# Sun Desktop Manager 1.0 安裝指南

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

文件號碼：819–6096–10

060406@14774

# 目錄

# 前言

本文件提供部署 Sun™ Desktop Manager 1.0 所需的安裝與配置步驟之說明。

## 簡介

Sun Desktop Manager 以提供桌面主機的集中配置爲目的。可以將設定指定給組織或網域結構的各種元素，使管理員可以有效地管理使用者群組或主機群組。

## 本書的組織方式

第 1 章提供 Sun Desktop Manager 的簡介。

第 2 章討論 Sun Desktop Manager 伺服器端的安裝。

第 3 章提供安裝 Java Desktop System Configuration Agent 的資訊。

第 4 章提供有關 Java Web Console 的安裝資訊。

附錄 A 包含配置參數資訊。

附錄 B 討論與 Desktop Manager 搭配使用 OpenLDAP 及 Active Directory。

附錄 C 提供組織對映的相關資訊。

## 相關書籍

- 「Sun Desktop Manager 1.0 管理指南」
- 「Sun Desktop Manager 1.0 Developer Guide」

# 文件、支援和訓練

| Sun 提供的服務 | URL | 說明 |
| --- | --- | --- |
| 文件 | http://www.sun.com/documentation/ | 下載 PDF 與 HTML 文件以及訂購書面列印的文件 |
| 支援與培訓 | http://sunsolve.sun.com | 取得技術支援、下載修補程式與 Sun 培訓課程的資訊 |

◆ ◆ ◆ 第 1 章

# 1 簡介與概念

本文件提供部署 Sun™ Desktop Manager 1.0 所需的安裝與配置步驟之說明。如需 Sun Desktop Manager 更詳盡的簡介，請參閱「Sun Desktop Manager 1.0 管理指南」。

## Sun Desktop Manager 簡介

Sun Desktop Manager 提供桌面主機的集中配置。可以將設定指定給組織或網域結構的各種元素，使管理員可以有效地管理使用者群組或主機群組。

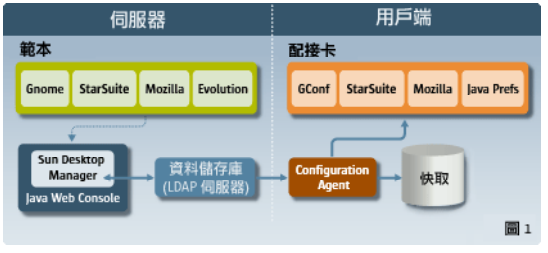


圖 1–1 Desktop Manager 架構

Desktop Manager 包括下列主要元件：

- 配置儲存庫
- 管理工具
- Desktop Manager 範本
- Configuration Agent
- 配置介面

配置資料集中儲存在配置儲存庫中。配置資料使用管理工具進行管理 (created/deleted/modified/assigned/unassigned)，其由 Web 型 Desktop Manager 圖形化使用者介面以及指令行介面所組成。Web 型管理工具會使用範本，於 Web 瀏覽器中描繪配置資料。

Configuration Agent 會由配置儲存庫中代表使用者應用程式擷取配置資料。代理程式會快取由中央配置儲存庫所擷取的資訊。

管理工具會完全與代理程式脫離，這表示管理工具僅會於配置儲存庫上運作。

使用者應用程式可使用配置介面，經由 Configuration Agent 查詢配置資料。

本產品爲下列配置系統直接支援設定的擷取及應用：

- GConf。Gnome 配置架構
- StarSuite Registry
- Mozilla Preferences
- Java Preferences

◆ ◆ ◆ 第 2 章

# 管理應用程式安裝

本章將提供安裝 Sun Desktop Manager 伺服器端元件的指示。

## Sun Desktop Manager

Desktop Manager 會提供可以在 Java Web Console 上執行的 Web 型管理工具。管理員可利用此使用者介面遍歷組織的階層，並定義桌面應用程式的各項策略。可以爲階層結構中的每個項目 (如組織、角色、使用者、網域和主機) 定義這些策略。Desktop Manager 採用多種配置範本來顯示各桌面應用程式 (如 Gnome、Mozilla、StarSuite 和 Evolution) 特有的設定。

### ▼ 安裝

**開始之前** Desktop Manager 需要安裝 Java Web Console 版本 2.2.5 或更高版本。請確定系統上已安裝了有效的版本。如需判別所安裝的版本有效與否，請以超級使用者身份 (root) 執行下列作業：

```
# smcwebserver status
```

---

**備註 –** Java Web Console 2.2.4 隨附於 Solaris™ 10 作業系統，但 Desktop Manager 需要版本 2.2.5 或更高的版本。Desktop Manager 歸檔的 server/console 目錄中含有版本 2.2.5 的副本。執行目錄中的 `./setup` 即可進行安裝。

---

系統上若未安裝 Java Web Console，或所安裝的版本對 Desktop Manager 而言無效，請參閱第 4 章的說明，先安裝或更新 Java Web Console。然後再回到本章繼續安裝 Desktop Manager。

1. **下載 Desktop Manager 壓縮歸檔，並將其內容解壓縮至暫存的目錄中。**

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

2. **請以超級使用者身份 (root) 及下列方式執行設定程序檔：**

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

3. **檢視安裝程序檔的輸出中是否有出現任何錯誤。**

   安裝若是成功，則安裝程序檔便會自動重新啓動 Java Web Console，讓您能夠使用 Web 瀏覽器存取 Desktop Manager。

## ▼ 操作

1. **在瀏覽器中鍵入以下 URL：**

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

2. **在登入畫面中，鍵入現有 Unix 使用者的使用者名稱和密碼。**

   隨即會開啓 Java Web Console。

3. **在主控台應用程式啓動頁面中，按一下 Desktop Manager 連結。**

   - **若要略過主控台應用程式啓動頁面，而直接前往 Desktop Manager，請在瀏覽器中輸入下列 URL：**

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

## ▼ 移除 Desktop Manager

◗ **若要從 Java Web Console 移除 Desktop Manager，請將暫存目錄變更至安裝此產品時所建立的該目錄，並以超級使用者身份 (root) 執行**

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

# 遷移問題

Desktop Manager 可與舊版的 Java Desktop System Configuration Manager (發行版本 1.0 和 1.1) 相容。但您仍須注意其間的一些差異。

舊版的 Configuration Manager 會將所有設定檔資料集中儲存在某特定的 LDAP 伺服器上。LDAP 伺服器的配置工作則會隨著整體的 Configuration Manager 安裝程序一併進行。其中亦包括對 LDAP 伺服器封裝認證之 LDAP 登入模組的配置。

現在 Desktop Manager 的各項必要配置步驟均會由精靈引導進行，再也不需要在安裝時執行任何配置。此外 Desktop Manager 亦可支援多種配置儲存庫。因此，您可以管理儲存在數個不同 LDAP 伺服器、檔案型儲存庫中的策略資料。您已不再需要配置特有的 LDAP 登入模組。

不同版本間的 LDAP 模式則無任何變更。您若已配置了舊版 Configuration Manager 版本所需的 LDAP 伺服器，則在切換至 Desktop Manager 時亦無需進行任何變更。換言之，您無需更新用戶端 (Java Desktop System Configuration Manager1.1. 代理程式) 或 LDAP 端，即可直接使用 Desktop Manager。

---

**備註 –** 在安裝 Desktop Manager 之前，應先移除系統上舊版的 Configuration Manager 或 Desktop Manager 安裝。若要移除舊版的安裝，請以超級使用者身份執行下列作業：

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

---

安裝完 Desktop Manager 之後，即可建立配置儲存庫，將其指向現有的 LDAP 伺服器：

### ▼ 建立配置儲存庫

1. **在瀏覽器中鍵入下列 URL**

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

2. **在登入畫面中，鍵入現有 Unix 使用者的使用者名稱和密碼。**

   隨即會開啓 Java Web Console。

3. **在主控台應用程式啓動頁面中，按一下 Sun Desktop Manager 1.0 連結。**

4. **按一下 [新增] 按鈕啓動配置儲存庫精靈。**

   此精靈會引導您完成配置 LDAP 配置儲存庫的必要步驟。

---

**注意 –** 此精靈可以自動將現有的策略資料遷移至新的 2.0 格式。遷移作業爲選擇性作業，其主要用途在改善新版 Sun Desktop Manager 1.0 代理程式的效能。若環境中仍須支援 Java Desktop System Configuration Manager 1.1 代理程式，請勿執行此遷移作業。

---

# Desktop Manager 疑難排解

## 無法安裝

徵兆：當 Java Web Console 安裝即將結束時，會出現訊息指出其因爲仍有應用程式尚未註冊而無法啓動。

可能原因：未安裝任何應用程式 (包括 Desktop Manager 在內)。

解決方案：安裝 Desktop Manager，再啓動 Java Web Console。

## 連線遭拒

徵兆：嘗試開啓適當的 URL (例如 `http://<hostname>.<domainname>:6789`)，卻收到連線被拒的訊息。

可能原因：伺服器上未執行 Java Web Console。

解決方案：若要啓動 Java Web Console，請以超級使用者身份執行下列指令：

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

## 無法登入

徵兆：在 Java Web Console 登入頁面上提供的使用者/密碼組合被拒。

可能原因：相應的 UNIX 使用者帳號不存在。

解決方案：請確定是否已在系統上配置了相應的 UNIX 使用者名稱和密碼。如有必要，請建立本機 UNIX 使用者帳號加以測試。

## 無 Desktop Manager 連結

徵兆：Java Web Console 應用程式的清單頁面未顯示 Sun Desktop Manager 連結。

可能原因：未安裝 Desktop Manager 模組。

解決方案：若要檢查 Java Web Console 中是否安裝了 Desktop Manager，請以超級使用者身份執行下列指令：

```
# smreg list -a
```

清單中若未包含 `com.sun.apoc.manager_<` *version*> 應用程式，便須重新安裝 Desktop Manager。

## 空指標異常，Tomcat/Java 錯誤或空白頁面

徵兆：啓動了 Desktop Manager，但只顯示空白頁面或一些錯誤訊息。

可能原因：錯誤中如有提到 `NoClassDefFoundError:sun/tools/javac/Main`，即表示 Java Web Console 使用了錯誤的 Java 版本。

解決方案：執行 `# smreg list -p`，並查看 `java.home` 特性，以檢查目前的 Java Web Console Java 環境。此特性必須指向有效的 Java 主目錄，且該主目錄必須爲 JDK。此值設定若是錯誤，便須執行下列指令：

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

**備註 –** *<JAVA_HOME>* 必須指向有效的安裝，例如內含 `bin` 子目錄的 `javac`。

接著請以下列指令重新啓動 Java Web Console：

```
# smcwebserver restart
```

## 無法連線至 SSL LDAP 伺服器

徵兆：在儲存庫建立精靈中提供 LDAP 伺服器的詳細資訊之後 (包括核取 [使用 SSL] 方塊) 按 [下一步]，會出現訊息方塊表示無法與伺服器取得聯繫。

可能原因：所提供的連接埠號不正確；未將 LDAP 伺服器配置成偵聽該連接埠上使用 SSL 的連線；或 Java Web Console 金鑰庫中缺乏正確的憑證。

解決方案：首先請確定是否已將 LDAP 伺服器配置成偵聽精靈所指定之連接埠上的 SSL 連線請求。若其設定正確，請檢查 Java Web Console 金鑰庫中確有「憑證授權單位」或 LDAP 伺服器憑證，且其位於 `/etc/opt/webconsole/keystore` 中。您可以使用 `keytool -import -file <certificate file> -keystore /etc/opt/webconsole/keystore` 指令增加憑證。該金鑰庫的預設密碼爲 **`changeit`**。您必須使用 `smcwebserver restart` 指令重新啓動 Java Web Console，此變更對於 Desktop Manager 才會生效。

## 無法寫入目錄

徵兆：建立檔案型或混合型後端時，發生「無法寫入目錄！」的錯誤。

可能原因：noaccess 使用者不具備正確的權限。

解決方案：爲 noaccess 使用者指定寫入權限。

◆ ◆ ◆ 第 3 章

# 3 用戶端元件

若要存取 Desktop Manager 的配置資料，桌面用戶端必須有 Java Desktop System Configuration Agent。Configuration Agent 可與遠端配置資料儲存庫及介面進行通訊，也可將資料整合至特定的配置系統中。目前支援的配置系統爲 GConf、Java Preferences、Mozilla Preferences 和 StarSuite Registry。

Solaris 10 作業系統會隨附 Configuration Agent。但 Desktop Manager 必須使用此工具的較新版本。新版本工具會在安裝 Desktop Manager 用戶端元件與相關修補程式時一併安裝。

安裝 Desktop Manager 用戶端元件：

1. 下載 Desktop Manager 壓縮歸檔，並將其內容解壓縮至暫存的目錄中。

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

2. 安裝建議的修補程式。

   這些修補程式位於 `SunDesktopMgr-1.0/<platform>/client/Patches` 目錄中。請遵循修補程式所附的安裝說明執行作業。

3. 以超級使用者身份 (root) 及下列方式執行設定程序檔：

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

## Configuration Agent

Configuration Agent 分屬於下表所列之各種套裝軟體的一部分。

| Solaris 套裝軟體名稱 | 說明 |
|---|---|
| SUNWapbas | 配置共用程式庫 |

| Solaris 套裝軟體名稱 | 說明 |
|---|---|
| SUNWapmsc | Configuration Agent 其他檔案 |
| SUNWapoc | Configuration Agent |
| SUNWapdc | Configuration Agent 精靈 |

安裝這些套裝軟體時，該 API 所需的檔案都會安裝。您可以手動安裝這些套裝軟體或經由 Java Desktop System 進行安裝。安裝後，您必須在系統中配置並啓用 Configuration Agent。

**備註 –** Configuration Agent 套裝軟體會在安裝 Java Desktop System 時，一併安裝在 Solaris 系統上；但 Desktop Manager 會在安裝期間修補這些檔案，以提供適當的功能層級。

若要存取遠端配置資料，Configuration Agent 需要最基本的啓動程式資訊，例如 LDAP 伺服器的主機名稱和連接埠。本資訊保存在一組特性檔案中，如 `policymgr.properties` 、`apocd.properties`、`os.properties` 等等。這些檔案以本機方式儲存在 `/etc/apoc` 目錄中。您可以手動方式或使用 Configuration Agent 配置精靈編輯這些特性檔案 (請參閱附錄 A) 。

配置精靈提供一個圖形化使用者介面，可指導您瞭解 Configuration Agent 的必要設定。精靈的每一個頁面都會有相應的說明螢幕。您可以使用 `/usr/bin/apoc-config` 程序檔，以超級使用者身份 (root) 啓動精靈。

**備註 –** 也可以不啓動圖形介面而啓動該精靈。例如，在主控台模式下執行 `/usr/bin/apoc-config -nodisplay` 以啓動精靈。

# 啓動程式資訊

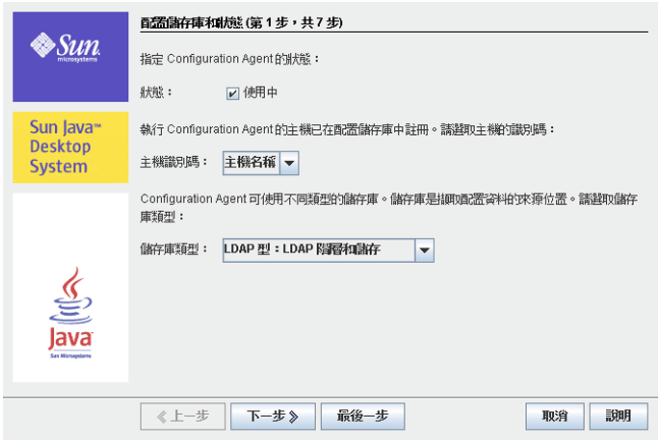


圖 3–1 Configuration Agent，配置儲存庫

---

**備註 –** 在適當的位置，關聯的特性檔案關鍵字在圓括號中進行指示。

---

- 狀態：Configuration Agent 的狀態。該核取方塊可用於啓動或關閉 Configuration Agent。若要使用配置儲存庫，Configuration Agent 必須處於使用中。啓動會自動由 Solaris 服務管理功能 ( smf(5) ) 中納入所需要的註冊。
- 主機識別碼 (HostIdentifierType)：可爲「主機名稱」或「IP 位址」。在搜尋主機特有策略資料時，Configuration Agent 會根據主機名稱或 IP 位址識別目前的主機。請根據主機在所選「環境類型」中的識別方式選擇正確的值。
- 環境類型：此設定可用於指示 Configuration Agent 您是在 LDAP 或檔案型儲存中定義組織階層與配置資料，或兩者中皆有定義。

---

**備註 –** 若要手動啓用或停用 Configuration Agent，請以 **root** 身份登入，再分別鍵入 `/usr/lib/apoc/apocd enable` 或 `/usr/lib/apoc/apocd disable` 指令。

---


**圖 3–2** Configuration Agent，LDAP 階層和檔案型儲存

---

**備註 –** 圖 3–2 的螢幕會隨著上一個畫面中所選之「環境類型」而不同。選擇 LDAP 或混合型環境類型時，必須具備伺服器識別碼、伺服器連接埠及字尾。選擇檔案型或混合型環境類型時，則須具備「配置設定 URL」。

---

- 伺服器識別碼：LDAP 伺服器的主機名稱。
- 伺服器連接埠：LDAP 伺服器的連接埠號。
- 字尾：LDAP 儲存庫的基底 DN。
- 配置設定 URL：指定檔案型儲存庫位置的 URL。

  若連線至第一個儲存庫失敗，URL 清單可用以指定備用的儲存庫。此清單可由一或多個以空格加以分隔的 URL 組成；每一個 URL 的格式均須爲 `file://<filepath>`、`http://<host>:<port>/<filepath>` 或 `https://<host>:<port>/<filepath>`。如需更多資訊，請參閱附錄 A。

  ---

  **備註 –** 代理程式會先嘗試使用 SSL 連線存取 LDAP 伺服器。此連線若是失敗，代理程式才會接著嘗試一般的 SSL 連線。

  SSL 連線要能夠成功，Java 執行階段環境的金鑰庫中必須存有適當的憑證。標準 JRE 的金鑰庫位於 `<installation directory>/lib/security/cacerts`，而標準 JDK 的金鑰庫則位於 `<installation directory>/jre/lib/security/cacerts`。無論是「憑證授權單位」或 LDAP 伺服器憑證，均須透過 `keytool -import -file <certificate file> -keystore <cacerts file location>` 指令增加至該金鑰庫內。該金鑰庫的預設密碼爲 **`changeit`**。

  ---

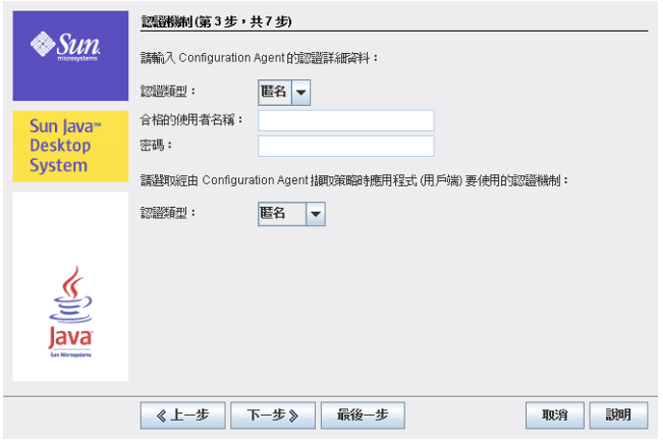


圖 3–3 Configuration Agent，驗證機制

- Configuration Agent 的認證類型：可爲「匿名」或「簡單」。若選取「匿名」，便會自動停用 [合格的使用者名稱] 和 [密碼] 欄位。
- [合格的使用者名稱] (AuthDn)：具有對儲存庫的讀取和搜尋存取權之使用者的完整 DN。
- [密碼] (Password)：已註冊 LDAP 使用者的密碼

**備註 –** 目錄中若已啓用匿名存取，即可將 [合格的使用者名稱] 和 [密碼] 設定留爲空白。

- 應用程式的 [認證類型] (AuthType)：依據 LDAP 伺服器驗證使用者的方法，可爲「匿名」或「GSSAPI」。

**備註 –** 如需更多資訊，請參閱第 24 頁的「資料存取/使用者認證」。

# 連接埠設定

Configuration Agent 使用兩個連接埠：

- [代理程式連接埠] (DaemonPort)：由代理程式用來與用戶端應用程式進行通訊 (預設爲 **38900**)。
- [管理連接埠] (DaemonAdminPort)：代理程式控制器程式 apocd 與代理程式通訊時，會使用此連接埠 (預設值爲 **38901**)。


**圖 3–4** Configuration Agent，連接埠設定

# 變更偵測間隔

Configuration Agent 使用以下兩種間隔定期檢查配置資料中的所有變更：

- [一般偵測間隔] (ChangeDetectionInterval)：桌面應用程式的 (用戶端的) 配置資料之變更偵測循環的間隔 (以分鐘爲單位)。

  **備註 –** 指定 **-1** 會關閉變更偵測。

- [代理程式設定的間隔] (DaemonChangeDetectionInterval)：代理程式特定配置設定之變更偵測循環的間隔 (以分鐘爲單位)。

  **備註 –** 指定 **-1** 會關閉變更偵測。

您可以使用一般偵測間隔來調準遠端配置資料變更至用戶端應用程式的傳遞。爲該設定提供的值，是遠端所做變更在用戶端應用程式中反映出來之前所需的最大時間長度 (以分鐘爲單位)。

較小的值將會導致 Configuration Agent 和 LDAP 伺服器活動的增加。因此，調整設定值時應該謹慎。例如您可以在初始部署階段時，將值設定爲一分鐘，以測試遠端配置對用戶端應用程式的影響。完成測試之後，請將此設定恢復爲初始值。

# 操作設定


圖 3–5 Configuration Agent，資料目錄

可以配置以下設定：

- [資料目錄] (DataDir)：用於儲存運行時間資料的目錄。預設爲 **/var/opt/apoc**。
- [快取的資料儲存生命] (TimeToLive)：非離線配置資料在本機資料庫中保留的間隔 (以分鐘爲單位)。


圖 3–6 Configuration Agent，請求處理和記錄

- [資源回收循環] (GarbageCollectionInterval)：本機配置資料庫中資源回收循環的間隔 (以分鐘爲單位)。
- [最大用戶端執行緒數] (MaxClientThreads)：可同時處理的用戶端請求的最大數目。
- [最大用戶端連線數] (MaxClientConnections)：用戶端連線的最大數目。

- [最大請求大小] (MaxRequestSize)：用戶端請求的最大大小。
- [連線逾時] (ConnectTimeout)：表示 LDAP 伺服器回覆連線請求所允許的間隔。預設值爲 1 秒。
- [記錄層級] (LogLevel)：代理程式記錄檔中詳細資訊的級別。記錄層級與 Java 記錄程式層級一致。下列層級會依其嚴重性遞減排列：
  - 嚴重
  - 警告
  - 資訊
  - 配置
  - 詳細
  - 較詳細
  - 最詳細

**備註 –** 大多數操作設定 ([資料目錄] 和 [連線逾時] 設定除外) 亦可透過 LDAP 伺服器中所儲存的相對應策略集中進行維護。如果您要使用此項功能，請勿透過精靈調整相應的設定，而是使用 Desktop Manager 中的 Configuration Agent 策略來集中指定操作設定。

# 套用代理程式設定

除「資料目錄」和「連線逾時」之外，透過 Desktop Manager 儲存於 LDAP 伺服器上的操作設定將會在代理程式配置的下一次變更偵測循環開始時自動生效 (請參閱 DaemonChangeDetectionInterval)。


**圖 3–7** Configuration Agent，摘要頁面

本機變更的所有其他設定均需要重新載入或重新啓動 Configuration Agent。如果您使用配置

精靈，重新載入或重新啓動則會自動執行。

**備註 –** 若要以手動方式重新啓動 Configuration Agent，請確定所有相關用戶端應用程式皆不在執行中，然後以 root 身份登入，並鍵入 `/usr/lib/apoc/apocd restart` 指令。

# 其他代理程式設定

**備註 –** 配置精靈中不會提供下列設定。

- 本機策略的應用程式 (ApplyLocalPolicy)：使用此設定可指定是否要將本機主機上的策略資料，提供給用戶端應用程式使用。值「true」表示應提供本機策略資料。值「false」表示不應提供本機策略資料。如需更多資訊，請參閱第 23 頁的「使用本機策略」。

# 使用本機策略

您可以將 Configuration Agent 配置成套用本機部署策略的配置設定，以此做爲所提供之全域策略的替代方案。請使用下列步驟部署這類本機策略：

## ▼ 部署本機策略

1. **使用 Desktop Manager 建立具有必要策略設定的設定檔。**
2. **使用 Desktop Manager 將設定檔匯出為壓縮檔。**
3. **在您的用戶端主機上建立 `${DataDir}/Policies/profiles/PROFILE_REPOSITORY_default` 目錄 (如其不存在)。**

   將 `${DataDir}` 對應至 Configuration Agent 的「資料目錄」值 (預設爲 `/var/opt/apoc`)。
4. **將之前所匯出的壓縮檔複製至 `${DataDir}/Policies/profiles/PROFILE_REPOSITORY_default`。**
5. **確定已將 Configuration Agent 配置成套用適用的本機策略 (如需更多資訊，請參閱第 23 頁的「其他代理程式設定」)。**

**備註 –** 若變更了 Configuration Agent 的「ApplyLocalPolicy」設定，必須以 root 身份登入，並鍵入 `/usr/lib/apoc/apocd reload` 指令重新載入 Configuration Agent。

所有以此方式部署的本機策略，均會在下一個 Configuration Agent 變更偵測週期中提供給用戶端使用。

## 自動重新啓動 Configuration Agent

萬一失敗，Configuration Agent 會自動重新啓動。服務管理功能 ( smf(5) ) 負責做出決定。若服務管理功能認爲不宜重新啓動 (例如失敗的次數過多)，便會將 Configuration Agent 置於維護模式中。

若未重新啓動 Configuration Agent，請以 root 身份登入，並執行 `/usr/lib/apoc/apocd disable` 指令暫時停用代理程式；再於修正造成代理程式故障的問題之後，執行 `/usr/lib/apoc/apocd enable` 指令重新啓用代理程式。

# 資料存取/使用者認證

Configuration Agent 根據桌面使用者的登入 ID 從 LDAP 伺服器中擷取資訊。組織對映檔案的 `User/UniqueIdAttribute` 設定會將登入 ID 對映至 LDAP 伺服器中的使用者元素。Configuration Agent 還擷取有關主機的資訊，例如主機的名稱或 IP 位址。此資訊會經由組織對映檔案的 `Host/UniqueIdAttribute` 設定對映至 LDAP 伺服器中的主機元素。如需有關組織對映的更多資訊，請參閱附錄 C。

存取 LDAP 伺服器有兩種方法，即匿名存取或使用 GSSAPI 存取。對於匿名存取，不需要在桌面上執行動作。對於 GSSAPI 方法，則必須在桌面上獲取 Kerberos 憑證。若要整合獲取的 Kerberos 憑證和使用者登入，則必須在 Java Desktop System 主機上安裝和配置 `pam_krb5` 模組。

您可以使用 `gdm` 來整合 Kerberos 和使用者登入，例如，透過使用下列 `/etc/pam.d/gdm` 檔案：

```
#%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
```

如果您按照此方法整合 Kerberos 和使用者登入，則應啓用螢幕保護程式的 Kerberos 支援。例如藉由使用下列 `/etc/pam.d/xscreensaver` 檔案：

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

# 介面

應用程式介面是 Desktop Manager 所支援之配置系統的延伸。介面會視配置系統的不同，而決定各應用程式是否可以選擇使用中央配置資料。受支援的配置系統包括：

- GConf：Gnome 配置系統，桌面和多數 Gnome 應用程式均會使用，如 Evolution。
- StarSuite Registry：StarSuite 和 OpenOffice.org 所使用的配置系統。
- Mozilla Preferences：Mozilla 所使用的配置系統。
- Java Preferences：供 Java 應用程式使用的配置 API。

此外亦提供桌面定義介面，可以將桌面啓動程式、功能表項目和啓動程式增加到使用者的桌面上。

## GConf 介面

GConf 介面屬於 Solaris 之 `SUNWapoc-adapter-gconf` 套裝軟體的一部分。當您從相應的 packageAdapter 安裝介面時，即會更新 `/etc/gconf/2/path` 中的 GConf 資料來源路徑，將 Desktop Manager 資源併入其中。介面提供的兩種資料來源包括：

- 「apoc:readonly:」：提供對策略中不受保護設定的存取權限。請在使用者設定與本機預設值之間插入此資料來源。
- 「apoc:readonly:mandatory@」：提供對策略中受保護設定的存取權限。請在本機強制設定與使用者設定之間插入此資料來源。

### GConf 介面配置

GConf 介面會在安裝時進行配置，但是其運作則須取決於代表強制中央設定及預設設定之兩項資料來源的 GConf 路徑檔案 (`/etc/gconf/2/path`) 存在與否而定。安裝之後，當此路徑檔案中包含了正確的資訊，以便於 GConf 能夠如預期般地選用中央設定時，管理員必須確定檔案中仍保有前綴詞爲「apoc」的資料來源，使之能夠在需要時修改該路徑，以納入其他自訂的資料來源。您亦須確定此資料來源位於代表強制中央設定之資料來源的本機強制設定與使用者設定之間，以及代表預設中央設定之資料來源的使用者設定與本機預設設定之間。

## Java Preferences 介面

「Java Preferences」介面屬於 Solaris 之 `SUNWapcj` 套裝軟體的一部分。

### Java Preferences 介面配置

「Preferences API」實作中會提供「Java Preferences」介面；「Preferences API」是其他現有實作 (如 JRE 所提供的預設檔案型系統) 所必須的包裝程式。若要能夠在使用「Preferences API」的 Java 應用程式中使用中央配置，必須撰寫該應用程式的啓動程序檔，並以 `/usr/lib/apoc/apocjlaunch` 程序檔爲其輔助程式。此程序檔需要定義一些環境變數，並在其尾端併入 `apocjlaunch` 程序檔 (會以必要的環境啓動 Java 應用程式)。必須設定的環境變數包括：

- `JAVA`：包含 Java 執行階段可執行檔的路徑。
- `APPLICATION`：包含該應用程式的一般 Java 執行階段呼叫尾隨部分。例如單一類別啓動的 *classname [arguments]* 或 jar 歸檔啓動的 `-jar` *jarname [arguments]*。

其他可以設定的選擇性環境變數包括：

- `CLASSPATH`：以冒號分隔的 jar 或類別檔案清單，必須是應用程式類別路徑的一部分。
- `DEFINES`：包含定義敘述的字串，必須是應用程式啓動的一部分。
- `PREFFACTORY`：應用程式必須使用之基礎「Preferences API」實作中的工廠類別名稱。

## Mozilla 介面

Mozilla 介面是 Solaris 之 `SUNWmozapoc-adapter` 套裝軟體的一部分。

### Mozilla 介面配置

Mozilla 介面會在安裝此產品時一併安裝，並不需要進行其他任何配置。

## StarSuite 介面

StarSuite 介面隨附於標準的 StarSuite 安裝中，可讓您存取設定檔配置資料，而不需要任何特殊的修改。

### StarSuite 介面配置

StarSuite 介面會在安裝此產品時一併安裝，並不需要進行其他任何配置。

# 桌面定義介面

「桌面定義」介面由下列套裝軟體組成：

| 套裝軟體名稱 | 說明 |
|---|---|
| SUNWapleg | 配置存取二進位檔 |
| SUNWardsa | 桌面定義介面 |
| SUNWardsa-misc | 介面的系統整合 |

這些套裝軟體會在安裝 Desktop Manager 用戶端元件時一併安裝，並不需要其他任何的設定。

## 桌面定義介面配置

「桌面定義」介面透過使用者登入時所使用的設定程序進行配置，並不需要其他任何的設定。

# 移除介面

Mozilla 及 StarSuite 介面會在移除這些產品時一併移除。只要將安裝一節所述之套裝軟體移除，並使用適當的套裝軟體管理系統工具，便可將 GConf、Java Preferences 及桌面定義介面移除。

移除「Java Preferences」介面之後，便不應再繼續使用因爲啓動 Java 應用程式而利用「Preferences API」所撰寫的啓動程序檔。由於部分必要的類別將無法再繼續使用，因此會導致需要使用這些類別的 Java 呼叫失敗。

# 介面疑難排解

可能會造成看不到相應應用程式之中央配置資料的問題中，有絕大部分是 Configuration Agent 所造成，這是因爲所有介面皆會使用 Configuration Agent 擷取資料所致。

中央配置中如有某項設定 (或其群組) 未見生效，可能是使用者已在應用程式中明確設定了該設定的值 (一般會透過產品的 [選項] 或 [喜好設定] 對話方塊)。此時除非將中央設定定義爲保護 (意即管理員強制使用該值，使用者無法加以修改)，否則使用者的喜好設定將優於 Desktop Manager 所設定的值。

# Configuration Agent 疑難排解

本節將針對 Configuration Agent 之性質及運作方式的相關問題提出解答，並提供疑難排解此代理程式問題的相關提示。

## 問題與回答

### Configuration Agent 是什麼？其如何運作？

Configuration Agent 是一種應用程式，專責策略快取與傳遞的工作。其設計建立目的在能夠集中配置桌面用戶端應用程式，但不會對這些應用程式及其執行所在主機的效能造成巨幅影響。若要達成上述目的，可執行下列作業：

- 將下載的策略快取至本機所提供的快取中，以供用戶端日後之用。
- 分享可以共用的昂貴資源 (例如對裝載有策略之 LDAP 伺服器的連線)。

用戶端應用程式與 Configuration Agent 之間的互動一般來說非常簡單，如下所述：

1. 使用者啓動任一項桌面用戶端應用程式 ( gconfd、Mozilla 或 StarSuite)。
2. 用戶端應用程式連線至 Configuration Agent。
3. 用戶端應用程式向 Configuration Agent 請求所需的策略資料。
4. Configuration Agent 從其快取中搜尋請求的策略資料。
5. 快取中若找不到策略資料，Configuration Agent 便會從預先配置之策略儲存庫下載所需的資料，再將此資料儲存到快取中。
6. 接著再將此策略資料傳送到提出請求的用戶端應用程式。
7. Configuration Agent 會監視策略儲存庫中之策略資料是否有所修改。
8. 當偵測到修改時，Configuration Agent 即會重新整理其快取，將其保持在最新狀態，並通知用戶端應用程式此項修改。

### 如何才能取得 Configuration Agent 並加以安裝？

Configuration Agent 隨附於 Solaris 10 中，預設會隨其一併安裝。

### 我剛安裝好 Solaris 10。接下來該如何做？

Configuration Agent 預設爲停用，並不會對其進行配置。若要使用 Configuration Agent，至少必須予以最低限度的配置，並加以啓用。完成這些步驟後，桌面用戶端應用程式便會在下次啓動時，自動開始使用您所提供的策略。

### 我想要配置 Configuration Agent。請問該如何做？

若要正確配置 Configuration Agent，請使用 [Configuration Agent 精靈]。您可使用超級使用者身份執行 `/usr/bin/apoc-config` 指令啓動精靈。精靈會引導您完成正確配置代理程式所需的步驟。通常只需備妥策略儲存庫的位置資訊，即可完成精靈所需的設定。

手動編輯 Configuration Agent 的配置檔是配置此程式的另一種方式。但由於以此方式配置此代理程式極易出錯，因此不建議使用。[Configuration Agent 精靈] 另外還包含其他邏輯，可以決定必須重新啓動或重新載入此代理程式的配置變更項目。

## 我想要啓用 Configuration Agent。請問該如何做？

您可以使用下列三種機制啓用此代理程式：

1. 使用 [Configuration Agent 精靈] (`/usr/bin/apoc-config`)，並將代理程式狀態設定爲「使用中」。
2. 使用 Configuration Agent 控制器程式 (`/usr/lib/apoc/apocd`)，並以超級使用者身份執行下列作業：

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

3. 使用 smf(5)，並以超級使用者身份執行下列作業：

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

## 我已配置並啓用了 Configuration Agent。要如何知道其是否運作？

瞭解 Configuration Agent 的配置是否正確及其運作與否最簡單的方法，便是使用 Desktop Manager 建立策略，並將此策略指定給使用者，再以該使用者身份登入桌面機器，然後驗證系統是否有使用您所建立的策略設定。桌面階段作業中有許多容易偵測到的策略設定，例如背景和主題。

## 為何要啓用 Configuration Agent？

Configuration Agent 是與 smf(5) 相容的服務，啓用代理程式的概念也與 smf(5) 相同。代理程式一經啓用後，即可隨時提供服務。當您啓用此代理程式時，會發生下列動作：

- 啓動代理程式
- 任何於此代理程式啓用之後啓動的桌面用戶端應用程式，皆可擷取策略資料。
- 系統開機時，代理程式會自動重新啓動

## 我如何得知啓用 Configuration Agent 與否？

您可以使用下列一種方法，判別 Configuration Agent 啓用與否：

- 使用 Configuration Agent 的控制器程式。以超級使用者身份鍵入下列指令：

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

  代理程式若爲啓用，控制器程式會傳回下列訊息：

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

  否則，控制器程式會傳回下列訊息：

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

- 使用 smf(5) 執行下列指令：

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

  代理程式若爲啓用，svcs 會傳回下列訊息：

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

  代理程式若爲停用，svcs 會傳回下列訊息：

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

  代理程式若處於維護模式，svcs 會傳回下列訊息：

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

## 我如何得知 Configuration Agent 是否已在執行中？

您可以使用下列一種方法，判別 Configuration Agent 是否已在執行中：

- 以超級使用者身份執行 Configuration Agent 控制器程式：

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

  如果代理程式已啓用，控制器程式就會傳回下列訊息：

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

  否則，控制器程式就會傳回下列訊息：

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

- 執行下列指令：

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

  代理程式若在執行中，svcs 會傳回下列訊息：

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

  代理程式若不在執行中，svcs 會傳回下列訊息：

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

  代理程式若處於維護模式，svcs 會傳回下列訊息：

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

- 執行下列指令：

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

  Configuration Agent 若在執行中，則 ps 輸出中應會出現下列相關 Java 程序：

  ```
   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
  ```

## 記錄檔位於何處？

需要疑難排解 Configuration Agent 的問題時，可以參考下列記錄檔：

- smf(5) 記錄檔：
  - `/var/svc/log/network-apocd-udp:default.log` 檔案會記錄嘗試啓動及停止 Configuration Agent 相關實例的事件。此檔案亦包含 Configuration Agent 控制器程式 `/usr/lib/apoc/apocd` 寫入其標準輸出中的訊息，以及 JVM 或 Configuration Agent 的輸出訊息。
  - `/var/svc/log/svc.startd.log` 記錄檔是 smf(5) 事件的概要記錄。例如啓動 Configuration Agent 時，若是在極短的期間內連續發生多次失敗，smf(5) 便可能只會將此狀況判定爲無法啓動 Configuration Agent。此時 smf(5) 會將 Configuration Agent 置於維護模式，並將此結果寫入記錄中。

  這兩份記錄檔在發生 Configuration Agent 啓動問題時非常有用。
- Configuration Agent 記錄：

  Configuration Agent 會將記錄訊息寫入預設記錄目錄 (`/var/opt/apoc/Logs`) 中的記錄檔內。Configuration Agent 的「資料目錄」爲 `/var/opt/apoc`。您可使用 [Configuration Agent 精靈] (`/usr/bin/apoc-config`) 應用程式變更此目錄的位置。變更 [Configuration Agent 精靈] 的「記錄層級」，即可變更記錄訊息的詳細程度。您若是認爲之前的 Configuration Agent 配置不正確，或發生其他類型的代理程式失敗，可以在查閱代理程式記錄檔案之前，先使用 [Configuration Agent 精靈] 將「記錄層級」設爲「最詳細」。此步驟可讓您獲取最大量的可用記錄資訊。
- 系統記錄：

  您也可以查看 `/var/adm/messages` 記錄檔，或 SunRay 機器上的 `/var/opt/SUNWut/log/messages` 記錄檔，以診斷 Configuration Agent 的問題。

## 我該如何提升代理程式記錄機制的詳細程度？

請參閱第 31 頁的 「記錄檔位於何處？」

## 何謂維護模式？

smf(5) 若是偵測到代理程式發生啓動或重新啓動問題時，便會將 Configuration Agent 置於維護模式。smf(5) 若無法啓動代理程式，會不斷嘗試重新啓動，直到啓動成功，或 smf(5) 判定代理程式無法啓動爲止。在後一種情況下，smf(5) 會將代理程式置於維護模式，藉此通知您必須處理所發生的問題。問題處理完畢之後，即可清除代理程式的 smf(5) 狀態，回復正常作業。

## 如何離開維護模式，亦即清除 smf(5) 狀態？

以超級使用者身份執行 `/usr/sbin/svcadm clear svc:/network/apocd/udp` 指令。

## Configuration Agent 未預期地停止執行；請問發生了何種狀況？

smf(5) 偵測到代理程式已停止執行，並嘗試予以重新啓動。不論何故，若重新啓動的嘗試接連失敗，則 smf(5) 便會將代理程式置於維護模式。重新啓動代理程式若是成功，便不會對執行中的桌面用戶端應用程式造成影響。這類用戶端應用程式皆會在代理程式重新啓動後，自動重新連線至代理程式。

## 啓用/啓動代理程式之後，是否需要重新啓動桌面用戶端應用程式？

您所需要採取的動作，取決於特定桌面用戶端應用程式啓動時，代理程式是否已經啓用/執行而定。若已啓用/執行代理程式，用戶端應用程式便會建立對該代理程式的連線，並在連線中斷時嘗試重新建立連線。換言之，您每次啓動、啓用或停用代理程式時，只要代理程式處於執行的狀態，用戶端應用程式便會自動嘗試重新連線至代理程式。若 Configuration Agent 在用戶端應用程式啓動時不爲啓用/執行，則用戶端應用程式便不會使用該代理程式，也不會在代理程式啓動後嘗試與其連線。這些實際上皆表示：

- 於代理程式啓用/執行之後啓動的桌面用戶端應用程式無需再重新啓動。
- 於代理程式尚未啓用/執行之前啓動的桌面用戶端應用程式必須重新啓動。

## 桌面用戶端應用程式似乎未使用配置的策略。我該如何做？

常見與 Configuration Agent 相關問題，便是無法查看所配置的策略對桌面用戶端應用程式的效果。此問題最常見的原因是代理程式配置錯誤、策略儲存庫配置錯誤，或無法使用策略儲存庫。下列準則有助於探索與解決這類問題：

- 確定已配置代理程式。
- 確定已啓用/執行代理程式。如需啓動代理程式，亦須重新啓動目前已開啓的桌面用戶端應用程式。
- 如問題依然無法解決，請暫時提升代理程式記錄機制的詳細程度，然後 (如有可能) 重新啓動代理程式，以取得代理程式啓動之後所有的完整詳細記錄資訊。
- 代理程式若無法正確啓動，請參閱第 33 頁的「啓動 Configuration Agent 時發生問題」一節。
- 代理程式若可正確啓動，但桌面用戶端應用程式卻未使用所提供的策略，請參閱「從執行中之 Configuration Agen 取得策略時發生問題」一節。

- 若仍然無法解決問題，請連絡「技術支援」。

## 啓動 Configuration Agent 時發生問題

Configuration Agent 若無法啓動，但已配置並啓用了 Configuration Agent，便須查閱記錄檔。 下列各節將說明此問題的常見錯誤。

### 無效或無法存取的代理程式資料目錄

代理程式會建立「Configuration Agent 資料目錄」，以儲存記錄檔、策略快取等。此目錄的預設位置爲 `/var/opt/apoc`。

當將「資料目錄」設在無法存取的位置 (即 `/dev/null/cant/write/here`) 時，Configuration Agent 便會將下列錯誤訊息寫入 smf(5) 記錄中。若要解決此問題，請使用 [Configuration Agent 精靈] (`/usr/bin/apoc-config`) 將「資料目錄」指向可存取的位置。

```
[ 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 ]
```

## 使用忙碌中的用戶端請求連接埠

Configuration Agent 會使用 TCP/IP 通訊端連線與桌面用戶端應用程式通訊。這些連線預設會透過連接埠 38900 加以建立。

當將 Configuration Agent 配置成使用連接埠 1234 (其他服務已在使用此連接埠) 時，會產生下列錯誤訊息。此錯誤訊息會記錄在 Configuration Agent 記錄中。若要解決此問題，請使用 [Configuration Agent 精靈] (`/usr/bin/apoc-config`)，將「代理程式連接埠」的設定變更成未使用的連接埠號。

```
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)
```

## 使用忙碌中的管理連接埠

Configuration Agent 會使用 TCP/IP 通訊端連線與 Configuration Agent 控制器程式 (/usr/lib/apoc/apocd) 通訊。這些連線預設會透過連接埠 38901 加以建立。

當將 Configuration Agent 配置成使用其他服務已在使用的連接埠 1234 時，Configuration Agent 記錄中會產生下列錯誤訊息。若要解決此問題，請使用 [Configuration Agent 精靈] (/usr/bin/apoc-config)，將「管理連接埠」的設定變更成未使用的連接埠號。

```
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
```

# 從執行中之 Configuration Agent 取得策略的問題

## 缺少配置儲存庫規格或其無效

Configuration Agent 必須連線至有效的配置儲存庫，才可下載及快取策略資訊。若未在代理程式配置中正確指定配置儲存庫 (例如使用無效的格式或未指定儲存庫)，便會在桌面用戶端應用程式啓動時，於 Configuration Agent 記錄中寫入類似於下列的錯誤。若要解決此問題，請使用 [Configuration Agent 精靈] (`/usr/bin/apoc-config`) 指定所要使用的配置儲存庫。

```
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
```

## 無法連線至策略儲存庫

Configuration Agent 必須連線至有效的配置儲存庫，才可下載及快取策略資訊。若無法建立連線，便會在桌面用戶端應用程式啓動時，於 Configuration Agent 記錄中寫入類似於下列的錯誤。下列情況有可能是以此方式建立的主機不存在、無法連線至該主機，或無法透過連接埠 389 存取 LDAP 伺服器。若要解決此問題，請使用 [Configuration Agent 精靈] (`/usr/bin/apoc-config`) 確定您已正確指定「策略儲存庫」；之後再確定您可存取此「策略儲存庫」。例如對於 LDAP 儲存庫，您必須確定 LDAP 伺服器是否正在執行中；是否可以在網路上使用裝載此 LDAP 伺服器的機器；以及指定的連接埠是否即是此 LDAP 伺服器所使用的連接埠。

若嘗試使用 SSL 連線存取 LDAP 伺服器，請確定金鑰庫中存有適當的憑證，且該金鑰庫與用以執行 Configuration Agent 的 Java 執行階段環境相關聯。如需有關 `apoc-config` 的更多資訊，請參閱第 15 頁的 「Configuration Agent」一節。

```
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
```

## 連線至未經配置的策略儲存庫

您必須先正確地配置「策略儲存庫」，Configuration Agent 才可在「策略儲存庫」中找到策略資料。若指定未經配置或配置錯誤的「策略儲存庫」，便會在桌面用戶端應用程式啓動時，於 Configuration Agent 記錄中寫入類似於下列的錯誤。若要解決此問題，請參閱本節。

```
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)
```

## 我在 Configuration Agent 記錄中看到「用戶端連線的最大數目」訊息。請問這代表什麼意思？

Configuration Agent 所啓用的各個桌面用戶端應用程式 (gconfd、Mozilla、StarSuite)，皆會在 Configuration Agent 執行時，開啓對 Configuration Agent 的連線。這類連線的限制則會在代理程式配置中指定。預設的連線限制爲 50。機器上如有多名使用者，可能須在 [Configuration Agent 精靈] (`/usr/bin/apoc-config`) 中變更「最大用戶端連線」設定，以提高此限制。當 Configuration Agent 達到最大連線數時，便會在 Configuration Agent 記錄中寫入類似於下列的錯誤訊息：

```
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.
```

## 我使用 Desktop Manager 修改了一些策略，但在我的用戶端機器上看不到這些修改。

設計 Configuration Agent 之初的假設之一，是 Desktop Manager 所建立的策略資料爲相對靜態，不會經常變更。此項假設導致代理程式會間歇性地查看「策略儲存庫」，以確定其是否有所修改。代理程式預設會每隔一小時檢查所有執行中之桌面應用程式的儲存庫。因此當您使用 Desktop Manager 執行變更之後，必須等待一小時，執行中的桌面應用程式才會收到此項變更的通知。如有需要，可以使用 [Configuration Agent 精靈] (`/usr/bin/apoc-config`) 變更「一般偵測間隔」的值，以增加檢查儲存庫的頻率。您也可以使用超級使用者身份執行 `/usr/lib/apoc/apocd change-detect` 指令，以強制 Configuration Agent 重新整理所有連線應用程式的策略資料。

◆ ◆ ◆ 第 4 章

# 4 Java Web Console

Java Web Console 的設計旨在生產一可共用於 Sun Microsystems 產品的網路型系統管理解決方案。使用者可以從中存取系統管理應用程式，而所有應用程式均提供一致的使用者介面。

主控台基於 Web 模型出於多種原因。但主要原因是讓系統管理員能夠使用 Web 瀏覽器來存取其系統管理應用程式。

Java Web Console 提供：

- 共用認證與授權
- 共用記錄
- 單一進入點，即透過基於 HTTPS 的同一連接埠存取所有系統管理應用程式
- 共同的外觀和感覺

主控台的優點是管理員僅需登入一次，即可使用主控台內部的所有應用程式。

# 安裝

## 系統需求

Java Web Console 支援多用戶端、伺服器作業系統以及數種瀏覽器。

### 用戶端

- Solaris 10 上的 Netscape™ 6.2x 及 7.x
- Windows 98、98 SE、ME、2000 及 XP 上的 Netscape 6.2x 及 7.x
- Windows 98、98 SE、ME、2000 及 XP 上的 Internet Explorer 5.5x 及 6.x
- Solaris 上的 Mozilla 1.4x
- Solaris 上的 Firefox 1.0

### 伺服器

- Solaris 10
- Red Hat Application Server 2.1、3.0
- SuSE Linux 8.0 或更高版本
- J2SE 版本 1.4.1_03 或更高版本

  如果在您的伺服器上偵測到 J2SE 1.4.1 或更早版本，安裝程式會提示您使用 Java Desktop System Management Tools CD 中的 J2SE 版本來升級安裝。
- Tomcat：4.0.3 或更高版本

  Tomcat 包括在 Java Desktop System Management Tools CD 中

## 安裝 Java Web Console

Java Web Console 2.2.4 是 Solaris 10 作業系統的一部分，但 Desktop Manager 則須使用版本 2.2.5。`server/console` 目錄的 Desktop Manager 歸檔中會提供版本 2.2.5 的副本。執行該目錄中的 `./setup` 即可進行安裝。

若已安裝 Java Web Console 3.0，便須先解除安裝版本 3.0，然後再安裝上述位於 `server/console` 目錄中的 Java Web Console 2.2.5。

# 執行主控台

若要註冊新的應用程式，一般只需要先停止再重新啓動 Java Web Console 伺服器即可。

**注意 –** 第一次啓動 Java Web Console 之前，請先確認是否已安裝了 Desktop Manager。主控台中至少須部署一個應用程式，否則 Java Web Console 將無法順利執行。

- 若要啓動 Java Web Console，請鍵入 `smcwebserver start`。
- 若要停止 Java Web Console，請鍵入 `smcwebserver stop`。
- 若要重新啓動 Java Web Console，請鍵入 `smcwebserver restart`。
- 若要存取 Java Web Console，請在瀏覽器中輸入下列 URL：`https://`*<hostname>*`.`*<domainname>*`:6789`

最新提供之 Java Web Console 除可支援 Unix 型認證之外，還可支援角色型的存取控制 (RBAC)。然而，您也可以配置其他驗證機制，如 LDAP 驗證。

**備註 –** 預設工作時段逾時爲 15 分鐘。您可以使用 `smreg` 指令配置逾時長度。例如，若要將逾時長度設爲 5 分鐘，請鍵入 `smreg add -p -c session.timeout.value=5`。

如需有關 Java Web Console 指令的更多資訊，請參閱 smcwebserver 和 smreg 線上手冊。

# 移除 Java Web Console

**注意 –** 在 Solaris 上無法移除 Java Web Console，因其隸屬於作業系統的一部分。

# Java Web Console 疑難排解

## 無法安裝 Java Web Console

徵兆：安裝結束時，會出現訊息指出 Java Web Console 因爲沒有註冊的應用程式而無法啓動。

可能原因：安裝 Desktop Manager 模組之後，便會啓動 Java Web Console。

## 連線被拒

徵兆：嘗試開啓適當的 URL (如 `https://`*<your.server>*`:6789`)，但連線卻被拒絕。

可能原因：伺服器上未執行 Java Web Console。

## 無法登入

**備註 –** 預設不會安裝 LDAP 登入模組。因此不會將登入憑證與 LDAP 伺服器中所儲存的憑證進行比較，而只需要一般的系統登入憑證即可。本小節的疑難排解僅適用於手動安裝 LDAP 登入模組的狀況。

徵兆：已進入 Web Console 的登入頁面，但所使用的使用者/密碼組合遭到拒絕。

可能原因：

- LDAP 伺服器不在執行中。
- Web Console 之 LDAP 認證模組的配置不正確。
- LDAP 伺服器上沒有使用者。
- 使用者在 LDAP 伺服器上使用了不同的密碼。

# 無 Desktop Manager 連結

徵兆：您登入了 Web Console，但應用程式清單頁面中卻沒有 Desktop Manager。

可能原因：

- 未安裝 Desktop Manager 模組。

# 空指標異常，Tomcat/Java 錯誤或空白

徵兆：開啓了 Desktop Manager，卻未顯示任何值，而只出現了空白頁面或一些錯誤。

可能原因：錯誤中如有 `NoClassDefFoundError:sun/tools/javac/Main` ，即表示 Java Web Console 使用了錯誤的 Java 安裝。

# 其他問題

Web 伺服器若未正常執行，可查看記錄檔中的相關訊息。這些記錄檔位於 `/var/log/webconsole/` 中。您可以使用 `smreg` 提高記錄的詳細程度：

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

原始設定可以下列方式加以儲存：

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

下列指令可觸發傾印配置資料庫中的所有資料：

```
smreg list
```

可能是裝載 Desktop Manager 的 Web 伺服器未正確關機，致使連接埠仍處於使用中的狀態。這會造成剛啓動的 Web 伺服器完全無法啓動。`smcwebserver start/restart` 指令如有發出錯誤訊息；或 Desktop Manager 在發出 `smcwebserver stop` 之後仍可進行存取；或剛啓動的伺服器仍依舊實例的方式執行，請檢查連接埠 6789 是否仍在使用中 (`netstat -a | grep 6789`)，或 Web 伺服器是否仍在執行中 (`ps -ef | grep java`)。不論爲上述何種情況，均應刪除相應的程序，以解除連接埠 6789 的使用中狀態。

◆ ◆ ◆ 附錄 A

# 配置參數

可以定義下列 Desktop Manager 元件的這些參數：

- Desktop Manager，位在定義「配置儲存庫」的檔案 (位於 `/etc/opt/SUNWapcmg/`) 中。
- Configuration Agent，位在 `/etc/apoc/policymgr.properties` 檔案中。
- Desktop Manager CLI，位在 `$HOME/pgtool.properties` 檔案中，但此 CLI 僅支援純 LDAP 儲存庫。

您可以在參數中加註前綴詞，以指出其所適用的儲存庫提供者。每個提供者都會優先考量具有前綴詞的參數。若未定義這類參數，便會使用不具前綴詞的參數。

**表 A–1** 前綴詞

| 前綴詞值 | 儲存庫提供者 |
|---|---|
| ORGANIZATION_ | 組織樹狀結構 |
| DOMAIN_ | [網域樹] 標籤 |
| PROFILE_ | 設定檔 |
| ASSIGNMENT_ | 指定 |
| LDAP_META_CONF_ | 若爲 LDAP 儲存庫時的對映資料 |

**表 A–2** 參數

| 名稱 | 說明 | 可能的值 | 預設值 |
|---|---|---|---|
| PROVIDER_URL | 指定儲存庫連線的 URL。若連線至第一個儲存庫失敗，可利用 URL 清單指定備用的儲存庫。 | 內含一或多個以空格分隔之 URL 的清單：每一個 URL 皆會採用下列其中一種格式：`ldap://<host>:<port>/<baseDN>`、`ldaps://<host>:<port>/<baseDN>`、`file://<filepath>`、`http://<host>:<port>/<filepath>`、`https://<host>:<port>/<filepath>`。 | 無，必要參數 |
| SECURITY_PRINCIPAL | 儲存庫連線的使用者名稱。 | 有權讀取及搜尋儲存庫之使用者的使用者名稱；若爲匿名連線，則無任何值。 | 無，匿名連線 |
| SECURITY_CREDENTIALS | SECURITY_PRINCIPAL 中所定義的使用者密碼。 | 雜亂或明文的密碼。 | 無 |
| SECURITY_CREDENTIALS_ENCODING | 指示 SECURITY_PRINCIPAL 中所定義的密碼是否爲雜亂密碼。警告：密碼的雜亂處理只會在密碼上加遮罩，不屬於任何安全加密類型。 | 若爲雜亂密碼，即爲「scramble」(精靈會在產生配置資料時自動完成此作業)。密碼若以明文顯示，即爲「none」；如需編輯密碼，請使用此值。 | 「none」 |
| MAX_SEARCH_RESULT | 搜尋任一儲存庫後所得到的結果數上限。備註：前綴詞方案不適用於此參數。 | 正數，0 表示無限制。 | 100 |

下列參數僅適用於 LDAP 儲存庫。

**表 A–3** LDAP 特有的參數

| 名稱 | 說明 | 可能的值 | 預設值 |
|---|---|---|---|
| AuthDn | 第一次存取 LDAP 儲存庫時所用之使用者的完全合格網域名稱，其目的在擷取 SECURITY_PRINCIPAL 中所定義的使用者。 | 有權讀取及搜尋儲存庫之使用者的使用者名稱；若爲匿名連線，則無任何值。 | 無，匿名存取 |

表 A–3 LDAP特有的參數 *(續)*

| 名稱 | 說明 | 可能的值 | 預設值 |
|---|---|---|---|
| Password | AuthDN 的密碼。 | 雜亂或明文的密碼。 | 無 |
| Password_ENCODING | 指出「密碼」中所定義的密碼是否爲雜亂密碼。警告：密碼的雜亂處理只會在密碼上加遮罩，不屬於任何安全加密類型。 | 若爲雜亂密碼，即爲「scramble」(精靈會在產生配置資料時自動完成此作業)。密碼若以明文顯示，即爲「none」；如需編輯密碼，請使用此值。 | 「none」 |
| Connect Timeout | 連線建立逾時 (以秒爲單位)。 | 正數，0 表示無時間限制。 | 1 |

**範例 A–1** 混合型後端範例

混合型後端範例，其中關於主機和使用者的資訊會從現有的 LDAP 儲存庫取得，並將設定檔及其指定儲存在檔案系統上。

```
#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
```

◆ ◆ ◆ 附錄 B

# 將 OpenLDAP 和 Active Directory 與 Desktop Manager 一起使用

## 將 OpenLDAP 伺服器與 Desktop Manager 一起使用

若要使用 OpenLDAP 伺服器做爲 Desktop Manager 資料的儲存庫，則必須延伸伺服器模式，以便具有儲存配置資料所用的物件類別與屬性。您可以在 `/usr/share/webconsole/apoc/deploy` 目錄中找到名爲 `apoc.schema` 的自訂模式檔。

必須將該檔案複製到 OpenLDAP 配置目錄 (`/etc/openldap`) 的子目錄 `schema` 中，並透過在位於該目錄的 `slapd.conf` 檔案中包含該檔案將其增加到 OpenLDAP 模式中。方法是將 `include /etc/openldap/schema/apoc.schema` 一行插入檔案中之模式包含序列的尾端。如需有關延伸 OpenLDAP 伺服器模式的更多資訊，請參閱伺服器的使用手冊。

只要延伸 OpenLDAP 伺服器模式，便可使用 Desktop Manager 所提供的「增加配置儲存庫」精靈完成其餘配置。

---

**備註 –** Desktop Manager Agent 將透過提供要從其獲得資料之使用者的 DN，但不提供密碼，匿名嘗試並連線至 OpenLDAP 伺服器。某些 OpenLDAP 伺服器的發行版本預設會停用此匿名認證模式，因此必須在 `slapd.conf` 檔案 (位於 OpenLDAP 配置目錄 `/etc/openldap`) 中所定義的一般伺服器參數中，增加 `allow bind_anon_cred` 一行，以啓動匿名認證模式。如需有關該參數的更多資訊，請參閱該伺服器的使用手冊。

---

## 將 Active Directory 伺服器與 Desktop Manager 一起使用

若要使用 Active Directory 伺服器做爲 Desktop Manager 資料的儲存庫，則必須延伸伺服器模式，以便具有儲存配置資訊所用的物件類別與屬性。您可以在 `/usr/share/webconsole/apoc/deploy` 目錄中找到名爲 `apoc-ad.ldf` 的延伸檔案。

必須將 `apoc-ad.ldf` 檔案匯入到 Active Directory 模式中，請使用以下步驟：

1. 啓用模式延伸。請參閱 Active Directory 說明文件，以取得有關如何執行該作業的更多資訊。

2. 從指令提示符號處執行以下指令：**ldifde -i -c "DC=Sun,DC=COM" <*BaseDN*> -f apoc-ad-registry.ldf** 。

---

**備註** – 請使用 Active Directory 基底 DN 替代 <*BaseDN*>。

---

只要延伸 Active Directory 伺服器模式，便可使用 Desktop Manager 所提供的「增加配置儲存庫」精靈完成其餘配置。

當「增加配置儲存庫」精靈提示您出示 LDAP 憑證時，請提供有權讀取此樹狀結構之使用者的完整網域名稱及密碼。此使用者可以是無法將 Active Directory 用於其他目的的使用者。請參閱 Active Directory 說明文件，以取得有關如何設定此類使用者的更多資訊。此外，Active Directory 的網域名稱對執行 Desktop Manager 的機器必須是已知的。方法是在該機器的 `/etc/hosts` 檔案中增加一行，將 Active Directory 伺服器的 IP 位址對映到其網域名稱。

若要從桌面主機擷取配置資料，則該主機亦須知道 Active Directory 的網域名稱。下列兩種方法可用於認證桌面使用者：匿名認證與使用 GSSAPI 認證。

- 若要使用匿名連線認證，則必須將 Active Directory 伺服器配置爲授與每個人讀取權限。請參閱 Active Directory 說明文件，以取得有關如何執行該作業的更多資訊。
- 若要使用 GSSAPI 進行認證，使用者必須通過 Active Directory 的認證，而其使用者憑證亦須能夠在系統上使用。只要在系統上配置 Kerberos 認證即可達成此目的，因爲 Kerberos 認證會在登入時產生這些憑證。如需有關如何執行此作業的更多資訊，請參閱所用系統的管理指南。

◆ ◆ ◆ 附錄 C

# 組織對映

## 組織對映

若要定義 LDAP 項目和 Desktop Manager 元素的對映，必須編輯 `Organization` 對映檔案。必須爲各種鍵提供符合 LDAP 系統訊息庫配置的值。

使用者元素除可由全部元素共用的物件類別加以識別外，也可由整個儲存庫中具有唯一值的屬性進行識別。可提供顯示名稱格式以決定使用者在管理應用程式中的顯示方式，或者如果組織中的使用者項目使用容器項目，則亦可定義該類項目。鍵名稱及其預設值包括：

```
# 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
```

角色元素可由其所用之可能物件類別清單以及相對應的命名屬性進行識別。這些清單所用的格式包括 `<item1>、<item2>、...、<itemN>`，必須相互一致。即，各清單必須含有相同的項目數，且第 n 個物件類別必須與第 n 個命名屬性配合使用。兩個鍵定義角色與使用者之間的關係，以及角色與主機之間的關係。*VirtualMemberAttribute* 鍵必須從使用者項目或主機項目指定可查詢其值的屬性。該鍵還必須包含項目所從屬角色的完整 DN。*MemberAttribute* 鍵必須指定使用者項目或主機項目中的屬性做爲搜尋篩選器。該鍵還必須包含使用者或主機所從屬角色的完整 DN。*VirtualMemberAttribute* 鍵可以是一種服務虛擬屬性，而 *MemberAttribute* 鍵則必須是可在篩選器中使用的實際屬性。鍵名稱及其預設值包括：

```
# 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
```

識別組織元素的方法與識別角色類似，均是利用格式相同的物件類別清單以及相應的命名屬性清單。鍵名稱及其預設值包括：

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

識別網域元素的方法與識別組織元素類似。鍵名稱及其預設值包括：

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

識別主機元素的方法與識別使用者元素類似。鍵名稱及其預設值包括：

```
# 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
```