Sun Desktop Manager 1.0 설치 설명서

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

부품 번호: 819–6094–10

060406@14774

목차

머리말

이 문서에서는 Sun™ Desktop Manager 1.0을 배포하는 데 필요한 설치 및 구성 단계에 대해 설명합니다.

개요

Sun Desktop Manager의 목표는 데스크탑 호스트에 중앙 집중식 구성을 제공하는 것입니다. 이 제품을 사용하면 조직 또는 도메인 구조의 다양한 요소에 설정을 지정할 수 있으므로 관리자가 사용자 또는 호스트 그룹을 효과적으로 관리할 수 있습니다.

이 책의 구성

1장에서는 Sun Desktop Manager에 대한 간략한 개요를 제공합니다.

2장에서는 Sun Desktop Manager 서버측 설치에 대해 설명합니다.

3장에서는 Java Desktop System Configuration Agent 설치 관련 정보를 제공합니다.

4장에서는 Java Web Console에 대한 설치 정보를 제공합니다.

부록 A에는 구성 매개 변수 정보가 포함되어 있습니다.

부록 B에서는 Desktop Manager와 함께 OpenLDAP 및 Active Directory를 사용하는 방법에 대해 설명합니다.

부록 C에서는 조직 매핑에 대한 정보를 제공합니다.

관련 문서

- **Sun Desktop Manager 1.0** 관리 설명서
- **Sun Desktop Manager 1.0 Developer Guide**

설명서, 지원 및 교육

Sun Function	URL	설명
설명서	http://www.sun.com/documentation/	PDF 및 설명서를 다운로드하고 인쇄된 설명서를 주문합니다.
지원 및 교육	http://sunsolve.sun.com	기술 지원 및 패치 다운로드가 가능하며, 교육 과정에 대한 정보를 얻을 수 있습니다.

◆ ◆ ◆ 1 장

1 개요 및 개념

이 문서에서는 Sun™ Desktop Manager 1.0을 배포하는 데 필요한 설치 및 구성 단계에 대해 설명합니다. Sun Desktop Manager에 대한 자세한 개요는 **Sun Desktop Manager 1.0** 관리 설명서를 참조하십시오.

Sun Desktop Manager 개요

Sun Desktop Manager는 데스크탑 호스트를 위한 중앙 집중식 구성을 제공합니다. 이 제품을 사용하면 조직 또는 도메인 구조의 다양한 요소에 설정을 지정할 수 있으므로 관리자가 사용자 또는 호스트 그룹을 효과적으로 관리할 수 있습니다.

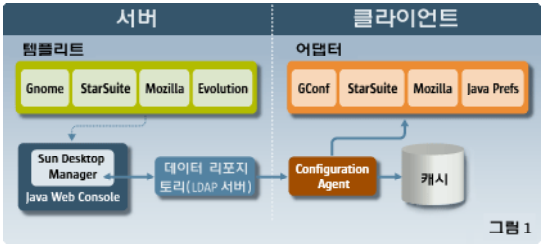

그림 1–1 Desktop Manager 아키텍처

Desktop Manager의 주요 구성 요소는 다음과 같습니다.

- 구성 리포지토리
- 관리 도구
- Desktop Manager 템플리트
- Configuration Agent
- 구성 어댑터

구성 데이터는 구성 리포지토리의 중앙에 저장됩니다. 구성 데이터는 웹 기반 Desktop Manager 그래픽 사용자 인터페이스와 명령줄 인터페이스로 구성되는 관리 도구를 사용하여 관리(작성/삭제/수정/지정/지정 취소)됩니다. 템플리트는 웹 기반 관리 도구가 구성 데이터를 웹 브라우저에 렌더링하는 데 사용됩니다.

Configuration Agent는 사용자 응용 프로그램 대신에 구성 리포지토리에서 구성 데이터를 검색합니다. 에이전트는 중앙 구성 리포지토리에서 검색한 정보를 캐시합니다.

관리 도구는 에이전트와 완전히 분리됩니다. 즉, 구성 리포지토리에서만 작동합니다.

사용자 응용 프로그램은 구성 어댑터를 사용하여 Configuration Agent에서 구성 데이터를 조회합니다.

이 제품은 다음과 같은 구성 시스템의 설정 검색과 적용을 직접 지원합니다.

- GConf. Gnome 구성 프레임워크
- StarSuite 레지스트리
- Mozilla 기본 설정
- Java 기본 설정

◆ ◆ ◆ 2 장

2 관리 응용 프로그램 설치

이 장에서는 Sun Desktop Manager의 서버측 구성 요소를 설치하는 방법에 대해 설명합니다.

Sun Desktop Manager

Desktop Manager는 Java Web Console에서 실행되는 웹 기반 관리 도구를 제공합니다. 관리자는 이 사용자 인터페이스를 통해 조직 계층 구조를 이동하면서 데스크탑 응용 프로그램에 대한 정책을 정의할 수 있습니다. 계층 구조에서 조직, 역할, 사용자, 도메인, 호스트 등 각 항목에 대해 정책을 정의할 수 있습니다. Desktop Manager는 다양한 구성 템플리트를 사용하여 Gnome, Mozilla, StarSite, Evolution 등의 데스크탑 응용 프로그램별 설정을 표시합니다.

▼ 설치

시작하기 전에 Desktop Manager는 Java Web Console 2.2.5 이상 버전을 설치해야 합니다. 시스템에 유효한 버전이 설치되어 있는지 확인하십시오. 유효한 버전이 있는지 확인하려면 수퍼유저(루트)로 다음을 실행합니다.

```
# smcwebserver status
```

주 – Java Web Console 2.2.4가 Solaris™ 10 운영 체제에 포함되어 있지만 Desktop Manager는 2.2.5 이상 버전이 필요합니다. 2.2.5 버전은 server/console 디렉토리의 Desktop Manager 아카이브에서 제공됩니다. 이 디렉토리에서 `./setup`을 실행하여 2.2.5 버전을 설치할 수 있습니다.

시스템에 Java Web Console이 설치되어 있지 않거나 설치된 버전이 Desktop Manager에 대해 유효하지 않은 경우 Java Web Console을 처음 설치 또는 업데이트 시에 4 장의 지침을 참조하십시오. 그런 다음 이 장으로 돌아가서 Desktop Manager 설치를 계속합니다.

1. **Desktop Manager zip 아카이브를 다운로드한 다음 임시 디렉토리에 압축을 풉니다.**

   ```
   # unzip SunDesktopMgr-1.0.zip
   ```

2. **수퍼유저(루트)로 다음을 통해 설치 스크립트를 실행합니다.**

   ```
   # cd SunDesktopMgr-1.0/<platform>/server/manager
   # ./setup
   ```

3. **설치 스크립트의 출력에서 오류가 있는지 확인합니다.**
   설치가 성공적으로 수행된 경우 설치 스크립트가 Java Web Console을 자동으로 다시 시작하므로 웹 브라우저에서 Desktop Manager에 액세스할 수 있습니다.

▼ 작업

1. **브라우저에 다음 URL을 입력합니다.**

   ```
   https://<hostname>.<domainname>:6789
   ```

2. **로그인 화면에 기존 Unix 사용자의 사용자 이름과 암호를 입력합니다.**
   Java Web Console이 열립니다.

3. **콘솔 응용 프로그램 시작 페이지에서 Desktop Manager 링크를 클릭합니다.**

   - **콘솔 응용 프로그램 시작 페이지를 건너뛰고 Desktop Manager로 바로 이동하려면 브라우저에 다음 URL을 입력합니다.**

     ```
     https://<hostname>.<domainname>:6789/apoc
     ```

▼ Desktop Manager 제거

- **Java Web Console에서 Desktop Manager를 제거하려면 설치를 위해 만든 임시 디렉토리로 변경하고 수퍼유저(루트)로 다음을 실행합니다.**

  ```
  # cd server/manager
  # ./setup -u
  ```

마이그레이션 문제

Desktop Manager는 이전 버전 Java Desktop System Configuration Manager(릴리스 1.0 및 1.1)와 호환됩니다. 그러나 알아 두어야 할 몇 가지 차이점이 있습니다.

이전 Configuration Manager 버전에서는 모든 프로필 데이터가 하나의 지정된 LDAP 서버에 저장됩니다. 이 LDAP 서버는 전체 Configuration Manager 설치 절차의 일부로 구성됩니다. 또한 LDAP 서버에 대한 인증을 캡슐화하는 LDAP 로그인 모듈 구성을 포함합니다.

Desktop Manager의 경우 이제 모든 필요한 구성 단계가 마법사를 기반으로 수행되므로 더 이상 설치 중에 구성 작업을 수행할 필요가 없습니다. Desktop Manager가 이제 여러 개의 구성 리포지토리를 지원합니다. 따라서, 서로 다른 여러 LDAP 서버, 파일 기반 리포지토리 등에 저장된 정책 데이터를 관리할 수 있습니다. 특정 LDAP 로그인 모듈 구성이 더 이상 필요하지 않습니다.

버전 간에 LDAP 스키마 변경 사항이 없습니다. 이전 Configuration Manager 버전에 대해 LDAP 서버를 이미 구성한 경우 Desktop Manager로 전환할 때 구성을 변경할 필요가 없습니다. 따라서, 클라이언트(Java Desktop System Configuration Manager1.1. Agent)측 또는 LDAP측에서 업데이트하지 않고도 Desktop Manager를 사용할 수 있습니다.

주 – Desktop Manager를 설치하기 전에 이전 Configuration Manager 또는 Desktop Manager 설치를 먼저 제거해야 합니다. 이전 설치를 제거하려면 수퍼유저로 다음을 실행합니다.

```
# cd server/manager
# ./setup -u
```

Desktop Manager를 설치한 후 기존 LDAP 서버를 가리키는 구성 리포지토리를 만들 수 있습니다.

▼ 구성 리포지토리 만들기

1. **브라우저에 다음 URL을 입력합니다.**

   ```
   https://<hostname>.<domainname>:6789
   ```

2. **로그인 화면에 기존 Unix 사용자의 사용자 이름과 암호를 입력합니다.**

   Java Web Console이 열립니다.

3. **콘솔 응용 프로그램 시작 페이지에서 Sun Desktop Manager 1.0 링크를 클릭합니다.**

4. **새로 만들기 버튼을 클릭하여 구성 리포지토리 마법사를 시작합니다.**

   마법사가 LDAP 기반 구성 리포지토리를 구성하는 데 필요한 단계를 안내합니다.

주의 – 마법사는 기존 정책 데이터를 새 2.0 형식으로 자동으로 마이그레이션합니다. 이 마이그레이션은 선택적이며 최신 Sun Desktop Manager 1.0 에이전트의 성능 향상을 위해 주로 사용될 수 있습니다. 사용자 환경에서 Java Desktop System Configuration Manager 1.1 에이전트를 계속해서 지원해야 하는 경우 이 마이그레이션을 수행하지 마십시오.

Desktop Manager 문제 해결

설치할 수 없는 경우

증상: Java Web Console 설치가 끝난 후 등록된 응용 프로그램이 없기 때문에 시작할 수 없다는 메시지가 표시됩니다.

예상 원인: Desktop Manager를 포함하여 설치된 응용 프로그램이 없습니다..

해결책: Desktop Manager를 설치한 다음 Java Web Console을 시작합니다.

연결이 거부된 경우

증상: 적절한 URL(예: `http://<`*hostname*`>.<`*domainname*`>:6789`)을 열려고 했지만 연결이 거부되었다는 메시지가 표시됩니다.

예상 원인: Java Web Console이 서버에서 실행되고 있지 않습니다.

해결책: Java Web Console을 시작하려면 수퍼유저로 다음 명령을 실행합니다.

```
#smcwebserver status
#smcwebserver start
```

로그인할 수 없는 경우

증상: Java Web Console 로그인 페이지에서 사용자/암호 조합이 거부되었습니다.

예상 원인: 해당 UNIX 사용자 계정이 없습니다.

해결책: 시스템에 해당 UNIX 사용자 이름과 암호가 구성되어 있는지 확인합니다. 필요한 경우 테스트를 위한 로컬 UNIX 사용자 계정을 만듭니다.

Desktop Manager 링크가 없는 경우

증상: Java Web Console 응용 프로그램 목록 페이지에 Sun Desktop Manager 링크가 표시되지 않습니다.

예상 원인: Desktop Manager 모듈이 설치되지 않았습니다.

해결책: Desktop Manager가 Java Web Console에 설치되어 있는지 확인하려면 수퍼유저로 다음 명령을 실행합니다.

```
# smreg list -a
```

목록에 `com.sun.apoc.manager_<` *version*`>` 응용 프로그램이 없는 경우 Desktop Manager를 다시 설치해야 합니다.

Null 포인터 예외, Tomcat/Java 오류 또는 빈 페이지

증상: Desktop Manager를 시작하지만 빈 페이지만 열리거나 오류 메시지가 표시됩니다.

예상 원인: `NoClassDefFoundError:sun/tools/javac/Main` 오류가 발생하는 경우 Java Web Console이 잘못된 Java 버전을 사용하고 있는 것입니다.

해결책: 현재 Java Web Console Java 환경은 `# smreg list -p`를 실행한 후 `java.home` 등록 정보를 조사하여 확인할 수 있습니다. 이 등록 정보가 유효한 Java 홈을 가리키고 홈이 JDK여야 합니다. 이 값이 잘못 설정된 경우 다음 명령을 실행해야 합니다.

```
# smreg add -p java.home=<JAVA_HOME>
```

주 – <*JAVA_HOME*>이 유효한 설치를 가리켜야 합니다. 예를 들어, `javac`는 `bin` 하위 디렉토리에 있을 수 있습니다.

그리고 나서 다음 명령을 사용하여 Java Web Console을 다시 시작해야 합니다.

```
# smcwebserver restart
```

SSL LDAP 서버에 연결할 수 없는 경우

증상: 리포지토리 만들기 마법사에 LDAP 서버 정보를 제공하고 SSL 사용 상자를 선택하고, 다음을 누르면 서버에 연결할 수 없다는 메시지 상자가 표시됩니다.

예상 원인: 잘못된 포트 번호를 제공했거나, LDAP 서버가 해당 포트에서 SSL을 사용하여 연결을 수신하도록 구성되어 있지 않거나, Java Web Console 키 저장소에 해당 인증서가 없습니다.

해결책: 먼저 LDAP 서버가 마법사에 지정된 포트에서 SSL 연결 요청을 수신하도록 구성되어 있는지 확인합니다. 올바르게 구성되어 있는 경우 인증 기관 또는 LDAP 서버 인증서가 `/etc/opt/webconsole/keystore`에 있는 Java Web Console 키 저장소에 있는지 확인합니다. `keytool -import -file <certificate file> -keystore /etc/opt/webconsole/keystore` 명령을 사용하여 인증서를 추가할 수 있습니다. 이 키 저장소의 기본 암호는 **`changeit`**입니다. 변경 사항이 Desktop Manager에 표시되게 하려면 `smcwebserver restart` 명령을 사용하여 Java Web Console을 다시 시작해야 합니다.

디렉토리에 쓸 수 없는 경우

증상: 파일 기반 또는 하이브리드 백엔드를 만드는 동안 “디렉토리에 쓸 수 없습니다!” 라는 오류가 발생합니다.

예상 원인: noaccess 사용자가 올바른 권한이 없습니다.

해결책: noaccess 사용자에게 쓰기 권한을 지정합니다.

◆ ◆ ◆ 3 장

3 클라이언트 구성 요소

Desktop Manager에서 구성 데이터에 액세스하려면 데스크탑 클라이언트에 Java Desktop System Configuration Agent가 필요합니다. Configuration Agent는 원격 구성 데이터 리포지토리 및 어댑터와 통신하고 데이터를 특정 구성 시스템에 통합합니다. 현재 지원되는 구성 시스템은 GConf, Java 기본 설정, Mozilla 기본 설정 및 StarSuite 레지스트리입니다.

Configuration Agent 버전은 Solaris 10 운영 체제와 함께 제공됩니다. 그러나, Desktop Manager에는 최신 버전의 도구가 필요합니다. 이 최신 버전은 Desktop Manager 클라이언트 구성 요소 및 관련 패치 설치의 일부로 설치됩니다.

Desktop Manager 클라이언트 구성 요소를 설치하려면 다음 작업을 수행합니다.

1. Desktop Manager zip 아카이브를 다운로드한 다음 임시 디렉토리에 압축을 풉니다.

   ```
   # unzip SunDesktopMgr-1.0.zip
   ```

2. 권장 패치를 설치합니다.

   이러한 패치는 `SunDesktopMgr-1.0/<platform>/client/Patches` 디렉토리에 제공됩니다. 각 패치와 함께 제공된 설치 지침을 따릅니다.

3. 수퍼유저(루트)로 다음을 통해 설치 스크립트를 실행합니다.

   ```
   # cd SunDesktopMgr-1.0/<platform>/client
   # ./setup
   ```

Configuration Agent

Configuration Agent는 다음 표에 나열된 여러 패키지 중 일부입니다.

Solaris 패키지 이름	설명
SUNWapbas	Configuration Shared 라이브러리

Solaris 패키지 이름	설명
SUNWapmsc	Configuration Agent 기타 파일
SUNWapoc	Configuration Agent
SUNWapdc	Configuration Agent 마법사

이 패키지를 설치할 때 이 API에 필요한 파일이 설치됩니다. 패키지는 수동으로 설치하거나 Java Desktop System 설치를 통해 설치할 수 있습니다. 설치한 후에는 시스템에서 Configuration Agent를 구성하여 활성화해야 합니다.

주 – Configuration Agent 패키지는 Java Desktop System 설치에 Solaris의 일부로 설치되지만, Desktop Manager는 설치 중에 이러한 파일에 패치를 적용하여 적절한 수준의 기능을 제공합니다.

원격 구성 데이터에 액세스하려면 Configuration Agent에 LDAP 서버의 호스트 이름, 포트 등 최소한의 부트스트랩 정보를 제공해야 합니다. 이 정보는 `policymgr.properties`, `apocd.properties` 및 `os.properties` 같은 등록 정보 파일 집합으로 유지 관리됩니다. 이 파일은 `/etc/apoc` 디렉토리에 로컬로 저장되어 있습니다. 이러한 등록 정보 파일을 수동으로 편집하거나(부록 A 참조) Configuration Agent의 구성 마법사를 사용할 수 있습니다.

구성 마법사는 Configuration Agent의 필요한 설정을 안내하는 그래픽 사용자 인터페이스를 제공합니다. 각 마법사 페이지에서는 해당 도움말 화면을 사용할 수 있습니다. `/usr/bin/apoc-config` 스크립트를 사용하여 수퍼 유저(root)로 마법사를 시작할 수 있습니다.

주 – 그래픽 인터페이스를 시작하지 않고 마법사를 시작할 수도 있습니다. 예를 들어, 콘솔 모드로 마법사를 시작하려면 `/usr/bin/apoc-config -nodisplay`를 실행합니다.

부트스트랩 정보

그림 3–1 Configuration Agent, 구성 리포지토리

주 – 해당되는 경우 연결된 등록 정보 파일 키가 괄호 안에 표시되어 있습니다.

- 상태: Configuration Agent의 상태. 이 확인란을 사용하여 Configuration Agent를 활성화하거나 비활성화할 수 있습니다. 구성 리포지토리를 사용하려면 Configuration Agent를 활성화해야 합니다. 활성화는 Solaris에서 smf(5)(service management facility)에 필요한 등록을 자동으로 포함합니다.
- 호스트 식별자(HostIdentifierType): "HostName" 또는 "IPAddress" 중 하나일 수 있습니다. 호스트별 정책 데이터를 검색할 경우 Configuration Agent는 호스트 이름이나 IP 주소를 기준으로 현재 호스트를 식별합니다. 선택한 컨텍스트 유형에서 호스트가 식별되는 방법을 기반으로 올바른 값을 선택합니다.
- 컨텍스트 유형: 조직 계층 및 구성 데이터가 LDAP 또는 파일 기반 저장소 중 하나 또는 둘 모두에 정의되는지 여부를 Configuration Agent에 나타내려면 이 설정을 사용합니다.

주 – Configuration Agent를 수동으로 활성화하거나 비활성화하려면 **root**로 로그인하고 각각 `/usr/lib/apoc/apocd enable` 또는 `/usr/lib/apoc/apocd disable` 명령을 입력합니다.

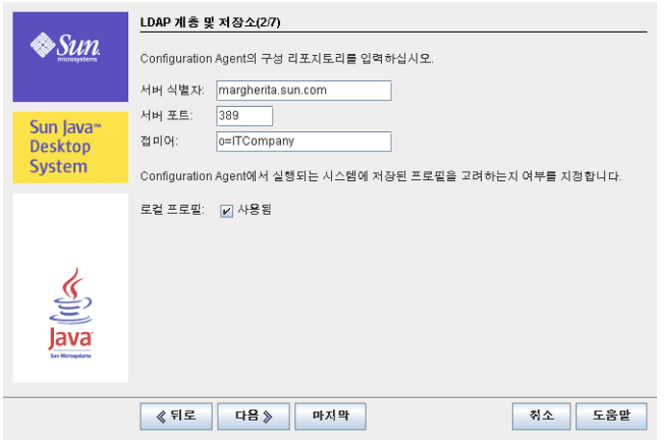

그림 3–2 Configuration Agent, LDAP 계층 및 파일 기반 저장소

주 – 그림 3–2의 화면은 이전 화면에서 선택한 컨텍스트 유형에 따라 달라집니다. LDAP 또는 하이브리드 컨텍스트 유형을 선택한 경우 서버 식별자, 서버 포트 및 접미어가 필요합니다. 파일 기반 또는 하이브리드 컨텍스트 유형을 선택한 경우 구성 설정 URL이 필요합니다.

- 서버 식별자: LDAP 서버의 호스트 이름
- 서버 포트: LDAP 서버의 포트 번호
- 접미어: LDAP 리포지토리의 기본 DN
- 구성 설정 URL: 파일 기반 리포지토리의 위치를 지정하는 URL.

  첫 번째 리포지토리에 대한 연결이 실패할 경우 URL 목록을 사용하여 폴백 리포지토리를 지정할 수 있습니다. 목록은 하나 이상의 공백으로 구분된 URL로 구성될 수 있으며 각 URL의 형식은 `file://<filepath>`, `http://<host>:<port>/<filepath>` 또는 `https://<host>:<port>/<filepath>`입니다. 자세한 내용은 부록 A를 참조하십시오.

  주 – 에이전트는 먼저 SSL 연결을 사용하여 LDAP 서버에 액세스하려고 시도합니다. 이 액세스가 실패하면 일반 SSL 연결을 시도합니다.

  SSL 연결이 성공하려면 Java Runtime Environment 키 저장소에 적절한 인증서가 있어야 합니다. 이 키 저장소는 표준 JRE의 경우 `<installation directory>/lib/security/cacerts`에 있고, 표준 JDK의 경우 `<installation directory>/jre/lib/security/cacerts`에 있습니다. `keytool -import -file <certificate file> -keystore <cacerts file location>` 명령을 사용하여 인증 기관 또는 LDAP 서버 인증서를 해당 저장소에 추가해야 합니다. 이 키 저장소의 기본 암호는 **`changeit`**입니다.

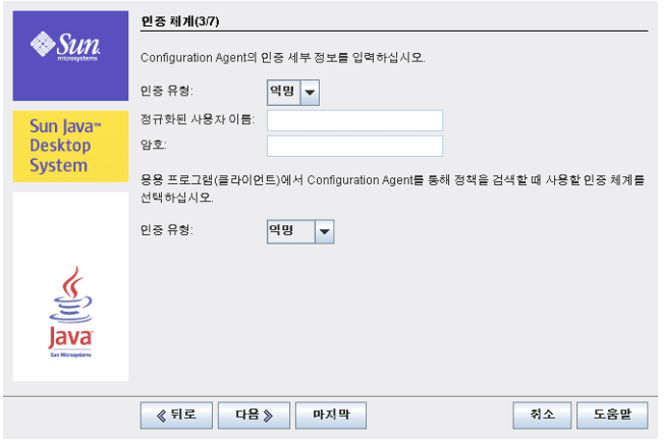

그림 3–3 Configuration Agent, 인증 체계

- Configuration Agent의 인증 유형: "익명" 또는 "단순" 중 하나일 수 있습니다. "익명"을 선택하면 정규화된 사용자 이름 및 암호 필드는 자동으로 사용할 수 없게 됩니다.
- 정규화된 사용자 이름(AuthDn): 리포지토리에 대한 읽기 및 검색 액세스 권한을 가진 사용자의 전체 DN
- 암호(Password): 등록된 LDAP 사용자 암호

주 – 디렉토리에서 익명 액세스가 활성화되면 정규화된 사용자 이름 및 암호 설정을 비워둘 수 있습니다.

- 응용 프로그램의 인증 유형(AuthType): LDAP 서버가 사용자를 인증하는 방법에 따라, 이 옵션은 "익명" 또는 "GSSAPI"가 될 수 있습니다.

주 – 자세한 내용은 24 페이지 "데이터 액세스/사용자 인증"을 참조하십시오.

포트 설정

Configuration Agent는 다음 두 포트를 사용합니다.

- 에이전트 포트(DaemonPort): 에이전트가 클라이언트 응용 프로그램과 통신하는 데 사용합니다(기본값: **38900**).
- 관리 포트(DaemonAdminPort): 에이전트 컨트롤러 프로그램인 apocd가 에이전트와 통신할 때 사용합니다(기본값: **38901**).

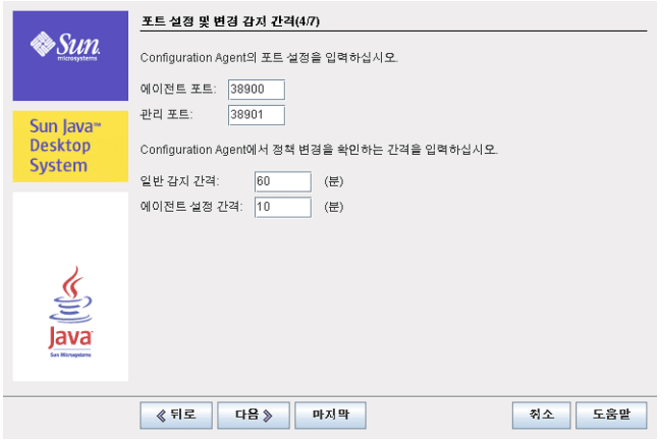

그림 3–4 Configuration Agent, 포트 설정

변경 감지 간격

Configuration Agent는 다음 두 가지 간격을 사용하여 구성 데이터의 변경 사항을 정기적으로 확인합니다.

- 일반 감지 간격(ChangeDetectionInterval): 데스크탑 응용 프로그램(클라이언트)의 구성 데이터에 대한 변경 감지 주기 간격(분)

  주 – -1로 지정하면 변경 감지가 수행되지 않습니다.

- 에이전트 설정 간격(DaemonChangeDetectionInterval): 에이전트 특정 구성 설정에 대한 변경 감지 주기 간격(분)

  주 – -1로 지정하면 변경 감지가 수행되지 않습니다.

일반 감지 간격을 사용하여 원하는 대로 원격 구성 데이터 변경 사항을 클라이언트측 응용 프로그램으로 전파할 수 있습니다. 이 설정에 지정되는 값은 원격 변경 사항이 클라이언트 응용 프로그램에서 적용될 때까지 최대 지연 시간(분)입니다.

작은 값을 지정하면 Configuration Agent와 LDAP 서버 작업이 증가하므로 설정 값을 변경할 때는 주의해야 합니다. 예를 들어, 초기 배포 단계에서 클라이언트 응용 프로그램에 대한 원격 구성의 영향을 테스트하기 위해 이 값을 1분으로 설정했다가 테스트가 끝나면 다시 이 설정을 원래 값으로 되돌릴 수 있습니다.

작동 설정

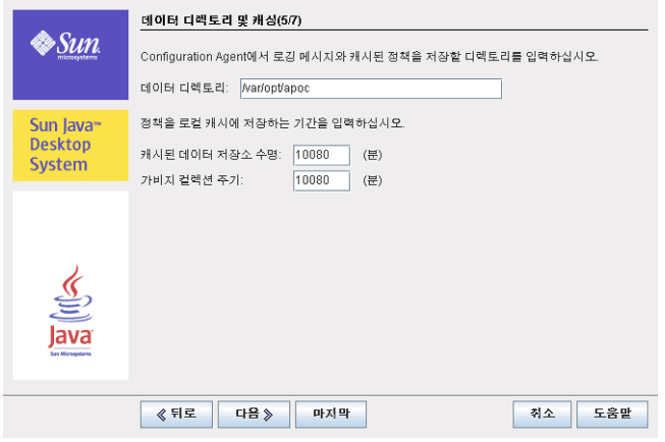

그림 3–5 Configuration Agent, 데이터 디렉토리

다음 설정을 구성할 수 있습니다.

- 데이터 디렉토리(DataDir): 런타임 데이터를 저장하는 데 사용되는 디렉토리. 기본값은 **`/var/opt/apoc`**입니다.
- 캐시된 데이터 저장소 수명(TimeToLive): 오프라인 상태가 아닌 구성 데이터가 로컬 데이터베이스에 보관되는 시간(분)

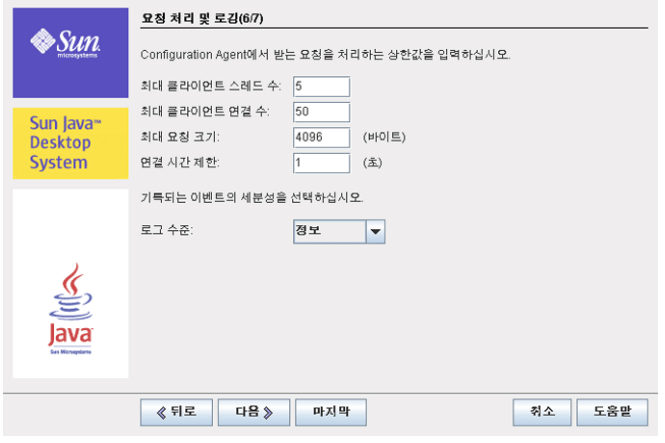

그림 3–6 Configuration Agent, 요청 처리 및 로깅

- 가비지 컬렉션 주기(GarbageCollectionInterval): 로컬 구성 데이터베이스의 가비지 컬렉션 주기 간격(분)

- 최대 클라이언트 스레드(MaxClientThreads): 동시에 처리될 수 있는 최대 클라이언트 요청 수
- 최대 클라이언트 연결(MaxClientConnections): 최대 클라이언트 연결 수
- 최대 요청 크기(MaxRequestSize): 최대 클라이언트 요청 크기
- 연결 시간 제한(ConnectTimeout): LDAP 서버가 연결 요청에 응답할 수 있는 간격. 기본값은 1초입니다.
- 로그 수준(LogLevel): 에이전트 로그 파일에 기록되는 세부 정보 수준. 로깅 수준은 Java Logger 수준과 같아야 합니다. 로깅 수준을 가장 높은 것부터 순서대로 나열하면 다음과 같습니다.
  - *SEVERE*
  - *WARNING*
  - *INFO*
  - *CONFIG*
  - *FINE*
  - *FINER*
  - *FINEST*

주 – 데이터 디렉토리 설정과 연결 시간 제한 설정을 제외한 대부분의 작동 설정은 LDAP 서버에 저장된 해당 정책을 통해 중앙 집중식으로 유지 관리할 수도 있습니다. 이 기능을 사용하려면 마법사를 사용하여 해당 설정을 적용하지 마십시오. 대신 Desktop Manager 내의 Configuration Agent 정책을 사용하여 작동 설정을 중앙 집중식으로 지정하십시오.

에이전트 설정 적용

"데이터 디렉토리" 및 "연결 시간 제한" 설정 외에 Desktop Manager를 사용하여 LDAP 서버에 저장된 작동 설정은 다음 에이전트 구성 변경 감지 주기에 자동으로 적용됩니다(DaemonChangeDetectionInterval 참조).

그림 3–7 Configuration Agent, 요약 페이지

로컬에서 변경된 다른 모든 설정의 경우에는 Configuration Agent를 다시 로드하거나 다시 시작해야 합니다. 구성 마법사를 사용하는 경우 다시 로드하거나 다시 시작하는 작업은 자동으로 수행됩니다.

주 – Configuration Agent를 수동으로 다시 시작하려면 관련된 클라이언트 응용 프로그램이 실행되고 있지 않은지 확인하고 루트로 로그인하여 `/usr/lib/apoc/apocd restart` 명령을 입력합니다.

추가 에이전트 설정

주 – 다음 설정은 구성 마법사에서 사용할 수 없습니다.

- 로컬 정책 적용(ApplyLocalPolicy): 로컬 호스트에 사용 가능한 정책 데이터를 클라이언트 응용 프로그램에 사용할 수 있는지 여부를 나타내려면 이 설정을 사용합니다. 값이 "true"이면 로컬 정책 데이터를 사용할 수 있습니다. 값이 "false"이면 로컬 정책 데이터를 사용할 수 없습니다. 자세한 내용은 23 페이지 “로컬 정책 사용”을 참조하십시오.

로컬 정책 사용

로컬로 배포된 정책의 구성 설정을 전역으로 사용 가능한 정책에 추가 또는 대체 항목으로 적용하도록 Configuration Agent를 구성할 수 있습니다. 이러한 로컬 정책을 배포하려면 다음 단계를 사용합니다.

▼ 로컬 정책 배포

1. **Desktop Manager를 사용하여 필수 정책 설정으로 프로필을 만듭니다.**
2. **Desktop Manager를 사용하여 프로필을 zip 파일로 내보냅니다.**
3. **클라이언트 호스트에서** `${DataDir}/Policies/profiles/PROFILE_REPOSITORY_default` **디렉토리를 만듭니다(없는 경우).**

   `${DataDir}`은 Configuration Agent의 데이터 디렉토리 값에 해당하며 기본적으로 `/var/opt/apoc`입니다.
4. **이전에 내보낸 zip 파일을** `${DataDir}/Policies/profiles/PROFILE_REPOSITORY_default`**로 복사합니다.**
5. **Configuration Agent가 사용 가능한 로컬 정책을 적용하도록 구성되어 있는지 확인합니다. 자세한 내용은 23 페이지 "추가 에이전트 설정"을 참조하십시오.**

주 – Configuration Agent의 "ApplyLocalPolicy" 설정을 변경하는 경우 `root`로 로그인한 다음 `/usr/lib/apoc/apocd reload` 명령을 입력하여 Configuration Agent를 다시 로드해야 합니다.

이러한 방법으로 배포되는 로컬 정책은 다음 Configuration Agent 변경 감지 주기 중에 클라이언트에서 사용할 수 있게 됩니다.

Configuration Agent 자동 재시작

오류가 발생할 경우 Configuration Agent가 자동으로 다시 시작됩니다. smf(5)(service management facility)가 이 의사 결정을 담당합니다. smf(5)가 재시작이 부적합하다고 판단하는 경우(예: 이미 너무 많은 오류가 발생한 경우) Configuration Agent는 유지 관리 모드로 전환됩니다.

Configuration Agent가 다시 시작되지 않는 경우 root로 로그인한 다음 `/usr/lib/apoc/apocd disable` 명령을 입력하여 에이전트를 일시적으로 비활성화하고, 에이전트 오류의 원인이 되는 문제를 해결한 다음 `/usr/lib/apoc/apocd enable` 명령을 실행하여 에이전트를 다시 활성화해야 합니다.

데이터 액세스/사용자 인증

Configuration Agent는 데스크탑 사용자의 로그인 ID에 따라 LDAP 서버로부터 정보를 검색합니다. 조직 매핑 파일의 `User/UniqueIdAttribute` 설정은 로그인 ID를 LDAP 서버의 사용자 요소로 매핑합니다. Configuration Agent는 호스트 이름이나 IP 주소와 같은 호스트 관련 정보도 검색합니다. 이 정보는 조직 매핑 파일의 `Host/UniqueIdAttribute` 설정을 통해 LDAP 서버의 호스트 요소로 매핑됩니다. 조직 매핑에 대한 자세한 내용은 부록 C을 참조하십시오.

LDAP 서버에 액세스하는 데는 익명과 GSSAPI의 두 가지 방법이 있습니다. 익명 액세스를 사용하는 경우 데스크탑에서 별도의 작업을 수행할 필요가 없습니다. GSSAPI 방법을 사용하려면 데스크탑에서 커버로스 자격 증명을 획득해야 합니다. 커버로스 자격 증명 획득과 사용자 로그인을 통합하려면 Java Desktop System 호스트에서 `pam_krb5` 모듈을 설치 및 구성해야 합니다.

`gdm`을 사용하여 커버로스와 사용자 로그인을 통합할 수도 있습니다. 예를 들어 다음 `/etc/pam.d/gdm` 파일을 사용합니다.

```
#%PAM-1.0
auth   required    pam_unix2.so  nullok #set_secrpc
auth   optional  pam_krb5.so use_first_pass missing_keytab_ok ccache=SAFE putenv_direct
account required    pam_unix2.so
password required    pam_unix2.so  #strict=false
session required    pam_unix2.so  # trace or none
session required    pam_devperm.so
session optional    pam_console.so
```

이런 방식으로 커버로스와 사용자 로그인을 통합하는 경우에는 화면 보호기의 커버로스 지원을 활성화해야 합니다. 예를 들어, 다음 `/etc/pam.d/xscreensaver` 파일을 사용합니다.

```
auth required pamkrb5.so use_first_pass missing_keytab_ok
ccache=SAFE putenv_direct
```

어댑터

응용 프로그램 어댑터는 Desktop Manager에서 지원되는 구성 시스템의 확장입니다. 다양한 응용 프로그램에서는 어댑터를 사용하여 구성 시스템에 따라 중앙 구성 데이터를 고려합니다. 지원되는 구성 시스템은 다음과 같습니다.

- GConf: Evolution처럼 데스크탑과 대부분의 Gnome 응용 프로그램에서 사용되는 Gnome 구성 시스템
- StarSuiteRegistry: StarSuite 및 OpenOffice.org에서 사용되는 구성 시스템
- Mozilla Preferences: Mozilla에 사용되는 구성 시스템
- Java Preferences: Java 응용 프로그램에 제공되는 구성 API

또한, 데스크탑 실행 프로그램, 메뉴 항목 및 시작 프로그램을 사용자 데스크탑에 추가하는 데스크탑 정의 어댑터가 제공됩니다.

GConf 어댑터

GConf 어댑터는 Solaris용 `SUNWapoc-adapter-gconf` 패키지의 일부입니다. 해당 packageAdapter에서 어댑터를 설치할 때 `/etc/gconf/2/path`에 있는 GConf 데이터 원본 경로는 Desktop Manager 소스를 포함하도록 업데이트됩니다. 어댑터는 다음 두 가지 데이터 원본을 제공합니다.

- "apoc:readonly:": 보호되지 않은 정책 설정에 대한 액세스를 제공합니다. 이 데이터 원본은 사용자 설정 뒤, 로컬 기본값 앞에 삽입합니다.
- "apoc:readonly:mandatory@": 보호된 정책 설정에 대한 액세스를 제공합니다. 이 데이터 원본은 로컬 필수 설정 뒤, 사용자 설정 앞에 삽입합니다.

GConf 어댑터 구성

GConf 어댑터는 설치의 일부로 구성되지만, 해당 작업은 필수 중앙 설정 및 기본 설정을 나타내는 두 데이터 원본의 GConf 경로 파일(`/etc/GConf/2/path`)에 따라 다릅니다. 이 경로 파일에는 시스템 설치 후 GConf가 중앙 설정을 예상대로 고려하는 데 필요한 정보가 포함되어 있지만, 관리자는 접두사 "apoc"가 붙은 데이터 원본이 파일에 있는지 확인하고, 추가 사용자 정의 데이터 원본을 포함하도록 해당 경로를 수정해야 합니다. 또한, 데이터 원본이 로컬 필수 설정과 필수 중앙 설정을 나타내는 데이터 원본의 사용자 설정 사이, 기본 중앙 설정을 나타내는 데이터 원본의 로컬 기본 설정과 사용자 설정 사이에 있는지 확인해야 합니다.

Java Preferences 어댑터

Java Preferences 어댑터는 Solaris용 `SUNWapcj` 패키지의 일부입니다.

Java Preferences 어댑터 구성

Java Preferences 어댑터는 다른 기존 구현에 대한 래퍼로 사용되어야 하는 기본 설정 API 구현으로 제공됩니다(예: JRE와 함께 제공된 기본 파일 기반 시스템). 기본 설정 API를 사용하는 Java 응용 프로그램의 중앙 구성을 사용하려면 `/usr/lib/apoc/apocjlaunch` 스크립트를 도우미로 사용하여 응용 프로그램에 대한 시작 스크립트를 작성해야 합니다. 이 스크립트는 몇 가지 환경 변수를 정의한 다음 `apocjlaunch` 스크립트(필수 환경에서 Java 응용 프로그램 시작)를 끝에 포함해야 합니다. 다음과 같은 환경 변수를 설정해야 합니다.

- `JAVA`: Java Runtime 실행 파일에 대한 경로 포함
- `APPLICATION`: 해당 응용 프로그램에 대해 정규 Java Runtime 호출의 뒷 부분을 포함합니다. 예를 들어, *classname [arguments]*(단일 클래스 시작) 또는 `-jar` *jarname [arguments]*(jar 아카이브 시작)를 포함합니다.

다음과 같은 선택적 추가 환경 변수를 설정할 수 있습니다.

- `CLASSPATH`: 콜론으로 구분된 jar 목록 또는 응용 프로그램 클래스 경로에 포함되어야 하는 클래스 파일
- `DEFINES`: 응용 프로그램 시작에 포함되어야 하는 정의 문을 포함하는 문자열
- `PREFFACTORY`: 응용 프로그램에서 사용해야 하는 기본 설정 API 구현의 팩토리 클래스 이름

Mozilla 어댑터

Mozilla 어댑터는 Solaris용 `SUNWmozapoc-adapter` 패키지의 일부입니다.

Mozilla 어댑터 구성

Mozilla 어댑터는 이 제품 설치의 일부로 설치되므로 추가 구성이 필요하지 않습니다.

StarSuite 어댑터

StarSuite 어댑터는 표준 StarSuite 설치에 포함되어 있으며 특별한 수정 없이 프로필 구성 데이터에 액세스할 수 있도록 지원합니다.

StarSuite 어댑터 구성

StarSuite 어댑터는 이 제품 설치의 일부로 설치되므로 추가 구성이 필요하지 않습니다.

데스크탑 정의 어댑터

데스크탑 정의 어댑터는 다음과 같은 패키지로 구성됩니다.

패키지 이름	설명
SUNWapleg	구성 액세스 바이너리
SUNWardsa	데스크탑 정의 어댑터
SUNWardsa-misc	어댑터에 대한 시스템 통합

이러한 패키지는 Desktop Manager 클라이언트 구성 요소를 설치할 때 설치되므로 추가 설정이 필요하지 않습니다.

데스크탑 정의 어댑터 구성

데스크탑 정의 어댑터는 사용자가 로그인할 때마다 설정 프로세스를 사용하여 구성되므로 추가 설정이 필요하지 않습니다.

어댑터 제거

Mozilla 및 StarSuite 어댑터는 이러한 제품을 제거하면 함께 제거됩니다. 해당 패키지 관리 시스템 도구를 사용하여 설치 절에서 설명한 패키지를 제거하면 GConf, Java Preferences 및 데스크탑 정의 어댑터를 제거할 수 있습니다.

Java Preferences 어댑터를 제거하면 기본 설정 API를 사용하여 Java 응용 프로그램을 시작하기 위해 작성된 시작 스크립트가 더 이상 사용되지 않습니다. 일부 필요한 클래스를 더 이상 사용할 수 없으므로 Java 호출에 실패합니다.

어댑터 문제 해결

해당 응용 프로그램에서 중앙 구성 데이터가 표시되지 않는 문제의 대부분은 모든 어댑터에서 데이터를 검색하는 데 사용되는 공통 메커니즘인 Configuration Agent로부터 비롯됩니다.

중앙 구성 변경이 지정된 설정 또는 그룹에 적용되지 않으면, 사용자는 응용 프로그램에서 주로 제품의 옵션 또는 기본 설정 대화 상자를 사용하여 해당 설정 값을 명시적으로 설정합니다. 이 경우, 중앙 설정이 보호됨으로 정의되어 있지 않으면(즉, 관리자가 해당 값을 강제로 적용하여 사용자가 수정할 수 없음) 사용자 기본 설정이 Desktop Manager를 사용하여 설정한 값보다 우선합니다.

Configuration Agent 문제 해결

이 절에서는 Configuration Agent의 특성 및 작업과 관련한 몇 가지 질문에 응답하고 에이전트 문제 해결을 위한 몇 가지 팁을 제공합니다.

질문과 대답

Configuration Agent는 무엇이고 어떻게 작동합니까?

Configuration Agent는 정책 캐싱 및 전달 응용 프로그램입니다. Configuration Agent는 데스크탑 클라이언트 응용 프로그램과 해당 응용 프로그램이 실행되는 호스트의 성능에 심각한 영향을 주지 않고 중앙에서 구성할 수 있도록 설계되었습니다. 이 작업은 다음과 같이 수행됩니다.

- 향후의 클라이언트 사용을 위해 로컬로 사용 가능한 캐시에 다운로드한 정책 캐싱
- 공유 가능하고 공유해야 하는 부담이 큰 리소스 공유(예: 정책이 호스팅되는 LDAP 서버에 연결)

클라이언트 응용 프로그램과 Configuration Agent 간의 상호 작용이 발생하는 일반적인 시나리오는 매우 간단하며 다음과 같이 설명될 수 있습니다.

1. 사용자가 관련 데스크탑 클라이언트 응용 프로그램(gconfd, Mozilla 또는 StarSuite) 중 하나를 시작합니다.
2. 클라이언트 응용 프로그램이 Configuration Agent에 연결됩니다.
3. 클라이언트 응용 프로그램이 Configuration Agent로부터 필요한 정책 데이터를 요청합니다.
4. Configuration Agent는 캐시에서 요청된 정책 데이터를 검색합니다.
5. 정책 데이터가 캐시에 없는 경우 Configuration Agent는 미리 구성된 정책 리포지토리에서 필요한 데이터를 다운로드하여 캐시에 저장합니다.
6. 정책 데이터가 요청한 클라이언트 응용 프로그램으로 전송됩니다.

7. Configuration Agent는 정책 리포지토리에서 정책 데이터에 대한 수정 사항을 모니터링합니다.
8. 수정 사항이 감지되면 Configuration Agent는 캐시를 새로 고쳐 최신 상태로 업데이트하고 클라이언트 응용 프로그램에 수정 사항을 알립니다.

Configuration Agent를 얻어서 설치하는 방법은 무엇입니까?

Configuration Agent는 기본적으로 Solaris 10과 함께 사용 가능하고 설치됩니다.

Solaris 10을 설치한 경우 다음에 수행할 작업은 무엇입니까?

Configuration Agent는 기본적으로 비활성화되어 있고 구성되어 있지 않습니다. Configuration Agent를 사용하려면 최소 수준 이상으로 구성하여 활성화해야 합니다. 이 단계를 완료하면 데스크탑 클라이언트 응용 프로그램이 자동으로 시작되고 다음 번에 시작될 때부터 사용자가 제공한 정책이 사용됩니다.

Configuration Agent를 구성하려면 어떻게 해야 합니까?

Configuration Agent를 올바르게 구성하려면 Configuration Agent 마법사를 사용합니다. 루트로 `/usr/bin/apoc-config` 명령을 실행하여 마법사를 시작할 수 있습니다. 마법사는 에이전트를 올바르게 구성하는 데 필요한 단계를 안내합니다. 대부분의 경우 마법사를 완료하는 데 절대적으로 필요한 유일한 정보는 정책 리포지토리의 위치입니다.

구성 파일을 수동으로 편집하여 Configuration Agent를 구성할 수도 있습니다. 수동으로 편집하여 구성하면 에이전트가 잘못 구성되기 쉬우므로 이 방법은 사용하지 않는 것이 좋습니다. 또한, Configuration Agent 마법사에는 에이전트를 다시 시작하거나 다시 로드하기 위해 특정 구성 변경이 필요한지 여부를 결정하는 추가 논리가 포함되어 있습니다.

Configuration Agent를 활성화하려면 어떻게 해야 합니까?

다음과 같은 세 가지 기법을 사용하여 에이전트를 활성화할 수 있습니다.

1. Configuration Agent 마법사(`/usr/bin/apoc-config`)를 사용하여 에이전트 상태를 활성으로 설정합니다.
2. Configuration Agent 컨트롤러 프로그램(`/usr/lib/apoc/apocd`)을 사용하여 루트로 다음을 실행합니다.

   ```
   /usr/lib/apoc/apocd enable
   ```

3. smf(5)를 사용하여 수퍼유저로 다음을 실행합니다.

   ```
   /usr/sbin/svcadm enable svc:/network/apocd/udp
   ```

Configuration Agent를 구성하여 활성화했습니다. 작동 여부를 확인하려면 어떻게 해야 합니까?

Configuration Agent가 올바르게 구성되어 있고 작동하는지 확인하는 가장 쉬운 방법은 Desktop Manager를 사용하여 정책을 만들고 해당 정책을 사용자에게 할당한 다음 데스크탑 컴퓨터에 해당 사용자로 로그인하여 정책 설정이 사용 중인지 확인하는 것입니다. 데스크탑 세션에는 배경, 주제 등과 같이 쉽게 감지될 수 있는 다양한 정책 설정이 있습니다.

Configuration Agent를 활성화해야 하는 이유는 무엇입니까?

Configuration Agent는 smf(5) 호환 서비스이므로 에이전트 활성화 알림이 smf(5)에서 제공됩니다. 에이전트가 활성화되면 서비스를 제공할 준비가 된 것입니다. 에이전트를 활성화하면 다음과 같은 동작이 나타납니다.

- 에이전트가 시작됩니다.
- 에이전트가 활성화된 후 시작된 모든 데스크탑 클라이언트 응용 프로그램에서 정책 데이터를 검색할 수 있습니다.
- 에이전트는 시스템이 부트하는 동안에 자동으로 다시 시작됩니다.

Configuration Agent가 활성화되는지 확인하려면 어떻게 해야 합니까?

다음 방법 중 하나를 사용하여 Configuration Agent가 활성화되는지 여부를 확인할 수 있습니다.

- Configuration Agent의 컨트롤러 프로그램을 사용합니다. 수퍼유저로 다음 명령을 입력합니다.

  ```
  /usr/lib/apoc/apocd is-enabled
  ```

  에이전트가 활성화되는 경우 컨트롤러 프로그램이 다음과 같은 메시지를 반환합니다.

  ```
  Checking Configuration Agent enabled status ... Enabled
  ```

  그렇지 않으면 컨트롤러 프로그램이 다음과 같은 메시지를 반환합니다.

  ```
  Checking Configuration Agent enabled status ... Not enabled
  ```

- smf(5)를 사용하여 다음 명령을 실행합니다.

  ```
  /usr/bin/svcs svc:/network/apocd/udp:default
  ```

  에이전트가 활성화되면 svcs는 다음과 같은 메시지를 반환합니다.

  ```
  STATE          STIME    FMRI
  online          8:36:04 svc:/network/apocd/udp:default
  ```

  에이전트가 비활성화되면 svcs는 다음과 같은 메시지를 반환합니다.

```
STATE          STIME    FMRI
disabled       15:58:34 svc:/network/apocd/udp:default
```

에이전트가 유지 관리 모드인 경우 svcs는 다음과 같은 메시지를 반환합니다.

```
STATE          STIME    FMRI
maintenance     8:38:42 svc:/network/apocd/udp:default
```

Configuration Agent가 실행 중인지 확인하려면 어떻게 해야 합니까?

다음 방법 중 하나를 사용하여 Configuration Agent가 실행 중인지 확인할 수 있습니다.

- 수퍼유저로 Configuration Agent 컨트롤러 프로그램을 실행합니다.

  ```
  /usr/lib/apoc/apocd status
  ```

  에이전트가 활성화되는 경우 컨트롤러 프로그램이 다음과 같은 메시지를 반환합니다.

  ```
  Checking Configuration Agent status ... Running
  ```

  그렇지 않으면 컨트롤러 프로그램이 다음과 같은 메시지를 반환합니다.

  ```
  Checking Configuration Agent status ... Not running
  ```

- 다음 명령을 실행합니다.

  ```
  /usr/bin/svcs svc:/network/apocd/udp:default
  ```

  에이전트가 실행 중이면 svcs는 다음과 같은 메시지를 반환합니다.

  ```
  STATE          STIME    FMRI
  online          8:36:04 svc:/network/apocd/udp:default
  ```

  에이전트가 실행 중이 아니면 svcs는 다음과 같은 메시지를 반환합니다.

  ```
  STATE          STIME    FMRI
  disabled       15:58:34 svc:/network/apocd/udp:default
  ```

  에이전트가 유지 관리 모드인 경우 svcs는 다음과 같은 메시지를 반환합니다.

  ```
  STATE          STIME    FMRI
  maintenance     8:38:42 svc:/network/apocd/udp:default
  ```

- 다음 명령을 실행합니다.

  ```
  ps -ef | grep apoc
  ```

  Configuration Agent가 실행 중이면 ps 출력에 다음과 같은 연관된 Java 프로세스가 표시되어야 합니다.

```
 daemon 29295 29294  0 13:05:22?  0:03 java -Djava.library.path=/usr/lib/apoc
 -cp /usr/share/lib/apoc/apocd.jar:/usr/s
 daemon 29294   1   0 13:05:22?  0:00 sh -c java
-Djava.library.path=/usr/lib/apoc -cp /usr/share/lib/apoc/apocd.jar:
 root 29345 28134  0 13:08:59 pts/1   0:00 grep apoc
```

로그 파일의 위치는 어디입니까?

Configuration Agent 문제를 해결해야 하는 경우 다음 로그 파일을 참조할 수 있습니다.

- smf(5) 로그 파일:
  - `/var/svc/log/network-apocd-udp:default.log` 파일은 Configuration Agent의 특정 인스턴스를 시작/중지하려는 시도와 관련된 이벤트를 기록합니다. 또한, 이 파일에는 Configuration Agent 컨트롤러 프로그램인 `/usr/lib/apoc/apocd`가 표준 출력에 쓰는 메시지와 JVM 또는 Configuration Agent의 출력 메시지가 포함되어 있습니다.
  - `/var/svc/log/svc.startd.log` 로그 파일에는 상위 수준의 smf(5) 이벤트 레코드가 보관됩니다. 예를 들어, Configuration Agent를 시작하려는 시도가 연속해서 여러 번 실패할 경우 smf(5)는 Configuration Agent를 시작할 수 없는 것으로 결정할 수 있습니다. 이 경우 smf(5)는 Configuration Agent를 유지 관리 모드로 전환하고 해당 효과를 로그 항목에 씁니다.

  이 두 로그 파일은 Configuration Agent 시작 관련 문제가 발생하는 경우에 유용합니다.
- Configuration Agent 로그:

  Configuration Agent는 기본 로그 디렉토리인 `/var/opt/apoc/Logs`에 있는 로그 파일에 로그 메시지를 씁니다. Configuration Agent의 "데이터 디렉토리"는 `/var/opt/apoc`입니다. Configuration Agent 마법사(`/usr/bin/apoc-config`) 응용 프로그램을 사용하여 이 디렉토리 위치를 변경할 수 있습니다. Configuration Agent 마법사에서 "로그 수준"을 변경하여 로그 메시지의 세분성을 변경할 수 있습니다. Configuration Agent를 잘못 구성했거나 다른 유형의 에이전트 오류가 발생하는 경우 에이전트 로그 파일을 참조하기 전에 Configuration Agent 마법사를 사용하여 로그 수준을 "Finest"로 설정합니다. 이 단계에서는 사용 가능한 로깅 정보를 최대한 수집할 수 있습니다.
- 시스템 로그:

  `/var/adm/messages` 로그 파일 또는 SunRay 시스템의 `/var/opt/SUNWut/log/messages` 로그 파일을 확인하여 Configuration Agent 관련 문제를 진단할 수도 있습니다.

에이전트 로깅 메커니즘의 세분성을 높이려면 어떻게 해야 합니까?

32 페이지 “로그 파일의 위치는 어디입니까?”를 참조하십시오.

유지 관리 모드는 무엇입니까?

smf(5)는 에이전트 시작 또는 다시 시작 문제가 감지될 경우 Configuration Agent를 유지 관리 모드로 전환합니다. smf(5)는 에이전트를 시작하지 못할 경우 에이전트가 성공적으로 시작되거나 smf(5)가 에이전트를 시작할 수 없다고 결정할 때까지 여러 번 다시 시작합니다.

후자의 경우 smf(5)는 에이전트를 유지 관리 모드로 전환하여 발생한 문제를 해결해야 함을 나타냅니다. 문제를 해결한 경우 에이전트의 smf(5) 상태를 지우고 일반 모드로 전환할 수 있습니다.

smf(5) 상태를 지우고 유지 관리 모드에서 나가려면 어떻게 해야 합니까?

수퍼유저로 `/usr/sbin/svcadm clear svc:/network/apocd/udp` 명령을 실행합니다.

Configuration Agent가 예기치 않게 중지되면 어떻게 됩니까?

smf(5)는 에이전트의 중지를 감지하고 다시 시작하려고 시도합니다. 어떠한 이유로 인해 연속적인 몇 번의 시도에도 다시 시작되지 않을 경우 smf(5)는 에이전트를 유지 관리 모드로 전환합니다. 에이전트가 성공적으로 다시 시작될 경우 실행 중인 데스크탑 클라이언트 응용 프로그램은 영향을 받지 않습니다. 이러한 클라이언트 응용 프로그램은 에이전트가 다시 시작되면 자동으로 다시 연결됩니다.

에이전트를 활성화/시작할 경우 데스크탑 클라이언트 응용 프로그램을 다시 시작해야 합니까?

실제로 수행할 작업은 특정 데스크탑 클라이언트 응용 프로그램이 시작될 때 에이전트가 활성/실행 중이었는지 여부에 따라 다릅니다. 에이전트가 활성/실행 중이었으면 클라이언트 응용 프로그램이 에이전트에 연결된 상태이며 연결이 끊어지면 다시 연결하려고 시도합니다. 즉 에이전트를 시작, 활성화 또는 비활성화할 때마다 클라이언트 응용 프로그램은 실행 중인 에이전트에 다시 연결하려고 항상 시도합니다. 클라이언트 응용 프로그램을 시작할 때 에이전트가 활성/실행 중이 아니면 클라이언트 응용 프로그램은 Configuration Agent를 사용하지 않고 에이전트가 시작되더라도 연결하려고 시도하지 않습니다. 요약하면 다음과 같습니다.

- 에이전트가 활성/실행 중일 때 시작된 데스크탑 클라이언트 응용 프로그램은 다시 시작할 필요가 없습니다.
- 에이전트가 활성/실행 중이 아닐 때 시작된 데스크탑 클라이언트 응용 프로그램은 다시 시작해야 합니다.

내 데스크탑 클라이언트 응용 프로그램이 구성된 정책을 사용하지 않는 것 같습니다. 어떻게 해야 합니까?

Configuration Agent와 관련한 가장 일반적인 문제는 데스크탑 클라이언트 응용 프로그램에 구성된 정책의 효과가 나타나지 않는 것입니다. 이 문제의 가장 일반적인 이유는 에이전트 또는 정책 리포지토리를 잘못 구성했거나 정책 리포지토리를 사용할 수 없기 때문입니다. 이런 경우의 문제는 다음 지침에 따라 찾아서 해결할 수 있습니다.

- 에이전트가 구성되어 있는지 확인합니다.
- 에이전트가 활성/실행 중인지 확인합니다. 에이전트를 시작해야 하는 경우 현재 열려 있는 데스크탑 클라이언트 응용 프로그램도 다시 시작해야 합니다.

- 문제가 지속되면 에이전트가 시작된 시간부터 전체 로그 메시지가 세분화되어 표시되도록 에이전트 로깅 메커니즘의 세분성을 일시적으로 늘리고, 가능하면 에이전트를 다시 시작하십시오.
- 에이전트가 제대로 시작되지 않으면 34 페이지 “Configuration Agent 시작 문제”를 참조하십시오.
- 에이전트가 제대로 시작되지만 데스크탑 클라이언트 응용 프로그램이 사용 가능한 정책을 사용하지 않는 경우 "실행 중인 Configuration Agent에서 정책 가져오기 문제" 절을 참조하십시오.
- 그래도 문제를 해결할 수 없는 경우 기술 지원 부서에 문의하십시오.

Configuration Agent 시작 문제

Configuration Agent를 시작할 수 없지만 Configuration Agent를 제대로 구성하여 활성화한 경우 로그 파일을 참조해야 합니다. 다음 절에서는 이 문제의 가장 일반적인 오류에 대해 설명합니다.

에이전트 데이터 디렉토리가 잘못되었거나 액세스할 수 없습니다.

Configuration Agent 데이터 디렉토리는 에이전트가 로그 파일, 정책 캐시 등을 만들고 저장하는 데 사용합니다. 이 디렉토리의 기본 위치는 `/var/opt/apoc`입니다.

Configuration Agent는 데이터 디렉토리가 액세스할 수 없는 위치(즉, `/dev/null/cant/write/here`)로 설정되어 있는 경우 smf(5) 로그에 다음과 같은 오류 메시지를 표시합니다. 이 문제를 해결하려면 Configuration Agent 마법사(`/usr/bin/apoc-config`)를 사용하여 데이터 디렉토리를 액세스 가능한 위치로 지정합니다.

```
[ Nov 17 14:35:38 Executing start method ("/usr/lib/apoc/apocd svcStart") ]
Starting Configuration Agent ... Warning: Cannot create Log directory
   ’/dev/null/cant/write/here/Logs’
Warning:Failed to create log file handler
Nov 17, 2005 2:35:39 PM com.sun.apoc.daemon.misc.APOCLogger config
CONFIG: Daemon configuration:
 MaxRequestSize = 4096
 DaemonAdminPort = 38901
 ThreadTimeToLive = 5
 DaemonChangeDetectionInterval = 10
 IdleThreadDetectionInterval = 15
 PROVIDER_URL =
 DataDir = /dev/null/cant/write/here
 ApplyLocalPolicy = true
 ChangeDetectionInterval = 60
 MaxClientConnections = 50
 GarbageCollectionInterval = 10080
 InitialChangeDetectionDelay = 10
 TimeToLive = 10080
 ConnectionReadTimeout = 5000
```

```
 DaemonPort = 38900
 LogLevel = FINEST
 MaxClientThreads = 5


Nov 17, 2005 2:35:39 PM Daemon main
FINER: THROW
com.sun.apoc.daemon.misc.APOCException
    at com.sun.apoc.daemon.apocd.Daemon.initAuthDir(Unknown Source)
    at com.sun.apoc.daemon.apocd.Daemon.init(Unknown Source)
    at com.sun.apoc.daemon.apocd.Daemon.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.Daemon.main(Unknown Source)
[ Nov 17 14:36:08 Method or service exit timed out.  Killing contract 980 ]
[ Nov 17 14:36:08 Method "start" failed due to signal KILL ]
```

이미 사용 중인 클라이언트 요청 포트 사용

Configuration Agent는 TCP/IP 소켓 연결을 사용하여 데스크탑 클라이언트 응용 프로그램과 통신합니다. 기본적으로 이러한 연결은 포트 38900을 통해 설정됩니다.

다른 서비스에서 이미 사용 중인 포트 1234를 사용하도록 Configuration Agent를 구성할 경우 다음과 같은 오류 메시지가 표시됩니다. 오류 메시지는 Configuration Agent 로그에 기록됩니다. 이 문제를 해결하려면 Configuration Agent 마법사(`/usr/bin/apoc-config`)를 사용하여 에이전트 포트 설정을 사용 중이 아닌 포트 번호로 변경합니다.

```
Nov 17, 2005 2:50:59 PM com.sun.apoc.daemon.misc.APOCLogger config
CONFIG: Daemon configuration:
 MaxRequestSize = 4096
 DaemonAdminPort = 38901
 ThreadTimeToLive = 5
 DaemonChangeDetectionInterval = 10
 IdleThreadDetectionInterval = 15
 PROVIDER_URL =
 DataDir = /var/opt/apoc
 ApplyLocalPolicy = true
 ChangeDetectionInterval = 60
 MaxClientConnections = 50
 GarbageCollectionInterval = 10080
 InitialChangeDetectionDelay = 10
 TimeToLive = 10080
 ConnectionReadTimeout = 5000
 DaemonPort = 1234
 LogLevel = FINEST
 MaxClientThreads = 5


Nov 17, 2005 2:50:59 PM com.sun.apoc.daemon.misc.APOCLogger info
INFO: Daemon starting
```

```
Nov 17, 2005 2:50:59 PM com.sun.apoc.daemon.misc.APOCLogger fine
FINE: Garbage collection scheduled ( interval = 10080 minutes )
Nov 17, 2005 2:50:59 PM Daemon main
FINER: THROW
com.sun.apoc.daemon.misc.APOCException: java.net.BindException: Address already in use
    at com.sun.apoc.daemon.transport.ChannelManager.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.Daemon.run(Unknown Source)
    at com.sun.apoc.daemon.apocd.Daemon.main(Unknown Source)
Caused by: java.net.BindException: Address already in use
    at sun.nio.ch.Net.bind(Native Method)
    at sun.nio.ch.ServerSocketChannelImpl.bind(ServerSocketChannelImpl.java:119)
    at sun.nio.ch.ServerSocketAdaptor.bind(ServerSocketAdaptor.java:59)
    at sun.nio.ch.ServerSocketAdaptor.bind(ServerSocketAdaptor.java:52)
```

이미 사용 중인 관리 포트 사용

Configuration Agent는 TCP/IP 소켓 연결을 사용하여 Configuration Agent 컨트롤러 프로그램(`/usr/lib/apoc/apocd`)과 통신합니다. 기본적으로 이러한 연결은 포트 38901을 통해 설정됩니다.

다른 서비스에서 이미 사용 중인 포트 1234를 사용하도록 Configuration Agent를 구성할 경우 다음과 같은 오류 메시지가 Configuration Agent 로그에 표시됩니다. 이 문제를 해결하려면 Configuration Agent 마법사(`/usr/bin/apoc-config`)를 사용하여 관리 포트 설정을 사용 중이 아닌 포트 번호로 변경합니다.

```
ONFIG: Daemon configuration:
 MaxRequestSize = 4096
 DaemonAdminPort = 1234
 ThreadTimeToLive = 5
 DaemonChangeDetectionInterval = 10
 IdleThreadDetectionInterval = 15
 PROVIDER_URL =
 DataDir = /var/opt/apoc
 ApplyLocalPolicy = true
 ChangeDetectionInterval = 60
 MaxClientConnections = 50
 GarbageCollectionInterval = 10080
 InitialChangeDetectionDelay = 10
 TimeToLive = 10080
 ConnectionReadTimeout = 5000
 DaemonPort = 38900
 LogLevel = FINEST
 MaxClientThreads = 5


Nov 17, 2005 2:55:11 PM com.sun.apoc.daemon.misc.APOCLogger info
INFO: Daemon starting
Nov 17, 2005 2:55:11 PM com.sun.apoc.daemon.misc.APOCLogger fine
```

```
FINE: Garbage collection scheduled ( interval = 10080 minutes )
Nov 17, 2005 2:55:11 PM com.sun.apoc.daemon.misc.APOCLogger fine
FINE: Client manager started
Nov 17, 2005 2:55:11 PM com.sun.apoc.daemon.misc.APOCLogger fine
FINE: Channel manager started
Nov 17, 2005 2:55:11 PM Daemon main
FINER: THROW
com.sun.apoc.daemon.misc.APOCException: java.net.BindException: Address already in use
    at com.sun.apoc.daemon.admin.AdminManager.initChannel(Unknown Source)
    at com.sun.apoc.daemon.admin.AdminManager.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.Daemon.run(Unknown Source)
    at com.sun.apoc.daemon.apocd.Daemon.main(Unknown Source)
Caused by: java.net.BindException: Address already in use
    at sun.nio.ch.Net.bind(Native Method)
    at sun.nio.ch.ServerSocketChannelImpl.bind(ServerSocketChannelImpl.java:119)
    at sun.nio.ch.ServerSocketAdaptor.bind(ServerSocketAdaptor.java:59)
    at sun.nio.ch.ServerSocketAdaptor.bind(ServerSocketAdaptor.java:52)
    ... 4 more
```

실행 중인 Configuration Agent에서 정책 가져오기 문제

구성 리포지토리 지정이 누락되었거나 잘못되었습니다.

정책 정보를 다운로드하여 캐시하려면 Configuration Agent를 유효한 구성 리포지토리에 연결해야 합니다. 에이전트 구성에서 구성 리포지토리를 올바르게 식별하지 않은 경우(예: 잘못된 형식을 사용했거나 리포지토리를 지정하지 않은 경우) 데스크탑 클라이언트 응용 프로그램을 시작하면 Configuration Agent 로그에 다음과 유사한 오류 메시지가 기록됩니다. 이 문제를 해결하려면 Configuration Agent 마법사(`/usr/bin/apoc-config`)를 사용하여 사용할 구성 리포지토리를 식별합니다.

```
FINER: New client added
Nov 18, 2005 1:59:22 PM com.sun.apoc.daemon.misc.APOCLogger finer
FINER: CreateSession transaction started
Nov 18, 2005 1:59:22 PM com.sun.apoc.daemon.misc.APOCLogger finer
FINER: Creating new client session
Nov 18, 2005 1:59:22 PM com.sun.apoc.daemon.misc.APOCLogger finest
FINEST: Authenticating user geoffh
Nov 18, 2005 1:59:22 PM com.sun.apoc.daemon.misc.APOCLogger finest
FINEST: Authentication successful
Nov 18, 2005 1:59:23 PM PolicyBackend openPolicyBackend
FINER: THROW
com.sun.apoc.daemon.misc.APOCException: com.sun.apoc.daemon.misc.APOCException:
com.sun.apoc.spi.environment.InvalidParameterException: The parameter organisation
PROVIDER_URL#protocol (null) is not valid, the value must be comprised in
{ldaps,ldap,https,http,file}.
    at com.sun.apoc.daemon.apocd.PolicyBackend.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.HostPolicyBackend.<init>(Unknown Source)
```

```
    at com.sun.apoc.daemon.apocd.PolicyBackendFactory.openPolicyBackend(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache$DataSource.openPolicyBackend(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache$DataSource.open(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache.createDataSources(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.CacheFactory.createNewCache(Unknown Source)
    at com.sun.apoc.daemon.apocd.CacheFactory.openCache(Unknown Source)
    at com.sun.apoc.daemon.apocd.Session.<init>(Unknown Source)
    at com.sun.apoc.daemon.transaction.CreateSessionTransaction.executeTransaction
(Unknown Source)
    at com.sun.apoc.daemon.transaction.Transaction.execute(Unknown Source)
    at com.sun.apoc.daemon.apocd.ClientEventHandler.handleEvent(Unknown Source)
    at com.sun.apoc.daemon.apocd.EventWorkerThread.run(Unknown Source)
Caused by: com.sun.apoc.daemon.misc.APOCException:
 com.sun.apoc.spi.environment.InvalidParameterException:
 The parameter organisation PROVIDER_URL#protocol (null) is not valid,
 the value must be comprised in {ldaps,ldap,https,http,file}.
    at com.sun.apoc.daemon.apocd.PolicyBackendFactory.openPolicyMgr(Unknown Source)
    ... 14 more
Caused by: com.sun.apoc.spi.environment.InvalidParameterException: The parameter
organisation PROVIDER_URL#protocol (null) is not valid, the value must be comprised in
{ldaps,ldap,https,http,file}.
    at com.sun.apoc.spi.PolicyMgrFactoryImpl.createPolicyMgr(Unknown Source)
    ... 15 more
Nov 18, 2005 1:59:23 PM PolicyBackend openPolicyBackend
```

정책 리포지토리에 연결할 수 없음

정책 정보를 다운로드하여 캐시하려면 Configuration Agent를 유효한 구성 리포지토리에 연결해야 합니다. 연결을 설정할 수 없는 경우 데스크탑 클라이언트 응용 프로그램을 시작하면 Configuration Agent 로그에 다음과 유사한 오류 메시지가 기록됩니다. 다음과 같은 경우 sobuild 호스트가 없거나 연결되지 않거나, 포트 389를 통해 LDAP 서버에 액세스할 수 없습니다. 이 문제를 해결하려면 에이전트 구성 마법사(`/usr/bin/apoc-config`)를 사용하여 정책 리포지토리를 올바르게 식별했는지 확인하고, 올바르게 식별한 경우 정책 리포지토리에 액세스 가능한지 확인합니다. 예를 들어, LDAP 리포지토리의 경우 LDAP 서버가 실행 중이고 LDAP 서버를 호스트하는 시스템이 네트워크에서 사용 가능하며, 지정한 포트가 LDAP 서버에 사용 중인지 확인해야 합니다.

SSL 연결을 사용하여 LDAP 서버에 액세스할 경우 Configuration Agent를 실행하는 데 사용되는 Java 런타임 환경과 연관된 키 저장소에서 적절한 인증서를 사용할 수 있는지 확인합니다. `apoc-config`에 대한 자세한 내용은 15 페이지 “Configuration Agent” 절을 참조하십시오.

```
FINER: New client added
Nov 18, 2005 2:17:43 PM com.sun.apoc.daemon.misc.APOCLogger finer
FINER: CreateSession transaction started
Nov 18, 2005 2:17:43 PM com.sun.apoc.daemon.misc.APOCLogger finer
FINER: Creating new client session
```

```
Nov 18, 2005 2:17:43 PM com.sun.apoc.daemon.misc.APOCLogger finest
FINEST: Authenticating user geoffh
Nov 18, 2005 2:17:43 PM com.sun.apoc.daemon.misc.APOCLogger finest
FINEST: Authentication successful
Nov 18, 2005 2:17:43 PM PolicyBackend openPolicyBackend
FINER: THROW
com.sun.apoc.daemon.misc.APOCException: com.sun.apoc.daemon.misc.APOCException:
com.sun.apoc.spi.OpenConnectionException: An error occured while connecting to
ldap://sobuild:389.
    at com.sun.apoc.daemon.apocd.PolicyBackend.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.HostPolicyBackend.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.PolicyBackendFactory.openPolicyBackend(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache$DataSource.openPolicyBackend(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache$DataSource.open(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache.createDataSources(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.CacheFactory.createNewCache(Unknown Source)
    at com.sun.apoc.daemon.apocd.CacheFactory.openCache(Unknown Source)
    at com.sun.apoc.daemon.apocd.Session.<init>(Unknown Source)
    at com.sun.apoc.daemon.transaction.CreateSessionTransaction.executeTransaction
(Unknown Source)
    at com.sun.apoc.daemon.transaction.Transaction.execute(Unknown Source)
    at com.sun.apoc.daemon.apocd.ClientEventHandler.handleEvent(Unknown Source)
    at com.sun.apoc.daemon.apocd.EventWorkerThread.run(Unknown Source)
Caused by: com.sun.apoc.daemon.misc.APOCException:
com.sun.apoc.spi.OpenConnectionException: An error occured while
connecting to ldap://sobuild:389. at
com.sun.apoc.daemon.apocd.PolicyBackendFactory.openPolicyMgr(Unknown Source)
    ... 14 more
Caused by: com.sun.apoc.spi.OpenConnectionException: An error occured while
connecting to ldap://noSuchHost:389.
    at com.sun.apoc.spi.ldap.LdapClientContext.prepareConnection(Unknown Source)
    at com.sun.apoc.spi.ldap.LdapClientContext.connect(Unknown Source)
    at com.sun.apoc.spi.ldap.LdapConnectionHandler.openAuthorizedContext(Unknown Source)
    at com.sun.apoc.spi.ldap.LdapConnectionHandler.connect(Unknown Source)
    at com.sun.apoc.spi.ldap.entities.LdapOrganizationProvider.open(Unknown Source)
    at com.sun.apoc.spi.PolicyMgrFactoryImpl.createPolicyMgr(Unknown Source)
    ... 15 more
Caused by: netscape.ldap.LDAPException: failed to connect to server sobuild:389 (91);
Cannot connect to the LDAP server
    at netscape.ldap.LDAPConnSetupMgr.connectServer(LDAPConnSetupMgr.java:422)
    at netscape.ldap.LDAPConnSetupMgr.openSerial(LDAPConnSetupMgr.java:350)
    at netscape.ldap.LDAPConnSetupMgr.connect(LDAPConnSetupMgr.java:244)
    at netscape.ldap.LDAPConnSetupMgr.access$0(LDAPConnSetupMgr.java:241)
    at netscape.ldap.LDAPConnSetupMgr$1.run(LDAPConnSetupMgr.java:179)
    at java.lang.Thread.run(Thread.java:595)
Nov 18, 2005 2:17:44 PM PolicyBackend openPolicyBackend
```

구성되지 않은 정책 리포지토리에 연결

Configuration Agent에서 정책 리포지토리의 정책 데이터를 찾으려면 먼저 정책 리포지토리를 올바르게 구성해야 합니다. 구성되지 않았거나 잘못 구성된 정책 리포지토리를 지정하면 데스크탑 클라이언트 응용 프로그램을 시작할 때 Configuration Agent 로그에 다음과 유사한 오류 메시지가 기록됩니다. 이 문제를 해결하려면 해당 절을 참조하십시오.

```
FINER: New client added
Nov 18, 2005 2:36:55 PM com.sun.apoc.daemon.misc.APOCLogger finer
FINER: CreateSession transaction started
Nov 18, 2005 2:36:55 PM com.sun.apoc.daemon.misc.APOCLogger finer
FINER: Creating new client session
Nov 18, 2005 2:36:55 PM com.sun.apoc.daemon.misc.APOCLogger finest
FINEST: Authenticating user geoffh
Nov 18, 2005 2:36:55 PM com.sun.apoc.daemon.misc.APOCLogger finest
FINEST: Authentication successful
Nov 18, 2005 2:36:55 PM PolicyBackend openPolicyBackend
FINER: THROW
com.sun.apoc.daemon.misc.APOCException: com.sun.apoc.daemon.misc.APOCException:
com.sun.apoc.spi.environment.RemoteEnvironmentException: Error on reading the
configuration data on LDAP server ldap://sobuild:389.
    at com.sun.apoc.daemon.apocd.PolicyBackend.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.HostPolicyBackend.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.PolicyBackendFactory.openPolicyBackend(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache$DataSource.openPolicyBackend(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache$DataSource.open(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache.createDataSources(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.CacheFactory.createNewCache(Unknown Source)
    at com.sun.apoc.daemon.apocd.CacheFactory.openCache(Unknown Source)
    at com.sun.apoc.daemon.apocd.Session.<init>(Unknown Source)
    at com.sun.apoc.daemon.transaction.CreateSessionTransaction.executeTransaction
(Unknown Source)
    at com.sun.apoc.daemon.transaction.Transaction.execute(Unknown Source)
    at com.sun.apoc.daemon.apocd.ClientEventHandler.handleEvent(Unknown Source)
    at com.sun.apoc.daemon.apocd.EventWorkerThread.run(Unknown Source)
```

Configuration Agent 로그에 "maximum number of client connections" 관련 메시지가 표시됩니다. 이 메시지의 의미는 무엇입니까?

Configuration Agent에서 활성화되는 각 데스크탑 클라이언트 응용 프로그램(gconfd, Mozilla, StarSuite)을 실행하는 경우 Configuration Agent에 연결합니다. 에이전트 구성에 해당 연결 수 제한이 지정되어 있습니다. 기본 연결 제한은 50입니다. 한 시스템을 여러 사용자가 사용하는 경우에는 Configuration Agent 마법사(`/usr/bin/apoc-config`)에서 "최대 클라이언트 연결 수" 설정을 변경하여 이 제한을 늘려야 할 수도 있습니다. Configuration Agent가 최대 연결 수에 도달하면 Configuration Agent 로그에 다음과 유사한 오류 메시지가 기록됩니다.

```
Nov 18, 2005 3:20:55 PM com.sun.apoc.daemon.misc.APOCLogger warning
WARNING: The maximum number of client connections ( 50 ) has been reached.
No new client connections can be established at this time.
```

Desktop Manager를 사용하여 일부 정책을 수정했지만 수정된 내용이 내 클라이언트 시스템에 표시되지 않습니다.

Desktop Manager에서 만든 정책 데이터는 상대적으로 정적이기 때문에 자주 변경할 수 없다는 가정하에 Configuration Agent를 설계했습니다. 이 가정에 따라 에이전트는 정책 리포지토리를 자주 참조하여 수정 사항이 있는지를 확인합니다. 기본적으로 에이전트는 실행 중인 모든 데스크탑 응용 프로그램에서 한 시간에 한 번씩 리포지토리를 확인합니다. 따라서, Desktop Manager를 사용하여 변경한 경우 실행 중인 데스크탑 응용 프로그램에서 변경 사항을 알려면 최대 1시간까지 기다려야 합니다. 원하는 경우 Agent Configuration 마법사(`/usr/bin/apoc-config`)를 사용하여 "일반 감지 간격" 값을 늘려 리포지토리 확인 빈도를 늘릴 수도 있습니다. 또는 수퍼유저로 `/usr/lib/apoc/apocd change-detect` 명령을 실행하여 Configuration Agent에서 연결된 모든 응용 프로그램에 대한 정책 데이터를 강제로 새로 고칠 수 있습니다.

◆ ◆ ◆ 4 장 4

Java Web Console

Java Web Console은 Sun Microsystems의 공통 웹 기반 시스템 관리 솔루션으로 설계되었습니다. 사용자가 이 콘솔을 통해 액세스하는 시스템 관리 응용 프로그램은 모두 동일한 사용자 인터페이스를 제공합니다.

다양한 이유로 해서 콘솔은 웹 모델을 기반으로 설계되었지만 주된 이유는 시스템 관리자가 웹 브라우저를 사용하여 시스템 관리 응용 프로그램에 액세스할 수 있도록 하기 위한 것입니다.

Java Web Console에서 제공하는 기능은 다음과 같습니다.

- 공통 인증 및 권한 부여
- 공통 로깅
- 동일한 HTTPS 기반 포트를 통해 모든 시스템 관리 응용 프로그램에 액세스할 수 있는 단일 진입점
- 공통된 모양 및 색감

콘솔의 이점은 관리자가 한 번만 로그인하면 콘솔 내에서 모든 응용 프로그램을 사용할 수 있다는 것입니다.

설치

시스템 요구 사항

Java Web Console은 다양한 클라이언트 및 서버 운영 체제와 여러 브라우저를 지원합니다.

클라이언트

- Solaris 10에서 실행되는 Netscape™ 6.2x 및 7.x
- Windows 98, 98 SE, ME, 2000 및 XP에서 실행되는 Netscape 6.2x 및 7.x
- Windows 98, 98 SE, ME, 2000 및 XP에서 실행되는 Internet Explorer 5.5x 및 6.x

- Solaris에서 실행되는 Mozilla 1.4x
- Solaris에서 실행되는 Firefox 1.0

서버

- Solaris 10
- Red Hat Application Server 2.1, 3.0
- SuSE Linux 8.0 이상
- J2SE Version 1.4.1_03 이상

  서버에서 J2SE 1.4.1 이전 버전이 검색되면 설치 프로그램에서 Java Desktop System Management Tools CD에 있는 J2SE 버전을 사용하여 설치를 업그레이드하라는 메시지를 표시합니다.
- Tomcat: 4.0.3 이상

  Tomcat은 Java Desktop System Management Tools CD에 포함되어 있습니다.

Java Web Console 설치

Java Web Console 2.2.4가 Solaris 10 운영 체제에 포함되어 있지만, Desktop Manager에는 2.2.5 버전이 필요합니다. 2.2.5 버전은 `server/console` 디렉토리의 Desktop Manager 아카이브에 제공됩니다. 해당 디렉토리에서 `./setup`을 실행하여 2.2.5 버전을 설치할 수 있습니다.

Java Web Console 3.0을 설치한 경우 3.0 버전을 제거한 다음 위에서 설명한 것과 같이 `server/console` 디렉토리에서 Java Web Console 2.2.5를 설치해야 합니다.

콘솔 실행

일반적으로 새 응용 프로그램을 등록하려면 Java Web Console 서버를 중지했다가 다시 시작하면 됩니다.

주의 – Java Web Console을 처음 시작하기 전에 Desktop Manager 설치를 완료해야 합니다. Java Web Console은 콘솔에 하나 이상의 응용 프로그램을 배포할 때까지는 성공적으로 실행되지 않습니다.

- Java Web Console을 시작하려면 `smcwebserver start`를 입력합니다.
- Java Web Console을 중지하려면 `smcwebserver stop`을 입력합니다.
- Java Web Console을 다시 시작하려면 `smcwebserver restart`를 입력합니다.
- Java Web Console에 액세스하려면 브라우저에 다음 URL을 입력합니다:
  `https://`*<hostname>*`.`*<domainname>*`:6789`

Java Web Console은 기본적으로 Unix 기반 인증과 역할별 접근 제어(RBAC)를 지원합니다. 그러나 LDAP 인증과 같은 다른 인증 체계를 구성할 수도 있습니다.

주 – 기본 세션 시간 제한은 15분입니다. `smreg` 명령을 사용하여 시간 제한을 구성할 수 있습니다. 예를 들어, 시간 제한을 5분으로 설정하려면 `smreg add -p -c session.timeout.value=5`를 입력합니다.

Java Web Console의 명령에 대한 자세한 내용은 smcwebserver 및 smreg 설명서 페이지를 참조하십시오.

Java Web Console 제거

주의 – Solaris를 사용하는 경우 Java Web Console이 운영 체제의 일부이므로 제거할 수 없습니다.

Java Web Console 문제 해결

Java Web Console을 설치할 수 없는 경우

증상: 설치가 끝나면 등록된 응용 프로그램이 없기 때문에 Java Web Console을 시작할 수 없다는 메시지가 표시됩니다.

예상 원인: Desktop Manager 모듈이 설치되면 Java Web Console을 시작합니다.

연결이 거부된 경우

증상: 적절한 URL(예: `https://`*<your.server>*`:6789`)을 열려고 했지만 연결이 거부되었습니다.

예상 원인: Java Web Console이 서버에서 실행되고 있지 않습니다.

로그인할 수 없는 경우

주 – 기본적으로 LDAP 로그인 모듈은 설치되지 않습니다. 따라서, 로그인 자격 증명이 LDAP 서버에 저장된 자격 증명과 비교되지 않고 일반 시스템 로그온 자격 증명만 필요합니다. 이 문제 해결 절은 LDAP 로그인 모듈을 수동으로 설치한 경우에만 적용됩니다.

증상: Web Console의 로그인 페이지에 연결했지만, 사용자/암호 조합이 거부됩니다.

예상 원인:

- LDAP 서버가 실행되고 있지 않습니다.
- Web Console LDAP 인증 모듈이 잘못 구성되었습니다.
- 사용자가 LDAP 서버에 없습니다.
- LDAP 서버에 있는 사용자의 암호가 다릅니다.

Desktop Manager 링크가 없는 경우

증상: Web Console에 로그인했지만 응용 프로그램 목록 페이지에 Desktop Manager가 없습니다.

예상 원인:

- Desktop Manager 모듈이 설치되지 않았습니다.

Null 포인터 예외, Tomcat/Java 오류 또는 비어 있음

증상: Desktop Manager를 열었지만 값이 표시되지 않고 빈 페이지가 열리거나 오류가 발생합니다.

예상 원인: `NoClassDefFoundError:sun/tools/javac/Main` 오류가 발생하는 경우 Java Web Console이 잘못된 Java 설치를 사용하고 있는 것입니다.

기타 문제

웹 서버가 제대로 실행되지 않는 경우 로그 파일의 정보를 참조할 수 있습니다. 로그 파일은 `/var/log/webconsole/`에 있습니다. `smreg`를 사용하여 로그 세부 수준을 늘릴 수 있습니다.

```
smreg add -p debug.trace.level=3
smreg add -p debug.trace.options=tmp
```

다음 명령을 사용하여 원래의 설정을 복원할 수 있습니다.

```
smreg add -p debug.trace.level=0
smreg add -p debug.trace.options=m
```

전체 구성 데이터베이스 덤프는 다음 명령을 통해 실행됩니다.

```
smreg list
```

Desktop Manager를 호스트하는 웹 서버가 제대로 종료되지 않아서 포트가 여전히 사용 중일 수 있습니다. 그러면 새로 시작된 웹 서버가 전혀 시작되지 않습니다. `smcwebserver start/restart` 명령이 오류 메시지를 표시하거나, `smcwebserver stop` 명령을 실행한 후에도 Desktop Manager에 액세스 가능하거나, 새로 시작된 서버가 이전 인스턴스처럼 동작하는

경우 포트 6789가 사용 중인지 확인하거나(`netstat -a | grep 6789`) 웹 서버가 여전히 실행 중인지 확인합니다(`ps -ef | grep java`). 둘 중 하나의 경우 해당 프로세스를 종료하고 포트 6789가 더 이상 사용되지 않도록 해야 합니다.

◆ ◆ ◆ 부 록 A

구성 매개 변수

이 매개 변수는 다음과 같은 Desktop Manager 구성 요소에 대해 정의될 수 있습니다.

- Desktop Manager(`/etc/opt/SUNWapcmg/`에 있는 구성 리포지토리를 정의하는 파일에서)
- Configuration Agent(`/etc/apoc/policymgr.properties` 파일에서)
- Desktop Manager CLI(`$HOME/pgtool.properties` 파일에서, CLI는 순수 LDAP 리포지토리만 지원하는 제한이 있음)

매개 변수를 앞에 붙여 적용되는 리포지토리 공급자를 나타낼 수 있습니다. 각 공급자에 대해 접두어가 있는 매개 변수가 가장 먼저 고려됩니다. 해당 매개 변수가 정의되어 있지 않은 경우 접두어가 없는 매개 변수가 사용됩니다

표 A–1 접두어

접두어 값	리포지토리 공급자
ORGANIZATION_	조직 트리
DOMAIN_	도메인 트리
PROFILE_	프로필
ASSIGNMENT_	할당
LDAP_META_CONF_	매핑 데이터(LDAP 리포지토리의 경우)

표 A–2 매개 변수

이름	설명	가능한 값	기본값

PROVIDER_URL	리포지토리에 대한 연결을 지정하는 URL. 첫 번째 리포지토리에 대한 연결이 실패할 경우 URL 목록을 사용하여 폴백 리포지토리를 지정할 수 있습니다.	하나 이상의 공백으로 구분된 URL 목록이며 각 URL의 형식은 다음 중 하나입니다.
		`ldap://<host>:<port>/<baseDN>`
		`ldaps://<host>:<port>/<baseDN>`
		`file://<filepath>`
		`http://<host>:<port>/<filepath>`
		`https://<host>:<port>/<filepath>`
			없음, 필수 매개 변수

SECURITY_PRINCIPAL	리포지토리 연결에 대한 사용자 이름	리포지토리에 대한 읽기 및 검색 액세스 권한이 있는 사용자의 사용자 이름 또는 익명 연결에 대한 값 없음	없음, 익명 연결
SECURITY_CREDENTIALS	SECURITY_PRINCIPAL에 정의된 사용자의 암호	일반 텍스트 또는 스크램블된 암호	없음
SECURITY_CREDENTIALS_ENCODING	SECURITY_PRINCIPAL에 정의된 암호가 스크램블되는지 여부를 나타냅니다. 경고: 암호 스크램블은 암호에 대한 마스크일 뿐이며, 보안 암호화 유형을 구성하지는 않습니다.	암호가 스크램블되는 경우 “스크램블”(구성 데이터를 생성할 때 마법사에서 자동으로 수행됨). 암호가 일반 텍스트로 표시되는 경우 “없음”. 암호를 편집하려면 이 값을 사용하십시오.	“없음”
MAX_SEARCH_RESULT	리포지토리 검색에 지정된 최대 결과 수. 주: 접두어 구성표는 이 매개 변수에 적용되지 않습니다.	양수, 0은 제한 없음을 의미합니다.	100

다음 매개 변수는 LDAP 리포지토리에만 적용됩니다.

표 A–3 LDAP별 매개 변수

이름	설명	가능한 값	기본값
AuthDn	사용자가 SECURITY_PRINCIPAL에 정의된 사용자를 검색하기 위해 LDAP 리포지토리에 처음 액세스할 때 사용할 사용자의 정규화된 DN입니다.	리포지토리에 대한 읽기 및 검색 액세스 권한이 있는 사용자의 사용자 이름 또는 익명 연결에 대한 값 없음	없음, 익명 액세스
암호	AuthDN의 암호	일반 텍스트 또는 스크램블된 암호	없음
Password_ENCODING	암호에 정의된 암호가 스크램블되는지 여부를 나타냅니다. 경고: 암호 스크램블은 암호에 대한 마스크일 뿐이며, 보안 암호화 유형을 구성하지는 않습니다.	암호가 스크램블되는 경우 “스크램블”(구성 데이터를 생성할 때 마법사에서 자동으로 수행됨). 암호가 일반 텍스트로 표시되는 경우 “없음”, 암호를 편집하려면 이 값을 사용하십시오.	“없음”
연결 시간 제한	연결 생성 시간 제한(초)	양수, 시간 제한이 없는 경우 0	1

예 A–1 하이브리드 백엔드의 예

하이브리드 백엔드의 예이며, 여기서 호스트 및 사용자에 대한 정보는 기존 LDAP 리포지토리에서 가져오고, 프로필 및 프로필 지정은 파일 시스템에 저장됩니다.

```
#Organization, Domain, MetaConf
PROVIDER_URL = ldap://server1.sun.com:389/o=apoc ldap://server2.sun.com:389/o=apoc
SECURITY_PRINCIPAL = jmonroe
SECURITY_CREDENTIALS = JmonroE
SECURITY_CREDENTIALS_ENCODING = none
AuthDn = cn=reader,ou=special users,o=apoc
Password = lakjflajf
Password_ENCODING = scramble
ConnectTimeout = 5

#Profile
PROFILE_PROVIDER_URL = file:///path/to/repository

#Assignment
ASSIGNMENT_PROVIDER_URL = file:///path/to/repository
```

◆ ◆ ◆ 부 록 B

OpenLDAP 및 Active Directory를 Desktop Manager와 함께 사용

OpenLDAP 서버를 Desktop Manager와 함께 사용

OpenLDAP 서버를 Desktop Manager 데이터를 위한 리포지토리로 사용하려면, 구성 데이터를 저장하는 데 사용되는 개체 클래스와 속성을 지원하도록 서버의 스키마를 확장해야 합니다. 사용자 정의 스키마 파일 `apoc.schema`는 `/usr/share/webconsole/apoc/deploy` 디렉토리에 있습니다.

OpenLDAP 구성 디렉토리(`/etc/openldap`)의 `schema` 하위 디렉토리에 이 파일을 복사하고 그 디렉토리에 있는 `slapd.conf` 파일에 포함시켜 OpenLDAP 스키마에 추가해야 합니다. 이렇게 하려면 해당 파일에 있는 스키마 순서의 끝에 `include /etc/openldap/schema/apoc.schema`라는 행을 삽입하면 됩니다. OpenLDAP 서버의 스키마 확장에 대한 자세한 내용은 서버의 설명서를 참조하십시오.

OpenLDAP 서버 스키마를 확장한 후 Desktop Manager의 구성 리포지토리 추가 마법사를 사용하여 나머지 구성을 완료할 수 있습니다.

주 – Desktop Manager Agent에서는 데이터가 필요한 사용자의 DN을 제공하되 암호는 제공하지 않음으로써 익명으로 OpenLDAP 서버에 연결합니다. 일부 OpenLDAP 서버 릴리스에서는 이 익명 인증 모드를 사용하지 않도록 기본적으로 설정되어 있습니다. 이 경우에는 OpenLDAP 구성 디렉토리(`/etc/openldap`)에 있는 `slapd.conf` 파일에 정의된 공통 서버 매개 변수에 `allow bind_anon_cred` 행을 추가하여 사용할 수 있도록 설정해야 합니다. 해당 매핑에 대한 자세한 내용은 서버의 설명서를 참조하십시오.

Active Directory 서버를 Desktop Manager와 함께 사용

Active Directory 서버를 Desktop Manager 데이터를 위한 리포지토리로 사용하려면, 구성 데이터를 저장하는 데 사용되는 개체 클래스와 속성을 지원하도록 서버의 스키마를 확장해야 합니다. 스키마 확장 파일 `apoc-ad.ldf`는 `/usr/share/webconsole/apoc/deploy` 디렉토리에 있습니다.

다음 단계를 따라 `apoc-ad.ldf` 파일을 Active Directory 스키마로 가져와야 합니다.

1. 스키마 확장을 활성화합니다. 해당 작업을 수행하는 방법에 대한 자세한 내용은 Active Directory 설명서를 참조하십시오.
2. 명령 프롬프트에서 다음을 실행합니다. **`ldifde -i -c "DC=Sun,DC=COM" <`*BaseDN*`> -f apoc-ad-registry.ldf`** .

주 – <*BaseDN*>을 Active Directory 기본 DN으로 바꿉니다.

Active Directory 서버 스키마를 확장한 후 Desktop Manager의 구성 리포지토리 추가 마법사를 사용하여 나머지 구성을 완료할 수 있습니다.

구성 리포지토리 추가 마법사에 LDAP 자격 증명을 묻는 메시지가 표시되면 트리에 대한 읽기 권한이 있는 사용자의 전체 DN과 암호를 제공합니다. 이 사용자는 다른 목적으로는 Active Directory를 사용할 수 없는 사용자일 수 있습니다. 해당 사용자를 설정하는 방법에 대한 자세한 내용은 Active Directory 설명서를 참조하십시오. 또한 Desktop Manager를 실행하는 시스템에서 Active Directory의 도메인 이름을 알고 있어야 합니다. 이렇게 하려면 Active Directory 서버의 IP 주소를 도메인 이름과 함께 그 시스템의 `/etc/hosts` 파일로 매핑하는 행을 추가하면 됩니다.

데스크탑 호스트에서 구성 데이터를 검색하려면 해당 호스트에서도 Active Directory의 도메인 이름을 알고 있어야 합니다. 데스크탑 사용자 인증은 익명과 GSSAPI의 두 가지 방법으로수행할 수 있습니다.

- 익명 연결을 사용하여 인증하려면 Active Directory 서버가 모든 사람에게 읽기 권한을 부여하도록 구성되어야 합니다. 해당 작업을 수행하는 방법에 대한 자세한 내용은 Active Directory 설명서를 참조하십시오.
- GSSAPI를 사용하여 인증하려면 사용자가 Active Directory에 대해 인증하고 사용자 자격 증명이 시스템에서 사용 가능해야 합니다. 그러기 위해서는 시스템에 커버로스 인증을 구성합니다. 그러면 로그인할 때 해당 자격 증명이 생성됩니다. 이 방법에 대한 자세한 내용은 해당 시스템 관리 설명서를 참조하십시오.

◆ ◆ ◆ 부록 C

조직 매핑

조직 매핑

LDAP 항목과 Desktop Manager 요소 간의 매핑을 정의하려면 `Organization` 매핑 파일을 편집해야 합니다. LDAP 리포지토리의 레이아웃에 맞는 값을 각 키에 입력해야 합니다.

사용자 요소는 모든 요소가 사용하는 개체 클래스와 전체 리포지토리에서 고유한 값을 갖는 속성으로 식별됩니다. 관리 응용 프로그램에서의 사용자 표시에 영향을 주는 표시 이름 형식을 제공할 수 있으며, 조직 내의 사용자 항목이 컨테이너 항목을 사용할 경우 컨테이너 항목을 선택적으로 정의할 수도 있습니다. 키 이름과 해당 기본값은 다음과 같습니다.

```
# Object class that all user entries use
User/ObjectClass=inetorgperson
# Attribute whose value in user entries is unique within the repository
User/UniqueIdAttribute=uid
# Optional container in organization entries of the user entries,
# remove line if not used
User/Container=ou=People
# Display name format within the management application
User/DisplayNameFormat=sn, givenname
```

역할 요소는 사용할 수 있는 개체 클래스 목록과 해당 이름 지정 속성으로 식별됩니다. 이 목록은 `<item1>,<item2>,...,<itemN>` 형식을 사용하며 정렬되어야 합니다. 즉, 동일한 수의 항목이 목록에 있어야 하고 n번째 개체 클래스에는 n번째 이름 지정 속성을 사용해야 합니다. 두 키는 역할과 사용자 간의 관계는 물론 역할과 호스트 간의 관계도 정의합니다. *VirtualMemberAttribute* 키는 사용자 또는 호스트 항목에서 쿼리할 수 있는 값을 가진 속성을 지정해야 하며 항목이 속해 있는 역할의 전체 DN을 포함해야 합니다. *MemberAttribute* 키는 검색 필터에 사용할 사용자 또는 호스트 항목의 속성을 지정해야 합니다. 사용자 또는 호스트가 속해 있는 역할의 전체 DN을 포함해야 합니다. *VirtualMemberAttribute* 키에는 서비스 클래스 가상 속성을 사용할 수 있지만 *MemberAttribute* 키는 필터에 사용할 수 있는 물리적 속성이어야 합니다. 키 이름과 해당 기본값은 다음과 같습니다.

```
# List of object classes for roles
Role/ObjectClass=nsRoleDefinition
```

```
# Aligned list of corresponding naming attributes
Role/NamingAttribute=cn
# Physical attribute (usable in a filter) containing the DNs
# of the roles of a user/host
Role/MemberAttribute=nsRoleDN
# Attribute whose query on a user or host return the DNs of the
# roles it belongs to
Role/VirtualMemberAttribute=nsRole
```

조직 요소는 정렬된 개체 클래스 목록과 해당 이름 지정 속성 목록을 사용하여 역할과 유사한 방식으로 식별됩니다. 키 이름과 해당 기본값은 다음과 같습니다.

```
# List of object classes for organizations
Organization/ObjectClass=organization
# Aligned list of corresponding naming attributes
Organization/NamingAttribute=o
```

도메인 요소는 조직 요소와 유사한 방식으로 식별됩니다. 키 이름과 해당 기본값은 다음과 같습니다.

```
# List of object classes for domains
Domain/ObjectClass=ipNetwork
# Aligned list of corresponding naming attributes
Domain/NamingAttribute=cn
```

호스트 요소는 사용자 요소와 유사한 방식으로 식별됩니다. 키 이름과 해당 기본값은 다음과 같습니다.

```
# Object class that all host entries use
Host/ObjectClass=ipHost
# Attribute whose value in host entries is unique within the repository
Host/UniqueIdAttribute=cn
# Optional container in domain entries of the host entries,
# remove line if not used
Host/Container=ou=Hosts
```