu 유틸리티를 su 명령에 연결합니다.

   ```
   ln -s /opt/CA/AccessControl//bin/sesu su_dir/su
   ```

   이렇게 하면 sesu 유틸리티가 실행되는 동안에도 사용자들이 su 명령을 계속 사용할 수 있습니다.

5. 다음 명령을 사용하여 CA Access Control 을 중지합니다.

   secons -s

6. 다음 명령을 사용하여 CA Access Control 구성 설정을 수정합니다.

   seini -s sesu.SystemSu *su_dir*/su.ORIG
   seini -s sesu.UseInvokerPassword yes

   토큰 SystemSu 는 CA Access Control 이 실행 중이지 않을 때 sesu 가 원래 시스템 su 유틸리티를 호출할 수 있도록 설정되었습니다.

   토큰 UseInvokerPassword 는 CA Access Control 이 root 암호나 다른 사용자의 암호 대신 사용자의 원래 암호를 묻도록 설정되었습니다. 사용자 전환이 허용되기 전에 사용자는 다시 인증되어야 합니다.

7. 다음 명령을 사용하여 CA Access Control 을 다시 로드합니다.

   seload

사용자가 시스템 su 유틸리티를 실행할 수 없도록 설정

sesu 유틸리티를 구성해도 root 또는 사용자의 암호로 su.ORIG(시스템 su 유틸리티의 변경된 이름)를 예전처럼 실행할 수 있습니다. 이것을 방지하려면 PROGRAM 클래스를 사용하여 CA Access Control 이 실행 중일 때 su.ORIG 가 실행되지 않도록 명시적으로 지정하십시오.

참고: CA Access Control 설치 및 구성 중 seuidpgm 을 사용한 경우 이 절차를 수행할 필요가 없습니다. su 는 su.ORIG 로 이름이 변경되었으므로 실행되지 않습니다.

사용자가 시스템 su 유틸리티를 실행할 수 없도록 하려면

1. selang 에서 다음 명령을 사용하여 CA Access Control 이 이름이 변경된 su 유틸리티를 모니터링하도록 설정합니다.

   nr program *su_dir*/su.ORIG defacc(x) own(nobody)

2. root 로 로그인한 후 다음 명령을 사용하여 파일 액세스 및 수정 시간을 변경합니다.

   touch *su_dir*/su.ORIG

   CA Access Control 은 su.ORIG 를 모니터링하지만 파일이 *수정*되었으므로 실행하지 않습니다.

Surrogate DO 기능 설정

작업자, 생산 직원 및 최종 사용자가 슈퍼 사용자만 수행할 수 있는 작업을 수행하는 경우가 있습니다. 다음과 같은 작업이 여기에 포함됩니다.

- CD-ROM 마운트
- 백업 스크립트 사용
- 프린터 설정

기존의 솔루션은 이러한 모든 사용자에게 슈퍼 사용자 암호를 제공하는 것이었지만 이 경우 사이트의 보안이 위협을 받게 됩니다. 암호의 보안을 유지하는 보안 방법을 사용하면 사용자의 일상적인 작업 수행 관련요청으로 인해 시스템 관리자의 업무량이 늘어납니다.

Surrogate DO(sesudo) 유틸리티로 이 문제를 해결할 수 있습니다. 이 유틸리티로 사용자는 SUDO 클래스에 정의된 동작을 수행할 수 있습니다. SUDO 클래스의 각 레코드에는 스크립트가 포함되고 스크립트를 실행할 수 있는 사용자와 그룹이 지정되며 목적에 따라 필요한 권한이 부여됩니다.

예를 들어, 사용자가 root 인 것처럼 CD-ROM 을 마운트하는 SUDO 리소스를 정의하려면 다음 명령을 입력하십시오.

```
newres SUDO MountCd data('mount /usr/dev/cdrom /cdr') targuid(root)
```

newres 명령은 MountCd 를 일부 사용자가 해당 root 권한을 가질 수 있는 보호된 작업으로 정의합니다. 이 예제에서는 root 가 대상 사용자 ID(사용자의 권한을 빌림)임을 보여주는 targuid(root) 매개 변수를 사용합니다. 실제로 이 예에서는 루트가 SUDO 레코드의 기본 대상이기 때문에 매개 변수가 필요하지 않습니다.

중요! 데이터 속성에서는 전체 절대 경로 이름을 사용하십시오. 상대 경로 이름을 사용하면 보호되지 않은 디렉터리에 있는 트로이 목마 프로그램이 실수로 실행될 수 있습니다.

또한 사용자는 authorize 명령을 사용하여 MountCd 작업을 수행할 수 있습니다. 예를 들어 사용자 *operator1* 이 CD-ROM 을 마운트할 수 있도록 하려면 다음 명령을 입력합니다.

```
authorize SUDO MountCd uid(operator1)
```

또한 authorize 명령을 사용하여 사용자가 보호된 작업을 수행할 수 없도록 할 수 있습니다. 예를 들어 사용자 *operator2* 가 CD-ROM 을 마운트할 수 없도록 하려면 다음 명령을 입력합니다.

```
authorize SUDO MountCd uid(operator2) access(None)
```

sesudo 유틸리티를 실행하면 보호된 작업이 수행됩니다. 예를 들어, *operator1* 사용자는 다음 명령을 사용하여 CD-ROM 을 마운트합니다.

```
sesudo MountCd
```

sesudo 유틸리티는 처음에는 사용자가 SUDO 작업을 수행할 수 있는 권한이 있는지 여부를 확인한 다음 사용자가 리소스에 대한 권한이 있는 경우 리소스에 정의되어 있는 명령 스크립트를 실행합니다. 이 예제의 경우 sesudo 는 operator1 이 MountCd 작업을 수행할 수 있는지 확인한 후 mount 명령인 /usr/dev/cdrom /cdr 을 호출합니다.

실행하기 전에 사용자 암호를 요청하도록 sesudo 를 지정하려면 PASSWORD 매개 변수를 포함하는 명령을 사용하여 SUDO 레코드를 정의하거나 수정하십시오. 이 매개 변수를 사용하지 않는 경우 사용자가 명령을 수행할 수 있는 기능은 SUDO 개체에 대한 액세스 규칙을 기반으로만 지정됩니다.

참고: sesudo 유틸리티와 SUDO 레코드 관리(editres 명령)에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

SUDO 레코드 정의

SUDO 클래스의 레코드에는 사용자가 빌린 권한으로 스크립트를 실행할 수 있는 명령 스크립트가 저장되어 있습니다. 권한을 빌려올 수 있는 자격은 스크립트를 실행하는 sesudo 명령과 SUDO 레코드가 엄격하게 제어합니다.

SUDO 레코드에서 comment 속성은 특수한 용도로 사용되며 흔히 data 속성이라는 다른 이름으로 알려져 있습니다.

데이터 속성의 값은 명령 스크립트이며, 금지하거나 허용해야 하는 스크립트 매개 변수 값을 하나 이상 선택적으로 추가할 수 있습니다. 전체 data 속성 값은 작은 따옴표로 묶어야 하고 트로이 목마가 실행되지 않도록 실행 파일은 전체 경로 이름으로 참조되어야 합니다.

다음은 data 속성의 형식입니다.

```
data('cmd[;[prohibited-values][;permitted-values]]')
```

금지된 값과 허용된 값 목록이 선택 사항이므로 간단한 data 속성 값은 다음과 같을 수 있습니다.

```
newres SUDO MountCd data('mount /dev/cdrom /cdr')
```

간단한 이 값은 sesudo MountCd 명령이 mount /dev/cdrom /cdr 스크립트를 실행한다는 것을 나타냅니다. 금지되는 특정 스크립트 매개 변수 값은 없으며, 모두 허용됩니다.

와일드카드와 적합한 변수를 사용하면 금지된 매개 변수와 허용된 매개 변수를 다양하게 지정할 수 있습니다. 사용할 수 있는 와일드카드는 표준 UNIX 와일드카드입니다. 변수는 다음과 같습니다.

변수	설명
$A	영문자 값
$G	기존 CA Access Control 그룹 이름
$H	사용자의 홈 경로 패턴
$N	숫자 값
$O	실행자의 사용자 이름
$U	기존 CA Access Control 사용자 이름
$e	매개 변수 없는 SUDO 명령
$f	기존 파일 이름
$g	기존 UNIX 그룹 이름
$h	기존 호스트 이름
$r	UNIX 읽기 권한을 가진 기존 UNIX 파일 이름
$u	기존 UNIX 사용자 이름
$w	UNIX 쓰기 권한을 가진 기존 UNIX 파일 이름
$x	UNIX 실행 권한을 가진 기존 UNIX 파일 이름

금지된 매개 변수 값의 목록을 스크립트에 추가하는 경우:

- 금지된 매개 변수 값과 스크립트를 세미콜론으로 구분하지만, 앞뒤에 작은 따옴표를 입력해야 합니다. 예를 들어, 사용자가 -9 를 사용할 수 없지만 다른 모든 매개 변수는 사용할 수 있게 하려면 다음 명령을 입력합니다.

  newres SUDO *scriptname* data('*cmd*;-9')

  여기서 *cmd* 는 스크립트를 나타냅니다.

  또한 매개 변수 값을 허용하지 않으면서 모든 매개 변수를 기본값으로 사용하려면 다음과 같이 SUDO 레코드를 정의하십시오.

  newres SUDO *scriptname* data('*cmd*;*')

- script 매개 변수에 둘 이상의 금지된 값이 있는 경우, 공백을 구분자로 사용하십시오. 예를 들어, 사용자가 -9 및 -HUP 이외의 모든 매개 변수를 사용할 수 있도록 지정하려면 다음 명령을 입력합니다.

  newres SUDO *scriptname* data('*cmd*;-9 -HUP')

- 둘 이상의 script 매개 변수에 금지된 값이 있는 경우, 파이프 문자(|)를 금지된 값 집합 사이의 구분자로 사용하십시오. 예를 들어, 사용자가 스크립트의 첫 번째 매개 변수로 -9 와 -HUP 를 사용하지 못하게 하고 두 번째 매개 변수로 기존 UNIX 사용자 이름을 사용하지 못하게 하려면 다음 명령을 입력하십시오.

  newres SUDO *scriptname* data('*cmd*;-9 -HUP | $u')

  스크립트에 나열한 매개 변수보다 많은 수의 매개 변수가 있는 경우 금지된 매개 변수의 마지막 집합은 나머지 모든 매개 변수에 적용됩니다.

허용된 매개 변수 값의 목록을 스크립트에 추가하는 경우:

- sesudo 유틸리티는 두 가지 검사를 수행합니다. 즉, 매개 변수 값은 금지된 해당 값과 일치하면 안 될 뿐만 아니라 허용된 해당 값 중 하나 이상과 일치해야 합니다.
- *허용된* 값의 목록과 *금지된* 값의 목록을 세미콜론으로 구분하지만, 두 개의 목록 앞뒤에는 각각 작은 따옴표를 입력해야 합니다. 금지된 값 목록이 없을 경우에도 세미콜론을 사용해야 하며 그렇지 않으면 허용하려는 값이 금지됩니다. 예를 들어 스크립트에 대한 매개 변수 값으로 NAME 값만 허용하려면 다음 명령을 입력하십시오.

  newres SUDO scriptname data('cmd;;NAME')

- 다른 목록에서도 다음 작업을 수행합니다.
  - script 매개 변수에 둘 이상의 허용된 값이 있는 경우, 공백을 구분자로 사용하십시오.
  - 둘 이상의 script 매개 변수에 허용된 값이 있는 경우, 파이프 문자(|)를 허용된 값 집합 사이의 구분자로 사용하십시오.

  예를 들어, 두 개의 매개 변수가 있는데 첫 번째 매개 변수는 숫자여야 하지만 UNIX 사용자 이름이 아니어야 하고 두 번째 매개 변수는 영문자여야 하지만 UNIX 그룹 이름이 아니어야 한다면 다음 명령을 입력합니다.

  newres SUDO *scriptname* data('*cmd*; $u | $g ; $N | $A')

  스크립트에 나열한 매개 변수보다 많은 수의 매개 변수가 있는 경우 허용된 매개 변수의 마지막 집합은 나머지 모든 매개 변수에 적용됩니다.

따라서 data 속성의 전체 형식은 먼저 스크립트가 온 다음 금지된 값이 매개 변수별로 오고 마지막으로 허용된 값이 매개 변수별로 옵니다.

```
data('cmd;
param1_prohib1 param1_prohib2 ... param1_ prohibN | \
param2_prohib1 param2_prohib2 ... param2_ prohibN | \
...
paramN_prohib1 paramN_prohib2 ... paramN_prohibN ; \
param1_permit1 param1_permit2 ... param1_ permitN | \
param2_permit1 param2_permit2 ... param2_ permitN |
...
paramN_permit1 paramN_permit2 ... paramN_permitN')
```

암호 공격 방지

권한이 부여되지 않은 가장 일반적인 액세스 유형은 암호를 추측하는 해커에 의한 액세스입니다. CA Access Control 에서는 암호 공격을 감지하고 방지하는 serevu 및 pam_seos 도구를 제공합니다.

암호 공격을 방지하는다른 방법은 암호 정책 규칙을 설정하여 사용자 환경에서 사용되는 암호를 제어하는 방법입니다.

serevu

serevu 데몬은 지정한 횟수 보다 많은 로그인을 시도한 사용자의 계정을 잠급니다. 이렇게 하면 추가 계정 입력 시도를 거부하여 잠재적인 암호 공격은 물론 "사전 공격"도 방지할 수 있습니다.

일반적으로 사용자 잠금 유틸리티를 사용할 때 시스템이 서비스 거부 공격에 노출될 수 있습니다. 하나의 공통적인 서비스 공격 유형은 시스템 관리자 계정으로 침투하려는 시도입니다. 몇 번의 시도 후에 시스템 관리자 계정이 해지되고 시스템 관리자는 더 이상 로그인할 수 없습니다. 유사한 공격이 모든 중요한 사용자 계정에 대해 수행될 경우 시스템은 더 이상 사용할 수 없게 되고 복구할 수도 없습니다. 이런 문제를 방지하기 위해 serevu 데몬은 다음과 같은 두 가지 모드의 작업을 제공합니다.

- 계정이 자동 복원된 후 일정 시간 동안 해지됩니다.
- 계정이 영구적으로 해지됩니다.

serevu 는 root 를 해지하지 않으므로 시스템이 잠기지 않습니다.

참고: serevu 데몬에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

참고: 성공적으로 root 에 대한 사전 공격을 방지하려면 root 사용자의 암호와 관련하여 특별한 주의를 기울여야 합니다.

pam_seos

pam_seos 는 CA Access Control 이 고급 계정 관리 기능에 사용하는 PAM(Pluggable Authentication Module)입니다. CA Access Control 은 로그인 프로그램의 로그인 절차 중에 pam_seos 를 호출합니다. 모듈이란 필수 기능을 필요할 때마다 제공하는, 동적으로 로드될 수 있는 공유 객체입니다.

다음 세 가지 작업을 수행하도록 pam_seos 를 구성할 수 있습니다.

- 로그인 오류 감지

  계정 관리 구성 요소가 실패한 모든 로그인 시도를 감지하고 감사 파일 및 실패한 특수 로그인 파일에 기록합니다. 이 모듈은 CA Access Control 이 액세스를 거부하지 않는 경우 UNIX 실패를 감지합니다.

  CA Access Control 은 실패한 로그인 시도 횟수를 특수 파일에 기록합니다. serevu 유틸리티는 이 파일을 읽고 정보를 사용하여 사용자 액세스를 취소할 경우와 시기를 확인합니다.

- 디버그 모드 제공

  CA Access Control 이 로그인을 거부할 때 일반적으로 로그인 세션 도중에는 거부 이유를 표시하지 않습니다. pam_seos 모듈의 디버그 모드가 설정되면 CA Access Control 은 로그인 거부 이유에 대한 간단한 설명을 표시합니다. 예를 들어 "유예 로그인"은 사용자에게 남은 로그인 시도 횟수가 없다는 의미입니다.

- 만료된 암호 및 유예 로그인 확인

  암호 관리 구성 요소는 사용자의 암호 만료 및 유예 로그인 수를 확인하는 segrace 유틸리티를 호출합니다. 사용자 암호가 만료되고 남은 유예 로그인 횟수가 없는 경우 segrace 는 사용자에게 암호 변경을 허용하는 sepass 유틸리티를 호출합니다.

참고: 암호 변경이 필요한 경우에만 CA Access Control 이 segrace 를 호출합니다.

참고: SSH 에서 실패한 로그인 이벤트를 가져오려면 사용하는 SSH 버전이 PAM 을 지원하도록 컴파일 및 구성되어야 합니다. 사용하는 SSH 버전이 PAM 을 사용하지 않는 경우 사용자가 실패한 로그인 규칙을 위반했는지 여부를 CA Access Control 가 파악할 수 없습니다.

설치 프로그램은 적합한 명령행을 pam.conf 구성 파일에 추가하고 기존 구성 파일을 /etc/pam.conf.bak 으로 저장합니다.

pam_seos 모듈은 seos.ini 파일을 통해 구성됩니다. 필요한 기능에 따라 [pam_seos] 섹션에 있는 다음 토큰을 설정하십시오.

암호 만료 및 유예 로그인 확인 기능을 사용하려면 seos.ini 파일에서 다음 토큰을 설정하십시오.

```
call_segrace = Yes
```

로그인 디버그 모드를 사용하려면 seos.ini 파일에서 다음 토큰을 설정하십시오.

```
debug_mode_for_user = Yes
```

serevu 에서 pam_seos 로그인 오류를 감지하도록 하려면 seos.ini 파일에서 다음 토큰을 설정하십시오.

```
serevu_use_pam_seos = Yes
```

제한 및 한계

이 절에서 설명하는 보호 방법에는 다음과 같은 제한 사항이 있습니다.

- Sun Solaris 의 경우 로그인 시도에 5 번 실패하면 serevu 가 통지됩니다.
- pam_seos 모듈은 PAM 을 지원하는 Sun Solaris, HP-UX 및 Linux 에서만 구현됩니다.

사용자 비활성 상태 검사

비활성 기능을 사용하면 조직에 더 이상 소속되지 않은 소유자의 계정으로 시스템에 무단 액세스하는 것을 금지할 수 있습니다. 비활성 일은 사용자가 로그인하지 않는 날입니다. 사용자 계정이 일시 중지되거나 로그인할 수 없는 제한 비활성 기간 일수를 지정할 수 있습니다. 계정이 일시 중지되면 해당 계정을 수동으로 다시 활성화해야 합니다.

참고: 비활성 검사에서 암호 변경은 활동으로 간주됩니다. 사용자의 암호가 변경되면 해당 사용자는 비활성이 되기 때문에 일시 중단할 수 없습니다.

USER 클래스 레코드 또는 GROUP 클래스 레코드의 비활성 속성으로 비활성 기간의 일 수를 설정할 수 있습니다. GROUP 클래스 레코드는 그룹을 프로필 그룹으로 가지는 사용자에게만 영향을 미칩니다. SEOS 클래스의 INACT 속성으로 전체 시스템에서 모든 사용자에 대한 비활성을 설정할 수도 있습니다.

selang 에서 다음 명령을 사용하여 비활성을 전체적으로 지정하십시오.

setoptions inactive (*numdays*)

그룹에 대한 전체 시스템의 비활성 설정보다 우선적되는 그룹에 대한일수를 설정하려면 다음 명령을 사용합니다.

editgrp *groupName* inactive (*numdays*)

사용자에 대한 그룹 및 전체 시스템 설정보다 우선되는 사용자에 대한 일수를 설정하려면 다음 명령을 사용합니다.

editusr *userName* inactive (*numdays*)

일시 중지된 사용자 계정을 다시 활성화하려면 다음 명령을 사용합니다.

editusr *userName* resume

일시 중지된 프로필 그룹을 다시 활성화하려면 다음 명령을 사용합니다.

editgrp *userName* resume

전체 시스템 수준의 비활성 로그인 확인을 비활성화하려면 다음 명령을 사용합니다.

setoptions inactive-

그룹에 대한 비활성 로그인 검사를 비활성화하려면 다음 명령을 사용합니다.

editgrp *groupName* inactive-

사용자에 대한 비활성 로그인 검사를 비활성화하려면 다음 명령을 사용합니다.

editusr *userName* inactive-

제 7 장: 사용자 암호 관리

이 섹션은 다음 항목을 포함하고 있습니다.

암호 제어

암호는 인증에 가장 많이 사용되는 장치이지만 암호 보호에는 다음과 같은 잘 알려진 문제가 있습니다.

- 간단한 암호는 추측하기 쉽습니다.
- 여러 해 동안 반복해서 계속 사용하는 암호는 결국 해킹됩니다.
- 네트워크를 통해 일반 텍스트 형태로 전송된 암호는 수신기에 의해 조작될 수 있습니다.

암호 정책 정의

가장 중요한 암호 규칙은 사용자가 일반 암호를 사용함으로써 자신의 암호가 명시적으로 또는 간접적으로 노출되게 하지 말아야 한다는 것입니다. 암호 보안을 적절하게 유지하려면 암호 사용에 관한 교육과 훈련이 필요합니다. CA Access Control 을 사용한다고 해서 이러한 교육 과정이 필요하지 않은 것은 아니지만 사용자가 최소한의 보안을 유지하는 암호를 사용하도록 규칙과 정책을 지정할 수는 있습니다. 지정할 수 있는 규칙은 다음과 같습니다.

- 새 암호는 기존 암호와 동일할 수 없습니다.
- 새 암호에 사용자 이름을 포함할 수 없습니다.
- 새 암호에 변경 중인 암호를 사용할 수 없습니다.
- 변경 중인 암호에 새 암호를 사용할 수 없습니다.
- 새 암호는 대/소문자에 관계없이 변경 중인 암호와 일치할 수 없습니다.
- 새 암호에는 지정된 최소 영숫자, 특수 문자, 숫자, 소문자 및 대문자를 사용해야 합니다.

- 새 암호에는 반복 문자를 사용하지 않아야 합니다.
- 새 암호는 seos.ini 파일의 Dictionary 토큰이 가리키는 사전에 제한된 단어 중 하나가 될 수 없습니다.
- 각 암호에는 최대 수명이 있어야 합니다. 즉, 사용자가 특정 간격 이후 새 암호를 선택하도록 암호가 만료되어야 합니다.
- 각 암호에는 최소 수명이 있어야 합니다. 최소 수명을 지정하여 너무 빨리 암호를 반복적으로 변경하지 않도록 할 수 있습니다. 암호를 신속하게 변경하면 암호 기록 목록을 오버플로한 후 이전 암호를 다시 사용할 수 있습니다.

중요! 암호 규칙은 기본 암호 도구가 아니라 sepass 에만 영향을 미칩니다. passwd 를 sepass 링크로 바꿔야 합니다.

암호 품질 검사 구성

암호 품질 검사를 구성하려면

1. CA Access Control 끝점 관리에서 "구성" 탭을 클릭합니다.

   구성 메뉴 옵션이 왼쪽에 나타납니다.

2. "기타" 섹션 옵션에서 "클래스 활성화"를 클릭합니다.

   "클래스 활성화" 페이지가 나타납니다.

3. "사용자 ID 제어" 섹션에서 PASSWORD 를 선택하고 "저장"을 클릭합니다.

   암호 품질 검사가 활성화됩니다.

4. "정책" 섹션 옵션에서 "사용자 암호 정책"을 클릭합니다.

   "사용자 암호 정책" 페이지가 나타납니다.

5. 암호 검사에 사용할 규칙을 정의하고 "저장"을 클릭합니다.

   이에 암호 검사에 대해 정의한 규칙이 암호가 변경될 때 적용됩니다.

6. (UNIX 에만 해당) sepass 유틸리티를 사용하여 새 암호를 업데이트합니다.

   참고: sepass 유틸리티에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

예: 암호 검사 규칙 정의

다음 selang 명령은 암호 품질 검사를 활성화하고 다음과 같은 최소값을 적용하는 암호 규칙을 정의합니다.

- 영숫자 문자 6 자
- 소문자 3 자
- 숫자 2 자

```
setoptions class+ (PASSWORD)
setoptions password(rules(alpha("6") lowercase("3") numeric("2")))
```

참고: setoptions 명령 형식에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

암호 변경

CA Access Control 에는 대부분의 사용자가 암호를 변경할 때 /bin/passwd 대신 사용하는 *ACInstallDir*/bin/sepass 실행 파일이 있습니다. 여기서 *ACInstallDir* 은 CA Access Control 의 설치 디렉터리로서, 기본 디렉터리는 /opt/CA/AccessControl/입니다.

- sepass 를 사용하는 경우에만 새 암호가 CA Access Control 암호 정책과 일치합니다. sepass 만이 새 암호와 암호가 변경된 날짜로 데이터베이스를 업데이트합니다. 또한 sepass 는 /bin/passwd 와 동일한 기능을 수행합니다.
- 원래 /bin/passwd 실행 파일은 CA Access Control 에 의해 수행된 암호 품질 검사를 *취*소한 후에 사용해야 합니다. 이 경우 원래 /bin/passwd 를 계속 사용할 수 있으며 CA Access Control 에서는 암호 품질을 검사하지 않고 시스템의 암호를 사용합니다.

selang 을 사용하여 암호를 변경할 수도 있습니다. 다음 명령을 입력하여 사용자에게 암호를 할당하십시오.

```
chusr userName password(string)
```

참고: 관리자가 다른 사용자의 암호를 변경하고 암호 검사가 활성화된 경우 사용자는 다음 로그인 시 암호를 변경해야 합니다.

암호 만료 및 유예 로그인

interval 매개 변수는 암호를 사용할 수 있는 최대 일 수를 설정합니다. 지정된 일수가 경과하면 CA Access Control 은 현재 암호가 만료되었음을 사용자에게 알립니다. 그러면 사용자가 암호를 즉시 갱신하거나 유예 로그인 횟수에 도달할 때까지 기존 암호를 계속 사용할 수 있습니다. 유예 로그인 횟수에 도달하여 더 이상 시스템에 액세스할 수 없을 때 새 암호를 선택하려면 시스템 관리자에게 문의해야 합니다.

암호 간격 지정

시스템에서 모든 사용자에게 새 암호를 묻기 전에 시스템 전체적인 수준에서 setoptions 명령을 사용하여 사용 기간을 지정합니다. segrace 유틸리티가 사용자 로그인 스크립트의 일부이거나 PAM 을 구성하여 segrace 를 호출하는 경우(기본 운영 체제에서 PAM 을 지원하는 경우) 지정된 일수에 도달하면 CA Access Control 은 현재 암호가 만료되었음을 사용자에게 알립니다. 그러면 사용자가 암호를 즉시 갱신하거나 유예 로그인 횟수에 도달할 때까지 기존 암호를 계속 사용할 수 있습니다. 유예 로그인 횟수에 도달하면 사용자의 시스템 액세스가 거부되므로 새 암호를 선택하려면 시스템 관리자에게 문의해야 합니다.

전체 시스템 수준에서 암호 간격을 설정하거나 취소하려면 다음 명령을 사용합니다.

setoptions password({interval(*NumDays*)|interval-})

NumDays 의 값은 0 또는 양의 정수여야 합니다. 간격이 0 이면 사용자에 대한 암호 간격 검사가 비활성화됩니다. 암호가 만료되지 않도록 하려면 간격을 0 으로 설정합니다. 0 간격은 보안 요구 사항이 낮은 사용자에게만 사용해야 합니다.

interva- 매개 변수는 암호 간격 설정을 취소합니다. 사용자에게 이 매개 변수 값을 가진 프로필 그룹이 있으면 그 값이 사용됩니다. 그렇지 않은 경우 setoptions 명령에서 설정된 기본값이 사용됩니다. 이 매개 변수는 chusr 또는 editusr 명령과 함께만 사용합니다.

개별 사용자 또는 그룹 암호 간격 설정

특정 사용자나 프로필 그룹에 대한 간격을 설정할 수도 있습니다. 이러한 설정은사용자나 그룹에 대한 시스템 전체 간격보다 우선됩니다. 지정된 일수에 도달하면 CA Access Control 은 현재 암호가 만료되었음을 사용자에게 알립니다. 그러면 사용자가 암호를 즉시 갱신하거나 유예 로그인 횟수에 도달할 때까지 기존 암호를 계속 사용할 수 있습니다. 유예 로그인 횟수에 도달하면 사용자의 시스템 액세스가 거부되므로 새 암호를 선택하려면 시스템 관리자에게 문의해야 합니다.

사용자에 대한 암호 간격을 설정하거나 취소하려면 다음 명령을 사용합니다.

editusr {interval(*NumDays*) | interval-}

그룹에 대한 암호 간격을 설정하거나 취소하려면 다음 명령을 사용합니다.

editgrp password{(interval(*NumDays*)) | (interval-)}

NumDays 의 값은 0 또는 양의 정수여야 합니다. 간격이 0 이면 암호 간격 검사가 비활성화됩니다. 암호가 만료되지 않게 하려면 간격을 0 으로 설정하십시오. 0 간격은 보안 요구 사항이 낮은 사용자에게만 사용해야 합니다.

interva- 매개 변수는 암호 간격 설정을 취소합니다. 암호 간격 설정이 취소되고 간격 값이 사용자 레코드에 설정된 경우 사용자 레코드의 값이 사용됩니다. 그렇지 않은 경우 setoptions 명령에서 설정된 기본값이 사용됩니다. 이 매개 변수는 setoptions, chgrp 또는 editgrp 명령과 함께만 사용하십시오.

유예 로그인

암호 검사가 활성화되면 CA Access Control 은 사용자가 로그인을 시도할 때마다 사용자의 암호가 만료되었는지 확인합니다. 암호가 만료된 경우 사용자가 더 이상 로그인할 수 없으면 몇 번 더 로그인할 수 있는 "유예" 기간이 제공됩니다.

유예 로그인 옵션은 암호 만료 후 사용자가 일시 중지될 때까지 허용되는 최대 로그인 횟수를 설정합니다. 유예 로그인 수는 0 에서 255 사이여야 합니다. 유예 로그인 횟수에 도달하고 나면 시스템에 대한 액세스가 거부되고 시스템 관리자에게 연락하여 새 암호를 선택해야 합니다. 유예 로그인 횟수가 0 으로 설정된 경우, 사용자는 로그인할 수 없습니다. 유예 로그인 수의 기본값은 5 입니다.

이 방법을 사용하여 사용자로 하여금 암호를 변경하도록 할 수 있습니다. 사용자 암호를 재설정하고 유예 로그인을 1 회 부여하면 사용자가 암호를 변경할 수 있습니다.

유예 로그인 추적

최종 사용자가 만료 후의 유예 로그인 횟수를 추적할 수 있으려면 사용자의 .login, .profile 또는 .cshrc 파일에 segrace 유틸리티 호출 명령을 삽입하십시오. 그러면 segrace 유틸리티는 남은 유예 로그인 횟수를 알려 주는 메시지를 표시합니다. segracex 유틸리티를 사용하여 사용자 암호의 만료 여부를 그래픽으로 확인할 수도 있습니다.

참고: segrace 및 segracex 유틸리티에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

유예 로그인 횟수에 대한 전체 시스템 기본값을 설정하려면 다음 명령을 입력합니다.

setoptions password(rules(grace(*nLogins*)))

특정 사용자에 대한 유예 로그인을 설정하거나 취소하려면 다음 명령을 입력합니다.

chusr *userName* {grace(*nLogins*) | grace-}

프로필 그룹에 대한 유예 로그인을 설정하거나 취소하려면 다음 명령을 입력합니다.

chgrp *groupName* {grace(*nLogins*) | grace-}

chusr 또는 chgrp 명령으로 설정한 값은 이 명령에 지정된 사용자에 대한 시스템 값보다 우선 적용됩니다.

참고: GROUP 클래스에 대한 grace 속성과 전역 유예 로그인 설정은 사용자의 암호가 만료된 **이후** 해당 사용자의 유예 로그인 횟수를 설정합니다. 그러나 USER 클래스의 유예 속성은 암호가 즉시 만료되도록 설정하며, 사용자 암호가 만료된 후에는 GROUP 레코드 또는 시스템 기본값을 사용하여 자동으로 유예 로그인이 설정됩니다. 그룹에 대한 암호 만료는 설정할 수 없으며 사용자에 대해서만 암호 만료를 설정할 수 있습니다.

제 8 장: 파일 및 프로그램 보호

이 섹션은 다음 항목을 포함하고 있습니다.

파일 및 디렉터리에 대한 액세스 제한

CA Access Control 은 UNIX 시스템의 사용 권한을 그대로 유지하지만 향상된 액세스 제어 계층을 추가합니다.

CA Access Control 은 다음과 같은 각각의 파일 액세스 작업을 차단하여 UNIX 로 제어를 반환하기 전에 사용자에게 특정 작업에 대한 권한이 부여되어 있는지 확인합니다. 액세스 유형은 괄호 안에 표시됩니다.

- 파일 작성(create)
- 읽기 권한으로 파일 열기(read)

  참고: 사용자가 파일에 대한 정보를 얻는 작업(예: ls -l)을 수행할 수 있는지 여부를 제어하기 위해 *read* 권한이 필요한 경우, STAT_intercept 구성 설정을 1 로 설정하십시오. 자세한 내용은 *Reference Guide*(참조 안내서)를 참조하십시오.
- 쓰기 권한으로 파일 열기(write)
- 파일 실행(execute)
- 파일 삭제(delete)
- 파일 이름 바꾸기(delete, rename)
- 사용 권한 비트 변경(chmod)
- 소유자 변경(chown)

- touch 명령 실행 등에 의한 타임스탬프 변경(utime)
- acledit 명령을 사용하여 액세스 제어 목록(ACL)을 지원하는 시스템의 Native ACL 편집(sec)
- 디렉터리 변경(chdir)

CA Access Control 액세스 검사는 다음과 같은 측면에서 기본 UNIX 권한 부여와 다릅니다.

- CA Access Control 은 유효 사용자 ID(euid)가 아닌 로그인한 사용자의 원래 사용자 ID 를 기준으로 권한 부여를 확인합니다. 예를 들면 *userA* 가 su 명령을 실행하여 다른 사용자 ID 로 교체한 경우 *userA* 는 여전히 *userA* 에게 허용되는 파일에 대한 액세스 권한만 가집니다. UNIX 에서와 같이 다른 사용자를 대리해도 대상 사용자의 파일에 대한 액세스 권한이 원래 사용자에게 자동으로 부여되지 *않습니다*.
- CA Access Control 은 슈퍼 사용자(root)에게 시스템의 모든 파일에 대한 자동 액세스 권한을 부여하지 않습니다. 슈퍼 사용자는 시스템의 다른 모든 사용자처럼 권한 부여가 검사됩니다.
- 권한 부여 검사는 CA Access Control 의 일반 및 조건부 액세스 목록, 날짜 및 시간 제한 사항, 보안 수준, 보안 범주 및 보안 레이블을 기반으로 합니다.
- 사용자에게 파일에 액세스할 권한을 특별히 부여하지 않은 경우 CA Access Control 은 해당 사용자가 파일에 액세스할 수 있도록 권한이 부여된 그룹에 속하는지 여부를 검사합니다.
- 각 파일 액세스는 일반 CA Access Control 감사 프로시저를 통해 감사됩니다.
- 파일을 삭제하는 경우 UNIX 에서는 사용자에게 부모 디렉터리에 대한 WRITE 권한이 필요한 반면 CA Access Control 에서는 해당 파일에 대한 DELETE 액세스 권한이 필요합니다.
- 파일 이름을 변경하려면 사용자에게 소스 파일에 대한 DELETE 액세스 권한과 대상 파일에 대한 RENAME 액세스 권한이 있어야 합니다. UNIX 에서는 사용자에게 부모 디렉터리에 대한 WRITE 액세스 권한이 있어야 합니다.
- 모든 사용자에게 /etc/passwd 및 /etc/group 파일의 기본 설정에 관계없이 이러한 파일에 대해 영구적인 READ 권한이 최소한 제공됩니다. 이렇게 하면 시스템이 종료되지 않습니다.

- CA Access Control 데이터베이스의 FILE 개체 소유자는 개체로 보호되는 파일에 대해 항상 모든 권한을 갖습니다.
- chdir 액세스 유형은 chdir 명령을 제어하며, UNIX 에서와 같이 실행되지 않습니다.

파일 보호 시스템의 제한 사항은 다음과 같습니다.

- 특수 _restricted 그룹의 구성원이 아닌 사용자와 관련하여 CA Access Control 은 다음과 같은 파일 및 디렉터리만 보호합니다.
  - 데이터베이스에서 개별 이름으로 정의된 파일 및 디렉터리
  - 데이터베이스에 정의된 이름 패턴과 일치하는 파일 및 디렉터리(예: /etc/*)

  그룹 _restricted 에 속한 사용자의 경우 모든 시스템 파일이 CA Access Control 로 보호됩니다. 데이터베이스에 정의되지 않은 파일에 대한 권한 부여는 FILE 클래스의 _default 레코드를 기반으로 합니다.
- CA Access Control 은 와일드카드를 사용하는 패턴을 비롯하여 보호가 필요한 리소스를 나타내는 모든 파일 이름과 디렉터리 이름의 테이블을 유지 관리합니다. 이 테이블에 사용할 수 있는 메모리 양은 제한되어 있습니다. 일반적으로, 데이터베이스의 개별 이름별로 정의할 수 있는 최대 파일 및 디렉터리 개수는 4096 개이며 최대 이름 패턴 개수는 512 개입니다.
- 일부 파일은 명시적 액세스 규칙이 없더라도 보호를 받습니다. 이러한 파일에는 CA Access Control 데이터베이스 파일, 감사 로그 및 구성 파일이 있습니다.

  참고: 자세한 내용은 *참조 안내서*의 FILE 클래스를 참조하십시오.

CA Access Control 은 다음과 같은 파일 액세스 유형을 지원합니다.

- ALL
- CHDIR
- CHMOD
- CHOWN
- CONTROL
- CREATE
- DELETE
- EXECUTE

- NONE
- READ
- RENAME
- SEC
- UPDATE
- UTIME
- WRITE

파일 보호 시스템은 중요한 데이터가 들어 있는 선택한 파일 집합을 보호하는 데 유용합니다. 예를 들어 CA Access Control 을 사용하여 다음과 같은 파일을 보호할 수 있습니다.

- /etc/passwd
- /etc/group
- /etc/hosts
- /etc/shadow

사용자 사이트에서 server 데몬에만 액세스가 허용되는 데이터베이스 및 중요한 기타 모든 파일을 보호하려면 CA Access Control 을 사용해야 합니다.

항상 액세스를 제어해야 하는 일부 파일은 규칙을 지정하지 않더라도 규칙에 의해 관리됩니다.

파일 보호 작동 방식

seosd 데몬은 시작할 때 데이터베이스에 정의된 개별 파일 개체에 UNIX stat 명령을 실행합니다. 그런 다음 각 파일 개체에 대한 항목을 포함하는 테이블을 메모리에 만듭니다. 또한 개별 파일에 대해 테이블에는 파일의 inode 및 장치가 포함되어 있습니다. 장치 및 inode 에 따라 보호되므로 CA Access Control 은 이러한 정보를 사용하여 파일에 대한 하드 링크도 보호할 수 있습니다. 데이터베이스는 파일의 inode 및 장치에 대한 정보를 보관하지 않습니다.

CA Access Control 을 통해 새 파일 규칙을 만들 경우:

- 파일이 UNIX 에 있는 경우 CA Access Control 은 파일에 대해 먼저 stat 명령을 수행한 다음 파일의 inode 및 장치 정보와 함께 새 항목을 파일 테이블에 추가합니다.
- 파일이 UNIX 에 없는 경우 CA Access Control 은 inode 및 장치 정보는 포함하지 않고 파일의 이름에 대한 새 항목을 파일 테이블에 추가합니다. 이 항목은 일반 파일 개체의 항목과 동일합니다. 동시에 커널은 내부 테이블에 inode 및 장치 정보를 만드는 동안 이 파일을 확인해야 한다는 표시를 저장합니다. 계속해서 파일이 작성되면 커널은 작성을 차단하고 파일의 inode 및 장치 정보를 seosd 에 알려주므로 seosd 가 파일 테이블에 있는 파일 항목을 업데이트할 수 있습니다.

파일을 삭제하면 CA Access Control 은 seosd 파일 테이블에서 해당 항목을 삭제하지만, 파일을 다시 작성하는 경우에는 CA Access Control 데이터베이스에 항목이 유지됩니다.

파일 보호

selang 에 보호된 파일을 정의하려면 다음 명령을 입력합니다.

```
newres FILE filename
```

예를 들어 /tmp/binary.bkup 라는 파일을 등록하려면 다음 명령을 입력합니다.

```
newres FILE /tmp/binary.bkup
```

참고: 파일 규칙의 액세스 유형을 지정하지 않고 파일 규칙을 정의하면 기본 액세스 NONE 이 할당됩니다. 이런 경우 파일 소유자만 해당 파일에 액세스할 수 있습니다.

보호된 대부분의 파일은 슈퍼 사용자가 액세스할 수 없게 보호해야 합니다. 그렇지 않으면 슈퍼 사용자의 암호를 알고 있는 사용자가 파일에 자동으로 액세스할 수 있습니다. 또한, 파일 소유자를 제외한 다른 모든 사용자가 파일에 액세스할 수 없게 할 수 있습니다.

비슷한 이름의 여러 파일을 보호하려면 와일드카드를 포함한 파일 이름 패턴을 사용하십시오. 사용할 수 있는 와일드카드는 *****(0 개 이상의 문자를 의미) 및 **?** (**/**와 다른 하나의 문자를 의미)입니다.

지정하는 패턴은 파일의 전체 경로 이름과 일치하여 /tmp/x* 패턴은 이름이 /tmp/x1, /tmp/xxx 및 /tmp/xdir/a 인 파일과 일치합니다.

CA Access Control 에서 지정할 수 *없는* 패턴은 **/***, **/tmp/*** 및 **/etc/***입니다.

중요! 파일 이름 패턴은 강력한 도구이므로 임의로 사용하지 마십시오.

예를 들어 /tmp 디렉터리에 있는 파일 이름이 a 로 시작해서 b 로 끝나는 모든 파일(예: /tmp/axyz/axyzb)이 보호 받는 파일로 다음 명령은 정의합니다.

```
newres FILE /tmp/a*b
```

FILE 리소스 이름의 와일드카드

파일 리소스 이름에 와일드카드를 사용하면 여러 파일에 해당하는 파일 레코드를 만들 수 있습니다. 파일 이름이 와일드카드 패턴과 일치할 경우 해당 파일은 레코드와 관련된 액세스 권한으로 보호됩니다.

사용할 수 있는 와일드카드는 다음과 같습니다.

- * - 수 제한 없는 모든 문자
- ? - 문자 한 개

실제 리소스 이름이 리소스 레코드 이름 여러 개와 일치할 경우에는 가장 긴 와일드카드가 아닌 일치 항목이 해당 리소스에 사용됩니다.

CA Access Control 에서는 FILE 리소스의 이름에 다음 패턴을 사용하지 *않습니다*.

- *
- /*
- /tmp/*
- /etc/*

예: FILE 리소스에 와일드카드 사용

FILE 리소스 /usr/lpp/bin/*는 /usr/lpp/bin 아래 있지만 깊게 중첩되지 않은 모든 파일과 하위 디렉터리를 보호합니다.

파일 액세스 제한

selang 에서 파일에 대한 슈퍼 사용자의 액세스를 제한하려면 더 긴 버전의 newres 명령을 사용하십시오. 예를 들어 myuser 사용자가 아닌 다른 모든 사용자뿐 아니라 슈퍼 사용자도 /tmp/binary.bkup 파일에 액세스할 수 없게 하려면 다음 selang 명령을 사용합니다.

```
newres FILE /tmp/binary.bkup owner(myuser) defaccess(N)
```

이 명령은 다음을 수행합니다.

1. /tmp/binary.bkup 을 보호되는 파일로 정의합니다.
2. 사용자 myuser 를 파일의 소유자로 설정하여 파일에 myuser 액세스 권한을 부여합니다.
3. 파일의 기본 액세스를 NONE 으로 설정하여 다른 모든 사용자가 파일에 액세스할 수 없도록 합니다. 다른 사용자에게 파일에 대한 액세스를 허용 하려면 그 파일에 대한 액세스 규칙을 명시적으로 정의해야 합니다.

중요! 루트 권한으로 selang 명령을 실행한 다음 명시적으로 다른 사용자를 파일의 소유자로 지정하지 않고 FILE 레코드를 정의하면 루트가 해당 파일의 소유자가 됩니다. 소유자인 루트(root 로 로그인한 모든 사용자를 의미)는 해당 파일에 자유롭게 액세스합니다.

참고: seos.ini 파일의 use_unix_file_owner 토큰을 yes 로 설정할 수 있습니다. 이렇게 하면 일반 UNIX 사용자는 자신이 소유한 파일에 대한 액세스 규칙을 정의할 수 있습니다.

파일 액세스 방지

소유자가 없는 FILE 레코드를 정의하는 것이 편리한 경우가 있습니다. selang 에서 소유자가 없는 FILE 레코드를 정의하려면 특수 소유자 "nobody"를 사용하십시오.

예를 들어 /tmp/binary.bkup 파일을 보호된 파일로 정의하고 모든 사용자가 이 파일에 액세스할 수 없게 하려면 다음 selang 명령을 입력합니다.

```
newres FILE /tmp/binary.bkup owner(nobody) defaccess(N)
```

이 newres 명령은 명령을 정의한 사용자가 root 인지 여부에 관계없이 파일에 액세스할 수 없게 합니다. 모든 사용자파일에 액세스할 수 없게 한 경우 한 명 이상의 사용자에게 그 파일에 명시적으로 액세스할 수 있게 권한을 부여해야 합니다.

사용자가 보호된 파일에 액세스할 수 있게 명시적으로 허용하려면 authorize 명령을 사용하십시오. 예를 들어 사용자 "*userJo*"에게 /tmp 디렉터리에 있는 Jo 로 시작하는 모든 파일에 대한 업데이트 권한을 부여하려면 다음 selang 명령을 입력합니다.

```
authorize FILE /tmp/Jo* uid(userJo) acc(Update)
```

참고: CA Access Control 은 해당 데이터베이스에 정의된 파일만 보호합니다.

사용자가 파일 정보를 가져오지 못하게 제한

사용자에게 파일이나 디렉터리에 대한 *읽기* 액세스 권한을 제공하지 않으면 사용자는 기본적으로 계속 stat 함수를 사용하여 파일 정보를 가져올 수 있습니다. 예를 들어 파일 /tmp/abc 에 대한 *읽기* 액세스 권한이 없는 사용자는 다음 작업을 수행할 수 있습니다.

```
ls -l /tmp/abc
```

읽기 액세스 권한이 없는 사용자가 파일 정보를 가져오지 못하게 하려면 STAT_intercept 구성 설정을 1 로 설정합니다.

참고: STAT_intercept 구성 설정에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

기본 액세스 권한 보기

_restricted 그룹에 있는 사용자의 기본 액세스를 보려면(일치하는 레코드가 없는 경우) selang showres 명령과 함께 클래스의 _default 레코드를 사용하십시오.

예를 들어 _restricted 그룹의 사용자가 CA Access Control 데이터베이스에 없는 파일에 대해 가진 기본 액세스 권한을 보려면 showres selang 명령을 사용하여 FILE 클래스의 _default 리소스를 표시합니다.

```
showres FILE _default
```

참고: 다른 모든 사용자는 특정 CA Access Control 데이터베이스 규칙으로 정의된 액세스 권한을 갖습니다.

조건부 액세스 제어 목록 사용

파일 액세스에 사용되는 프로그램에서 파일 액세스 권한을 조건부로 만들 수 있습니다. 이 방법으로 파일 액세스를 조건부로 만드는 작업을 프로그램 경로 지정이라고 합니다.

참고: 파일에 액세스하도록 지정된 프로그램이 셸 스크립트인 경우 셸 스크립트의 첫 줄에는 #!/bin/sh 가 있어야 합니다.

다음 코드는 암호 변경 프로그램 /bin/passwd 를 제어하는 /etc/passwd 파일을 임의의 프로세스로 업데이트할 수 있는 예제입니다. /bin/passwd 에서 시작되지 않은 /etc/passwd 파일에 대한 모든 액세스 시도가 차단됩니다.

```
newres FILE /etc/passwd owner(nobody) defaccess(R)
authorize FILE /etc/passwd gid(users) access(U) via(pgm(/bin/passwd))
```

newres 명령은 /etc/passwd 파일을 CA Access Control 에 정의하고 파일 소유자를 포함한 모든 사용자가 파일을 읽을 수 있도록 허용합니다. /bin/passwd 프로그램에서 액세스된 경우 authorize 명령을 통해 모든 사용자가 파일에 액세스할 수 있습니다. 암호 파일을 이러한 방식으로 보호하면, 사용자가 /bin/passwd 프로그램을 사용하지 않는 경우 /etc/passwd 파일에 항목을 삽입하는 트로이 목마나 "users" 그룹의 사용자에 의한 암호 파일 업데이트가 차단됩니다.

조건부 액세스 목록은 DBMS(데이터베이스 관리 시스템)의 파일에 대한 액세스를 제어하는 데도 유용합니다. 일반적으로 사용자가 데이터베이스 공급업체에서 제공한 프로그램과 유틸리티를 통해서만 해당 파일에 액세스할 수 있게 허용해야 합니다. 다음 명령을 생각해 봅시다.

```
authorize FILE /usr/dbms/xyz uid(*) via(pgm(/usr/dbms/bin/pgm1)) access(U)
authorize FILE /usr/dbms/xyz uid(*) via(pgm(/usr/dbms/bin/pgm2)) access(U)
```

DBMS 바이너리 디렉터리에 속한 프로그램 pgm1 이나 pgm2 에 의해 액세스하는 경우 이러한 authorize 명령 집합을 통해 모든 CA Access Control 사용자는 DBMS 시스템의 xyz 파일에 액세스할 수 있습니다. 사용자 피연산자에 별표를 사용할 때는 주의하십시오. 별표는 CA Access Control 에 정의된 모든 사용자를 지정합니다. 기본 액세스 권한이 CA Access Control 에 정의되지 않은 사용자에게도 적용된다는 점을 제외하면 별표의 사용은 개념적인 면에서 기본 액세스 권한과 유사합니다. CA Access Control 데이터베이스에 정의되지 않은 사용자에 대해서는 _undefined 그룹을 사용할 수 있습니다.

Unicenter TNG 달력 ACL 속성을 사용하여 Unicenter TNG calendar 상태에 따라 특정 사용자 및 그룹에 대한 액세스를 허용하거나 거부할 수도 있습니다. Unicenter TNG calendar 에 대한 두 가지 유형의 ACL 속성에는 regular 및 restrictive 가 있습니다.

예를 들어, 다음 명령은 george 라는 사용자를 basecalendar 라는 일반 달력의 조건부 액세스 제어 목록에 추가합니다.

```
auth file file1 uid(george) calendar(basecalendar) access(rw)
```

또한 다음 명령은 Unicenter TNG calendar 에서 george 라는 사용자를 제거합니다.

```
auth- file file2 uid(george) calendar(basecalendar)
```

네거티브 액세스 제어 목록 사용

NACL(네거티브 액세스 제어목록)을 사용하여 사용자나 액세스 관련 액세스 유형을 거부할 수 있습니다.

CA Access Control 언어(selang)를 사용하여 액세스를 거부하려면 다음 명령을 사용합니다.

```
auth className resourceName [gid(group-name...)] \
[uid({user-name...|*})] [deniedaccess(accessvalue)]
```

_abspath 그룹으로 트로이 목마 차단

$PATH 변수의 상대 경로 이름, 특히 "현재 디렉터리"를 의미하는 점(.) 경로 이름은 보안상의 약점이 됩니다. 다음과 같은 시나리오를 생각해 봅시다.

- root 의 PATH 변수 맨 위가 현재(.) 디렉터리입니다.
- 악의적인 사용자가 파괴적인 프로그램, 트로이 목마를 작성하여 /tmp/ls 로 저장합니다.
- 이 과정에서 악의 있는 사용자가 예상한 대로 루트는 /tmp 디렉터리에서 ls 명령을 실행합니다. 일반적인 ls 명령을 실행하는 대신 완전한 관리 권한을 가진 루트로 /tmp 디렉터리에서 저장되어 있던 트로이 목마를 실행합니다.

이 보안 취약점을 해결하기 위해 CA Access Control 은 _abspath 라는 사용자 그룹을 제공합니다. _abspath 그룹의 모든 구성원은 프로그램을 호출할 때 상대 경로 이름을 사용할 수 없습니다.

사용자를 다른 그룹에 추가하는 것처럼 _abspath 그룹에 추가할 수 있습니다. 다음 로그인 시 유효한 사용자는 프로그램에 액세스할 때 상대적 경로 이름을 사용할 수 없습니다.

Native UNIX 보안 동기화

CA Access Control 권한은 기본 UNIX 권한보다 더 복잡하지만 사용자가 가진 기본 UNIX 권한을 CA Access Control 권한으로 동기화할 수 있습니다. 즉, 사용 권한을 일치시킬 수 있습니다. 하지만 동기화에는 다음과 같은 몇 가지 한계가 있습니다.

- 동기화를 수행하면 되돌릴 수 없습니다. 일단 동기화가 적용되면 새로 호출된 모든 CA Access Control 인증 명령을 제어할 수 있지만 기존 액세스 규칙에는 적용되지 않습니다.
- CA Access Control 에서 사용자가 부여한 사용 권한은 UNIX 로 전달되지만 UNIX 에서 부여한 사용 권한은 CA Access Control 로 전달되지 않습니다.
- UNIX 자체 사용 권한 시스템의 제한 사항 때문에 UNIX 에서는 단순한 CA Access Control 권한 형식을 두 개 이상 채택하지 못할 수 없습니다. ACL(액세스 제어 목록) 기능이 있는 UNIX 버전의 경우에도 CA Access Control ACL 의 복잡성을 모두 반영하지 못할 수 있습니다.

CA Access Control 로 동기화할 수 있는 ACL 이 있는 UNIX 플랫폼은 Sun Solaris, HP-UX 및 Tru64 입니다.

해당 ACL 이 없는 경우에도 기존 UNIX rwx 권한을 가능한 범위까지 CA Access Control 권한으로 동기화할 수 있습니다.

동기화는 authorize 명령의 UNIX 옵션과 seos.ini 파일 SyncUnixFilePerms 토큰의 조합으로 제어됩니다.

- UNIX 옵션을 포함하면 CA Access Control 뿐만 아니라 UNIX 의 구현에 대해서도 authorize 명령이 호출됩니다. 이 명령은 이전에 사용 권한이 없었던 UNIX 사용 권한도 부여할 수 있습니다.

  UNIX 옵션을 사용하지 *않으면* selang 명령이 UNIX 보안에 적용되지 않습니다. 또한 UNIX 에서 금지 사항을 유지하면 CA Access Control 권한이 적용되지 않습니다. 따라서 selang 에서 UNIX 제한을 해결할 수 있는 유일한 방법은 authorize 명령의 UNIX 옵션을 사용하는 것입니다.

- authorize 명령에서 UNIX 옵션은 SyncUnixFilePerms 토큰이 seos.ini 파일의 [seos] 섹션에 올바르게 설정된 경우에만 작동합니다. 이 토큰에 허용된 값은 다음과 같습니다.

  - **no** 는 액세스 제어 목록(ACL) 허용 권한을 동기화하지 않는다는 의미입니다. 이것이 기본값입니다.
  - **warn** 은 ACL 사용 권한을 동기화하지 않지만 CA Access Control 사용 권한과 기본 UNIX 사용 권한이 충돌하면 경고 메시지를 표시하도록 지정합니다.
  - **traditional** 은 CA Access Control ACL 에 따라 그룹의 rwx 권한을 조정하도록 지정하며 개별 사용자의 사용 권한은 UNIX 에 복사되지 않습니다.
  - **acl** 은 CA Access Control ACL 에 따라 UNIX ACL 을 조정하도록 지정합니다.
  - **force** 는 CA Access Control defaccess 사용 권한에 따라 UNIX 전체 액세스 특성을 조정하도록 지정합니다.

  SyncUnixFilePerms 토큰 값의 변경 사항은 seosd 데몬을 다시 시작한 후에만 적용됩니다.

예: 동기화

다음 예제에는 /var/temp/newdata 라는 파일과 fowler 라는 사용자가 있으며, FILE 클래스의 레코드가 이 파일을 나타낸다고 가정합니다.

1. seosd 데몬을 종료하면 seos.ini 파일을 편집할 수 있습니다.

   ```
   # secons -s
   ```

2. 허용 권한이 있는 사용자로 로그인하고 seos.ini 파일을 편집하여 다음과 같이 [seos] 섹션에 SyncUnixFilePerms 행을 만듭니다.

   ```
   SyncUnixFilePerms = acl
   ```

   acl 은 UNIX 옵션이 CA Access Control ACL 에 따라 UNIX ACL 을 조정함을 의미합니다. 이 토큰이 acl 로 설정된 상태로 있는 한 UNIX 옵션에 이 함수가 있습니다.

3. seosd 데몬을 다시 시작합니다.

   ```
   # seosd
   ```

4. selang 을 호출하고 다음 selang 명령을 실행합니다.

   ```
   authorize FILE /var/tmp/newdata uid(fowler) access(r w) unix
   ```

   이 명령은 fowler 에게 새 데이터 파일에 대한 Read 및 Write 권한을 제공하며 UNIX 옵션을 지정하여 해당하는 기본 UNIX 사용 권한을 부여합니다.

HP-UX 제한 사항

HP-UX 의 액세스 제어 목록(ACL)은 CA Access Control 의 ACL 반영 방법에 따른 제한이 있습니다.

- HP-UX 에서 ACL 은 사용자와 그룹 *조합*마다 액세스를 지정합니다. 따라서 할당된 액세스는 사용자의 주 그룹도 지정된 경우에만 지정된 사용자에게 적용됩니다.

  반면 CA Access Control 은 사용자 또는 그룹마다 액세스 권한을 할당하지만 조합마다 할당하지는 않습니다.

  그에 따라 CA Access Control 권한은 사용자 또는 그룹이 "*" 또는 "any"에 상응하게 설정된 HP-UX 사용자/그룹 조합에 매핑됩니다.
- HP-UX 는 볼륨 관리자(LVM)에서 제어하는 파일 시스템에서 ACL 을 지원하지 않습니다. 따라서 일부 중요한 HP-UX 시스템은 일반적으로 "루트" 파일 시스템에서만 ACL 동기화를 허용합니다.
- HP-UX 의 ACL 은 16 개 항목으로 제한됩니다. CA Access Control 동기화는 사용 가능한 항목을 최대한 효율적으로 사용하지만 16 개 항목으로 모든 CA Access Control ACL 을 완전하게 반영하는 데는 부족할 수 있습니다.

Sun Solaris 제한 사항

Sun Solaris 에서 기본 UNIX ACL 은 /tmp 디렉터리에 구현되지 않습니다.

중요한 파일 모니터링

Watchdog 은 지정된 다른 모든 파일뿐 아니라 setuid/setgid 프로그램의 이진 파일도 보호할 수 있습니다. seoswd 유틸리티(Watchdog 데몬)는 다음과 같은 두 가지 문제를 계속 검사합니다.

- seosd 데몬이 활성 상태이며 응답하는지 여부. 필요한 경우 watchdog 데몬은 seosd 데몬을 다시 시작합니다.
- 사용자가 트러스트된 프로그램이나 파일을 수정했는지 여부. 수정한 경우 seoswd 는 이러한 파일이 실행되지 않도록 합니다.

seosd 데몬이 생성(fork)되면 자동으로 seoswd 프로그램이 실행되어 Watchdog 을 시작합니다.

참고: seoswd 에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

seos.ini 파일에는 watchdog 의 시간 초과 값과 검색을 제어하는 여러 가지 토큰이 포함됩니다. 또한 이 값에 대한 최신 문서도 포함됩니다.

참고: seos.ini 파일에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

Watchdog 을 사용하여 일반 파일 변경 시 감사 레코드 생성을 비롯하여 일반 파일에 setuid 및 setgid 프로그램에 대해 만든 백그라운드 검사와 동일한 백그라운드 검사를 수행할 수 있습니다.

예를 들어 보안 관리자만 /etc/inittab 파일을 수정할 수 있도록 허용된 구성을 생각해 보십시오. CA Access Control 이 파일을 모니터링하고 수정 시 경고를 생성하도록 하려면 selang 에서 다음 명령을 사용합니다.

```
newres SECFILE /etc/inittab
```

이제 /etc/inittab 파일의 수정 사항이 즉시 모니터링됩니다.

내부 파일 보호

설치 중 CA Access Control 은 다음과 같은 두 가지 유형의 내부 파일을 보호하기 위한 규칙을 작성합니다:

- 내부 규칙 - 구성 파일, 로그 파일, 데이터베이스 파일을 보호합니다.

  내부 규칙은 삭제할 수 없습니다.
- 기본 규칙 - 통신을 암호화하고 인증하는 데 사용하는 루트 및 서버 인증서와 같은 민감한 파일을 보호합니다.

  설치 후에 기본 규칙을 삭제할 수 있습니다.

내부 파일 규칙

내부 파일 규칙은 구성 파일, 로그 파일, 데이터베이스 파일을 보호합니다. 내부 파일 규칙은 selang 에서 보이지 않으며 감지되지 않습니다.

CA Access Control 이 내부 파일 규칙을 사용하여 보고하는 파일은 다음 액세스 권한을 갖습니다.

- CA Access Control 내부 프로세스에 대한 완전한 액세스
- 기타 모든 접근자에 대한 읽기 및 실행(해당하는 경우) 액세스

FILE 규칙을 작성하여 내부 파일 규칙을 대체할 수 있습니다. 이러한 FILE 규칙을 삭제하면 CA Access Control 이 내부 파일 규칙으로 되돌립니다.

CA Access Control 은 내부 파일 규칙을 사용하여 다음 파일을 보호합니다. 표의 두 번째 열에는 파일 위치를 지정하는 구성 설정(해당하는 경우)이 나열되어 있습니다.

참고: 일부 파일 위치는 내부적으로 정의되며 해당 구성 설정이 없습니다. 이러한 파일의 위치는 구성할 수 없습니다.

파일	seos.ini 의 구성 설정 및 섹션	기본 파일 위치
모든 데이터베이스 파일	[seosd] dbdir	*ACInstallDir*/seosdb
seos.ini	-	*ACInstallDir*
privpgms.ini	-	*ACInstallDir*/etc
loginpgms.ini	-	*ACInstallDir*/etc
xdmpgms.init	-	*ACInstallDir*/etc
nfsdevs.init	[seosd] nfs_devices	*ACInstallDir*/etc
osver	-	*ACInstallDir*/etc
accommon.ini	-	*ACSharedDir*
seos.audit	[logmgr] audit_log	*ACInstallDir*/log
seos.audit.bak*	[logmgr] audit_back	*ACInstallDir*/log
seos.error	[logmgr] error_log	*ACInstallDir*/log
kbl.audit	[kblaudit] audit_log	*ACInstallDir*/log

파일	seos.ini 의 구성 설정 및 섹션	기본 파일 위치
kbl.audit.bak	[kblaudit] audit_back	*ACInstallDir*/log
kbl.error	[kblaudit] error_log	*ACInstallDir*/log

참고: 구성 설정에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

기본 파일 규칙

CA Access Control 은 민감한 파일을 보호하기 위해 설치 중 기본 파일 규칙을 만듭니다. 기본 파일 규칙은 selang 에서 보이며 삭제될 수 있습니다.

다음 표는 CA Access Control 이 기본 파일 규칙을 사용하여 보호하는 민감한 파일과 이 파일에 대한 액세스 권한 및 허용된 접근자를 나열합니다.

표에서 *PMDBDir* 는 정책 모델 데이터베이스(PMDB)가 있는 디렉터리이고, *pmd_name* 은 각 정책 모델의 이름입니다. 기본적으로 *PMDBDir* 는 *ACInstallDir*/policies 에 있습니다. *PMDBDir* 의 위치는 seos.ini 파일의 pmd 섹션에 있는 _pmd_directory_ token 에 정의되어 있습니다.

파일	기본 액세스	허용된 접근자
ACInstallDir/data/crypto/crypto.dat	없음	sechkey
ACInstallDir/data/crypto/def_root.pem*	없음	sechkey
ACInstallDir/data/crypto/sub.key	없음	sechkey
ACInstallDir/data/crypto/sub.pem	없음	sechkey
ACInstallDir/log/policyfetcher.log	읽기	+policyfetcher
ACInstallDir/ladb/*db.la*	읽기	sebuildla
/etc/passwd	모두	모두
/etc/shadow	모두	모두
PMDBDir/pmd_name/hsock	Read, Write, Execute, Cre, Chown, Chmod, Utime	seagent, sepmdd
PMDBDir/pmd_name/pmd.ini	읽기	seagent, sepmdd

파일	기본 액세스	허용된 접근자
PMDBDir/pmd_name/seos_*	Read, Write, Execute, Cre, Chown, Chmod, Utime	seagent, sepmdd
PMDBDir/pmd_name/socket	Read, Write, Execute, Cre, Chown, Chmod, Utime	seagent, sepmdd

setuid 및 setgid 프로그램 보호

setuid(사용자 ID 설정) 프로그램은 UNIX 에서 가장 많이 사용되는 프로그램입니다. setuid 프로그램을 호출하는 프로세스는 자동으로 setuid 프로그램 소유자의 ID 를 얻습니다. setuid 프로그램의 소유자가 루트인 경우 일반 사용자는 setuid 프로그램을 실행하면 자동으로 슈퍼 사용자가 됩니다. setuid 프로그램이 시작되면 이 프로세스는 슈퍼 사용자가 권한을 갖는 모든 작업을 수행합니다. setuid 프로그램이 필요한 작업만 수행하도록 하십시오. setuid 프로그램 내의 백 도어 또는 셸은 사용자에게 시스템의 모든 것에 대한 액세스를 허용합니다.

CA Access Control 은 PROGRAM 클래스를 사용하여 setuid 및 setgid 프로그램을 보호합니다. 설치 시 CA Access Control 은 기본적으로 모든 프로그램 실행을 허용합니다. 데이터베이스에서 트러스트된 프로그램을 정의한 후 CA Access Control 의 동작을 변경하여 setuid 또는 setgid 프로그램이 트러스트된 프로그램으로 정의되지 않은 경우에는 해당 프로그램을 실행하지 못하도록 할 수 있습니다. 예를 들어 /bin/ps(프로세스 상태 프로그램)를 setgid 프로그램으로 실행할 수 있게 하려면 다음과 같은 selang 명령을 사용하십시오.

```
newres PROGRAM /bin/ps defaccess(EXEC)
```

CA Access Control 은 /bin/ps 프로그램을 트러스트된 프로그램으로 등록합니다. 그런 다음 해당 CRC, inode 수, 크기, 장치 번호, 소유자, 그룹, 사용 권한 비트, 마지막으로 수정한 시간 및 데이터베이스의 PROGRAM 클래스에 있는 레코드의 기타 디지털 서명(옵션)을 계산하고 저장합니다.

Watchdog 은 프로그램의 CRC, 크기, inode 및 기타 특징을 정기적으로 확인합니다. 이러한 값이 변경되면 Watchdog 은 트러스트된 프로그램 목록에서 프로그램을 제거하고 이러한 프로그램에 대한 액세스를 거부하도록 seosd 에 요청합니다. 이렇게 하면 setuid 프로그램을 수정하거나 이동하여 프로그램을 잘못 사용하는 일은 없습니다. 예제 newres 명령에서 사용 권한은 데이터베이스에 정의되지 않은 사용자를 포함하여 모든 사용자가 /bin/ps 명령을 실행할 수 있게 합니다.

언트러스트된 setuid 프로그램은 UNIX 기반 운영 체제의 가장 위험한 보안 허점일 수 있습니다. 보안 관리자는 트러스트된 프로그램의 액세스 규칙을 사용함으로써 테스트와 점검을 통해 무결성이 확인된 트러스트된 프로그램에만 setuid 를 사용하도록 제한할 수 있습니다. 하지만 모든 사용자가 트러스트된 실행 파일을 자동으로 시작할 수는 없습니다. 액세스 규칙으로 setuid 프로그램에 액세스가 허용된 사용자와 그룹을 명시적으로 지정해야 합니다. 예를 들어 다음과 같은 selang 명령 세트는 /bin/su 의 실행을 시스템 부서 사용자(sysdept 그룹)에게만 권한을 부여합니다.

```
newres PROGRAM /bin/su defaccess(NONE)
authorize PROGRAM /bin/su gid(sysdept) access (EXEC)
```

별표(*)를 사용하여 데이터베이스에 정의된 모든 사용자를 지정합니다. 예를 들어 CA Access Control 에 정의된 모든 사용자가 su 명령을 수행하도록 허용하려면 다음 명령을 입력합니다.

```
authorize PROGRAM /bin/su uid(*) access(EXEC)
```

이 설명은 setgid 실행 파일에도 적용됩니다.

nr 및 er 명령을 사용하여 PROGRAM 클래스에서 setuid 및 setgid 프로그램을 등록할 수 있습니다. 비슷한 방법으로 setuid 및 setgid 가 아닌 중요한 프로그램을 PROGRAM 클래스에 등록할 수 있습니다. 권한 없는 사용자가 이러한 프로그램을 업그레이드할 수 없도록 이러한 프로그램에 대한 파일 규칙을 정의합니다. 프로그램 실행이 언트러스트된 경우(프로그램이 업그레이드 이후 다시 트러스트되지 않고 실행된 경우) 프로그램 실행을 허용하려면 blockrun 속성을 no 로 설정합니다.

- blockrun 속성이 yes 로 설정된 경우 프로그램이 다시 트러스트될 때까지 프로그램이 실행되지 않으며 관련 PACL 이 허용한 파일에 액세스할 수 없습니다. PACL 은 프로그램이 다시 트러스트될 때까지 효율적으로 비활성화됩니다.
- blockrun 속성이 no 로 설정된 경우 프로그램이 실행되지만 관련 PACL 에서 허용하는 리소스에 액세스할 수 없습니다.

blockrun 속성 값을 yes 로 설정하려면 다음 editres/newres 명령을 사용합니다.

```
er program /bin/p blockrun
```

blockrun 속성 값을 no 로 설정하려면 다음 editres/newres 명령을 사용합니다.

```
er program /bin/p blockrun-
```

기본적으로, PROGRAM 클래스에 등록된 모든 프로그램의 경우 blockrun 속성은 yes 로 설정됩니다. eos.ini 파일에서 SetBlockRun 토큰을 사용하여 이 속성을 변경할 수 있습니다. 자세한 내용은 seos.ini 파일을 참조하십시오.

참고: CA Access Control 은 FILE 클래스가 아닌 PROGRAM 클래스를 사용하여 setuid 및 setgid 프로그램을 보호합니다.

setuid/setgid 프로그램 자동 정의

CA Access Control 은 setuid 및 setgid 프로그램을 모두 자동으로 정의하는 방법을 제공합니다. /bin/seuidpgm 유틸리티 프로그램을 사용하여 모든 setuid 프로그램 및 해당 사용 권한을 정의하는 명령 집합을 작성합니다.

예를 들어, setuid 및 setgid 프로그램에 대해 전체 파일 시스템을 스캔하고 /tmp/pgm_script 파일에 생성된 selang 명령을 작성하려면 다음 selang 명령을 입력합니다.

```
# seuidpgm -qln / -x /home > /tmp/pgm_script
```

실행하기 전에 필요에 따라 seuidpgm 으로 생성된 출력 파일을 편집하고 수정할 수 있습니다.

참고: seuidpgm 유틸리티에 대한 자세한 내용은 *참조 안내서*를 참조하십시오. setuid 나 setgid 이외의 프로그램에 유사한 보호를 제공하는 방법에 대해 알아 보려면 *참조 안내서*의 SECFILE 클래스를 참조하십시오.

조건부 액세스

기타 복잡한 사용 권한 기술은 조건부 액세스 규칙입니다. 예를 들어 지문 판독기를 사용하여 사용자의 ID 를 확인한 후 사용자를 슈퍼 사용자로 허용하는 securedSU 라는 su 명령의 안전 버전이 있다고 가정합니다.

이 프로그램에서 *UserX* 가 슈퍼 사용자가 될 수 있는지 확인하는 한 가지 방법은 조건부 액세스 규칙을 다음과 같이 설정하는 것입니다. 규칙을 설정하기 전에 USER.root 에 대해 defaccess(none)를 설정해야 합니다.

```
authorize SURROGATE USER.root uid(UserX) via(pgm(securedSU))
```

로그인 명령 보호

/bin/login 을 슈퍼 사용자로만 사용하도록 제한하는 것이 좋습니다. 그렇지 않으면 다른 사용자의 암호를 알고 있는 사용자가 그 사용자로 로그인한 후 그 사람의 암호를 제공하여 모든 대리 및 터미널 제한을 바이패스할 수 있습니다.

selang 에서 /bin/login 사용 권한을 변경하려면 다음 명령을 사용합니다.

```
chres LOGINAPPL /bin/login defaccess(N) owner(root)
```

일반 프로그램 보호

CA Access Control 에서는 setuid 및 setgid 프로그램을 보호하는 방식과 같은 방식으로 일반 프로그램을 보호할 수도 있습니다. 이렇게 하려면 PROGRAM 클래스의 blockrun 속성을 사용자가 선택한 값으로 설정합니다.

참고: 가능한 옵션에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

커널 모듈 로드 및 언로드 보호

*커널 모듈*은 UNIX 운영 체제의 구성 요소로서, 실행 중인 커널을 확장하기 위해 로드할 수 있으며 커널이 필요 없을 때 언로드할 수 있습니다. 메모리 리소스를 낭비할 필요 없이 필요에 따라 기능을 로드할 수 있으므로 이 기능은 유연성을 더해줍니다. 그렇지 않으면 메모리 리소스에서 기본 커널의 모든 예상 가능한 기능을 감당해야 합니다.

CA Access Control 에서 커널 모듈 보호를 비활성화하거나 활성화할 수 있습니다. 커널 모듈 보호를 활성화하면 CA Access Control 이 커널 모듈을 로드 및 언로드하는 시스템 호출을 차단한 다음 데이터베이스에서 KMODULE 클래스의 레코드인 관련 레코드에 대해 요청된 액세스 권한을 확인합니다. 커널 모듈 레코드인 CA Access Control 에 대한 액세스가 요청되면 요청된 액세스는 "load" 또는 "unload"입니다.

Linux 가 아닌 모든 시스템에서 KMODULE 레코드 이름은 전체 경로가 아니라 커널 모듈 파일 이름과 일치해야 합니다. 이는 모듈 이름이 파일 이름과 동일하기 때문입니다. Linux 에서 KMODULE 레코드 이름은 커널 모듈 이름과 일치해야 하고 이 이름은 실제 파일 이름과 다를 수 있습니다. Linux 에서 파일 이름을 변경해도 Linux 가 사용하는 모듈 이름은 변경되지 않고 KMODULE 레코드는 유효하게 유지됩니다.

커널 모듈 로드에 대한 파일 경로 검사를 활성화하고 요청된 액세스가 load 인 경우 CA Access Control 은 다음과 같은 추가 검사를 수행합니다.

- KMODULE 레코드의 filepath 속성에 유효한 절대 파일 경로만 포함되어 있는지 검사합니다.
- 경로 이름 filepath 의 파일에 KMODULE 레코드 이름과 일치하는 모듈이 포함되어 있는지 검사합니다.
- 커널 모듈이 KMODULE 속성(Linux 가 아닌 시스템의 filepath, Linux 시스템의 signature)과 일치하는지 검사합니다.

  참고: CA Access Control 은 Linux 시스템에서 커널 모듈 파일에 대한 고유 서명을 생성하고 이 서명을 signature 속성 값으로 커널 모듈 레코드에 삽입합니다. CA Access Control 에서는 각 액세스의 서명을 확인합니다. CA Access Control 이 서명을 자동으로 계산하고 삽입하기 때문에 서명을 직접 입력할 필요가 없습니다. 그러나 seretrust 유틸리티를 사용하여 이 작업을 수행할 수 있습니다.

추가 정보:

커널 모듈 로드 시 파일 경로 검사 활성화 및 비활성화 (페이지 114)

커널 모듈 보호

커널 모듈의 로드 및 언로드를 보호하여 운영 체제를 보호할 수 있습니다.

커널 모듈을 보호하려면

1. 커널 모듈 보호를 활성화했는지 확인하십시오.
2. CA Access Control 에서 KMODULE 레코드를 만듭니다.

   커널 모듈을 만들려면 다음을 정의해야 합니다.

   - 커널 모듈 이름

     Linux 가 아닌 모든 시스템에서 KMODULE 레코드 이름은 전체 경로가 아니라 커널 모듈 파일 이름과 일치해야 합니다. 이는 모듈 이름이 파일 이름과 동일하기 때문입니다. Linux 에서 KMODULE 레코드 이름은 커널 모듈 이름과 일치해야 하고 이 이름은 실제 파일 이름과 다를 수 있습니다.

   - 레코드 소유자(기본값은 모듈을 만드는 사용자)
   - (선택 사항) 커널 모듈 파일의 절대 파일 경로 또는 파일 경로 목록(모듈 버전이 여러 개 있는 경우)

   참고: HP 및 Solaris 시스템의 경우 특수 커널 모듈 _ALL_MODULES 를 정의하여 모든 커널 모듈 언로드를 보호할 수 있습니다.
3. 모듈을 로드 및 언로드할 권한이 부여된 사용자 또는 그룹을 정의합니다.

예: selang 명령을 사용하여 커널 모듈 보호

다음 selang 명령은 커널 모듈 serial.o 를 CA Access Control 에 정의 및 권한 부여하고 엔터프라이즈 사용자 kadmin 에게 이 커널 모듈을 로드 및 언로드할 권한을 부여합니다.

```
newres kmodule serial.o owner(kadmin) defaccess(none) \
filepath(/lib/modules/2.2.19/serial.o:/lib/modules/2.2.20/serial.o)
authorize kmodule serial.o access(load, unload) xuid(kadmin)
```

커널 모듈 보호 활성화 및 비활성화

커널 모듈 보호가 활성화되면 CA Access Control 은 CA Access Control 데이터베이스에 정의된 커널 모듈의 로드 및 언로드를 확인합니다.

기본적으로 CA Access Control 에서는 커널 모듈 보호를 활성화합니다.

커널 모드 보호를 활성화하거나 비활성화하려면 setoptions 명령 등을 사용하여 KMODULE 클래스를 활성화 또는 비활성화합니다.

예: selang 을 사용하여 커널 모드 보호 활성화

다음 selang 명령은 커널 모드 보호를 활성화합니다.

```
setoptions class+(kmodule)
```

예: selang 을 사용하여 커널 모드 보호 비활성화

다음 selang 명령은 커널 모드 보호를 비활성화합니다.

```
setoptions class-(KMODULE)
```

커널 모듈 로드 시 파일 경로 검사 활성화 및 비활성화

커널 모듈 보호가 활성화되면 커널 모듈 로드 시 파일 경로 검사를 활성화할 수도 있습니다. 이 옵션이 활성화되면 CA Access Control 은 로드되는 커널 모듈이 KMODULE 레코드의 filepath 속성과 일치(Linux 가 아닌 시스템)하거나 KMODULE 레코드 서명과 일치(Linux 시스템)하는지 확인합니다.

파일 경로 검사를 활성화하려면 구성 파일 seos.in 의 seosd 섹션에서 special_check 토큰을 yes 로 설정합니다(기본값은 no).

CA Access Control 에서는 파일 경로 검사와 커널 모드 보호가 둘 다 활성화된 경우에만 파일 경로 검사를 수행합니다.

예: seini 유틸리티를 사용하여 커널 모듈 로드에 대한 파일 경로 검사 활성화

커널 모듈 로드에 대한 파일 경로 검사를 활성화하려면 다음과 같이 seini 및 secons 유틸리티를 사용합니다.

```
seini -s seosd.special_check yes
secons -rl
```

kill 명령으로부터 이진 파일 보호

데이터베이스 서버나 응용 프로그램 데몬처럼 업무 수행에 중요한 프로세스를 서비스 거부 공격으로부터 보호해야 합니다. 기본 UNIX 보안 시스템은 프로세스 사용자 ID 를 해당 프로세스를 기준으로 보호합니다. 이것은 기본 UNIX 에서 루트가 모든 프로세스에 모든 작업을 수행할 수 있음을 의미합니다. CA Access Control 은 프로세스에서 실행 중인 실행 파일을 기준으로 규칙을 정의하여 UNIX 에 프로세스 보호를 강화합니다. CA Access Control 프로세스 보호는 프로세스의 사용자 ID 와 *관련이 없습니다.* PROCESS 클래스의 레코드는 CA Access Control 이 보호하는 모든 프로세스를 정의해야 합니다.

예를 들어 ASCII 뷰어 /bin/more 가 강제 중단되지 않도록 하려면 다음 프로시저를 수행합니다.

1. selang 을 시작합니다.
2. 다음 selang 명령을 입력합니다.

   ```
   newres PROCESS /bin/more defaccess(N) owner(nobody)
   ```

   이 명령은 /bin/more 를 강제 중단 시도로부터 보호할 프로세스로 정의합니다. 따라서 기본 액세스는 *NONE*(N)입니다. **owner(nobody)** 설정은 이 규칙을 정의한 사용자조차도 /bin/more 프로세스를 강제 중단시킬 수 없도록 합니다.
3. selang 을 종료합니다.
4. 2 단계에서 정의한 규칙을 테스트합니다.
   a. 다음 명령을 입력합니다.

      ```
      /bin/more /tmp/seosd.trace
      ```

   b. /bin/more 가 즉시 종료되지 않을 만큼 /tmp/seosd.trace 파일이 충분히 크다고 가정하고 Ctrl+Z 를 눌러 /bin/more 프로세스를 일시 중단합니다.
   c. 다음 명령을 입력하여 일시 중단된 작업의 강제 중단을 시도해 봅니다.

      ```
      kill %1
      ```

      사용자 시도가 실패하고 CA Access Control 에서 "사용 권한이 거부되었습니다"라는 메시지를 표시합니다.

특정 사용자가 /bin/more 프로세스를 종료할 수 있도록 하는 예외를 만들려면 다음 selang 명령을 입력합니다.

```
authorize PROCESS /bin/more uid(username)
```

참고: 동일한 프로시저를 사용하여 시스템에서 다른 바이너리 실행 파일이 종료되지 않도록 보호합니다.

CA Access Control 은 일반 중지 신호(SIGTERM) 및 응용 프로그램이 마스크할 수 없는 중지 신호(SIGKILL 및 SIGSTOP)를 보호합니다. SIGHUP 또는 SIGUSR1 과 같은 다른 신호는 그 프로세스가 kill 신호를 무시할지 또는 반응할지 여부를 결정하도록 통과시킵니다.

제 9 장: 로그인 명령 제어

이 섹션은 다음 항목을 포함하고 있습니다.

로그인 프로세스 제어

CA Access Control 에서는 터미널 및 응용 프로그램에 의한 두 가지 로그인 보호를 제공합니다. TERMINAL 클래스를 사용하면 어떤 터미널 또는 호스트에서 어떤 사용자가 로그인할 수 있는지 설정할 수 있습니다.

참고: TERMINAL 클래스에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

LOGINAPPL 클래스를 사용하면 어떤 사용자 또는 그룹이 어떤 로그인 응용 프로그램(예: telnet, ftp, rlogin)을 사용하여 로그인할 수 있는지 제어할 수 있습니다. 클래스의 액세스 규칙을 설정하여 각 로그인 응용 프로그램에 대한 특정 규칙을 정의합니다. 예를 들면 모든 사용자가 자신의 호스트에 ftp 를 사용할 수 있게 하고 제한된 수의 사용자가 자신의 시스템에 telnet 을 사용할 수 있으며 아무도 시스템에 rlogin 을 사용할 수 없도록 하는 규칙을 정의할 수 있습니다. LOGINAPPL 클래스의 각 레코드는 특정 로그인 응용 프로그램에 대한 액세스 규칙을 정의합니다.

예: LOGINAPPL

예를 들어 ftp 응용 프로그램을 사용하는 익명 사용자만 허용하려면 다음 절차를 수행하십시오.

1. 다음 selang 명령을 사용하여 기본 액세스 권한을 none 으로 변경합니다.

   ```
   cr LOGINAPPL FTP defaccess(NONE) owner(nobody)
   ```

2. 다음 selang 명령을 사용하여 익명의 사용자가 ftp 를 사용하도록 허용합니다.

   ```
   auth LOGINAPPL FTP uid(anonymous) access(X)
   ```

account 그룹의 사용자가 텔넷만 사용하도록 제한하려면:

1. 다음 selang 명령을 사용하여 rlogin 및 rsh 의 사용을 차단합니다.

   ```
   auth LOGINAPPL(RLOGIN RSH) gid(account) access(N)
   ```

2. 다음 selang 명령을 사용하여 account 그룹이 텔넷을 사용하도록 허용합니다.

   ```
   auth LOGINAPPL TELNET gid(account) acc(X)
   ```

참고: 위의 예는 RLOGIN 및 RSH 제한을 보여 주지만 다른 로그인 프로그램도 포함되어야 합니다.

새 로그인 프로그램을 추가하거나 사용할 때마다 새 LOGINAPPL 레코드를 추가해야 합니다.

로그인 차단 시퀀스는 항상 setgid 또는 setgroup 이벤트와 함께 시작되며 이를 *트리거*라고 합니다. 이 시퀀스는 사용자 ID 를 로그인한 실제 사용자로 변경하는 setuid 이벤트로 끝납니다.

로그인 응용 프로그램은 CA Access Control 이 로그인 활동을 모니터링하는 데 사용하는 다양한 시스템 호출을 실행합니다. 이러한 로그인 시퀀스는 표준 로그인 응용 프로그램에 미리 설정되어 있습니다. 이는 CA Access Control 추적 파일을 살펴보면 알 수 있습니다.

참고: LOGINAPPL 클래스 및 시퀀스 설정에 대한 자세한 내용은 *selang 참조 안내서*를 참조하십시오.

SFTP 로그인 차단 활성화

사용자가 SFTP 를 사용하여 끝점에 로그인하면 SFTP 응용 프로그램이 SSH 를 사용하여 사용자를 인증합니다. CA Access Control 이 SFTP 응용 프로그램으로부터의 로그인 시도를 차단할 때는 기본적으로 이 로그인을 SSH 로그인으로 간주하며, 로그인 시도를 허용 또는 거부하기 위해 SSH LOGINAPPL 레코드에 대한 규칙을 사용합니다.

CA Access Control 이 SFTP 와 SSH 로그인 시도를 구분하도록 구성하고, SFTP 와 SSH 로그인에 대한 서로 다른 규칙을 작성하려면 SFTP 로그인 차단을 활성화해야 합니다.

SFTP 로그인 차단을 활성화하려면

1. 끝점에서 명령 프로그램 창을 엽니다.
2. 다음 selang 명령을 입력합니다.

   er LOGINAPPL SSH loginflags(EXECLOGIN)

   이 명령은 SSH 로그인에 대한 트리거가 프로세스가 수행하는 첫 번째 EXEC 작업이 되도록 지정합니다.
3. 다음 selang 명령을 입력합니다.

   er LOGINAPPL SFTP loginpath(*path*) defaccess(a)

   loginpath(*path*)

   SFTP 응용 프로그램에 대한 전체 경로를 정의합니다.

   이 명령은 SFTP 로 명명된 LOGINAPPL 레코드를 만들고, SFTP 로그인 응용 프로그램에 대한 경로를 정의하고, 추가 제한 사항이 없는 경우 모든 사용자가 SFTP 를 사용하여 끝점에 로그인할 수 있게 지정합니다.

예: SFTP 로그인 차단 활성화

이 예는 /usr/libexec/openssh/sftp-server 에 있는 SFTP 로그인 응용 프로그램에 대한 SFTP 로그인 차단을 활성화합니다. 첫 번째 selang 명령은 또한 CA Access Control 이 PAM 로그인에 대해 PAM 로그인 차단을 사용하도록 지정합니다.

```
er LOGINAPPL SSH loginflags(EXECLOGIN, PAMLOGIN)
er LOGINAPPL SFTP loginpath(/usr/libexec/openssh/sftp-server) defaccess(a)
```

참고: LOGINAPPL 클래스에 대한 자세한 내용은 *selang 참조 안내서*를 참조하십시오.

포괄적인 로그인 응용 프로그램 제어

CA Access Control 은 또한 일반 로그인 응용 프로그램을 제어하고 보호할 수 있습니다. 이는 특정 규칙을 일반 패턴과 연결하는 로그인 응용 프로그램 그룹을 보호할 수 있다는 의미입니다. 일반 로그인 응용 프로그램을 정의하려면 LOGINAPPL 클래스를 사용합니다.

포괄적인 로그인 응용 프로그램 정의

selang 을 사용하여 일반 로그인 응용 프로그램을 정의하려면 LOGINPATH 매개 변수의 경우를 제외하고 일반 로그인 제한을 설정할 때와 동일한 명령을 사용합니다. 이 명령에는 [,], *, ?와 같은 문자 중 하나 이상을 사용하는 정규식으로 구성된 일반 경로가 포함되어야 합니다. 예를 들어 일반 telnet 응용 프로그램을 정의하려면 다음 명령을 실행합니다.

```
er LOGINAPPL GENERIC_TELNET loginpath(/usr/sbin/in.tel*)
```

일반 로그인 프로그램 차단

일반 로그인 제한에서는 활성화된 규칙이 명확합니다. 즉, loginpath 속성에 대해 차단된 로그인 프로그램이 지정되어 있는 LOGINAPPL 개체가 데이터베이스에 있을 경우 해당 개체의 규칙이 적용됩니다.

그러나 일반 LOGINAPPL 개체의 경우 CA Access Control 은 다음을 수행합니다.

1. seosd 는 차단된 로그인 응용 프로그램에 대해 정확히 일치하는 개체를 검색합니다. (LOGINAPPL 개체의 일치하는 로그인 경로) 일치하는 개체가 발견되면 해당 규칙이 적용됩니다.
2. 일치하는 개체를 찾지 못한 경우에는 일치하는 일반 로그인 경로를 이용하여 LOGINAPPL 개체를 계속 검색합니다.
3. 일치하는 개체가 여러 개이면 구체적으로 일치하는 내용이 더 많은 개체의 규칙이 적용됩니다.

터미널을 사용할 사용자 권한 정의

침입자가 시스템에 액세스하는 것을 차단하는 가장 효율적인 방법 중 하나는 터미널 보호입니다(즉, 로그인 소스). 소스는 사용자가 로그인하는 호스트 또는 터미널(예: X 터미널 또는 콘솔)일 수 있습니다.

그러나 요즘의 아키텍처로 이루어진 터미널은 더 이상 텔레타이프 컴퓨터(텔레타이프 컴퓨터용으로 UNIX 가 개발되었음)가 아닙니다. 대부분의 사이트에는 가상 터미널 서버(PTS) 또는 X window manager 를 통해 "가상 터미널"이 할당되어 있고 터미널의 이름은 보안 시스템에 대한 의미 없는 기호일 뿐입니다. CA Access Control 은 터미널로서 인식되는 항목을 보호합니다. CA Access Control 에서 다음 세 가지 방법 중 하나로 터미널을 정의할 때 CA Access Control 은 로그인 단계에서 터미널 보호를 구현합니다.

- 사용자가 XDM 로그인 창을 통해 X 터미널에서 로그인하면 CA Access Control 은 해당 로그인 요청에 사용되는 터미널이 되도록 /etc/hosts, NIS 또는 DNS 에서 호스트 이름으로 변환된 X 터미널의 IP 주소를 받습니다. 또한 CA Access Control 은 호스트 이름으로의 변환이 실패하거나 IP 사용을 선호하는 경우 IP 주소를 사용하여 보호할 수도 있습니다.
- 사용자가 단순(dumb) 터미널에서 로그인하는 경우 TTY 이름이 터미널을 식별합니다.
- 사용자가 네트워크에서 (telnet, rlogin, rsh 등을 통해) 로그인하면 (/etc/hosts, NIS, DNS 를 통해) 호스트 이름으로 변환되는 IP 주소가 터미널 이름이 됩니다.

이 호스트를 TERMINAL 클래스에 정의하고 해당 사용자와 그룹을 개체의 액세스 목록에 추가하여 특정 호스트에 대한 로그인 규칙을 정의할 수 있습니다. 또한 각 로그인 소스에 대해 TERMINAL 개체에 날짜 및 시간 제한 사항을 설정하여 이 호스트 또는 터미널에서 로그인이 허용되는 날짜 및 시간을 제한할 수 있습니다. TERMINAL 클래스에 와일드카드를 사용하여 패턴(호스트 이름 또는 IP 주소)과 일치하는 호스트를 정의할 수 있습니다.

대부분의 경우 슈퍼 사용자 또는 시스템 관리자와 같이 중요한 권한이 부여된 사용자는 안전한 위치의 터미널로 제한해야 합니다. 슈퍼 사용자로 시스템에 침입하려는 침입자 및 해커는 자신의 원격 스테이션에서 침입할 수 없습니다. 안전한 위치에 있는 허가된 터미널 중 하나에서 작업해야 시스템에 침입할 수 있기 때문입니다.

네트워크에서 로그인하면 사용자가 호스트 콘솔을 사용 중이라는 보장이 없습니다. 사용자는 해당 호스트에 연결된 터미널을 사용 중일 수도 있고 요청 호스트에서 서비스 수신이 허가된 네트워크의 다른 노드에서 통신 중일 수도 있습니다. 사용자가 다른 호스트에서 로그인하도록 허용하는 것은 특정 스테이션의 사용자뿐 아니라 해당 스테이션에서 허가한 다른 터미널의 사용자에게도 로그인을 허용한다는 것을 의미합니다. 부서 간 차단을 확실히 하려면 터미널 그룹을 정의하고 각 부서의 사용자가 해당 부서의 터미널 그룹에서만 작업할 수 있도록 합니다.

다른 리소스와 달리 터미널 권한 부여에서는 사용자에게 정보 액세스 권한이 더 많이 부여될수록 사용자의 터미널 권한은 더 적게 부여되어야 합니다. 슈퍼 사용자는 안전하지 않은 원격 터미널에서 누구도 루트로 로그인하지 못하도록 하기 위해서 터미널 액세스가 가장 제한된 사용자여야 합니다.

터미널을 정의할 때 CA Access Control 은 터미널 정의의 소유자를 명시적으로 지정할 것을 요구합니다. 그 이유는 루트가 보안 관리자로서 기본적으로 터미널의 소유자가 되면 슈퍼 사용자가 터미널에 로그인할 수 있기 때문입니다. 대부분의 경우 이는 바람직하지 않습니다. 우연히 허점을 만들 수 있는 실수를 방지하기 위해 CA Access Control 은 터미널을 정의할 때 소유자를 정의하도록 합니다.

터미널 tty34 를 정의하려면 다음 명령을 사용합니다.

```
newres TERMINAL tty34 defaccess(none) owner(userA)
```

이 명령은 터미널 tty34 에 대한 레코드를 생성하고 기본 액세스를 NONE 으로 설정하며 userA 를 소유자로 정의합니다. userA 는 터미널의 소유자로서 터미널 tty34 를 통해 시스템에 자동으로 허용되어 들어갈 수 있다는 것에 유의하십시오.

모든 사용자가 터미널 tty34 에서 로그인하지 못하도록 하려면 "nobody"를 소유자로 지정합니다.

```
newres TERMINAL tty34 defaccess(none) owner(nobody)
```

특정 터미널에서 로그인하는 사용자를 허용하려면 다음 명령을 입력합니다.

```
authorize TERMINAL tty34 uid(USR1)
```

이 명령은 USR1 이 터미널 tty34 에서 로그인하도록 허용합니다.

터미널 사용 권한은 그룹에도 부여될 수 있습니다. 예를 들어 다음 명령은 그룹 DEPT1 의 구성원이 터미널 tty34 를 사용하도록 허용합니다.

```
authorize TERMINAL tty34 gid(DEPT1)
```

터미널 그룹을 정의하려면 다음 명령을 입력합니다.

```
newres GTERMINAL TERM.DEPT1 owner(ADM1)
```

터미널 그룹 TERM.DEPT1 에 구성원 터미널을 추가하려면 다음 명령을 입력합니다.

```
chres GTERMINAL TERM.DEPT1 mem(tty34, tty35)
```

이 터미널 그룹을 사용하도록 USR1 을 인증하려면 다음 명령을 입력합니다.

```
authorize GTERMINAL TERM.DEPT1 uid(USR1)
```

이렇게 하면 USR1 에 tty34 및 tty35 를 모두 사용할 수 있도록 권한을 부여합니다.

root 사용자에 대한 터미널 제한

또 다른 고려 사항은 TERMINAL 클래스의 기본 규칙입니다. 초기 구현 단계에서 기본값은 정의되지 않은 항목을 허용하도록 설정됩니다. TERMINAL 의 경우 이 설정은 하나의 결점이 될 수도 있습니다.

일단 TERMINAL 클래스의 기본값을 READ 로 설정하면 root 를 포함하여 모든 사용자가 데이터베이스에서 특정 TERMINAL 레코드를 가지고 있지 않은 모든 터미널로부터 로그인할 수 있습니다. 사용자는 슈퍼 사용자가 모든 터미널에서 로그인하도록 원하지는 않습니다. 그러나 TERMINAL 클래스의 기본값을 [없음]으로 설정하여 데이터베이스의 각 터미널을 강제로 정의할 수 있지만 이는 바람직하지 않습니다.

먼저 TERMINAL 클래스의 기본값을 READ 로 설정하여 루트를 비롯한 모든 사용자가 데이터베이스에 특정 TERMINAL 레코드를 가지지 않는 터미널에서 로그인하도록 할 수 있습니다. 슈퍼 사용자라고 하더라도 몇몇 터미널에서는 로그인이 허용되지 않도록 할 수 있습니다. 그러나 TERMINAL 클래스의 기본값을 NONE 으로 설정하여 데이터베이스에 각 터미널을 정의하도록 하는 방법도 있습니다. 이 방법은 일부 사이트에서는 비현실적일 수 있습니다.

이 문제를 해결하기 위해 CA Access Control 은 TERMINAL 클래스의 _default 레코드 내에서 액세스 제어 목록 정의를 지원합니다. 다음 명령은 최소한의 노력으로 루트를 두 가지 터미널로 제한하는 방법을 보여 줍니다.

newres TERMINAL term2 defaccess(N) owner(root)
newres TERMINAL _default defaccess(R)
authorize TERMINAL _default uid(root) access(N)
처음 두 개의 명령은 term1 및 term2 를 root 가 소유한 터미널로 정의하기 때문에 슈퍼 사용자로 로그인하기에 적합합니다. newres TERMINAL _default 및 chres 명령은 기본 액세스를 READ 로 설정하기 때문에 모든 사용자가 데이터베이스에서 정의되지 않은 모든 터미널에 액세스할 수 있습니다. authorize 명령은 정의되지 않은 터미널에 대한 슈퍼 사용자의 액세스를 명시적으로 거부합니다.

처음 두 명령은 루트가 소유하는 터미널로서 term1 과 term2 를 정의하기 때문에 슈퍼 사용자 로그인에 적합합니다. newres TERMINAL _default 및 chres 명령을 통해 기본 액세스를 READ 로 설정하면 모든 사용자는 데이터베이스에 정의되지 않은 모든 터미널에 액세스할 수 있습니다. authorize 명령은 슈퍼 사용자의 정의되지 않은 터미널 액세스를 명시적으로 차단합니다.

참고: UACC 클래스가 아직 존재하며 이 클래스를 사용하여 리소스의 기본 액세스를 지정할 수 있습니다. 그러나 작업을 쉽게 하려면 _default 레코드를 사용하여 리소스의 기본 액세스를 지정하는 것이 좋습니다.

권장되는 제한 사항

TERMINAL 클래스의 기본 액세스가 READ 로 설정된 경우에는 loopback 및 localhost 터미널과 스테이션 호스트 이름 사용을 제한해야 합니다. 이러한 터미널의 사용을 허용하면 대상 사용자의 암호를 아는 경우 다른 모든 사용자가 자신의 사용자 ID 를 대체할 수 있게 됩니다. 예를 들어 다음과 같은 경우를 생각해 봅시다.

- 터미널 T 에 슈퍼 사용자로 로그인할 수 없습니다.
- 사용자 U 가 root 에 대해 사용자 ID 를 대체할 권한이 없습니다.
- 사용자 U 가 슈퍼 사용자 암호를 알아냈습니다.

- 모든 사용자가 터미널 루프백에서 로그인할 수 있습니다.
- 사용자 U 는 명령 telnet 루프백을 간단하게 수행하고, 사용자 ID root 를 지정하고, 암호를 입력하여 이 액세스 규칙 설정을 바이패스할 수 있습니다. 이제 슈퍼 사용자 로그인을 허용하지 않는 터미널 T 에서 슈퍼 사용자 세션이 시작됩니다. 이와 유사한 방식으로 사용자는 로컬 호스트 또는 스테이션의 호스트 이름을 이용하여 액세스 규칙을 바이패스할 수 있습니다.

사용자 U 는 telnet loopback 명령을 실행하고 사용자 ID 를 루트로 지정한 다음 암호를 입력하여 이 액세스 규칙 세트를 바이패스할 수 있습니다. 이제 슈퍼 사용자 로그인이 허용되어선 안될 터미널 T 에서 슈퍼 사용자 세션이 시작되었습니다. 이와 비슷하게 사용자는 로컬 호스트 또는 스테이션의 호스트 이름을 사용하여 액세스 규칙을 바이패스할 수 있습니다.

이러한 세 가지 취약점을 제한하려면 다음 정의를 사용합니다.

```
newres TERMINAL loopback defaccess(N) owner(nobody)
newres TERMINAL localhost defaccess(N) owner(nobody)
chres TERMINAL hostname defacc(N) owner(nobody)
```

이 보안 위반을 예방하는 다른 방법은 텔넷, ftp 등에 대한 TCP 요청을 로컬 호스트로부터 제한하는 것입니다.

또한 다른 선택 사항으로는 TERMINAL 그룹의 기본 액세스를 NONE 으로 설정한 다음 TERMINAL 및 GTERMINAL 규칙을 지정하는 것입니다.

암호 확인 및 로그인 제한

CA Access Control 은 /bin/login 실행 파일을 바꾸지 않습니다. CA Access Control 이 실행 중인 경우에도 섀도 암호 파일인 /etc/passwd 또는 NIS passwd 맵과 비교하는 암호 확인은 계속됩니다. 또한 CA Access Control 은 다음 절에 설명되어 있는 추가 검사를 수행합니다.

로그온 검사

로그인 프로세스가 인증 단계를 통과하면 CA Access Control 은 프로세스를 차단하고 다음 사항을 검사합니다.

- 암호가 만료되었습니까?

  암호가 만료된 경우 사용자는 액세스가 거부되기 전에 경고와 함께 유예 로그인 횟수를 받습니다. 액세스가 거부되면 보안 관리자는 사용자의 암호를 다시 할당해야 합니다. 유예 로그인 횟수는 사용자 암호 정책에 의해 결정되는데 setoptions 명령을 사용하여 전역적으로 지정하거나 chgrp 명령으로 프로필 그룹에 대해 지정할 수 있습니다.

  참고: setoptions 명령에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

  segrace 유틸리티를 사용하면 사용자에게 남은 유예 로그인 횟수, 사용자의 기존 암호가 만료되기까지 남은 날짜 수, 사용자가 마지막으로 로그온한 날짜와 시간 및 로그온을 수행한 터미널을 볼 수 있습니다.

  참고: segrace 명령에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

- 사용자가 인가된 터미널에서 로그온합니까?

  미리 정의된 제한을 기준으로 현재 시간과 요일로 로그인을 허용하는지 여부

- 미리 정의된 제한에 따라 현재 시간 및 요일에 로그인을 허용해야 합니까?

  이 사용자 이름이 미리 정의된 일 수 이상 사용되지 않았는지 여부

- 사용자 이름이 미리 정의된 날짜 수를 초과할 때까지 사용되지 않았습니까?

  그런 경우 액세스가 거부됩니다 (기본값은 90 일입니다. 변경하려면 setoptions 를 사용합니다).

시간 및 날짜 로그인 규칙 정의

활동이 적은 시간의 정보 보안이 가장 취약합니다. 심야 시간대와 주말은 감사(Audit) 레코드를 모니터할 수 있는 권한을 가진 사람이 소수이기 때문에 침입에 적합한 시간입니다. 적절한 터미널 권한 규칙을 설정하면 침입자는 보호된 위치의 터미널을 사용할 수 밖에 없습니다. 요일(DOW) 및 시간(TOD) 액세스 규칙을 설명하면 침입자는 회사의 업무 시간 중에 침입 시도를 해야 합니다. 이 조합은 외부 침입을 매우 제한합니다.

사용자 로그인 날짜와 시간 제한은 사용자별로 다르게 설정합니다. 사용자에 대해 DOW 및 TOD 로그인 제한을 정의하려면 다음 명령을 사용합니다.

```
chusr USR1 restrictions(days(Mon,Tue,Wed)time(800:1700))
```

또한 날짜 매개 변수는 일주일 내내 로그인을 허용하는 ANYDAY 값 및 월요일부터 금요일까지만 로그인을 허용하는 WEEKDAYS 값을 수락합니다. 또한 시간 매개 변수는 하루 중 언제라도 로그인을 허용하는 ANYTIME 값을 수락합니다.

참고: DOW 및 TOD 제한을 데이터베이스에서 정의된 많은 리소스에 적용할 수 있습니다. 이 기능은 특히 터미널과 터미널 그룹의 사용 시간을 제한하는 데 유용합니다.

참고: DOW 및 TOD 제한은 데이터베이스에 정의된 *많은* 리소스에 적용할 수 있습니다. 이 기능은 특히 터미널 및 터미널 그룹에 사용 제한 기간을 지정할 때 유용합니다.

동시 로그인 비활성화

대부분의 UNIX 기반 운영 체제는 동시 로그인을 허용합니다. 그러나 사용자는 사용 중인 특정 터미널에서 계속 로그인할 수 있습니다. secons 명령을 다음 스위치와 함께 사용합니다.

사용자가 로그인한 후 CA Access Control 은 사용자가 자신의 동시 로그인 권한을 비활성화할 수 있도록 하여 다른 사용자가 다른 터미널에서 이 사용자로 로그인하지 못하도록 합니다. 그러나 사용자는 자신이 사용 중인 특정 터미널에서는 반복적으로 로그인할 수 있습니다. secons 명령을 다음 스위치와 함께 사용합니다.

```
# secons -d- (동시 로그인 비활성화)
# secons -d+          (동시 로그인 활성화)
```

어떤 사용자건 -d 옵션을 실행할 수 있습니다. 다른 옵션들은 ADMIN 또는 OPERATOR 특성을 가진 사용자만 사용할 수 있습니다. 동시 로그인을 비활성화하지 않으려는 사용자는 초기 스크립트에서 이 명령을 사용할 수 있습니다. 이 명령을 사용하면 사용자는 창을 여러 개 열 수는 있지만 두 번째 터미널에서는 로그인할 수 없습니다.

참고: secons -d- 명령을 사용하여 동시 로그인을 방지하는 경우, 로그아웃하기 전에 secons -d+를 사용하여 시스템에서 잠기지 않도록 해야 합니다. 동시 로그인 복원을 잊어버리고 다시 로그인을 시도하는 경우, CA Access Control 은 실행 중인 동일한 사용자 ID 를 가진 프로세스가 없을 경우에만 로그인을 허용합니다.

사용자의 동시 로그인 제한

CA Access Control 은 다음 두 가지 방법으로 동시 로그인 수를 제어할 수 있습니다.

관리자 수준

데이터베이스에서 사용자가 가질 수 있는 동시 세션 수의 시스템 수준 정의를 설정합니다. 이 방법으로 로그인한 사용자는 같은 이름으로 더 많은 로그인 세션을 차단하여 자신을 보호합니다.

사용자 수준

사용자는 자신에게 허용된 동시 로그인 수를 개별적으로 제어합니다. 로그인하면 사용자는 자신의 이름으로 다른 로그인 세션의 옵션을 차단하여 자신을 보호할 수 있습니다.

참고: 동시 로그인 수는 사용자가 특정 터미널에서 실행하는 세션 수와는 다릅니다. 한 터미널에서의 다중 세션은 단일 로그인으로 간주됩니다. 동시 로그인 제한은 사용자가 동시에 로그인할 수 있는 *터미널*의 수를 제한하는 것입니다. 각 터미널에서의 로그인 수를 제한하는 것이 아닙니다.

전역적으로 동시 로그인 제한

selang 에서 다음 명령을 입력합니다.

setoptions maxlogins(*NumLogins*)

개별적으로 동시 로그인 제한

selang 에서 다음 명령을 입력합니다.

chusr *username* maxlogins(*NumLogins*)

사용자에 대한 동시 로그인 제한 설정은 시스템 수준의 제한보다 우선 적용됩니다. CA Access Control 에서 특정 사용자에게 동시 로그인 제한을 적용하지 않도록 하려면 해당 사용자의 동시 로그인 제한을 0 으로 설정합니다. (최대 동시 로그인 수를 한 번으로 설정하는 경우 selang 을 사용할 수 없습니다).

로그인 이벤트 인식

CA Access Control 은 프로세스의 사용자 ID 를 변경하려는 시도를 모두 로그인 이벤트로 취급하지는 않습니다. 일반적으로 프로그램은 setuid 시스템 호출을 통해 사용자 ID 변경을 시도합니다. SURROGATE 클래스는 반드시 로그인 이벤트로 간주되지는 않는 이런 이벤트를 제어하는데 CA Access Control 의 관점에서 사용자 ID 를 반드시 변경하는 것은 아닙니다.

CA Access Control 은 항상 사용자가 처음 로그인할 때 사용했던 원래 사용자 ID 를 유지합니다. 일상적인 setuid 시스템 호출로는 CA Access Control 이 사용자 ID 변경 사항을 등록하지 않습니다.

CA Access Control 이 ID 변경 사항을 인식하려면 이 이벤트를 로그인 이벤트로 인식해야 합니다. 다음 규칙을 사용하는 로그인 이벤트를 인식합니다.

- ID 변경을 시도하는 프로그램은 *로그인 프로그램*으로 정의됩니다. LOGINAPPL 클래스의 모든 프로그램은 로그인 프로그램입니다.
- 프로그램은 LOGINAPPL 클래스에서 해당 정의에 따라 일련의 시스템 호출을 실행합니다.

selang 또는 CA Access Control 끝점 관리에서 관리 세션을 시작하면 CA Access Control 은 더미 로그인 이벤트를 수행합니다. 이는 실제 로그인이 아니며 CA Access Control 이 로그인 검사와 유사한 특정 내부 검사를 수행하는 것입니다.

참고: 자세한 내용은 *selang 참조 안내서*의 LOGINAPPL 클래스에 대한 SEQUENCE 속성을 참조하십시오.

관리 세션을 시작할 때 관리할 컴퓨터의 사용자 이름을 확인합니다. 사용자가 세션을 실행하는 터미널에 대한 WRITE 액세스 권한을 가지고 있는 경우에만 이 컴퓨터에 액세스하여 관리할 수 있습니다.

예를 들어 사용자가 Minerva 호스트에 로그인하여 Artemis 호스트에서 CA Access Control 을 관리하려면 다음 두 조건이 충족되어야 합니다.

- Minerva(또는 관련 이름이 완벽하게 정규화된 이름)라는 TERMINAL 개체는 Artemis 의 데이터베이스 레코드에 있습니다.
- 사용자는 이 개체의 ACL 에 WRITE 권한과 함께 나열됩니다.

다른 모든 사용자 권한 확인보다 이러한 조건 확인 작업이 먼저 수행됩니다. 데이터베이스 관리 권한도 필요합니다.

제 10 장: TCP/IP 서비스 보호

중요한 데이터를 포함하는 파일 서버에서는 TCP/IP 서비스를 보호하는 것이 가장 중요합니다. 이 서버는 호스트가 알 수 없는 컴퓨터 또는 침입자가 아닌 트러스트된 스테이션에만 특정 서비스를 제공해야 합니다.

이 섹션은 다음 항목을 포함하고 있습니다.

TCP/IP 서비스 제한

개방형 네트워크에서 모든 스테이션은 네트워크에 있는 다른 컴퓨터에 서비스를 요청할 수 있습니다. TCP/IP 프로토콜을 사용하여 많은 서비스를 제공할 수 있습니다. rlogin, rcp, rsh, ftp, telnet, rexec 등과 같은 서비스는 모든 UNIX 기반 운영 체제에 공통적인 것입니다. 나머지는 내부 및 타사 소프트웨어에서 제공됩니다.

CA Access Control 은 호스트 컴퓨터에서 TCP/IP 의 허용 프로세스를 차단하고 허용 프로그램을 정상적으로 계속할지 무시할지 여부를 결정합니다. CA Access Control 의 결정은 사용자가 정의한 호스트와 서비스를 제어하는 액세스 규칙을 따릅니다. 데이터베이스에서 TCP/IP 액세스 규칙을 작성하여 특정 컴퓨터에서 파일 전송, 원격 로그인 및 원격 셸과 같은 서비스를 받을 수 있는 컴퓨터와 네트워크를 지정할 수 있습니다.

다음 예는 원하지 않는 외부인을 효과적으로 차단하도록 TCP/IP 액세스 규칙을 정의하고 설정하는 방법을 보여 줍니다. 모든 데이터베이스를 개발할 시간이 없는 경우 데이터베이스에서 정의되지 않은 스테이션이 서비스를 받을 수 있도록 지정할 수 있습니다. 그런 경우 UACC 클래스의 HOST 레코드를 다음과 같이 설정합니다.

```
chres UACC HOST defaccess(READ)
```

로컬 호스트에서 TCP/IP 서비스에 대한 액세스 규칙을 갖게 될 스테이션은 HOST 클래스에 있는 데이터베이스의 레코드에서 정의합니다. 각 스테이션에 대해 허용된 서비스는 레코드에 나열됩니다. 예를 들어 다음과 같은 명령 시퀀스는 스테이션 ws5 에 대한 레코드를 정의하고 이 스테이션이 로컬 호스트로부터 TCP/IP 서비스 수신을 거부합니다.

```
newres HOST ws5
authorize HOST ws5 service(*) access(NONE)
```

다음 명령을 실행하면 ws5 가 로컬 컴퓨터에 대한 telnet 을 수행할 수 있습니다.

```
authorize HOST ws5 service(telnet)
```

이러한 설정을 사용하여 사용자는 로컬 컴퓨터에 대한 telnet 을 수행할 수 있으며 이를 위해 원격 사용자는 로컬 시스템을 사용하기 전에 사용자 이름과 암호를 지정해야 합니다. 스테이션이 로컬 컴퓨터에서 모든 TCP/IP 서비스를 받도록 하려면 서비스 키워드에 별표를 사용합니다. 예를 들면 다음 명령을 사용하여 ws5 는 로컬 컴퓨터로부터 TCP/IP 서비스를 호출할 수 있습니다.

```
authorize HOST ws5 service(*)
```

이 서비스는 여러 가지 방법으로 지정되는데, 그 중에는 *포트 번호*가 포함됩니다. 포트 번호는 서비스의 식별 번호입니다. 모든 서비스에는 포트 번호가 있으며 포트 번호는 /etc/services 파일의 서비스에 매핑됩니다. 다음과 같은 방법으로 서비스를 지정할 수 있습니다.

- /etc/services 파일에 정의된 이름으로 지정
- 포트 번호로 지정
- 포트 번호의 범위로 지정
- /etc/rpc 시스템 파일에 나열된 RPC 포트로 지정

예를 들어 다음 명령을 실행하면 ws5 가 포트 번호 7045 에서 7050 사이의 TCP/IP 서비스를 모두 받을 수 있습니다.

```
authorize HOST ws5 service(7045-7050)
```

대부분의 경우 각 컴퓨터에 대해 사용 권한을 결정하는 것보다 호스트 그룹을 정의하여 해당 그룹에 권한을 한 번만 설정하는 것이 훨씬 효율적입니다. CA Access Control 은 GHOST 클래스를 제공하며 각 GHOST 레코드는 호스트 그룹을 정의합니다. GHOST 레코드를 정의하고 호스트를 멤버 목록에 추가하려면 다음 명령을 입력합니다.

```
newres GHOST gh1 mem(ws2, ws3, ws5)
authorize GHOST gh1 service(ftp)
```

newres 명령은 ws2, ws3, ws5 멤버를 포함하는 gh1 이라는 호스트 그룹을 정의합니다. authorize 명령을 사용하면 세 스테이션 모두 ftp(파일 전송) 서비스를 수신할 수 있습니다.

호스트 그룹을 관리하는 것이 스테이션을 개별적으로 관리하는 것보다 쉽지만 효율성 향상을 위해 CA Access Control 은 네트워크 액세스 규칙 정의도 지원합니다. 네트워크는 HOSTNET 클래스에서 정의합니다. 예를 들어 다음과 같은 명령 세트를 생각해 봅시다.

```
newres HOSTNET hn1 mask(255.555.0.0) match(192.168.0.0)
authorize HOSTNET hn1 service(*) access(NONE)
authorize HOSTNET hn1 service(ftp)
```

- 첫 번째 줄에서 newres 명령은 hn1 이라고 하는 네트워크를 정의합니다. 마스크 값과 일치 값을 사용하여 처음 두 개의 한정자가 192.168 인 IP 주소를 가진 컴퓨터는 hn1 네트워크로부터 오는 것으로 지정합니다.
- 두 번째 및 세 번째 줄의 조합은 hn1 네트워크의 모든 스테이션이 호스트 컴퓨터에서 다른 모든 서비스를 제외하고 ftp 만 수행하도록 합니다.

TCP/IP 액세스 규칙을 정의하기 위해 CA Access Control 이 제공하는 다른 방법은 이름-패턴 액세스 규칙입니다. CA Access Control 은 와일드카드를 사용하여 HOSTNP 클래스(호스트 이름 패턴)에서 일반 레코드의 정의를 지원합니다.

참고: CA Access Control 에서 문자열 일치를 수행하는 방법에 대한 자세한 내용은 *selang 참조 안내서*를 참조하십시오.

예를 들어 다음 명령 시퀀스를 실행하면 이름이 "lin"으로 시작하고 ".org.com"으로 끝나는 모든 호스트가 로컬 호스트의 모든 TCP/IP 서비스를 받을 수 있습니다.

```
newres HOSTNP lin*.org.com
authorize HOSTNP lin*.org.com service(*).
```

참고: NIS 가 관리하는 호스트는 별칭이 아닌 NIS 맵에 나타나는 공식 이름으로 식별해야 합니다. 다음 단원에 있는 차트는 TCP/IP 검사 과정을 요약해서 보여 줍니다.

TCP 클래스 사용

또는 TCP 클래스를 사용하여 호스트 대신 서비스별로 보호를 지정할 수 있습니다.

참고: TCP 클래스에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

TCP 클래스를 사용하면 들어오는 서비스 *및* 나가는 서비스를 제어할 수 있습니다.

예를 들어 다음 명령은 READ(서비스 사용 가능을 의미)를 기본 액세스 유형으로 하는 ftp 서비스의 레코드를 만들지만 이름 패턴 PUBLIC*와 일치하는 호스트가 서비스를 받지 못하게 합니다.

```
newres TCP ftp defaccess(READ)
authorize- TCP ftp hostnp(PUBLIC*) access(N)
```

또한 특정 사용자 또는 그룹이 특정 서비스만 받도록 지정할 수 있습니다. 예를 들어 모든 사용자가 ftp 를 통해 hermes 라는 호스트에 액세스하도록 허용하고 acctng 라는 그룹의 구성원만 telnet 을 통해 hermes 에 액세스하도록 지정하려면 다음 명령을 입력합니다.

```
newres HOST hermes
newres TCP ftp owner(nobody) defaccess(read)
newres TCP telnet owner(nobody) defaccess(read)
authorize TCP ftp uid(*) host(hermes) access(write)
authorize TCP telnet gid(acctng) host(hermes) access(write)
```

참고: defaccess(read)는 나가는 서비스를 비활성화합니다. defaccess(write)는 들어오는 서비스를 비활성화합니다.

HOST 클래스가 활성화 상태인 경우(액세스에 대한 기준으로 사용) TCP 클래스는 효과적으로 활성화될 수 없습니다. 호스트(HOST) 클래스의 활성화 상태를 해제하려면 setoptions class- HOST 명령을 사용할 수 있습니다. 그런 다음 TCP 보호(TCP) 클래스를 활성화하려면 setoptions class+ TCP 명령을 사용합니다. HOST 클래스의 활성화 상태를 해제하면 GHOST, HOSTNET, HOSTNP 의 활성화 상태도 자동적으로 해제됩니다.

또한 TCP 클래스가 활성화되면 setoptions 명령 클래스- CONNECT 를 사용하여 CONNECT 클래스를 비활성화할 수 있습니다.

네트워크 차단용 스트림 모듈

기본적으로 TCP 클래스는 활성화되지 않습니다. TCP 클래스, CONNECT 클래스 또는 HOST 클래스를 활성화하기 전에 스트림 모듈을 활성화해야 합니다.

Solaris 에서 CA Access Control 스트림 모듈을 로드하려면 다음 단계를 수행합니다.

1. CA Access Control 을 중지합니다. 다음 명령을 입력합니다.

   secons -s

2. 다음 명령을 입력합니다.

   SEOS_load -s

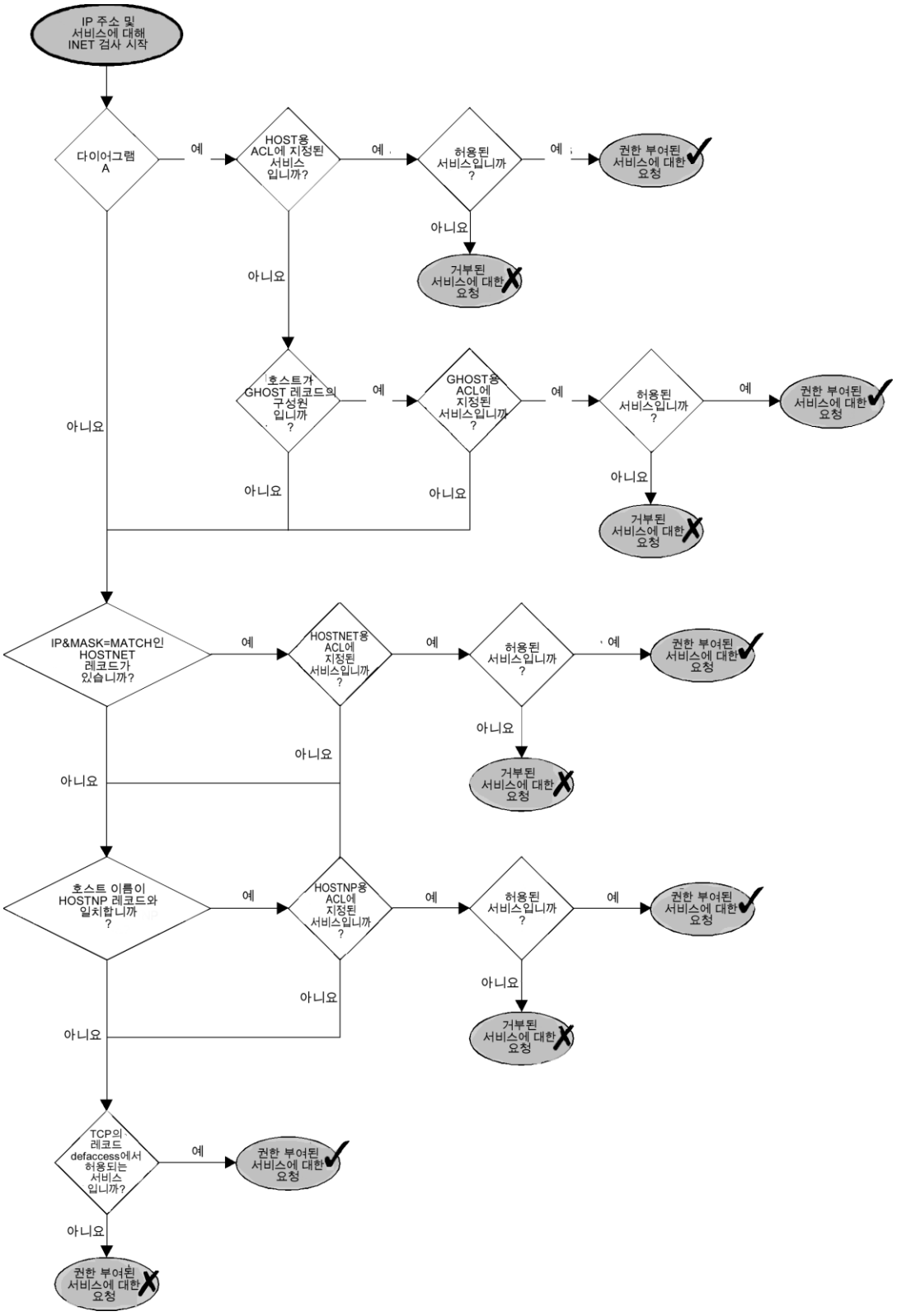
3. CA Access Control 을 시작하고 다음 명령을 입력합니다.

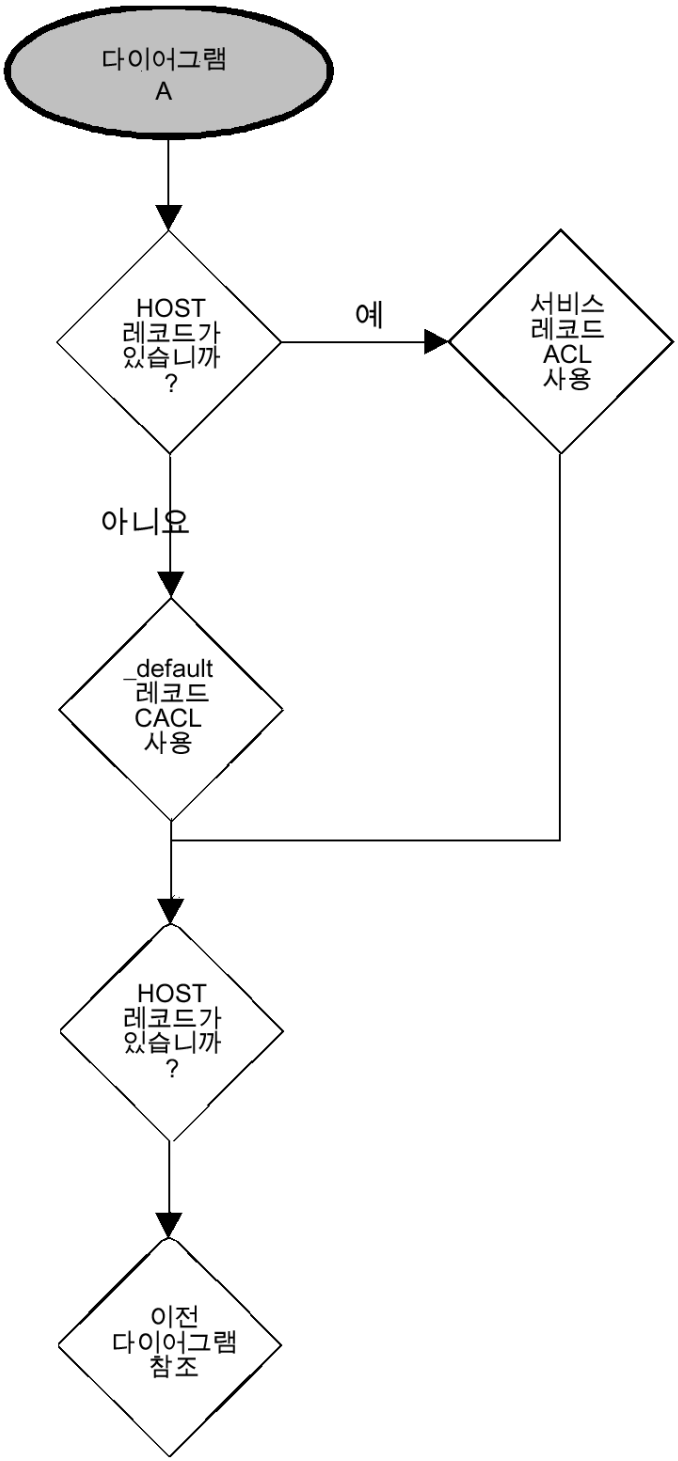
   seload

참고: 스트림 모듈이 로드되지 않은 상태에서 TCP 클래스를 활성화하려고 하면 다음과 같은 오류가 나타납니다.

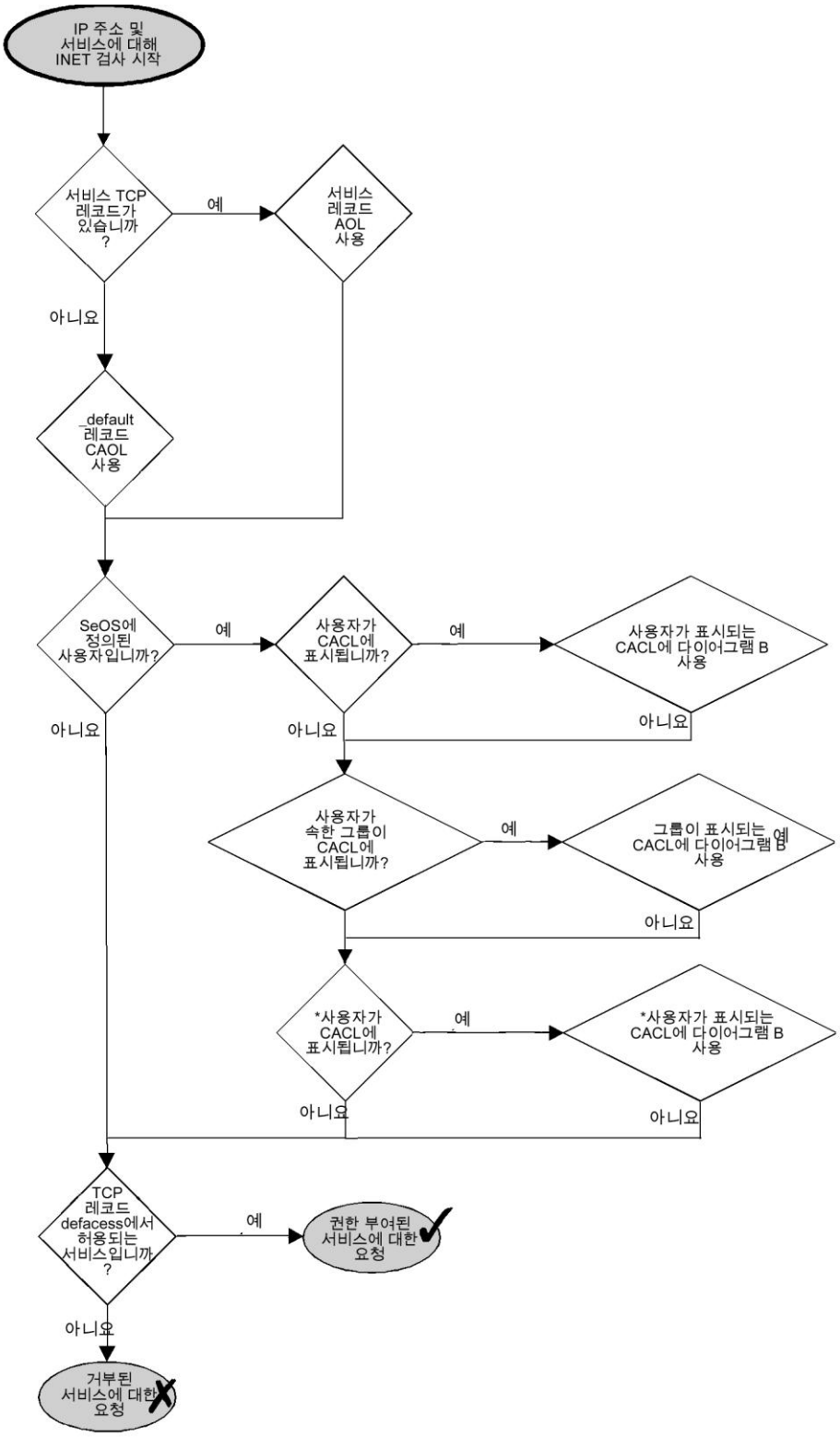
오류: 스트림이 로드되지 않으면 *className* 클래스를 활성화할 수 없습니다.
SEOS_load -s 를 사용하여 스트림을 로드합니다.

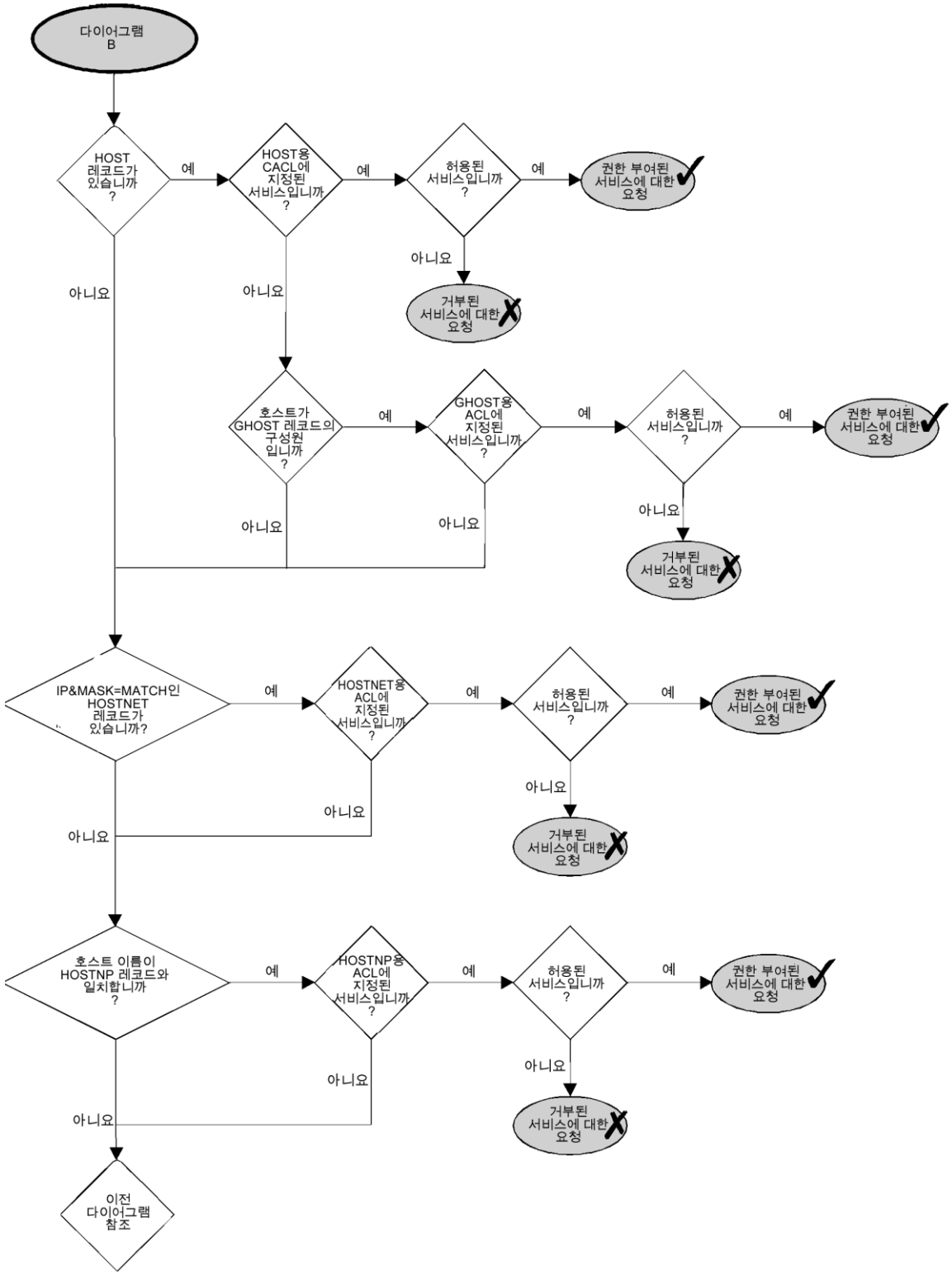
들어오는 권한 부여에 대한 알고리즘은 다음과 같습니다.


IP 주소 및 서비스에 대해 INET 검사 시작
다이어그램 A
예
HOST용 ACL에 지정된 서비스 입니까?
예
허용된 서비스입니까 ?
예
권한 부여된 서비스에 대한 요청
아니요
거부된 서비스에 대한 요청
아니요
호스트가 GHOST 레코드의 구성원 입니까 ?
예
GHOST용 ACL에 지정된 서비스입니까 ?
예
허용된 서비스입니까 ?
예
권한 부여된 서비스에 대한 요청
아니요
아니요
아니요
거부된 서비스에 대한 요청
아니요
IP&MASK=MATCH인 HOSTNET 레코드가 있습니까?
예
HOSTNET용 ACL에 지정된 서비스입니까 ?
예
허용된 서비스입니까 ?
예
권한 부여된 서비스에 대한 요청
아니요
아니요
아니요
거부된 서비스에 대한 요청
호스트 이름이 HOSTNP 레코드와 일치합니까 ?
예
HOSTNP용 ACL에 지정된 서비스입니까 ?
예
허용된 서비스입니까 ?
예
권한 부여된 서비스에 대한 요청
아니요
아니요
아니요
거부된 서비스에 대한 요청
TCP의 레코드 defaccess에서 허용되는 서비스 입니까?
예
권한 부여된 서비스에 대한 요청
아니요
권한 부여된 서비스에 대한 요청


다이어그램
A
HOST
레코드가
있습니까
?
예
서비스
레코드
ACL
사용
아니요
_default
레코드
CACL
사용
HOST
레코드가
있습니까
?
이전
다이어그램
참조

나가는 권한 부여에 대한 알고리즘은 다음과 같습니다.


IP 주소 및 서비스에 대해 INET 검사 시작
서비스 TCP 레코드가 있습니까 ?
예
서비스 레코드 AOL 사용
아니요
_default 레코드 CAOL 사용
SeOS에 정의된 사용자입니까?
예
사용자가 CACL에 표시됩니까?
예
사용자가 표시되는 CACL에 다이어그램 B 사용
아니요
아니요
사용자가 속한 그룹이 CACL에 표시됩니까?
예
그룹이 표시되는 CACL에 다이어그램 B 사용
아니요
*사용자가 CACL에 표시됩니까?
예
*사용자가 표시되는 CACL에 다이어그램 B 사용
아니요
아니요
아니요
TCP 레코드 defacess에서 허용되는 서비스입니까 ?
예
권한 부여된 서비스에 대한 요청
아니요
거부된 서비스에 대한 요청


다이어그램
B
HOST
레코드가
있습니까
?
예
HOST용
CACL에
지정된
서비스입니까
?
예
허용된
서비스입니까
?
예
권한 부여된
서비스에 대한
요청
아니요
거부된
서비스에 대한
요청
아니요
아니요
호스트가
GHOST 레코드의
구성원
입니까
?
예
GHOST용
ACL에
지정된
서비스입니까
?
예
허용된
서비스입니까
?
예
권한 부여된
서비스에 대한
요청
아니요
아니요
아니요
거부된
서비스에 대한
요청
IP&MASK=MATCH인
HOSTNET
레코드가
있습니까?
예
HOSTNET용
ACL에
지정된
서비스입니까
?
예
허용된
서비스입니까
?
예
권한 부여된
서비스에 대한
요청
아니요
아니요
거부된
서비스에 대한
요청
아니요
호스트 이름이
HOSTNP 레코드와
일치합니까
?
예
HOSTNP용
ACL에
지정된
서비스입니까
?
예
허용된
서비스입니까
?
예
권한 부여된
서비스에 대한
요청
아니요
아니요
거부된
서비스에 대한
요청
아니요
이전
다이어그램
참조

제 11 장: 정책 모델 관리

이 섹션은 다음 항목을 포함하고 있습니다.

정책 모델 데이터베이스

수십 또는 수백 개의 데이터베이스를 개별적으로 관리하는 것은 실용적이지 않습니다. CA Access Control 은 하나의 중앙 데이터베이스에서 여러 데이터베이스를 관리할 수 있는 구성 요소인 정책 모델 서비스를 제공합니다. 정책 모델 서비스를 사용하는 것은 선택사항이지만 큰 사이트에서 이 서비스를 사용하면 관리 작업이 상당히 간단해집니다.

PMD(정책 모델) 서비스에서는 PMDB(정책 모델 데이터베이스)를 사용합니다. 다른 CA Access Control 데이터베이스와는 달리 PMDB 에는 사용자, 그룹, 보호된 리소스, 리소스에 대한 액세스를 제어하는 규칙 등이 포함됩니다. 또한 PMDB 에는 *구독자* 데이터베이스 목록도 포함됩니다. 각 구독자는 별도의 컴퓨터에 있는 CA Access Control 데이터베이스이거나, 동일한 컴퓨터 또는 다른 컴퓨터에 있는 또 다른 PMDB 입니다. 구독자를 업데이트하는 PMDB 를 구독자의 *부모*라고 합니다.

PMDB 는 권한 제한 및 액세스 규칙이 유사한 여러 데이터베이스를 관리하는 데 유용한 도구입니다.

참고: PMDB 를 관리하는 방법(sepmd 유틸리티)에 대한 자세한 내용은 *참조 안내서*를 참조하십시오. selang 을 사용하여 원격으로 PMDB 를 관리하는 방법에 대한 자세한 내용은 *selang 참조 안내서*를 참조하십시오.

디스크에서 PMDB 의 위치

모든 PMDB 는 공용 디렉터리(컴퓨터당 하나)에 있습니다. 이 디렉터리의 이름은 seos.ini 파일의 [pmd] 섹션에 있는 _pmd_directory_ 토큰에 의해 지정됩니다. _pmd_directory_의 기본값은 *ACInstallDir*/policies 입니다. 여기서 *ACInstallDir* 은 CA Access Control 의 설치 디렉터리입니다(기본값: /opt/CA/AccessControl/).

각 PMDB 는 공용 디렉터리의 하위 디렉터리를 사용합니다. 하위 디렉터리의 이름은 정책 모델 이름이 됩니다. pmd.ini 파일을 비롯한 하위 디렉터리의 파일에는 정책 모델을 정의하는 데 필요한 데이터가 모두 들어 있습니다.

로컬 PMDB 관리

CA Access Control 에서는 로컬 PMDB 를 관리하기 위한 다음과 같은 몇 가지 유틸리티를 제공합니다.

sepmd

다음을 수행할 수 있는 PMDB 관리 유틸리티입니다.

- 구독자 관리
- 업데이트 파일 잘라내기
- 관리자 이중 제어
- 정책 모델 로그 파일 관리
- 기타 관리 작업 수행

sepmdadm

PMDB 를 작성하고 사용자 계층 구조 설정에 필요한 설정을 사용하여 PMDB 를 구성합니다.

참고: 정책 모델 유틸리티에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

원격 PMDB 관리

또한 CA Access Control 은 pmd 환경에서 사용할 수 있는 여러 selang 명령을 제공합니다. 다음 명령을 사용하여 PMDB 를 원격으로 관리할 수 있습니다.

backuppmd

PMDB 백업

createpmd

PMDB 를 작성합니다.

deletepmd

PMDB 를 삭제합니다.

findpmd

컴퓨터에 있는 모든 PMDB 이름을 표시합니다.

listpmd

PMDB 에 대한 다음 정보를 나열합니다.

- 구독자 및 구독자 상태
- PMDB 에 대한 설명 및 PMDB 상태
- 업데이트 파일의 명령과 해당 오프셋
- 오류 로그의 내용

pmd

다음을 수행할 수 있는 PMDB 관리 명령입니다.

- 사용할 수 없는 구독자 목록에서 구독자 제거
- 정책 모델 오류 로그 지우기
- 정책 모델 잠금 및 잠금 해제
- 정책 모델 데몬 시작 및 중지
- 업데이트 파일 잘라내기
- 초기화 파일 다시 로드

restorepmd

백업 파일로부터 PMDB 를 복원합니다.

subs

다음을 수행할 수 있는 PMDB 구독 명령입니다.

- 부모 PMDB 에 기존 구독자 추가
- 부모 PMDB 에 새 구독자 추가
- 데이터베이스(CA Access Control 또는 다른 PMDB)에 부모 PMDB 할당

subspmd

로컬 데이터베이스에 부모 PMDB 를 할당합니다.

unsubs

PMDB 에서 구독자를 제거합니다.

참고: pmd 환경에서 사용할 수 있는 selang 명령에 대한 자세한 내용은 *selang 참조 안내서*를 참조하십시오.

아키텍처 종속성

CA Access Control 을 배포할 때는 사용자 환경의 계층 구조를 고려해야 합니다. 많은 사이트에서 네트워크는 다양한 아키텍처를 포함합니다. 트러스트된 프로그램 목록과 같은 일부 정책 규칙은 아키텍처에 따라 다릅니다. 반면 대부분의 규칙은 시스템 아키텍처와 관련이 없습니다.

계층을 사용하여 두 종류 규칙 모두를 포함할 수 있습니다. 아키텍처와 관련되지 않은 규칙에 대해 전역 데이터베이스를 정의하고 이 데이터베이스에 아키텍처에 따라 달라지는 규칙을 정의하는 구독자 PMDB 를 지정할 수 있습니다.

참고: 루트 PMBD 와 모든 구독자는 사용자 환경의 실제 필요에 따라 같은 컴퓨터나 개별 컴퓨터에 있을 수 있습니다.

예: 두 개 계층으로 이루어진 배포 계층 구조

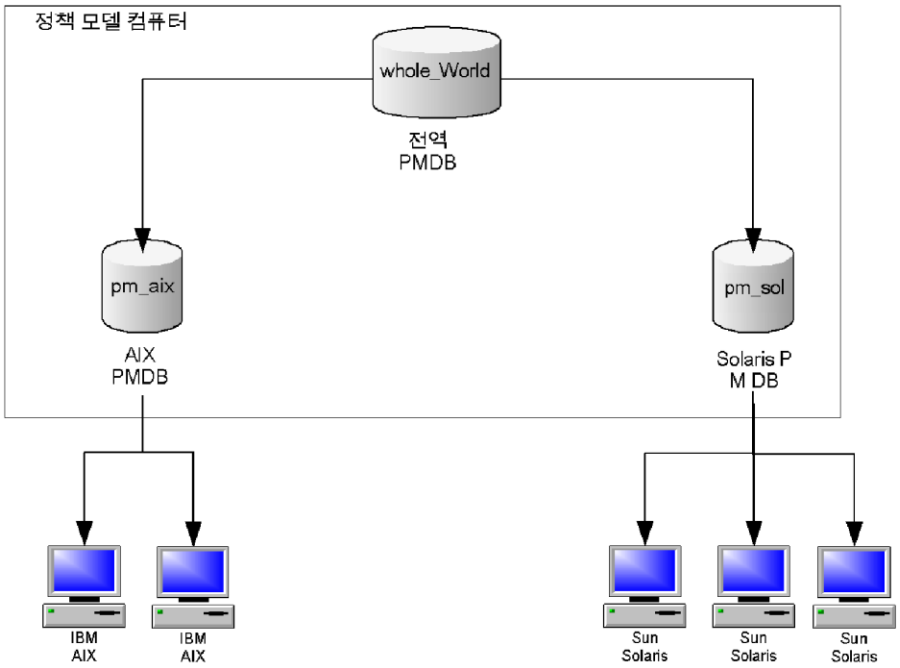
다음 UNIX 예제는 약간만 수정하면 Windows 아키텍처에도 적용됩니다.

이 예에서 사이트는 IBM AIX 및 Sun Solaris 시스템으로 구성됩니다. IBM AIX 의 트러스트된 프로그램 목록이 Sun Solaris 의 트러스트된 프로그램 목록과 다르기 때문에 PMDB 는 아키텍처 종속성을 고려해야 합니다.

다중 아키텍처 PMDB 를 설정하려면 PMDB 를 다음과 같이 설정하십시오.

1. 이름이 whole_world 인 PMDB 를 정의하여 사용자, 그룹 및 기타 아키텍처 독립적인 모든 정책을 포함합니다.
2. 이름이 pm_aix 인 PMDB 를 정의하여 IBM AIX 고유의 모든 규칙을 포함합니다.
3. 이름이 pm_sol 인 PMDB 를 Sun Solaris 고유의 모든 규칙을 포함하도록 정의합니다.

   pm_aix 및 pm_solaris PMDB 는 whole_world PMDB 의 구독자입니다. 사이트의 모든 IBM AIX 컴퓨터는 pm_aix 의 구독자입니다. 사이트의 모든 Sun Solaris 컴퓨터는 pm_sol 의 구독자입니다. 이 개념은 다음 차트에 설명되어 있습니다.



4. 사용자 추가 또는 SURROGATE 규칙 설정과 같은 플랫폼에 독립적인 명령을 whole_world 에 입력하면 사이트의 모든 데이터베이스가 자동으로 업데이트됩니다.
5. pm_aix 에 트러스트된 프로그램을 추가하면 IBM AIX 컴퓨터만 업데이트되고 Sun Solaris 시스템에는 영향을 주지 않습니다.

중앙에서 정책을 관리하기 위한 방법

CA Access Control 을 사용하면 다음 방법으로 단일 컴퓨터에서 여러 데이터베이스를 관리할 수 있습니다.

- **자동 규칙 기반 정책 업데이트** - 중앙 데이터베이스(PMDB)에서 정의한 일반 규칙은 구성된 계층의 데이터베이스에 자동으로 전파됩니다.

  참고: 이중 제어 (페이지 174)는 이 방법으로만, UNIX 에서만 사용할 수 있습니다. 자동 규칙 기반 정책 업데이트의 이중 제어에 대한 자세한 내용은 *UNIX 용 끝점 관리 안내서*를 참조하십시오. 자동 규칙 기반 정책 업데이트에 대한 자세한 내용은 *Windows 용 끝점 관리 안내서*를 참조하십시오.

- **고급 정책 관리** - 사용자가 배포하는 정책(규칙 그룹)은 호스트 또는 호스트 그룹 할당에 따라 모든 데이터베이스에 전파됩니다. 정책을 배포 취소(제거)하고 배포 상태와 배포 위반을 확인할 수도 있습니다. 이 기능을 사용하려면 추가 구성 요소를 설치하고 구성해야 합니다.

  참고: 고급 정책 관리에 대한 자세한 내용은 *엔터프라이즈 관리 안내서*를 참조하십시오.

자동 규칙 기반 정책 업데이트

중앙 데이터베이스에서 만든 단일 규칙 정책 업데이트(일반 selang 규칙)은 자동으로 구독자 데이터베이스에 전파됩니다. 동일한 데이터베이스에 여러 컴퓨터를 구독하고 데이터베이스를 서로 구독하여 계층을 작성할 수 있습니다. 설치 후 자동 규칙 기반 정책 업데이트에 맞게 환경을 구성해야 합니다.

참고: 이 정책 관리 방법에서는 계층 간에 단일 규칙 정책 업데이트만 만들 수 있습니다. 다른 기능은 고급 정책 관리 및 보고를 구현해야 사용할 수 있습니다.

자동 규칙 기반 정책 업데이트의 작동 방법

자동 규칙 정책 업데이트에 맞게 환경을 구성하면 중앙 데이터베이스에서 정의하는 각 규칙이 다음과 같은 방법으로 모든 구독자에게 자동으로 전파됩니다.

1. 하나 이상의 구독자가 있는 모든 PMDB 에 대해 규칙이 정의됩니다.
2. PMDB 에서 모든 구독자 데이터베이스에 명령을 보냅니다.
3. 구독자 데이터베이스에서 전파된 명령을 적용합니다.
   a. 구독자 데이터베이스가 응답하지 않는 경우 PMDB 는 구독자 데이터베이스가 업데이트될 때까지 일정한 간격(기본값: 30 분마다)으로 명령을 보냅니다.

      또는 구독자 컴퓨터의 seos.ini 파일에 있는 [pmd] 섹션에서 pull_option 토큰을 yes 로 설정하여 구독자 데이터베이스가 사용 가능하게 된 경우 바로 업데이트할 수 있습니다.
   b. 구독자 데이터베이스가 응답하지만 명령 적용을 거부하는 경우 PMDB 는 해당 명령을 정책 모델 오류 로그 (페이지 168)에 기록합니다.
4. 구독자 데이터베이스가 다른 구독자의 부모인 경우 해당 구독자에 명령을 보냅니다.

예: 계층에 있는 모든 컴퓨터에서 사용자 제거

rmusr 명령을 사용하여 PMDB 에서 사용자를 삭제하면 동일한 rmusr 명령이 모든 구독자 데이터베이스에 보내집니다. 이런 방법으로 하나의 rmusr 명령을 통해 여러 컴퓨터의 많은 데이터베이스에서 사용자를 제거할 수 있습니다.

PMDB 를 사용하여 구성 설정을 전파하는 방법

정책 모델의 구성을 편집하면 새 구성 값이 정책 모델의 구독자로 전파됩니다.

다음 프로세스는 구성 업데이트가 정책 모델의 구독자에게 전파되는 방법을 설명합니다.

1. 정책 모델의 구성 값을 하나 이상 편집합니다.
2. 정책 모델이 새 구성 값을 가상 구성 파일에 기록합니다.

   참고: 가상 구성 파일은 audit.cfg 파일에 대한 값을 포함하지 않습니다. 정책 모델은 이 파일에 대한 변경 내용을 가상 구성 파일에 기록하지 않습니다.
3. 정책 모델은 새 구성 값을 해당 구독자에게 보냅니다.
4. selang 명령은 각 구독자를 새 구성 값으로 업데이트합니다.

가상 구성 파일

각 정책 모델에는 해당 구독자에 대한 구성 값을 포함하는 가상 구성 파일이 있습니다. 가상 구성 파일은 PMD 디렉터리에 cfg_*configname* 이란 이름으로 위치합니다. 여기서 *configname* 은 정책 모델 구성의 이름입니다.

가상 구성 파일은 audit.cfg 파일에 있는 구성 값을 포함하지 않습니다.

새 구독자가 구성되는 방법

정책 모델은 기존 구성 값을 사용하여 각각의 새 구독자를 구성합니다. 기존 구성 값은 가상 구성 파일에 저장되어 있습니다.

참고: 가상 구성 파일은 audit.cfg 파일에 있는 구성 값을 저장하지 않습니다. 새 구독자를 만들기 전에 audit.cfg 파일에 대해 수행한 모든 변경 내용은 새 구독자에게 전파되지 않습니다.

다음 프로세스는 정책 모델이 새 구독자를 구성하는 방법을 설명합니다.

1. 정책 모델에 새 구독자를 만듭니다.
2. 정책 모델이 해당 가상 구성 파일에 있는 값을 읽습니다.
3. 정책 모델은 가상 구성 파일에 있는 구성 값을 updates.dat 파일에 추가합니다. updates.dat 파일은 또한 정책에 대한 액세스 규칙을 포함합니다.
4. 정책 모델은 updates.dat 파일을 새 구독자에게 보냅니다.
5. selang 명령은 updates.dat 파일에 있는 값을 사용하여 새 구독자를 구성합니다.

계층을 설정하는 방법

CA Access Control 은 정책 모델 서비스를 사용하여 규칙 기반 정책 업데이트를 구성된 계층에 전파합니다. 여러 CA Access Control 컴퓨터가 동일한 PMDB 를 구독하게 하고 PMDB 를 서로 구독하게 하여 계층을 작성할 수 있습니다.

자동 규칙 기반 정책 업데이트를 사용하려면 다음 작업을 수행하십시오.

1. 마스터 PMDB 를 작성하고 구성합니다 (페이지 154).
2. (선택 사항) 구독자 PMDB 를 작성하고 구성합니다 (페이지 156).
3. *끝점*이라고 하는 구독 컴퓨터에 대해 부모 PMDB 를 정의합니다 (페이지 159).

참고: 다음 절에서는 PMDB 계층을 설정하는 방법에 대해 설명합니다. PMDB 를 작성한 다음 계층을 설정하는 다른 방법이 있습니다. 정책 모델 유틸리티에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

마스터 PMDB 작성 및 구성

중앙 위치에서 정책을 관리할 수 있으려면 먼저 마스터 PMDB 를 작성하고 구성해야 합니다. 로컬 호스트에서 이 작업을 수행하려면 sepmdadm 명령을 사용할 수 있습니다.

참고: 다음 절차에서는 sepmdadm 명령의 대화형 형식을 보여 줍니다. 모든 입력에 대한 명령줄 매개 변수 사용 방법에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

마스터 PMDB 를 작성 및 구성하려면

1. 명령줄에서 다음 명령을 입력합니다.

   sepmdadm -i

   CA Access Control 에서 정책 모델 데이터베이스 관리 스크립트(sepmdadm)를 시작하고 선택할 수 있는 옵션이 있는 메뉴를 표시합니다.

2. 1 을 입력하여 첫 번째 옵션(마스터 PMDB 를 작성하고 구독자 정의)을 선택합니다.

   스크립트가 구성되고 관련 질문이 표시됩니다.

3. Enter 키를 눌러 계속합니다.

   계속해서 첫 번째 질문이 표시됩니다.

   참고: CA Access Control 이 실행되고 있지 않으면 경고가 표시되며 스크립트를 다시 실행하기 전에 CA Access Control 을 시작할 수 있습니다.

4. 작성할 정책 모델의 이름을 입력합니다.

   정책 모델 이름이 등록되고 계속 진행됩니다.

5. 지정할 첫 번째 구독자 컴퓨터의 이름을 입력합니다.

   첫 번째 구독자 이름이 등록되고 다음 구독자의 이름을 입력하라는 메시지가 표시됩니다.

6. 계속해서 원하는 수의 구독자 이름을 입력한 다음 구독자 이름을 입력하지 않고 Enter 키를 누릅니다.

   모든 구독자 이름이 등록되고 계속 진행됩니다.

   참고: 각 구독자 컴퓨터는 해당 부모 PMDB 를 가리켜야 합니다.

7. NIS, NIS+ 또는 DNS 를 실행하고 있는 경우에는 PMDB 변경 사항으로 NIS/DNS 테이블을 업데이트할지 여부를 선택합니다.

   PMDB 의 사용자 및 그룹에 대한 업데이트가 실행됩니다. 테이블은 사용자와 사용자 특성에 대한 정보를 제공합니다. 예를 선택하면 정책 모델을 통해 업데이트된 UNIX 사용자나 UNIX 그룹 역시 NIS 암호 및 그룹 파일에서 업데이트됩니다.

   a. NIS/DNS 테이블을 업데이트하려면 **y** 를 입력합니다.

      이제 NIS 암호 및 그룹 파일의 위치를 묻는 메시지가 표시됩니다.

      a. NIS 암호 파일의 전체 경로를 입력합니다.

         전체 경로가 등록되고 계속 진행됩니다.

      b. NIS 그룹 파일의 전체 경로를 입력합니다.

         전체 경로가 등록되고 계속 진행됩니다.

   b. NIS/DNS 테이블을 업데이트하려면 **n** 을 입력하거나 Enter 를 누릅니다.

      사용자 답변이 등록되고 계속 진행됩니다.

8. PMDB 에 대한 특별한 특성을 부여할 사용자를 입력합니다.

   a. 원하는 수의 CA Access Control 관리자 이름을 입력한 다음 관리자 이름을 입력하지 않고 Enter 를 누릅니다.

      관리자는 PMDB 의 속성을 변경할 권한이 있습니다.

      참고: PMDB 에서 한 명 이상의 관리자를 정의해야 합니다. *root* 가 기본값입니다.

   b. 원하는 수의 엔터프라이즈 사용자 관리자 이름을 입력한 다음 관리자 이름을 입력하지 않고 Enter 를 누릅니다.

   c. 원하는 수의 CA Access Control 감사자 이름을 입력한 다음 감사자 이름을 입력하지 않고 Enter 를 누릅니다.

      감사자는 PMDB 의 감사 로그 파일을 볼 수 있는 권한이 있습니다.

   d. 원하는 수의 엔터프라이즈 사용자 감사자 이름을 입력한 다음 감사자 이름을 입력하지 않고 Enter 를 누릅니다.

e. 원하는 수의 CA Access Control 암호 관리자 이름을 입력한 다음 암호 관리자 이름을 입력하지 않고 Enter 를 누릅니다.

f. 원하는 수의 엔터프라이즈 사용자 암호 관리자 이름을 입력한 다음 암호 관리자 이름을 입력하지 않고 Enter 를 누릅니다.

암호 관리자는 PMDB 에서 암호를 변경할 수 있는 권한이 있습니다.

사용자 답변이 등록되고 계속 진행됩니다.

9. 원하는 수의 관리 터미널을 입력한 다음 관리 터미널을 입력하지 않고 Enter 키를 누릅니다.

스크립트에서 모든 관리 터미널을 등록한 다음 사용자가 선택한 내용을 보고하고 내용이 맞는지 확인합니다.

10. 선택한 내용을 확인하려면 Enter 를 누르고 스크립트를 다시 실행하여 새로 입력하려면 **n** 을 입력합니다.

선택 내용을 확인하면 사용자가 제공한 응답에 따라 새 PMDB 가 작성됩니다.

추가 정보:

구독자 PMDB 작성 및 구성 (페이지 156)
구독 컴퓨터의 부모 PMDB 정의 (페이지 159)

구독자 PMDB 작성 및 구성

마스터 PMDB 를 구성한 후 계층 구조를 확장하려면 구독자 PMDB 를 작성하고 구성해야 합니다. 로컬 호스트에서 이 작업을 수행하려면 sepmdadm 명령을 사용할 수 있습니다.

참고: 다음 절차에서는 sepmdadm 명령의 대화형 형식을 보여 줍니다. 모든 입력에 대한 명령줄 매개 변수 사용 방법에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

구독자 PMDB 를 작성 및 구성하려면

1. 명령줄에서 다음 명령을 입력합니다.

   sepmdadm -i

   CA Access Control 에서 정책 모델 데이터베이스 관리 스크립트(sepmdadm)를 시작하고 선택할 수 있는 옵션이 있는 메뉴를 표시합니다.

2. 2 을 입력하여 두 번째 옵션(보조 PMDB 를 작성하고 구독자 및 부모 정의)을 선택합니다.

   스크립트가 구성되고 관련 질문이 표시됩니다.

3. Enter 키를 눌러 계속합니다.

   계속해서 첫 번째 질문이 표시됩니다.

   참고: CA Access Control 이 실행되고 있지 않으면 경고가 표시되며 스크립트를 다시 실행하기 전에 CA Access Control 을 시작할 수 있습니다.

4. 작성할 정책 모델의 이름을 입력합니다.

   정책 모델 이름이 등록되고 계속 진행됩니다.

5. 지정할 첫 번째 구독자 컴퓨터의 이름을 입력합니다.

   첫 번째 구독자 이름이 등록되고 다음 구독자의 이름을 입력하라는 메시지가 표시됩니다.

6. 계속해서 원하는 수의 구독자 이름을 입력한 다음 구독자 이름을 입력하지 않고 Enter 키를 누릅니다.

   모든 구독자 이름이 등록되고 계속 진행됩니다.

   참고: 각 구독자 컴퓨터는 해당 부모 PMDB 를 가리켜야 (페이지 159) 합니다.

7. 부모 PMDB 의 이름을 입력합니다.

   부모 PMDB 이름이 등록되고 계속 진행됩니다.

   참고: sepmdadm 에서는 각 구독 데이터베이스에 대해 부모를 하나만 입력할 수 있습니다. 그러나 각 데이터베이스에 대해 여러 부모를 정의할 수 있습니다. 이렇게 하려면 pmd.ini 구성 파일의 parent_pmd 토큰을 수정합니다. 이 토큰 사용에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

8. NIS, NIS+ 또는 DNS 를 실행하고 있는 경우에는 PMDB 변경 사항으로 NIS/DNS 테이블을 업데이트할지 여부를 선택합니다.

   PMDB 의 사용자 및 그룹에 대한 업데이트가 실행됩니다. 테이블은 사용자와 사용자 특성에 대한 정보를 제공합니다. 예를 선택하면 정책 모델을 통해 업데이트된 UNIX 사용자나 UNIX 그룹 역시 NIS 암호 및 그룹 파일에서 업데이트됩니다.

   a. NIS/DNS 테이블을 업데이트하려면 **y** 를 입력합니다.

      이제 NIS 암호 및 그룹 파일의 위치를 묻는 메시지가 표시됩니다.

      a. NIS 암호 파일의 전체 경로를 입력합니다.

         전체 경로가 등록되고 계속 진행됩니다.

      b. NIS 그룹 파일의 전체 경로를 입력합니다.

         전체 경로가 등록되고 계속 진행됩니다.

   b. NIS/DNS 테이블을 업데이트하려면 **n** 을 입력하거나 Enter 를 누릅니다.

      사용자 답변이 등록되고 계속 진행됩니다.

9. PMDB 에 대한 특별한 특성을 부여할 사용자를 입력합니다.

   a. 원하는 수의 CA Access Control 관리자 이름을 입력한 다음 관리자 이름을 입력하지 않고 Enter 를 누릅니다.

      관리자는 PMDB 의 속성을 변경할 권한이 있습니다.

      참고: PMDB 에서 한 명 이상의 관리자를 정의해야 합니다. *root* 가 기본값입니다.

   b. 원하는 수의 엔터프라이즈 관리자 이름을 입력한 다음 관리자 이름을 입력하지 않고 Enter 를 누릅니다.

   c. 원하는 수의 CA Access Control 감사자 이름을 입력한 다음 감사자 이름을 입력하지 않고 Enter 를 누릅니다.

      감사자는 PMDB 의 감사 로그 파일을 볼 수 있는 권한이 있습니다.

   d. 원하는 수의 엔터프라이즈 사용자 감사자 이름을 입력한 다음 감사자 이름을 입력하지 않고 Enter 를 누릅니다.

e. 원하는 수의 CA Access Control 암호 관리자 이름을 입력한 다음 암호 관리자 이름을 입력하지 않고 Enter 를 누릅니다.

   암호 관리자는 PMDB 에서 암호를 변경할 수 있는 권한이 있습니다.

f. 원하는 수의 엔터프라이즈 사용자 암호 관리자 이름을 입력한 다음 암호 관리자 이름을 입력하지 않고 Enter 를 누릅니다.

사용자 답변이 등록되고 계속 진행됩니다.

10. 원하는 수의 관리 터미널을 입력한 다음 관리 터미널을 입력하지 않고 Enter 키를 누릅니다.

    스크립트에서 모든 관리 터미널을 등록한 다음 사용자가 선택한 내용을 보고하고 내용이 맞는지 확인합니다.

11. 선택한 내용을 확인하려면 Enter 를 누르고 스크립트를 다시 실행하여 새로 입력하려면 **n** 을 입력합니다.

    선택 내용을 확인하면 사용자가 제공한 응답에 따라 새 PMDB 가 작성됩니다.

구독 컴퓨터의 부모 PMDB 정의

끝점 컴퓨터를 PMDB 에 대한 구독자로 설정하려면 PMDB 에 구독자 이름을 등록하는 것 외에 추가 작업을 수행해야 합니다. 또한 구독자 컴퓨터에서도 절차를 완료해야 합니다.

구독 컴퓨터의 부모 PMDB 를 정의하려면

1. 구독자 컴퓨터의 명령줄에서 sepmdadm 을 대화형 모드로 시작합니다.

   ```
   sepmdadm -i
   ```

   CA Access Control 에서 정책 모델 데이터베이스 관리 스크립트(sepmdadm)를 시작하고 선택할 수 있는 옵션이 있는 메뉴를 표시합니다.

2. 3 을 입력하여 세 번째 옵션(로컬 호스트의 부모 및 암호 PMDB 정의)을 선택합니다.

   스크립트가 구성되고 관련 질문이 표시됩니다.

3. Enter 키를 눌러 계속합니다.

   계속해서 대화형 스크립트에 첫 번째 질문이 표시됩니다.

   참고: CA Access Control 이 실행되고 있으면 경고가 표시되며 스크립트를 다시 실행하기 전에 CA Access Control 을 중지할 수 있습니다.

4. 부모 PMDB 의 이름을 입력합니다.

   부모 PMDB 이름이 등록되고 계속 진행됩니다.

5. 부모 암호 PMDB 의 이름을 입력합니다.

   스크립트에서 부모 암호 PMDB 이름을 등록한 다음 사용자가 선택한 내용을 보고하고 내용이 맞는지 확인합니다.

6. 선택한 내용을 확인하려면 Enter 를 누르고 스크립트를 다시 실행하여 새로 입력하려면 **n** 을 입력합니다.

   선택한 내용을 확인하면 이러한 입력을 사용하여 구독자 컴퓨터가 설정됩니다.

참고: sepmdadm 에서는 각 구독 데이터베이스에 대해 부모를 하나만 입력할 수 있습니다. 그러나 각 데이터베이스에 대해 여러 부모를 정의할 수 있습니다. 이렇게 하려면 seos.ini 구성 파일의 parent_pmd 토큰을 수정합니다. 이 토큰 사용에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

UID/GID 동기화

관리자는 사용자의 경우 UID 기준으로 참조하고 그룹의 경우 GID 기준으로 참조하는 메시지를 받을 수 있습니다. UID 및 GID 는 어디에서나 동일한 의미를 가져야 합니다.

기본적으로 PMDB 는 모든 위치에서 새 사용자와 그룹에 대한 동일한 UID 및 GID 를 사용하려고 시도하지만 시작 시 사용자가 필요한 조건을 제공할 수 있습니다. 동일한 passwd 파일과 group 파일로 시작하도록 pmd.ini 파일의 synch_uid 토큰을 yes 로 설정합니다. 로컬 데이터베이스가 PMDB 의 구독자이고 해당 PMDB 가 구독자 데이터베이스에 대한 새 사용자 및 새 그룹의 유일한 소스인 경우 로컬 데이터베이스, PMDB 및 PMDB 구독자에 대한 UID 사이의 호환성과 GID 사이의 호환성을 신뢰할 수 있습니다.

PMDB 또는 다른 구독자 컴퓨터에서 이미 사용 중인 UID 를 사용하여 새 사용자를 작성한 경우 해당 구독자의 개별 업데이트는 실패하지만 이러한 충돌이 없는 다른 모든 구독자 컴퓨터에서는 업데이트가 성공적으로 수행됩니다.

암호 및 그룹 파일을 동기화하는 다른 방법은 새로운 각 사용자의 UID 와 새로운 각 그룹의 GID 를 명시적으로 지정하는 것입니다.

사용자 및 그룹 동기화

다양한 데이터베이스의 사용자 및 그룹 목록이 항상 올바르게 일치하게 하려면 동일한 초기 목록 집합이 필요합니다. 암호 파일과 그룹 파일은 매우 중요하기 때문에 로컬 사용자 및 그룹 정보를 누적하기 전에 동기화해야 합니다.

사용자 및 그룹을 동기화하려면

1. /etc/passwd 파일 및 /etc/group 파일을 정책 모델 디렉터리로 복사합니다.

   정책 모델 디렉터리 (페이지 146)에 있는 이전 passwd 및 group 파일을 삭제하는 일회성 절차입니다.

   참고: 섀도 파일을 사용 중인 경우 암호를 동기화하려면 secrepsw 유틸리티를 사용하는 것이 좋습니다. 자세한 내용은 *Reference Guide*(참조 안내서)를 참조하십시오.

2. /etc/passwd 파일과 /etc/group 파일을 각 구독자 컴퓨터에 복사하여 사용자 스테이션의 해당 파일과 동일하게 만듭니다.

3. PMDB 가 있는 컴퓨터에서 pmd.ini 파일의 synch_uid 토큰이 yes 로 설정되어 있는지 확인합니다.

   기본적으로 synch_uid 토큰의 값은 yes 입니다. 하지만 구독자 데이터베이스가 독자적인 기본 UID 및 GID 를 갖게 하려면(즉, PMDB 의 해당 ID 와 일치할 필요가 없음) synch_uid 를 no 로 설정할 수 있습니다.

UID 를 명시적으로 지정

PMDB 와 해당 PMDB 구독자 모두에게 동일한 UID 또는 GID 를 보내는 다른 방법은 새 사용자를 작성할 때 이를 명시적으로 설정하는 것입니다.

UID 를 명시적으로 지정하려면 각 newusr 명령에 userid 또는 groupid 매개변수를 사용합니다.

예: 지정한 UID 를 사용하여 새 사용자 작성

새 사용자 terry_jones 의 UID 를 1234 로 명시적으로
설정하려면(데이터베이스에 해당 UID 를 사용하는 사용자가 없다고 가정)
다음 명령을 입력합니다.

```
newusr terry_jones unix (userid(1234))
```

지정한 UID 가 PMDB 에서 이미 사용 중인 경우 PMDB 는 자동으로
업데이트되지 않지만 명령은 계속 다른 구독자 데이터베이스에 전파됩니다.
다른 데이터베이스 중에 특정 UID 를 사용 중인 모든 데이터베이스에서
구독자의 개별 업데이트가 실패하지만 이러한 충돌이 없는
데이터베이스에서는 업데이트가 성공합니다.

정책 모델에서 구독자를 업데이트하는 방법

구독자를 업데이트할 때 정책 모델에서는 다음 작업을 수행합니다.

1. 정책 모델은 구독자 이름이 정책 모델에서 추가되거나 삭제될 때 구독자 이름을 정규화하려고 시도합니다.
2. PMDB 데몬인 sepmdd 는 _QD_timeout_ 토큰에 정의된 시간 동안 구독자 데이터베이스 업데이트를 시도합니다.
3. 최대 시간이 경과하고 데몬이 구독자를 업데이트할 수 없는 경우 해당 구독자를 건너뛰고 목록의 나머지 구독자를 업데이트합니다.
4. 구독자 목록의 최초 검사를 완료하면 sepmdd 는 두번째 검사를 수행합니다. 여기서는 첫번째 검사에서 업데이트에 실패한 구독자를 업데이트하려고 시도합니다. 두번째 검사 중에는 연결 시스템 호출 시간(약 90 초)이 만료될 때까지 구독자를 업데이트합니다.

참고: _QD_timeout_ 토큰은 seos.ini 및 pmd.ini 파일에 둘 다 있을 수 있습니다. 토큰이 두 파일 모두에 있는 경우 sepmdd 는 pmd.ini 파일의 값을 사용합니다.

참고: PMDB 에서 구독자에게 업데이트를 전파하는 동안 오류가 발생하면 sepmdd 데몬은 정책 모델 오류 로그 파일 (페이지 168)에 항목을 작성합니다. 기본적으로 이 ERROR_LOG 파일은 PMDB 디렉터리 (페이지 146)에 있습니다.

정책 모델 데이터베이스 업데이트

PMDB 가 있는 컴퓨터에서 작업할 때 PMDB 가 자동으로 업데이트되지는 않습니다. PMDB 를 업데이트하려면 해당 PMDB 를 대상 데이터베이스로 지정해야 합니다.

대상 데이터베이스를 지정하려면 selang 명령 셸에서 hosts 명령을 사용합니다.

hosts *pmd_name@pmd_host*

이제 모든 selang 명령으로 지정한 정책 모델 데이터베이스가 업데이트됩니다. 그런 다음 명령이 이 컴퓨터와 모든 구독자 컴퓨터의 활성 데이터베이스에 자동으로 전파됩니다.

예: 대상 PMDB 지정

myPMD_host 에서 대상 데이터베이스를 policy1 로 설정하려면 다음 명령을 사용합니다.

hosts policy1@myPMD_host

이제 newusr 명령을 입력하면 새 사용자가 이 컴퓨터와 모든 구독자 컴퓨터의 활성 데이터베이스뿐 아니라 policy1 데이터베이스에도 추가됩니다.

업데이트 파일 정리

sepmd 유틸리티는 pmd.ini 파일에 받은 각 업데이트를 자동으로 씁니다. 이 파일이 지나치게 커지지 않게 하려면 파일에서 처리된 업데이트를 정기적으로 삭제하는 것이 좋습니다.

업데이트 파일을 정리하려면 다음 명령을 사용합니다.

sepmd -t *pmdbName* auto

sepmd 는 전파되지 않은 첫 번째 업데이트 항목의 오프셋을 계산하여 그 이전의 모든 업데이트 항목을 삭제합니다.

참고: sepmd 유틸리티에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

업데이트 파일 암호화

PMDB 를 생성한 후 sepmdd 를 시작하기 *전에* updates.dat 파일에 저장된 정보가 암호화되도록 지정할 수 있습니다.

업데이트 파일을 암호화하려면 pmd.ini 파일의 [pmd] 섹션에서 UseEncryption 토큰을 yes 로 설정합니다.

updates.dat 파일의 암호를 해독하려면 -de 스위치와 함께 sepmd 유틸리티를 사용합니다.

참고: sepmd 에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

구독자 제외

구독자가 부모 PMDB 에서 업데이트를 받지 못하도록 구독자를 건너뛸 수 있습니다.

로컬 호스트를 제외하려면 pmd.ini 파일에서 exclude_localhost 토큰을 yes 로 설정합니다.

제외 목록에 구독자를 추가하려면 exclude_file 토큰(파일 이름)을 설정합니다.

구독자가 업데이트를 받게 하려면 구독자를 제외 목록에서 제거합니다.

암호 전파

사용자가 sepass 유틸리티를 사용하여 암호를 변경하면 새 암호는 일반적으로 컴퓨터의 부모 PMDB 로 전달됩니다. 부모 PMDB 는 seos.ini 파일의 [seos] 섹션에 있는 parent_pmd 또는 passwd_pmd 토큰에 정의됩니다. 그러나 사용자가 sepass 유틸리티를 사용하여 암호를 변경하는 경우 해당 사용자의 새 암호를 별도의 PMDB 로 보내 해당 PMDB 가 전파하도록 지정할 수도 있습니다.

새 사용자의 암호를 별도의 PMDB 로 보내려면 newusr, chusr 또는 editusr 명령과 함께 pmdb 매개 변수를 사용합니다.

예: 암호 전파를 위해 별도의 PMDB 지정

sepass 를 사용하여 만든 사용자 Tony 의 새 암호를 별도의 PMDB pw_pmdb@name1.yourorg.com 으로 보내 해당 PMDB 가 전파하도록 지정하려면 다음 명령을 입력합니다.

editusr tony pmdb(pw_pmdb@name1.yourorg.com)

구독자 제거

특정 구독자에 더 이상 업데이트를 전파하지 않으려면 해당 구독자를 제거해야 합니다. 또는 업데이트를 받지 못하도록 구독자를 제외 (페이지 164)할 수 있습니다.

구독자를 제거하려면

1. 구독자 목록에서 컴퓨터를 제거합니다.

   sepmd -u *PMDB_name computer_name*

   컴퓨터가 정책 모델 구독 목록에서 제거됩니다.

2. 구독 목록에서 제거한 컴퓨터에서 seosd 를 종료합니다.

   secons -s

   seosd 데몬이 종료합니다.

3. 구독 목록에서 제거한 컴퓨터의 seos.ini 파일에 있는 [seos] 섹션에서 parent_pmd 토큰 값을 삭제합니다.

   컴퓨터에서 더 이상 부모 PMDB 의 업데이트를 받지 않습니다.

4. seosd 를 다시 시작합니다.

   구독 목록에서 제거한 컴퓨터의 활성 데이터베이스는 더 이상 지정한 PMDB 의 구독자가 아닙니다.

참고: 데이터베이스가 PMDB 에서 구독 취소된 경우 PMDB 는 더 이상 명령을 보내지 않습니다.

업데이트 필터링

PMDB 를 통해 다른 구독자 데이터베이스에 있는 다른 데이터 하위 집합을 업데이트하려면 구독자 데이터베이스로 보낼 레코드를 정의해야 합니다.

업데이트를 필터링하려면

1. PMDB 가 구독자의 하위 집합에 대한 부모 역할을 하도록 구성합니다 (페이지 159).
2. 부모 PMDB 의 pmd.ini 파일에 있는 *filter* 토큰을 수정하여 같은 컴퓨터에서 설정한 필터 파일을 가리킵니다.

   그러면 구독자 데이터베이스에 대한 업데이트가 해당 필터를 통과하는 레코드로 제한됩니다.

참고: 기본 UNIX 환경에서 join 또는 join- selang 명령을 실행하면 CA Access Control 이 해당 명령을 그룹 변경(cg)으로 변경합니다. 기본 UNIX 환경에서 join 또는 join- 명령을 필터링하려면 필터 파일에 다음 줄을 사용합니다.

MODIFY UNIX GROUP *GroupName* USERS NOPASS

기본 UNIX 환경에서 사용자 이름을 기준으로 join 또는 join- 명령을 필터링할 수는 없습니다. 이 규칙은 다른 환경의 join 또는 join- 명령에는 적용되지 않습니다.

정책 모델 필터 파일

필터 파일은 각각 6 개의 필드가 있는 줄로 구성됩니다. 이 필드에는 다음에 대한 정보가 포함되어 있습니다.

- 허용되거나 거부된 액세스 형식.

  예: READ 또는 MODIFY
- 영향 받는 환경. 예:

  예: AC 또는 기본
- 레코드 클래스.

  예: USER 또는 TERMINAL
- 규칙이 적용되는 클래스 내 개체.

  예를 들어 User1, AuditGroup, 또는 TTY1 입니다.

- 레코드가 허용하거나 취소한 속성.

  예를 들어 필터 줄의 OWNER 와 FULL_NAME 은 이러한 속성을 갖는 명령이 필터링된다는 것을 나타냅니다. *참조 안내서*에 나타난 대로 각 속성을 정확히 입력해야 합니다.

- 레코드를 구독자 데이터베이스로 전달해야 하는지 여부.

  PASS 또는 NOPASS

필터 파일의 각 줄에 다음 규칙이 적용됩니다.

- 필드에 별표(*)를 사용하여 가능한 모든 값을 표시할 수 있습니다.
- 두 개 이상의 행이 동일한 레코드를 포함할 경우, *첫 번째* 적용 가능한 행이 사용됩니다.
- 필드는 공백으로 구분합니다.
- 필드에 여러 개의 값이 있는 경우 세미콜론으로 값을 구분합니다.
- #으로 시작하는 줄은 설명 줄로 간주됩니다.
- 빈 줄은 허용되지 않습니다.

예: 필터 파일

다음 예에서는 필터 파일의 행을 설명합니다.

CREATE	AC	USER	*	FULL_NAME;OBJ_TYPE	NOPASS
액세스 형식	환경	class	레코드 이름 (* =모두)	properties	처리

이 예에서 이 줄이 있는 파일의 이름을 TTY1_FILTER 로 지정하고 pmd.ini 파일에서 filter=/opt/CA/AccessControl//TTY1_FILTER 가 되도록 PMDB TTY1 를 편집하면 PMDB TTY1 은 FULL_NAME 및 OBJ_TYPE 속성을 사용하여 새 사용자를 작성하는 레코드를 구독자에게 전파하지 않습니다.

정책 모델 오류 로그 파일

정책 모델 오류 로그는 시간순으로 구성되며 다음과 비슷합니다.

오류 텍스트	오류 범주
20 Nov 03 11:56:07 (pmdb1): fargo nu u5 0 Retry 오류: 로그인 절차가 실패했습니다.(10068) 오류: 상위가 아닌 PMDB에서 업데이트할 수 없습니다. (pmdb1@name.company.com) (10104)	구성 오류
20 Nov 03 19:53:17 (pmdb1): fargo nu u5 0 Retry 오류: 연결하지 못했습니다.(10071) 호스트에 연결할 수 없습니다.(12296)	연결 오류
20 Nov 03 11:57:06 (pmdb1): fargo nu u5 560 Cont 오류: 사용자 u5를 작성하지 못했습니다.(10028) 이미 있음(-9) 20 Nov 03 11:57:06 (pmdb1): fargo nu u5 1120 Cont 오류: 사용자 u5를 작성하지 못했습니다.(10028) 이미 있음(-9)	데이터베이스 업데이트 오류

정책 모델 오류 로그는 이진 형식이기 때문에 다음 명령을 입력해야만 볼 수 있습니다.

ACInstallDir/bin sepmd -e *pmdname*

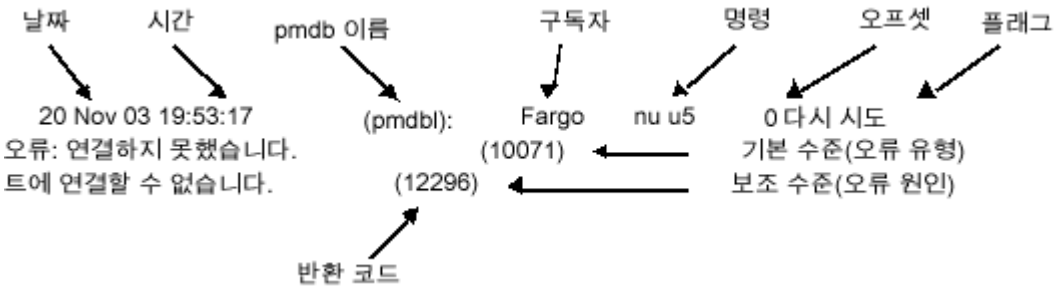
참고: rm 등과 같은 UNIX 명령을 사용하여 오류 로그를 수동으로 삭제하지 마십시오. 로그를 삭제하려면 다음 명령만 사용해야 합니다.

ACInstallDir/bin sepmd -c *pmdname*

중요! CA Access Control r5.1 이상 버전의 오류 로그 형식은 이전 버전 형식과 호환되지 않습니다. sepmd 는 이러한 이전 버전의 오류 로그를 처리할 수 없습니다. 이 형식의 버전으로 업그레이드하는 경우 이전의 오류 로그는 ERROR_LOG.bak 으로 복사되고 sepmdd 를 시작할 때 새 로그 파일이 생성됩니다.

예: PMDB 업데이트 오류 메시지

다음은 일반적인 오류 메시지의 예입니다.



- 맨 위 행은 항상 날짜, 시간 및 구독자로 구성됩니다. 그런 다음 오류를 생성한 명령과 오프셋(10 진수 형식)을 차례로 표시하여 업데이트 파일 내에서 실패한 업데이트의 위치를 나타냅니다. 마지막으로 PMDB 에서 업데이트를 자동으로 다시 시도할지 여부를 나타내는 플래그가 표시됩니다.
- 두 번째 행에는 기본 수준 메시지(발생 오류 유형) 및 해당 반환 코드의 예가 표시됩니다.
- 세 번째 행에는 보조 수준 메시지(오류가 발생한 이유)와 반환 코드의 예가 표시됩니다.

예: 오류 메시지

명령은 두 개 이상의 오류를 생성하고 표시할 수도 있습니다. 또한 하나의 오류가 기본 수준 메시지 및/또는 보조 수준 메시지로 구성될 수도 있습니다.

다음 오류는 하나의 메시지 수준만 가집니다.

Fri Dec 29 10:30:43 2003 CIMV_PROD:해제하지 못했습니다. 반환 코드 = 9241

이 메시지는 sepmd pull 이 이미 사용 가능한 구독자를 해제하려고 할 때 표시됩니다.

정책 모델 백업

PMDB 를 백업할 때는 정책 모델 데이터베이스에 있는 데이터를 다른 디렉터리로 복사합니다. 여기에는 다음이 포함됩니다.

- 정책 정보
- 정책 모델 구독자의 목록
- 구성 설정
- 레지스트리 항목
- updates.dat 파일

다른 플랫폼, 운영 체제, CA Access Control 버전을 사용하는 백업 파일로부터 PMDB 를 복원할 수 없습니다. 정책 모델은 반드시 동일한 플랫폼, 운영 체제, CA Access Control 버전을 실행하는 호스트에 백업해야 합니다.

sepmd 를 사용하여 PMDB 백업

PMDB 를 백업할 때 정책 모델 데이터베이스의 데이터를 지정된 디렉터리로 복사합니다. 백업된 PMDB 파일은 가급적 CA Access Control 액세스 규칙에 의해 보호되는 안전한 곳에 보관해야 합니다.

sepmd 유틸리티를 사용하여 로컬 호스트의 PMDB 를 백업할 수 있습니다. selang 명령을 사용하여 원격 호스트의 PMDB 를 백업할 수도 있습니다.

참고: PMDB 를 재귀적으로 백업할 수 있습니다. 재귀적 백업은 계층에 있는 모든 PMDB 를 지정하는 호스트에 백업하고, 백업이 호스트로 이동되었을 때 구독이 계속 유효하도록 PMDB 구독자를 수정합니다. 동일한 호스트에 마스터 및 자식 PMDB 가 배포된 경우에는 재귀적 백업만 사용할 수 있습니다.

sepmd 를 사용하여 PMDB 를 백업하려면

1. 다음 명령을 사용하여 PMDB 를 잠급니다.

   sepmd -bl *pmdb_name*

   PMDB 가 잠기며, 이제 구독자에게 어떤 명령도 보낼 수 없습니다.

2. 다음 작업 중 하나를 수행합니다.

   - 다음 명령을 사용하여 PMDB 를 백업합니다.

     sepmd -bh *pmdb_name* [*destination_directory*]

   - 다음 명령을 사용하여 PMDB 를 재귀적으로 백업합니다.

     sepmd -bh *pmdb_name* [*destination_directory*] [*backup_host_name*]

   참고: 대상 디렉터리를 지정하지 않으면 백업이 다음 디렉터리에 지정됩니다.

   ACInstallDir/data/policies_backup/pmdb_name

3. 다음 명령을 사용하여 PMDB 를 잠금 해제합니다.

   sepmd -ul *pmdb_name*

   PMDB 가 잠금 해제되며, 이제 구독자에게 명령을 보낼 수 있습니다.

selang 을 사용하여 PMDB 백업

PMDB 를 백업할 때 정책 모델 데이터베이스의 데이터를 지정된 디렉터리로 복사합니다. 백업된 PMDB 파일은 가급적 CA Access Control 액세스 규칙에 의해 보호되는 안전한 곳에 보관해야 합니다.

selang 명령을 사용하여 로컬 또는 원격 호스트의 PMDB 를 백업할 수 있습니다. sepmd 유틸리티를 사용하여 로컬 호스트에서 PMDB 를 백업할 수도 있습니다.

참고: PMDB 를 재귀적으로 백업할 수 있습니다. 재귀적 백업은 계층에 있는 모든 PMDB 를 지정하는 호스트에 백업하고, 백업이 호스트로 이동되었을 때 구독이 계속 유효하도록 PMDB 구독자를 수정합니다. 동일한 호스트에 마스터 및 자식 PMDB 가 배포된 경우에는 재귀적 백업만 사용할 수 있습니다.

selang 을 사용하여 PMDB 를 백업하려면

1. (선택 사항) selang 을 사용하여 원격 호스트로부터 PMDB 에 연결하려는 경우 다음 명령으로 PMDB 호스트에 연결합니다.

   host *pmdb_host_name*

2. 다음 명령을 사용하여 PMD 환경으로 이동합니다.

   env pmd

3. 다음 명령을 사용하여 DMS 를 잠급니다.

   pmd *pmdb_name* lock

   PMDB 가 잠기며, 이제 구독자에게 어떤 명령도 보낼 수 없습니다.

4. 다음 명령을 사용하여 DMS 데이터베이스를 백업합니다.

   backuppmd *pmdb_name* [destination(*destination_directory*)] [hir_host(*host_name*)]

   참고: 대상 디렉터리를 지정하지 않으면 백업이 다음 디렉터리에 지정됩니다.

   ACInstallDir/data/policies_backup/pmdbName

5. 다음 명령을 사용하여 PMDB 를 잠금 해제합니다.

   pmd *pmdb_name* unlock

   PMDB 가 잠금 해제되며, 이제 구독자에게 명령을 보낼 수 있습니다.

정책 모델 복원

정책 모델이 복원되면 CA Access Control 은 백업 PMDB 파일을 지정된 디렉터리에 복사합니다. 다음을 포함하여 원래 PMDB 파일에 있던 모든 항목이 새 PMDB 디렉터리에 복사됩니다.

- 정책 정보
- 정책 모델 구독자의 목록
- 구성 설정
- 레지스트리 항목
- updates.dat 파일

대상 디렉터리에 기존 PMBD 가 있는 경우 CA Access Control 은 복원 파일을 대상 디렉터리에 복사하기 전에 기존 파일을 삭제합니다.

다른 플랫폼, 운영 체제, CA Access Control 버전을 사용하는 백업 파일로부터 PMDB 를 복원할 수 없습니다. 정책 모델은 반드시 동일한 플랫폼, 운영 체제, CA Access Control 버전을 실행하는 호스트에 백업해야 합니다.

PMDB 복원

PMDB 를 복원할 때 CA Access Control 은 PMDB 백업 파일의 데이터를 사용자가 지정한 디렉터리로 복사합니다. 복원을 수행하는 터미널에서 CA Access Control 이 실행 중이어야 합니다.

참고: 다른 터미널에서 PMDB 를 백업 및 복원하는 경우 PMDB 는 복원된 PMDB 데이터베이스에서 터미널 리소스를 자동으로 업데이트하지 않습니다. 복원된 PMDB 에 새 터미널 리소스를 추가해야 합니다. 새 터미널 리소스를 추가하려면 복원된 PMDB 를 중지하고, *selang -p pmdb* 명령을 실행한 다음, 복원된 PMDB 를 시작하십시오.

PMDB 를 복원하려면 PMDB 를 복원할 터미널에서 다음 중 *하나*를 실행하십시오.

- sepmd -restore 유틸리티
- selang restore pmd 명령

참고: 유틸리티에 대한 자세한 내용은 *참조 안내서*를 참조하십시오. selang 명령에 대한 자세한 내용은 *selang 참조 안내서*를 참조하십시오.

이중 제어

이중 제어는 PMDB 업데이트 과정을 다음 두 단계로 나누는 작업 방법입니다.

- 하나 이상의 명령으로 구성된 트랜잭션 작성.

  ADMIN 특성을 가진 사용자인 *maker* 가 PMDB 를 업데이트하는 하나 이상의 명령을 입력합니다. 트랜잭션은 고유한 ID 번호를 받고 실행되기 전에 파일에서 처리되기를 기다립니다.

- 트랜잭션에 실행을 위한 권한 부여.

  ADMIN 특성을 가진 *다른*사용자인 *checker* 가 트랜잭션 중에 명령을 잠그고 명령을 확인하며 이 명령에 권한을 부여하거나 거부합니다. 트랜잭션에 권한이 부여되면 PMDB 에서 명령이 실행됩니다. 트랜잭션이 거부되면 트랜잭션은 삭제되고 PMDB 는 업데이트되지 않습니다. Checker 는 트랜잭션 중에 일부 명령에 권한을 부여할 수 없으며 다른 일부 권한을 거부할 수 없습니다. 트랜잭션은 전체로 처리되어야 합니다.

  참고: find 및 show 명령만 checker 의 권한 부여가 필요하지 않습니다.

sepmd 유틸리티의 매개 변수를 사용하면 maker 는 처리되지 않은 트랜잭션을 나열, 검색, 편집 또는 삭제할 수 있습니다. 또한 checker 는 트랜잭션을 잠가 트랜잭션에 권한을 부여하거나 거부할 수 있고 트랜잭션을 나중에 처리하거나 다른 checker 가 처리하도록 트랜잭션의 잠금을 해제할 수도 있습니다.

sepmdd 데몬이 start_transaction 명령을 수신하면 고유 번호를 자식 프로세스에게 전달합니다. 자식 프로세스는 이 식별 번호를 이용하여 추가 명령에 태그를 붙이고 이 번호는 새 트랜잭션에 추가되어 sepmdd 데몬의 메모리에 보관됩니다. sepmdd 가 end_transaction 명령을 수신하면 권한 부여 알고리즘이 실행됩니다. 권한 부여 알고리즘은 트랜잭션 명령이 트랜잭션의 Maker 에 속해 있는지, 실행에 앞서 처리 대기 중인 다른 트랜잭션에 의해 명령 객체가 잠겨 있지 않은지를 검사합니다.

트랜잭션이 처리되기 전에는 동일한 객체를 다른 트랜잭션에서 사용할 수 없습니다. 검사를 통과하면 관련 객체는 잠기고 트랜잭션은 고유한 일련 번호를 할당 받으며 데이터는 파일에 저장됩니다. 각각의 트랜잭션은 서로 다른 파일에 저장됩니다.

참고: sepmd 유틸리티나 sepmdd 데몬에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

이중 제어 활성화

이중 제어를 사용하면 PMDB 를 업데이트하는 책임을 maker 와 checker 두 사람에게 분담할 수 있습니다.

이중 제어를 활성화하려면 pmd.ini 파일의 is_maker_checker 토큰 *및* seos.ini 파일의 [pmd] 섹션을 yes 로 설정합니다.

```
is_maker_checker=yes
```

참고: 이 토큰 값을 설정하기 *전에* 정책 모델 maker 를 작성합니다.

트랜잭션 작성 또는 편집

이중 제어를 활성화한 경우 maker 가 트랜잭션을 작성하고 checker 가 이러한 트랜잭션을 처리해야 합니다.

트랜잭션을 작성하려면

1. 다음 조건이 맞는지 확인합니다.
   - maker 에게 ADMIN 권한이 있습니다.
   - 사용자와 관련된 명령이 없습니다. 즉, 자신을 변경하는 명령을 입력할 수 없습니다.
   - 명령의 개체 중에 어느 것도 Checker 가 아직 처리하지 않은 다른 트랜잭션의 일부가 아닙니다.
   - 명령의 모든 개체가 있습니다.
   - 다른 maker 가 호출한 기존 트랜잭션을 편집하고 있지 않습니다. 소유한 트랜잭션만 편집할 수 있습니다.
2. maker PMDB 에 연결합니다.

   ```
   hosts maker@
   ```

   호스트 명령은 사용자와 PMDB(Maker)를 연결해 줍니다. 이중 제어가 활성화되면 PMDB 의 이름은 항상 "maker"입니다. hosts 명령을 입력하면 호스트 연결의 성공 여부를 나타내는 메시지가 보고됩니다.
3. 트랜잭션을 시작합니다.

   ```
   start_transaction transactionName
   ```

   트랜잭션을 입력하거나 업데이트할 때 start_transaction 명령을 첫 번째 단계로 사용합니다. 트랜잭션을 설명하거나 트랜잭션에 최대 256 자의 영숫자로 원하는 이름을 지정할 수 있습니다.

4. 트랜잭션을 입력합니다.

   명령의 목록입니다. 예:

   ```
   newusr mary owner(bob) audit(failure,loginfailure)
   chres TERMINAL tty30 defaccess(read) \
   restrictions(days(weekdays) time(0800:1800))
   ```

5. 트랜잭션을 종료합니다.

   ```
   end_transaction
   ```

   트랜잭션이 완료되었습니다. 사용자는 트랜잭션에 할당된 고유 ID 번호를 부여 받습니다. 명령은 파일에 위치합니다. 이 파일에서 사용자는 처리 준비 중인 Checker 가 명령을 잠글 때까지 파일에 액세스하거나 파일을 변경할 수 있습니다.

   참고: 트랜잭션을 나중에 편집하려면 트랜잭션 ID 번호를 기록해 둡니다.

트랜잭션을 편집하려면

- end_transaction 명령을 입력하면 ID 번호가 표시됩니다. 이 번호는 트랜잭션을 식별하는 고유 번호입니다. 나중에 트랜잭션을 덮어쓰려는 경우 해당 프로세스는 이름 뒤에 트랜잭션의 ID 번호를 파일에 추가하는 것을 제외하고는 새 트랜잭션을 작성하는 프로세스와 동일합니다. 원하는 모든 변경 사항을 파일에 입력할 수 있습니다. 예:

  ```
  hosts maker@
  start_transaction transactionName transactionId
  ```

  그런 다음 해당 명령을 입력하여 트랜잭션을 업데이트할 수 있습니다.

  ```
  chusr mary category (FINANCIAL)
  end_transaction
  ```

- 다음 매개 변수를 사용하여 처리되지 않은 특정 트랜잭션을 표시합니다. eTrustACDir/bin 경로 상에 있는지 확인합니다.

  현재 위치가 *ACInstallDir*/bin 경로인지 확인합니다. 여기서 *ACInstallDir* 은 CA Access Control 의 설치 디렉터리이고 기본값은 /opt/CA/AccessControl/입니다.

명령(매개 변수 포함)	설명
sepmd -m l	매개 변수를 호출한 사용자의 처리되지 않은 트랜잭션을 나열합니다.
sepmd -m la	처리 대기 중인 모든 maker 의 모든 트랜잭션을 나열합니다.

명령(매개 변수 포함)	설명
sepmd -m lo	매개 변수를 호출한 사용자의 트랜잭션을 제외한 모든 maker 의 트랜잭션을 나열합니다. 목록의 각 트랜잭션에는 Maker 이름, 트랜잭션 ID 번호 및 Maker 가 입력한 경우 트랜잭션에 대한 설명이 포함됩니다.

- 다음 명령을 사용하여 표준 출력에 대한 특정 트랜잭션을 검색합니다.

  sepmd -m r transactionId

- 다음 명령을 사용하여 특정 트랜잭션을 삭제합니다.

  sepmd -m d transactionId

트랜잭션 검사 및 처리

이중 제어를 활성화한 경우 maker 가 작성한 트랜잭션을 checker 가 처리해야 합니다.

트랜잭션을 확인하려면

1. 다음 조건이 맞는지 확인합니다.
   - checker 에게 ADMIN 권한이 있습니다.
   - 다른 Checker 가 트랜잭션을 잠그지 않습니다.
   - 사용자와 관련된 명령이 없습니다. 즉, 자신이 포함된 명령을 처리할 수 없습니다.
2. *ACInstallDir*/bin 경로로 이동합니다.

   여기서 *ACInstallDir* 은 CA Access Control 의 설치 디렉터리이고 기본값은 /opt/CA/AccessControl/입니다.
3. 실행 전에 처리 대기 중인 트랜잭션을 봅니다.

   sepmd -m la

   또는 자신이 생성한 것을 제외한 모든 트랜잭션을 봅니다.

   sepmd -m lo

   각 트랜잭션에는 Maker 의 이름, 트랜잭션의 ID 번호, 트랜잭션 이름 또는 설명이 포함됩니다.
4. 트랜잭션을 처리하기 전에 트랜잭션을 잠급니다.

   sepmd -m r *transactionId*

   참고: 잠겨 있는 트랜잭션은 변경할 수 없습니다.

5. 트랜잭션을 처리합니다.

   sepmd -m p *transactionId code*

   코드

   다음 중 *하나*가 될 수 있습니다.

   - **0** - 트랜잭션이 거부됩니다.

     이런 경우에 트랜잭션의 모든 명령은 삭제되고 PMDB 에서 변경은 실행되지 않습니다.

   - **1** - 트랜잭션에 권한이 부여됩니다.

     트랜잭션의 명령이 바로 PMDB 에서 실행됩니다.

   - **2** - 트랜잭션의 잠금이 해제됩니다.

     트랜잭션은 대기 중인 트랜잭션의 큐로 반환되며 나중에 다른 Checker 에 의해 처리될 수 있습니다.

   명령의 성공 여부를 보여주는 메시지가 표시됩니다.

참고: maker 및 checker 에 대한 자세한 내용은 *참조 안내서*의 sepmd 유틸리티와 *selang 참조 안내서*의 start_transaction 명령을 참조하십시오.

seagent 및 sepmdd 데몬 사용

seagent 데몬은 원격 컴퓨터의 요청을 받아 이를 PMDB 에 적용하는 일을 담당합니다. 또한 seagent 데몬은 요청을 seosd 로 보냅니다. sepmdd 데몬은 PMDB 데몬입니다. 이 단원에서는 이들 데몬이 PMDB 환경에서 함께 작업하는 방법에 대해 설명합니다.

seagent 데몬

seagent 데몬은 seoslang 및 seoslang2 TCP 서비스(각각의 기본값이 8890 과 8891) 연결을 대기합니다. 연결 요청이 도착하면 seagent 는 연결 중 통신을 처리하기 위해 자식 프로세스를 생성한 후 계속해서 새 연결을 기다립니다.

사용자가 selang 으로 *hosts* 명령을 입력하면 seagent 는 사용자가 연결되어 있는 컴퓨터에서 child 프로세스를 생성합니다. 그런 다음 하위 프로세스가 명령어 인터페이스에서 명령을 받아 sepmdd 데몬에 전달합니다.

sepmdd 데몬

sepmdd 데몬은 다음 기능을 수행합니다.

- PMDB 관리
- 구독자 데이터베이스 관리
- 변경 내용을 PMDB 에서 구독자 데이터베이스로 전파

seagent 가 PMDB 에 액세스해야 하는 경우 seagent 가 sepmdd 데몬을 자동으로 시작합니다. 일반적으로 사용자는 sepmdd 를 명시적으로 실행할 필요가 없습니다.

참고: AC 환경에서 sepmdd 는 루트가 *아닌* 논리적 사용자 _seagent 에서 실행됩니다. sepmdd 를 사용하여 PMDB 디렉터리 액세스를 제한하는 등과 같이 리소스에 대한 액세스를 허용하거나 제한하려면 _seagent 관련 규칙을 만듭니다.

섀도 파일 사용

일반적으로 sepmdd 는 원시 환경을 업데이트할 때 섀도 파일을 사용하지 않습니다. 섀도 파일을 직접 설정할 수 있습니다. 이렇게 하려면 pmd.ini 파일의 [pmd] 섹션에서 UseShadow 토큰을 yes 로 설정합니다.

UseShadow 토큰을 yes 로 설정하면 sepmdd 는 PMDB 와 같은 디렉터리에 있는 기본 섀도 파일을 사용합니다. 섀도 파일의 위치를 변경하려면 pmd.ini 의 [pmd] 섹션에서 YpServerSecure 토큰을 사용하여 새 위치를 지정합니다.

YpServerSecure 를 사용하여 섀도 파일의 위치를 로컬 호스트의 섀도 파일(예: /etc/shadow)로 변경하면 sepmdd 는 UseSystemFiles 토큰을 yes 로 설정합니다.

중요! UseSystemFiles 토큰을 직접 변경하지 마십시오. sepmdd 또는 seagent 데몬이 자동으로 변경합니다.

참고: seagent 또는 sepmdd 데몬에 대한 자세한 내용은 *참조 안내서*의 seagent 및 sepmdd 유틸리티를 참조하십시오.

메인프레임 암호 동기화

CA Access Control 은 CA Access Control 을 실행하는 Windows 또는 UNIX 컴퓨터와 CA Top Secret, CA ACF2 또는 RACF 보안 제품(및 CA Common Services CAICCI 패키지)을 실행하는 메인프레임 간 암호 동기화를 지원합니다. 동기화는 표준 CA Access Control 암호 정책 모델 방법을 사용하여 수행됩니다.

메인프레임 사용자의 암호 변경 사항은 암호 정책 모델 계층 구조의 모든 컴퓨터에 전파됩니다.

제 12 장: 일반 보안 기능

이 섹션은 다음 항목을 포함하고 있습니다.

유휴 스테이션 보호

터미널이 활성화된 상태로 열려 있을 때 정보는 매우 취약합니다. 사이트의 모든 터미널은 이미 로그인되어 작업 준비 상태이기 때문에 예를 들어 점심 시간 같은 시점에 터미널에 들어온 침입자가 네트워크 경로를 알기 위해 암호를 분석하거나 복잡한 장비를 사용할 필요가 없습니다. 바탕 화면으로 다시 돌아가기 전에 암호를 입력해야 하는 화면 보호기는 매우 유용한 장치이지만 보안 관리자는 모든 사용자가 이런 화면 보호기를 사용하는지 확인할 수 없습니다.

CA Access Control 은 터미널과 스테이션이 지정된 시간보다 오래 유휴 상태에 있을 때마다 터미널과 스테이션을 잠가서 보호하는 화면 잠금 유틸리티인 selock 을 제공합니다. 다시 잠금을 해제할 때 암호를 입력하는 대화 상자가 표시됩니다. 1 분 내에 정확한 암호를 입력하지 않으면 터미널은 잠금 상태를 유지합니다. selock 유틸리티는 selock 이 활성화된 동안 사용자가 자신의 암호를 변경하더라도 화면 잠금을 해제할 수 있는 사용자의 암호를 찾을 수 있습니다.

참고: 화면 잠금 유틸리티 selock 에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

요구 사항에 맞는 selock 옵션을 사용하도록 선택해야 합니다.

- 보안 수준은 낮지만 사용이 편리함

  -timeout 옵션을 사용하여 시간 초과를 큰 값(예: 10 분)으로 설정하고 -lock-timeout 옵션을 사용하여 잠금 시간 초과를 더 큰 값(예: 60 분)으로 설정합니다. 이렇게 하면 selock 이 절전 모드로 전환하여 사용자 작업을 자주 중단시키는 경우를 방지할 수 있습니다. 또한 이 설정은 터미널이 연장된 기간 동안 비활성화된 경우에만 사용자 화면을 잠급니다.

- 보안 수준은 높지만 사용하기 불편함

  -timeout 옵션을 사용하여 작은 값(예: 1 분)을 시간 초과로 설정하고 -lock-timeout 옵션을 사용하여 0 분과 2 분 사이의 작은 값을 잠금 시간 초과로 설정합니다. 이렇게 하면 항상 터미널 액세스를 중지한 후 바로 사용자의 작업이 숨겨지고 액세스를 복원하기 위해 암호가 필요합니다. 절전 모드가 시작된 이후에 터미널을 다시 활성화하기 위해 selock 이 항상 암호 입력을 요구하도록 하려면 -lock-timeout 옵션을 사용하여 잠금 시간 제한을 0 으로 설정하십시오.

- 사용자가 시스템에 로그인할 때마다 자동으로 실행되도록 selock 명령을 X 시작 셸에 추가할 수 있습니다. 스크립트는 root ID 가 아니라 사용자 ID 로 실행해야 합니다. selock 명령을 시작 스크립트에 통합하는 방법은 사이트의 특정 환경에 따라 다릅니다.

  참고: 시작 스크립트에 대한 자세한 내용은 사용하는 UNIX 시스템의 설명서를 참조하십시오.

보호 모드

selock 은 다음과 같은 세 가지 작업 모드를 제공합니다.

모니터 모드

모니터 모드는 selock 의 초기 모드입니다. 이 모드에서는 키보드와 마우스 활동이 모니터됩니다. selock 에서 제한 시간 동안 키보드나 마우스 활동을 감지할 수 없고 transparent 매개 변수가 off 인 경우 selock 은 절전 모드로 자동 전환합니다. 모니터 모드에서 절전 모드로 전환하기 위해 암호를 입력할 필요가 없습니다.

절전 모드

절전 모드에서 selock 은 빈 화면과 움직이는 시스템 아이콘을 표시합니다. 빈 화면과 움직이는 아이콘을 통해 작업 시 두 가지 장점이 있습니다.

- 권한 없는 사용자가 화면을 볼 위험이 줄어듭니다.
- 감소한 화면 잔상

selock 옵션을 사용하여 아이콘의 모양과 위치를 조정할 수 있습니다. selock 이 키보드 또는 마우스 활동을 감지하면 즉시 절전 모드에서 모니터 모드로 돌아가서 화면 디스플레이를 절전 모드로 전환하기 전 내용으로 복원합니다. 모니터 모드에서 절전 모드로 전환하기 위해 암호를 입력할 필요가 없습니다.

lock-timeout 매개 변수에서 지정한 시간 동안 selock 이 절전 모드로 유지되면 잠금 모드로 자동 전환됩니다. selock 은 절전 모드에서 잠금 모드로 전환할 때 아무런 표시도 나타내지 않습니다.

잠금 모드

기본 설정을 사용한 잠금 모드에서 selock 은 움직이는 아이콘을 검은색 배경 위에 계속해서 표시합니다. selock 이 키보드나 마우스 활동을 감지하면 사용자의 암호 입력을 요구하는 대화 상자가 나타납니다.

사용자가 올바른 암호를 입력하면 selock 은 다시 모니터 모드로 전환됩니다. 사용자가 잘못된 암호를 입력할 경우 암호 입력 대화 상자가 닫히고 selock 은 그대로 잠금 모드에 있습니다.

-transparent 옵션을 *on* 으로 설정하면 selock 은 화면을 잠그지만 화면 내용을 표시하고 진행 중인 프로세스를 업데이트합니다. 화면 배경은 화면이 잠겼음을 나타내도록 변경됩니다. 잠금 모드를 사용할 때 절전 모드는 실행되지 않습니다.

유휴 상태일 때 스테이션을 잠금으로 설정

selock 유틸리티를 사용하면 유휴 스테이션을 잠그면 유휴 상태인 스테이션에 대한 무단 액세스를 방지할 수 있습니다.

유휴 상태일 때 스테이션을 잠금으로 설정하려면

1. (선택 사항) DISPLAY 환경 변수를 설정합니다.

   selock 명령을 실행시키려면 DISPLAY 환경 변수를 설정해야 합니다. 그러나 대신 selock 명령으로 대상 디스플레이를 직접 지정할 수 있습니다.

2. selock 명령을 사용자의 로그인 스크립트(.login 파일)에 포함합니다.

   또는 /etc/login 이나 /etc/cshrc 파일에 selock 명령을 포함할 수도 있습니다.

참고: 두 명의 사용자는 잠긴 화면의 잠금을 항상 해제할 수 있습니다. 기본적으로 현재 사용자와 root 가 여기에 해당됩니다. 그러나 seos.ini 파일의 [selock] 섹션에 있는 unlocking_user 토큰에 다른 사용자 이름을 지정하면 root 를 다른 사용자로 바꿀 수 있습니다. -user 옵션을 사용하여 selock 을 실행하면 현재 사용자를 다른 사용자로 바꿀 수 있습니다.

예: 시작 파일의 유휴 스테이션 잠금 명령

다음은 X 시작 파일에 적합한 일반적인 시작 명령입니다.

```
selock -display $DISPLAY -timeout 5
```

이 명령은 터미널이 비활성화되고 5 분 후 selock 을 활성화합니다.

전역 xstartup 스크립트에 다음 명령줄을 입력하는 것이 좋습니다. xstartup 스크립트는 대개 /usr/lib/X11/xdm/Xstartup 디렉터리에 있습니다.

```
selock -display $DISPLAY -user $USER -timeout 3 &
```

이 명령문을 통해 X 터미널을 사용하는 모든 사용자는 터미널 잠금 프로그램을 사용할 수 있니다.

화면 잠금 아이콘 변경

selock 이 사용하는 기본 시스템 아이콘은 CA Access Control 로고이고 ACInstallDir/data/admin/Selogo.xpm 파일에 있습니다.

사용자 정의 아이콘을 선택하려면 이 파일을 바꾸십시오.

참고: 아이콘 파일은 XPM 버전 3.3 버전 형식이어야 합니다.

API 를 사용한 리소스 보호

CA Access Control 에 포함되지 않은 리소스(즉, 내부 리소스)를 정의한 경우 API 를 사용하여 이 리소스를 보호할 수 있습니다. 각 API 에는 다음과 같은 두 가지 레이어가 있습니다.

함수 라이브러리

프로그래머가 CA Access Control 권한 부여 엔진을 사용할 수 있도록 합니다.

사용자 종료

시스템 관리자가 사이트의 요구 사항에 맞게 CA Access Control 동작을 조정할 수 있도록 합니다.

참고: CA Access Control API 에 대한 자세한 내용은 *SDK 안내서*를 참조하십시오.

스택 오버플로 보호: STOP

스택 오버플로를 사용하면 해커가 root 사용자(슈퍼 사용자)로 여러 차례 원격 또는 로컬 시스템에서 임의의 명령을 실행할 수 있습니다. 해커는 운영 체제 또는 다른 프로그램의 버그를 이용하여 이 작업을 수행합니다. 이러한 버그로 인해 사용자는 실행될 다음 명령을 변경하여 프로그램 스택을 덮어쓸 수 있습니다.

스택 오버플로는 간단한 버그가 아닙니다. 반환 주소를 의미 있는 주소로 덮어씀으로써 결과적으로 일반적으로 동일한 블록에 있는 권한 없는 코드에 대해 전송된 제어를 수행하는 블록을 만들 수 있습니다.

STOP(스택 오버플로 보호)은 해커가 스택 오버플로를 작성하고 사용하여 시스템에 침입하는 것을 금지하는 기능입니다.

참고: Linux 네이티브 스택 무작위화(ExecShield randomize)가 사용되는 경우 Red Hat Linux 와 SuSE Linux 의 STOP 기능은 활성화되지 않습니다.

Linux s390 RHEL 4 에서 네이티브 스택 무작위화는 작동하지 않으며, STOP 을 활성화하려면 네이티브 스택 무작위화를 비활성화야 합니다. 네이티브 스택 무작위화를 비활성화하려면 다음 명령을 입력하십시오.

```
echo 0 > /proc/sys/kernel/exec-shield-randomize
```

스택 오버플로 보호(STOP) 시작 및 중지

STOP 을 처음 설치하면 스택 오버플로 보호가 기본적으로 활성화됩니다. 스택 오버플로 보호를 비활성화하려면 seos.ini 파일의 [seos_syscall] 섹션에서 토큰을 변경한 후 CA Access Control 을 다시 시작해야 합니다. 이렇게 하려면 다음 seini 명령을 사용합니다.

```
seini -s SEOS_syscall.STOP_enabled 0
```

또한 seos.ini 파일을 수동으로 변경할 수 있습니다.

스택 오버플로 보호(STOP)를 다시 활성화하려면 토큰 값을 1 로 변경한 후 CA Access Control 을 다시 시작합니다.

참고: STOP 이 Sun Solaris 7 시스템에서 활성화되면 dbx 프로그램이 제대로 작동하지 않습니다. STOP 에 의해 보호되는 시스템에서 dbx 를 사용해야 하는 경우 먼저 STOP 을 비활성화해야 합니다.

리소스의 요일/시간 액세스 규칙 정의

CA Access Control 을 사용하여 리소스 액세스에 대한 요일 및 시간 제한을 지정할 수 있습니다. 이 기능은 TERMINAL 액세스, SURROGATE 요청 및 사용자 정의 리소스에 이용할 수 있습니다. 예를 들어 다음 규칙은 주말과 매일 8 시와 19 시 사이 이외의 시간에 터미널 ws3 을 완전히 비활성화합니다.

```
chres TERMINAL ws3 restrictions(days(weekdays) time(0800:1900))
```

이 시간 외에는 해당 스테이션으로부터의 로그인 요청을 수락하지 않습니다.

CA Access Control 을 사용하면 업무 외 시간에는 높은 권한이 부여된 사용자로 대체하려는 요청을 거부할 수 있습니다. 사용자 AcctMgr 이 금융 트랜잭션을 수행할 수 있는 계정 관리자이며 근무 시간 및 주중에만 제한된 AcctMgr 로 로그인할 수 있다고 가정합니다. 침입자 또는 권한이 없는 직원이 **su** AcctMgr 명령을 호출하여 AcctMgr 계정에 액세스를 시도할 수 있습니다. 다음 명령을 사용하여 지정된 시간 이외에 사용자 이름을 AcctMgr 로 대체하지 못하도록 합니다.

```
chres SURROGATE USER.AcctMgr restrictions(days(weekdays) time(0800:1900))
```

내부 응용 프로그램을 구현하는 데 사용하는 사용자 정의 abstract 클래스를 비롯하여 보호되는 모든 리소스에 대해 동일한 기술을 구현할 수 있습니다.

B1 보안 수준 인증

CA Access Control 에는 다음과 같은 B1 "Orange Book" 기능이 포함되어 있습니다.

- 보안 범주
- 보안 레이블
- 보안 수준

보안 수준

보안 수준 검사를 활성화하면 CA Access Control 은 다른 권한 부여 검사 외에 보안 수준 검사도 수행합니다. 보안 수준은 사용자와 리소스에 할당할 수 있는 1 에서 255 사이의 양수입니다. 사용자가 보안 수준이 할당된 리소스에 대한 액세스를 요청하면 CA Access Control 은 리소스의 보안 수준과 사용자의 보안 수준을 비교합니다. 사용자의 보안 수준이 리소스의 보안 수준보다 크거나 같은 경우 CA Access Control 은 다른 권한 부여 검사를 계속하고, 그렇지 않은 경우에는 사용자의 리소스 액세스를 거부합니다.

SECLABEL 클래스가 활성화된 경우 CA Access Control 은 리소스 및 사용자의 보안 레이블과 연관된 보안 수준을 사용하며 리소스 및 사용자 레코드에 명시적으로 설정된 보안 수준은 무시됩니다.

보안 수준을 검사하여 리소스를 보호하려면 리소스의 레코드에 보안 수준을 할당합니다. newres 또는 chres 명령의 level 매개 변수는 리소스에 보안 수준을 할당합니다.

사용자가 보안 수준 검사에 의해 보호된 리소스에 액세스하려면 사용자의 레코드에 보안 수준을 할당합니다. newusr 또는 chusr 명령의 level 매개 변수는 사용자에게 보안 수준을 할당합니다.

보안 수준 확인 활성화

다음 setoptions 명령을 사용하여 보안 수준 검사를 활성화할 수 있습니다.

```
setoptions class+ (SECLEVEL)
```

보안 수준 확인 비활성화

다음 setoptions 명령을 사용하여 보안 수준 검사를 비활성화할 수 있습니다.

```
setoptions class- (SECLEVEL)
```

보안 범주

보안 범주 검사를 활성화하면 CA Access Control 은 다른 권한 부여 검사 외에 보안 범주도 검사합니다. 사용자가 하나 이상의 보안 범주가 할당된 리소스에 대한 액세스를 요청하면 CA Access Control 은 리소스 레코드의 보안 범주 목록을 사용자 레코드의 보안 범주 목록과 비교합니다. 리소스에 할당된 모든 범주가 사용자 범주 목록에 나타나면 CA Access Control 은 다른 권한 부여 검사를 계속하고, 그렇지 않은 경우에는 사용자의 리소스 액세스가 거부됩니다.

SECLABEL 클래스가 활성화되면 CA Access Control 은 리소스 및 사용자의 보안 레이블과 관련된 보안 범주 목록을 사용합니다. 사용자 및 리소스 레코드의 범주 목록은 무시됩니다.

보안 범주 검사로 리소스를 보호하려면 리소스 레코드에 하나 이상의 보안 범주를 할당합니다. newres 또는 chres 명령의 category 매개 변수를 사용하여 리소스에 보안 범주를 할당합니다.

사용자에게 보안 범주 검사에 의해 보호되는 리소스에 대한 액세스를 허용하려면 하나 이상의 보안 범주를 사용자 레코드에 할당합니다. newusr 또는 chusr 명령의 category 매개 변수를 사용하여 사용자에게 보안 범주를 할당합니다.

보안 범주 확인 활성화

다음 setoptions 명령을 사용하여 보안 범주 검사를 활성화할 수 있습니다.

```
setoptions class+ (CATEGORY)
```

보안 범주 확인 비활성화

다음 setoptions 명령을 사용하여 보안 범주 검사를 비활성화할 수 있습니다.

```
setoptions class-(CATEGORY)
```

보안 범주 정의

CATEGORY 클래스의 리소스를 정의하여 보안 범주를 정의합니다. 다음 newres 명령으로 보안 범주를 정의합니다.

newres CATEGORY *name*

여기서 *name* 은 보안 범주의 이름입니다.

보안 범주 "*Sales*"를 정의하려면 다음 명령을 입력합니다.

newres CATEGORY Sales

보안 범주 "*Sales*"와 "*Accounts*"를 정의하려면 다음 명령을 입력합니다.

newres CATEGORY (Sales,Accounts)

보안 범주 나열

데이터베이스에서 정의된 모든 보안 범주 목록을 표시하려면 다음과 같이 show 명령을 사용합니다.

find CATEGORY

보안 범주 목록이 화면에 표시됩니다.

보안 범주 삭제

CATEGORY 클래스의 레코드를 제거하여 보안 범주를 삭제합니다. 다음 rmres 명령은 보안 범주를 제거합니다.

rmres CATEGORY *name*

여기서 *name* 은 보안 범주의 이름입니다.

보안 범주 "*Sales*"를 제거하려면 다음 명령을 입력합니다.

rmres CATEGORY Sales

보안 레이블

특정 보안 수준과 0 개 이상의 보안 범주 사이의 관계를 나타내는 보안 레이블입니다.

보안 레이블 검사를 활성화하면 CA Access Control 은 다른 권한 부여 검사 외에 보안 레이블 검사도 수행합니다. 사용자가 보안 레이블이 할당된 리소스에 대한 액세스를 요청하면 CA Access Control 은 리소스 레코드의 보안 레이블에 지정된 보안 범주 목록과 사용자 레코드의 보안 레이블에 지정된 보안 범주 목록을 비교합니다. 리소스의 보안 레이블에 할당된 모든 범주가 사용자의 보안 레이블에 나타나는 경우 CA Access Control 은 보안 수준 검사를 계속합니다. 그렇지 않은 경우 사용자의 리소스 액세스가 거부됩니다. CA Access Control 은 리소스 레코드의 보안 레이블에 지정된 보안 수준과 사용자 레코드의 보안 레이블에 지정된 보안 수준을 비교합니다. 사용자의 보안 레이블에 할당된 보안 수준이 리소스의 보안 레이블에 할당된 보안 수준보다 크거나 같을 경우 CA Access Control 은 다른 권한 부여 검사를 계속하고, 그렇지 않은 경우에는 사용자의 리소스 액세스가 거부됩니다.

보안 레이블 검사가 활성화되면 사용자 및 리소스 레코드에 지정된 보안 범주 및 보안 수준은 무시되며 보안 레이블 정의에 지정된 보안 수준 및 범주만 사용됩니다.

보안 레이블 검사로 리소스를 보호하려면 해당 리소스 레코드에 보안 레이블을 할당합니다. newres 또는 chres 명령의 label 매개 변수는 리소스에 보안 레이블을 할당합니다.

사용자가 보안 레이블 검사로 보호되는 리소스에 액세스할 수 있도록 하려면 사용자의 레코드에 보안 레이블을 할당합니다. newusr 또는 chusr 명령의 label 매개 변수를 사용하여 사용자에게 보안 레이블을 할당합니다.

보안 레이블 확인 활성화

다음 setoptions 명령을 사용하여 보안 레이블 검사를 활성화할 수 있습니다.

```
setoptions class+(SECLABEL)
```

보안 레이블 확인 비활성화

다음 setoptions 명령을 사용하여 보안 레이블 검사를 비활성화할 수 있습니다.

```
setoptions class-(SECLABEL)
```

보안 레이블 정의

SECLABEL 클래스의 리소스를 정의하여 보안 레이블을 정의합니다. 다음 newres 명령은 보안 레이블을 정의합니다.

newres SECLABEL *name* category(*securityCategories*) level(*securityLevel*)

설명:

name

보안 레이블의 이름을 지정합니다.

securityCategories

보안 범주 목록을 지정합니다. 하나 이상의 목록을 지정하려면 보안 범주 이름을 공백 또는 쉼표로 구분합니다.

securityLevel

보안 수준을 지정합니다. 1 과 255 사이의 정수를 사용합니다.

보안 레이블 Managers 가 보안 범주 Sales 및 Accounts 를 포함하고 보안 수준이 95 가 되도록 정의하려면 다음 명령을 입력합니다.

newres SECLABEL Manager category(Sales,Accounts) level(95)

보안 레이블 나열

데이터베이스에서 정의된 모든 보안 레이블 목록을 표시하려면 다음과 같이 show 명령을 사용합니다.

find SECLABEL

보안 레이블 목록이 화면에 표시됩니다.

보안 레이블 삭제

SECLABEL 클래스의 레코드를 제거하여 보안 레이블을 삭제합니다. 다음 rmres 명령은 보안 레이블을 제거합니다.

rmres SECLABEL *name*

여기서 *name* 은 보안 레이블의 이름입니다.

보안 범주 *"Managers"*를 제거하려면 다음 명령을 입력합니다.

rmres SECLABEL Manager

제 13 장: 이벤트 감사

이 섹션은 다음 항목을 포함하고 있습니다.

감사 규칙 설정

보안 감사를 위해 CA Access Control 은 데이터베이스에 정의된 감사 규칙에 따라 액세스 거부 또는 허용 이벤트에 대한 감사 레코드를 유지합니다.

모든 접근자와 리소스는 다음 값 중 하나 이상으로 설정될 수 있는 AUDIT 속성을 가집니다.

FAIL

접근자의 리소스 액세스 실패를 로그 파일에 기록합니다.

SUCCESS

접근자의 리소스 액세스 성공을 로그 파일에 기록합니다.

LOGINFAIL

접근자의 모든 로그온 실패를 로그 파일에 기록합니다. 이 값은 리소스에 적용되지 않습니다.

LOGINSUCCESS

접근자의 모든 성공적인 로그온을 로그 파일에 기록합니다. 이 값은 리소스에 적용되지 않습니다.

ALL

접근자의 경우 FAIL, SUCCESS, LOGINFAIL 및 LOGINSUCCESS 와 같은 정보를 로그 파일에 기록하고, 리소스의 경우 FAIL 및 SUCCESS 와 같은 정보를 로그 파일에 기록합니다.

NONE

접근자 또는 리소스와 관련된 어떤 정보도 로그 파일에 기록하지 않습니다.

TRACE

ALL 과 동일한 정보 및 모든 시스템 이벤트를 기록합니다. 이 값은 리소스에 적용되지 않습니다.

데이터베이스에서 접근자 또는 리소스 레코드를 작성하거나 업데이트할 때마다 AUDIT 속성을 지정할 수 있습니다. 또한 기록된 이벤트의 전자 메일 통지를 전송해야 하는지 여부와 전자 메일 통지의 수신인을 지정할 수 있습니다.

감사 로그의 레코드는 이러한 감사 규칙에 따라 누적됩니다. 이벤트를 어떤 기준에 따라 로그 파일에 기록할지 여부는 다음 사항에 따라 결정됩니다.

- 리소스 또는 접근자에 AUDIT(ALL)이 있는 경우, CA Access Control 에서 보호되는 해당 리소스에 대한 접근자와 모든 이벤트에 대한 모든 로그인 이벤트가 액세스 성공 여부에 상관 없이 기록됩니다.
- CA Access Control 에서 보호되는 리소스에 대한 액세스가 성공하고 사용자 또는 리소스에 AUDIT(SUCCESS)가 있는 경우 이벤트가 기록됩니다.
- CA Access Control 에서 보호되는 리소스에 대한 액세스가 실패하고 접근자 또는 리소스에 AUDIT(FAIL)이 있는 경우 이벤트가 기록됩니다.

또한 사용자를 추적 가능으로 설정하면 해당 사용자에 대한 추적 레코드가 기록될 때마다 해당 감사 레코드가 감사 로그에 기록됩니다.

CA Access Control 이 감사 로그에 기록하는 감사 이벤트 정의

CA Access Control 은 감사 로그에 성공 및 실패한 액세스를 기록합니다. 감사 대상 접근자 또는 리소스의 AUDIT 속성 값을 변경하여 CA Access Control 이 감사 로그에 기록하는 액세스 이벤트를 정의합니다. 이 방법을 사용하여 CA Access Control 이 감사 로그에 모든 추적 이벤트를 기록하도록 지정할 수도 있습니다.

AUDIT 속성을 사용하여 CA Access Control 이 감사 로그에 기록하는 감사 이벤트를 지정합니다. selang 또는 CA Access Control 끝점 관리를 사용하여 다음과 같이 리소스 또는 접근자에 대한 AUDIT 속성을 설정하십시오.

AUDIT 값	CA Access Control 이 기록하는 대상	해당되는 대상
FAIL	액세스 실패	사용자 및 리소스
SUCCESS	액세스 성공	사용자 및 리소스
LOGINFAIL	로그인 실패	사용자
LOGINSUCCESS	로그인 성공	사용자
ALL	FAIL, SUCCESS, LOGINFAIL, LOGINSUCCESS, INTERACTIVE 에 해당	사용자 및 리소스
TRACE	ALL 및 모든 시스템 이벤트에 해당	사용자
INTERACTIVE	UNIX 컴퓨터의 사용자 세션	사용자
NONE	로깅 없음	사용자 및 리소스

참고: 사용자의 감사 속성이 설정되어 있지 않으면 그룹 또는 프로필 그룹의 AUDIT 값이 CA Access Control 이 해당 사용자에 대해 사용하는 감사 모드에 영향을 줄 수 있습니다.

사용자 세션 로깅이 작동하는 방법

사용자 세션 로깅은 끝점에서 사용자 활동을 추적하고, 세션을 재생하고, 이 세션 중 사용자가 입력한 명령을 볼 수 있게 해 줍니다.

세션 로거는 /etc/shells 파일에 수록된 모든 프로그램에 대한 입력을 로깅합니다. 예를 들어, /usr/bin/passwd 가 /etc/shells 에 수록되었고 passwd 를 사용하여 암호를 변경하는 경우, seaudit 유틸리티는 세션 로그를 표시할 때 변경된 암호를 표시합니다. 세션 로깅을 구현하기 전에 /etc/shells 파일을 검토할 것을 권장합니다.

다음 프로세스는 사용자 세션 로깅이 작동하는 방법에 대해 설명합니다.

1. 키보드 로거 옵션을 활성화하여 CA Access Control 을 설치합니다.

   키보드 로거를 사용하도록 CA Access Control 매개 변수를 사용자 지정하십시오.

   참고: 키보드 로거는 설치 후 seos.ini 파일에서 활성화할 수 있습니다.

2. CA Access Control 을 시작합니다.

   키보드 로거 데몬인 KBLAudMngr 가 실행 중인지 확인하십시오. CA Access Control 데몬의 상태를 보려면 issec 유틸리티를 사용하십시오.

3. 세션 로깅을 활성화하려면 추적할 사용자에게 INTERACTIVE 속성을 할당합니다. 예:

   - selang:

     ```
     eu user1 audit(interactive)
     ```

   - CA Access Control 끝점 관리:

     "사용자 속성" 창의 "감사" 탭에서 "대화식" 상자를 선택하십시오.

   CA Access Control 이 해당 사용자 계정에 대해 세션 로깅을 활성화합니다.

4. 사용자가 끝점에 로그인하면 CA Access Control 은 사용자 세션을 기록합니다. 사용자가 끝점에서 로그아웃하면 세션이 종료됩니다.
5. CA Access Control 이 기록된 세션을 kbl.audit 로그 파일에 저장합니다. 이 파일은 다음 디렉터리에 있습니다.

   /opt/CA/AccessControl/log
6. kbl.audit 로그 파일의 내용을 보려면 -kbl 명령과 함께 seaudit 유틸리티를 사용합니다. 예:

   ./seaudit -kbl -sid 65223 -rp

   참고: seaudit -kbl 명령에 대한 자세한 내용은 *참조 안내서*를 참조하십시오. 회사의 호스트에서 사용자 세션을 수집하여 보고서를 생성하려면 CA Access Control 끝점을 CA Enterprise Log Manager 와 통합하는 것이 좋습니다. CA Enterprise Log Manager 와의 통합에 대한 자세한 내용은 *통합 안내서*를 참조하십시오.

CA Access Control 이 사용자의 감사 모드를 결정하는 방법

사용자에 대한 감사 모드는 CA Access Control 이 해당 사용자에 대한 감사 로그로 전달하는 감사 이벤트를 지정합니다. 다음 프로세스는 CA Access Control 이 사용자에 대한 감사 모드를 결정하는 방식을 설명합니다.

1. CA Access Control 은 USER 또는 XUSER 클래스에 있는 사용자의 레코드에 AUDIT 속성 값이 있는지 확인합니다.

   사용자의 레코드에 AUDIT 속성 값이 있으면 CA Access Control 은 이 값을 해당 사용자의 감사 모드로 사용합니다.
2. CA Access Control 은 사용자가 프로필 그룹에 할당되었는지 여부를 확인합니다. 사용자가 프로필 그룹에 할당된 경우 CA Access Control 은 GROUP 클래스의 프로필 그룹 레코드에 AUDIT 속성 값이 있는지 확인합니다.

   사용자가 프로필 그룹에 할당되어 있고 프로필 그룹의 레코드에 AUDIT 속성 값이 있는 경우 CA Access Control 은 이 값을 해당 사용자의 감사 모드로 사용합니다.

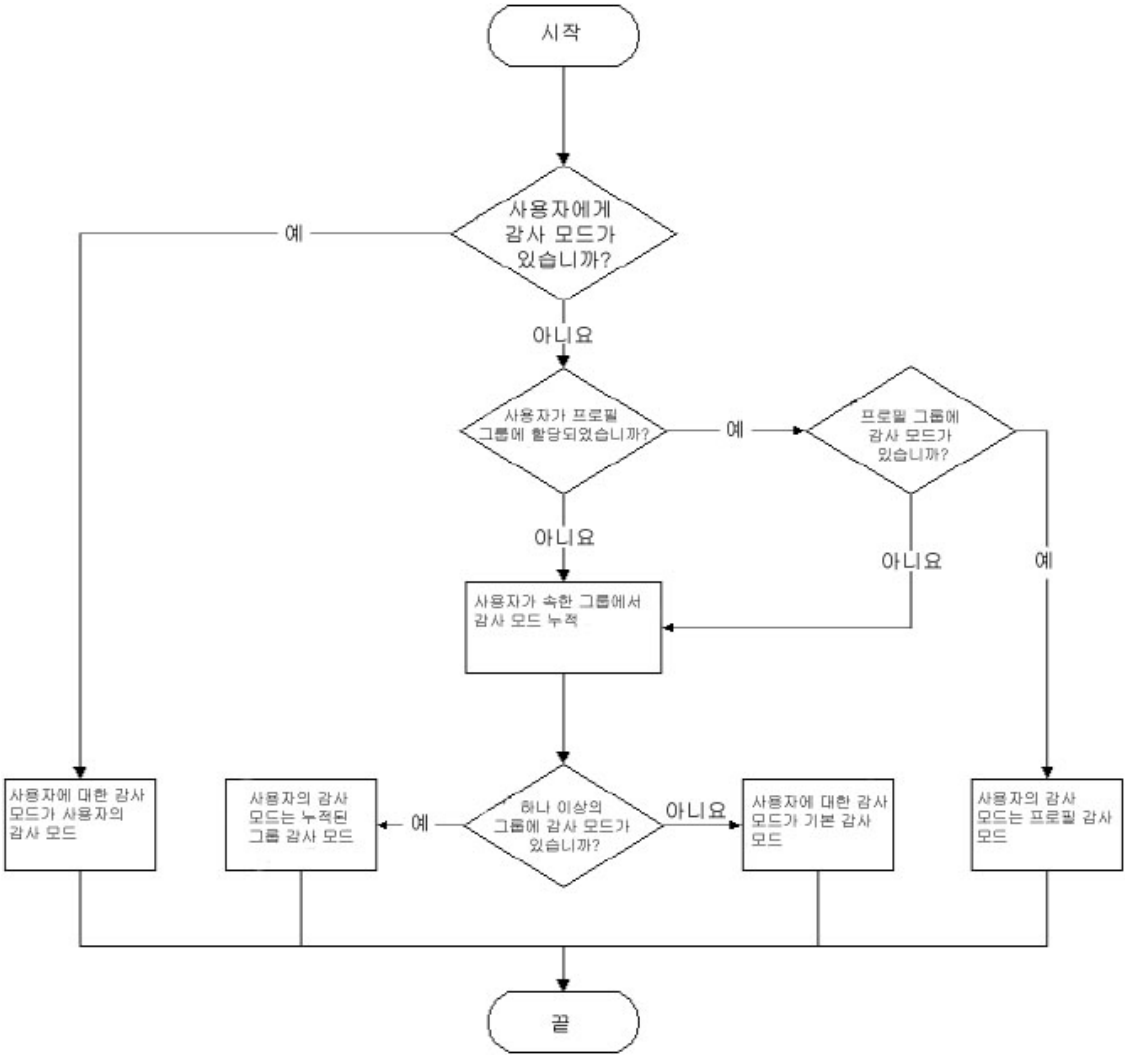
3. CA Access Control 은 사용자가 그룹의 구성원이지 확인합니다. 사용자가 그룹 구성원인 경우 CA Access Control 은 GROUP 또는 XGROUP 클래스의 그룹 레코드에 AUDIT 속성 값이 있는지 확인합니다.

   사용자가 그룹 구성원이고 그룹의 레코드에 AUDIT 속성 값이 있는 경우 CA Access Control 은 이 값을 해당 사용자의 감사 모드로 사용합니다. 사용자가 그룹 구성원이 아니거나 그룹의 레코드에 AUDIT 속성 값이 없는 경우 CA Access Control 은 해당 사용자에게 시스템 전체 감사 모드를 할당합니다.

   참고: 사용자가 서로 다른 감사 모드를 갖는 여러 그룹에 속한 구성원인 경우 사용자의 감사 모드는 누적됩니다. 이러한 사용자의 감사 모드는 속한 그룹의 모든 감사 모드의 합계입니다.

참고: CA Access Control 이 그룹의 AUDIT 속성 값을 사용하여 사용자의 감사 모드를 결정하는 경우, 사용자가 로그인된 상태에서 그룹의 감사 모드를 변경하면 로그인된 사용자의 감사 모드 또한 변경됩니다. 그룹 감사 모드의 변경 사항이 반영되도록 사용자가 로그오프할 필요는 없습니다.

다음 그림은 CA Access Control 이 사용자에 대한 감사 모드를 결정하는 방식을 설명합니다.



예: 그룹별 감사

사용자 Jan 은 그룹 A 및 그룹 B 의 구성원입니다. 그룹 A 의 감사 모드는 'FAIL'이고 그룹 B 의 감사 모드는 'SUCCESS'입니다. Jan 이 두 그룹 모두의 구성원이므로 Jan 은 누적된 감사 모드인 'FAIL' 및 'SUCCESS'를 갖습니다.

추가 정보:

CA Access Control 이 프로필 그룹을 사용하여 사용자 속성을 파악하는 방법 (페이지 41)

사용자 및 엔터프라이즈 사용자의 기본 감사 모드

사용자(USER 개체)를 만들 때 CA Access Control 은 개체에 기본 AUDIT_MODE 를 할당합니다. AUDIT_MODE 속성의 기본 값은 Failure, SuccessLogin, SuccessFailure 입니다.

엔터프라이즈 사용자(XUSER 개체)를 만들 때 기본적으로 CA Access Control 은 개체에 기본 AUDIT_MODE 값을 할당하지 않습니다.

참고: (UNIX) USER 개체에 대한 AUDIT_MODE 속성의 기본값을 변경하려면 lang.ini 파일의 [newusr] 섹션에 있는 DefaultAudit 값을 편집하십시오.

일부 사용자에 대해 기본 감사 값으로 변경

r12.0 SP1 CR1 이전 버전에서는 다음 접근자에 대한 기본 감사 모드가 'None'이었습니다.

- 해당 USER 클래스 레코드에 정의된 AUDIT 값이 없는 사용자 및 정의된 AUDIT 값이 있는 프로필 그룹에 연결되지 않은 사용자
- 데이터베이스에 정의되지 않은 모든 사용자(_undefined 사용자 레코드로 표시됨)

  참고: 엔터프라이즈 사용자의 경우 CA Access Control 은 어떠한 사용자도 정의되지 않은 사용자로 간주하지 않습니다. _undefined 사용자의 속성은 이 경우 관련이 없습니다.

r12.0 SP1 CR1 부터 이러한 접근자에 대한 기본 감사 모드는 Failure, LoginSuccess, LoginFailure 입니다. 이전 방식을 사용하려면 이러한 사용자에 대해 AUDIT 속성의 값을 'None'으로 지정하십시오.

GROUP 레코드에 대한 AUDIT 속성 값 변경

다음의 두 가지 기능이 있는 GROUP 레코드가 있는 경우:

- 한 세트의 사용자에 대한 감사 정책을 정의하는 프로필
- 두 번째 사용자 세트에 대한 컨테이너

r12.0 SP1 CR1 부터 GROUP 레코드도 두 번째 사용자 세트에 대한 감사 정책을 정의합니다. 이 방식 변경에 따른 잠재적인 문제를 방지하려면 두 번째 사용자 세트에 대한 별도의 GROUP 을 만드십시오.

경고 모드

*경고 모드*는 리소스에 적용할 수 있는 속성이고 클래스에 적용할 수 있는 옵션입니다. 경고 모드가 리소스나 클래스에 적용되고 액세스가 액세스 규칙을 위반하면 CA Access Control 은 감사 로그 항목과 반환 코드 W 를 기록하지만 리소스 액세스를 허용합니다. 클래스가 경고 모드에 있는 경우에는 해당 클래스의 모든 리소스가 경고 모드에 있습니다.

CA Access Control 이 전체 적용 모드에 있는 경우에만 경고 모드가 적용됩니다.

참고: "전체 적용 모드"는 UNIX 용 CA Access Control 에서 지원하는 유일한 모드입니다. Windows 용 CA Access Control 에서는 감사 전용 모드도 지원합니다.

액세스 정책을 도입하거나 수정할 때 경고 모드를 사용할 수 있습니다. 이 작업을 수행할 경우 감사 로그를 검토하여 원하는 정책을 적용하기 전에 해당 정책 결과를 미리 볼 수 있습니다. seaudit 명령을 사용하여 감사 로그를 표시할 수 있습니다.

클래스에 *warning* 속성이 있으면 해당 클래스에 경고 모드를 적용할 수 있습니다. 리소스 그룹이나 클래스가 경고 모드에 있으면 액세스 규칙이 위반될 때 CA Access Control 은 액세스를 허용하고 리소스 그룹이나 클래스가 아니라 리소스를 참조하는 항목을 감사 로그에 기록합니다.

리소스와 클래스에 대한 경고 모드 설정은 독립적입니다. 리소스에 경고 모드를 적용하면 리소스가 클래스에 속하고 해당 클래스에서 경고 모드를 제거하는 경우에도 리소스가 경고 모드로 유지됩니다.

참고: 리소스나 클래스에 *warning* 속성이 있는 경우에만 해당 리소스나 클래스에 경고 모드를 적용할 수 있습니다. 일부 리소스나 클래스에는 이 속성이 포함되어 있지 않습니다.

리소스에 경고 모드 적용

리소스에 경고 모드를 적용하여 액세스 규칙을 적용할 필요 없이 해당 규칙의 영향을 모니터링할 수 있습니다.

참고: 개별 리소스에 경고 모드를 적용할 뿐만 아니라 클래스에 경고 모드를 적용 (페이지 203)할 수 있습니다.

리소스에 경고 모드를 적용하려면

1. CA Access Control 끝점 관리에서 경고 모드를 적용하려는 리소스를 편집합니다.

   해당 "수정" 페이지가 나타납니다.

2. "감사" 탭을 클릭합니다.

   리소스의 "감사 모드" 페이지가 나타납니다.

3. "경고 모드"를 선택하고 "저장"을 클릭합니다.

   이제 수정한 리소스가 경고 모드에 있습니다.

참고: 경고 모드에서 CA Access Control 은 액세스가 허용되지만 액세스 규칙이 위반될 때 항상 경고 레코드를 감사 로그에 기록합니다. 이 작업을 수행하기 위해 리소스에 대한 audit 속성을 설정할 필요가 없습니다.

sereport 유틸리티(보고서 번호 6)를 사용하여 경고 모드에서 모든 리소스를 확인할 수 있습니다.

예: 파일에 경고 모드 적용

다음 selang 예에서는 c:\myfile 파일에 경고 모드를 적용합니다.

```
chres FIlE c:\myfile warning
```

예: 파일에서 경고 모드 지우기

다음 selang 예에서는 c:\myfile 파일을 경고 모드에서 해제합니다.

```
chres FIlE c:\myfile warning-
```

이제 myfile 에 대한 경고 모드가 활성 상태가 아니므로 CA Access Control 은 myfile 의 액세스 규칙을 적용합니다.

예: 터미널에 경고 모드 적용

다음 selang 예에서는 myterminal 터미널에 경고 모드를 적용합니다.

```
chres terminal myterminal warning
```

CA Access Control 은 myterminal 터미널을 통한 권한 있는 사용자의 액세스를 허용하지만 일반적으로 해당 터미널을 통한 액세스가 거부된 사용자에 대한 감사 레코드를 기록합니다.

클래스에 경고 모드 적용

개별 레코드에 경고 모드를 적용하지 않고 클래스의 모든 레코드에 경고 모드를 적용할 수 있습니다. 경고 모드를 사용하여 액세스 규칙을 적용하지 않고 해당 규칙을 모니터링할 수 있습니다.

경고 모드로 클래스 사용

1. CA Access Control 끝점 관리에서 다음을 수행하십시오.
   a. "구성"을 클릭합니다.
   b. "클래스 활성화"를 클릭합니다.

   "클래스 활성화" 페이지가 나타납니다.
2. "경고" 열에서 경고 모드로 사용할 클래스에 대한 확인란을 선택합니다.
3. "저장"을 클릭합니다.

   CA Access Control 옵션이 성공적으로 업데이트되었음을 알리는 메시지가 표시됩니다.

경고 모드에 어떤 리소스가 있는지 알아보기

CA Access Control 을 구현할 때 경고 모드를 임시 방법으로 사용해야 합니다. 사용자들이 필요한 리소스에 대해 필요한 액세스를 가지고 있는 것으로 확인되면 경고 모드를 종료해야 합니다. 그러면 CA Access Control 이 관련된 규칙의 적용을 시작합니다.

경고 모드에 어떤 리소스가 있는지 알아보려면 경고 모드와 함께 모든 리소스를 보여주는 보고서를 작성할 수 있습니다.

보고서를 작성하려면 다음 명령을 입력합니다.

```
sereport -r 6
```

CA Access Control 이 보고서를 작성합니다.

참고: sereport 유틸리티에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

경고 모드에 있는 클래스 찾기

CA Access Control 을 구현할 때 경고 모드를 임시 방법으로 사용해야 합니다. 사용자들이 필요한 리소스에 대해 필요한 액세스를 가지고 있는 것으로 확인되면 경고 모드를 종료해야 합니다. 그러면 CA Access Control 이 관련된 규칙의 적용을 시작합니다.

경고 모드에 있는 클래스를 찾으려면 CA Access Control 이 이 데이터를 표시하게 합니다.

이 데이터를 표시하려면 다음 selang 명령을 입력합니다.

```
setoptions cwamlist
```

CA Access Control 이 경고 모드에 있는 클래스를 나타내는 테이블을 표시합니다.

참고: setoptions 에 대한 자세한 내용은 *selang 참조 안내서*를 참조하십시오.

감사 로그

감사 레코드는 감사 로그라는 파일에 저장됩니다. 감사 로그의 위치는 seos.ini 파일에 지정됩니다. seaudit 유틸리티나 CA Access Control 끝점 관리는 감사 로그에 기록된 이벤트를 열거하고 이벤트를 시간 제한 또는 이벤트 유형별로 필터링하는 등의 작업에 사용할 수 있습니다.

참고: seaudit 에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

감사 로그는 로컬로 저장되지만 CA Access Control 을 사용하면 로그 라우팅 기능을 통해 감사 정보를 배포할 수 있습니다. 기존 감사 로그를 테이프에 보관하면 나중에 이벤트를 스캔할 수 있습니다.

기본적으로 root 사용자가 seosd 프로그램을 실행하기 때문에 권한 부여 데몬 seosd 는 root 소유권으로 감사 로그를 작성합니다. 마찬가지로 root 에게만 부여된 읽기/쓰기 권한으로 감사 로그가 작성됩니다.

다른 사용자가 root 를 su(사용자 대체)를 실행하지 않고도 감사 로그를 읽을 수 있도록 하기 위해 CA Access Control 은 로그 파일에 할당되는 그룹 소유권을 지정하는 seos.ini 파일에 두 개의 항목을 포함합니다.

- 항목 하나는 감사 로그용입니다.

  사이트의 감사자가 auditforce 라는 그룹의 전체 멤버라고 가정합시다. 이러한 사용자들이 로컬 감사(Audit) 로그 파일을 탐색할 수 있도록 하려고 합니다. seos.ini 파일을 편집하여 [logmgr] 섹션에 있는 audit_log 토큰을 auditforce 로 설정합니다. 그러면 CA Access Control 에서 auditforce 그룹에게 로컬 감사 로그에 대한 읽기 권한을 부여합니다. 이 때부터 사용자 스테이션에서 생성되는 모든 로컬 감사 로그의 소유자는 auditforce 그룹이 됩니다.

  로그 라우팅 데몬은 동일한 토큰을 참조하여 데몬 자신이 생성하고 수집한 감사 로그에 대한 액세스 권한이 필요한 사용자를 확인합니다. 감사 로그는 다른 파일과 마찬가지로 액세스가 제어되며 CA Access Control 규칙에 따라 사용자가 감사 로그에 액세스하지 못할 수 있습니다.

- 다른 항목은 오류 로그에 관한 것으로, 해당 파일에 대한 그룹 소유권을 지정할 때 동일한 방식으로 사용됩니다.

시스템 감사자

시스템 감사자는 AUDITOR 특성이 할당되는 사용자입니다. 시스템 감사자로 정의된 사용자에게는 사용자와 리소스에 할당된 감사 특성 변경 등의 감사 작업을 수행할 수 있는 권한이 부여됩니다.

감사 작업은 중앙에서 수행될 수 있습니다. 단일 호스트에서 네트워크상에 있는 여러 스테이션의 감사 정보를 수집하려면 감사자가 로그 라우팅 기능을 사용해야 합니다.

로그 라우팅 기능 설정

로그 라우팅 기능을 설정하려면

1. 로그 라우팅 구성 파일을 생성합니다.

   seos.ini 파일에서 RouteFile 토큰에 대해 달리 지정하지 않는 한 CA Access Control 은 사용자의 로그 라우팅 구성 파일 이름이 *ACInstallDir*/log/selogrd.cfg 일 것으로 예상합니다.

   여기서 *ACInstallDir* 은 CA Access Control 의 설치 디렉터리이고 기본값은 /opt/CA/AccessControl/입니다.

   ACInstallDir/samples/selogrd.init 디렉터리에서 샘플 로그 라우팅 구성 파일을 찾을 수 있습니다. 또는 다음 세 개의 명령줄로 구성된 간단한 로그 라우팅 구성 파일을 작성할 수 있습니다.

   규칙
   host destination
   .

   destination 에는 감사 레코드를 수신하게 될 호스트의 이름을 입력합니다. 모든 클래스, 리소스, 접근자 및 결과가 기록됩니다.

   참고: 구성 파일 구문에 대한 자세한 내용은 *Utilities Guide* 의 selogrd 유틸리티를 참조하십시오.

2. 감사 정보를 라우팅할 모든 호스트에서 에미터 데몬(emitter)을 시작하고, 감사 정보를 수집할 모든 호스트에서 수집기 데몬(collector)을 실행합니다.

   참고: 이러한 데몬 사용에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

파일 통보

로그 컴파일링 기능 이외에 로그 라우팅 기능을 사용하여 알림을 호스트의 디스플레이 화면, 전자 메일 주소 또는 기타 대상으로 보낼 수 있습니다. 스테이션의 감사 로그 또는 컬렉터 데몬이 스테이션으로 보낸 로그의 정보를 기반으로 알림을 보낼 수 있습니다.

그러한 통보를 설정하려면 로그 라우팅 구성 파일 *및* selang 명령을 사용해야 합니다. 예를 들어, root 사용자에 대한 setuid 요청이 성공적으로 수행될 때 이를 사용자 John 에게 알리려고 하는 경우 다음을 수행합니다.

1. 다음 selang 명령을 실행합니다.

   ```
   chres SURROGATE USER.root notify(John)
   ```

   여기서 chres 명령은 누군가가 root 로 사용자를 서로게이트할 때마다 특수 감사 로그 레코드가 작성되고 seosd 데몬이 John 이라는 사용자에게 알리도록 지정합니다. 또한 데몬은 *알림 레코드*라는 특수한 감사 레코드를 생성합니다.

2. 하나 이상의 리소스에 대한 통보를 지정한 경우에는 로그 라우팅 구성 파일에 다음 3 행을 추가할 수 있습니다.

   ```
   Rule2
   notify default
   .
   ```

   이 행은 로그 라우팅 에미터가 알림 감사 레코드에 대한 메일 메시지를 작성하도록 합니다.

   참고: 구성 파일 형식과 로그 라우팅 데몬 설정 방법에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

로그 라우팅

CA Access Control 은 로그 라우팅 데몬인 selogrd 를 사용하여 선택한 로컬 감사 로그 레코드를 특정 호스트에 배포하고, 감사 로그 레코드 형식을 전자 메일 메시지, ASCII 파일 또는 사용자 창으로 변경하고, 감사된 이벤트에 따라 알림 메시지를 전송합니다.

감사 레코드 라우팅을 결정하기 위해 selogrd 는 구성 파일인 selogrd.cfg 를 사용합니다. 이 파일은 라우팅하거나 라우팅하지 않을 감사 로그 레코드와 라우팅 위치를 나타내는 목록입니다. 이 파일에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

로그 라우팅 구성

seosd 가 시작될 때 selogrd 또는 selogrcd 를 자동으로 시작하려면 [daemons] 섹션의 seos.ini 토큰 selogrd 또는 selogrcd 를 yes 로 설정하십시오. 그런 다음 seload 를 실행하면 seload 가 데몬을 시작합니다.

예를 들어 soes.ini 의 [daemons] 섹션에 있는 해당 토큰은 다음과 같습니다.

```
selogrd = yes
selogrcd = yes
```

로그 라우팅 기능은 RPC 를 사용하여 감사 레코드를 라우팅하므로 방화벽 뒤에 로그 감사 수집기 위치를 지정하면 portmapper 가 서버 데몬에 할당하는 포트를 알 수 없기 때문에 UDP 포트의 단순한 차단이 허용되지 않습니다. 이 문제를 해결하기 위해 ServicePort 토큰을 사용하여 미리 정의된 포트를 서버 데몬에 할당할 수 있습니다.

방화벽이 네트워크 외부 포트 111(portmapper 포트)을 허용할 경우 서버에 있는 seos.ini 파일만 변경해야 합니다. 방화벽이 보호된 네트워크의 portmapper 에 대한 통신을 허용하지 않을 경우 클라이언트와 서버는 모두 지정된 포트에서 동일해야 합니다.

클라이언트와 서버 둘 다의 seos.ini 파일에서 ServicePort 토큰에 동일한 값을 설정하여 이 문제를 해결할 수 있습니다. 데몬이 지정된 포트로 바인딩됨을 의미하는 숫자를 지정하거나 서비스 이름을 지정할 수 있습니다. 서비스 이름을 지정할 경우 클라이언트 및 서버는 모두 동일한 서비스를 확인해야 합니다. 예를 들어 서비스 이름으로 seoslogr 을 지정하는 경우 클라이언트 및 서버의 /etc/services 파일에 다음 명령을 추가하십시오.

```
seoslogr 2022/udp # Audit log-routing
```

클라이언트 및 서버가 NIS 를 사용하여 서비스를 확인하는 경우 NIS 서비스 맵을 업데이트해야 합니다.

감사 로그 라우팅 암호화

감사 로그 레코드를 암호화할 수 있습니다. 암호화를 사용할 때 selogrd 데몬은 수집기(selogrcd 또는 감사 로그 라우터)로 보내기 전에 감사 로그 레코드를 암호화합니다. 그런 다음 수집기는 수신된 레코드를 해독합니다.

CA Access Control 에서는 selogrd 에 대한 두 가지 암호화 스타일인 CA Access Control 표준 암호화 및 adcipher 를 통한 감사 로그 암호화를 제공합니다. 암호화하는 경우 selogrd 는 seos.in 파일의 [selogrd] 섹션에 지정된 것처럼 공유 라이브러리 객체의 함수를 사용합니다.

표준 암호화는 공유 라이브러리인 libcrypt 를 사용하고 감사 암호화는 CipherName 토큰에 지정된 파일의 함수를 사용합니다. 기본적으로 파일 이름은 adcipher 로, 원하는 공유 라이브러리에 대한 심볼릭 링크입니다. CA Access Control 설치 프로세스 중에 lib1des, lib3des, libIDEA 및 libblowfish 의 공유 라이브러리 네 개가 CA Access Control/lib 디렉터리에 저장됩니다.

CA Access Control 은 공유 라이브러리의 표준 암호화 키를 유지 관리하는 반면, 감사 암호화는 KeyFile 토큰에 의해 지정된 대로 개별 파일을 사용합니다(기본값: adcipher.bin).

UseEncryption 토큰을 사용하여 암호화 유형을 결정합니다.

- CA Access Control 표준 암호화를 사용하려면 UseEncryption=native 를 지정합니다.
- adcipher 를 통해 감사 로그 암호화를 사용하려면 UseEncryption=eTrust 를 지정하고 CipherName 및 KeyFile 토큰에 적합한 값을 입력하십시오.
- selogrd 암호화를 비활성화하려면 UseEncryption=no 를 지정하십시오.

RefuseUnencrypted 토큰을 사용하여 암호화되지 않은 감사를 허용하거나 거부합니다. 이 토큰은 UseEncryption 토큰과 함께 사용되고 UseEncryption 이 no 로 설정된 경우에는 중복됩니다.

- 암호화되지 않은 감사를 거부하려면 RefuseUnencrypted=yes 를 지정합니다.
- 암호화된 감사와 암호화되지 않은 감사를 모두 허용하려면 RefuseUnencrypted=no 를 지정합니다.

참고: selogrcd 데몬은 seos.ini 파일에 있는 동일한 토큰을 사용합니다.

암호화 키를 변경하려면 이 장의 sechkey 유틸리티를 사용하십시오.

중요! 레코드를 감사 수집기로 보낼 경우 selogrd 와 수집기가 모두 동일한 공유 암호화 파일과 암호화 키를 사용해야 합니다.

전자 메일을 사용하여 감사 로그 레코드 전송

selogrd 는 전자 메일 대상으로 직접 레코드를 전송할 수 있습니다. 메일 프로그램(mailer) 유틸리티를 통해 전자 메일 메시지를 보내거나(이전 방법), SMTP 를 사용하여 직접 메일 교환 서버로 전자 메일 메시지를 보낼 수 있습니다.

감사 로그 레코드를 메일 교환 서버에 직접 보내려면 seos.ini 파일의 [selogrd] 섹션에 있는 UseSmtpMail 토큰을 설정합니다.

또한 다음 항목을 지정할 수 있습니다.

- StmpTimeLimit 토큰을 사용할 때 메일 서버가 응답하지 않는 경우 시간 제한
- SmtpMailFrom 토큰을 사용할 때 "From:" 메일 헤더 필드
- SmtpMailServer 토큰을 사용할 때 메일 서버 호스트 주소

참고: 이 방법에서는 UNIX 메일 유틸리티를 사용하지 않습니다. 그 대신 메일 서버와의 직접 연결을 설정하고 SMTP 프로토콜을 사용하여 메일을 보냅니다.

SNMP 트랩 구성

인터넷 네트워크 관리 프로토콜 SNMP(Simple Network Management Protocol)를 사용하는 시스템에 대해 CA Access Control 감사 레코드를 사용하여 SNMP 트랩을 생성하도록 selogrd 를 구성할 수 있습니다.

SNMP 트랩을 구현하려면 먼저 CA Access Control 라이브러리에 제공된 SNMP 공유 객체를 찾은 다음 이러한 공유 객체를 사용하여 selogrd 를 올바르게 구성합니다.

참고: selogrd 의 SNMP 확장을 사용하려는 경우 CA Access Control 이 기본 위치(/opt/CA/AccessControl/)에 설치되어 있지 않으면 selogrd 를 실행하기 전에 환경 변수를 설정하십시오. 환경 변수는 다음과 같습니다. 여기서 *ACInstallDir* 는 CA Access Control 을 설치한 디렉터리입니다.

- AIX 의 경우 LIBPATH 를 *ACInstallDir*/lib 로 설정합니다.
- Solaris 의 경우 LD_LIBRARY_PATH 를 *ACInstallDir*/lib 로 설정합니다.
- LINUX 의 경우 LD_LIBRARY_PATH 를 *ACInstallDir*/lib 로 설정합니다.
- HP 의 경우 SHLIB_PATH 를 *ACInstallDir*/lib 로 설정합니다.

일반적으로 *ACInstallDir*/lib 디렉터리에 있는 공유 개체는 snmp.*xx* 및 libsnmp.*xx* 라고 하며 *xx* 확장자는 플랫폼에 따라 다릅니다. 가능한 확장자는 다음과 같습니다.

- **.o** - AIX 플랫폼
- **.sl** - HP 플랫폼
- **.so** - 기타 모든 플랫폼

CA Access Control 이 기본 위치에 설치되지 않고 SNMP 확장 selogrd 를 사용하려면 selogrd 를 실행하기 전에 다음과 같이 환경 변수를 설정해야 합니다.

- AIX 의 경우 LIBPATH 를 *ACInstallDir*/lib 로 설정합니다.
- Solaris 의 경우 LD_LIBRARY_PATH 를 *ACInstallDir*/lib 로 설정합니다.
- LINUX 의 경우 LD_LIBRARY_PATH 를 *ACInstallDir*/lib 로 설정합니다.
- HP 의 경우 SHLIB_PATH 를 *ACInstallDir*/lib 로 설정합니다.

여기서 *ACInstallDir* 은 CA Access Control 을 설치한 디렉터리입니다.

공유 개체를 사용하도록 selogrd 를 구성하려면

1. *ACInstallDir*/etc/selogrd.ext 파일을 생성합니다.
2. snmp.so 에 대한 적절한 경로가 포함된 *ACInstallDir*/etc/selogrd.ext 파일에 단일 줄을 추가하여 SNMP 공유 개체가 있는 위치를 정의합니다. 다른 공유 개체를 자동으로 연결하려면 이 공유 개체만 지정하면 됩니다. 예:

   ```
   snmp /opt/CA/AccessControl//lib/snmp.so
   ```

3. 마지막으로 selogrd.cfg 파일을 구성하여 SNMP 트랩을 트리거하는 동작 유형과 SNMP 트랩이 트리거될 때 통보할 위치를 지정해야 합니다. 구성은 다른 감사 알림을 위한 snmp 로 지정된 배달 시스템을 갖춘 구성과 매우 유사합니다.

   예를 들어 CA Access Control 이 시작 및 종료될 때 SNMP 트랩을 활성화하고 이러한 SNMP 트랩에 대한 알림을 AuditPC 로 전송한다고 가정합니다. 이렇게 하려면 selogrd.cfg 구성 파일에 다음 섹션을 추가할 수 있습니다.

   ```
   snmpRule
   snmp AuditPC
   include Class(START).
   include Class(SHUTDOWN).
   .
   ```

마찬가지로 SNMP 트랩을 다른 작업이나 액세스 유형별로 활성화하거나 이러한 SNMP 트랩을 다른 위치로 보낼 수 있습니다.

사용자 추적 필터 마이그레이션

사용자를 추적 가능하게 설정하면, 해당 사용자에 대해 추적 레코드가 기록될 때마다 seos.audit 파일에 일치하는 감사 레코드가 기록됩니다. CA Access Control 의 이전 릴리스에서는 이러한 감사 레코드가 trcfilter.init 파일에 의해 필터링되었습니다. CA Access Control r12.0 SP1 및 이후 버전에서는 사용자 추적 레코드에 의해 생성되는 감사 레코드가 audit.cfg 파일(이 파일은 다른 모든 감사 레코드를 필터링함)에 의해 필터링됩니다.

감사 레코드 필터를 trcfilter.init 에서 audit.cfg 로 수동으로 마이그레이션해야 합니다. 필터를 마이그레이션하지 않으면 사용자 추적에 의해 생성되는 감사 레코드가 필터링되지 않습니다.

참고: 추적 레코드는 계속 trcfilter.init 에 의해 필터링되므로 추적 필터를 trcfilter.init 에서 audit.cfg 로 마이그레이션하지 마십시오.

사용자 추적 필터를 마이그레이션하려면

1. trcfilter.init 에서 마이그레이션해야 하는 사용자 추적 필터를 찾습니다.

   seos.ini 파일의 seosd 섹션에 있는 trace_filter 설정에 따라 이 파일의 위치가 결정됩니다.

2. audit.cfg 에서 다음을 입력합니다. 여기에서 *usertracefilter* 는 trcfilter.init 의 사용자 추적 필터입니다.

   TRACE;*;*;*;*;*usertracefilter*

3. (선택 사항) 마이그레이션해야 하는 각 사용자 추적 필터에 대해 1-2 단계를 반복합니다.

예: 사용자 추적 필터 마이그레이션

이 예에서 다음 사용자 추적 필터는 trcfilter.init 파일에 있습니다.

```
*ExampleFilter
```

이 사용자 추적 필터를 마이그레이션하려면 audit.cfg 파일의 새 행에 다음을 입력하십시오.

```
TRACE;*;*;*;*;*ExampleFilter
```

제 14 장: 관리 인증 범위

이 섹션은 다음 항목을 포함하고 있습니다.

전역 권한 부여 특성

전역 권한 부여 특성은 사용자 레코드에 설정됩니다. 각 전역 권한 부여 특성을 사용하여 사용자가 특정 유형의 기능을 수행할 수 있습니다. 이 단원에서는 각 전역 권한 부여 특성의 기능과 제한에 대해 설명합니다.

ADMIN 특성

ADMIN 특성을 사용하면 CA Access Control 에서 대부분의 명령을 실행할 수 있습니다. ADMIN 특성을 가진 데이터베이스에 정의된 사용자는 데이터베이스에서 사용자, 그룹 및 리소스를 정의하고 업데이트할 수 있습니다. 이 특성은 CA Access Control 에서 가장 강력한 특성이지만 다음과 같은 제한이 있습니다.

- 데이터베이스에서 한 명의 사용자만 ADMIN 특성을 가지는 경우 해당 사용자를 삭제할 수 없고 ADMIN 특성을 레코드에서 제거할 수 없습니다.
- ADMIN 특성은 있지만 AUDITOR 특성이 없는 사용자는 사용자, 그룹 또는 리소스에서 실행된 감사 유형(감사 모드)을 변경할 수 없습니다. ADMIN 특성이 있으며 사용자, 그룹 또는 리소스의 감사 특성을 변경해야 하는 경우 사용자 자신에게 AUDITOR 특성을 할당합니다.
- ADMIN 특성이 있는 사용자는 슈퍼 사용자(UNIX 의 root 계정 또는 Windows 의 Administrator 계정)를 삭제할 수 없지만 root 를 ADMIN 이 아닌 사용자로 설정할 수 있습니다.

AUDITOR 특성

AUDITOR 특성을 가진 사용자는 시스템 사용을 모니터할 수 있습니다. AUDITOR 특성을 가진 사용자의 명시된 권한은 다음과 같습니다.

- 사용자는 데이터베이스에 정보를 표시할 수 있습니다.

  감사자는 selang 명령 *showusr*, *showgrp*, *showres* 및 *showfile* 을 실행할 수 있습니다.

- 사용자는 기존 레코드에 대한 감사 모드를 설정할 수 있습니다.

  감사자는 selang 명령 *chusr*, *chgrp*, *chres* 및 *chfile* 을 실행할 수 있습니다.

OPERATOR 특성

OPERATOR 특성을 가진 사용자는 모든 파일에 대한 READ 권한이 있습니다. 이 권한을 사용하면 데이터베이스의 모든 내용을 나열할 수 있을 뿐만 아니라 백업 작업을 실행할 수 있습니다. 운영자는 showusr, showgrp, showres, showfile, find 명령을 사용하여 데이터베이스 레코드를 나열할 수 있습니다. OPERATOR 특성은 사용자가 secons 유틸리티를 사용하게 합니다.

참고: secons 유틸리티에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

PWMANAGER 특성

PWMANAGER 특성은 일반 사용자에게 chusr 또는 sepass 명령을 사용하여 다른 사용자의 암호를 변경할 수 있는 권한을 제공합니다.

참고: PWMANAGER 가 ADMIN 사용자의 암호를 변경하게 하려면 setoptions 명령의 *cng_adminpwd* 옵션을 설정합니다. 자세한 내용은 *selang 참조 안내서*를 참조하십시오.

PWMANAGER 권한에는 showusr 및 find 명령의 사용도 포함됩니다.

PWMANAGER 의 권한에는 showusr 및 find 명령의 사용이 포함됩니다.

참고: 사용자의 nochngpass 속성이 yes 로 설정된 경우 PWMANAGER 는 해당 사용자의 암호를 변경할 수 없습니다.

SERVER 특성

많은 다른 보안 모델과 같이 CA Access Control 은 일반 사용자에게 "사용자 A 가 리소스 X 에 액세스할 수 있습니까?"와 같은 질문을 허용하지 않으며 유일하게 허용되는 질문은 "리소스 X 에 내가 액세스할 수 있습니까?"입니다. 그러나 데이터베이스 서버 서비스나 내부 응용 프로그램과 같이 많은 사용자에게 서비스를 제공하는 프로세스는 다른 사용자를 위한 권한 부여를 요청할 수 있게 허용되어야 합니다.

SERVER 특성을 사용하면 프로세스에서 사용자에 대한 권한 부여를 요청할 수 있습니다. SERVER 특성을 가진 사용자는 SEOSROUTE_VerifyCreate API 를 실행할 수 있습니다.

참고: 서버 특성 및 CA Access Control API 에 대한 자세한 내용은 *SDK 안내서*를 참조하십시오.

IGN_HOL 특성

IGN_HOL 특성을 통해 사용자는 holiday 레코드에 정의된 기간에는 언제든지 로그인할 수 있습니다. HOLIDAY 클래스의 각 레코드는 사용자가 로그인하기 위해 추가 권한이 필요할 때 하나 이상의 기간을 정의합니다. IGN_HOL 특성을 사용하면 holiday 레코드에 정의된 기간과 관계없이 사용자가 언제든지 로그인할 수 있습니다.

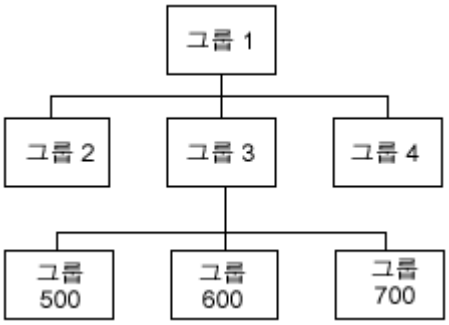
참고: HOLIDAY 클래스에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

그룹 권한 부여

그룹 권한 부여 특성을 설명하기 전에 부모-자식 관계의 개념을 이해할 필요가 있습니다.

부모-자식 관계

부모-자식 관계라고도 하는 종속 및 부모 그룹의 개념은 그룹 관리 권한을 설명할 때 중요합니다. 하나의 그룹은 하나 이상의 그룹의 부모(상위 그룹)가 될 수 있습니다. *child* 또는 종속 그룹은 하나의 parent 만을 포함할 수 있습니다. 그룹에 부모를 할당하는 것은 선택 사항입니다. 다음 그림을 참조하십시오.



그룹 1 은 세 그룹(20, 30, 40)의 부모 그룹입니다. 그룹 30 은 세 그룹(500, 600, 700)의 부모 그룹입니다. 그룹 600 은 그룹 30 의 유일한 부모 그룹입니다. 그룹 1 에는 부모 그룹이 없습니다.

그룹 권한 부여 특성

리소스 레코드와 접근자 레코드 등을 포함한 모든 레코드에는 소유자가 있습니다. 레코드를 소유하는 것은 레코드를 표시, 편집 및 제거할 권한을 갖는 것입니다.

그룹은 고유의 레코드를 소유할 수 있습니다. 하지만 레코드를 소유하는 그룹에서 특정 권한을 부여 받은 사용자만 레코드를 관리할 수 있습니다. 이러한 특별한 사용자들은 자신의 사용자 레코드에 그룹 권한 부여 특성 세트를 갖습니다. 그룹 권한 부여 특성은 다음과 같습니다.

- GROUP-ADMIN
- GROUP-AUDITOR
- GROUP-OPERATOR
- GROUP-PWMANAGER

권한을 부여받은 사용자만이 실행할 수 있는 join 명령으로 이 특성을 설정합니다. join 명령은 사용자를 그룹에 할당하고 사용자의 그룹 권한 부여 특성(있는 경우)을 지정합니다.

GROUP-ADMIN 특성

추가 정보:

소유권 (페이지 221)

GROUP-ADMIN 특성

그룹 관리 권한 부여 특성을 가진 사용자는 레코드 세트를 생성할 수 있습니다. 레코드를 생성하려면 그룹 관리자가 레코드의 소유자를 지정해야 합니다.

사용자가 그룹 권한 부여 특성을 갖고 있는 그룹이 레코드의 소유자가 되어야 합니다. 이 그룹이 다른 그룹의 부모이면 소유자도 하위 그룹 중 하나에 속할 수 있습니다. 전체 레코드 세트를 그룹 범위라고 합니다. 제공된 권한 부여 예에서는 그룹 범위 개념을 설명합니다.

GROUP-ADMIN 특성을 가진 사용자는 자신의 그룹 범위 내의 레코드에 대해 다음과 같은 액세스 권한을 가집니다.

액세스	설명	명령
읽기	레코드 속성을 표시합니다.	showusr, showgrp, showres, showfile
작성	데이터베이스에 새 레코드를 생성합니다. 소유자를 지정해야 합니다.	newusr, newgrp, newres, newfile
수정	레코드 속성을 변경합니다.	chusr, chgrp, chres, chfile
삭제	데이터베이스에서 레코드를 제거합니다.	rmusr, rmgrp, rmres, rmfile
연결	사용자를 그룹에 조인하거나 그룹에서 사용자를 분리합니다.	join, join-

GROUP-ADMIN 특성은 또한 다음과 같은 제한이 있습니다.

- GROUP-ADMIN 사용자는 리소스가 자신에게 액세스할 수 없도록 만들 수 없습니다. 따라서,
  - GROUP-ADMIN 사용자는 자신의 보안 수준보다 높은 보안 수준을 할당할 수 없습니다.
  - GROUP-ADMIN 사용자는 자신이 소유하지 않은 보안 범주나 보안 레이블을 할당할 수 없습니다.
- GROUP-ADMIN 사용자는 데이터베이스에서 슈퍼 사용자 사용자(UNIX 의 root 계정 또는 Windows 의 Administrator 계정)를 삭제할 수 없습니다.
- 일부 제한 사항이 이 장의 전역 권한 부여 특성에서 설명된 전역 권한 부여 특성에 적용됩니다.
  - GROUP-ADMIN 사용자는 데이터베이스에서 ADMIN 사용자 레코드만 삭제할 수 없습니다.
  - GROUP-ADMIN 사용자는 데이터베이스에 있는 마지막 ADMIN 사용자의 레코드에서 ADMIN 속성을 제거할 수 없습니다.
  - AUDITOR 속성이 없는 GROUP-ADMIN 사용자는 감사 모드를 업데이트할 수 없습니다. AUDITOR 속성이 있는 GROUP-ADMIN 사용자만 감사 모드를 업데이트할 수 있습니다.
  - GROUP-ADMIN 사용자는 모든 사용자에 대해 ADMIN, AUDITOR, OPERATOR, PWMANAGER 및 SERVER 의 전역 권한 부여 특성을 설정할 수 없습니다.

GROUP-AUDITOR 특성

GROUP-AUDITOR 속성을 가진 사용자는 그룹 범위 내의 모든 레코드에 대한 속성을 나열할 수 있습니다. 또한 그룹 감사자는 그룹 범위에 속해 있는 모든 레코드에 대해 감사 모드를 설정할 수 있습니다.

GROUP-OPERATOR 특성

GROUP-OPERATOR 속성을 가진 사용자는 그룹 범위 내의 모든 레코드에 대한 속성을 나열할 수 있습니다.

GROUP-PWMANAGER 특성

GROUP-PWMANAGER 특성을 가진 사용자는 그룹 범위에 속해 있는 레코드를 소유한 사용자의 암호를 변경할 수 있습니다.

소유권

데이터베이스의 모든 레코드(접근자 레코드 및 리소스 레코드 포함)에는 소유자가 있습니다. 데이터베이스에 레코드를 추가할 때 소유자 매개 변수를 사용하여 명시적으로 소유자를 할당할 수도 있고 CA Access Control 가 레코드 소유자로서 레코드를 정의하는 사용자를 할당하도록 할 수도 있습니다.

다음 중 해당 사항이 *하나라도* 있는 경우 접근자가 레코드를 소유합니다.

- 접근자가 레코드의 소유자로 정의되었습니다.
- 접근자가 레코드의 소유자로 정의된 그룹의 구성원이고 GROUP-ADMIN 속성이 있는 그룹에 속합니다.
- 접근자가 해당 리소스가 구성원인 리소스 그룹 레코드의 소유자입니다.

데이터베이스에서 레코드를 소유하는 사용자나 그룹을 제거하면 레코드는 더 이상 소유자를 갖지 않습니다.

레코드를 소유하는 사용자는 자신들이 소유한 레코드에 대해 다음과 같은 액세스 권한을 가집니다.

액세스	설명	명령
읽기	레코드 속성을 표시합니다.	showusr, showgrp, showres, showfile
수정	레코드 속성을 변경합니다.	chusr, chgrp, chres, chfile
삭제	데이터베이스에서 레코드를 제거합니다.	rmusr, rmgrp, rmres, rmfile
연결	사용자를 그룹에 조인하거나 그룹에서 사용자를 분리합니다.	join, join-

사용자 또는 그룹이 특정 레코드에 대한 소유 권한을 갖도록 하고 싶지 않으면 해당 레코드 및 해당 레코드가 구성원으로 속한 모든 리소스 그룹 레코드에 소유자 *nobody* 를 할당하십시오.

소유권의 제한 사항은 다음과 같습니다.

- 데이터베이스에서 마지막 ADMIN 사용자의 소유자는 해당 사용자 레코드를 삭제할 수 없습니다.
- AUDITOR 특성이 없는 소유자는 감사(Audit) 모드를 업데이트할 수 없습니다. AUDITOR 특성이 있는 소유자만 감사(Audit) 모드를 업데이트할 수 있습니다.
- 슈퍼 사용자(UNIX 의 root 계정 또는 Windows 의 Administrator 계정)의 소유자는 데이터베이스에서 root 를 삭제할 수 없습니다.
- 소유자는 액세스할 수 없는 리소스를 만들 수 없습니다. 따라서,
- 소유자는 액세스할 수 없는 리소스를 만들 수 없습니다. 따라서,
  - 소유자는 자신의 보안 수준보다 높은 보안 수준을 할당할 수 없습니다.
  - 소유자는 자신이 갖고 있지 않은 보안 범주나 보안 레이블을 할당할 수 없습니다.

파일 소유권

CA Access Control 은 파일 소유자가 FILE 클래스에서 레코드를 정의하여 파일을 보호하게 합니다. 파일의 소유자는 해당 파일의 레코드에 대해 모든 권한을 가지기 때문에 newfile, chfile, showfile, authorize 및 authorize- 명령을 해당 레코드에 대한 모든 매개 변수와 함께 사용하여 파일을 보호할 수 있습니다.

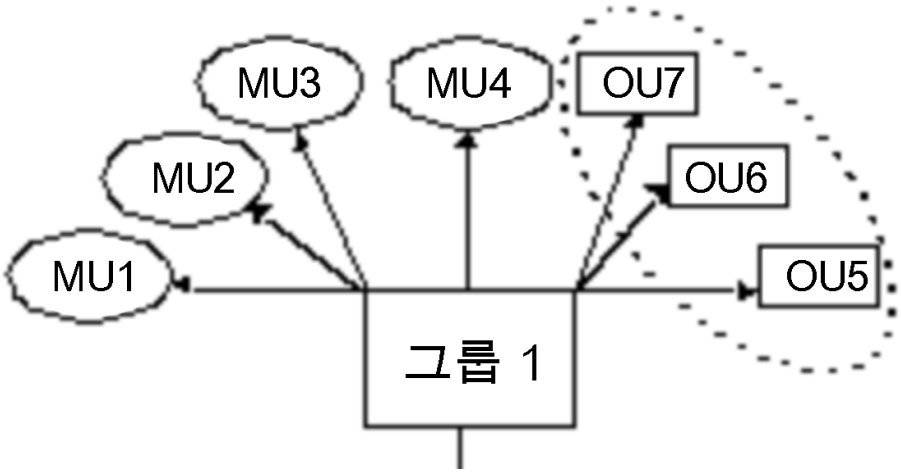
UNIX 의 경우 사용자가 파일을 생성하면 UNIX 는 그 사용자를 파일 소유자로 할당합니다. CA Access Control 에서는 이 기능이 명시적으로 비활성화되지 않는 한 UNIX 파일 소유자가 FILE 레코드를 정의할 수 있습니다. 파일 소유자가 파일(FILE) 레코드를 정의하지 못하도록 하려면 seos.ini 파일에서 [seos] 섹션의 use_unix_file_owner 토큰을 no 로 설정해야 합니다(기본 설정).

권한 부여 예제

다음은 그룹 권한 부여 특성, 상위 관계, 소유권, 구성원, 그룹 범위 등의 개념을 설명하는 다이어그램입니다. 이러한 다이어그램에서 특정 사용자와 그룹만 설명하지만 소유권의 개념은 리소스 및 파일 레코드에도 적용됩니다.

단일 그룹 권한 부여

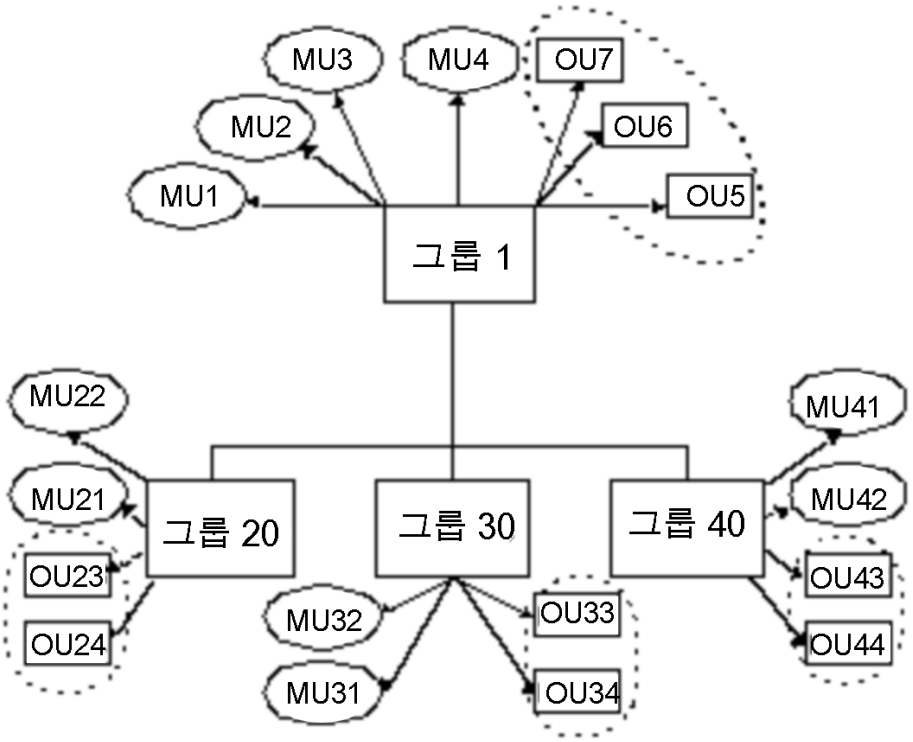
다음 다이어그램에서 네 사용자(MU1, MU2, MU3, MU4)는 그룹 1 의 구성원입니다. 또한 그룹 1 은 세 사용자(OU5, OU6, OU7)를 소유합니다. 구성원 MU4 는 GROUP-ADMIN 특성을 가집니다.



타원은 사용자 MU4 가 실행한 명령의 그룹 범위를 나타냅니다. 또한 그룹 1 이 소유한 모든 사용자(OU5, OU6, OU7)를 포함합니다.

부모 및 자식 그룹

다음 다이어그램에서 네 사용자(MU1, MU2, MU3, MU4)는 그룹 1 의 구성원입니다. 또한 그룹 1 은 세 사용자(OU5, OU6, OU7)를 소유합니다. 구성원 MU4 는 레코드에 설정된 GROUP-ADMIN 특성을 가집니다.



그룹 1 은 세 그룹(20, 30, 40)의 부모 그룹입니다. 각 하위 그룹에는 그룹의 구성원인 두 명의 사용자와 그룹에서 소유한 두 명의 사용자가 있습니다.

네 개의 타원은 사용자 MU4 가 실행하는 명령의 그룹 범위를 나타냅니다. 또한 그룹 1 이 소유한 모든 사용자와 그룹 1 의 하위 그룹에서 소유한 사용자를 포함합니다. MU4 의 그룹 범위에 속하는 사용자는 OU5, OU6, OU7, OU23, OU24, OU33, OU34, OU43 및 OU44 입니다.

사용자, 그룹 또는 리소스를 소유한 그룹 20, 30 또는 40 에 하위 그룹이 있는 경우 이러한 그룹이 소유한 레코드도 사용자 MU4 가 실행하는 명령 그룹 범위에 속합니다.

하위 관리

보안 관리자(ADMIN 특성을 가진 사용자)는 일반 사용자에게 특정 관리 권한을 부여할 수 있습니다. 이러한 일반 사용자를 하위 관리자라고 부릅니다. 하위 관리자는 지정된 CA Access Control 클래스 또는 개체만 관리할 수 있는 권한을 가지고 있습니다. 예를 들어, 하위 관리자는 사용자와 그룹 개체만 관리할 수 있는 권한을 부여받을 수 있습니다. 하위 관리자에게 클래스의 특정 개체에 대한 관리 권한을 부여하여 보다 높은 수준의 하위 관리를 설정할 수 있습니다.

사용자, 그룹 및 리소스의 하위 관리자는 selang 을 사용하여 이러한 리소스와 관련된 관리 작업을 수행할 수 있습니다.

일반 사용자에게 특정 관리 권한을 부여하는 방법

ADMIN 특성을 가진 관리자 사용자는 CA Access Control 에서 거의 모든 작업을 실행할 수 있으므로 특정 관리 작업을 하위 관리자에게 위임할 수 있습니다. 이렇게 하려면 다음과 같이 CA Access Control 데이터베이스에서 사용자가 수행해야 하는 특정 관리 작업을 제어하는 클래스에 대한 권한을 사용자에게 부여해야 합니다.

1. 위임할 작업을 제어하는 하나 이상의 클래스를 식별합니다.

   예를 들어 CA Access Control 에서는 USER 및 GROUP 클래스를 사용하여 접근자 리소스를 생성합니다. 접근자 관리를 위임하려면 ADMIN 클래스의 USER 및 GROUP 레코드를 사용해야 합니다.

2. 한 명 이상의 하위 관리자에게 ADMIN 클래스의 적용 가능한 리소스에 대한 권한을 부여합니다.

   예를 들어 하위 관리자가 사용자 레코드를 표시 및 수정하게 하려면 사용자에게 ADMIN 클래스의 USER 레코드에 대한 *읽기* 및 *수정* 액세스 권한을 부여합니다.

ADMIN 클래스

ADMIN 클래스에서 레코드의 ACL(접근자 제어 목록)에 나열된 사용자인 하위 관리자는 ADMIN 특성을 가진 사용자와 유사한 권한을 갖습니다. 그러나 클래스에 의한 액세스(ADMIN) 클래스의 레코드에 대한 액세스 제어 목록(ACL)에 있는 사용자 권한은 레코드가 표시하는 특정 클래스로 제한됩니다. 예를 들어 클래스에 의한 액세스 클래스의 사용자 ID 교체(SURROGATE) 레코드는 사용자 ID 교체(SURROGATE) 클래스의 레코드를 관리할 수 있는 사용자를 결정합니다.

참고: CA Access Control 클래스에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

ACL 에 있는 사용자는 ADMIN 클래스의 특정 레코드에 대해 다음 명령을 실행할 수 있습니다.

액세스	설명	명령
읽기	클래스에서 레코드 속성을 표시합니다.	showusr, showgrp, showres, showfile, find
작성	클래스에서 새 데이터베이스 레코드를 작성합니다.	newusr, newgrp, newres, newfile
수정	클래스에서 속성을 변경합니다.	chusr, chgrp, chres, chfile
삭제	데이터베이스에서 기존 클래스 레코드를 제거합니다.	rmusr, rmgrp, rmres, rmfile
연결	그룹에서 사용자를 추가 및 제거합니다. 이 액세스는 GROUP 레코드의 ACL 에서만 유효합니다.	join, join-
암호	데이터베이스 내부의 모든 사용자 암호와 암호 특성을 제어합니다. 이 액세스는 PWMANAGER 특성이 있는 사용자에게 허용되는 액세스와 동일한 권한을 부여합니다. 이 액세스는 USER 레코드의 ACL 에서만 유효합니다.	chusr

ADMIN 클래스 권한이 있는 사용자에게는 다음과 같은 제한 사항이 있습니다.

- ADMIN 클래스에 있는 USER 레코드의 ACL 에 정의된 사용자는 데이터베이스의 마지막 ADMIN 사용자를 삭제할 수 없습니다.
- ADMIN 클래스 사용자는 자신이 소유한 사용자에 대한 전역 권한 특성인 ADMIN, AUDITOR, OPERATOR 및 PWMANAGER 을 설정할 수 없습니다.
- 모든 ADMIN 클래스 사용자가 감사 모드를 업데이트할 수 있는 것은 아닙니다. AUDITOR 특성을 가진 ADMIN 클래스 사용자만 감사 모드를 업데이트할 수 있습니다.
- ADMIN 클래스 사용자는 슈퍼 사용자(UNIX 의 root 계정 또는 Windows 의 Administrator 계정)를 삭제할 수 없지만 root 를 NOADMIN 으로 설정할 수 있습니다.
- ADMIN 클래스 사용자는 액세스할 수 없는 리소스를 만들 수 없습니다. 따라서,
  - ADMIN 클래스 사용자는 자신의 보안 수준보다 높은 보안 수준을 할당할 수 없습니다.
  - ADMIN 클래스 사용자는 자신이 갖고 있지 않은 보안 범주나 보안 레이블을 할당할 수 없습니다.

이러한 제한 사항은 B1 보안 수준 인증의 일부입니다.

환경 고려 사항

데이터베이스에서 정보를 업데이트할 수 있는지 여부를 결정하는 요인들 중 하나는 사용자의 직위입니다.

원격 관리 제한 사항

네트워크를 통해 원격 스테이션에 액세스하고 원격 스테이션에서 데이터베이스를 업데이트할 수 있습니다. 원격 스테이션 상의 데이터베이스를 업데이트하려면 사용자와 사용자의 터미널 모두에게 권한이 있어야 합니다.

- 사용자는 원격 스테이션의 데이터베이스에서 명시적으로 사용자로 정의되어야 합니다. 실행하려는 명령에 관계없이 원격 스테이션의 데이터베이스에 있는 사용자 레코드에 적절한 특성을 설정해야 합니다.
- 원격 스테이션에 액세스할 수 있는 쓰기 권한을 부여하는 규칙에 로컬 터미널의 요구 내용을 명시적으로 언급해야 합니다. 그렇지 않으면 여기에서 CA Access Control 관리 작업을 수행할 수 없습니다.

  기본 액세스 필드인 _default 또는 UACC 클래스를 통해 WRITE 권한을 사용하면 원격 스테이션에서 selang 명령 셸을 입력할 수 있습니다. 그러나 selang 명령을 실행하거나 원격 데이터베이스에 액세스할 수는 *없습니다.* 읽기 권한이 있으면 원격 스테이션에 로그인할 수는 있어도 CA Access Control 관리 작업을 수행할 수는 없습니다.

  다음은 WRITE 권한과 READ 권한의 차이를 설명하는 예입니다.

  1. 새 터미널의 기본 액세스 권한을 READ 로 지정하여 관리자가 터미널에서 로그인할 수는 있지만, 데이터베이스를 조작하지는 못하도록 하려면 다음 명령을 실행합니다.

     ```
     newres TERMINAL tty13 defacc(read)
     ```

  2. 사용자 ADMIN1 에게 새 터미널에서 데이터베이스를 조작할 수 있는 권한을 부여하려면 다음 명령을 실행합니다.

     ```
     authorize TERMINAL tty13 uid(ADMIN1) access(r,w)
     ```

UNIX 환경

UNIX 에서 사용자 및 그룹을 관리하는 경우 전역 또는 그룹 권한 부여 특성을 가진 CA Access Control 사용자는 CA Access Control 에서와 동일한 권한과 제한을 UNIX 에 대해 가집니다.

seosd 데몬이 실행되지 *않을* 경우(예: 설치 시) selang 을 사용하려면 다음 규칙을 따라야 합니다.

- selang 명령에 -I 옵션을 포함시켜야 합니다.
- selang 사용자는 root 여야 합니다. 이 고유 root 권한에는 일반적인 UNIX 제한 사항이 적용됩니다.

Windows 환경

기본 Windows 환경에 해당

CA Access Control 이 실행될 때 selang 을 사용하여 네이티브 Windows 환경에서 리소스를 변경하면 CA Access Control 에이전트는 적절한 Windows 리포지토리에서 이 리소스를 변경합니다. 리소스를 변경하기 위한 추가적인 Windows 권한은 필요 없습니다. 즉, 전역 또는 그룹 권한 부여 특성이 있는 CA Access Control 의 사용자가 네이티브 Windows 환경에서 selang 명령을 수행하면 이 사용자는 Windows 에 대해 CA Access Control 과 동일한 권한 및 제한이 부여됩니다.

CA Access Control 이 실행 중이지 않을 때 네이티브 Windows 환경에서 리소스를 변경하기 위해 selang 을 사용하는 경우에는 다음 규칙을 반드시 따라야 합니다.

- selang 명령에 -l 옵션을 포함시켜야 합니다.
- ADMIN 특성 또는 관리자에 준하는 권한이 있어야 합니다.
- 리소스를 변경하려면 충분한 Windows 권한이 있어야 합니다.

  이 제한 사항은 CA Access Control 에이전트가 아닌 selang 프로세스가 Windows 리포지토리에서 리소스를 변경하기 때문입니다.

예를 들어, 사용자 Emma 는 C:\tmp.txt 파일의 소유자를 변경하기 위해 네이티브 Windows 환경에서 selang 명령을 사용하려고 합니다. CA Access Control 이 실행 중인 경우, Emma 는 파일 소유자를 변경하기 위해 충분한 CA Access Control 권한이 필요하지만 추가 Windows 권한은 필요 없습니다. CA Access Control 이 실행 중이지 않으면 Emma 는 파일 소유자를 변경하기 위해 CA Access Control 과 Windows 의 권한이 모두 필요합니다.

제 15 장: 성능 향상

이 섹션은 다음 항목을 포함하고 있습니다.

Global Access Check 사용

GAC(전역 액세스 검사) 기능은 액세스 규칙이 변경될 가능성이 없는 자주 여는 보호된 파일에 대해 신속한 액세스를 제공합니다.

CA Access Control 관리자는 GAC 를 사용하여 read, write, chown, chmod, rename, unlink, utimes, chattr, link, chdir, create 및 all 에 대한 규칙을 캐시할 수 있으므로 제어를 seosd 에 전달하지 않고도 파일에 대한 적절한 액세스가 허용될 수 있습니다. 기본값은 all 입니다. 그러나 실행 요청은 보안 루프홀을 발생시킬 수 있기 때문에 GAC 에 적합하지 않습니다.

CA Access Control 은 GAC 를 사용하지 않고도 사용자나 프로그램이 보호된 파일에 액세스하려 할 때마다 철저한 보안 검사를 실행합니다. 자주 액세스되는 파일은 액세스 사용 권한을 확인하기 위해 철저한 검사를 반복적으로 수행할 필요가 있습니다.

CA Access Control 관리자는 GAC 를 통해 자주 액세스되는 보호된 파일에 대한 보안 검사를 단축할 수 있습니다. CA Access Control 관리자는 단축된 검사에 적합한 파일을 선택할 수 있습니다. CA Access Control 이 보안 검사 단축을 허용하기 전에 설정된 규칙을 기반으로 파일에 대한 전체 보안 검사를 먼저 수행해야 합니다. 규칙은 일반 파일 이름과 액세스 목록으로 구성됩니다. 규칙은 사용자에 따라 캐시됩니다.

보호된 파일과 관련된 규칙에 실제로 변경 사항이 있으면 적절한 GAC 기능을 통해 간단한 보안 검사 테이블이 플러시되고 초기 전체 보안 검사가 시작되기 때문에 간단한 검사를 위해 특정 파일을 선택하는 것은 바람직합니다.

참고: GAC 제한 사항은 이 기능이 root 를 제외한 모든 사용자에 대해 작동한다는 것을 의미합니다.

GAC 의 작동 방식

CA Access Control 은 지정된 파일에 대한 액세스를 모니터링하고 실행 과정에서 허용된 액세스 테이블을 작성합니다. 이는 GAC 규칙을 설정하기 위해 사전에 지정하는 파일입니다.

CA Access Control 은 사용자에게 특정 파일에 대해 부여할 특정 액세스 수준을 결정할 때마다 다음 두 가지 추가 조건을 만족하는지 확인합니다.

- 허용된 액세스는 무조건적이기 때문에 시간, 일, 실행된 프로그램 또는 기타 조건 등에 대해 독립적입니다.
- 파일이 파일 마스크의 미리 선택된 집합 중 하나와 일치합니다.

참고: 파일 규칙은 파일에 대한 액세스 권한을 정의합니다.

이러한 조건이 충족되면 CA Access Control 은 UID-파일 규칙-액세스의 3 개 항목을 생성하여 이 3 개 항목으로 구성된 테이블에 저장합니다. 이 테이블은 데이터베이스 액세스 규칙을 해석하기 전에 검사됩니다. 사용자가 파일에 액세스를 시도할 때마다 이 테이블을 필터링 메커니즘으로 참조합니다.

테이블을 가장 적절하게 설명하는 용어는 do-not-call-me 테이블이며 그 이유는 이 테이블에는 한 번 인식되면 더 이상 액세스 사용 권한 검사를 수행할 필요가 없는 파일 마스크 목록을 포함하기 때문입니다. 파일 마스크의 목록 내 지정된 파일에 대해 항상 액세스가 허용되므로 이 테이블은 항상 권한 부여 테이블이라고도 부릅니다.

사용자가 파일에 액세스를 시도할 때마다 테이블을 참조합니다. 파일이 테이블에 있는 트리플릿 중 하나와 일치하면 seosd 에 대한 제어 전달 없이 적절한 액세스가 허용됩니다. 이는 액세스 규칙 분석을 바이패스합니다. 결과적으로 액세스 규칙 데이터베이스를 참조하지 않고 테이블에 저장된 트리플릿을 기준으로 이 패턴과 일치하는 파일에 대한 모든 액세스가 허용됩니다.

새 액세스 규칙이 데이터베이스에 추가될 때마다 전체 테이블이 플러시되고 참조 프로세스가 처음부터 시작됩니다.

GAC 구현

GAC 를 설정하려면 자주 액세스되는 파일 집합에 대한 마스크를 선택하고 이러한 파일 마스크를 포함하는 GAC 파일을 설정한 다음 캐싱 프로세스를 시작해야 합니다.

GAC 규칙 설정

참고: 데이터베이스의 파일 규칙은 FILE 클래스 매개 변수와 파일 마스크를 사용하여 생성됩니다. 규칙은 파일 마스크와 일치하는 모든 파일에 적용됩니다. FILE 액세스 유형에는 모두(all), chdir, 제어(control), 작성(create), 삭제(delete), 실행(execute), 없음(none), 읽기(read), 이름 바꾸기(rename), sec, 업데이트가(update), utime 및 쓰기(write) 등이 있습니다.

데이터베이스에 정의된 파일 규칙에서 캐시할 파일 마스크를 선택합니다. 파일 마스크 목록을 데이터베이스에 표시될 때와 똑같은 형식으로 *ACInstallDir*/etc/GAC.init 파일(*ACInstallDir* 은 CA Access Control 의 설치 디렉터리이며 기본적으로 /opt/CA/AccessControl/)에 입력합니다.

각 마스크는 별도의 행에 지정되어야 합니다. 예를 들어 데이터베이스에 /tmp/mydir/*에 대한 파일 마스크가 포함되어 있으며 이를 캐시하려면 다음 줄을 *ACInstallDir*/etc/GAC.init 파일에 추가합니다.

```
/tmp/mydir/*
```

참고: 특정 파일 이름은 GAC.init 파일에 지정될 수 없습니다. 파일 마스크만 사용됩니다.

GAC 시작

CA Access Control 의 현재 버전을 GAC 호환 버전으로 변환하려면 캐싱에 적합한 파일 마스크를 포함하는 *ACInstallDir*/etc/GAC.init 파일을 준비합니다. 파일 마스크만 사용됩니다.

다음 예제는 단 한 줄만 있는 *ACInstallDir*/etc/의 GAC.init 파일입니다.

```
/IBBS/REL63/*
```

GAC 제한 사항

GAC 구현은 특히 파일 액세스가 초당 수백 번 시도되는 경우 매우 효과적이지만 다음과 같은 제한 사항을 가지고 있습니다.

- 기본적으로 GAC 규칙은 루트 사용자(대개 ADMIN)에게 적용할 수 없습니다. 루트에 적용할 수 있는 규칙을 만들려면 seos.ini 파일의 [SEOS_syscall] 섹션에서 다음 토큰을 설정합니다.

  ```
  GAC_root=1
  ```

  토큰의 기본값은 0 입니다. 기본값을 복원하려면 토큰을 0 으로 설정하거나 토큰을 제거합니다.
- 조건적으로 보호된 파일 규칙(예: 일 또는 시간 제한 사항, 프로그램 경로 등)을 테이블에 포함하지 않아야 합니다. 이러한 파일 규칙을 GAC.init 파일에 지정하면 일 또는 시간 제한 사항과 다른 제한 사항이 더 이상 적용되지 않습니다.
- 감사(ALL) 또는 감사(success) 특성이 있는 파일 규칙을 GAC.init 파일에 포함하지 않아야 합니다. 이러한 파일 규칙을 GAC.init 파일에 지정하면 성공적인 액세스의 감사가 기록되지 않습니다.

- 필터링 프로세스는 실제(현재) UID 를 사용하며 이 UID 는 실행 시 프로세스와 연결됩니다. 이 UID 는 현재 UID 가 아니라 사용자가 처음에 로그인할 때 사용한 원래 UID 를 추적하는 CA Access Control 에게 허점을 제공합니다. CA Access Control 은 좀 더 신뢰성있는 보안을 제공하기 위해 UID 사용 추적을 구현합니다.

  사용자가 이러한 루프홀을 활용하는 방식에 대해 예를 들어 살펴 보겠습니다. 사용자 Tony 는 Accounts/tmp 파일에 액세스할 권한이 없습니다. 따라서 Tony 는 Accounts/tmp 에 액세스할 권한이 *있는*사용자 Sandra 를 /bin/su 를 통해 대리합니다. Sandra 가 Accounts/tmp 파일에 이미 액세스한 경우 이 파일은 Sandra UID 를 통해 do-not-call-me 테이블에 나타납니다. Tony 는 Sandra 의 UID 를 사용하여 이 파일에 대한 액세스 권한을 부여받습니다. 이는 커널 코드가 UID 의 기록을 유지 관리하지 않기 때문에 가능합니다.

  그러나 Sandra 가 이전에 이 파일에 액세스하지 않은 경우 seosd 를 사용하여 일반적인 방식으로 액세스 권한을 검사하고 Tony 는 파일에 대한 액세스가 거부됩니다. 이 루프홀을 종료하려면 ADMIN 사용자가 데이터베이스의 SURROGATE 개체를 보호해야 합니다. 예를 들어, ADMIN 사용자는 다음 규칙을 데이터베이스에 추가할 수 있습니다.

  ```
  newres SURROGATE USER.Sandra default(N) owner(nobody)
  ```

  이 명령은 Tony 가 su 명령을 사용하여 Sandra 의 액세스 권한을 가져올 수 없도록 합니다.

- 접근자가 root 이면 캐싱 시스템에 아무런 영향이 없습니다. 이는 데이터베이스를 반드시 참조하는 경우에만 root 에 대한 액세스 권한이 부여되기 때문입니다.

GAC 문제 해결

GAC 가 작동하고 있는지 확인하기 위해 다음과 같이 GAC 를 테스트할 수 있습니다.

1. 추적(Trace)을 사용합니다(secons -t+).
2. GAC.init 에 지정된 파일 마스크 중 하나에 해당되는 파일에 액세스합니다. 첫 번째 파일 액세스는 추적에 기록되어야 합니다.
3. 다시 파일 액세스를 시도합니다. 두 번째 파일 액세스는 추적에 기록되지 않아야 합니다.

   기록된 경우 GAC 가 작동하지 않는 것입니다. GAC.init 에 올바른 형식이 포함되어 있는지 확인합니다.

리소스 캐시 사용

CA Access Control 이 제공하는 다른 성능 개선 도구는 리소스 캐시(파일 캐시)입니다.

캐시는 FILE 클래스의 리소스에 대한 권한 부여 요청(허용 또는 거부)에 대한 이전 응답을 "기억"합니다. 결과는 파일 이름, 사용자 이름 및 권한 부여 응답(액세스 모드, 프로그램 이름 및 결과)과 함께 저장됩니다. 동일한 권한 부여가 요청되면 캐시 메모리 테이블에 저장된 최근 응답을 통해 해당 요청에 대해 응답합니다. 이렇게 하면 CA Access Control 이 요청을 재평가할 필요가 없으므로 시간이 절약되어 CA Access Control 은 응답을 바로 반환할 수 있습니다. 규칙이 변경되면 캐시는 자동으로 즉시 동기화됩니다.

캐시는 런타임 테이블입니다. 관리자는 다음 두 가지 방법으로 캐시를 구성할 수 있습니다.

- seos.ini 파일에 초기화 매개 변수를 설정합니다.
- 캐싱을 ON 또는 OFF 로 전환하고 런타임 시 매개 변수를 변경합니다.

보안 관리자는 테이블 크기, 테이블 제거 간격, seos.ini 파일의 토큰이 있는 다른 내부 테이블 매개 변수를 정의할 수 있습니다.

관리자 권한을 가진 사용자는 캐시 테이블을 ON 또는 OFF 로 전환, 캐시 매개 변수 변경, 캐시 테이블을 표준 출력에 쓰기 등의 작업을 수행할 수 있습니다.

참고: secons 유틸리티나 seos.ini 초기화 파일의 [seosd] 섹션에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

권장 사항 조정

이 권장 사항을 사용하여 성능을 더욱 향상시킬 수 있습니다.

- 세 개의 테이블(풀) 중 하나가 최대 레코드 수를 가지고 있고 다른 테이블은 그렇지 않은 경우 전체 테이블의 크기를 확장합니다.

  참고: 세 개의 테이블은 파일, 사용자 및 권한 부여입니다.

  풀이 낮게 설정되어 있으면 이 설정을 높여 풀을 확장합니다.
- 반드시 필요한 경우가 아니면 토큰을 최대 크기로 설정하지 마십시오. 레코드를 스캔하는 경우 테이블이 클수록 시간이 더 걸립니다.

네트워크 캐시 사용

네트워크 또는 IP 캐싱 기능은 수락된 들어오는 TCP 요청을 저장하기 때문에 TCP 요청은 데이터베이스에 전달되지 않고 syscall 함수를 통해 자동으로 허용됩니다. 이 기능은 여러 개의 들어오는 TCP 연결을 실행하는 호스트 성능을 향상시킵니다.

IP 캐싱 기능을 활성화하려면 seos.ini 파일의 [seosd] 섹션에서 다음 토큰을 변경한 다음 CA Access Control 을 다시 시작합니다.

network_cache_timeout

캐시 테이블 정리 시간 간격을 지정합니다. 이 토큰은 수락 요청에 대한 시간 제한을 설정하는 경우 중요합니다.

UseNetworkCache

IP 캐시를 활성화하려면 이 토큰을 yes 로 설정합니다.

캐싱을 활성화하면 승인된 모든 TCP 연결은 커널 테이블에 저장됩니다. 레코드는 피어 IP 주소, 피어 포트 및 로컬 포트로 이루어집니다. 새 연결은 모두 이 캐시에서 검색됩니다. IP 주소, IP 포트 및 로컬 포트에 대한 일치하는 데이터 세트를 찾으면 연결이 즉시 허용됩니다. 따라서 연결에 걸리는 시간이 단축됩니다.

실제 경로 캐시 사용

CA Access Control 은 파일 시스템의 정보를 사용하므로 파일 이름 확인은 시간이 걸리는 프로세스입니다. CA Access Control 의 커널은 해당 이벤트를 차단할 때 노드 번호를 전체 파일 이름으로 변환합니다. 실제 경로 캐싱은 파일 이름을 내부 테이블에 저장합니다.

이 기능을 사용하려면 seos.ini 파일의 [SEOS_syscall] 섹션에서 cache_enabled 토큰을 1 로 설정합니다. 파일 이름은 inode 번호 및 장치 번호의 데이터 쌍으로 테이블에 캐시됩니다.

참고: seos.ini 초기화 파일에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

Fork 동기화 사용

새 프로세스가 작성되면 seos.ini 파일의 [SEOS_syscall] 섹션에 있는 포크 동기화 토큰(synchronize_fork)이 포크 이벤트 동작을 관리합니다. 이 토큰의 값을 낮게 설정하면 포크 이벤트가 자주 발생하므로 성능을 향상시킬 수 있습니다.

참고: seos.ini 초기화 파일에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

높은 우선 순위 사용

CA Access Control 은 일부 플랫폼에서 seosd 데몬에 대해 실시간 우선 순위를 설정할 수 있는 옵션을 제공합니다. 이 기능을 활성화하려면 seos.ini 파일의 [seosd] 섹션에 있는 rt_priority 토큰을 yes 로 설정합니다. 실시간으로 실행하면 시스템 성능이 향상됩니다.

참고: seos.ini 초기화 파일에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

프로세스 파일 시스템 바이패스

시스템 로드를 줄이기 위해 파일이 프로세스 파일 시스템(/proc)에 속할 때 CA Access Control 이 파일 액세스를 검사하도록 할지 여부를 지정할 수 있습니다.

이 기능을 활성화하려면 seos.ini 파일의 [SEOS_syscall] 섹션에 있는 proc_bypass 토큰을 사용합니다. 이 토큰은 CA Access Control 이 프로세스 파일 시스템에 액세스해야 할 때마다 바이패스되는 액세스 정보를 저장합니다.

참고: seos.ini 파일 토큰에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

실제 경로 바이패스

상대 파일 경로가 아닌 절대 경로로 파일을 검색하면 시스템 로드가 가중되며 이 검색을 바이패스하면 파일 이벤트 속도가 빨라집니다.

이 바이패스를 활성화하려면 seos.ini 파일의 [SEOS_syscall] 섹션에 있는 bypass_realpath 토큰을 1 로 설정합니다. 이 토큰을 활성화하면 CA Access Control 은 심볼 링크와 같은 실제 파일 이름을 포함하지 않습니다.

참고: seos.ini 파일 토큰에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

중요! 이 기능은 상대 경로로 파일에 액세스한 경우 일반 규칙이 작동하지 않는보안에 영향을 줄 수 있기 때문에 신중하게 사용해야 합니다.

트러스트된 프로세스 권한 부여 바이패스

CA Access Control 을 사용하면 프로그램을 트러스트된 것으로 정의할 수 있습니다. CA Access Control 은 트러스트된 프로그램과 해당 프로그램의 자식 프로그램을 테이블에 저장합니다. 트러스트된 프로세스 및 해당 포트와 관련된 모든 이벤트(인바운드 *및* 아웃바운드)는 전체 네트워크 바이패스의 일부로서 권한이 없어도 허용됩니다.

이러한 프로그램을 지정하려면 SPECIALPGM 클래스를 사용합니다.

- 지정된 프로그램에 대한 파일 및 네트워크 이벤트를 바이패스하려면 pbf 및 pbn 값을 가진 PGMTYPE 속성을 사용합니다.
- 지정된 프로그램에 대한 setuid 및 setgid 이벤트를 바이패스하려면 surrogate 값을 가진 PGMTYPE 속성을 사용합니다.
- 지정된 프로그램에 대한 모든 CA Access Control 권한 부여를 바이패스하려면 fullbypass 값을 가진 PGMTYPE 속성을 사용하십시오.

  CA Access Control 은 PGMTYPE(fullbypass) 속성을 가진 프로세스를 무시하며, 어떠한 프로세스 이벤트의 기록도 CA Access Control 감사, 추적, 디버그 로그에 나타나지 않습니다.
- 지정된 프로그램에서 호출되는 모든 프로그램에 바이패스를 전파하려면 propagate 값을 가진 PGMTYPE 속성을 사용합니다.

  참고: 보안 권한은 PBF, PBN, DCM, FULLBYPASS, SURROGATE 권한을 사용할 때만 전파할 수 있습니다.

네트워크 작업 포트 무시

CA Access Control 권한 부여 없이 특정 TCP/IP 포트와 관련된 모든 연결 이벤트(들어가는 연결 *및* 나가는 연결)를 허용하려면 이러한 포트를 무시하도록 정의할 수 있습니다. 이 포트를 무시하면 시스템 로드를 줄이고 이벤트 처리 속도를 높일 수 있습니다. 무시된 연결 이벤트는 감사 및 추적 파일에 로깅되지 않습니다.

참고: CA Access Control 에서는 네트워크 연결 이벤트만 무시할 수 있으며 네트워크 연결을 사용하는 후속 이벤트(예: 파일 열기)는 무시할 수 없습니다.

트러스트된 들어가는 연결은 나가는 연결과 별도로 지정됩니다.

- *들어오는* 연결을 무시하려면 seos.ini 파일의 [seosd] 섹션에 있는 *bypass_TCPIP* 구성 설정을 수정하십시오.
- *나가는* 연결을 무시하려면 seos.ini 파일의 [seosd] 섹션에 있는 *bypass_outgoing_TCPIP* 구성 설정을 수정하십시오.

참고: seos.ini 초기화 파일, 토큰 업데이트 및 영향을 주는 변경 사항에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

예: 들어오는 텔넷 이벤트 무시

bypass_TCPIP 구성 설정을 23(텔넷 포트)으로 설정하면 해당 워크스테이션에 텔넷으로 연결할 때 감사 및 추적 파일에 더 이상 네트워크 이벤트가 로깅되지 않습니다. 하지만 ssh, login, FTP 및 네트워크 연결을 사용하는 후속 이벤트(예: 파일 열기)처럼 다른 서비스와 관련된 이벤트는 계속 로깅됩니다.

예: 나가는 FTP 이벤트 무시

bypass_outgoing_TCPIP 구성 설정을 21(FTP 포트)로 설정하면 해당 워크스테이션에서 FTP 로 연결할 때 감사 및 추적 파일에 더 이상 네트워크 이벤트가 로깅되지 않습니다. 하지만 ssh, login, 텔넷 및 네트워크 연결을 사용하는 후속 이벤트(예: 파일 열기)처럼 다른 서비스와 관련된 이벤트는 계속 로깅됩니다.

감사 및 추적 로드 감소

CA Access Control 은 파일 시스템을 사용하여 감사 데이터와 추적 데이터를 유지합니다. 시스템의 프로세스 대부분은 CA Access Control 이 이 파일 시스템에 기록하는 동안 차단될 수 있습니다. 파일 시스템 액세스 시간을 단축하는 방법은 다음과 같습니다.

- 필요한 리소스 및 액세스에 대한 감사 모드만 설정합니다.
- 필요한 경우에만 추적을 엽니다.
- 감사 파일, 추적 파일 및 CA Access Control 데이터베이스 파일을 가장 빠르게 사용할 수 있는 파일 시스템에 저장합니다.
- lookaside 데이터베이스 디렉터리를 빠른 파일 시스템에 저장합니다.

데이터베이스 로드 감소

데이터베이스에 규칙을 정의하는 방식은 시스템 성능에 영향을 줍니다.

- 일반적으로 사용되는 디렉터리에 대한 일반 규칙은 많은 확인 절차가 필요하기 때문에 시스템 로드를 증가시킬 수 있습니다.

  예를 들어 /usr/lib/*를 보호하면 시스템에서 이루어지는 모든 동작이 CA Access Control 의 검사를 받게 됩니다. 성능을 향상시키려면 자주 사용되는 파일에 대해 일반 규칙을 사용하지 않는 것이 좋습니다.
- 사용자 및 리소스의 계층 구조를 복잡하게 구성하면 모든 종속성을 가져오거나 확인하기 위해 시스템 로드가 필요합니다. 성능을 향상시키려면 데이터베이스에 복잡한 계층 구조를 사용하지 않는 것이 좋습니다.

PMDB 업데이트 개선

정책 모델은 한 루프에서 한 번에 하나씩 순환적으로 명령을 구독자에게 보냅니다. 각 루프 기간 동안 정책 모델이 각 구독자에게 보내는 최대 명령 수를 제어하려면 부록 "pmd.ini 파일"의 "pmd" 절에 설명된 updates_in_chunk 토큰을 사용합니다.

이 토큰 값을 증가시키는 경우 정책 모델은 적은 수의 주기를 사용하여 명령을 보냅니다. 각 루프가 끝날 때마다 정책 모델은 새 요청을 확인합니다. 토큰을 높게 설정한 경우 정책 모델은 새 요청을 자주 확인하지 않습니다.

예를 들어, sepmd -n 옵션을 사용하여 새 구독자를 정책 모델에 추가하면 다른 구독자가 정책 모델이 보내는 명령을 이미 수신했기 때문에 토큰 값은 증가합니다. 정책 모델은 다른 구독자에게 명령을 보내는 시간을 줄이고 새 구독자에게 명령을 보내는 시간을 더 많이 할당하여 구독자 추가 시간을 줄일 수 있습니다.

참고: 이 토큰 값을 100 이상으로 설정하지 마십시오.

Watchdog 성능 개선

시스템 로드를 줄이려면 Watchdog 데몬(seoswd)을 설정하여 보안 파일을 계속 스캔하지 않고 정기적으로 스캔합니다. 시스템 로드가 낮을 때마다 Watchdog 을 스캔하도록 지정할 수 있습니다.

이 기능을 활성화하려면 seos.ini 파일의 [seoswd] 섹션에 있는 IgnoreScanInterval 토큰을 사용하고 시간 간격 및 시작 시간을 위한 추가 토큰을 설정합니다.

참고: 이러한 토큰에 대한 자세한 내용은 *참조 안내서*의 seos.ini 초기화 파일을 참조하십시오.

클래스 매개 변수 개선

CA Access Control 의 클래스 활성화 및 클래스 권한 부여 기능을 사용하여 성능을 향상시킬 수 있습니다.

클래스 활성화

CA Access Control 은 CLASS 가 데이터베이스에서 활성 상태인지 비활성 상태인지에 대한 정보를 저장합니다. CA Access Control 이 시작되면 활성 클래스 목록을 SEOS_syscall 에 전달하므로 CA Access Control 은 계속 이들 클래스를 차단할 필요가 없습니다. CA Access Control 이 클래스를 차단하는 경우는 사용자가 클래스의 활동 상태를 변경할 때뿐입니다. 클래스가 비활성인 경우 리소스에 대한 액세스는 차단되지 않습니다.

FILE, HOST, TCP, CONNECT 및 PROCESS 클래스에 비활성 클래스 바이패스를 사용할 수 있습니다.

클래스 권한 부여

리소스 클래스인 SEOS 는 CA Access Control 권한 부여 시스템의 동작을 제어합니다. SEOS 클래스에는 클래스의 활성 상태 여부를 지정하는 수정 가능한 속성을 가지고 있습니다. 사용하지 않는 클래스를 setoptions 명령을 사용하여 비활성화하여 권한 부여 시간을 줄일 수 있습니다.

이름 확인

seos.ini 파일의 [seosd] 섹션에 포함된 GroupidResolution, HostResolution, ServiceResolution, UseridResolution 등의 토큰은 CA Access Control 이 이름 확인을 수행하는 방식을 제어합니다. 이러한 토큰을 적절하게 설정하여 성능을 향상시킬 수 있습니다.

또한 시스템 이름 확인을 사용하는 대신에 lookaside 데이터베이스를 작성할 수 있습니다. 성능을 향상시키려면 lookaside 데이터베이스 옵션을 선택합니다. 이 기능에 대한 토큰은 lookaside_path 및 use_lookaside 를 포함합니다.

참고: 이러한 토큰에 대한 자세한 내용은 *참조 안내서*의 seos.ini 초기화 파일을 참조하십시오.

CA Access Control 이 UID 를 사용자 이름으로, GID 를 그룹 이름으로, ipaddr 을 호스트 이름으로, 포트를 서비스로 변환해야 하는 경우 CA Access Control 성능에 영향을 줄 수 있습니다. CA Access Control 에서 이러한 변환을 수행하는 방법은 seos.ini 파일의 특정 토큰, 특히 under_NIS_server, use_lookaside, GroupidResolution, HostResolution, ServiceResolution, UseridResolution 및 resolve_timeout 토큰 값에 따라 달라집니다.

Native OS 메커니즘에서 이러한 변환을 수행하면 시스템 성능에 미치는 영향이 비교적 적습니다. ipaddr 을 호스트 이름으로 변환하는 경우 DNS 와 같은 외부 메커니즘이 변환을 수행해야 하므로 시스템 성능이 크게 저하될 수 있습니다. 이러한 성능 저하는 seosd 가 호스트 이름이 수신되기를 기다리는 동안 CA Access Control 에서 차단한 다른 모든 프로세스도 seosd 의 처리가 완료될 때까지 기다려야 하기 때문에 발생합니다.

- under_NIS_server 토큰 값을 no 로 설정할 경우 seosd 를 통해 UNIX 에서는 다음 파일로부터 데이터를 가져와서 UID, GID, IP 주소 및 포트 번호를 변환합니다.

스테이션 유형	소스
독립 실행형	Seosd 는 변환 시 다음 파일을 사용합니다. ■ 사용자 이름에서 UID 에 대해/etc/passwd 파일 사용 ■ 그룹 이름에서 GID 에 대해 /etc/group 파일 사용 ■ 호스트 이름에서 IP 주소에 대해 /etc/hosts 파일 사용 ■ 서비스 이름에서 서비스 포트에 대해 /etc/services 파일 사용
NIS 클라이언트	정보 출처는 운영 체제 및 운영 체제의 버전 번호에 따라 다릅니다. 일반적으로 /etc 파일과 NIS 서버로부터 정보를 가져옵니다. 그러나 일부 시스템에서는 /etc 파일에서 정보를 가져오지 않으며 시스템 구성 중에 변환 순서가 변경됩니다. 예를 들어 Solaris 2.x 시스템에서 /etc/nsswitch.conf 파일이 변환 순서를 결정합니다.
DNS 클라이언트	사용자, 그룹 및 서비스 변환은 /etc 파일을 사용하여 수행됩니다. 호스트 이름은 DNS 서버에 대한 호출에 의해 변환되고, 일부 시스템에서는 /etc/hosts 파일도 읽어들입니다.
NIS 및 DNS 클라이언트	DNS 는 호스트 이름 변환에 대한 ipaddr 을 수행합니다. 사용자, 그룹 및 서비스 변환은 NIS 클라이언트 변환과 동일한 방식으로 수행됩니다.

- under_NIS_server 토큰 값을 yes 로 설정하면 seosd 는 자체 변환을 수행합니다. seosd 가 변환을 위해 데이터를 캐시하는 경우 해당 데이터의 소스는 다음과 같습니다.

스테이션 유형	소스
NIS 서버	일반적으로 서버 컴퓨터는 서버와 클라이언트처럼 작동하고, 모든 유형의 변환에 대해 NIS 서버 데몬을 참조합니다. NIS 확인 맵의 원본을 포함하는 파일은 보통 /var/yp 에 있지만, 사이트 구성과 운영 체제의 유형 및 버전에 따라 파일 위치가 달라질 수 있습니다.
DNS 서버	사이트 구성에 따라 변환을 위해 사용되는 정보 소스가 달라집니다. DNS 에서 확인 데이터베이스를 검색할 수 없으므로 CA Access Control 은 캐싱을 사용할 수 없으며 lookaside 데이터베이스를 사용해야 합니다. sebuildla 유틸리티가 호스트 목록 파일을 사용하도록 lookaside 데이터베이스를 구성해야 합니다. 자세한 내용은 이 장의 sebuildla 를 참조하십시오.
기타 모든 유형	DNS 서버와 동일합니다.

CA Access Control 버전 2 이상에서는 seosd 가 GroupidResolution, HostResolution, ServiceResolution, UseridResolution 및 resolve_timeout 토큰을 사용하여 변환 프로세스를 제어할 수도 있습니다. 이러한 토큰에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

제 16 장: UNIX Exit 사용

이 섹션은 다음 항목을 포함하고 있습니다.

UNIX Exit

UNIX Exit 는 셸 스크립트 또는 실행 파일처럼 사용자가 지정하는 프로그램으로서 정의된 다른 CA Access Control 작업이 수행된 결과로 자동으로 실행됩니다. CA Access Control 에서는 CA Access Control 커널 모듈을 로드하거나 언로드할 때 또는 특정 selang 명령을 실행할 때 UNIX Exit 를 지원합니다. 예를 들어 추가하는 새 사용자 각각에 대해 초기화 프로세스를 실행할 수 있습니다.

UNIX Exit 는 다음 중 한 가지 이상의 경우에 실행됩니다.

- pre-update exit - *사용자* 또는 *그룹* 레코드를 업데이트하는 각 selang 명령 *이전*에 실행
- post-update exit - *사용자* 또는 *그룹* 레코드를 업데이트하는 각 selang 명령 *이후*에 실행
- pre-load exit - *SEOS_load* 가 CA Access Control 커널을 로드하기 *이전*에 실행
- post-load exit - *SEOS_load* 가 CA Access Control 커널을 로드한 *이후*에 실행
- pre-unload exit - *SEOS_load -u* 가 CA Access Control 커널을 언로드하기 *이전*에 실행
- post-unload exit - *SEOS_load -u* 가 CA Access Control 커널을 언로드한 *이후*에 실행

사용자 또는 그룹 레코드 업데이트 Exit

UNIX exit 는 명령줄 인터페이스(selang) 또는 GUI (예: CA Access Control 끝점 관리)인지 여부에 상관없이 사용자 및 그룹 레코드를 업데이트하는 selang 명령이 UNIX 환경에서 실행될 때마다 호출됩니다.

*업데이트*라는 용어는 사용자 또는 그룹 레코드를 작성, 수정 또는 삭제하는 작업입니다. 사용자나 그룹을 쿼리한다고 해서 UNIX Exit 가 실행되지는 않습니다. 다음은 UNIX Exit 가 실행되도록 할 수 있는 명령입니다.

- newusr
- newgrp
- chusr
- chgrp
- editusr
- editgrp
- rmusr
- rmgrp

UNIX 의 관점에서 각 exit 프로세스는 루트 프로세스로 실행되지만 CA Access Control 의 관점에서는 에이전트 ID _seagent 로 실행됩니다.

제공된 selang Exit 스크립트의 작동 방법

CA Access Control 은 현재 selang 명령의 특징과 상태에 따라 다른 프로그램을 호출할 때 마스터 스크립트로 사용할 수 있는 스크립트를 제공합니다. CA Access Control 의 일부로 제공되는 exit 스크립트는 *ACInstallDir*/exits/lang_exit.sh 입니다(여기서 *ACInstallDir* 은 CA Access Control 설치 디렉터리). 이 스크립트의 작동 방식은 다음과 같습니다.

1. CA Access Control 은 스크립트의 세 매개 변수에 자동으로 값을 할당합니다.

매개 변수	가능한 값
CLASS	USER | GROUP
ACTION	CREATE | MODIFY | DELETE

매개 변수	가능한 값
STAGE	PRE | POST

매개 변수는 CA Access Control 이 사용자 또는 그룹을 처리 중인지 여부, 사용자 또는 그룹이 작성, 삭제 또는 수정되는 중인지 여부 그리고 selang 명령이 실행 전(PRE)인지 실행 후(POST)인지 여부를 나타냅니다.

스크립트로 호출하는 프로그램에 매개 변수 값을 전달할 수 있습니다.

매개 변수	가능한 값
EXEC_RV	exit 명령의 성공 여부를 결정하는 데 사용하는 UNIX 명령의 반환 값을 수신합니다. PRE 명령의 경우 반환 값은 항상 0 입니다. POST 명령의 경우 이 값을 사용하여 exit 를 실행할지 건너 뛸지 여부를 결정합니다. 이 매개 변수의 사용 방법에 대한 예는 *ACInstallDir*/samples/exis_src 를 참조하십시오.

2. CA Access Control 은 CLASS 또는 STAGE 매개 변수를 사용하여 해당 디렉터리에서 프로그램을 찾습니다.

   ACInstallDir/exits/USER_PRE/
   ACInstallDir/exits/USER_POST/
   ACInstallDir/exits/GROUP_PRE/
   ACInstallDir/exits/GROUP_POST/

3. 해당 디렉터리에서 CA Access Control 은 파일 이름이 대문자 S 로 시작하고 적절한 작업을 참조하며 다음 형식을 갖는 모든 프로그램을 선택합니다.

   Snnaction_string

   여기서 *nn* 은 실행 시퀀스에서 프로그램 순서를 정의하는 두 자리 십진수이고, *action* 은 CREATE, MODIFY 또는 DELETE 중 하나이고, *string* 은 설명 문자열입니다.

4. CA Access Control 은 프로그램 이름의 두 번째와 세 번째 문자의 숫자 순으로 모든 프로그램을 실행합니다.

예: UNIX Exit 스크립트

사용자를 삭제하고 다음 파일을 포함하는 *ACInstallDir*/exits/USER_PRE/ 디렉터리를 삭제하려고 합니다.

- S10CREATE_precustom.sh
- S10DELETE_precustom.sh
- S99DELETE_prermusrdir.sh

사용자를 삭제하는 명령을 실행하면 사용자를 작성하는 것이 아니라 삭제하는 것이므로 첫 번째 프로그램은 실행되지 않습니다. 첫 글자 S 다음의 두 숫자를 기준으로 한 순서에 따라 두 번째 프로그램이 실행된 다음 세 번째 프로그램이 실행됩니다.

selang Exit 에 전달할 수 있는 인수

Exit 를 작성할 때 앞에서 언급한 매개 변수 세 개(CLASS, ACTION 및 STAGE)를 비롯하여 이름 및 권한과 같은 표준 CA Access Control 데이터를 모두 이용할 수 있습니다. 또한 Exit 스크립트에만 사용할 추가 사용자 또는 그룹 데이터를 지정할 수도 있습니다. 사용자 또는 그룹에 대한 추가 데이터를 저장하려면 newusr, chusr, newgrp 또는 chgrp 명령을 이용하여 작은따옴표 안에 사용자 또는 그룹의 UNIX APPL 속성 값으로 데이터를 정의합니다. 예:

```
chusr JONESY unix APPL('HIRED=MAY93,CLEARANCE=2')
```

exit 프로그램은 작은 따옴표 사이에 있는 모든 내용을 처리할 수 있어야 합니다.

실행할 Exit 프로그램 지정

실행할 Exit 프로그램을 CA Access Control 에 알리려면 seos.ini 파일의 [lang] 섹션을 수정합니다. CA Access Control 은 pre-user, post-user, pre-group, post-group exit 를 위한 lang_exit.sh 스크립트를 제공합니다. 종료를 지정하지 않거나 사용자 지정 종료를 생성할 수도 있습니다.

자체 selang exit 를 정의하려면 seos.ini 의 [lang] 섹션에서 원하는 대로 모든 설정을 구성합니다.

참고: exit 는 전체 경로 이름을 exit 토큰 값으로 표시하는 경우에만 호출됩니다.

예: selang Exit 지정

다음 예제에서는 groupcheck 프로그램이 그룹 작업 이전에 실행되고, flag_exceptions 프로그램이 그룹 작업 이후에 실행되며, lang_exit.sh 프로그램이 사용자 작업 이후에 실행되고, 사용자 작업 이전에는 어떤 exit 프로그램도 실행되지 않도록 seos.ini 토큰을 설정합니다. seos.ini 파일 토큰을 다음과 같이 설정합니다.

```
[lang]
pre_group_exit = /opt/CA/AccessControl//exits/groupcheck
post_group_exit = /opt/CA/AccessControl//exits/flag_exceptions
post_user_exit = /opt/CA/AccessControl//exits/lang_exit.sh
```

시간 초과 및 기타 실패

pre-update exit 가 시간을 초과하거나 16 이상의 반환 코드를 반환하는 경우 eTrust AC 는 exit 프로세스를 종료하고 오류 메시지를 표시한 다음 eTrust AC 업데이트 명령 실행을 중단합니다. 0 이 아닌 반환 값은 실패를 나타냅니다.

- *pre*-update exit 가 시간 초과되거나 반환 코드가 16 이상이면 CA Access Control 은 exit 프로세스를 종료하고 오류 메시지를 표시한 다음 CA Access Control 업데이트 명령 실행을 중단합니다. 다른 양의 반환 코드는 명령 실행의 중단하지 않습니다.
- *post*-update exit 가 시간 초과되거나 0 이 아닌 값을 반환하면 CA Access Control 은 exit 프로세스를 종료하고 오류 메시지를 표시합니다. 이미 실행된 CA Access Control 업데이트 명령은 그대로 유효합니다.

selang Exit 샘플

다음 디렉터리에 있는 스크립트를 검토하여 권장되는 스크립트 작성 방법을 익힐 수 있습니다.

ACInstallDir/samples/exits-src
ACInstallDir/samples/sample_exits

CA Access Control 커널 로더 Exit

CA Access Control 커널이 로드되거나 언로드될 때마다(SEOS_load) UNIX Exit 가 호출됩니다. 이 Exit 를 사용하면 CA Access Control 커널을 로드하거나 언로드할 때 운영 체제 및 타사 프로그램을 처리할 방법을 정의할 수 있습니다. 예를 들어 커널 언로드 UNIX exit 를 사용하면 *SEOS_load -u* 를 실행할 때 CA Access Control 이 언로드되지 않도록 하는 프로세스를 자동으로 중지했다가 나중에 다시 시작할 수 있습니다.

운영 체제에 따라서는 CA Access Control 에 몇 가지 커널 로드 Exit 나 커널 언로드 Exit, 또는 두 가지 모두가 함께 제공됩니다.

참고: CA Access Control 커널이 언로드되지 않도록 하는 프로세스 식별에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

커널 로드 Exit 의 작동 방법

CA Access Control 에서는 운영 체제와 타사 프로세스를 제어할 수 있도록 CA Access Control 커널 확장을 로드할 때 UNIX exit 를 자동으로 호출할 수 있습니다.

SEOS_load 를 실행하면 CA Access Control 이 다음 작업을 수행합니다.

1. 다음 디렉터리에서 프로그램을 찾습니다.

   ACInstallDir/exits/LOAD

2. 파일 이름이 다음과 같은 형식인 프로그램을 모두 선택합니다.

   SEOS_load_*string*.always

   여기서 *string* 은 설명 문자열입니다.

3. *ACInstallDir*/exits/LOAD 디렉터리에서 찾은 각 파일을 사전순으로 실행합니다.

   SEOS_load_*string*.always -pre

   각 파일을 *-pre* 매개 변수를 사용하여 실행합니다. 이렇게 하면 커널이 로드되기 전에 필요한 작업을 수행하고 매개 변수를 찾는 exit 를 작성할 수 있습니다.

   참고: Exit 에서 0 이 아닌 값을 반환하면 CA Access Control 은 exit 프로세스를 종료하고 오류 메시지를 표시한 다음 커널 로드를 중단합니다.

4. 커널(SEOS_syscall)을 로드합니다.
5. *ACInstallDir*/exits/LOAD 디렉터리에서 찾은 각 파일을 사전순으로 실행합니다.

   SEOS_load_*string*.always -post

   각 파일을 *-post* 매개 변수를 사용하여 실행합니다. 이렇게 하면 커널이 로드된 후에 필요한 작업을 수행하고 매개 변수를 찾는 exit 를 작성할 수 있습니다.

   참고: exit 에서 0 이 아닌 값을 반환하면 CA Access Control 은 exit 프로세스를 종료하고 오류 메시지를 표시합니다. 이미 로드된 CA Access Control 커널은 로드된 상태로 유지됩니다.

커널 언로드 Exit 의 작동 방법

CA Access Control 에서는 운영 체제와 타사 프로세스를 제어할 수 있도록 CA Access Control 커널 확장을 언로드할 때 UNIX exit 를 자동으로 호출할 수 있습니다.

SEOS_load -u 를 실행하면 CA Access Control 이 다음 작업을 수행합니다.

1. 다음 디렉터리에서 프로그램을 찾습니다.

   ACInstallDir/exits/LOAD

2. 파일 이름이 다음과 같은 형식인 프로그램을 모두 선택합니다.

   SEOS_unload_*string*.always

   여기서 *string* 은 설명 문자열입니다.

3. *ACInstallDir*/exits/LOAD 디렉터리에서 찾은 각 파일을 사전순으로 실행합니다.

   SEOS_load_*string*.always -pre

   각 파일을 *-pre* 매개 변수를 사용하여 실행합니다. 이렇게 하면 커널이 언로드되기 전에 필요한 작업을 수행하고 매개 변수를 찾는 exit 를 작성할 수 있습니다.

   참고: Exit 에서 0 이 아닌 값을 반환하면 CA Access Control 은 exit 프로세스를 종료하고 오류 메시지를 표시한 다음 커널 언로드를 중단합니다.

4. 커널을 언로드해 봅니다.

   커널이 언로드되지 *않는* 경우:

   a. 파일 이름이 다음과 같은 형식인 프로그램을 모두 선택합니다.

      SEOS_unload_*string*.opt

   b. *ACInstallDir*/exits/LOAD 디렉터리에서 찾은 각 파일을 사전순으로 실행합니다.

      SEOS_unload_*string*.opt -pre

      각 파일을 *-pre* 매개 변수를 사용하여 실행합니다. 이렇게 하면 커널이 언로드되기 전에 필요한 선택적 작업을 추가로 수행하고 매개 변수를 찾는 exit 를 작성할 수 있습니다.

      참고: Exit 에서 0 이 아닌 값을 반환하면 CA Access Control 은 exit 프로세스를 종료하고 오류 메시지를 표시한 다음 커널 언로드를 중단합니다.

   c. 커널을 언로드합니다.

   d. *ACInstallDir*/exits/LOAD 디렉터리에서 찾은 각 파일을 사전순으로 실행합니다.

      SEOS_unload_*string*.opt -post

      각 파일을 *-post* 매개 변수를 사용하여 실행합니다. 이렇게 하면 커널이 언로드되기 전에 필요한 선택적 작업을 추가로 수행하고 매개 변수를 찾는 exit 를 작성할 수 있습니다.

      참고: exit 에서 0 이 아닌 값을 반환하면 CA Access Control 은 exit 프로세스를 종료하고 오류 메시지를 표시합니다. 이미 언로드된 CA Access Control 커널은 언로드된 상태로 유지됩니다.

5. *ACInstallDir*/exits/LOAD 디렉터리에서 찾은 각 파일을 사전순으로 실행합니다.

   SEOS_unload_*string*.always -post

   각 파일을 *-post* 매개 변수를 사용하여 실행합니다. 이렇게 하면 커널이 로드된 후에 필요한 작업을 수행하고 매개 변수를 찾는 exit 를 작성할 수 있습니다.

   참고: exit 에서 0 이 아닌 값을 반환하면 CA Access Control 은 exit 프로세스를 종료하고 오류 메시지를 표시합니다. 이미 언로드된 CA Access Control 커널은 로드되지 않은 상태로 유지됩니다.

제 17 장: LDAP 와의 상호 작용

이 섹션은 다음 항목을 포함하고 있습니다.

사용자 이름 전송

CA Access Control 과 LDAP 를 둘 다 사용하는 경우 직접 작성한 스크립트를 이용하여 사용자 이름을 두 프로그램 간에 전송할 수 있습니다. 이를 위해 세 가지 샘플 스크립트가 제공됩니다.

중요! sebuildla 및 필수 LDAP 구성 설정을 설정하려면 LDAP 에 대해 잘 알고 있고 ldapsearch 명령을 실행할 수 있어야 합니다. ldap(1), ldapsearch(1)에 대한 man 페이지와 LDAP 클라이언트 설명서에서 설정 정보를 검토하는 것이 좋습니다.

제공되는 스크립트 중 ldap2seos 와 seos2ldap 는 전체 사용자 집합을 CA Access Control 에서 LDAP 서버로 내보내고 LDAP 서버에서 CA Access Control 로 가져옵니다.

세 번째 샘플 스크립트인 S50CREATE_Ldap_u.sh 는 새로 생성된 UNIX 사용자 이름을 CA Access Control 에서 LDAP 로 자동 전송합니다.

샘플 스크립트를 사용하려면 TCL 셸 환경에 액세스할 수 있어야 합니다. 이 스크립트에서는 언어 클라이언트 API(LCA) 라이브러리 확장인 tcllca.so 를 사용합니다.

참고: LCA 및 TCL 확장에 대한 자세한 내용은 각각 *SDK 안내서*에서 언어 클라이언트 API 및 부록 LCA 확장을 참조하십시오.

TCL 이 없는 경우 Larry Virden 이 comp.lang.t_c_l 에 매월 게시하는 FAQ 를 참조하십시오. 이 FAQ 는 MIT 웹 사이트 및 Terafirm 웹 사이트에서도 확인할 수 있습니다.

또한 TCL 소식, 문서 및 리소스에 대해서는 Sun 웹 사이트도 참조할 수 있습니다.

S50CREATE_Ldap_u

S50CREATE_Ldap_u.sh 는 생성될 때 새 UNIX 사용자를 LDAP 로 업로드합니다.

CA Access Control 은 새 UNIX 사용자를 자동으로 LDAP 서버로 가져오기 위한 샘플 셸 스크립트를 제공합니다. 필요한 스크립트는 예제에 따라 다를 수 있습니다.

제공된 exit 스크립트를 이미 사용하고 있는 경우 샘플 셸 스크립트를 사용하려면 다음을 수행합니다.

1. S50CREATE_Ldap_u.sh 파일을 *ACInstallDir*/exits/USER_POST 디렉터리로 복사합니다. 이 디렉터리에서 스크립트는 post-user exit 가 됩니다.
2. seos.ini 파일의 [ldap]에서 base_entry 토큰을 LDAP 기본 항목으로 설정합니다.

   예를 들어 캐나다에 있는 ServerWorld 라는 조직의 경우 기본 항목은 o=ServerWorld, c=CA 입니다.
3. 동일한 섹션에서 호스트 이름을 LDAP 서버의 호스트 이름으로 설정합니다. 경로를 LDAP 기본 디렉터리로 설정합니다. 예제 스크립트는 해당 디렉터리 아래의 bin 디렉터리에서 명령줄 유틸리티를 찾습니다.

일반 이름(cn)은 사용자의 전체 이름에서 파생됩니다. 예를 들어 CA Access Control 데이터베이스에 사용자 이름과 성만 포함된 경우 이 이름과 성이 일반 이름이 됩니다. You 는 반드시 일반 이름이 되므로 사용자 이름에 기본으로 사용하지 않는 것이 좋습니다.

selang 을 사용하여 UNIX 에 연속적으로 추가된 각 사용자는 LDAP 서버에 자동으로 업로드됩니다. 사용자가 이미 LDAP 에 있으면 오류 메시지가 나타납니다.

이 스크립트로 사용자를 추가할 때 관련된 LDAP 응답과 경고(있는 경우)가 /tmp/add_User2Ldap.tcl.log 파일에 수집됩니다. 오류를 검사하기 위해 vi 또는 다른 표준 UNIX 편집기를 사용하여 이 파일을 검사할 수도 있습니다. 새 사용자를 추가할 때마다 새로운 응답과 경고를 이 파일에 덮어씁니다.

제 18 장: 설정 구성

CA Access Control 에서는 CA Access Control 끝점 구성 설정을 원격으로 관리할 수 있습니다. 이렇게 하려면 CA Access Control 끝점 관리 또는 selang 구성 환경을 사용합니다.

이 섹션은 다음 항목을 포함하고 있습니다.

구성 설정

CA Access Control 에서는 사용되는 끝점 및 정책 모델 구성 설정을 다음 위치에 저장합니다.

- Windows 컴퓨터의 경우 Windows 레지스트리
- UNIX 컴퓨터의 경우 초기화 파일(.ini)

참고: 작성할 수 있는 구성 설정과 그 의미에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

구성 설정 변경

CA Access Control 및 정책 모델이 작동하는 방식을 변경하려면 구성 설정을 변경해야 합니다.

구성 설정을 변경하려면

1. CA Access Control 끝점 관리에서 다음을 수행하십시오.
   a. "구성"을 클릭합니다.
   b. "원격 구성"을 클릭합니다.

   "원격 구성" 페이지가 나타납니다.
2. "원격 구성" 섹션 페이지의 왼쪽에서 필요한 경우 구성 트리를 확장하여 수정할 구성 설정이 있는 섹션을 표시한 다음 해당 섹션을 선택합니다.

   "섹션: *sectionName* 시스템 토큰" 페이지가 나타나고 모든 구성 설정이 표시됩니다.
3. 필요한 구성 설정을 찾아서 편집한 다음 "토큰 저장"을 클릭합니다.

   변경된 구성 설정이 저장됩니다.

감사 구성 설정 변경

CA Access Control 이 감사 레코드를 생성하고 저장하는 방법에 수정하려면 감사 구성 파일의 설정을 변경해야 합니다. selang 명령을 사용하여 감사 구성 파일의 설정을 변경할 수 있습니다.

감사 구성 설정을 변경하려면

1. (선택 사항) selang 을 사용하여 원격 호스트에 연결하려는 경우 다음 명령으로 호스트에 연결합니다.

   host *host_name*
2. 다음 명령을 사용하여 구성 환경으로 이동합니다.

   env config
3. editres 구성 명령을 사용하여 구성 설정을 필요에 따라 수정합니다.

   감사 구성 설정이 변경됩니다.

예: 감사 구성 파일 수정

다음 예제는 감사 구성 파일에 한 줄을 추가합니다.

```
er CONFIG audit.cfg line+("FILE;*;Administrator;*;R;P")
```

부록 A: NIS 구성

이 섹션은 다음 항목을 포함하고 있습니다.

설치 정보

참고: 이 단원에서는 설치 스크립트에 관한 내용을 보충합니다. 이 부록에서는 사용자가 NIS(Network Information Systems) 또는 DNS(Domain Name Services) 및 UNIX 이름 확인 개념에 대해 잘 알고 있다고 가정합니다.

CA Access Control 을 설치하는 동안 두 가지 옵션 중 하나를 사용하여 사용자 ID 를 사용자 이름에 대해, 그룹 ID 를 그룹 이름에 대해, 호스트 IP 주소를 호스트 이름에 대해, 그리고 서비스 포트를 서비스 이름에 대해 확인할 수 있습니다.

- 시스템에서 net cashing 데몬에 대한 바이패스를 정의하는 시스템 기능을 사용합니다.
  - Digital DEC UNIX 를 사용하고 이러한 서버가 NIS 서버가 *아닌 경우* 기본적으로 이름 확인을 위해 시스템 기능을 사용합니다.
  - Digital DEC UNIX 를 사용하고 있고 이 서버가 NIS 서버 *인 경우* 설치 과정 중에 lookaside 데이터베이스를 사용하거나 net caching 데몬의 바이패스를 정의하는 시스템 기능을 사용하는 두 가지 옵션 중 하나를 선택하라는 메시지가 표시됩니다.
- sebuildla 유틸리티에 의해 생성되는 lookaside 데이터베이스를 사용합니다.
  - *NIS 서버*에서 실행되도록 구성된 CA Access Control 을 사용하는 경우에는 lookaside 데이터베이스를 사용합니다.
  - HP-UX 11.0 이상, Sun Solaris 2.6 이상, IBM AIX 5.1L 이상 및 지원되는 모든 Linux 플랫폼의 경우 설치 기본값은 lookaside 데이터베이스 사용입니다.

참고: IBM AIX 플랫폼의 경우 시스템 기능 사용 옵션이 없기 때문에 lookaside 데이터베이스를 사용해야 합니다.

이름 확인

CA Access Control 은 액세스 시스템 리소스에 대한 요청을 차단하여 이 요청을 허용할지 거부할지 결정합니다. 데이터베이스에 정의된 액세스 규칙 및 정책을 기준으로 결정이 이루어집니다. 액세스 시스템 리소스에 대한 요청 차단은 커널 수준에서 실행됩니다.

호스트, 그룹, 사용자 및 서비스를 제어하기 위해 커널 및 관련 시스템 호출에서는 이름 대신 코드 또는 숫자(즉 IP 주소, 그룹 ID, 사용자 ID 및 서비스 번호)를 사용합니다. CA Access Control 은 이름을 기준으로 액세스 규칙을 정의합니다. CA Access Control 은 이름을 커널에 의해 인식되는 코드로 변환합니다. 이 프로세스를 이름 확인이라고 합니다.

Sun Solaris 2.5 이상을 실행하는 스테이션을 제외한 독립형 스테이션에서는 로컬 사용자 그룹 및 호스트 파일(/etc/passwd, /etc/group, and /etc/hosts)을 통해 직접 이름 확인이 이루어집니다. CA Access Control 에서 이름을 확인해야 할 경우에는 시스템 함수를 호출하여 관련 파일을 읽습니다.

하지만 대규모 네트워크에서 이러한 정보는 로컬로 저장되는 경우가 거의 없습니다. NIS, DNS 또는 둘 모두 사용하는 경우 이름 확인을 수행하는 동안 참조할 수 있는 로컬 파일이 없습니다. 정보는 네트워크를 통해 서버에서 요청되고 수신됩니다.

NIS/DNS 클라이언트에서 이름 확인

CA Access Control 은 클라이언트 전용 NIS 또는 DNS 스테이션(자체 서버가 아님)에서 다음과 같이 이름 확인을 수행합니다.

1. CA Access Control 에서 관련 서버에 연결하기 위한 네트워크 요청을 생성합니다.
2. CA Access Control 커널 확장이 요청을 차단합니다.
3. CA Access Control 커널 확장은 해당 요청이 CA Access Control 프로세스에 의해 내부적으로 작성된 것을 알고 있기 때문에 이 요청을 허용합니다.
4. NIS 또는 DNS 서버와 연결되어 이름 확인에 필요한 정보를 검색합니다.
5. 이름이 확인되면 CA Access Control 은 원래 액세스 요청을 허용할지, 거부할지 여부를 결정하는 프로세스를 계속 수행합니다.

표준 CA Access Control 구성인 경우 CA Access Control 이 클라이언트 서버에서 이름 확인을 간단히 처리할 수 있습니다.

서버에서 이름 확인: 교착 상태

CA Access Control 은 클라이언트로서 속해 있는 서버에서 다음과 같이 이름 확인을 수행합니다.

1. CA Access Control 에서 관련 서버에 연결하기 위한 네트워크 요청을 생성합니다.
2. 커널 익스텐션은 이 요청을 차단합니다.
3. 커널 확장은 해당 요청이 CA Access Control 프로세스에 의해 내부적으로 작성된 것을 알고 있기 때문에 이 요청을 허용합니다.
4. 동일한 스테이션에 있는 NIS 또는 DNS 서버는 네트워크 연결을 허용하는 요청을 생성합니다.
5. 커널 익스텐션은 이 요청을 차단합니다.
6. 커널 확장은 CA Access Control 프로세스가 이 요청을 하지 않았음을 알고 있습니다. seosd 결정을 기다리는 요청 큐에 해당 요청을 둡니다.
7. seosd 데몬은 현재 교착 상태에 있습니다. 이름 확인을 완료하기 위해 필요한 응답을 기다리는 중이지만 이러한 응답을 제공해야 하는 프로세스는 seosd 가 네트워크 연결을 수락하는 권한을 이 프로세스에게 부여할 때까지 작업을 진행할 수 없습니다. 첫 번째 요청은 두 번째 요청을 생성하고 교착 상태를 일으킵니다.

Sun Solaris 에서 이름 확인: 교착 상태

Sun Solaris 에서 이름을 확인하려면 *nscd* 캐시에 액세스해야 합니다. nscd 는 가장 일반적으로 이름 서비스 요청에 대한 캐시를 제공하는 프로세스입니다. nscd 는 암호, 그룹 및 호스트 데이터베이스에 대한 캐싱을 제공합니다.

캐시는 영구적이지 않습니다. 따라서 암호, 그룹 및 호스트 데이터베이스가 변경되거나 TTL(time-to-live) 스탬프가 만료되면 무효가 됩니다.

Sun Solaris 를 설치하면 이전 단원에서 설명했던 것처럼 교착 상태를 일으킬 수 있습니다. 여기서 CA Access Control 과 nscd 프로세스 간의 상호 작용에 의해 교착 상태가 발생합니다.

1. 이름을 확인할 때 CA Access Control 은 nscd 캐시에 액세스합니다.
2. nscd 프로세스에서 캐시가 너무 오래 되었다고 판단되면 이 단계에서 nscd 프로세스는 암호, 그룹 및 호스트 데이터베이스(로컬 또는 서버에서)에 액세스하여 정보를 새로 고치려고 시도합니다.
3. 이 데이터베이스에 대한 액세스 요청은 커널 익스텐션에 의해 차단됩니다. CA Access Control 프로세스가 요청을 하는 것이 아니므로 해당 요청은 seosd 결정을 기다리는 큐에 추가됩니다. 하지만 seosd 가 이전 요청을 아직 처리하는 중이므로 이 요청을 처리할 수 없습니다. 첫 번째 요청은 두 번째 요청을 생성하고 교착 상태를 일으킵니다.

교착 상태 방지: Lookaside 데이터베이스

교착 상태를 방지하기 위해 seos.ini 구성 파일의 under_NIS_server 토큰은 기본적으로 yes 로 설정됩니다. 토큰은 NIS, DNS 또는 nscd 캐시 대신 자체의 내부 이름 확인 테이블을 사용하도록 CA Access Control 에 알립니다. 달리 지정하지 않는 한 이 테이블은 메모리에 위치합니다.

CA Access Control 의 내부 이름 확인은 NIS 이름 확인보다 훨씬 빠르며, 심지어 파일을 사용하는 것보다도 빠르기 때문에 CA Access Control 의 내부 이름 확인을 사용하면 교착 상태의 위험이 없는 환경에서도 성능을 향상시킬 수 있습니다.

참고: 교착 상태 데이터베이스의 내부 이름 확인 테이블에 대한 캐시는 없습니다. CA Access Control 은 열린 파일 핸들을 사용하여 테이블의 데이터를 읽습니다.

디스크에 이름 확인 테이블 저장

CA Access Control 이름 확인 테이블은 CA Access Control 이 시작되는 동안 생성됩니다. 메모리 저장소에 메모리 부하가 발생할 수 있기 때문에 이 테이블은 메모리가 아닌 디스크에서 관리되어야 합니다. 또한 정보를 메모리에서 읽을 때 정적입니다. 따라서 는 CA Access Control 은 사용자, 그룹 또는 호스트 정보의 변경 사항을 알지 못합니다. 메모리의 테이블을 업데이트하는 유일한 방법은 CA Access Control 을 다시 시작하는 것입니다.

데이터를 최신 상태로 유지할 수 있도록 CA Access Control 은 내부 이름 확인 테이블을 디스크에 저장하는 lookaside 데이터베이스를 제공합니다.

참고: lookaside 데이터베이스를 구현하려면 seos.ini 구성 설정을 사용해야 합니다. seos.ini 구성 설정에 대한 자세한 내용은 *참조 안내서*를 참조하십시오.

lookaside 데이터베이스 설정

참조(lookaside) 데이터베이스에 있는 네 개의 테이블은 userdb.la, groupdb.la, hostdb.la 및 servdb.la 입니다. 이 네 개의 테이블에서는 사용자, 그룹, 호스트 및 서비스 이름 확인 요청을 처리합니다. 이 테이블은 seos.ini 파일의 lookaside_path 토큰에서 지정한 디렉터리에 있으며 이 디렉터리는 기본적으로 /opt/CA/AccessControl//ladb 입니다.

4 개 테이블이 있는 lookaside 데이터베이스

4 개의 테이블이 있는 lookaside 데이터베이스를 설치하려면 다음 중 하나를 수행합니다.

- CA Access Control 을 설치하는 중이라면 lookaside 데이베이스를 작성할지 묻는 메시지가 표시될 때 yes 라고 답합니다.
- CA Access Control 을 이미 설치했다면
  a. seos.ini 의 [seosd] 섹션에서 다음 토큰을 **yes** 로 변경합니다.
     - under_NIS_server
     - use_lookaside
  b. sebuildla -a 를 실행하여 네 개의 테이블을 모두 생성합니다.

4 개 이하의 테이블이 있는 lookaside 데이터베이스

또한 하나, 두 개 또는 세 개의 테이블을 작성할 수도 있습니다. 예를 들어 lookaside 데이터베이스를 사용하여 호스트만 확인하려는 경우 다음 단계를 수행합니다.

1. CA Access Control 을 설치한 후 seos.ini 파일의 [seosd] 섹션에서 다음 토큰을 변경합니다.
   - under_NIS_server 를 비워둡니다.
   - HostResolution 를 ladb 로 설정합니다.
2. sebuildla -h 를 실행하여 로컬 호스트 및 DNS 호스트를 포함한 모든 호스트의 테이블을 생성합니다.

   또는

   sebuildla -e 를 실행하여 로컬 호스트의 테이블만 작성합니다(etc/hosts 에 정의).

다른 테이블과 함께 lookaside 데이터베이스를 생성하려면 seos.ini 파일에서 적절한 토큰을 사용하고 sebuildla 를 사용하여 적절한 옵션을 실행합니다.

참고: 이러한 토큰에 대한 자세한 내용은 *참조 안내서*의 seos.ini 초기화 파일을 참조하십시오. sebuildla 에 대한 자세한 내용은 *Utilities Guide* 를 참조하십시오.

중요! 호스트를추가할 때마다 sebuildla 를실행합니다.

lookaside 데이터베이스의 작동 방식

lookaside 데이터베이스에 포함된 네 개의 테이블(groupdb.la, hostdb.la, servdb.la, and userdb.la)에는 그룹, 호스트, 서비스 및 호스트 이름에 대한 확인(resolution) 정보가 들어 있습니다. 이 테이블은 seos.ini 파일의 lookaside_path 토큰에서 지정한 디렉터리에 있으며 이 디렉터리는 기본적으로 /opt/CA/AccessControl//ladb 입니다.

CA Access Control 의 내부 이름 확인은 NIS 이름 확인보다 빠르며 th 파일을 찾는 것보다 훨씬 빠릅니다.

lookaside 데이터베이스 구현

참고: 여기에서 간략하게 설명하는 문제와 해결 방법은 정보 제공을 위해서만 제공됩니다. 실제 설정은 설치에 따라 다를 수 있으며 대부분의 사용자는 어떤 조치도 취할 필요가 없습니다.

다음은 CA Access Control 이 lookaside 데이터베이스를 구현하는 방식을 개략적으로 설명한 것입니다.

- seos.ini 파일에 있는 관련 토큰이 설정됩니다.
- /opt/CA/AccessControl//exits 디렉터리에 있는 관련 심볼 링크가 정의됩니다.
- 명령 /opt/CA/AccessControl//bin/sebuildla -a 가 참조(lookaside) 데이터베이스를 만들기 위해 호출되었습니다.

sebuildla 유틸리티는 th 파일, NIS 등과 같은 기본 확인 메커니즘을 사용하여 참조(lookaside) 데이터베이스를 작성합니다.

암호, 홈 디렉터리 위치 또는 gecos 등과 같이 보호해야 할 중요한 정보는 lookaside 테이블에 없습니다. lookaside 데이터베이스 테이블에는 숫자로 구성된 ID 와 이름만 들어 있습니다.

lookaside 데이터베이스가 작성되면 sebuildla 유틸리티를 사용하여 업데이트합니다. CA Access Control 을 다시 시작할 필요는 *없습니다*.

호스트 lookaside 테이블 업데이트

호스트 lookaside 테이블을 업데이트해야 합니다. 이를 위해서는 sebuildla -h 를 정기적으로 실행해야 합니다(사이트마다 다름). 크론(cron) 작업을 사용하여 이를 수행합니다.

selang 을 사용하는 UNIX 사용자나 그룹 데이터베이스를 변경할 때마다 sebuildla 유틸리티를 실행해야 합니다. CA Access Control 에서는 이를 위해 적절한 매개 변수를 사용하여 sebuildla 를 실행하는 exit 스크립트를 제공합니다.