



## 銀河麒麟高級伺服器操作系統 V10

---

系統管理員手冊

麒麟軟體有限公司

2024 年 01 月

## 目 錄

麒麟軟體有限公司（簡稱“麒麟軟體”） .....	1
銀河麒麟最終用戶使用許可協議 .....	4
銀河麒麟操作系統隱私政策聲明 .....	9
特別提示說明 .....	17
<b>第一章 基本系統配置 .....</b>	<b>18</b>
1.1. 系統地區和鍵盤配置 .....	18
1.1.1. 配置系統地區 .....	18
1.1.2. 配置鍵盤佈局 .....	19
1.1.3. 其他資源 .....	20
1.2. 網路訪問配置 .....	21
1.2.1. 動態網路配置 .....	21
1.2.2. 靜態網路配置 .....	21
1.2.3. 配置 DNS .....	21
1.3. 日期和時間配置 .....	21
1.3.1. <i>timedatectl</i> 工具使用說明 .....	22
1.3.2. <i>date</i> 工具使用說明 .....	24
1.3.3. <i>hwclock</i> 工具使用說明 .....	26
1.4. 用戶配置 .....	27
1.5. Kdump 機制 .....	28
1.5.1. <i>Kdump</i> 命令行配置 .....	29
1.6. 獲取特權 .....	31
1.6.1. <i>su</i> 命令工具 .....	31
1.6.2. <i>sudo</i> 命令工具 .....	31
<b>第二章 基本開發環境 .....</b>	<b>33</b>
2.1. Qt-5.14.2 .....	33

2.2. GCC-7.3 .....	33
2.3. GDB-9.2 .....	33
2.4. Python3-3.7.9 .....	35
2.5. Openjdk-1.8.0 .....	36
<b>第三章 常用圖形化工具 .....</b>	<b>37</b>
3.1. 刻錄工具 .....	37
3.2. 磁片 .....	37
3.2.1. 磁片管理 .....	37
3.2.2. 磁片管理工具使用 .....	39
3.3. 遠程桌面 .....	44
3.3.1. VNC 查看器 .....	44
3.3.2. 遠程查看程式 SSH .....	45
3.4. Cockpit 遠程管理 .....	48
3.4.1. Cockpit .....	48
3.4.2. 啟動和查看 Cockpit 服務 .....	48
3.4.3. Cockpit Web 控制臺 .....	49
3.5. 系統日誌 .....	56
<b>第四章 安裝和管理軟體 .....</b>	<b>58</b>
4.1. 檢查和升級軟體包 .....	58
4.1.1. 軟體包升級檢查 .....	58
4.1.2. 升級軟體包 .....	58
4.1.3. 利用系統光碟與 dnf 離線升級系統 .....	60
4.2. 管理軟體包 .....	61
4.2.1. 檢索軟體包 .....	61
4.2.2. 安裝包列表 .....	61
4.2.3. 顯示軟體包資訊 .....	62
4.2.4. 安裝軟體包 .....	63

4.2.5. 下載軟體包 .....	63
4.2.6. 刪除軟體包 .....	63
4.3. 管理軟體包組 .....	64
4.3.1. 軟體包組列表 .....	64
4.3.2. 安裝軟體包組 .....	65
4.3.3. 刪除軟體包組 .....	66
4.4. 軟體包操作記錄管理 .....	66
4.4.1. 查看操作 .....	66
4.4.2. 審查操作 .....	68
4.4.3. 恢復與重複操作 .....	69
<b>第五章 基礎服務 .....</b>	<b>69</b>
5.1. 使用 <code>systemd</code> 管理系統服務 .....	69
5.1.1. <code>Systemd</code> 介紹 .....	69
5.1.2. 管理系統服務 .....	74
5.1.3. 管理目標 .....	80
5.1.4. 在遠程機器上使用 <code>systemd</code> .....	83
5.1.5. 創建和修改 <code>systemd</code> 單元檔 .....	84
5.2. OpenSSH .....	98
5.2.1. SSH 協議 .....	98
5.2.2. SSH 連接的事件序列 .....	101
5.2.3. 配置 OpenSSH .....	104
5.2.4. 不只是一個安全的 <code>Shell</code> .....	115
5.3. TigerVNC .....	118
5.3.1. VNC 服務端 .....	118
5.3.2. 共用一個已存在的桌面 .....	122
5.3.3. VNC 查看器 .....	123
<b>第六章 伺服器 .....</b>	<b>125</b>

6.1. Web 伺服器 .....	125
6.1.1. Apache HTTP 伺服器 .....	125
6.2. 目錄伺服器 .....	141
6.2.1. OpenLDAP .....	141
6.2.2. 安裝 OpenLDAP 組件 .....	144
6.2.3. 配置 OpenLDAP 伺服器 .....	148
6.2.4. 使用 LDAP 應用的 SELinux 策略 .....	161
6.2.5. 運行 OpenLDAP 服務 .....	162
6.2.6. 配置系統使用 OpenLDAP 作為驗證 .....	163
6.3. 檔和列印伺服器 .....	165
6.3.1. Samba .....	165
6.3.2. FTP .....	180
6.3.3. 列印設置 .....	186
6.4. 使用 chrony 套件配置 NTP .....	187
6.4.1. chrony 套件介紹 .....	187
6.4.2. 理解 CHRONY 及其配置 .....	189
6.4.3. 使用 chrony .....	194
6.4.4. 為不同的環境設置 chrony .....	199
6.4.5. 使用 chronyc .....	201
6.5. 配置 NTP 使用 NTPD .....	202
6.5.1. NTP 介紹 .....	202
6.5.2. NTP 分層 .....	202
6.5.3. 理解 NTP .....	203
6.5.4. 理解 drift 檔 .....	204
6.5.5. UTC, TIMEZONES 和 DST .....	204
6.5.6. NTP 身份驗證選項 .....	204
6.5.7. 在虛擬機中管理時間 .....	204
6.5.8. 理解閏秒 .....	204

6.5.9. 理解 <i>ntpd</i> 配置檔 .....	205
6.5.10. 理解 <i>ntpd</i> 的 <i>sysconfig</i> 檔 .....	207
6.5.11. 禁止 <i>chrony</i> .....	207
6.5.12. 檢查 <i>NTP</i> 守護進程是否安裝 .....	207
6.5.13. <i>ntpd</i> 的安裝 .....	208
6.5.14. 檢查 <i>ntp</i> 的狀態 .....	208
6.5.15. 配置防火牆允許 <i>ntp</i> 包進入 .....	208
6.5.16. 配置 <i>ntpdate</i> 伺服器 .....	209
6.5.17. 配置 <i>ntp</i> .....	210
6.5.18. 配置硬體時鐘更新 .....	215
6.5.19. 配置時鐘源 .....	216
6.6. 使用 <i>ptp4l</i> 配置 <i>PTP</i> .....	216
6.6.1. <i>PTP</i> 介紹 .....	216
6.6.2. 使用 <i>PTP</i> .....	217
6.6.3. 和多個介面使用 <i>PTP</i> .....	219
6.6.4. 指定一個配置檔 .....	220
6.6.5. 使用 <i>PTP</i> 管理客戶端 .....	220
6.6.6. 同步時鐘 .....	221
6.6.7. 驗證時間同步 .....	222
6.6.8. 使用 <i>NTP</i> 服務 <i>PTP</i> 時間 .....	223
6.6.9. 使用 <i>PTP</i> 服務 <i>NTP</i> 時間 .....	224
6.6.10. 使用 <i>timemaster</i> 同步 <i>PTP</i> 或 <i>NTP</i> 時間 .....	225
6.6.11. 提高準確性 .....	228
<b>第七章 監控和自動化 .....</b>	<b>230</b>
7.1. 系統監控工具 .....	230
7.1.1. 查看系統進程 .....	230
7.1.2. 查看記憶體使用情況 .....	234
7.1.3. 查看 <i>CPU</i> 使用 .....	235

7.1.4. 查看硬體資訊 .....	241
7.1.5. 檢查硬體錯誤 .....	242
7.1.6. 使用 <i>Net-SNMP</i> 監控性能 .....	243
7.2. 查看和管理日誌檔 .....	255
7.2.1. 日誌檔的位置 .....	256
7.2.2. <i>Rsyslog</i> 的基本配置 .....	257
7.2.3. 使用新的配置格式 .....	270
7.2.4. 使用 <i>Rsyslog</i> 佇列 .....	273
7.2.5. 在日誌伺服器上配置 <i>rsyslog</i> .....	283
7.2.6. 使用 <i>Rsyslog</i> 模組 .....	285
7.2.7. <i>Syslogd</i> 服務和日誌的交互 .....	288
7.3. <i>Syslogd</i> 日誌結構 .....	290
7.3.1. 從日誌中導入數據 .....	291
7.3.2. 過濾結構化消息 .....	293
7.3.3. 解析 <i>JSON</i> .....	293
7.3.4. 向 <i>MongoDB</i> 中存儲消息 .....	294
7.4. 調試 <i>Rsyslog</i> .....	294
7.5. 使用日誌 .....	295
7.5.1. 查看日誌檔 .....	296
7.5.2. 訪問控制 .....	296
7.5.3. 使用 <i>Live view</i> .....	297
7.5.4. 過濾消息 .....	297
7.5.5. 使能持續存儲 .....	300
7.6. 自動化系統任務 .....	301
7.6.1. <i>Cron</i> 和 <i>Anacron</i> .....	302
7.6.2. 安裝 <i>Cron</i> 和 <i>Anacron</i> .....	302
7.6.3. 運行 <i>Crond</i> 服務 .....	303
7.6.4. 配置 <i>Anacron</i> 任務 .....	304

7.6.5. 配置 Cron 任務 .....	308
7.6.6. 控制對 Cron 的訪問 .....	310
7.6.7. Cron 任務的黑白名單 .....	311
7.6.8. At 和 Batch .....	312
<b>第八章 系統安全 .....</b>	<b>318</b>
8.1. 安全基礎服務 .....	318
8.1.1. 防火牆 .....	318
8.1.2. 審計管理(audit) .....	334
8.2. 安全增強組件 .....	345
8.2.1. KYSEC 安全機制 .....	345
8.2.2. 數據隔離保護機制 .....	348
8.2.3. 強制訪問控制 .....	351
8.2.4. 三權分立機制 .....	357
8.2.5. 核外安全功能及配置 .....	363
8.3. 麒麟安全管理工具-安全中心 .....	370
8.4. 麒麟檔保護箱 .....	371
<b>第九章 FAQ .....</b>	<b>372</b>
9.1. 版本查詢方法 .....	372
9.2. 字體安裝方法 .....	372
9.3. 詳細包資訊查詢 .....	373
9.4. 檢查包是否被篡改 .....	374



## 麒麟軟體有限公司（簡稱“麒麟軟體”）

為順應產業發展趨勢、滿足國家戰略需求、保障國家網路空間安全、發揮中央企業在國家關鍵資訊基礎設施建設中主力軍作用，中國電子資訊產業集團有限公司（簡稱“中國電子”）於 2019 年 12 月將旗下天津麒麟資訊技術有限公司和中標軟體有限公司強強整合，成立麒麟軟體有限公司（簡稱“麒麟軟體”），打造中國操作系統核心力量。

麒麟軟體主要面向通用和專用領域打造安全創新操作系統產品和相應解決方案，以安全可靠操作系統技術為核心，現已形成銀河麒麟伺服器操作系統、桌面操作系統、嵌入式操作系統、麒麟雲、操作系統增值產品為代表的產品線。麒麟操作系統能全面支持飛騰、鯤鵬、龍芯等六款主流國產 CPU，在安全性、穩定性、易用性和系統整體性能等方面遠超國內同類產品，實現國產操作系統的跨越式發展。目前，公司旗下產品已全面應用於黨政、金融、交通、通信、能源、教育等重點行業，服務用戶覆蓋所有的中央部委、政府機關、地市黨委。根據賽迪顧問統計，麒麟軟體旗下操作系統產品，連續 12 年位列中國 Linux 市場佔有率第一名。

麒麟軟體注重核心技術創新，2018 年榮獲“國家科技進步一等獎”，2020 年發佈的銀河麒麟操作系統 V10 被國資委評為“2020 年度央企十大國之重器”，相關新聞入選中央廣播電視總臺“2020 年度國內十大科技新聞”。麒麟軟體榮獲“中國電力科學技術進步獎一等獎”、“中國品牌日電子資訊行業國貨新品”等國家級、省部級和行業獎項 400 餘個，並被授予“國家規劃佈局內重點軟體企業”、“國家高技術產業化示範工程”、“科改示範行動企業”、“國有重點

企業管理標杆創建行動標杆企業”等稱號。通過 CMMI5 級評估，現有博士後工作站、省部級企業技術中心、省部級基礎軟體工程中心等，先後申請專利 551 項，其中授權專利 214 項，登記軟體著作權 553 項，主持和參與起草國家、行業、聯盟技術標準 60 餘項。

麒麟軟體在北京、天津、上海、長沙、廣州、深圳、太原、鄭州、武漢、南京、南昌、濟南、南寧、成都、瀋陽、廈門等地設有分支機構，服務網點遍佈全國 31 個省會城市和 2 個計畫單列市。

麒麟軟體高度重視生態體系建設，與眾多軟體廠商、集成商建立長期合作夥伴關係，建設完整的自主創新生態鏈，為國家網信領域安全創新提供有利支撐。截止 2024 年 1 月 16 日，麒麟軟體已與 19300 多家廠商建立合作，完成超 422 萬項軟體認證和適配，生態適配官網累計註冊用戶數超 5.9 萬+人。

麒麟軟體積極貫徹人才是第一資源的理念，以麒麟軟體教育發展中心為組織平臺，聯合政產學研各方力量，探索中國特色的網信人才培養模式，目前已形成了源自麒麟軟體核心技術的“5 序”培訓認證體系、課件體系、教材體系、師資體系、平臺體系，並與工信部教育與考試中心聯合推出“百城百萬”操作系統培訓專項行動，持續為我國培養各類操作系統專業人才。

在開源建設方面，麒麟軟體正式發佈中國首個桌面操作系統根社區 openKylin，通過構建自有可靠的開源軟體供應鏈，具備自主可持續發展能力，持續貢獻主流上游開源專案，攜手十餘家產業同仁共建 openKylin，力爭成為具有國際影響力的頂級開源社區。此外，麒麟軟體在 OpenStack 社區貢獻位列國內第一、全球第三；作為 openEuler 開源社區發起者，以 Maintainer 身份承

擔 80 個專案，除華為公司外貢獻第一；主導開發優麒麟開源操作系統，全球累計下載量數千萬次，活躍愛好者和開發者數十萬人。

## 銀河麒麟最終用戶使用許可協議

尊敬的銀河麒麟操作系統及相關產品用戶（以下稱“您”或“貴機構”）：

首先感謝您選用由麒麟軟體有限公司開發並製作發行的銀河麒麟操作系統軟體產品。

請在打開本軟體介質包之前，仔細閱讀本協議條款、提供的所有補充許可條款（統稱“協議”）及銀河麒麟操作系統隱私政策聲明。一旦您打開本軟體介質包，即表明您已接受本協議的條款，本協議將立即生效，對您和本公司雙方具有法律約束力。

### 1. 使用許可

按照已為之支付費用的用戶數目及電腦硬體類型，麒麟軟體有限公司（下稱“麒麟軟體”）向您授予非排他、不可轉讓的許可，僅允許內部使用由麒麟軟體提供的隨附軟體和文檔以及任何錯誤糾正（統稱“本軟體”）。除非另有明確約定，您無權向任何第三方進行分許可。

#### – 教育機構使用許可

在遵守本協議的條款和條件的情況下，如果貴機構是教育機構，麒麟軟體給予貴機構非獨占、不可轉讓的許可，允許貴機構僅在內部使用隨附的未經修改的二進位格式的軟體。此處的“在內部使用”是指由在貴機構入學的學生、貴機構教員和員工使用軟體。

#### – 字型軟體使用

軟體中包含生成字體樣式的軟體（“字型軟體”）。貴機構不可從軟體中分離字型軟體。貴機構不可改動字型軟體，以新增此等字型軟體被作為軟體的一部分

交付予貴機構時所不具備的任何功能。貴機構不可將字型軟體嵌入作為商業產品提供以換取收費或其他報酬的檔。

## 2. 限制

本軟體受到版權（著作權）法、商標法和其他法律及國際知識產權公約的保護。麒麟軟體和/或其許可方保留對本軟體的所有權及所有相關的知識產權。對於麒麟軟體或其許可方的任何商標、服務標記、標識或商號的任何權利、所有權或利益，本協議均不作任何授權。

### 關於複製、修改及分發

如果在所有複製品中維持本協議書不變，您可以且必須根據《GNU GPL-GNU 通用公共許可證》複製、修改及分發銀河麒麟操作系統軟體產品中遵守《GNU GPL-GNU 通用公共許可證》協議的軟體，其他不遵守《GNU GPL-GNU 通用公共許可證》協議的銀河麒麟操作系統軟體產品必須根據符合相關法律之其他許可協議進行複製、修改及分發，但任何以銀河麒麟操作系統軟體產品為基礎的衍生發行版未經麒麟軟體有限公司的書面授權不能使用任何麒麟軟體有限公司的商標或其他任何標誌。

特別注意：該複製、修改及分發不包括本產品中包含的任何不適用《GNU GPL-GNU 通用公共許可證》的軟體，如銀河麒麟操作系統軟體產品中包含的輸入法軟體、字形檔軟體、第三方應用軟體等。除非適用法律禁止實施，否則您不得對上述軟體進行複製、修改（包括反編譯或反向工程）、分發。

## 3. 有限擔保

麒麟軟體向您擔保，自購買或其他合法取得之日起九十（90）天內（以收

據副本為憑證），本軟體的存儲介質（如果有的話）在正常使用的情況下無材料和工藝方面的缺陷。除上述內容外，本軟體按“原樣”提供。在本有限擔保項下，您的所有補償及麒麟軟體的全部責任為由麒麟軟體選擇更換本軟體介質或退還本軟體的購買費用。

#### **4. 擔保的免責聲明**

除非在本協議中有明確規定，否則對於任何明示或默示的條件、陳述及擔保，包括對適銷性、對特定用途的適用性或非侵權性的任何默示的擔保，均不予負責，但上述免責聲明被認定為法律上無效的情況除外。

#### **5. 責任限制**

在法律允許範圍內，無論在何種情況下，無論採用何種有關責任的理論，無論因何種方式導致，對於因使用或無法使用本軟體引起的或與之相關的任何收益損失、利潤或數據損失，或者對於特殊的、間接的、後果性的、偶發的或懲罰性的損害賠償，麒麟軟體或其許可方均不承擔任何責任（即使麒麟軟體已被告知可能出現上述損害賠償）。根據本協議，在任何情況下，無論是在合同、侵權行為（包括過失）方面，還是在其他方面，麒麟軟體對您的責任將不超過您就本軟體所支付的金額。即使上述擔保未能達到其基本目的，上文所述的限制仍然適用。

#### **6. 終止**

本協議在終止之前有效。您可以隨時終止本協議，但必須同時銷毀本軟體的全部正本和副本。如果您未遵守本協議的任何規定，則本協議將不經麒麟軟體發出通知立即終止。終止時，您必須銷毀本軟體的全部正本和副本，並且需承擔因未遵守本協議而導致的法律責任。

## 7. 法律適用

與本協議相關的任何爭議解決（包括但不限於訴訟、仲裁等）均適用中華人民共和國法律。任何其他國家和地區的法律規則不予適用。

## 8. 可分割性

如果本協議中有任何規定被認定為無法執行，則刪除相應規定，本協議仍然有效，除非該刪除會防礙各方根本目的的實現（在這種情況下，本協議將立即終止）。

## 9. 完整性

本協議是您與麒麟軟體就其標的達成的完整協議。它取代此前或同期的所有和本協議不一致的口頭或書面往來資訊、建議、陳述和擔保，有關報價、訂單、回執或各方之間就本協議標的進行的其他往來通信中的任何衝突條款或附加條款，均以本協議為準。對本協議的任何修改均無約束力，除非通過書面進行修改並由每一方的授權代表簽字。

## 10. 商標和標識

貴機構承認並與麒麟軟體有著以下共識，即麒麟軟體擁有麒麟軟體、銀河麒麟商標，以及所有與麒麟軟體、銀河麒麟相關的商標、服務標記、標識及其他品牌標識（“麒麟軟體標記”）。貴機構對麒麟軟體標記的任何使用都應有利於麒麟軟體。

## 11. 源代碼

本軟體可能包含源代碼，其提供之唯一目的是在符合本協議條款之規定時供參考之用。源代碼不可再分發，除非在本協議中有明確規定。

## **12. 因侵權而終止**

如果本軟體成為或在任一方看來可能成為任何知識產權侵權索賠之標的，則任一方可立即終止本協議。

產品中本協議提供中英文兩種版本，以上任何內容如有歧義，以中文版本為準。



## 銀河麒麟操作系統隱私政策聲明

版本發佈日期：2019 年 12 月 18 日

版本生效日期：2019 年 12 月 18 日

尊敬的銀河麒麟操作系統用戶（以下簡稱“您”），銀河麒麟操作系統系列軟體產品是由麒麟軟體有限公司（以下簡稱“我們”或“麒麟軟體”）研製發行的，用於辦公或構建企業及政府的資訊化基礎設施。

麒麟軟體非常重視您的個人資訊和隱私保護，在您使用本產品的過程中，我們會按照《銀河麒麟操作系統隱私政策聲明》（以下簡稱“本聲明”）收集、存儲、使用您的個人資訊。為了保證對您的個人隱私資訊合法、合理、適度的收集、使用，並在安全、可控的情況下進行傳輸、存儲，我們制定了本聲明。我們將向您說明收集、保存和使用您的個人資訊的方式，以及您訪問、更正、刪除和保護這些資訊的方式。我們將會按照法律要求和業界成熟安全標準，為您的個人資訊提供相應的安全保護措施。如您點擊或勾選“同意”並確認提交，即視為您同意本隱私政策聲明，並同意我公司將按照本政策來收集、存儲和使用您的相關資訊。

本聲明將幫助您瞭解以下內容：

- 一、關於收集和使用涉及您的個人資訊
- 二、如何存儲和保護涉及您的個人資訊
- 三、如何管理您的個人資訊
- 四、關於第三方軟體的隱私說明
- 五、關於未成年人使用產品

## 六、本聲明如何更新

## 七、如何聯繫我們

### 一、如何收集和使用您的個人資訊

#### 1. 收集涉及您的個人資訊的情況

我們在您使用銀河麒麟操作系統產品過程中收集相關的資訊，主要為了向您提供更高質量、更易用的產品和更好的服務。

1) 銀河麒麟操作系統的產品授權許可機制，會根據您所使用電腦的網卡、固件和主板等資訊通過加密機制和轉換方法生成申請產品正式授權許可的機器碼；您將該機器碼發給麒麟軟體商務人員根據合同及相關協議可申請正式許可。該機器碼不包含您所使用電腦的網卡、固件和主板等設備具體資訊。

2) 銀河麒麟操作系統應用商店的伺服器端，會根據您所使用電腦的 CPU 類型資訊以及 IP 地址進行連接；實現您方便快捷使用應用商店。您所使用電腦的 IP 地址可能會記錄在應用商店的伺服器端系統的日誌中。

3) 銀河麒麟操作系統的升級更新，會根據您所使用電腦的 IP 地址進行連接；以便實現您確認是否更新升級系統。

4) 使用銀河麒麟操作系統產品過程中，因業務往來及技術服務等您提供的電子郵箱、電話、姓名等個人資訊。

5) 銀河麒麟操作系統可能提供生物識別相關功能，會存儲身份鑒別相關的資訊在您的機器。這部分資訊我們不收集和上傳伺服器。

以後銀河麒麟操作系統產品升級過程中，如新增涉及個人資訊收集部分，將及時更新本部分內容。

## 2. 使用涉及您的個人資訊的情況

我們嚴格遵守法律法規的規定及與用戶的約定，將收集的資訊用於以下用途。若我們超出以下用途使用您的資訊，我們將再次向您進行說明，並征得您的同意。

我們會將收集的資訊用於以下用途：

**產品功能：**主要涉及產品許可機制、應用商店使用、系統更新維護、生物識別等需要。

**安全保障：**為保障您使用銀河麒麟操作系統的安全，我們會利用相關資訊協助提升產品的安全性、可靠性和可持續服務。

**與您溝通：**我們會利用收集的資訊（例如您提供的電子郵件地址、電話等）直接與您溝通。例如，業務聯繫、技術支持或服務回訪。

**產品改進：**將收集的資訊用於改進產品當前的易用性、缺陷以及提升產品用戶體驗等。

為了遵從相關法律法規、部門規章、政府指令的相關要求。

## 3. 資訊的分享及對外提供

我們不會共用或轉讓您的個人資訊至第三方，但以下情況除外：

1) 獲取您的明確同意：經您事先同意，我們可能與第三方分享您的個人資訊；

2) 為實現外部處理的目的，我們可能會與關聯公司或其他第三方合作夥伴（第三方服務供應商、承包商、代理、應用開發者等）分享您的個人資訊，讓他們按照我們的說明、隱私政策以及其他相關的保密和安全措施來為我們處理上述資訊，並用於向您提供我們的服務，實現“如何收集和使用您的個人資訊”部分所

述目的。如我們與上述關聯公司或第三方分享您的資訊，我們將會採用加密、匿名化處理等手段保障您的資訊安全。

3) 我們不會對外公開披露所收集的個人資料，如必須公開披露時，我們會向您告知此次公開披露的目的、披露資訊的類型及可能涉及的敏感資訊，並征得您的明示同意。

4) 隨著我們業務的持續發展，我們有可能進行合併、收購、資產轉讓等交易，我們將告知相關情形，按照法律法規及不低於本聲明所要求的標準繼續保護或要求新的控制者繼續保護您的個人資料。

5) 我們可能基於法律要求或相關部門的執法要求披露您的個人資料。

如我們使用您的個人資料，超出了與收集時所聲稱的目的及具有直接或合理關聯的範圍，我們將在使用您的個人資料前，再次向您告知並征得您的明示同意。

根據相關法律法規以及國家標準，在以下情況下我們可能會收集、使用您的個人資料，征得授權同意的例外情況：

- 1) 與國家安全、國防安全等國家利益直接相關的；
- 2) 與公共安全、公共衛生、公眾知情等重大公共利益直接相關的；
- 3) 與犯罪偵查、起訴、審判和判決執行等直接相關的；
- 4) 出於維護您或其他個人的生命、財產等重大合法權益但又很難得到您本人同意的；
- 5) 所收集的個人資料是您自行向社會公眾公開的；
- 6) 從合法公開披露的資訊中收集的個人資料，如合法的新聞報導、政府資訊公開等管道；

- 7) 根據您要求簽訂和履行合約所必需的；
- 8) 用於維護所提供的產品或服務的安全穩定運行所必需的。如發現、處置產品或服務故障；
- 9) 出於公共利益開展統計或學術研究所必需，且其對外提供學術研究或描述的結果時，對結果中所包含的個人資訊進行去標識化處理的；
- 10) 法律法規規定的其他情形。

## 二、我們如何存儲和保護涉及您的個人資訊

### 1. 資訊存儲的地點

我們會按照法律法規規定，將在中國境內收集和產生的個人資訊存儲於中國境內。

### 2. 資訊存儲的期限

一般而言，我們僅為實現目的所必需的時間保留您的個人資訊。記錄在日誌中的資訊會按配置在一定期限保存及自動刪除。

當我們的產品或服務發生停止運營的情形時，我們將以通知、公告等形式通知您，在合理的期限內刪除您的個人資訊或進行匿名化處理，並立即停止收集個人資訊的活動。

### 3. 我們如何保護這些資訊

我們努力為用戶的資訊安全提供保障，以防止資訊的丟失、不當使用、未經授權訪問或披露。

我們將在合理的安全水準內使用各種安全保護措施以保障資訊的安全。例如，我們會使用加密技術（例如，SSL/TLS）、匿名化處理等手段來保護您的個人資

訊。

我們建立專門的管理制度、流程和組織以保障資訊的安全。例如，我們嚴格限制訪問資訊的人員範圍，要求他們遵守保密義務，並進行審計。

4.若發生個人資訊洩露等安全事件，我們會依法啟動應急預案，阻止安全事件擴大，並以推送通知、公告等形式告知您安全事件的情況、事件可能對您的影響以及我們將採取的補救措施。我們還將按照法律法規和監管部門要求，上報個人資訊安全事件的處置情況。

### **三、如何管理您的個人資訊**

如果擔心因使用銀河麒麟操作系統產品導致個人資訊的洩露，您可根據個人及業務需要考慮暫停或不使用涉及個人資訊的相關功能，如產品正式授權許可、應用商店、系統更新升級、生物識別等。

在使用銀河麒麟操作系統之上使用第三方軟體時，請注意個人隱私保護。

### **四、關於第三方軟體的隱私說明**

您在使用銀河麒麟操作系統之上安裝或使用第三方軟體時，第三方軟體的隱私保護和法律責任由第三方軟體自行負責。

您在使用銀河麒麟操作系統之上安裝或使用第三方軟體時，請您仔細閱讀和審查對應的隱私聲明或條款；注意個人隱私保護。

### **五、關於未成年人使用產品**

銀河麒麟操作系統系列產品僅供成年人使用，如果您是未成年人，則需要您的監護人同意您使用本產品並同意相關服務條款。父母和監護人也應採取適當的預防措施保護未成年人，包括監督其對銀河麒麟操作系統系列產品的使用。

## 六、本聲明如何更新

我們保留適時更新本聲明的權利，當本聲明發生變更時，我們會通過產品安裝過程或公司網站向您展示變更後的聲明，只有在獲取您的同意後，我們才會按照更新後的聲明收集、使用、存儲您的個人資料。

## 七、如何聯繫我們

如您對本聲明存在任何疑問，或任何相關的投訴、意見，請聯繫麒麟軟體客服熱線 400-089-1870、官方網站（[www.kylinos.cn](http://www.kylinos.cn)）以及麒麟軟體進行諮詢或反映。您可以通過發送郵件至 [market@kylinos.cn](mailto:market@kylinos.cn) 方式與我們聯繫。

受理您的問題後，我們會及時、妥善處理。一般情況下，我公司將在 15 個工作日內給予答復。

本聲明自更新之日起生效，同時提供中英文兩種版本，以上任何條款如有歧義，以中文版本為準。

最近更新日期：2024 年 11 月 28 日

麒麟軟體有限公司

地址：天津市濱海高新區塘沽海洋科技園信安創業廣場 3 號樓（300450）

北京市海澱區北四環西路 9 號銀穀大廈 20 層（100190）

長沙市開福區三一大道 156 號工美大廈 10 樓（410073）

電話：天津（022）58955650 北京（010）51659955 長沙（0731）88280170

傳真：天津（022）58955651 北京（010）62800607 長沙（0731）88280166

公司網站：[www.kylinos.cn](http://www.kylinos.cn)

電子郵件：[support@kylinos.cn](mailto:support@kylinos.cn)



## 特別提示說明

銀河麒麟高級伺服器操作系统 V10 同源支持飛騰、龍芯、申威、兆芯、海光、鯤鵬等自主 CPU 平臺。本手冊主要面向系統管理員及相關技術人員，如本手冊未能詳細描述之處，有需要請致電麒麟軟體有限公司技術服務部門。

 **重要：**

本手冊中命令、操作步驟等舉例僅供參考，命令執行的輸出資訊等在不同 CPU 平臺或因操作系统或組件的版本升級可能有少許差異；本手冊儘量加以說明。如有差異之處，請以銀河麒麟高級伺服器操作系统 V10 在具體 CPU 平臺上實際操作或輸出資訊為準。

## 第一章 基本系統配置

這部分涵蓋了基本的系統管理任務，如鍵盤配置、日期和時間配置、用戶和組群配置以及授權配置。

### 1.1. 系統地區和鍵盤配置

系統地區配置是指系統服務和用戶介面的語言環境配置。鍵盤佈局配置是指文本控制臺和圖形用戶介面的鍵盤佈局規則。這些設置可以通過修改 `/etc/locale.conf` 配置檔或使用 `localectl` 命令。此外，您可以在用戶圖形介面來執行任務，詳情請參考安裝手冊。

#### 1.1.1. 配置系統地區

系統地區配置檔為 `/etc/locale.conf`，在系統啟動時引導 `systemd` 守護進程。這個配置檔可以被每一個服務或者用戶繼承，單個服務或者用戶也可修改配置檔。例如語言為英語，地區為德國的 `/etc/locale` 檔的配置內容如下：

```
LANG=de_DE.UTF-8
LC_MESSAGES=C
```

`LC_MESSAGES` 選項決定了診斷消息的標準輸出文本格式。其他選項說明總結在表 1-1 在所示。

表 1-1 在 `/etc/locale.conf` 檔中可配置項

配置項	描述
LANG	提供系統時區的默認值
LC_COLLATE	定義該環境的排序和比較規則

LC_CTYPE	用於字元分類和字串處理，控制所有字元的處理方式，包括字元編碼，字元是單字節還是多位元組，如何列印等。 是最重要的一個環境變數
LC_NUMERIC	非貨幣的數字顯示格式
LC_TIME	時間和日期格式
LC_MESSAGES	提示資訊的語言

#### 1.1.1.1. 顯示當前配置

Localectl 命令可用於配置語言環境和鍵盤佈局。顯示當前配置，可使用如下命令：

```
#localectl status
```

#### 1.1.1.2. 顯示可用地區列表

顯示可用地區列表可使用如下命令：

```
#localectl list-locales | grep en_
```

#### 1.1.1.3. 配置地區

配置系統默認地區，需要以 root 用戶身份運行：

```
#localectl set-locale LANG=locale
```

用戶可以配置適合的地區標示符以代替 locale，可通過 localectl list-locales 檢索適合的地區。

### 1.1.2. 配置鍵盤佈局

鍵盤佈局配置是指文本控制臺和圖形用戶界面的鍵盤佈局規則。

### 1.1.2.1. 顯示當前配置

Localectl 命令可用於配置語言環境和鍵盤佈局。顯示當前配置，可使用如下命令：

```
#localectl status
```

### 1.1.2.2. 顯示可用鍵盤佈局列表

顯示可用鍵盤佈局列表可使用如下命令：

```
#localectl list-keymaps
```

### 1.1.2.3. 配置鍵盤

配置系統默認鍵盤佈局，需要以 root 用戶身份運行：

```
#localectl set-keymap {map}
```

用戶可以配置適合的鍵盤佈局標示符以代替 {map}，如“cz”，可通過 localectl list-keymaps 檢索適合的鍵盤佈局。該命令還可用於配置 X11 窗口的鍵盤佈局映射，但使用 --no-convert 參數的話則不生效。同樣也可用以下命令單獨配置 X11 窗口的鍵盤佈局：

```
#localectl set-x11-keymap cz
```

如果用戶希望 X11 窗口和命令行終端的鍵盤佈局不一樣，可以使用如下命令：

```
#localectl --no-convert set-x11-keymap cz
```

### 1.1.3. 其他資源

其他官方配置系統地區和鍵盤佈局的內容可以參考安裝手冊。同時還可參考

1.6 獲取特權章節和 5.1 章節。

## 1.2. 網路訪問配置

### 1.2.1. 動態網路配置

打開終端，以網口 eth0 為例：

```
#nmcli conn add connection.id eth0-dhcp type ether
ifname eth0 ipv4.method auto
```

其中“eth0-dhcp”為連接的名字，可以根據自己的需要命名方便記憶和操作的網口；“ifname eth0”為配置的網口，根據自己的設備情況按需調整。

### 1.2.2. 靜態網路配置

打開終端，以網口 eth0 為例：

```
#nmcli conn add connection.id eth0-static type ether
ifname eth0 ipv4.method manual ipv4.address
192.168.1.10/24 ipv4.gateway 192.168.1.254 ipv4.dns
192.168.1.254
```

其中“eth0-static”為連接的名字，可以根據自己的需要命名方便記憶和操作的網口；“ifname eth0”為配置的網口，根據設備情況按需調整；IP、子網掩碼、網關根據實際網路按需配置。

### 1.2.3. 配置 DNS

打開終端，編輯/etc/resolv.conf，設置 nameserver：

```
# Generated by NetworkManager
nameserver 10.1.10.1
```

## 1.3. 日期和時間配置

操作系統區分以下兩種時區：

- 即時時間（RTC），通常作為物理時鐘，它可以獨立於系統當前狀態計時，在主機關機情況下也可計時。
- 系統時間，是基於即時時間的由操作系統內核維護的軟體時間。等系統啟動內核初始化系統時間後，系統時間就獨立於即時時間自行計時。

系統時間通常還保持一套世界統一時間（UTC），用於轉換系統的不同時區，本地時間就是用戶所在時區的真實時間。

操作系統提供了三種命令行時間管理工具，`timedatectl`、`date` 和 `hwclock`。

以下將分別介紹各個工具的使用。

### 1.3.1. `timedatectl` 工具使用說明

#### 1.3.1.1. 顯示當前日期和時間

命令 `timedatectl` 可以顯示當前系統時間和機器的物理時間及其詳細資訊。

如下示例是未啟用 NTP 時鐘同步的系統時間：

```
#timedatectl
```

變更 `chrony` 或 `ntpd` 服務狀態不會主動通知 `timedatectl` 工具，如果想要更新服務的配置資訊，請執行以下命令：

```
#systemctl restart systemd-timedated.service
```

#### 1.3.1.2. 變更當前時間

以 `root` 用戶運行以下命令可以修改當前時間：

```
#timedatectl set-time HH: MM: SS
```

其中 `HH` 代表小時，`MM` 代表分鐘，`SS` 代表秒數，均需兩位表示。這個命令同樣可以更新系統時間和物理時間，效果類似於 `date --set` 和 `hwclock`

--systohc 命令。

系統默認時間配置基於 UTC，如果想基於本地時間來配置系統時間，需要以 root 用戶運行以下命令修改。

```
#timedatectl set-local-rtc boolean
```

如果基於本地時間，需要將 boolean 配置為 yes（或者 y, true, t 或者 1）。如果使用 UTC 時間，則要將 boolean 配置為 no（或者 n, false, f 或者 0）。系統默認 boolean 為 no。

#### 1.3.1.3. 變更當前日期

以 root 用戶運行以下命令可以修改當前日期：

```
#timedatectl set-time YYYY-MM-DD
```

其中 YYYY 代表年份，需 4 位數表示；MM 代表月份，需兩位數表示；DD 代表日期，需兩位表示。如果還需要配置時間，可以補充上時間參數，示例如下：

```
#timedatectl set-time '2020-02-17 23:26:00'
```

#### 1.3.1.4. 修改時區

執行以下命令可以顯示當前時區：

```
#timedatectl show
```

以 root 用戶運行以下命令可以修改當前時區，如修改為“上海”：

```
#timedatectl set-timezone Asia/Shanghai
```

顯示所有時區命令如下：

```
#timedatectl list-timezones
```

### 1.3.1.5. 同步系統與遠程伺服器時間

以 root 用戶運行以下命令可以啟用/禁用時間同步服務：

```
#timedatectl set-ntp boolean
```

啟用與禁用需要配置 **boolean** 值為 **yes** 或者 **no**。例如需要自動同步一個遠程時間伺服器，可以執行以下命令：

```
#timedatectl set-ntp yes
```

## 1.3.2. date 工具使用說明

### 1.3.2.1. 顯示當前日期和時間

命令 **date** 可以顯示當前系統時間、時區、日期等資訊。並可以通過參數 **--utc** 顯示當前時區時間。通過“**format**”標示符來輸出特定狀態。常用的 **format** 說明如下：

表 1-2 參數介紹

參數	描述
%H	以 HH 格式輸出當前小時
%M	以 MM 格式輸出當前分鐘
%S	以 SS 格式輸出當前秒數
%d	以 DD 格式輸出當前日期
%m	以 MM 格式輸出當前月份
%Y	以 YYYY 格式輸出當前年份
%Z	顯示時區制式，例如 C EST



%F	以 YYYY-MM-DD 格式輸出當前年月日，等價於參數 %Y-%m-%d
%T	以 HH:MM:SS 格式輸出當前時間，等價於參數 %H:%M:%S

示例如下：

#### **#date**

```
Mon Feb 17 17:30:24 CEST 2020
```

#### **#date --utc**

```
Mon Feb 17 15:30:34 UTC 2020
```

#### **#date+"%Y-%m-%d%H%M"**

```
2020-02-17 17:30
```

### 1.3.2.2. 變更當前時間

以 root 用戶運行以下命令可以修改當前時間：

```
#date --set HH: MM: SS
```

其中 HH 代表小時，MM 代表分鐘，SS 代表秒數，均需兩位表示。這個命令同樣可以更新系統時間和物理時間，效果類似於 `hwclock -systohc` 命令。

系統默認時間配置基於本地時間，如果想基於 UTC 時間來配置系統時間，需要以 root 用戶運行以下命令修改。

```
#date --set HH: MM: SS --utc
```

### 1.3.2.3. 變更當前日期

以 root 用戶運行以下命令可以修改當前日期：

```
#date --set YYYY-MM-DD
```

其中 YYYY 代表年份，需 4 位數表示；MM 代表月份，需兩位數表示；DD 代表日期，需兩位表示。如果還需要配置時間，可以補充上時間參數，示例如下：

```
#date --set 2020-02-20 23:26:00
```

### 1.3.3. hwclock 工具使用說明

#### 1.3.3.1. 顯示當前日期和時間

命令 `hwclock` 可以顯示當前系統時間、時區、日期等資訊。並可以通過參數 `--utc` 或 `--localtime` 顯示當前 UTC 時區時間和本地時間。示例如下：

```
#hwclock
Thur 13 Feb 2020 04:23:46 PM CEST -0.329272 seconds
```

#### 1.3.3.2. 變更當前日期和時間

以 `root` 用戶運行以下命令可以修改當前時間：

```
#hwclock --set --date "dd mmm yyyy HH:MM"
```

其中 `dd` 代表日期 `HH` 代表小時，`MM` 代表分鐘，`SS` 代表秒數，均需兩位表示。`Mmm` 代表月份，以月份英文三位字母簡寫表示，`yyyy` 代表年份，以四位數字表示。這個命令通過參數 `--utc` 或 `--localtime` 區分配置當前 UTC 時區時間和本地時間

基於 UTC 時間來配置系統時間，需要以 `root` 用戶運行以下命令修改，示例如下。

```
#hwclock --set --date "20 Feb 2020 21:17" --utc
```

#### 1.3.3.3. 同步系統與遠程伺服器時間

以 `root` 用戶運行以下命令同步遠程時間：

```
#hwclock --systohc
```

## 1.4. 用戶配置

用戶配置可以對用戶進行創建與管理，點擊**開始->控制面板->用戶帳戶->創建一個新帳戶**可以新建用戶，並且可以配置頭像、是否自動登錄以及用戶類型，輸入用戶名以及密碼，點擊**創建用戶**即可創建成功。



圖 1-1 創建帳戶

用戶創建成功後，可以對帳戶密碼以及頭像進行重新更改，點擊“**更換密碼**”以及“**更換頭像**”即可更改，點擊“**刪除用戶**”即可將用戶刪除。



圖 1-2 用戶帳戶

## 1.5. Kdump 機制

Kdump 是基於 kexec 的內核崩潰轉儲機制，在系統崩潰、死鎖或死機時用來轉儲記憶體運行參數的一個工具和服務，用來捕獲內核崩潰的時候產生的 crash dump。Kdump 是迄今為止最可靠的內核轉存機制，最大的優點在於崩潰轉儲數據可從一個新啟動內核的上下文中獲取，而不是從已經崩潰內核的上下文。

Kdump 需要兩個不同目的的內核，生產內核和捕獲內核。生產內核是捕獲內核服務的對像：如果系統一旦崩潰，那麼正常的內核就沒有辦法工作了，在這個時候將由 Kdump 產生一個用於捕獲當前運行資訊的內核，該內核會與相應的 ramdisk（虛擬記憶體盤：將記憶體模擬成硬碟的技術）一起組建一個微環境，

將此時的記憶體中的所有運行狀態和數據資訊收集到一個 `dump core` 檔中，一旦記憶體資訊收集完成，系統將會自動重啟。

Kdump 機制主要包括兩個組件：`kdump` 和 `kexec`。

`kdump` 使用 `kexec` 啟動到捕獲內核，以很小記憶體啟動以捕獲轉儲鏡像。生產內核保留了記憶體的一部分給捕獲內核啟動用。由於 `kdump` 利用 `kexec` 啟動捕獲內核，繞過了 BIOS，所以第一個內核的記憶體得以保留。這是內核崩潰轉儲的本質。

`kexec` 是一個快速啟動 kernel 的機制，它運行在某一正在運行的 kernel 中，啟動一個新的 kernel 而且不用重新經過 BIOS 就可以完成啟動。因為一般 BIOS 都會花費很長的時間，尤其是在大型並且同時連接許多外部設備的 Server 上的環境下，BIOS 會花費更多的時間。

`kexec` 包括 2 個組成部分：一是內核空間的系統調用 `kexec_load`，負責在生產內核啟動時將捕獲內核加載到指定地址。二是用戶空間的工具 `kexec-tools`，他將捕獲內核的地址傳遞給生產內核，從而在系統崩潰的時候能夠找到捕獲內核的地址並運行。

### 1.5.1. Kdump 命令行配置

#### 1.5.2.1. 安裝 Kdump 需要的軟體包

表 1-6 Kdump 所需的軟體包

軟體包名稱	軟體包說明
<code>kdump</code>	<code>kdump</code> 軟體包
<code>kexec-tools</code>	<code>kexec</code> 軟體包， <code>kdump</code> 用到的各種工具都在此包中

kernel-debuginfo	用來分析vmcore檔
kernel-debuginfo-common	kernel-debuginfo依賴包

使用Kdump服務，需先安裝這些工具包。安裝命令如下：

```
dnf install kdump kexec-tools kernel-debuginfo-common
kernel-debuginfo
```

#### 1.5.2.2.配置 grub

Kdump 的使用需要配置 kdump kernel 的記憶體區域。Kdump 要求操作系統正常使用的時候，不能使用 kdump kernel 所佔用的記憶體，配置這個需要修改/boot/grub/grub.conf 檔，修改用到的引導部分，加入 crashkernel。

Crashkernel 的格式如下：

```
crashkernel=nn[KMG]@ss[KMG]
```

其中 nn 表示要為 crashkernel 預留多少記憶體，ss 表示為 crashkernel 預留記憶體的起始位置。

修改完成並重啟後，可以通過 cat /proc/cmdline 查看 kernel 啟動配置選項，其中已經加入了 crashkernel 項。

#### 1.5.2.3.啟動和查看 Kdump 服務

查看 Kdump 服務命令如下：

```
#systemctl status kdump.service
```

如果 Kdump 服務未開啟，使用如下命令來啟動 kdump 服務：

```
#systemctl start kdump.service
```

## 1.6. 獲取特權

系統普通用戶的許可權有不同的限制，某些情況下普通需用需要執行管理員用戶許可權才能執行的命令，此時可以通過 `su` 或者 `sudo` 命令獲得管理員許可權特權。

### 1.6.1. su 命令工具

用戶使用 `su` 命令時，需要輸入 `root` 用戶密碼，驗證通過後可以獲取 `root` 的腳本環境。一旦通過 `su` 命令登入，這個用戶的所有操作均視為 `root` 用戶操作。由於 `su` 可以獲取 `root` 全部許可權，並因此獲取其他用戶的許可權，可能存在一定安全問題。因此可以通過管理員組群 `wheel` 來進行限制。以 `root` 用戶執行以下命令：

```
#usermod -G wheel username
```

當將用戶加入 `wheel` 組群後，可以限制只有這個組群的用戶可以使用 `su` 命令訪問。配置 `su` 的 `PAM` 可以編輯 `/etc/pam.d/su` 檔，通過添加刪除 `#` 字元來確認添加或刪除相應內容。

```
#auth required pam_wheel.so use_uid
```

上述內容表示管理員組群 `wheel` 內的用戶可以通過 `su` 訪問其他用戶。

### 1.6.2. sudo 命令工具

`sudo` 命令允許系統管理員讓普通用戶執行一些或者全部的 `root` 命令。當可信用戶執行 `sudo` 命令時，需要提供他們自己的用戶密碼，然後以 `root` 許可權執行命令。

基本的 `sudo` 命令如下：

```
#sudo command
```

`sudo` 命令有很大的彈性，只有在 `/etc/sudoers` 檔中被允許的用戶可以執行在他們自己的 `shell` 環境中執行 `sudo` 命令，而不是 `root` 的 `shell` 環境。這意味著在 7 系列中 `root` 的 `shell` 環境是被禁止訪問的。

配置 `sudo` 必須通過編輯 `/etc/sudoers` 檔，而且只有管理員用戶才可以修改它，必須使用 `visudo` 編輯。之所以使用 `visudo` 有兩個原因，一是它能夠防止兩個用戶同時修改它；二是它也能進行一些的語法檢查。以 `root` 身份用 `visudo` 打開配置檔，輸入以下內容：

```
#juan ALL=(ALL) ALL
```

這條資訊意思是 `juan` 用戶可以以任何主機連接並通過 `sudo` 執行任何命令。

下麵這條資訊說明 `users` 用戶可以本地主機可以執行 `/sbin/shutdown -h now` 命令：

```
%users localhost=/sbin/shutdown -h now
```



## 第二章 基本開發環境

### 2.1. Qt-5.14.2

Qt 是一個跨平臺的桌面、嵌入式和移動應用程式開發框架。只需重新編譯即可將現有的桌面或嵌入式應用程式帶到移動設備中。有很強的圖形功能和性能。Qt5 是 Qt 的最新版本，開發人員能夠以直觀的用戶介面針對多個目標開發應用程式。通過 Qt5 中改進的 JavaScript 和 QML 支持，開發人員可以更加高效和靈活，同時仍支持 C++ 和 Qt Widget。Qt5 與 Qt4 高度相容，並借助模組化的代碼庫和 Qt Platform Abstraction，增強了代碼的可移植性。

當前更新至版本 5.14.2，更多新特性請查閱

<https://doc.qt.io/archives/qt-5.14>

### 2.2. GCC-7.3

GCC 是由 GNU 開發的編程語言編譯器，支持多種語言的編譯，例如 C，C++，Objective-C，Java 等，同時包含這些語言的庫檔。GCC 是一種開源的開發工具，支持多種體系架構，易於擴展和測試。

主要特性包括：

- 支持 GNU 標準；
- 編譯器基於 GPL 標準；
- 具有不斷更新的運行時庫，調試效率高；

詳細用法或其他資料請查閱 <https://gcc.gnu.org/>。

### 2.3. GDB-9.2

GDB 是 GNU 代碼調試器，允許查看程式內部執行流程，或者程式在發生異常時的狀態。GDB 的功能主要包括：

執行一些能夠影響程式運行結果的操作；

在指定的條件下停止程式；

在程式停止運行時，檢查此時程式內部發生了什麼；

修改程式，以驗證程式 bug 對程式的影響，同時瞭解到程式中許多其他的內容，例如變數取值等。

GDB 支持以下編程語言：

- Ada
- Assembly
- C
- C++
- D
- Fortran
- Go
- Objective-C
- OpenCL
- Modula-2
- Pascal
- Rust

終端控制臺輸入命令“gdb”，即可打開 GDB 調試工具。

```
[root@localhost ~]# gdb
GNU gdb (GDB) KylinOS 9.2-7.p02.ky10
Copyright (C) 2020 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-kylin-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
  <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word".
(gdb) █
```

圖 2-1 顯示頁面

當前更新至版本 9.2，詳細用法或其他資料請查閱

<http://gnu.org/software/gdb/>。

## 2.4. Python3-3.7.9

Python 是一種清晰而強大的面向對象編程語言，可與 Perl、Ruby、Scheme 或 Java 相媲美。

Python 的一些顯著特性：

語法簡潔，程式易於閱讀。

程式運行簡單，這使得 Python 成為許多編程任務的理想選擇，同時又不影響可維護性。

附帶一個大型標準庫，支持許多常見的編程任務，如連接網路伺服器、用正則運算式搜索文本、讀取和修改檔等。

Python 的交互模式使得測試簡短的代碼片段變得很容易，python 開發環境叫做 IDLE。

通過添加以編譯語言(如 C 或 C++)實現的新模組，可以輕鬆地進行擴展。

也可以嵌入到應用程式中以提供可編程介面。

在任何地方運行，包括 Mac OS X，Windows，Linux 和 Unix，非官方版本也可用於 Android 和 iOS。

Python 是一款免費軟體。下載或使用 Python，或者將它包含在您的應用程式中不需要任何費用，Python 也可以被自由修改和重新發佈。

Python 的一些編程語言特性包括：

有多種基本數據類型可用：number(floating point, complex 和 unlimited-length long integers)、strings(包括 ASCII 和 Unicode)、lists 和 dictionaries。

Python 支持帶有類和多重繼承的面向對象編程。

代碼可以分成模組和包。

該語言支持引發和捕獲異常，從而實現更清晰的錯誤處理。

數據類型是強類型和動態類型。混合不相容的類型(例如，試圖添加一個字串和一個數字)會導致引發異常，從而更快地發現錯誤。

Python 包含高級編程特性，如 generators 和 list comprehensions。

Python 的自動記憶體管理使您不必手動分配和釋放代碼中的記憶體。

當前更新至版本 3.7.9，詳細資料請查閱 <https://www.python.org/>。

## 2.5. Openjdk-1.8.0

Openjdk 作為 GPL 許可的 Java 平臺開源化實現，由 Sun 公司開發，提供了一個 java 的運行環境，支持 Solaris, Linux, Mac OS X 或 Windows 多種操作系統。

新版本的新特性主要有：

Lambda 運算式和 Stream API；

時間與日期 API；

構造器引用；

紅黑樹的使用使運行速度更快；

減少空指針異常等。

當前更新至版本 1.8.0，詳細資料請查閱 <http://openjdk.java.net>。

## 第三章 常用圖形化工具

### 3.1. 刻錄工具

光碟刻錄器是一款便捷的刻錄工具，點擊**開始->所有程式->系統工具->光碟刻錄器**，可以創建音頻光碟、數據 CD/DVD、視頻 DVD/SVCD 以及鏡像的刻錄，如下圖：



圖 3-1 刻錄工具

### 3.2. 磁片

#### 3.2.1. 磁片管理

##### 3.2.1.1. 磁片管理工具介紹

磁片管理工具是一款能夠查看並管理磁片分區以及創建和恢復分區的工具，可以用它進行創建和格式化分區、掛載和卸載卷組以及其他相關的磁片操作。

##### 3.2.1.2. 磁片管理工具介面展示

點擊左下角-【開始】，介面會彈出菜單欄，依次選擇**所有程式->附件->磁片**。



圖 3-2 顯示應用程式

磁片管理介面，左側顯示目前現有設備和支持設備目錄，右側顯示選中設備的詳細使用情況。能夠識別到外部硬碟和額外的硬碟驅動器。

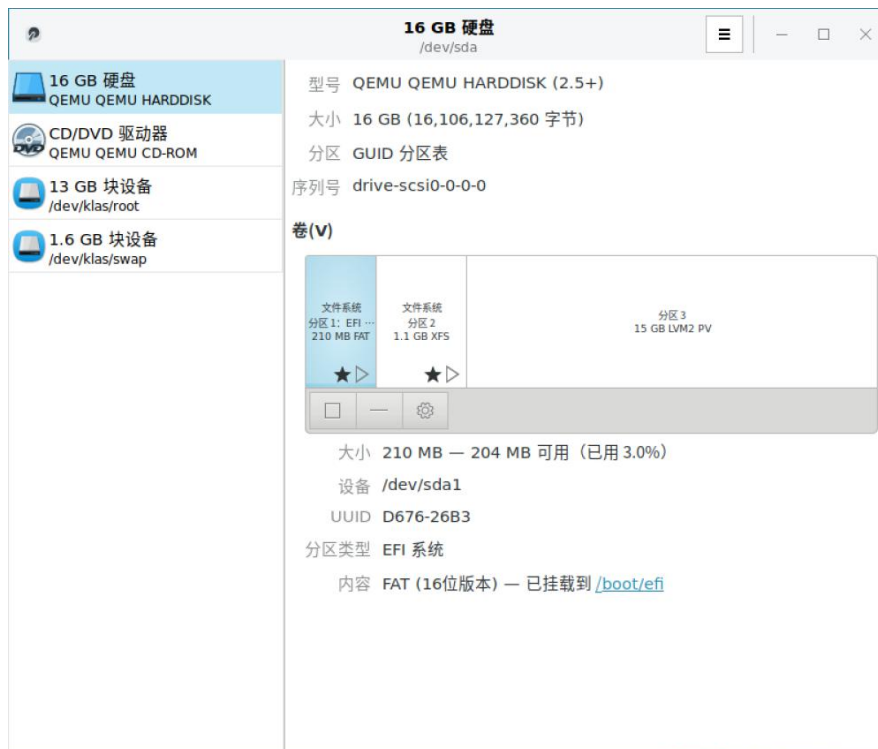


圖 3-3 磁片管理介面

### 3.2.2. 磁片管理工具使用

#### 3.2.2.1. 分區創建

選擇剩餘空間，點擊【+】圖示彈出創建分區頁面，拖動按鈕選擇創建分區的大小，點擊下一步，輸入卷名和選擇類型後點擊【創建】，創建完成後圖形界面顯示已創建分區。

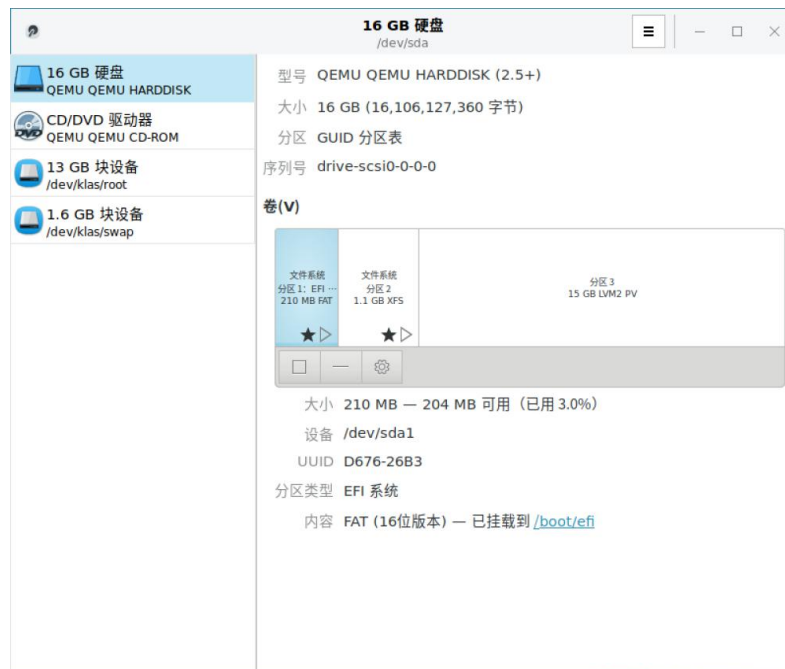


圖 3-4 創建分區

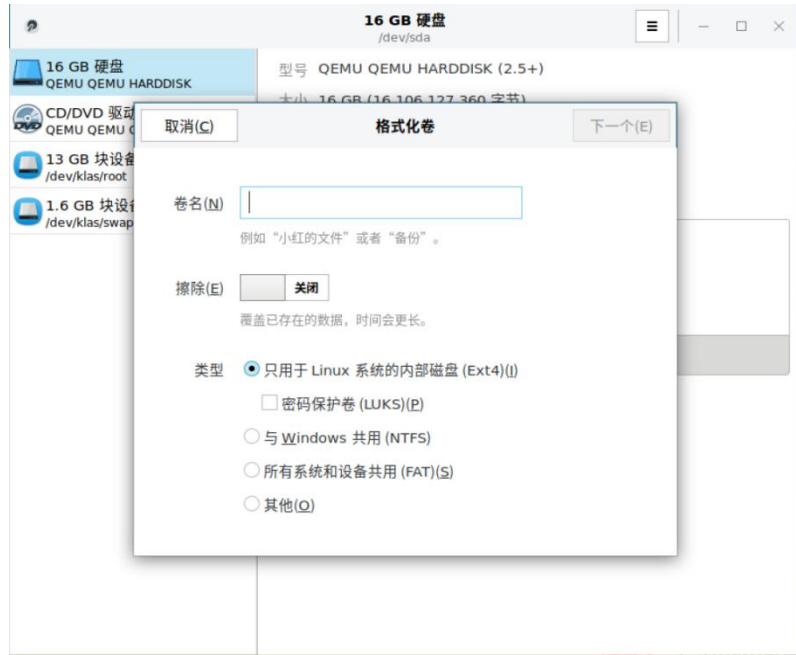


圖 3-5 格式化卷

### 3.2.2.2. 分區格式化

選擇需要格式化的磁片或分區，選中後點擊設置彈出下拉菜單，選擇格式化分區，輸入卷名，選擇類型後，點擊下一步，點擊【格式化】，磁片開始格式化，格式化成功後即可正常使用。



圖 3-6 格式化



### 3.2.2.3. 分區編輯

選擇需要編輯的分區，點擊下方【設置】按鈕彈出下拉菜單，點擊【編輯分區】，在彈出的編輯分區窗口中選擇對應的類型，點擊更改即可。

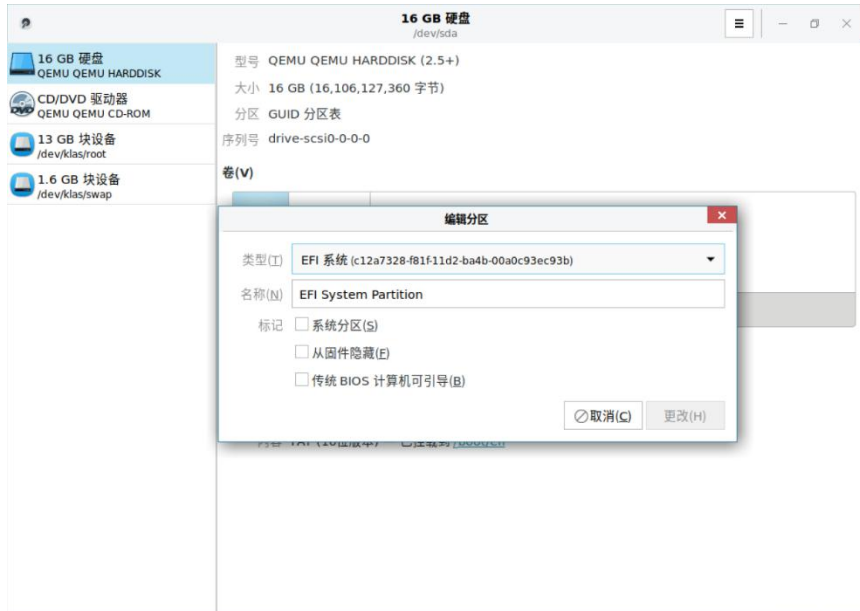


圖 3-7 編輯分區

### 3.2.2.4. 編輯檔系統

選擇需要修改名稱的分區，點擊【設置】按鈕，彈出下拉菜單，點擊【編輯檔系統】，彈出編輯檔系統的窗口，輸入新的分區名稱，點擊更改即可。



### 圖 3-8 編輯檔系統

#### 3.2.2.5. 分區大小調整

選擇需要修改大小的分區，點擊【設置】按鈕，彈出下拉菜單，點擊【調整大小】，彈出調整大小窗口，拖動選擇大小或者手動輸入大小，點擊調整大小即可。

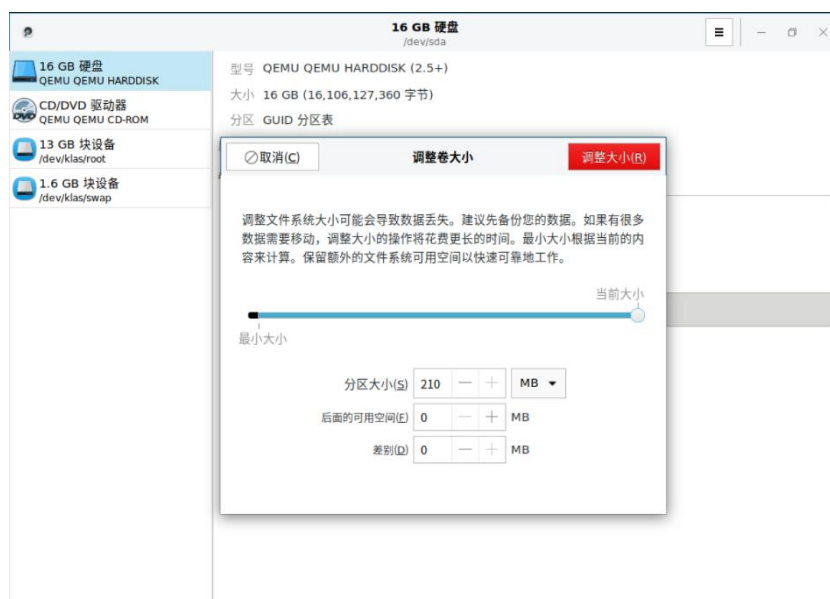


圖 3-9 編輯調整大小

#### 3.2.2.6. 分區卸載和掛載

選擇需要卸載的分區，點擊下側的【□】圖案進行卸載，

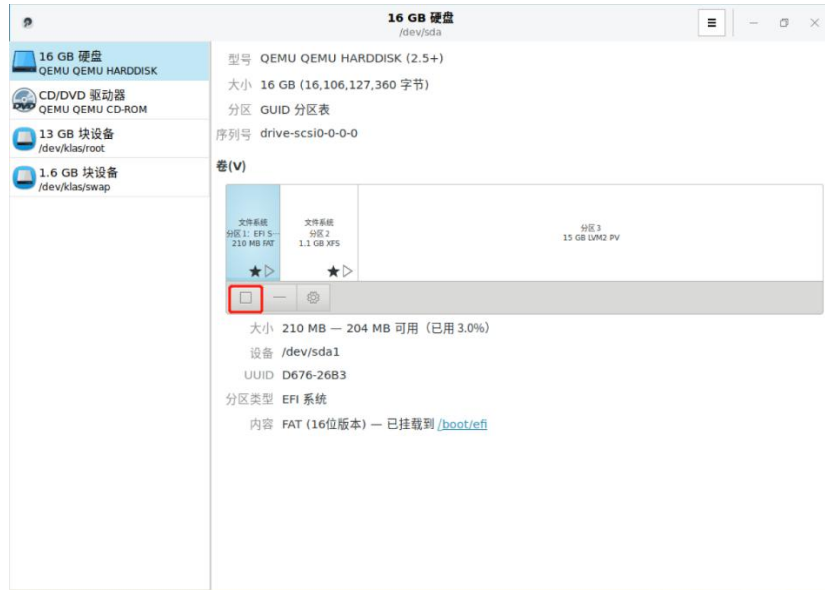


圖 3-10 分區卸載

卸載完成後，內容顯示為未掛載，此時卸載時的□變成了▷



圖 3-11 分區卸載完成

點擊【▷】進行掛載，掛載完成後，內容顯示為已掛載到對應目錄。



圖 3-12 分區掛載

### 3.2.2.7. 分區刪除

選擇要刪除的分區，點擊【-】彈出刪除分區的提示，點擊【刪除】即可。

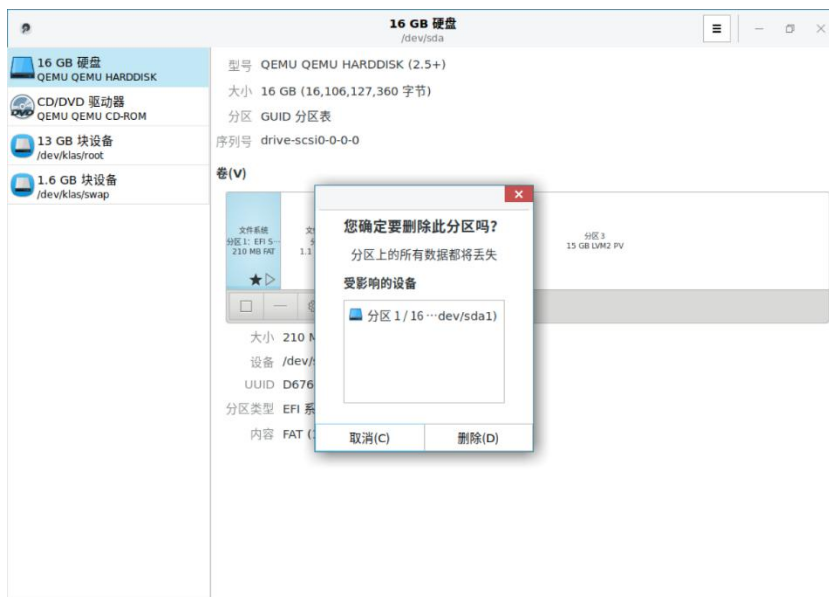


圖 3-13 分區刪除

## 3.3. 遠程桌面

### 3.3.1. VNC 查看器

TigerVNC Viewer 是一款方便的遠程桌面查看端, 點擊開始->所有程式->

互聯網->TigerVNC 查看器，輸入 VNC 伺服器資訊點擊連接，即可連接。



圖 3-14 VNC 查看器

點擊“選項”按鈕，可以對壓縮、安全、輸入、螢幕以及雜項等進行相關配置。



圖 3-15 VNC 查看器配置

### 3.3.2. 遠程查看程式 SSH

ssh 工具可以讓用戶登錄到一臺遠程機器上，並在該機器上執行命令。它是對 rlogin、rsh 和 telnet 程式的一個安全替換。和 telnet 命令相似，使用以下命令登錄到一臺遠程機器上：

```
$ssh hostname
```

例如，要登錄到一臺名為 `penguin.example.com` 的遠程主機，可以在 shell 命令行提示符下輸入以下命令：

```
$ssh penguin.example.com
```

該命令將會以用戶正在使用的本地機器的用戶名登錄。如果用戶想指定一個不同的用戶名，請使用以下命令：

```
$ssh username@hostname
```

例如，以 `USER` 登錄到 `penguin.example.com`，請輸入以下命令：

```
$ssh USER@penguin.example.com
```

用戶第一次連接時，將會看到和如下內容相似的資訊：

```
The authenticity of host 'penguin.example.com' can't be
established.
ECDSA key fingerprint is
SHA256:lxY64icRYc/h7XSOvUVywS7t7ThtmOsPT1s07wDD5P8.
Are you sure you want to continue connecting
(yes/no/[fingerprint])?
```

在回答對話框中的問題之前，用戶應該始終檢查指印是否正確。用戶可以詢問服務端的管理員以確認密鑰是正確的。這應該以一種安全的、事先約定好的方式進行。如果用戶可以使用服務端的主機密鑰，可以使用以下 `ssh-keygen` 命令來檢查指印：

```
#ssh-keygen -l -f /etc/ssh/ssh_host_ecdsa_key.pub
256
SHA256:b0gGbl+Xk/l+Ve76j3mqpYSty0n3gR9Qilsd+8oV3Gln
o comment (ECDSA)
```

輸入 `yes` 接受密鑰並確認連接。用戶將會看到一個有關服務端已經被添加到已知的主機列表中的通告，以及一個輸入密碼的提示：

```
Warning: Permanently added 'penguin.example.com' (ECDSA)
to the list of known hosts.
USER@penguin.example.com's password:
```

重要說明：如果 SSH 服務端的主機密鑰改變了，客戶端將會通知用戶連接不能繼續，除非將服務端的主機密鑰從 `~/.ssh/known_hosts` 檔中刪除。然而，在進行此操作之前，請聯繫 SSH 服務端的系統管理員，驗證服務端沒有受到攻擊。

要從 `~/.ssh/known_hosts` 檔中刪除一個密鑰，可以使用如下命令：

```
#ssh-keygen -R penguin.example.com #Host
penguin.example.com found: line 15 type ECDSA
/home/USER/.ssh/known_hosts updated. Original contents
retained as /home/USER/.ssh/known_hos ts.old
```

在輸入密碼之後，用戶將會進入遠程主機's `shell` 命令行提示符下。可選地，`ssh` 程式可以用來在遠程主機上執行一條命令，而不用登錄到 `shell` 命令行提示符下：

```
ssh [username@]hostname command
```

例如，`/etc/kylin-release` 檔提供有關操作系統版本的資訊。要查看 `penguin.example.com` 上該檔的內容，輸入：

```
$ssh USER@penguin.example.com cat /etc/kylin-release
USER@penguin.example.com's password:
Kylin Linux Advanced Server release V10 (Lance)
```

在用戶輸入正確的密碼之後，將會顯示遠程主機的操作系統版本資訊，然後

用戶將返回到本地的 shell 命令行提示符下。

### 3.4. Cockpit 遠程管理

#### 3.4.1. Cockpit

Cockpit 是一個 Web 控制臺，具有易於使用的基於 Web 的介面，使您可以在伺服器上執行管理任務。

Cockpit Web 控制臺使您可以執行多種管理任務，包括：管理服務，管理用戶帳號，管理和監視系統服務，配置網路介面和防火牆，查看系統日誌，管理虛擬機，創建診斷報告，設置內核轉儲配置，配置 SELinux，更新軟體，管理系統訂閱等。

Cockpit Web 控制臺使用與終端相同的系統 API，並且在終端中執行的任務會迅速反映在 Web 控制臺中。此外，您還可以直接在 Web 控制臺中或通過終端配置設置。

#### 3.4.2. 啟動和查看 Cockpit 服務

使用如下命令啟動和查看 cockpit 服務：

```
#systemctl enable --now cockpit.socket
```

```
[root@localhost ~]# systemctl enable --now cockpit.socket  
Created symlink /etc/systemd/system/sockets.target.wants/cockpit.socket → /usr/lib/systemd/system/cockpit.socket.
```

```
#systemctl status cockpit.socket
```

```
[root@localhost ~]# systemctl status cockpit.socket  
● cockpit.socket - Cockpit Web Service Socket  
Loaded: loaded (/usr/lib/systemd/system/cockpit.socket; enabled; vendor preset: disabled)  
Active: active (listening) since Fri 2020-03-20 15:07:57 CST; 25min ago  
Docs: man:cockpit-ws(8)  
Listen: [::]:9090 (Stream)  
Tasks: 0  
Memory: 4.0K  
CGroup: /system.slice/cockpit.socket  
3月 20 15:07:57 localhost.localdomain systemd[1]: Starting Cockpit Web Service Socket.  
3月 20 15:07:57 localhost.localdomain systemd[1]: Listening on Cockpit Web Service Socket.
```



如果您正在系統上運行 `firewalld`，則需要使用如下命令打開防火牆中的 Cockpit 端口 9090。

```
#firewall-cmd --add-service=cockpit --permanent
#firewall-cmd --reload
```

```
[root@localhost ~]# firewall-cmd --add-service=cockpit --permanent
success
[root@localhost ~]# firewall-cmd --reload
success
```

### 3.4.3. Cockpit Web 控制臺

Cockpit 使用 9090 端口，並且為 SSL 加密訪問，通過瀏覽器登錄，在網路瀏覽器中，通過以下 URL 打開 Cockpit 網路控制臺：

本地：`https://localhost:9090`

遠程：使用伺服器的主機名或者 IP 地址，例如：

`https://192.168.17.205:9090`

登錄時瀏覽器會提示鏈接不安全，點擊添加例外。

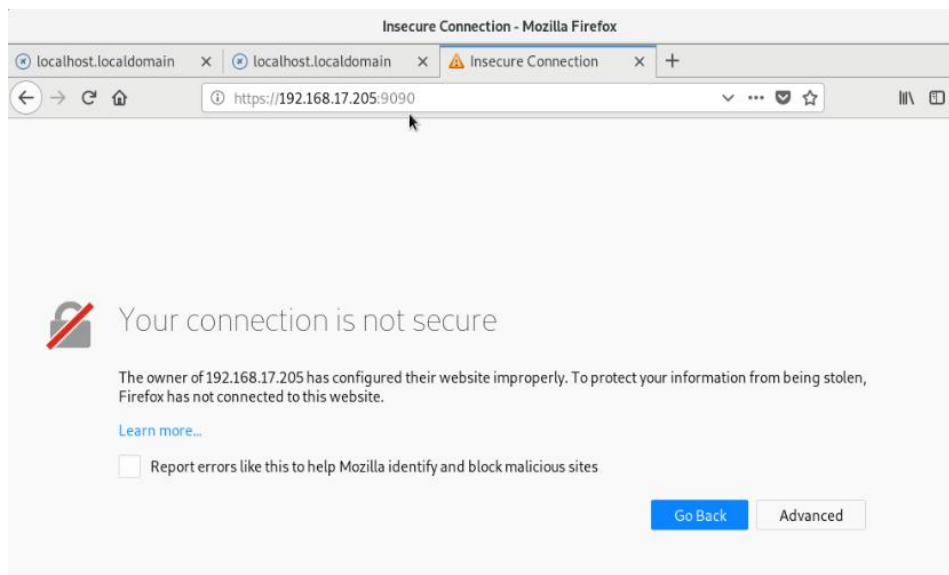


圖 3-16 瀏覽器登錄提示

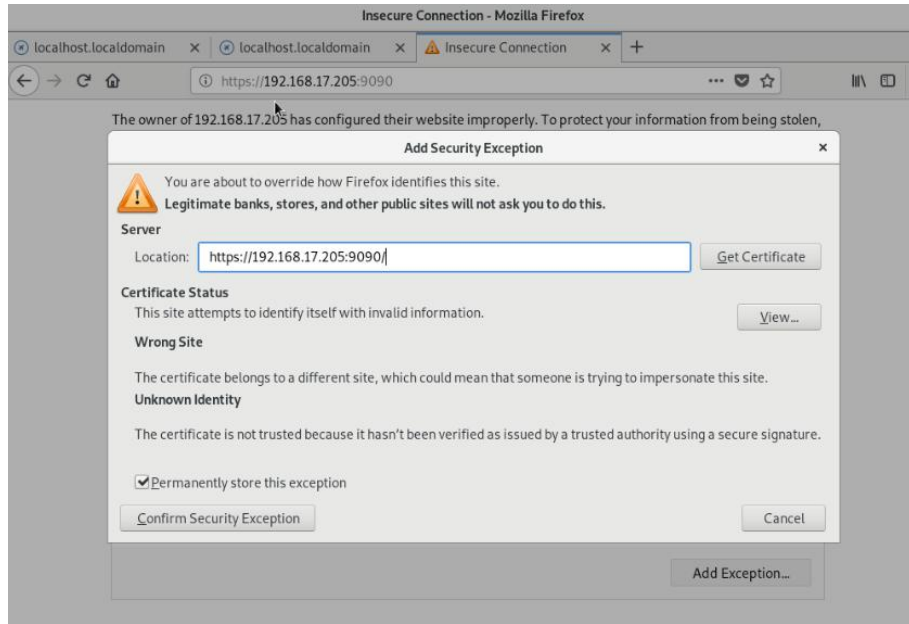


圖 3-17 添加例外

### 3.4.3.1. 登錄

在 Web 控制臺登錄螢幕中，輸入系統用戶名和密碼進行登錄，如圖所示。如果用戶帳戶具有 `sudo` 特權，則可以執行管理任務，例如在 Web 控制臺中安裝軟體，配置系統或配置 SELinux 等。

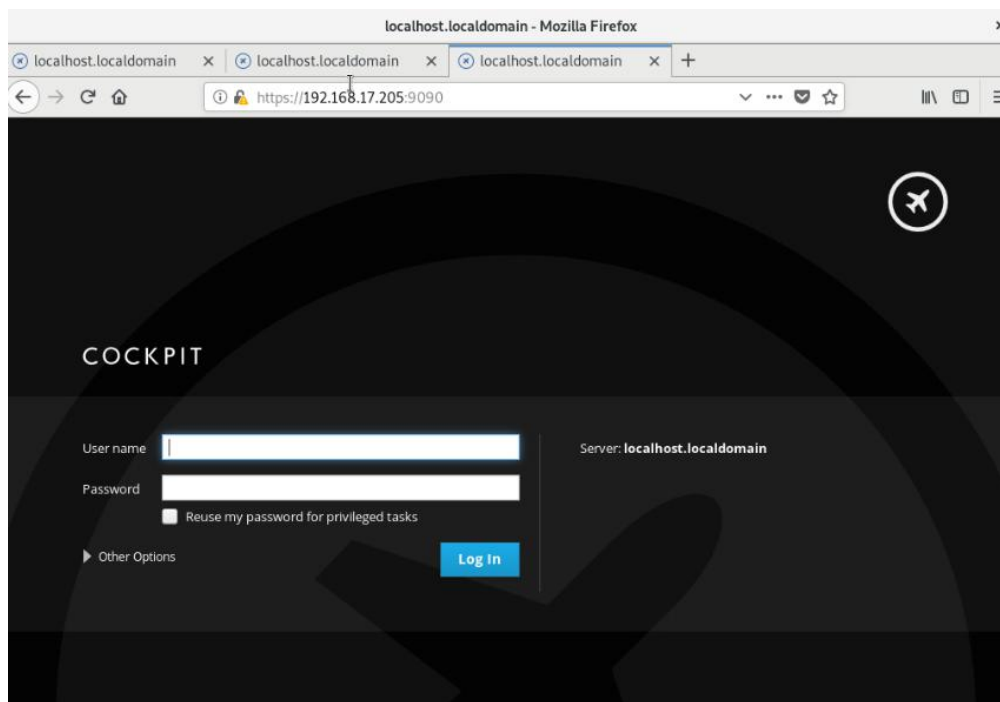


圖 3-18 登錄介面

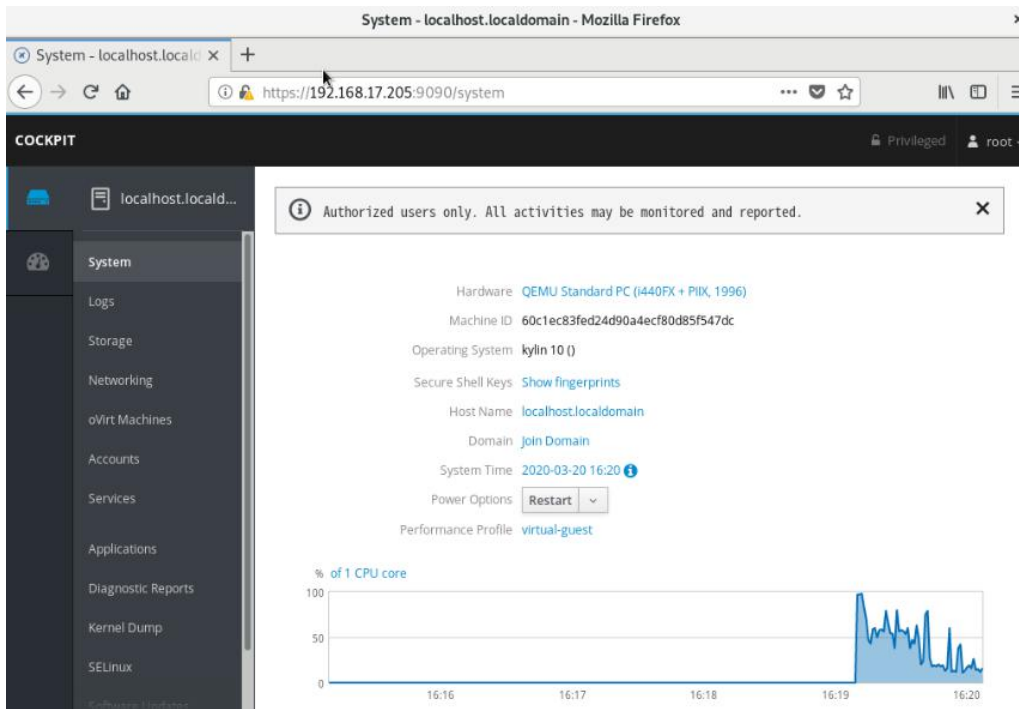


圖 3-19 登錄後介面

### 3.4.3.2. 語言選擇

可以對語言進行切換，如圖所示。

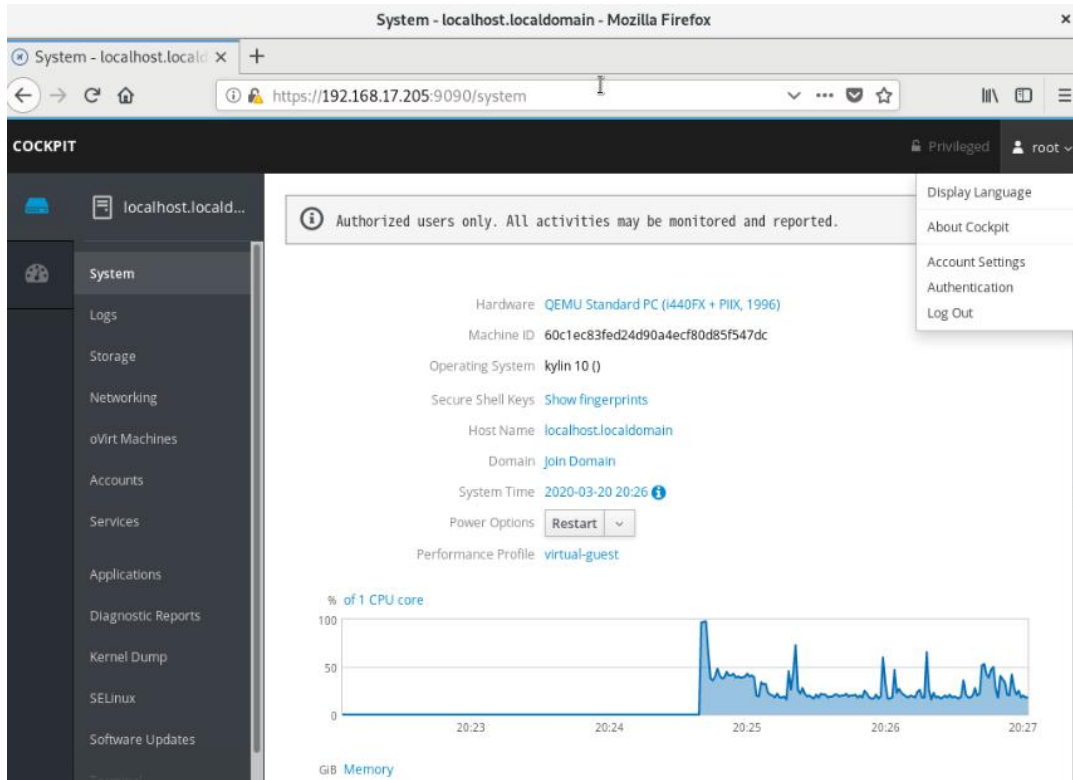


圖 3-20 語言切換

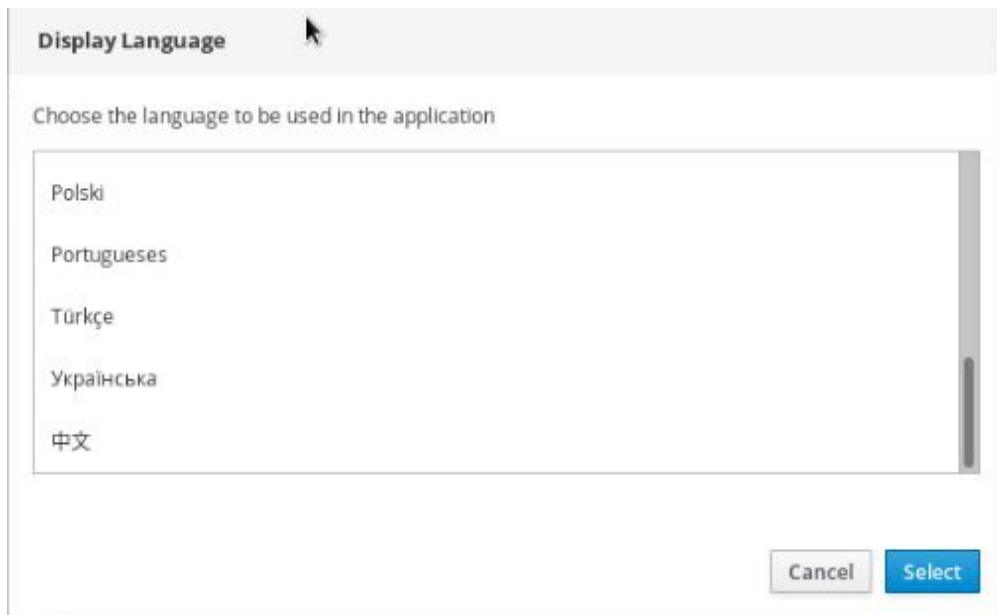


圖 3-21 語言選擇

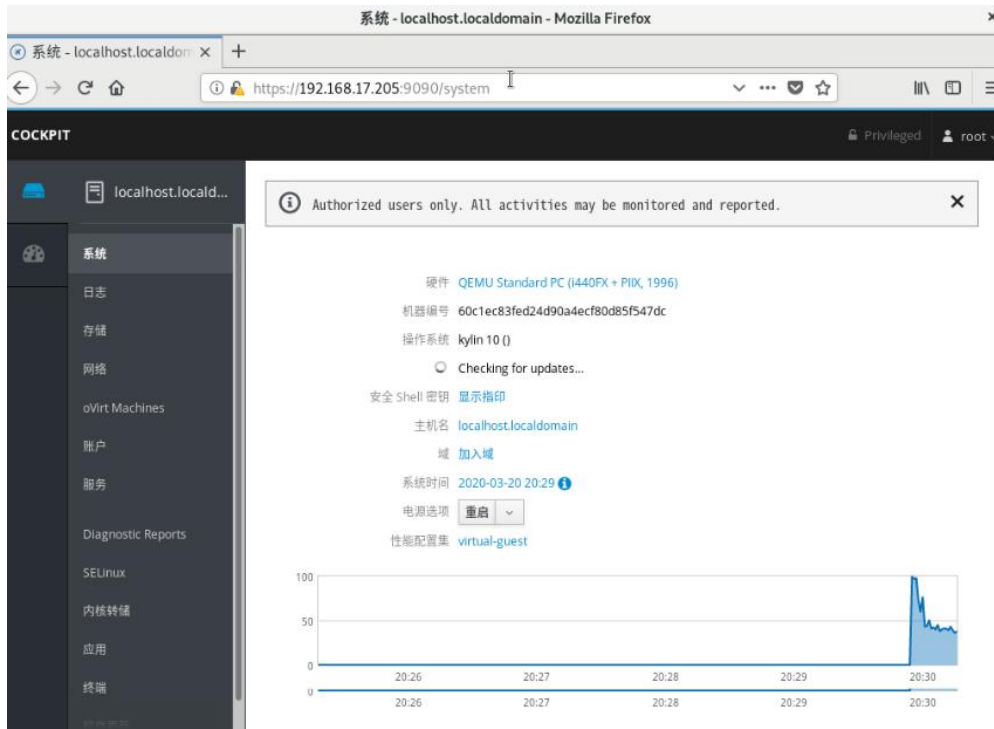


圖 3-22 語言選定

### 3.4.3.3. 日誌

Cockpit 將日誌資訊按時間和嚴重性來進行不同的分類。

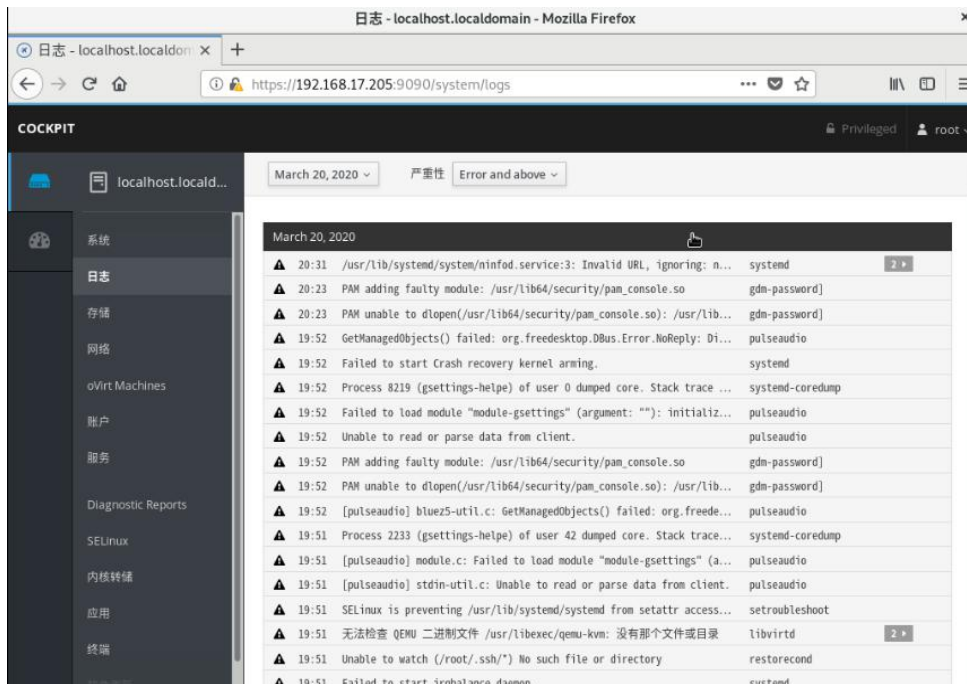


圖 3-23 日誌介面

### 3.4.3.4. 網路

網路頁面可以看到兩個可視化發送和接收速度的圖。還可以看到可用網卡的列表，可以對網卡進行綁定設置，橋接，以及配置 VLAN。點擊網卡就可以進行編輯操作。

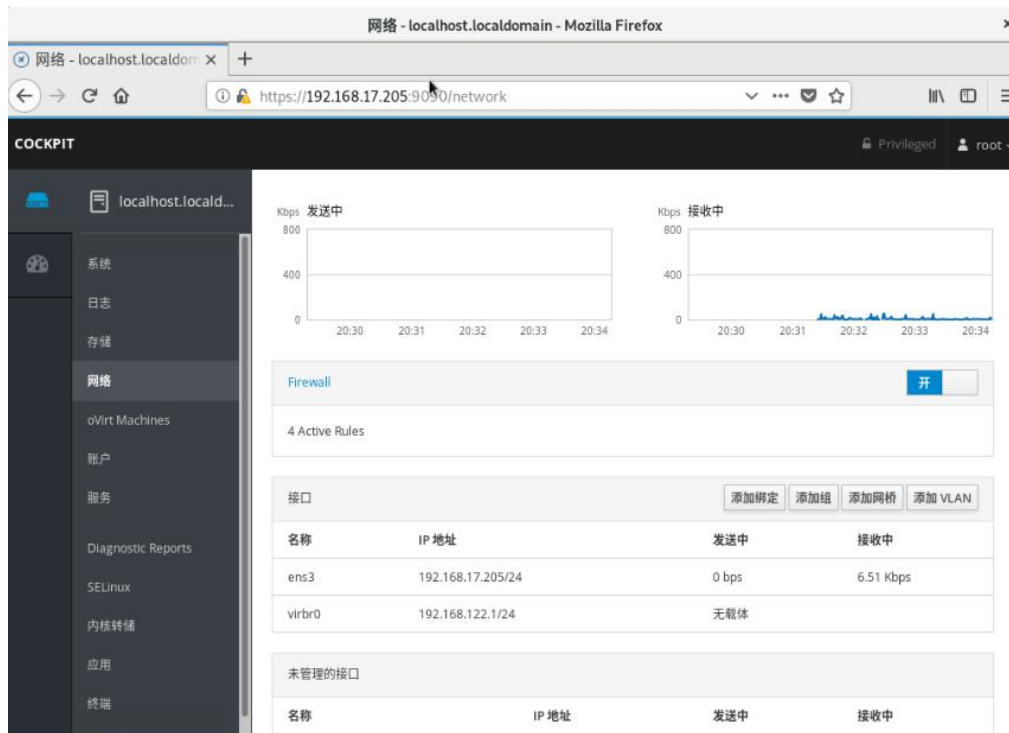


圖 3-24 網路介面

### 3.4.3.5. 服務

服務被分成了 5 個類別：目標、系統服務、套接字、計時器和路徑。

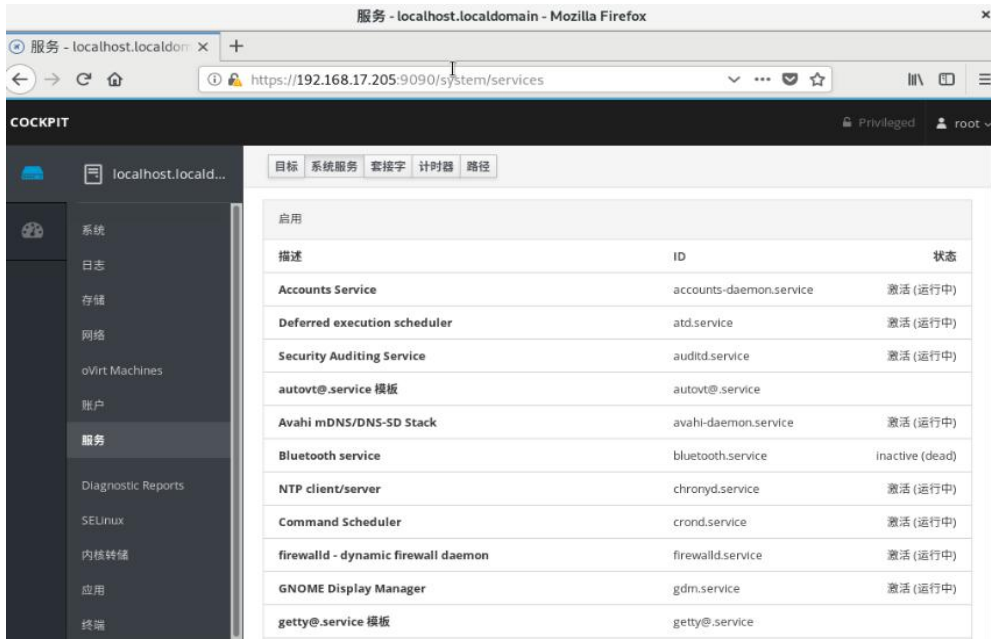


圖 3-25 服務介面

### 3.4.3.6. 終端

Cockpit 介面提供即時終端執行任務，可以根據需求在 Web 介面和終端之間自由切換，可以快速執行任務，操作非常方便。

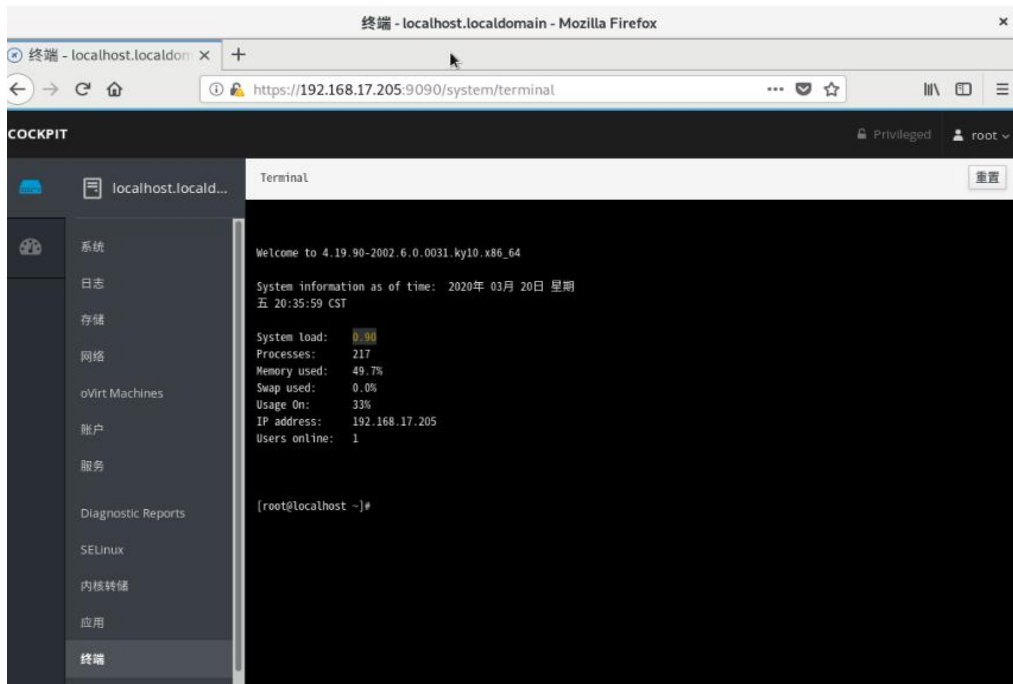


圖 3-26 終端介面

### 3.4.3.7. 儀錶盤

可以利用儀錶盤進行多主機監控，點擊右側的“+”添加需要監控的主機。

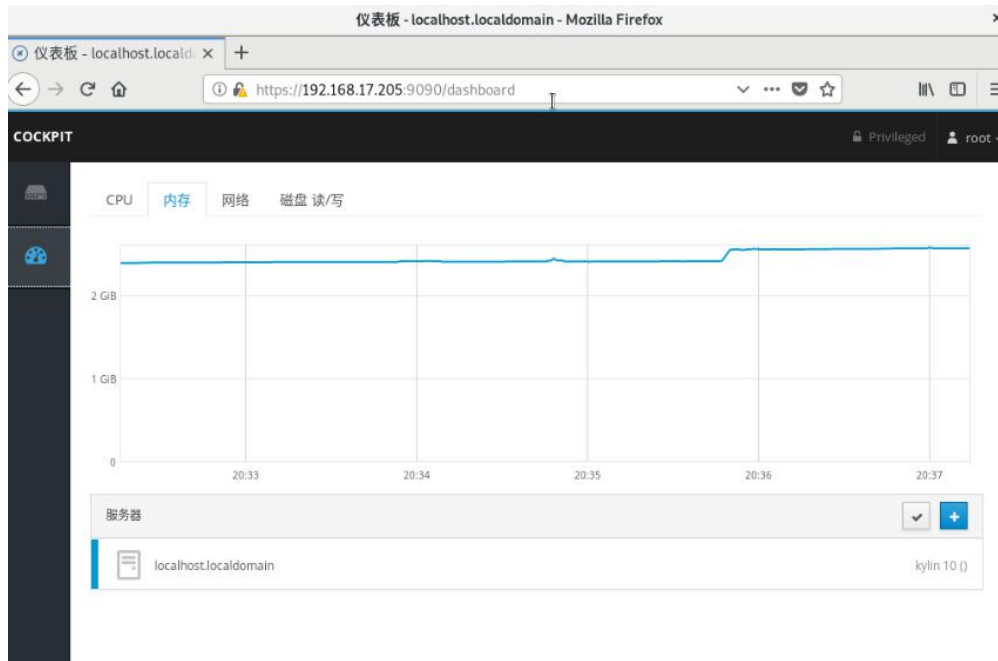


圖 3-27 儀錶盤介面

## 3.5. 系統日誌

銀河麒麟V10操作系統提供了一個基於圖形用戶介面的日誌管理工具-麒麟日誌查看器，使用該工具能夠對於系統各方面的日誌進行方便的查看、搜索、過濾和導出。

日誌查看器具有以下特點：

智能化收集展示：即時同步收集展示系統內日誌資訊，根據日誌類型進行歸類顯示。同時，具有過濾和聚合功能，對重複日誌資訊進行合併統計顯示。

標準化全景態勢：提供系統日誌、啟動日誌、登錄日誌、應用日誌以及安全日誌。通過安全視角將事件標準化描述，包含目標事件等級、對象類型、時間、事件詳細資訊。



模組化維護擴展：採用模組化、可插拔架構設計，每類日誌組件能夠以模組化橫向擴展，對不同類別日誌獨立維護，具有靈活易用、可維護特徵。

詳細內容請參考《銀河麒麟日誌查看器用戶手冊》。

## 第四章 安裝和管理軟體

dnf 是新一代的軟體包管理器，首先出現在 Fedora 18 這個發行版中。而在 Fedora 22 中，它取代了 yum，正式成為了 Fedora 22 的包管理器。

dnf 包管理器克服了 yum 包管理器的一些瓶頸，提升了包括用戶體驗、記憶體佔用、依賴分析和運行速度等多方面的內容。dnf 使用 rpm、libsolv 和 hawkey 庫進行包管理操作，可以同 yum 同時使用。

### 4.1. 檢查和升級軟體包

#### 4.1.1. 軟體包升級檢查

查看系統裏已經安裝的軟體包有哪些可以升級可以執行以下命令，以 X86 平臺示例如下：

```
[root@localhost ssh]# dnf check-update
上次元数据过期检查: 2:09:08 前, 执行于 2022年07月20日 星期三 09时54分49秒。
anaconda.x86_64                               33.19-31.p12.ky10                               ks10-adv-os
anaconda-core.x86_64                          33.19-31.p12.ky10                               ks10-adv-os
anaconda-tui.x86_64                           33.19-31.p12.ky10                               ks10-adv-os
```

示例說明：

- PackageKit——軟體包名稱；
- x86\_64——該軟體包支持的 CPU 架構；
- 33.19-31.p12.ky10——可升級的軟體包版本；
- ks10-adv-os——可升級的軟體包所存儲倉庫。

#### 4.1.2. 升級軟體包

dnf 支持一次升級單個/批量軟體包，並同時安裝/更新相應的依賴包。

##### 1. 升級單一軟體包命令：

```
#dnf update {package_name}
```

升級 kernel 軟體包命令為例：

```
[root@localhost ssh]# dnf update kernel
上次元数据过期检查: 2:01:20 前, 执行于 2022年07月20日 星期三 09时54分49秒。
依赖关系解决。
=====
Package                Architecture    Version          Repository        Size
-----
安装:
kernel                 x86_64         4.19.90-51.0.v2207.ky10  ks10-adv-os-koji  749 k
安装依赖关系:
kernel-core            x86_64         4.19.90-51.0.v2207.ky10  ks10-adv-os-koji  38 M
kernel-modules         x86_64         4.19.90-51.0.v2207.ky10  ks10-adv-os-koji  21 M
kernel-modules-extra   x86_64         4.19.90-51.0.v2207.ky10  ks10-adv-os-koji  2.2 M
=====
事务概要
-----
安装 4 软件包
总下载: 62 M
安装大小: 86 M
确定吗? [y/N]:
```

上述輸出的說明如下：

- a) Package: 用戶需要下載升級的軟體包和依賴軟體包。
- b) Architecture: 該軟體包所屬的架構。
- c) Version: 軟體包升級後的版本。
- d) Repository: 可升級軟體包所屬倉庫。
- e) Size: 軟體包大小。
- f) dnf 默認會顯示升級軟體包的基本資訊，並提示是否確認安裝，用戶可以在使用 dnf 命令時添加參數-y，效果等同於出現 Is this ok [y/N]: 時輸入 yes。
- g) 安裝過程中如果出現錯誤導致安裝過程終止，可以使用 dnf history 命令查看詳細描述。

如果需要安裝一組軟體包，可以以 root 用戶執行命令：

```
#dnf groupupdate group_name
```

## 2. 批量升級軟體包及其依賴

如果需要升級系統所有軟體包，可以使用以下命令：

```
#dnf update
```

#### 4.1.3. 利用系統光碟與 dnf 離線升級系統

dnf 可與 yum 使用相同的配置檔，即配置 dnf 源可直接對 /etc/yum.repos.d/ 中的 .repo 檔進行編輯。當系統處於離線狀態或者無法訪問官方更新源時，可以利用更新的系統光碟創建本地源並進行升級。步驟如下：

1. 創建系統光碟掛載目錄，以 root 用戶執行：

```
#mkdir {mount_dir}
```

2. 將系統安裝光碟掛載至該目錄，以 root 用戶執行

```
#mount -o loop {iso_name} {mount_dir}
```

3. 將系統光碟中的 media.repo 檔從掛載目錄拷貝至 /etc/yum.repos.d/

目錄下：

```
#cp mount_dir/media.repo /etc/yum.repos.d/new.repo
```

4. 編輯 /etc/yum.repos.d/new.repo 配置檔以添加光碟路徑：

```
#baseurl=file://mount_dir
```

5. 更新 dnf 源並進行升級，以 root 用戶執行：

```
#dnf update
```

6. 升級成功後，卸載系統光碟掛載目錄：

```
#umount mount_dir 或者 rmdir mount_dir
```

如果不再使用這個 dnf 源進行安裝和升級，可以以 root 用戶刪除檔：

```
#rm /etc/yum.repos.d/new.repo
```

## 4.2. 管理軟體包

dnf 提供了完整操作系統軟體包管理功能，包括檢索、查看資訊、安裝和刪除。

### 4.2.1. 檢索軟體包

執行 `dnf search` 命令可以檢索軟體包，例如檢索包含“mesh”字段的軟體包，以 X86 平臺示例如下：

```
#dnf search mesh
```

如果 `dnf` 檢測的結果繁多，可以通過 `shell` 本身的 `grep` 或者正則運算式進行過濾。

### 4.2.2. 安裝包列表

顯示已安裝和可安裝的軟體包列表可以執行以下命令：

```
#dnf list all
```

顯示包括某些字元的已安裝和可安裝軟體包列表可以執行以下命令：

```
#dnf list glob_expression...
```

顯示 `abrt` 相關軟體包列表的命令如下：

```
#dnf list abrt-addon\* abrt-plugin\*
```

顯示包括某些字元的已安裝軟體包列表可以執行以下命令：

```
#dnf list installed glob_expression...
```

顯示包括 `krb` 的所有已安裝軟體包示例如下：

```
#dnf list installed "krb?-*"
```

顯示包括某些字元的可安裝軟體包列表可以執行以下命令：

```
#dnf list available glob_expression...
```

顯示所有可用的 gstreamer plug-ins 軟體包列表：

```
#dnf list available gstreamer\*plugin\*
```

查看軟體倉庫

成功註冊後，可使用 dnf 來管理軟體包。

查看可用的軟體倉庫可以使用以下命令：

```
#dnf repolist
```

如果想顯示更多資訊可以加上 -v 選項，或者用 dnf repoinfo 命令輸出資訊。

```
#dnf repolist -v
```

```
#dnf repoinfo
```

如果需要顯示所有可用和不可用的軟體倉庫，可以使用以下命令：

```
#dnf repolist all
```

#### 4.2.3. 顯示軟體包資訊

顯示一個或多個軟體包可以使用以下命令：

```
#dnf info package_name...
```

顯示軟體包 abrt 詳細資訊的命令：

```
#dnf info abrt
```

顯示軟體包 yum 詳細資訊的命令：

```
#dnf info yum
```

#### 4.2.4. 安裝軟體包

用戶可以以 root 用戶使用以下命令安裝軟體包

```
#dnf install package_name
```

安裝 sqlite 的 i686 架構的軟體包示例：

```
#dnf install sqlite.i686
```

除了安裝軟體包，還可以安裝具體的二進位檔，您可以輸入檔地址，以 root 用戶執行安裝：

```
#dnf install /usr/sbin/named
```

安裝命令如下：

```
#dnf install httpd
```

如果要安裝本地軟體包，可以執行：

```
#dnf localinstall path
```

#### 4.2.5. 下載軟體包

在執行安裝流程中，顯示以下選項是：

```
...  
Total size: 1.2 M  
Is this ok [y/N]:  
...
```

輸入 y，可以執行軟體包下載。

#### 4.2.6. 刪除軟體包

刪除軟體包可以執行以下命令：

```
dnf remove package_name...
```

刪除 totem 軟體包示例：

```
dnf remove totem
```

### 4.3. 管理軟體包組

軟體包組可以搜集一系列特定功能軟體包，比如系統工具和視頻軟體包組。

安裝軟體包組可以一起安裝其依賴。

#### 4.3.1. 軟體包組列表

Summary 選項可以顯示軟體包可用組的數量：

```
dnf groups summary
```

以下為輸出示例：

```
#dnf groups summary
```

```
可用組： 8
```

顯示某個軟體包組的全部資訊可以用以下命令：

```
#dnf groups info glob_expression...
```

以下為 Server 組輸出示例：

```
#dnf groups info Server
```



```
[root@localhost ~]# dnf groups info Server
Last metadata expiration check: 0:00:22 ago on Tue 09 Aug 2022 01:50:10 PM CST.
Environment Group: Server
Description: An integrated, easy-to-manage server.
Mandatory Groups:
  Base
  Container Management
  Core
  Hardware Support
  Headless Management
  Server product core
  Standard
Optional Groups:
  Basic Web Server
  DNS Name Server
  Debugging Tools
  FTP Server
  File and Storage Server
  Hardware Monitoring Utilities
  Infiniband Support
  Mail Server
  Network File System Client
  Performance Tools
  Remote Management for Linux
  Virtualization Hypervisor
  Windows File Server
```

#### 4.3.2. 安裝軟體包組

每個軟體包組都有自己的組 ID，要顯示包組 id 可以使用以下命令：

```
#dnf group list ids
```

查找開發軟體包組列表的示例：

```
#dnf groups list ids deve\*
```

```
[root@localhost ~]# dnf group list ids deve\*
Last metadata expiration check: 0:12:05 ago on Tue 09 Aug 2022 01:50:10 PM CST.
Available Groups:
  Development Tools (development)
```

軟體包組的安裝可以通過軟體包組名稱安裝，也可通過包組 id 安裝。

```
#dnf group install "group name"
#dnf group install groupid
```

也可用通過以下兩種命令安裝：

```
#dnf install @group
#dnf install @^group
```

下麵是 4 種安裝開發工具軟體分組的示例：

```
#dnf group install "Development Tools"  
#dnf group install development  
#dnf install @"Development Tools"  
#dnf install @development
```

#### 4.3.3. 刪除軟體包組

可以通過軟體包組名或者軟體包組 id 刪除軟體包。以 root 許可權執行：

```
#dnf group remove group_name  
#dnf group remove groupid
```

如果軟體分組有@標籤，也可用以下命令刪除。以 root 身份執行：

```
#dnf remove @group  
#dnf remove @^group
```

刪除 KDE 桌面軟體分組示例：

```
#dnf group remove "Development Tools"  
#dnf group remove development  
#dnf remove @"Development Tools"  
#dnf remove @development
```

#### 4.4. 軟體包操作記錄管理

dnf 可以使用 dnf history 命令進行管理操作。

##### 4.4.1. 查看操作

顯示以往 20 條 dnf 操作記錄，可以使用以下命令。以 root 許可權執行：

```
#dnf history list 1..20
```

ID	命令行	日期和时间	操作	更改
20	install libcap-devel	2021-06-08 14:02	I, U	2
19	update yum	2021-05-31 14:01	I, U	9 EE
18	install bind-utils	2021-05-28 09:57	Install	10
17	install net-tools	2021-05-25 09:56	Install	1
16	install docker	2021-05-24 15:15	Install	2 EE
15	install zlib-devel	2021-05-24 13:29	Install	1
14	install php-devel	2021-05-24 13:26	Install	8
13	install telnet	2021-05-20 08:54	Install	1
12	update	2021-05-17 13:49	E, I, U	17 EE
11	update	2021-05-13 08:43	Upgrade	13 EE
10	install php-mbstring	2021-04-30 17:33	Install	2
9	install php-mysqli	2021-04-30 17:32	Install	2
8	install php-json	2021-04-30 17:32	Install	1
7	install php-xml	2021-04-30 17:32	Install	1
6	install php-fpm	2021-04-30 17:25	Install	2
5	install -y php	2021-04-30 17:23	Install	3
4	install -y mariadb-server	2021-04-30 16:01	Install	6
3	install -y httpd	2021-04-30 15:55	Install	7
2	update	2021-04-30 15:46	Upgrade	19 EE
1		2021-04-30 15:10	Install	742 EE

如果想顯示某一部分 dnf 操作記錄，可以使用以下命令。以 root 許可權執行：

```
#dnf history list start_id. . end_id
```

顯示過去 5 條 dnf 資訊示例如下：

```
#dnf history list 1..5
```

ID	命令行	日期和时间	操作	更改
5	install -y php	2021-04-30 17:23	Install	3
4	install -y mariadb-server	2021-04-30 16:01	Install	6
3	install -y httpd	2021-04-30 15:55	Install	7
2	update	2021-04-30 15:46	Upgrade	19 EE
1		2021-04-30 15:10	Install	742 EE

以上 dnf history list 輸出顯示內容說明如下：

ID——識別特定記錄的標示數；

Command line——簡要描述操作內容；

Date and time——該條記錄的日期和時間；

Action(s)——描述操作類型；

Altered——記錄操作影響的條目數。

下表是 Action 的不同說明：

表 4-1 Action 說明

Action	縮寫	描述
Downgrade	D	下載更新包
Erase	E	刪除軟體包
Install	I	安裝軟體包
Obsoleting	O	軟體包標注廢棄
Reinstall	R	軟體包重裝
Update	U	升級軟體包

#### 4.4.2. 審查操作

需要顯示某條操作記錄的具體綜述資訊，可以執行以下命令：

```
#dnf history {id}
```

其中 id 是操作的 id。

如果需要顯示某條操作記錄的詳細資訊，可以使用以下命令：

```
#dnf history info {id}
```

如果需要顯示某一階段操作記錄的詳細資訊，可以使用以下命令：

```
#dnf history info start_id . . end_id
```

示例如下：

```
#dnf history info 4 . . 5
```

```
事务 ID: 4..5
起始时间 : 2021年04月30日 星期五 16时01分07秒
起始 RPM 数据库 : 747:8d43d8bfa0749f64ad69cd404fda7570feaaa531
结束时间 : 2021年04月30日 星期五 17时23分47秒 (82 分钟)
结束 RPM 数据库 : 756:44808f00897bf23cccd210933568326838a78c37
用户 : root <root>
返回码 : 成功
Releasever : 10
命令行 : install -y mariadb-server
命令行 : install -y php
注释 :
注释 :
已改变的包:
  安装 mariadb-common-3:10.3.9-8.ky10.x86_64 @ks10-adv-os
  安装 mariadb-errmsg-3:10.3.9-8.ky10.x86_64 @ks10-adv-os
  安装 mariadb-server-3:10.3.9-8.ky10.x86_64 @ks10-adv-os
  安装 mariadb-3:10.3.9-8.ky10.x86_64 @ks10-adv-os
  安装 perl-DBD-MySQL-4.046-6.ky10.x86_64 @ks10-adv-os
  安装 perl-DBI-1.642-2.ky10.x86_64 @ks10-adv-os
  安装 php-cli-7.2.10-12.ky10.x86_64 @ks10-adv-os
  安装 php-common-7.2.10-12.ky10.x86_64 @ks10-adv-os
  安装 php-7.2.10-12.ky10.x86_64 @ks10-adv-os
```

#### 4.4.3. 恢復與重複操作

如果想要撤銷某個 dnf 操作，可以以 root 許可權執行以下操作：

```
#dnf history undo {id}
```

如果需要重複某個 dnf 操作，可以以 root 許可權執行以下操作：

```
#dnf history redo {id}
```

## 第五章 基礎服務

該章節介紹如何配置系統服務、後臺 daemon 以及遠程訪問 Kylin Linux Advanced Server V10 系統。

### 5.1. 使用 systemd 管理系統服務

#### 5.1.1. Systemd 介紹

systemd 是 Linux 下一個與 SysV 和 LSB 初始化腳本相容的系統和服務管理器。systemd 使用 socket 和 D-Bus 來開啟服務，提供基於守護進程的按需

啟動策略，保留了 Linux cgroups 的進程追蹤功能，支持快照和系統狀態恢復，維護掛載和自掛載點，實現了各服務間基於從屬關係的一個更為精細的邏輯控制，擁有前衛的並行性能。systemd 無需經過任何修改便可以替代 sysvinit。

systemd 開啟和監督整個系統是基於 unit 的概念。unit 是由一個與配置檔對應的名字和類型組成的(例如：avahi.service unit 有一個具有相同名字的配置檔，是守護進程 Avahi 的一個封裝單元)。unit 有以下幾種類型：

**表 5-1unit 類型介紹**

Unit 類型	檔尾碼	描述
Service unit	.service	守護進程的啟動、停止、重啟和重載是此類型
Target unit	.target	此類 unit 為其他 unit 進行邏輯分組
Automount unit	.automount	此類 unit 封裝系統結構層次中的一個自掛載點
Device unit	.device	此類 unit 封裝一個存在於 Linux 設備樹中的設備
Mount unit	.mount	此類 unit 封裝系統結構層次中的一個掛載點
Path unit	.path	此類 unit 為檔系統中的一個檔或者目錄
Scope unit	.scope	此類 unit 為外部創建進行
Slice unit	.slice	此類 unit 為一組管理系統進程的分層 unit

Snapshot unit	.snapshot	與 target unit 相似，快照本身不做什麼，唯一的目的就是引用其他 unit
Socket unit	.socket	此類 unit 封裝系統和互聯網中的一個 socket
Swap unit	.swap	此類 unit 封裝 swap 設備或者 swap 檔
Timer unit	.timer	此類 unit 封裝系統時間

Systemd unit 的檔目錄說明如下：

表 5-2 Systemd unit 介紹

目錄	描述
/usr/lib/systemd/system/	RPM 包安裝時發佈的 Systemd units
/run/systemd/system/	運行時創建的 Systemd units，該目錄任務會覆蓋安裝時自帶的 Systemd units
/etc/systemd/system/	系統創建與管理的 Systemd units，該目錄任務會覆蓋運行時 Systemd units

#### 5.1.1.1. 主要特性

systemd 提供以下特性：

➤ 基於 socket 的並行性能：為了加速整個系統啟動和並行啟動更多的進程，systemd 在實際啟動守護進程之前創建 socket，然後傳遞 socket 給守護進程。在系統初始化時，首先為所有守護進程創建 socket，然後再啟動所有的守護進程。如果一個服務因為需要另一個服務的支持而沒有完全啟動，而這個連接可能正在提供服務的佇列中排隊，那麼這個客戶端進程在這次請求中就處於阻

塞狀態。不過只會有這一個客戶端進程會被阻塞，而且僅是在這一次請求中被阻塞。服務間的依賴關係也不再需要通過配置來實現真正的並行啟動(因為一次開啟了所有的 `socket`，如果一個服務需要其他的服務，它顯然可以連接到相應的 `socket`)。

- **D-Bus 啟動策略啟動服務：**通過使用匯流排啟動策略，服務可以在接入時馬上啟動。同時，匯流排啟動策略使得系統可以用微小的消耗實現 D-Bus 服務的提供者與消費者的同步開啟請求。(同時開啟多個服務，如果一個比匯流排啟動策略中其他服務快就在 D-Bus 中排隊其請求，直到其他管理確定自己的服務資訊為止)。

- 提供守護進程的按需啟動策略。

- 保留了使用 Linux `cgroups` 進程的追蹤功能：每一個執行了的進程獲得它自己的一個 `cgroup`，配置 `systemd` 使其可以存放在 `cgroup` 中已經經過外部配置的服務非常簡單。(如使用 `libcgroups utilities`)。

- 支持快照和系統狀態恢復：快照可以用來保存/恢復系統初始化時所有的服務和 `unit` 的狀態。它有兩種主要的使用情況：允許用戶暫時進入一個像 "Emergency Shell" 的特殊狀態，終止當前的服務；提供一個回到先前狀態的簡單方法，重新啟動先前暫時終止的服務。

- 維護掛載和自掛載點：`systemd` 監視所有的掛載點的進出情況，也可以用來掛載或卸載掛載點。`/etc/fstab` 也可以作為這些掛載點的一個附加配置源。通過使用 `comment=fstab` 選項您甚至可以標記 `/etc/fstab` 條目使其成為由 `systemd` 控制的自掛載點。



➤ 現了各服務間基於依賴關係的一個精細的邏輯控制：`systemd` 支持服務 (或 `unit`) 間的多種依賴關係。在 `unit` 配置檔中使用 `After/Before`、`Requires` 和 `Wants` 選項可以固定 `unit` 啟動的順序。`Requires` 和 `Wants` 表示一個正向(強制或可選)的需求和依賴關係，`Conflicts` 表示一個負向的需求和依賴關係。其他選項較少用到。如果一個 `unit` 需要啟動或關閉，`systemd` 就把它和它的依賴關係添加到臨時執行列表，然後確認它們的相互關係是否一致(或所有 `unit` 的先後順序是否含有迴圈)。如果答案是否的話，`systemd` 將嘗試修復它，刪除可以消除迴圈的無用工作。

#### 5.1.1.2. 相容性

➤ 與 `SysV` 初始化腳本相容：如果可能，它會利用 `LSB` 和 `chkconfig` 的頭資訊內容，否則，就使用其他可用資訊，如：`/etc/rc.d`。這些初始化腳本僅僅作為一個附加的配置源，以減少 `systemd` 服務固有的路徑數目。

➤ `/etc/fstab` 配置檔：這只是另一個配置源。通過使用 `comment=fstab` 選項標記 `/etc/fstab` 條目，使 `systemd` 可以控制自掛載點。

➤ 支持簡單的範本/實例機制：例如只有一個作為示例的 `getty@.service` 檔，而不是為六個 `getty` 都準備一個配置檔。介面部分甚至可以被直接繼承，也就是說，可以簡單的調用 `avahi-autoipd@eth0.service` 服務配置 `dhcpcd@eth0.service`，使得字串 `eth0` 的值可以直接通過通配符匹配得到。

➤ 在一定程度上相容 `/dev/initctl`。這個相容性實際上是為了執行 `FIFO-activated` 服務。(只是簡單地把原先的請求轉換成為 `D-Bus` 請求)事實上，這也意味著舊的 `Upstart` 和 `sysvinit` 中的 `shutdown`、`poweroff` 和其他相似

命令可以在 `systemd` 中繼續使用。

- 與 `utmp` 和 `wtmp` 相容。
- 它可以控制由它催生的每一個程式。
- 本地配置檔使用與 `.desktop` 檔相近的語法：很多軟體架構中都有這個簡

單的語法的分析器。它也可以借用已有的國際化的服務描述工具，語法都是相似的，沒有必要再學習新的語法。

### 5.1.2. 管理系統服務

`systemctl` 是最主要的工具。它融合 `service` 和 `chkconfig` 的功能於一體。

您可以使用它永久性或只在當前會話中啟用/禁用服務。

表 5-3 `service` 與 `systemctl` 的區別

<b>service</b>	<b>systemctl</b>	<b>描述</b>
<code>service name start</code>	<code>systemctl start name.service</code>	用來啟動一個服務(並不會重啟現有的)
<code>service name stop</code>	<code>systemctl stop name.service</code>	用來停止一個服務(並不會重啟現有的)
<code>service name restart</code>	<code>systemctl restart name.service</code>	用來停止並啟動一個服務
<code>service name condrestart</code>	<code>systemctl try-restart name.service</code>	重啟一個正在運行的服務
<code>service name reload</code>	<code>systemctl reload name.service</code>	當支持時, 重新裝載配置

		檔而不中斷等待操作
service name status	systemctl status name.service systemctl is-enabled name.service	查詢服務狀態
service --status-all	systemctl list-units --type=service --all	顯示所有服務狀態

**表 5-4 chkconfig 與 systemctl 的區別**

在下次啟動時或滿足其他觸發條件時設置服務為啟用	chkconfig name on	systemctl enable name.service
在下次啟動時或滿足其他觸發條件時設置服務為禁用	chkconfig name off	systemctl disable name.service
用來檢查一個服務在當前環境下被配置為啟用還是禁用。	chkconfig --list name	systemctl status name.service systemctl is-enabled name.service
輸出在各個運行級別下服務的啟用和禁用情況	chkconfig --list	systemctl list-unit-files --type service
顯示 unit 前置啟動服務	chkconfig --list	systemctl list-dependencies --after
顯示 unit 後導啟動服務	chkconfig --list	systemctl list-dependencies --before

務		
---	--	--

### 5.1.2.1. 顯示服務

輸出啟動的單元：

```
#systemctl
```

以下命令等效：

```
#systemctl list-units
```

如需輸出啟動服務的單元，執行以下命令：

```
#systemctl list-units --type service
```

輸出加載服務的單元，執行以下命令：

```
#systemctl list-units --type service --all
```

輸出所有服務的單元，執行以下命令：

```
#systemctl list-unit-files --type service
```

以下為顯示當前所有啟動服務的命令：

```
#systemctl list-units --type service
```

顯示所有已安裝服務單元命令如下：

```
#systemctl list-unit-files --type service
```

### 5.1.2.2. 顯示服務狀態

顯示服務狀態命令：

```
#systemctl status name.service
```

表 5-5 服務單元資訊列表

檔	描述
Loaded	服務是否加載
Active	服務是否啟動
Main PID	當前系統服務的 PID
Status	當前系統服務的附加資訊
Process	相關進程附加資訊
CGroup	相關 CGroup 附加資訊

顯示 firewalld.service 服務狀態示例：

```
#systemctl status firewalld.service
```

```
[root@localhost ~]# systemctl status firewalld.service
● firewalld.service - firewalld - dynamic firewall daemon
  Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
  Active: active (running) since Thu 2022-08-04 09:01:39 CST; 4h 51min ago
    Docs: man:firewalld(1)
  Main PID: 1093 (firewalld)
    Tasks: 2
   Memory: 8.7M
   CGroup: /system.slice/firewalld.service
           └─1093 /usr/bin/python3 /usr/sbin/firewalld --nofork --nopid
```

顯示 firewalld.service 服務啟動前置依賴服務示例：

```
#systemctl list-dependencies --after firewalld.service
```

```
[root@localhost ~]# systemctl list-dependencies --after firewalld.service
firewalld.service
● └─dbus.service
● └─dbus.socket
● └─polkit.service
● └─system.slice
● └─basic.target
● └─└─.mount
● └─└─systemd-ask-password-plymouth.path
● └─└─tmp.mount
● └─└─paths.target
● └─└─└─systemd-ask-password-console.path
● └─└─└─systemd-ask-password-wall.path
```

顯示 firewalld.service 服務啟動後導服務示例：

```
#systemctl list-dependencies --before firewalld.service
```

```
[root@localhost ~]# systemctl list-dependencies --before firewalld.service
firewalld.service
● └─docker.service
● └─multi-user.target
●   └─systemd-update-utmp-runlevel.service
●     └─graphical.target
●       └─systemd-update-utmp-runlevel.service
●         └─shutdown.target
●           └─shutdown.target
● └─network-pre.target
```

### 5.1.2.3. 啟動服務

啟動服務命令：

```
#systemctl start name.service
```

啟動 httpd 服務示例：

```
#systemctl start httpd.service
```

### 5.1.2.4. 停止服務

停止服務命令：

```
#systemctl stop name.service
```

停止 httpd 服務示例：

```
#systemctl stop httpd.service
```

### 5.1.2.5. 重啟服務

重啟服務命令：

```
#systemctl restart name.service
```

重啟 httpd 服務示例：

```
#systemctl restart httpd.service
```

僅重啟正在運行的服務命令：

```
#systemctl try-restart name.service
```

重載服務命令：

```
#systemctl reload name.service
```

重載 httpd 服務示例：

```
#systemctl reload httpd.service
```

#### 5.1.2.6. 啟用服務

啟用服務命令：

```
#systemctl enable name.service
```

重新啟用服務命令：

```
#systemctl reenable name.service
```

啟用 httpd 服務示例：

```
#systemctl enable httpd.service
Created symlink
/etc/systemd/system/multi-user.target.wants/httpd.service →
/usr/lib/systemd/system/httpd.service.
```

#### 5.1.2.7. 禁用服務

禁用服務命令：

```
#systemctl disable name.service
```

禁用 bluetooth 服務示例：

```
#systemctl disable bluetooth.service
Removed /etc/systemd/system/dbus-org.bluez.service.
Removed
/etc/systemd/system/bluetooth.target.wants/bluetooth.
```

```
service.
```

### 5.1.3. 管理目標

啟動級別（runlevel）是一個舊的概念。現在，systemd 引入了一個和啟動級別功能相似又不同的概念——目標（target）。不像數字表示的啟動級別，每個目標都有名字和獨特的功能，並且能同時啟用多個。一些目標繼承其他目標的服務，並啟動新服務。systemd 提供了一些模仿 sysvinit 啟動級別的目標，仍可以使用舊的 telinit 啟動級別命令切換。

啟動級別 0、1、3、5、6 都被賦予特定用途，並且都對應一個 systemd 的目標。要實現用戶自定義啟動級別功能，可以以原有的啟動級別為基礎，創建一個新的目標 `/etc/systemd/system/<新目標>`（可以參考 `/usr/lib/systemd/system/graphical.target`），創建 `/etc/systemd/system/<新目標>.wants` 目錄，向其中加入額外服務的鏈接（指向 `/usr/lib/systemd/system/` 中的單元檔）。

表 5-6 SysV 啟動級別與 systemd 目標對比

運行級別	目標單元	描述
0	runlevel0.target, poweroff.target	中斷系統（halt）
1	runlevel1.target, rescue.target	單用戶模式
2	runlevel2.target	用戶自定義啟動級別，通常識別為級別 3
3	runlevel3.target, multi-user.target	多用戶，無圖形介面。用戶可以通過



		終端或網路登錄
4	runlevel4.target, multi-user.target	用戶自定義啟動級別，通常識別為級別 3
5	runlevel5.target, graphical.target	多用戶，圖形介面。繼承級別 3 的服務，並啟動圖形介面服務
6	runlevel6.target, reboot.target	重啟
emergency	emergency.target	急救模式（Emergency shell）

表 5-7 SysV 初始化命令與 systemctl 對比

舊命令	新命令	描述
runlevel	systemctl list-units --type target	顯示當前目標
telinit runlevel	systemctl isolate name.target	變更當前目標

#### 5.1.3.1. 查看默認目標

查看默認目標命令：

```
#systemctl get-default
```

以下為查看默認目標單元的示例：

```
#systemctl get-default  
graphical.target
```

#### 5.1.3.2. 查看當前目標

查看當前目標命令：

```
#systemctl list-units --type target
```

輸出加載目標的單元，執行以下命令：

```
#systemctl list-units --type target --all
```

以下為顯示當前所有啟動目標的示例：

```
#systemctl list-units --type target
```

#### 5.1.3.3. 變更默認目標

變更默認目標命令：

```
#systemctl set-default name.target
```

以下變更多用戶目標示例：

```
#systemctl set-default multi-user.target
Remove /etc/systemd/system/default.target
Created symlink
/etc/systemd/system/default.target → /usr/lib/systemd/system/multi-user.target.
```

#### 5.1.3.4. 變更當前目標

變更當前目標命令：

```
#systemctl isolate name.target
```

以下變更多用戶目標示例：

```
#systemctl isolate multi-user.target
```

#### 5.1.3.5. 切換救援模式

`systemd.unit=rescue.target` 是一個設置基本系統和救援 shell 的特殊 target unit (與運行級 1 相似)；

進入救援模式命令：

```
#systemctl rescue
```

如果需要進入救援模式且不輸出日誌，輸以下命令：

```
#systemctl --no-wall rescue
```

切換救援模式示例：

```
#systemctlrescue
```

#### 5.1.3.6. 切換緊急模式

`systemd.unit=emergency.target` 與傳遞保留參數的 `init=/bin/sh` 給系統使系統從該狀態啟動相似。

進入緊急模式命令：

```
#systemctl emergency
```

如果需要進入緊急模式且不輸出日誌，輸以下命令：

```
#systemctl --no-wall emergency
```

#### 5.1.4. 在遠程機器上使用 **systemd**

連接遠程機器使用 `systemd` 命令如下：

```
#systemctl --host user_name@host_name command
```

遠程管理命令舉例：

```
#systemctl -H root@server-01.example.com status  
httpd.service
```

## 5.1.5. 創建和修改 **systemd** 單元檔

### 5.1.5.1. 單元檔介紹

單元檔通常包括三個部分：

【Unit】：通用配置項，包括該 unit 的基本資訊。

【Unit type】：Unit 類型，不同類型定義可以參考 **systemd** 簡介內容。

【Install】：包括需要被安裝、啟用和禁用的服務內容。

表 5-8 【Unit】字段配置項說明

配置項	描述
Description	一些描述，顯示給用戶介面看的，可以是任何字串，一般是關於服務的說明。
Documentation	指定參考文檔的列表，以空格分開的 URL 形式，如 <code>http://,https://,file:,info:,man</code> ，這是有順序的，最好是先解釋這個服務的目的是什麼，然後是它是如何配置的，再然後是其他檔，這個選項可以多次指定，會將多行的合併，如果指定了一個空的，那麼會重置此項，之前的配置不再起作用。
After/Before	配置服務間的啟動順序，比如一個 <code>foo.service</code> 包含了一行 <code>Before=bar.service</code> ，那麼當他們同時啟動時， <code>bar.service</code> 會等待 <code>foo.service</code> 啟動完成後才啟動。注意這個設置和 <code>Requires=</code> 是相互獨立的，同時包含

	<p><b>After=</b>和 <b>Requires=</b>也是常見的。此選項可以指定一次以上，這時是按順序全部啟動。</p>
<b>Requires</b>	<p>指定此服務依賴的其他服務，如果本服務被啟動，那麼 <b>Requires</b> 後面的服務也會被啟動，反之，如果 <b>Requires</b> 後面的服務被停止或無法啟動，則本服務也會停止。這個選項可以指定多次，那麼就要求所有指定的服務都被啟動。需要注意的是這個選項不影響啟動或停止的順序，啟動順序使用單句的 <b>After=</b>和 <b>Before=</b>來配置。例如，如果 <b>foo.service</b> 依賴 <b>bar.service</b>，但是只配置了 <b>Requires=</b>而沒有 <b>After=</b>或 <b>Before=</b>，那麼 <b>foo.service</b> 啟動時會同時啟動 <b>foo.service</b> 和 <b>bar.service</b>。通常使用 <b>Wants=</b>代替 <b>Requires=</b>是更好的選擇，因為系統會更好的處理服務失敗的情況。</p>
<b>Wants</b>	<p>相對弱化的 <b>Requires=</b>，這裏列出的服務會被啟動，但如果無法啟動或無法添加到事務處理，並不影響本服務做為一個整體的啟動。這是推薦的兩個服務關聯的方式。這種依賴也可以配置檔外，通過 <b>.wants/</b>目錄添加，具體可以看上面的說明。</p>
<b>Conflicts</b>	<p>配置一個依賴衝突，如果配置了某些項，那麼，當一個服務啟動時，或停止此處列出的服務，反過來，如果這</p>

	<p>裏列出的服務啟動，那麼本服務就會停止，即後啟動的才起作用。注意，此設置和 <b>After=</b> 和 <b>Before=</b> 是互相獨立的。如果服務 A 和 B 衝突，且在 B 啟動的時候同時啟動，那麼有可能會啟動失敗（兩者都是必需的）或修改以修復它（兩者之一或兩者都不是必需的），後一種情況，會將不需要的依賴刪除，或停止衝突。</p>
--	--

表 5-9 【 Service 】 字段配置項

配置項	描述
Type	<p>設置進程的啟動類型，必須是下列值之一：</p> <p>simple, forking, oneshot, dbus, notify, idle</p> <ul style="list-style-type: none"> <li>➢ 如果設為"simple"(設置了 ExecStart=但未設置 BusName=時的默認值)，那麼表示 ExecStart=所設定的進程就是該服務的主進程。</li> <li>➢ 如果此進程需要為其他進程提供服務，那麼必須在該進程啟動之前先建立好通信管道(例如套接字)，以加快後繼單元的啟動速度。</li> <li>➢ "dbus"(設置了 ExecStart=與 BusName=時的默認值)與"simple"類似，不同之處在於該進程需要在 D-Bus 上獲得一個由 BusName=指定的名稱。systemd 將會在啟動後繼單元之前，首先確保該進程已經成功的獲取了指定的 D-Bus 名稱。設置為此類型相當於隱含的依賴於</li> </ul>

dbus.socket 單元。

> "oneshot"(未設置 ExecStart=時的默認值)與 "simple"類似，不同之處在於該進程必須在 systemd 啟動後繼單元之前退出。此種類型通常需要設置 RemainAfterExit=選項。

> 如果設為"forking"，那麼表示 ExecStart=所設定的進程將會在啟動過程中使用 fork()系統調用。這是傳統 UNIX 守護進程的經典做法。也就是當所有的通信管道都已建好、啟動亦已成功之後，父進程將會退出，而子進程將作為該服務的主進程繼續運行。對於此種進程，建議同時設置 PIDFile=選項，以幫助 systemd 準確定位該服務的主進程，進而加快後繼單元的啟動速度。

> "notify"與"simple"類似，不同之處在於該進程將會在啟動完成之後通過 sd\_notify(3)之類的介面發送一個通知消息。systemd 將會在啟動後繼單元之前，首先確保該進程已經成功的發送了這個消息。如果設置為此類型，那麼 NotifyAccess=將只能設置為"all"或者"main"(默認)。注意，目前 Type=notify 尚不能在 PrivateNetwork=yes 的情況下正常工作。

> "idle"與"simple"類似，不同之處在於該進程將會被延遲到所有的操作都完成之後再執行。這樣可以避免控制

	<p>臺上的狀態資訊與 shell 腳本的輸出混雜在一起。</p>
ExecStart	<p>在啟動該服務時需要執行的命令行(命令+參數)。</p> <p>僅在設置了 <code>Type=oneshot</code> 的情況下，才可以設置任意個命令行(包括零個)，否則必須且只能設置一個命令行。</p> <p>多個命令行既可以在同一個 <code>ExecStart=</code> 中設置，也可以通過設置多個 <code>ExecStart=</code> 來達到相同的效果。</p> <p>如果設為一個空字元串，那麼先前設置的所有命令行都將被清空。如果不設置任何 <code>ExecStart=</code> 指令，那麼必須確保設置了 <code>RemainAfterExit=yes</code> 指令。</p> <p>命令行必須以一個絕對路徑表示的可執行檔開始，並且其後的那些參數將依次作為 <code>"argv[1] argv[2] ..."</code> 傳遞給被執行的進程。</p> <p>如果在絕對路徑前加上可選的 <code>"@"</code> 首碼，那麼其後的那些參數將依次作為 <code>"argv[0] argv[1] argv[2] ..."</code> 傳遞給被執行的進程。</p> <p>如果在絕對路徑前加上可選的 <code>"-"</code> 首碼，那麼即使該進程以失敗狀態(例如非零的返回值或者出現異常)退出，也會被視為成功退出。可以同時使用 <code>"-"</code> 與 <code>"@"</code> 首碼，且順序任意。</p> <p>如果設置了多個命令行，那麼這些命令行將以其在單元檔中出現的順序依次執行。</p> <p>如果某個無 <code>"-"</code> 首碼的命令行執行失敗，那麼剩餘的命令行將</p>



	<p>不會被執行，同時該單元將變為失敗(<b>failed</b>)狀態。</p> <p>當未設置 <b>Type=forking</b> 時，這裏設置的命令行所啟動的進程將被視為該服務的主守護進程。</p>
<b>ExecStop</b>	<p>這是一個可選的指令，用於設置當該服務被要求停止時所執行的命令行。語法規則與 <b>ExecStart=</b>完全相同。</p> <p>執行完此處設置的命令行之後，該服務所有剩餘的進程將會根據 <b>KillMode=</b>的設置被殺死(參見 <b>systemd.kill(5)</b>手冊)。</p> <p>如果未設置此選項，那麼當此服務被停止時，該服務的所有進程都將會根據 <b>KillMode=</b>的設置被立即全部殺死。</p> <p>與 <b>ExecReload=</b>一樣，也有一個特殊的環境變數<b>\$MAINPID</b>可以用於表示主進程的 <b>PID</b>。</p> <p>一般來說，僅僅設置一個結束服務的命令，而不等待其完成，是不夠的。</p> <p>因為當此處設置的命令執行完之後，剩餘的進程會被 <b>SIGKILL</b>信號立即殺死，這可能會導致數據丟失。</p> <p>因此，這裏設置的命令必須是同步操作，而不能是非同步操作。</p>
<b>ExecReload</b>	<p>這是一個可選的指令，用於設置當該服務被要求重新載入配置時所執行的命令行。語法規則與 <b>ExecStart=</b>完全相同。</p> <p>另外，還有一個特殊的環境變數<b>\$MAINPID</b>可以用於表示主進程的 <b>PID</b>，例如可以這樣使用：</p> <pre><b>/bin/kill -HUP \$MAINPID</b></pre>

	<p>注意，像上例那樣，通過向守護進程發送復位信號，強制其重新加載配置檔，並不是一個好習慣。</p> <p>因為這是一個非同步操作，所以不適用於需要按照特定順序重新加載配置檔的服務。</p> <p>我們強烈建議將 <code>ExecReload=</code> 設置為一個能夠確保重新加載配置檔的操作同步完成的命令行。</p>
Restart	<p>當服務進程正常退出、異常退出、被殺死、超時的時候，是否重新啟動該服務。</p> <p>"服務進程"是指 <code>ExecStart=,ExecStartPre=,ExecStartPost=,ExecStop=,ExecStopPost=,ExecReload=</code> 中設置的進程。</p> <p>當進程是由於 <code>systemd</code> 的正常操作(例如 <code>systemctl stop restart</code>)而被停止時，該服務不會被重新啟動。</p> <p>"超時"可以是看門狗的"keep-alive ping"超時，也可以是 <code>systemctl start reload stop</code> 操作超時。</p> <p>該選項可以取下列值之一：</p> <p><code>no,on-success,on-failure,on-abnormal,on-watchdog,on-abort,always</code></p> <p>"no"(默認值)表示不會被重啟。"always"表示會被無條件的重啟。</p>

	<p>"on-success"表示僅在服務進程正常退出時重啟，所謂"正常退出"是指： 退出碼為"0"，或者進程收到 SIGHUP,SIGINT,SIGTERM,SIGPIPE 信號並且退出碼符合 SuccessExitStatus=的設置。</p> <p>"on-failure"表示僅在服務進程異常退出時重啟，所謂"異常退出"是指： 退出碼不為"0"，或者進程被強制殺死(包括"core dump"以及 收到 SIGHUP,SIGINT,SIGTERM,SIGPIPE 之外的其他信號)， 或者進程由於看門狗或者 systemd 的操作超時而被殺死。</p>
RemainAfterExit	<p>可設為"yes"或"no"(默認值)，表示當該服務的所有進程全部退出之後，是否依然將此服務視為活動(active)狀態。</p>

表 5-10 【install】字段配置項

配置項	描述
Alias	<p>在安裝使用應該使用的額外名字（即別名）。名字必須和服務本身有同樣的尾碼（即同樣的類型）。這個選項可以指定多次，所有的名字都起作用，當執行 <code>systemctl enable</code> 命令時，會建立相應的鏈接。</p>
RequiredBy WantedBy	<p>在.wants/或.requires/子目錄中為服務建立相應的鏈接。這樣做的效果是當列表中的服務啟動，本服務也會</p>

	啟動。
Also	當此服務安裝時同時需要安裝的附加服務。如果用戶請求安裝的服務中配置了此項，則 <code>systemctl enable</code> 命令執行時會自動安裝本項所指定的服務。
DefaultInstance	表示 <code>unit</code> 啟用時默認的實例。

下麵是 `postfix.service` 單元檔的示例：

```
[Unit]
Description=Postfix Mail Transport Agent
After=syslog.target network.target
Conflicts=sendmail.service exim.service
[Service]
Type=forking
PIDFile=/var/spool/postfix/pid/master.pid
EnvironmentFile=/etc/sysconfig/network
ExecStartPre=/usr/libexec/postfix/aliasesdb
ExecStartPre=/usr/libexec/postfix/chroot-update
ExecStart=/usr/sbin/postfix start
ExecReload=/usr/sbin/postfix reload
ExecStop=/usr/sbin/postfix stop
[Install]
WantedBy=multi-user.target
```

這個示例中，【Unit】字段表述服務名稱與依賴衝突資訊。【Service】包括基本的服務資訊。`EnvironmentFile` 表述預定義的該服務的環境變數，`PIDFile` 表示服務使用靜態的PID。最後，【Install】字段顯示依賴該服務的內容。

### 5.1.5.2. 創建自定義單元檔

創建自定義單元檔的步驟：

1) 準備自定義服務的執行檔。

可執行檔可以是腳本，也可以是軟體提供者的程式，如果需要，為自定義服務的主進程準備一個 PID 檔，一保證 PID 保持不變。另外還可能需要的配置環境變數的腳本，確保所有腳本都有可執行屬性並且不需要交互。

2) 在 `/etc/systemd/system/` 目錄創建單元檔，並且保證只能被 root 用戶

編輯：

```
#touch /etc/systemd/system/{name}.service
#chmod 664 /etc/systemd/system/{name}.service
```

檔不需要執行許可權。

3) 打開 `{name}.service` 檔，添加服務配置，各種變數如何配置視所添加

的服務類型而定，下麵是一個依賴網路服務的配置例子：

```
[Unit]
Description=service_description
After=network.target
[Service]
ExecStart=path_to_executable
Type=forking
PIDFile=path_to_pidfile
[Install]
WantedBy=default.target
```

4) 通知 `systemd` 有個新服務添加：

```
#systemctl daemon-reload
#systemctl start name.service
```

創建 `emacs.service` 檔的例子：

- 1) 創建檔，並確保正確許可權：

```
#touch /etc/systemd/system/emacs.service  
#chmod 664 /etc/systemd/system/emacs.service
```

- 2) 添加配置資訊：

```
[Unit]  
Description=Emacs:theextensible,self-documentingtextedi  
tor  
[Service]  
Type=forking  
ExecStart=/usr/bin/emacs--daemon  
ExecStop=/usr/bin/emacsclient--eval"(kill-emacs)"  
Environment=SSH_AUTH_SOCKET=%t/keyring/ssh  
Restart=always  
[Install]  
WantedBy=default.target
```

- 3) 通知 systemd 並開啟服務：

```
#systemctl daemon-reload  
#systemctl start emas.service
```

創建 sshd-second 服務的例子：

- 1) 拷貝 sshd\_config 檔為 sshd-second.config

```
#cp /etc/ssh/sshd{,-second}_config
```

- 2) 編輯 sshd-second\_config 檔，添加 22220 的端口，和 PID 檔：

```
Port 22220  
PidFile /var/run/sshd-second.pid
```

如果還需要修改其他參數，請閱讀幫助。

3) 拷貝單元檔：

```
#cp /usr/lib/systemd/system/sshd{,-second}.service
```

4) 編輯單元檔/usr/lib/systemd/system/sshd-second.service

修改描述字段：

```
Description=OpenSSH server daemon
```

添加 sshd.service 服務在 After 關鍵字之後：

```
After=syslog.target    network.target    auditd.service
sshd.service
```

移除 sshdkey 創建：

```
ExecStartPre=/usr/sbin/sshd-keygen
```

在執行腳本裏，添加第二個 sshd 服務的配置檔：

```
ExecStart=/usr/sbin/sshd -D -f /etc/ssh/sshd-second_config
$OPTIONS
```

修改後的 sshd-second.service 檔內容如下：

```
ExecStart=/usr/sbin/sshd-D-f/etc/ssh/sshd-second_config
$OPTIONS
```

5) 如果使用 SELinux，添加 tcp 端口，負責 sshd-second 服務的端口就會被拒絕綁定：

```
#semanage port -a -t ssh_port_t -p tcp 22220
```

6) 設置開機啟動並測試：

```
#systemctl enable sshd-second.service  
#ssh -p 22220 user@server
```

確保防火牆端口也開放。

### 5.1.5.3. 修改已經存在的單元檔

`systemd` 單元配置檔默認保存在 `/usr/lib/systemd/system/` 目錄，系統管理員不建議直接修改這個目錄下的檔，自定義的檔在 `/etc/systemd/system/` 目錄下，如果有擴展的需求，可以使用以下方案：

創建一個目錄 `/etc/systemd/system/unit.d/`，這個是最推薦的一種方式，可以參考初始的單元檔，通過附件配置檔來擴展默認的配置，對默認單元檔的升級會被自動升級和應用。

從 `/usr/lib/systemd/system/` 拷貝一份原始配置檔到 `/etc/systemd/system/`，然後修改。複製的版本會覆蓋原始配置，這種方式不能增加附件的配置包，用於不需要附加功能的場景。

如果需要恢復到默認的配置檔，只需要刪除 `/etc/systemd/system/` 下的配置檔就可以了，不需要重啟機器，使用如下命令應用改變就可以：

```
#systemctl restart name.service
```

擴展默認單元配置檔

為了擴展默認的單元檔配置，需要先在 `/etc/systemd/system/` 下創建一個目錄，用 `root` 執行類似下麵的命令：

```
#mkdir /etc/systemd/system/name.service.d
```

在剛才創建的目錄之下創建配置檔，必須以 `.conf` 檔結尾。



例如創建一個自定義的依賴檔，內容如下：

```
[Unit]
Requires=new_dependency
After=new_dependency
```

另外一個例子，可以配置重啟的時候，在主進程退出後 30 秒在重啟，配置

例子如下：

```
[Unit]
Requires=new_dependency
After=new_dependency
```

推薦每次只產生一個小檔，每個檔只聚焦完善一個功能，這樣配置檔很容易被移除或者鏈接到其他服務的配置目錄中。

為了應用剛才的修改，使用 root 執行以下操作：

```
systemctl daemon-reload
systemctl restart name.service
```

例子：擴展 httpd.service 服務配置

為了使 httpd 服務啟動的時候執行用戶自定義的腳本，需要修改 httpd 的單元配置檔，執行以下幾步操作，首先創建一個自定義檔的目錄及自定義檔：

```
#mkdir /etc/systemd/system/httpd.service.d
#touch
/etc/systemd/system/httpd.service.d/custom_script.conf
```

假設自定義檔位置在 /usr/local/bin/custom.sh，將這個資訊添加到 custom\_script.conf 自定義腳本中：

```
[Service]
ExecStartPost=/usr/local/bin/custom.sh
```

應用更改：

```
#systemctl daemon-reload
#systemctl restart httpd.service
```

## 5.2. OpenSSH

SSH（Secure Shell）是一個使用客戶端-服務端架構，使得兩個系統之間的安全通信變得容易的協議，它能夠讓用戶遠程登錄到服務端主機系統中。和其他諸如 FTP 或者 Telnet 的遠程通信協議不同，SSH 加密了登錄會話，致使入侵者難以通過連接獲取未加密的密碼。

ssh 程式被設計用於替換諸如 telnet 或者 rsh，這些比較舊的、安全性不高的、用來登錄遠程主機系統的終端應用程式。與其相關的一個叫做 scp 的程式，用於替換諸如 rcp 這樣用來在主機之間複製檔的老程式。因為這些老舊的應用程式不會加密在客戶端和服務端之間傳遞的密碼，所以應該盡可能地避免使用它們。使用安全方法登錄遠程系統，能夠同時降低客戶端系統和遠程主機系統的風險。

銀河麒麟高級伺服器操作系統包含了通用的 OpenSSH 安裝包——openssh，以及 OpenSSH 服務端安裝包——openssh-server 和 OpenSSH 客戶端安裝包——openssh-clients。注意，OpenSSH 安裝包依賴於 OpenSSL 的安裝包 openssl-libs，這個包安裝了幾個重要的加密庫，使得 OpenSSH 能夠提供加密通信。

### 5.2.1. SSH 協議

#### 5.2.1.1. 為什麼使用 SSH？

潛在的入侵者有許多可以使用的工具，能夠讓他們中斷、攔截和重新路由網

路流量，從而試圖獲取對一個系統的訪問許可權。一般來說，這些威脅可以分為以下幾類：

### 1) 攔截兩個系統之間的通信

攻擊者可能位於通信雙方所在網路中的某處，複製他們之間傳遞的任何資訊。他可能會攔截並保留資訊，或者修改資訊後將其發送給計畫中的接收者。

這種攻擊通常是通過使用數據包嗅探器來進行的，數據包嗅探器是一種非常常見的網路工具，可以用來捕獲網路流中的數據包，並分析其內容。

### 2) 冒充特定主機

攻擊者的系統被配置來冒充傳送的預期接收者。如果攻擊者的策略成功了，用戶系統將不會發現它其實是在跟一個錯誤的主機進行通信。

這種攻擊可以通過使用一種名為“DNS 緩存投毒”的技術，或者通過所謂的“IP 欺騙”來實現。在第一種示例中，入侵者使用一臺被攻破的 DNS 伺服器來將客戶系統指向一臺惡意複製主機。在第二種示例中，入侵者發送偽造的網路數據包，讓這個數據包看起來好像是從一臺可信任的主機發出的。

這些技術都能夠攔截可能存在的敏感資訊，而且如果這些攔截是出於懷有敵意的原因，那麼結果將是災難性的。如果使用 SSH 來進行遠程登錄和文件複製，那麼就能夠極大地減少這些安全威脅。這是因為 SSH 客戶端和服務端使用數字簽名來驗證身份。此外，客戶端和服務端系統之間的所有通信都是加密的。不管嘗試在通信的哪一方進行身份欺騙都是行不通的，因為每一個數據包都是使用一個只有本地系統和遠程系統才知道的密鑰來加密的。

### 5.2.1.2. 主要特性

SSH 協議提供了以下保護措施：

#### 1) 無法偽裝預期的服務端

在初始連接之後，客戶端能夠驗證它當前所連接的服務端的確是它之前所連接過的同一個服務端。

#### 2) 無法捕獲認證資訊

客戶端使用強 128 位加密演算法來將它的認證資訊傳遞給服務端。

#### 3) 無法攔截通信

在一個會話中所有發送和接收的數據都是使用 128 位加密演算法加密的，使得攔截到的傳輸內容要解密閱讀是極度困難的。

此外，SSH 協議還提供了以下選項：

#### 1) 提供了在網路上使用圖形化應用程式的安全手段

通過使用名為“X11 轉發”的技術，客戶端能夠從服務端轉發 X11（X 窗口系統）應用程式。

#### 2) 提供了一種保護其他不安全協議的方式

SSH 協議對它發送和接收的一切進行加密。通過使用名為“端口轉發”的技術，SSH 服務端可以成為一個保護其他不安全協議（例如 POP）的導管，從而增加整體系統和數據的安全性。

#### 3) 可以用來創建安全通道

OpenSSH 服務端和客戶端可以被配置來為服務端和客戶端機器之間的網路流，創建一個類似於虛擬專用網路的隧道。

#### 4) 支持 Kerberos 認證

OpenSSH 服務端和客戶端可以被配置為使用 Kerberos 網路認證協議的 GSSAPI（通用安全服務應用程式編程介面）實現來進行認證。

#### 5.2.1.3. 協議版本

SSH 目前存在兩個版本：版本 1 和較新的版本 2。銀河麒麟高級伺服器操作系統中的 OpenSSH 套件使用 SSH 版本 2，該版本具有增強的密鑰交換演算法，不易受到版本 1 中已知漏洞的攻擊。然而，為了相容性的原因，OpenSSH 套件同樣也支持版本 1 的連接。

重要說明：

為了確保您的連接的最佳安全性，建議您只要可能就使用只相容 SSH 版本 2 的服務端和客戶端。

#### 5.2.2. SSH 連接的事件序列

以下一系列事件可以保護兩個主機之間的 SSH 通信的完整性。

- 1) 進行加密握手，以便客戶端能夠驗證它正在跟正確的服務端進行通信；
- 2) 採用對稱加密演算法對客戶端和遠程主機之間的連接的傳輸層進行加密；
- 3) 客戶端向服務端進行自我認證；
- 4) 客戶端通過加密連接和遠程主機進行交互。

#### 5.2.2.1. 傳輸層

傳輸層的主要角色是確保兩個主機之間的通信是安全可靠的，包括在認證的時候以及在隨後的通信期間。傳輸層通過對數據進行加密和解密處理，以及通過

提供對數據包發送和接收時的完整性保護，來實現這一目標。傳輸層也提供壓縮、加速資訊傳輸的功能。

SSH 客戶端聯繫服務端時，將進行密鑰交換，使得兩個系統之間能夠正確地構建傳輸層。密鑰交換時的步驟如下：

- 1) 交換密鑰；
- 2) 確定公鑰加密演算法；
- 3) 確定對稱加密演算法；
- 4) 確定消息認證演算法；
- 5) 確定哈希演算法。

在密鑰交換期間，服務端用一個唯一的主機密鑰向客戶端表明自己的身份。如果客戶端以前從未和這個特定的服務端通信過，服務端的主機密鑰對於客戶端來說是未知的，客戶端將不會連接。OpenSSH 通過接受服務端的主機密鑰來規避這個問題。用戶知曉，並且新的主機密鑰已經被接受和驗證之後，繼續後續過程。在之後的連接中，客戶端會用已保存的版本來檢查服務端的主機密鑰，以確保客戶端確實是在和預期的服務端通信。如果在將來主機密鑰發生了變化，用戶必須在連接之前刪除客戶端已經保存的版本。

重要說明：

攻擊者可能會在最初的聯繫階段偽裝成 SSH 服務端，因為本地系統並不知道預期的服務端和攻擊者設置的假服務端之間有什麼區別。為了防止這樣的事情發生，請在第一次連接或者主機密鑰不匹配時，聯繫服務端管理員以驗證 SSH 服務端的真實完整性。

SSH 被設計成幾乎可以和任何類型的公鑰演算法或者編碼格式相容。在最初的密鑰交換創建了一個用於交換的哈希值和一個共用的密值後，兩個系統立即開始計算新的密鑰和演算法，以保護將在連接上發送的認證和數據。

在使用指定的密鑰和演算法傳輸一定量（準確的量取決於 SSH 實現）的數據之後，將會發生又一次密鑰交換，生成另一套哈希值和一個新的共用密值。即使攻擊者能夠確定哈希值和共用密值，這一資訊也僅在非常有限的一段時間內有用。

#### 5.2.2.2. 認證

一旦傳輸層構建好一個安全隧道在兩個系統之間傳遞資訊，服務端告知客戶端其所支持的不同的認證方法，例如使用一個私鑰編碼的簽名，或者輸入一個密碼。然後客戶端嘗試使用所支持的其中一種方法向服務端進行認證。

SSH 服務端和客戶端可以配置使用不同類型的認證方式，讓各方都具有最佳的控制度。服務端可以決定基於其安全模型，它可以支持哪些加密方法；客戶端可以從可用的選項中選擇嘗試認證方法的順序。

#### 5.2.2.3. 通道

在 SSH 傳輸層完成一次成功的認證之後，將會通過被稱為“多路技術”（多路複用連接由多個信號組成，這些信號通過一個共用的、通用的介質進行發送。對 SSH 來說，不同的通道都在一個共同的安全連接中發送）的一種技術打開多重通道。每一條通道負責處理不同的終端會話和轉發 X11 會話。

不論是客戶端還是服務端都可以創建新的通道。每一個通道將在連接的兩端被賦予一個編號。當客戶端嘗試打開一個新的通道時，客戶端把請求和通道編號

一起發送出去。這些資訊被服務端保存起來，用於將通信導向對應的通道。如此一來，不同類型的會話不會相互影響，並且當一個指定的會話結束之後，關閉它的通道不會中斷主要的 SSH 連接。

通道也可以支持流控制，可以讓通道以一種有秩序的方式發送和接收數據。以這種方式，只有當客戶端接收到通道已經打開的消息，數據才會通過通道發送。

客戶端和服務端自動協商每條通道的特徵，取決於客戶端請求的服務類型以及用戶連接到網路的方式。這樣可以在不必改變協議的基礎設施的情況下，為處理不同類型的遠程連接帶來很大的靈活性。

### 5.2.3. 配置 OpenSSH

#### 5.2.3.1. 配置檔

配置檔有兩個不同的系列，一系列用於客戶端程式（例如 `ssh`、`scp` 和 `sftp`），另外一系列用於服務端（`sshd` 守護進程）。

系統範圍的 SSH 配置資訊保存在 `/etc/ssh/` 目錄下，詳見表 5-11 系統範圍的配置檔。特定用戶的 SSH 配置資訊保存在 `~/.ssh/` 目錄下（該目錄在特定用戶的家目錄裏），詳見表 5-12 特定用戶的配置檔。

表 5-11 系統範圍的配置檔

檔	描述
<code>/etc/ssh/moduli</code>	包含了 Diffie-Hellman 密鑰交換需要使用的 Diffie-Hellman 組， Diffie-Hellman 密鑰交換對於構建安



檔	描述
	全的傳輸層是關鍵的。當密鑰在一個 SSH 會話的最初階段被交換的時候，一個共用的密值被創建，該密值無法由任何單獨的一方所確定。這個密值隨後被用來提供主機認證。
/etc/ssh/ssh_config	默認的 SSH 客戶端配置檔。注意，如果 ~/.ssh/config 存在的話，該檔的配置將被 ~/.ssh/config 覆蓋。
/etc/ssh/sshd_config	sshd 守護進程的配置檔。
/etc/ssh/ssh_host_ecdsa_key	sshd 守護進程所使用的 ECDSA 私鑰。
/etc/ssh/ssh_host_ecdsa_key.pub	sshd 守護進程所使用的 ECDSA 公鑰。
/etc/ssh/ssh_host_ed25519_key	SSH 協議版本 1 的 sshd 守護進程所使用的 RSA 私鑰。
/etc/ssh/ssh_host_ed25519_key.pub	SSH 協議版本 1 的 sshd 守護進程所使用的 RSA 公鑰。
/etc/ssh/ssh_host_rsa_key	SSH 協議版本 2 的 sshd 守護進程所使用的 RSA 私鑰。
/etc/ssh/ssh_host_rsa_key.pub	SSH 協議版本 2 的 sshd 守護進程所使用的 RSA 公鑰。
/etc/pam.d/sshd	sshd 守護進程的 PAM 配置檔。

檔	描述
/etc/sysconfig/sshd	sshd 服務的配置文件。

表 5-12 特定用戶的配置文件

檔	描述
~/.ssh/authorized_keys	為服務端保留一個授權的公鑰列表。當客戶端連接到服務端時，服務端通過檢查保存在該檔中的它的簽名公鑰來認證客戶端。
~/.ssh/id_ecdsa	包含用戶的 ECDSA 私鑰。
~/.ssh/id_ecdsa.pub	用戶的 ECDSA 公鑰。
~/.ssh/id_rsa	SSH 協議版本 2 的 ssh 所使用的 RSA 私鑰。
~/.ssh/id_rsa.pub	SSH 協議版本 2 的 ssh 所使用的 RSA 公鑰。
~/.ssh/identity	SSH 協議版本 1 的 ssh 所使用的 RSA 私鑰。
~/.ssh/identity.pub	SSH 協議版本 1 的 ssh 所使用的 RSA 公鑰。
~/.ssh/known_hosts	包含用戶訪問的 SSH 服務端的主機密鑰。該檔對於確保 SSH 客戶端正在連接正確的 SSH 服務端是很重要的。

獲取有關 SSH 配置文件中所使用的各種指令的資訊，請參考 `ssh_config(5)` 和 `sshd_config(5)` 手冊頁面。

#### 5.2.3.2. 啟動 OpenSSH 服務端

您必須安裝 `openssh-server` 包之後才能運行 OpenSSH 服務端。

要在當前會話中啟動 `sshd` 守護進程，請以 `root` 用戶在 `shell` 命令行提示符下輸入以下命令：

```
#systemctl start sshd.service
```

要在當前會話中停止運行 `sshd` 守護進程，請以 `root` 用戶在 `shell` 命令行提示符下輸入以下命令：

```
#systemctl stop sshd.service
```

如果您想在系統啟動時自動啟動守護進程，請以 `root` 用戶在 `shell` 命令行提示符下輸入以下命令：

```
#systemctl enable sshd.service
Created symlink
/etc/systemd/system/multi-user.target.wants/sshd.service →
/usr/lib/systemd/system/sshd.service.
```

#### 5.2.3.3. 使用 SSH 進行遠程連接

為了讓 `SSH` 真正發揮作用，應該禁止使用不安全的連接協議。否則，用戶的密碼可能在一個會話中使用 `SSH` 時被保護得很好，但是卻在之後使用 `Telnet` 登錄時被捕獲了。需要禁用的服務包括 `telnet`、`rsh`、`rlogin` 和 `vsftpd`。

#### 5.2.3.4. 使用基於密鑰的認證

為了更進一步的提高系統安全，可以生成 `SSH` 密鑰對，然後強制使用基於密鑰的認證，並禁用密碼認證。要這樣做，請在 `vi` 或者 `nano` 等文本編輯器中打開 `/etc/ssh/sshd_config` 配置檔，並將 `PasswordAuthentication` 選項修改為如下內容：

## PasswordAuthentication no

如果您不是在一個新的默認安裝的系統中進行操作，請檢查配置檔確保沒有設置 `PubkeyAuthentication no` 選項。如果是遠程連接上的，而不是使用的控制臺或者帶外訪問，建議在禁用密碼認證之前先測試基於密鑰的登錄過程是否可用後再配置 `PubkeyAuthentication` 為 `no`。

為了能夠使用 `ssh`、`scp` 或者 `sftp` 從一個客戶端機器連接到服務端，請按照 5.2.3.5 生成密鑰對章節生成一個授權密鑰對。注意，這些密鑰必須針對每個用戶分別生成。

銀河麒麟高級伺服器操作系統 V10 默認使用版本 2 的 SSH 協議和 RSA 密鑰。

重要說明：

如果您以 `root` 的身份完成了步驟，那麼只有 `root` 用戶能夠使用這些密鑰。

備註：

如果您要重裝您的系統，又想保留之前生成的密鑰對，請備份 `~/.ssh/` 目錄。在重裝後，將其複製回您的家目錄。這一過程需要讓系統上的所有用戶執行，包括 `root` 用戶。

### 5.2.3.5. 生成密鑰對

要生成 SSH 協議版本 2 的 RSA 密鑰對，請按以下步驟操作：

1) 在 `shell` 命令行提示符下輸入以下命令生成 RSA 密鑰對：

```
$ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key
```

```
(/home/USER/.ssh/id_rsa):
```

- 2) 按回車鍵確認新創建的密鑰的默認路徑，即`~/.ssh/id_rsa`。
- 3) 輸入一個口令，並且在提示確認的時候再次輸入該口令。為了安全起見，請不要使用和您登錄您的帳戶相同的密碼。
- 4) 默認情況下，將`~/.ssh/`目錄的許可權設置為 `rwX-----` 或者以八進制標注表示的 `700`。這是為了確保只有對應的用戶 `USER` 能夠查看其內容。  
如果有必要，可以使用以下命令來進行確認：

```
$ls -ld ~/.ssh  
drwx-----. 2 USER USER 54 Nov 25 16:56 /home/USER/.ssh/
```

- 5) 使用以下格式的命令，將公鑰複製到一臺遠程機器上：

```
ssh-copy-id user@hostname
```

如果公鑰尚未安裝的話，該命令會複製最近一次修改的`~/.ssh/id*.pub`公鑰。可選的，您也可以指定公鑰的檔案名：

```
ssh-copy-id -i ~/.ssh/id_rsa.pub user@hostname
```

該命令會將`~/.ssh/id_rsa.pub`的內容複製到您想連接的機器的`~/.ssh/authorized_keys`檔中。如果`authorized_keys`檔已經存在了，密鑰將會追加到該檔的末尾。

要生成 SSH 協議版本 2 的 ECDSA 密鑰對，請按以下步驟操作：

- 1) 在 shell 命令行提示符下輸入以下命令生成 ECDSA 密鑰對：

```
$ssh-keygen -t ecdsa  
Generating public/private ecdsa key pair.
```

```
Enter file in which to save the key
(/home/USER/.ssh/id_ecdsa):
```

- 2) 按回車鍵確認新創建的密鑰的默認路徑，即`~/.ssh/id_ecdsa`。
- 3) 輸入一個口令，並且在提示確認的時候再次輸入該口令。為了安全起見，請不要使用和您登錄您的帳戶相同的密碼。
- 4) 默認情況下，將`~/.ssh/`目錄的許可權設置為 `rwX-----` 或者以八進制標注表示的 `700`。這是為了確保只有對應的用戶 `USER` 能夠查看其內容。

如果有必要，可以使用以下命令來進行確認：

```
$ls -ld ~/.ssh
drwx-----. 2 USER USER 54 Nov 25 16:56 /home/USER/.ssh/
```

- 5) 使用以下格式的命令，將公鑰複製到一臺遠程機器上：

```
ssh-copy-id user@hostname
```

如果公鑰尚未安裝的話，該命令會複製最近一次修改的`~/.ssh/id*.pub`公鑰。可選的，您也可以指定公鑰的檔案名：

```
ssh-copy-id -i ~/.ssh/id_ecdsa.pub user@hostname
```

該命令會將`~/.ssh/id_ecdsa.pub`的內容複製到您想連接的機器的`~/.ssh/authorized_keys`檔中。如果`authorized_keys`檔已經存在了，密鑰將會追加到該檔的末尾。

重要說明：

私鑰僅供您個人使用，絕不要把它給任何人，這一點是非常重要的。

#### 5.2.3.6. OpenSSH 客戶端

您必須安裝 `openssh-clients` 包後，才能從客戶端機器連接到一個

OpenSSH 服務端（請參見 4.2.4 安裝軟體包。瞭解如何在銀河麒麟高級伺服器操作系統中安裝新的包）。

#### 5.2.3.7. 使用 ssh 工具

ssh 工具可以讓您登錄到一臺遠程機器上，並在上面執行命令。它是對 rlogin、rsh 和 telnet 程式的一個安全替換。

和 telnet 命令相似，使用以下命令登錄到一臺遠程機器上：

```
ssh hostname
```

例如，要登錄到一臺名為 penguin.example.com 的遠程主機，可以在 shell 命令行提示符下輸入以下命令：

```
#ssh penguin.example.com
```

該命令將會以您正在使用的本地機器的用戶名登錄。如果您想指定一個不同的用戶名，請使用以下命令：

```
ssh username@hostname
```

例如，以 USER 登錄到 penguin.example.com，請輸入以下命令：

```
$ssh USER@penguin.example.com
```

您第一次連接時，將會看到和如下內容相似的資訊：

```
The authenticity of host 'penguin.example.com' can't be
established.
ECDSA key fingerprint is
SHA256:Ixy64icRYc/h7XS0vUVywS7t7ThtmOsPT1s07wDD5P8.
Are you sure you want to continue connecting
(yes/no/[fingerprint])?
```

在回答對話框中的問題之前，用戶應該始終檢查指印是否正確。用戶可以詢

問服務端的管理員以確認密鑰是正確的。這應該以一種安全的、事先約定好的方式進行。如果用戶可以使用服務端的主機密鑰，可以使用以下 `ssh-keygen` 命令來檢查指印：

```
#ssh-keygen -l -f /etc/ssh/ssh_host_ecdsa_key.pub
256
SHA256:b0gGbl+Xk/l+Ve76j3mqpYSty0n3gR9Qilsd+8oV3GI
no comment (ECDSA)
```

輸入 `yes` 接受密鑰並確認連接。您將會看到一個有關服務端已經被添加到已知的主機列表中的通告，以及一個輸入密碼的提示：

```
Warning: Permanently added 'penguin.example.com'
(ECDSA) to the list of known hosts.
USER@ penguin.example.com's password:
```

重要說明：

如果 SSH 服務端的主機密鑰改變了，客戶端將會通知用戶連接不能繼續，除非將服務端的主機密鑰從 `~/.ssh/known_hosts` 檔中刪除。然而，在進行此操作之前，請聯繫 SSH 服務端的系統管理員，驗證服務端沒有受到攻擊。

要從 `~/.ssh/known_hosts` 檔中刪除一個密鑰，可以使用如下命令：

```
#ssh-keygen -R penguin.example.com
```

在輸入密碼之後，您將會進入遠程主機의 `shell` 命令行提示符下。

可選地，`ssh` 程式可以用來在遠程主機上執行一條命令，而不用登錄到 `shell` 命令行提示符下：

```
ssh [username@]hostname command
```



例如，`/etc/kylin-release` 檔提供有關操作系統版本的資訊。要查看 `penguin.example.com` 上該檔的內容，輸入：

```
$ssh USER@penguin.example.com cat /etc/kylin-release
USER@penguin.example.com's password:
Kylin Linux Advanced Server release V10 (Lance)
```

在您輸入正確的密碼之後，將會顯示遠程主機的操作系統版本資訊，然後您將返回到本地的 `shell` 命令行提示符下。

### 5.2.3.8. 使用 `scp` 工具

`scp` 可以用來在主機之間通過一個安全加密的連接傳輸檔。在設計上，它和 `rcp` 非常相似。

要傳輸一個本地檔到遠程系統中，可以使用如下形式的命令：

```
scp localfile username@hostname:remotefile
```

例如，如果您想將 `taglist.vim` 傳輸到名為 `penguin.example.com` 的遠程主機用戶家目錄上，可以在 `shell` 命令行提示符下輸入以下命令：

```
#scp taglist.vim USER@penguin.example.com:~
```

一次可以指定多個檔。要傳輸 `.vim/plugin/` 目錄下的檔和目錄到遠程主機 `penguin.example.com` 的相同目錄下，可以輸入以下命令：

```
$scp -r .vim/plugin/*
USER@penguin.example.com:~:vim/plugin/
```

要將一個遠程的檔傳輸到本地系統上，可以使用以下語法：

```
scp username@hostname:remotefile localfile
```

例如，要從遠程主機上下載.vimrc 配置檔，可以輸入：

```
$scp USER@penguin.example.com:.vimrc .vimrc
```

#### 5.2.3.9. 使用 sftp 工具

sftp 工具可以用來打開一個安全的、互動式的 FTP 會話。在設計上，它類似於 ftp，不同之處在於 sftp 使用了一個安全的加密連接。

要連接到遠程系統中，可以使用如下形式的命令：

```
sftp username@hostname
```

例如，要使用 USER 用戶登錄到名為 penguin.example.com 的遠程主機上，可以輸入：

```
$sftp USER@penguin.example.com
USER@penguin.example.com's password:
Connected to penguin.example.com.
sftp>
```

當您輸入正確的密碼之後，您將會看到一個 sftp 的命令行提示符。sftp 工具可以使用一系列和 ftp 類似的命令（參見表 5- 13）。

**表 5-13 常用的 sftp 命令**

命令	描述
ls [directory]	列出一個遠程目錄的內容。如果沒有提供任何目錄名稱，默認使用當前的工作目錄。
cd directory	切換遠程工作目錄到指定的目錄下。
mkdir directory	創建一個遠程目錄。

命令	描述
<code>rmdir path</code>	刪除一個遠程目錄。
<code>put localfile [remotefile]</code>	將本地檔傳輸到遠程主機上。
<code>get remotefile [localfile]</code>	將遠程主機上的檔下載到本地主機上。

要獲取可用命令的完整列表，請參考 `sftp(1)` 用戶手冊頁面。

#### 5.2.4. 不只是一個安全的 Shell

一個安全的命令行介面只不過是 SSH 眾多使用方式的開端。給予合適的帶寬，可以通過 SSH 通道進行 X11 會話的轉發。或者，通過 TCP/IP 轉發，系統間以前的不安全端口連接可以映射到指定的 SSH 通道上。

##### 5.2.4.1. X11 轉發

要通過 SSH 連接打開 X11 會話，可以使用以下形式的命令：

```
ssh -Y username@hostname
```

例如，要使用 USER 用戶登錄到名為 `penguin.example.com` 的遠程主機上，可以輸入：

```
$ssh -Y USER@penguin.example.com
USER@penguin.example.com's password:
```

當從安全 shell 命令行提示符下運行一個 X 程式時，SSH 客戶端和服務端會創建一個新的安全通道，X 程式數據通過這個通道透明地發送到客戶端機器上。

注意，在發生 X11 轉發之前，必須在遠程系統中安裝好 X 窗口系統。以 root 用戶輸入以下命令可以安裝 X11 軟體包分組：

```
#dnf group install "X Window System"
```

要瞭解關於軟體包分組的更多資訊，請參見 4.2 管理軟體包。

X11 轉發非常有用。例如，X11 轉發可以用來為【列印設置】工具創建一個安全的互動式會話。要這樣做，先使用 ssh 連接到服務端，然後輸入：

```
$system-config-printer &
```

【列印設置】工具將會顯示出來，讓遠程用戶可以在遠程主機上安全地配置列印。

#### 5.2.4.2. 端口轉發

SSH 可以通過端口轉發安全加固其他不安全的 TCP/IP 協議。在使用這一技術時，SSH 服務端成為了 SSH 客戶端的一個加密導管。

端口轉發的工作原理是將客戶端的一個本地端口映射到服務端的一個遠程端口上。SSH 可以從服務端將任意端口映射到客戶端的任意端口上。這一技術並不需要端口號互相匹配。

重要說明：

要設置端口轉發監聽 1024 以下的端口，需要 root 級別的訪問許可權。

要創建一個監聽 localhost 上的連接的 TCP/IP 端口轉發通道，可以使用以下形式的命令：

```
ssh -L local-port:remote-hostname:remote-port  
username@hostname
```

例如，要使用 POP3 通過一個加密連接檢查名為 mail.example.com 的伺服器上的郵件，可以使用以下命令：

```
$ssh -L 1100:mail.example.com:110 mail.example.com
```

一旦客戶端機器和郵件伺服器之間的端口轉發通道準備就緒後，就可以使用一個 POP3 郵件客戶端在 localhost 上使用 1100 端口來檢查新郵件了。任何在客戶端系統上發往 1100 端口的請求，都將被安全地導向 mail.example.com 伺服器。

如果 mail.example.com 沒有運行 SSH 服務端，但是同一網路中的另一臺機器運行了 SSH 服務端，那麼 SSH 仍然可以用來對連接進行安全加固。當然，使用的命令會略有不同：

```
$ssh -L 1100:mail.example.com:110 other.example.com
```

在這一示例中，從客戶端機器的 1100 端口發出的 POP3 請求，將通過 22 端口上的 SSH 連接被轉發到 SSH 服務端 other.example.com。然後，other.example.com 連接到 mail.example.com 上的 110 端口來檢查新郵件。注意，在使用這一技術時，只有客戶端和 other.example.com 服務端之間的連接是安全的。

端口轉發也可以用來通過網路防火牆安全地獲取資訊。如果防火牆配置為允許放行使用標準端口（即 22 端口）的 SSH 數據流，但是阻塞了對其他端口的訪問，那麼在要在兩臺主機之間使用被阻塞端口建立連接仍然是可能的，只要將它們之間的通信通過一條建立好的 SSH 連接進行重定向即可。

重要說明：

以這種方式來使用端口轉發技術對連接進行轉發，將允許客戶端系統上的任何用戶都能連接到相應的服務上。如果客戶端系統被入侵，攻擊者也同樣能夠訪問轉發的服務。

擔心端口轉發的系統管理員，可以在服務端將`/etc/ssh/sshd_config` 檔中的 `AllowTCPForwarding` 選項設置為 `No`，並重啟 `sshd` 服務，來禁用端口轉發功能。

### 5.3. TigerVNC

TigerVNC ( Tiger Virtual Network Computing ) 是一個圖形化桌面共用系統，可以讓您遠程控制其他電腦。

TigerVNC 採用服務端-客戶端架構：服務端共用它的輸出 ( `vncserver` )，客戶端 ( `vncviewer` ) 連接到客戶端。

#### 重要說明：

相比以往銀河麒麟高級伺服器操作系統，銀河麒麟高級伺服器操作系統 V10 中的 TigerVNC 使用 `systemd` 系統管理守護進程進行配置。配置檔 `/etc/sysconfig/vncserver` 被替換成了 `/etc/systemd/system/vncserver@.service`。

#### 5.3.1. VNC 服務端

`vncserver` 工具用來啟動一個 VNC ( Virtual Network Computing ) 桌面。它將使用適當的選項運行 `Xvnc`，並在 VNC 桌面中啟動一個窗口管理器。`vncserver` 允許用戶在一臺機器上並行地運行多個獨立的會話，之後這些會話可以被任意數量的客戶端從任意位置訪問。

##### 5.3.1.1. 安裝 VNC 服務端

要安裝 TigerVNC 服務端，請以 `root` 用戶執行以下命令：

```
#dnf install tigervnc-server
```

### 5.3.1.2. 配置 VNC 服務端

VNC 服務端可以被配置了為一個或多個用戶（倘若這些用戶帳戶存在於系統上）啟動一個顯示，可以配置諸如顯示設置、網路地址和端口，以及安全設置等可選參數。

(1) 為指定用戶複製 vncserver@.service 檔到/etc/systemd/system 目錄

(以 smbuser 用戶為例)

```
#cp /usr/lib/systemd/system/vncserver@.service
/etc/systemd/system/vncserver-smbuser@.service
```

(2) 修改 service 檔將<USER>替換成實際的用戶

```
#vim /etc/systemd/system/vncserver-smbuser@.service
```

```
#The vncserver service unit file
#
#Quick HowTo:
#1. Copy this file to /etc/systemd/system/vncserver@.service
#2. Replace <USER> with the actual user name and edit vncserver
#   parameters appropriately
#3. Run `systemctl daemon-reload`
#4. Run `systemctl enable vncserver@:<display>.service`
#
#DO NOT RUN THIS SERVICE if your local area network is
#untrusted! For a secure way of using VNC, you should
#limit connections to the local host and then tunnel from
#the machine you want to view VNC on (host A) to the machine
#whose VNC output you want to view (host B)
#
#[user@hostA ~]#ssh -v -C -L 590N:localhost:590M hostB
#
```

```
#this will open a connection on port 590N of your hostA to hostB's
port 590M
#(in fact, it ssh-connects to hostB and then connects to localhost (on
hostB).
#See the ssh man page for details on port forwarding)
#
#You can then point a VNC client on hostA at vncdisplay N of localhost
and with
#the help of ssh, you end up seeing what hostB makes available on
port 590M
#
#Use "-nolisten tcp" to prevent X connections to your VNC server via
TCP.
#
#Use "-localhost" to prevent remote VNC clients connecting except
when
#doing so through a secure tunnel. See the "-via" option in the
#`man vncviewer' manual page.
```

```
[Unit]
```

```
Description=Remote desktop service (VNC)
```

```
After=syslog.target network.target
```

```
[Service]
```

```
Type=forking
```

```
WorkingDirectory=/home/smbuser
```

```
User=smbuser
```

```
Group=smbuser
```

```
PIDFile=/home/smbuser/.vnc/%H%i.pid
```

```
ExecStartPre=/bin/sh -c '/usr/bin/vncserver -kill %i > /dev/null 2>&1
```

```
|| :'
```



```
ExecStart=/usr/bin/vncserver -autokill %i
```

```
ExecStop=/usr/bin/vncserver -kill %i
```

```
Restart=on-success
```

```
RestartSec=15
```

```
[Install]
```

```
WantedBy=multi-user.target
```

(3) 執行 `systemctl daemon-reload`

### 5.3.1.3. 啟動 VNC 服務端

要啟動或者開機自啟動 VNC 服務，請在命令行中指定顯示編號。在 5.3.1.2 配置 VNC 服務端下的示例：“為單一用戶配置 VNC 顯示”中使用的配置檔擔任著範本的角色，檔中的 %i 將被 `systemd` 用顯示編號替換。使用一個有效的顯示編號來執行以下命令：

```
#systemctl start  
vncserver-USER_1@:display_number.service
```

您也可以讓服務在系統啟動時自動啟動。之後，當您登錄時，`vncserver` 已經自動啟動了。以 `root` 用戶執行以下命令：

```
#systemctl enable  
vncserver-USER_1@:display_number.service
```

這時候，其他用戶可以使用一個 VNC 查看程式，以及定義好的顯示編號和密碼來連接到 VNC 服務端。假若已經安裝好了一個圖形化桌面，將會顯示該桌面的一個實例。這個實例和目標機器上正在顯示的實例是不相同的。

為兩個用戶和兩個不同的顯示配置 VNC 服務端

對於兩個已經配置好的 VNC 服務端 `vncserver-USER_1@.service` 和

vncserver-USER\_2@.service, 您可以啟用不同的顯示編號。例如, 以下命令將會使 USER\_1 的 VNC 服務端在顯示編號 3 上啟動, 而 USER\_2 的 VNC 服務端將在顯示編號 5 上啟動:

```
#systemctl start vncserver-USER_1@:3.service
#systemctl start vncserver-USER_2@:5.service
```

#### 5.3.1.4. 終止 VNC 會話

類似於啟用 vncserver 的開機自啟動, 您也可以禁用該服務的開機自啟動:

```
#systemctl disable
vncserver-USER_1@:display_number.service
```

或者, 當您的系統正在運行時, 您可以以 root 用戶執行以下命令來停止 VNC 服務:

```
#systemctl stop
vncserver-USER_1@:display_number.service
```

#### 5.3.2. 共用一個已存在的桌面

默認情況下, 一個已登錄的用戶擁有一個由 X 服務端提供的、在顯示編號 0 上的桌面。用戶可以使用 TigerVNC 服務端的 x0vncserver 來共用他們的桌面。

##### 實例: 共用一個 X 桌面

要使用 x0vncserver 共用一個已登錄用戶的桌面, 請按以下步驟操作:

1) 以 root 用戶執行以下命令:

```
#dnf install tigervnc-server
```

2) 為用戶設置 VNC 密碼:

```
#vncpasswd
```

Password:

Verify:

3) 以已登錄用戶的身份執行以下命令：

```
#x0vncserver -PasswordFile=.vnc/passwd  
-AlwaysShared=1
```

倘若防火牆已經配置了允許連接 5900 端口，遠程查看器現在就可以連接到顯示編號 0 上，並查看已登錄用戶的桌面了。請參見 5.3.3.2 為 VNC 配置防火牆。

### 5.3.3. VNC 查看器

vncviewer 是一個用來顯示圖形用戶介面並遠程控制 vncserver 的程式。

為了操作 vncviewer，有一個包含許多條目的彈出菜單，通過這些條目可以執行諸如切換到/切換出全屏模式、退出查看器等多種操作。可選地，您也可以通過終端來操作 vncviewer。在命令行中輸入 vncviewer -h 可以列出 vncviewer 的參數。

#### 5.3.3.1. 安裝 VNC 查看器

要安裝 TigerVNC 的客戶端 vncviewer，請以 root 用戶執行以下命令：

```
#dnf install tigervnc
```

連接到 VNC 服務端

一旦配置好了 VNC 服務端，您就可以從任何 VNC 查看器連接到該服務端了。

#### 5.3.3.2. 為 VNC 配置防火牆

當使用非加密連接時，firewalld 可能會阻止連接。為了讓 firewalld 允許

VNC 數據包通過，您可以開放指定的 TCP 數據流端口。當使用 `-via` 選項時，數據流通過 SSH 做了重定向，而 SSH 的端口在 `firewalld` 中默認是開放的。

備註：

VNC 服務端的默認端口是 5900。要得到遠程桌面將被訪問的端口號，可以用默認端口號加上用戶分配的顯示編號。例如，對於第二個顯示幕： $2 + 5900 = 5902$ 。

#### 5.3.3.3. 使用 SSH 連接到 VNC 服務端

VNC 是一個很明顯的文本網路協議，在通信中沒有相應的安全機制來應對可能的攻擊。為了使通信安全，您可以通過使用 `-via` 選項來加密您的服務端-客戶端連接。這將會在 VNC 服務端和客戶端之間創建一個 SSH 隧道。

加密 VNC 服務端-客戶端連接的命令格式如下：

```
vncviewer -via user@host:display_number
```

## 第六章 伺服器

### 6.1. Web 伺服器

Web 伺服器能夠為用戶提供一種基於 web 的網路服務，通常以網頁形式呈現給用戶所需的網路資訊。基於插件機制，現在可以流覽更多的各種文檔了。因為 web 伺服器使用的是超文本傳輸協議（HTTP），所以也稱 web 伺服器為 HTTP 伺服器。

#### 6.1.1. Apache HTTP 伺服器

銀河麒麟高級伺服器操作系統中提供的可用 web 伺服器是 Apache HTTP 伺服器（httpd），httpd 的版本為 2.4，它是由 Apache 軟體基金會開發的一種開源 web 伺服器。

##### 6.1.1.1. 重要變更

httpd 服務控制

隨著遷移廢棄了 SysV 初始化腳本，系統管理員應該使用 apachectl 和 systemctl 命令來管理服務，而不是使用 service 命令了。下麵給出了查看 httpd 服務的例子：

命令

```
service httpd graceful
```

被替換成了

```
apachectl graceful
```

針對 httpd 的 systemd 單元檔和初始化腳本，有如下所示的不同操作：

- 當重新加載服務時，默認使用的是 graceful 重啟方式

- 當停止服務時，默認使用的是 graceful 停止方式

命令

```
service httpd configtest
```

被替換成了

```
apachectl configtest
```

私用 /tmp

為了提高系統安全性，systemd 單元檔運行 httpd 守護進程時，所使用的 /tmp 是私用的，和系統的 /tmp 是分開的。

配置佈局

當前系統中，用於模組加載的配置檔都存放在 /etc/httpd/conf.modules.d/ 目錄下。為 httpd 提供的一些可加載附加模組的配置檔也存放於此，如：php。/etc/httpd/conf/httpd.conf 檔中的主要配置項 Include 就是用來描述 /etc/httpd/conf.modules.d/ 目錄下的 include 檔的。這就意味著在 httpd.conf 的主體之前，先要處理 conf.modules.d 下的全部配置檔。在 /etc/httpd/conf.d 目錄下的檔，需要由 IncludeOptional 指示項，在 httpd.conf 檔的最後加以描述，也即是說，這些檔會在 httpd.conf 主體之後被處理。

由 httpd 軟體包提供的一些其他配置檔：

- /etc/httpd/conf.d/autoindex.conf --- 配置 mod\_autoindex 目錄索引
- /etc/httpd/conf.d/userdir.conf --- 配置可訪問用戶目錄，例如：

`http://example.com/~username/` ; 為了安全起見, 應該禁用這種默認訪問。

➤ `/etc/httpd/conf.d/welcome.conf` --- 在使用早期版本時, 當 `http://localhost/` 沒有內容的情況下, 就將其設置成了歡迎頁面。

### 默認配置

默認情況下, 提供的 `httpd.conf` 檔是最小配置。許多常見的配置項, 如: 以前默認配置中的 `Timeout` 或 `KeepAlive` 都不再被明確設置了; 硬編碼設置也被取代了。針對所有配置指示項的硬編碼設置在手冊中有詳細說明。

### 不相容的語法變更

如果需要從現有的 `httpd 2.2` 配置遷移到 `httpd 2.4`, 則存在有大量前後不相容的 `httpd` 配置語法需要修改。更多有關升級部分的 `Apache` 文檔, 請參看 <http://httpd.apache.org/docs/2.4/upgrading.html> 。

### 處理模型

在銀河麒麟高級伺服器操作系統的早期版本中, 不同的多處理模型 (MPM) 提供了不同的 `httpd` 二進位檔。如: 基於子進程模型可以用“`prefork`”代表 `/usr/sbin/httpd` ; 基於線程模型可以用“`worker`”代表 `/usr/sbin/httpd.worker` 。

在銀河麒麟高級伺服器操作系統中, 只使用了單一的 `httpd`。3 個 MPM 作為可加載模組還是有效的: `worker`, `prefork` (default) 和 `event`, 當需要加載一個 MPM 模組時, 可以通過編輯 `/etc/httpd/conf.modules.d/00-mpm.conf` 配置檔, 把相關 MPM 模組前的注釋符 `#` 去掉, 就可以實現 MPM 模組的加載了。

## 包的變更

由獨立的分包 `mod_ldap` 實現模組的 LDAP 身份驗證和授權。由新的分包 `mod_session` 提供模組 `mod_session` 和 `helper`。由新的分包 `mod_proxy_html` 提供模組 `mod_proxy_html` 和 `mod_xml2enc`。這些軟體包都可以從可選頻道中獲得。

## 打包檔系統佈局

不再使用 `/var/cache/mod_proxy/` 目錄，取而代之的是 `/var/cache/httpd/` 目錄下的 `proxy` 和 `ssl` 子目錄。

`httpd` 軟體包所提供的打包內容已經從 `/var/www` 遷移到了 `/usr/share/httpd/`。

> `/usr/share/httpd/icons/` --- 包含了一組用於目錄索引的圖示。早期版本在 `/var/www/icons/` 目錄下，現在遷移到了 `/usr/share/httpd/icons` 目錄下。在默認情況下，`http://localhost/icons/` 是有效的。在 `/etc/httpd/conf.d/autoindex.conf` 檔中，可以配置圖示的位置和可用性。

> `/usr/share/httpd/manual/` --- `/var/www/manual/` 目錄已經遷移到了 `/usr/share/httpd/manual/` 目錄。這個目錄包含了來自於 `httpd-manual` 軟體包的內容。包含了 `httpd` 的 HTML 版手冊。如果安裝了這個軟體包，則 `http://localhost/manual/` 是有效的。在 `/etc/httpd/conf.d/manual.conf` 檔中，可以配置手冊的位置和可用性。

> `/usr/share/httpd/error/` --- `/var/www/error/` 目錄已經遷移到了 `/usr/share/httpd/error/` 目錄。在默認配置中，已刪除了自定義的多國語言



HTTP 錯誤資訊包的配置。在 `/usr/share/doc/httpd-VERSION/httpd-multilang-errordoc.conf` 檔中，提供了配置檔樣例。

### 身份驗證，授權和訪問控制

用於控制身份驗證、授權和訪問控制的配置指令已發生了明顯的改變。在現有配置檔中使用的 `Order`、`Deny` 和 `Allow` 指令應適用於新的 `Require` 語法。有關 Apache 文檔的更多詳細資訊，請查看 <http://httpd.apache.org/docs/2.4/howto/auth.html>。

### suexec

為了提高系統的安全性，已不再安裝 `suexec` 二進位了，取而代之的是檔系統能力位集，允許更嚴格的許可權設置。結合這一改變，`suexec` 也不再使用 `/var/log/httpd/suexec.log` 日誌檔了。相反，日誌資訊都送到了 `syslog`；默認情況下，將出現在 `/var/log/secure` 日誌檔中。

### 模組介面

由於 `httpd` 改變了模組介面定義，因此，基於 `httpd 2.2` 構建的第 3 方模組對 `httpd 2.4` 是不相容的。這樣，就需要根據 `httpd 2.4` 模組介面定義，對那些模組代碼做必要的修改，最後，再重構。有關 `httpd2.4` API 變更的詳細資訊列表，請參看

[http://httpd.apache.org/docs/2.4/developer/new\\_api\\_2\\_4.html](http://httpd.apache.org/docs/2.4/developer/new_api_2_4.html)。

用於從源構建模組的 `apxs` 可執行檔已經從 `/usr/sbin/apxs` 遷移到了 `/usr/bin/apxs`。

### 被刪除模組

在銀河麒麟高級伺服器操作系統中，已經被刪除的 `httpd` 模組列表如下：

`mod_auth_mysql`, `mod_auth_pgsq`

在 `mod_authn_dbd` 中，`httpd 2.4` 提供了內部的 SQL 資料庫認證機制。

`mod_perl`

在 `httpd 2.4` 中，不再正式支持 `mod_perl` 了。

`mod_authz_ldap`

在 `httpd 2.4` 的分包 `mod_ldap` 中，用 `mod_authz_ldap` 提供了對 LDAP 的支持。

#### 6.1.1.2. 更新配置

為了更新 Apache HTTP Server version 2.2 配置檔，需要完成以下步驟：

1. 由於模組名稱有可能變更了，因此，需要先確認所有模組名稱是否正確。針對已經變更了名稱的每個模組，需要有針對性地調整 `LoadModule` 指示項；
2. 在需要加載第三方模組前，需要重新編譯它們。這通常意味著需要身份驗證和授權模組；
3. 如果要使用 `mod_userdir` 模組，則在 `UserDir` 指示項中，需要提供目錄名稱（通常為 `public_html`）；
4. 如果要使用 Apache HTTP 安全伺服器的話，則有關啟用安全套接字層（SSL）協議的更多重要資訊，請參看 6.1.1.8 啟用 `mod_ssl` 模組。

可以用以下命令，檢查配置檔中可能出現的錯誤。

```
#apachectl configtest
Syntax OK
```

有關如何將 Apache HTTP 伺服器的配置從 2.2 版本升級 2.4 版本，請參看

<http://httpd.apache.org/docs/2.4/upgrading.html> 。

### 6.1.1.3. 運行 httpd 服務

本章描述如何啟動，停止和重啟 Apache HTTP，以及如何檢查 Apache HTTP 的當前狀態。為了能使用 httpd 服務，應先用以下命令確認是否已經安裝了 httpd。

```
#dnf install httpd
```

通常，在銀河麒麟高級伺服器操作系統中，有關一些目標概念和如何管理系統服務的更多資訊，請參看 5.1 使用 systemd 管理系統服務。

#### 啟動服務

由 root 用戶執行以下命令，啟動 httpd 服務：

```
#systemctl start httpd.service
```

執行以下命令，可以使得在系統引導時，自動啟動 httpd 服務：

```
#systemctl enable httpd.service
Created symlink from
/etc/systemd/system/multi-user.target.wants/httpd.service to
/usr/lib/systemd/system/httpd.service.
```

#### 備註：

如果要以安全伺服器方式啟動 Apache HTTP 伺服器，則操作系統啟動後，會提示需要一個用 SSL 私鑰加密的密碼。

#### 停止服務

由 root 用戶執行以下命令，停止 httpd 服務：

```
#systemctl stop httpd.service
```

執行以下命令，可以避免 httpd 服務隨操作系統自啟動：

```
#systemctl disable httpd.service
Removed                               symlink
/etc/systemd/system/multi-user.target.wants/httpd.service.
```

重啟服務

有以下 3 種方法，可以重啟 httpd 服務：

由 root 用戶執行以下命令，完全重啟服務

```
#systemctl restart httpd.service
```

這裏，首先要停止這個運行著的 httpd 服務，然後再重啟它。通常，在安裝或刪除一個動態加載模組（如：PHP）時，會用這個命令。

由 root 用戶執行以下命令，可以重新加載配置：

```
#systemctl reload httpd.service
```

這將會導致運行著的 httpd 服務去重新加載它的配置檔。當前正在處理的任何請求都將會被中斷，客戶端瀏覽器上也會看到有錯誤資訊提示或頁面不完整。

為了重新加載配置而又不影響當前執行著的請求，可以由 root 用戶執行以下命令：

```
#apachectl graceful
```

這將會導致運行著的 httpd 服務去重新加載它的配置檔，當前正在處理的任何請求都將會沿用舊的配置繼續處理。

在銀河麒麟高級伺服器操作系統中，有關如何管理系統服務的更多資訊，請參看 5.1 使用 systemd 管理系統服務。

### 驗證服務狀態

在 shell 提示符下，執行以下命令，可以檢查運行著的 httpd 服務的狀態：

```
#systemctl is-active httpd.service
active
```

#### 6.1.1.4. 編輯配置檔

默認情況下，當啟動 httpd 時，會讀取表 6-1 httpd 服務配置檔中列出的 httpd 服務配置檔。

**表 6-1 httpd 服務配置檔**

路 徑	描 述
/etc/httpd/conf/httpd.conf	主要配置檔。
/etc/httpd/conf.d/	包含在主配置檔中的其他配置檔的所在目錄。

雖然，默認配置能適合於大多數情況的使用，然而，熟悉一些其他更重要的配置選項也是很有必要的。注意：為了使任一修改都能生效，完成修改後必須重啟 web 伺服器。想瞭解如何重啟 httpd 服務的更多資訊，請參看 6.1.1.3 運行 http 的服務中的“重啟服務”。

可以用以下命令，檢查配置檔中可能出現的錯誤：

```
#apachectl configtest
Syntax OK
```

解決錯誤的簡單方法就是將配置檔恢復到修改前的狀態。

### 6.1.1.5. 使用模組

作為模組化結構的應用，**httpd** 伺服器發佈時帶有大量的動態共用對象（**DSOs**），根據需要它們可以在運行中被動態加載或卸載。在銀河麒麟高級伺服器操作系統中，這些模組被存放在 `/usr/lib64/httpd/modules/` 目錄下。

#### 加載模組

為了加載指定的 **DSO** 模組，需要用到 **LoadModule** 指示項。注意：以獨立軟體包形式提供的模組，通常，在 `/etc/httpd/conf.d/` 目錄下都有它自己的配置檔。

#### 實例：加載模組

```
LoadModule ssl_module modules/mod_ssl.so
```

完成後，需要重啟 **web** 伺服器來加載模組。有關如何重啟 **httpd** 服務的更多資訊，請參看 6.1.1.3 運行 **http** 的服務中的“重啟服務”。

#### 寫模組

如果要創建新的 **DSO** 模組，則需要安裝 **httpd-devel** 軟體包。執行以下命令完成安裝。

```
#dnf install httpd-devel
```

這個軟體包包含有構建模組所需的 **include** 檔，**header** 檔和編譯生成工具 **Apache eXtenSion(apxs)** 應用程式。

一旦寫好源碼，就可用以下命令來構建模組了：

```
#apxs -i -a -c module_name.c
```

模組生成後，就可以用加載 **Apache HTTP** 伺服器其他模組的方法來加載它。

### 6.1.1.6. 設置虛擬主機

可以使用 Apache HTTP 伺服器的內置虛擬主機特性，實現基於不同 IP，主機名和端口號提供的不同網路資訊服務。

為了設置基於主機名的虛擬主機，可以拷貝 `/usr/share/doc/httpd/httpd-vhosts.conf` 配置檔到 `/etc/httpd/conf.d/` 目錄下，並替換“`@@Port@@`”和“`@@ServerRoot@@`”為相應服務端口和 web 根目錄。根據需要自定制的選項，如下圖實例所示。

#### 實例：虛擬主機配置實例

```
<VirtualHost *:80>
    ServerAdmin webmaster@penguin.example.com
    DocumentRoot "/www/docs/penguin.example.com"
    ServerName penguin.example.com
    ServerAlias www.penguin.example.com
    ErrorLog "/var/log/httpd/penguin.example.com-error_log"
    CustomLog  "/var/log/httpd/penguin.example.com-access_log"
    common
</VirtualHost>
```

注意 `ServerName` 必須是有效的 DNS 名。`<VirtualHost>` 容器的可自定義性很好，可以接受主伺服器中大多數指示項的配置。容器中包含的 `User` 和 `Group` 在此不支持了，取而代之的是 `SuexecUserGroup`。

備註：

如果配置的虛擬主機監聽端口不是默認的，則需要確認是否根據要求，修改了 `/etc/httpd/conf/httpd.conf` 中的全局指示項 `Listen`。

為了啟動新配置的虛擬主機，需要重啟 web 伺服器。有關如何重啟 httpd

服務的更多資訊，請參看 6.1.1.3 運行 http 的服務中的“重啟服務”。

#### 6.1.1.7. 創建 SSL 伺服器

安全套接字層（SSL）是一種能使服務器和客戶端進行安全數字通訊的加密協議。傳輸層安全（TLS）協議就是 SSL 的擴展和改進版本，能夠確保隱私和數據的安全性和完整性。Apache HTTP 服務器和 mod\_ssl 模組的結合，使得使用了 OpenSSL 工具包的模組，能夠提供對 SSL/TLS 的支持，通常稱它為 SSL 伺服器。銀河麒麟高級服務器操作系統也支持基於 TLS 實現的 Mozilla NSS。mod\_nss 模組提供了對 Mozilla NSS 的支持。

不像 HTTP 連接，任何能夠攔截到它的人都可以讀和修改它。所謂 HTTPS 就是在 HTTP 上增加了對 SSL/TLS 的支持，它能保護傳輸內容不被竊聽和修改。本章主要介紹在 Apache HTTP 服務器配置上，如何啟用此類模組的一些基本方法，指導如何生成私鑰和自簽證書的過程。

#### 概述證書和安全

密鑰主要用於安全通訊。在傳統或對稱加密技術中，事務的兩端運用相同的密鑰來解碼彼此的傳輸。然而，在公共或非對稱加密技術中，有 2 個密鑰共存：一個是要保密的私鑰；另一個是可以共用的，公共公鑰。使用公鑰編碼的數據只能用對應的私鑰來解碼；用私鑰編碼的數據只能用對應的公鑰來解碼。

為了基於 SSL 實現安全通訊，SSL 服務器必須要使用由證書頒發機構（CA）簽名的數字證書。證書中包含有服務器的各種屬性（例如：服務器主機名；公司名；地理位置等等），數字簽名使用的是 CA 的私鑰。這個簽名可以確保特定證書頒發機構已簽名了證書，並且不能以任何方式修改它。



當瀏覽器基於 web 要建立新的 SSL 連接時，會檢查 web 伺服器提供的證書。如果證書沒有被可信 CA 機構簽名，或者證書中的主機名和連上來的主機名不匹配，web 伺服器會拒絕建立通訊，通常用戶也會收到相應的錯誤資訊。

默認情況下，大多數瀏覽器都配置有一組被廣泛使用的合法簽名證書。正因為如此，設置安全伺服器時，總可以選擇一個合適的 CA 證書，這樣，最終用戶就可以建立可信連接了，否則，會收到相應的錯誤資訊，這時，需要人為接受一個證書。由於用戶可以忽略證書錯誤，但是，卻會使得攻擊者可以攔截連接，因此，強力推薦盡可能使用可信 CA。相關的更多資訊，請參看表 6-2 查看瀏覽器中通常可用的 CA。

表 6-2 查看瀏覽器中通常可用的 CA

web 瀏覽器	鏈 接
Mozilla Firefox	Mozilla root CA 列表
Opera	Opera 可用的根證書
Internet Explorer	Microsoft Windows 可用的根證書
Chromium	Chromium project 可用的根證書

在建立 SSL 伺服器時，需要生成一個證書請求和一個密鑰，然後，發送證書請求，公司的身份證明和支付到證書頒發機構。一旦 CA 驗證了您的證書請求和身份證明，它將會發送一個可以和服務器一起使用的簽名證書給您。另外，可以創建一個自簽名的證書，它不包含有 CA 的數字簽名，因此，這僅僅適用於以測試為目的場合。

### 6.1.1.8. 啟用 mod\_ssl 模組

如果想基於 mod\_ssl 模組，建立一個 SSL 或 HTTPS 伺服器的話，則不能有另外一個應用或模組（如：mod\_nss）使用相同的端口。端口 443 是默認 HTTPS 端口。

為了使用 mod\_ssl 模組和 OpenSSL 工具包，建立一個 SSL 伺服器，則需要安裝 mod\_ssl 和 openssl 軟體包。由 root 用戶執行以下命令就可以完成安裝：

```
#dnf install mod_ssl openssl
```

此時，會創建 mod\_ssl 的配置檔/etc/httpd/conf.d/ssl.conf，默認情況下，它包含在 Apache HTTP 伺服器的主配置檔中。為了加載模組，需要重啟 httpd 服務，請參看 6.1.1.3 運行 http 的服務中的“重啟服務”。

重要說明：

正如 POODLE: SSLv3 vulnerability (CVE-2014-3566)中所描述的，Kylin 不推薦啟用 ssl，建議僅使用 TLSv1.1 或 TLSv1.2。使用 TLSv1.1 可以實現向後相容。雖然許多產品多已支持使用 SSLv2 或 SSLv3 協議了，並默認啟用了它們，但是，現在還是不適合強力推薦使用它們。

在 mod\_ssl 中啟用和禁用 SSL 和 TLS

為了啟用和禁用指定版本的 SSL 和 TLS 協議，既可以在配置檔中，通過在“##SSL Global Context”部分添加/刪除 SSLProtocol 指示項來全局啟用/禁用 SSL，也可以在每個“VirtualHost”的“#SSL Protocol support”中單獨編輯默認項來實現。如果沒有在每個域的主機部分指定它，則將會繼承全局部分的

設置。為了禁用一個協議版本，管理員既可以僅在“SSL Global Context”部分來指定 SSLProtocol，也可以在每個域的虛擬主機部分指定它，注意要帶上參數 all。

#### 6.1.1.9. 使用存在的密鑰和證書

可以用已有密鑰和證書來配置 SSL 伺服器，而無需創建新的。但是，也有 2 種情形是不可以的：

##### 1. 正在修改 IP 或功能變數名稱

證書是針對特定的 IP 和功能變數名稱而生成的。因此，只要其中之一發生了變化，證書就會無效。

##### 2. 正在更改由 VeriSign 頒發了證書的伺服器軟體

VeriSign 是一個常用的證書頒發機構，它會針對特定的軟體，IP 地址和功能變數名稱頒發證書。一旦更改了軟體產品，證書就會無效。

無論上述哪種情況發生，都需要重新生成證書。有關更多相關資訊，請參看

#### 6.1.1.9 生成新密鑰和證書。

如果要使用已有的密鑰和證書，則要分別遷移相關檔到 /etc/pki/tls/private/ 和 /etc/pki/tls/certs/ 目錄下。可以執行以下命令實現：

```
#mv key_file.key /etc/pki/tls/private/hostname.key
#mv certificate.crt /etc/pki/tls/certs/hostname.crt
```

然後，在 /etc/httpd/conf.d/ssl.conf 配置檔中，添加下列行：

```
SSLCertificateFile /etc/pki/tls/certs/hostname.crt
SSLCertificateKeyFile /etc/pki/tls/private/hostname.key
```

為了使更新的配置馬上生效，請參看 6.1.1.3 運行 http 的服務中的重啟服務。

例如：使用來自 Kylin 安全 web 伺服器的密鑰和證書，如下所示：

```
#mv /etc/httpd/conf/httpsd.key
/etc/pki/tls/private/kylinos.example.com.key
#mv /etc/httpd/conf/httpsd.crt
/etc/pki/tls/certs/kylinos.example.com.crt
```

#### 6.1.1.10. 生成新密鑰和證書

為了生成新密鑰和證書，在系統中必須安裝 OpenSSL 軟體包。由 root 用戶執行以下命令可以完成安裝：

```
#dnf install openssl
```

這個軟體包提供了一組生成和管理 SSL 證書和密鑰的工具及 `genkey` 應用程式，它能幫助我們生成密鑰。

重要說明：

如果想用一個新證書替換掉已有的有效證書，那就只需指定一個不同的序列號。這樣，可以確保客戶端瀏覽器知曉證書更新了，避免訪問頁面出錯。為了用自定義序列號創建新證書，需要由 root 用戶用以下命令為 `hostname.key` 重新生成證書：

```
#openssl req -x509 -new -set_serial number -key hostname.key
-out hostname.crt
```

備註：

在系統中，如果有針對特指主機名的密鑰檔存在的話，要由 root 用戶執行以下命令，刪除此密鑰檔：

```
#rm /etc/pki/tls/private/hostname.key
```

### 6.1.1.11. 為 HTTP 和 HTTPS 配置防火牆

默認情況下，銀河麒麟高級伺服器操作系統是不啟用 HTTP 和 HTTPS 的。

把系統配置成一個 web 伺服器後，利用 firewalld 服務，通過防火牆，就可以啟用 HTTP 和 HTTPS 了。

由 root 用戶執行以下命令，可以啟用 HTTP：

```
#firewall-cmd --add-service http
success
```

由 root 用戶執行以下命令，可以啟用 HTTPS：

```
#firewall-cmd --add-service https
success
```

注意系統重啟後，這些修改就失效了。為了使這些修改永久生效，只需要在執行上述命令時，加上--permanent 選項。

檢查 HTTP 和 HTTPS 的網路訪問許可

由 root 用戶執行以下命令，可以檢查通過防火牆允許訪問的那些配置：

```
#firewall-cmd --list-all
```

這是一個默認安裝的例子，雖然防火牆啟用了，但是，HTTP 和 HTTPS 是不允許通過防火牆被訪問的。

一旦允許通過防火牆可以訪問 HTTP 和 HTTPS 了，那麼執行上述命令，就會看到 services 行上會有如下的資訊：

```
services: dhcpv6-client http https ssh
```

## 6.2. 目錄伺服器

### 6.2.1. OpenLDAP

LDAP（輕量目錄訪問協議）是一組開放的協議，用來訪問在網路上集中存

儲的資訊。它基於 X.500 目錄共用標準，但是更少的複雜度和資源密集型，所以，LDAP 被稱作 X.500 標準基礎上產生的一個簡化版本。

像 X.500 一樣，LDAP 採用目錄組織分層方式。這些目錄可以存儲各種資訊，如名字，地址，和電話號碼，甚至可以像類似於網路資訊服務(NIS)來使用，確保任何人在任何機器上通過 LDAP 允許的網路都可以訪問他們的帳戶。

LDAP 通常用於集中管理用戶和組，用戶身份驗證或系統配置。它也能作為一個虛擬電話目錄，允許用戶方便的訪問其他用戶的資訊。此外，它可以引用用戶到其他 LDAP 伺服器中，從而提供一個特別的全球資訊的存儲庫。然而，它是最常見的是應用於單個組織機構，例如大學、政府部門和私人公司。

#### 6.2.1.1. LDAP 介紹

使用 client-server 體系結構，LDAP 創建一個通過網路可訪問的中心資訊目錄。當客戶端嘗試修改這個目錄裏面的資訊時，伺服器端會驗證用戶是否有修改的許可權，然後如果有需求會添加或更新入口。為了確保對話是安全的，Transport Layer Security (TLS)密碼協議被用來防止通過截獲傳輸來攻擊。

LDAP 伺服器支持多種資料庫系統，管理員可以根據不同需求，有更多的選擇。因為已經有定義好的編程介面，能夠和 LDAP 伺服器進行通信的應用程式有很多，並且在數量和品質上都在增加。

#### 6.2.1.2. LDAP 術語

下麵列出在本章中使用 LDAP-specific 術語：

**entry**

LDAP 目錄的單一組件。每一個 entry 通過單獨的分辨名來定義（DN）

## attribute

資訊是直接和 entry 關聯的。例如，假如一個組織表示為一個 LDAP 的 entry，和該組織關聯的 attributes 包括地址，傳真機號等等。類似的，個人表示為一個 entry，包括個人電話號碼或郵箱地址。

一個 attribute 可以是一個單一的值，或者是一組無序的值列表。雖然某些屬性是可選的，但其他是必需的。必要的 attributes 必須使用 objectClass 來定義，並且在 /etc/openldap/slapd.d/cn=config/cn=schema/ 目錄下的 schema 檔中被找到。

該 attribute 的聲明和它的值被稱為 Relative Distinguished Name (RDN)，不同於 DN，RDN 對於一個 entry 來說是獨一無二的。

## LDIF

LDAP 數據交換格式 (LDIF)，是 LDAP entry 的純文本表現方式，如下所示：

```
[id] dn: distinguished_name
attribute_type: attribute_value...
attribute_type: attribute_value...
...
```

id 選項是一個應用定義的數字，用來編輯 entry。每一個 entry 可以包含多個 attribute\_type 和 attribute\_value 組，只有它們在相應的檔中被定義了。在 entry 最後，包含一空白行。

### 6.2.1.3. OpenLDAP 特性

OpenLDAP 提供了很多重要的特性：

- 支持 LDAPv3——在 LDAP 版本 2 當中，該協議中許多修改設計是用來保證 LDAP 更加安全。其他改進，包括支持 Simple Authentication and Security Layer (SASL), Transport Layer Security (TLS) 和 Secure Sockets Layer (SSL) 等協議；
- LDAP 使用 IPC——使用 inter-process communication (IPC)，加強了安全性通過消除通過網路進行對話的需求。；
- IPv6 的支持——OpenLDAP 支持 IPv6 網路協議；
- LDIFv1 的支持——OpenLDAP 支持 LDIF 版本 1；
- 更新後的 C 介面——當前的 C 介面增強了程式員連接和使用 LDAP 目錄伺服器的方法；
- 加強了獨立的 LDAP 伺服器——包括了更新了的訪問控制系統，線程池，更好的工具等等。

#### 6.2.1.4. OpenLDAP 伺服器安裝

在 Kylin 系統上安裝 LDAP 伺服器步驟如下：

- 1) 安裝 OpenLDAP 組件，參見 6.2.2 安裝 OpenLDAP 組件；
- 2) 配置參見 6.2.3 配置 OpenLDAP 伺服器；
- 3) 啟動 slapd 服務，參見 6.2.5 運行 OpenLDAP 服務；
- 4) 使用 ldapadd 功能添加 entries 給 LDAP 目錄；
- 5) 使用 LDAPsearch 功能確保 slapd 服務已經正確獲得資訊。

#### 6.2.2. 安裝 OpenLDAP 組件

OpenLDAP 的函數庫和工具由以下包提供：



表 6-3 OpenLDAP 包列表

包	描述
openldap	該包包含運行 OpenLDAP 伺服器 and 客戶端應用所必須的函數庫。
openldap-clients	該包包含了可以訪問和修改 LDAP 伺服器的命令行程序。
openldap-servers	該包包含了運行 LDAP 伺服器的服務以及配置程式，包含了單獨的 LDAP Daemon, slapd。
openldap-devel	開發包。
migrationtools	通過 migrationtools 實現 OpenLDAP 用戶及用戶組的添加，導入系統帳戶。

此外，以下包通常和 LDAP 伺服器使用：

表 6-4 常用附加 LDAP 包列表

包	描述
nss-pam-ldapd	該包包含 nslcd，一個本地的 LDAP 名稱服務，允許用戶執行本地查詢。
mod_ldap	該包包含 mod_authnz_ldap 和 mod_LDAP 模組，其中 mod_authnz_ldap 模組是為 Apache HTTP Server 的 LDAP 驗證模組。該模組能針對 LDAP 目錄夠驗證用戶的證書，並且能夠強制訪問控制基於用戶名，完全的 DN，組關

	<p>係，一個任意 attribute，或者是一個完整的過濾字串。這個 mod_LDAP 模組包含在同一個包中，提供一個可配置的共用記憶體 cache，為了避免重複的 HTTP 請求，支持 SSL/TLS。</p>
--	---

使用 dnf 安裝以下包：

```
dnf install package...
```

例如，安裝 LDAP 伺服器

```
#dnf install openldap openldap-clients openldap-servers
```

注意您必須擁有超級用戶許可權，使用 root 用戶進行登錄後運行命令。如果想知道更多安裝新軟體包的資訊，請參考“4.2.4 安裝軟體包”。

#### 6.2.2.1. OpenLDAP 伺服器端程式

為了執行管理任務，這些 openldap-servers 包安裝了以下組件和 slapd 服務：

**表 6-5 OpenLDAP 服務端工具列表**

命令	描述
slapacl	允許您檢查訪問屬性的列表。
slapadd	允許您添加 entries 從 LDIF 檔到 LDAP 目錄。
slapauth	允許您檢查認證和許可權 ID 列表。
slapcat	允許您從 LDAP 目錄中以 LDIF 格式保存 entries。
slapdn	允許您檢查 DN 列表。

slapindex	允許您根據當前內容重新編排 slapd 目錄。創建 OpenLDAP 目錄索引，提高查詢效率。
slappasswd	允許您創建一個加密的用戶密碼，為 ldapmodify 組件，或者用在 slapd 配置檔中。
slapschema	允許您通過相應的模式檢查資料庫是否符合規範。
slaptest	允許您檢查 LDAP 伺服器配置。

#### 6.2.2.2. OpenLDAP 的客戶端程式

openldap-clients 安裝包包含以下組件，功能包括在 LDAP 目錄下添加，修改和刪除 entries：

**表 6-6 OpenLDAP 客戶端工具列表**

命令	描述
ldapadd	允許您給 LDAP 目錄添加 entries，可以通過一個檔或者是標準輸入，該命令是 ldapmodify -a 的一個鏈接。
ldapcompare	允許您拿一個給定的 attribute 和 LDAP 目錄 entry 進行比較。
ldapdelete	允許您刪除一個 LDAP 目錄 entry。
ldapexop	允許您使用 LDAP 擴展操作。
ldapmodify	允許您修改 LDAP 目錄 entry，通過檔或標準輸入。
ldapmodrdn	允許您修改修改 LDAP entry 的 RDN 值。
ldappasswd	允許您修改 LDAP 用戶密碼。

ldapsearch	允許您修改查找 LDAP entry。
ldapurl	允許您生成或分解 LDAP URLS。
ldapwhoami	允許您在 LDAP 伺服器上執行我是誰操作。

除了 ldapsearch 命令外，其他命令建議使用檔的方式進行修改，而不是直接使用命令進行修改。可以通過 man 命令查看檔格式。

### 6.2.2.3. LDAP 客戶端應用介紹

雖然有許多 LDAP 客戶端可以創建和修改伺服器端目錄，但是銀河麒麟高級伺服器操作系統沒有包含任何其中一種。常見的能夠以只讀模式訪問伺服器目錄的應用，包括 Mozilla, Thunderbird, Evolution, 或者 Ekiga。

### 6.2.3. 配置 OpenLDAP 伺服器

默認的，OpenLDAP 配置檔保存在/etc/openldap 目錄下。下麵的表包含了最重要的一些檔和目錄。

表 6-7 OpenLDAP 配置目錄和文件列表

路徑	描述
/etc/openldap/ldap.conf	使用 OpenLDAP 庫函數的客戶端應用的配置檔，包括 ldapadd, ldapsearch, Evolution 等等
/etc/openldap/slapd.d/	Slapd 的配置檔

OpenLDAP 不再使用/etc/openldap/slapd.conf 配置檔，它使用一個配置資料庫在 /etc/openldap/slapd.d/ 目錄。假如您之前的安裝已經有了

slapd.conf 檔，您可以使用以下命令進行轉換：

```
#slaptest -f /etc/openldap/slapd.conf -F
/etc/openldap/slap.d/
```

slapd 配置包括 LDIF entries，在一個分層的目錄組織結構中，可以進行相應的編輯。

### 6.2.3.1. 修改全局配置

LDAP 伺服器的全局配置檔保存在 /etc/openldap/slapd.d/cn=config.ldif 檔中。常用以下指令：

olcAllows

olcAllows 命令運行您指定哪些特性是可以使用的，使用以下格式：

```
olcAllows: feature
```

可用的特性，參照表 6-8 可用的 olcAllows 選項。默認的選項是 bind\_v2。

表 6-8 可用的 olcAllows 選項

選項	描述
Bind_v2	LDAP 可接受的 bind 請求
Bind_anon_cred	當 DN 是空的時候接受匿名的 bind
Bind_anon_dn	當 DN 是非空的時候接受匿名 bind
Update_anon	接受匿名升級操作
Proxy_authz_anon	接受匿名代理控制

例如：使用 olcAllows 指令

```
olcAllows: bind_v2 update_anon
```

**olcConnMaxPending**

**olcConnMaxPending** 命令允許您指定匿名會話最大請求等待數，命令如

下：

```
olcConnMaxPending: 100
```

**olcConnMaxPendingAuth**

**olcConnMaxPendingAuth** 命令您指定驗證過的會話最大請求等待數，命

令如下：

```
olcConnMaxPendingAuth : number
```

默認值是 1000。

例如：使用 **olcConnMaxPendingAuth** 命令

```
olcConnMaxPendingAuth : 1000
```

**olcDisallows**

**olcDisallows** 命令允許您指定哪些特性不可用。命令如下：

```
olcDisallows: feature...
```

可接受的特性列表參考表 6-9 可用的 **olcDisallows** 選項。沒有默認不可使用的特性。

表 6-9 可用 `olcDisallows` 選項

選項	描述
<code>bind_anon</code>	不允許接收匿名 <code>bind</code> 請求
<code>bind_simple</code>	不允許簡單的 <code>bind</code> 驗證機制
<code>tls_2_anon</code>	不允許增強匿名會話當收到 <code>STARTTLS</code> 命令
<code>tls_authc</code>	當已經驗證後不允許 <code>STARTTLS</code> 命令

例如：使用 `olcDisallows` 命令

```
olcDisallows: bind_anon
```

`olcIdleTimeout`

`olcIdleTimeout` 命令允許您指定在關閉一個 `idle` 連接的時候，可以等待的時間。命令如下：

```
olcIdleTimeout: number
```

該選項默認值是不使用（意思是設置為 0）

例如：使用 `olcIdleTimeout` 命令

```
olcIdleTimeout: 180
```

`olcLogFile`

`olcLogFile` 命令指定一個檔記錄日誌資訊，命令如下：

```
olcLogFile: file_name
```

日誌資訊默認使用錯誤標準輸入格式

例如：使用 `olcLogFile` 命令

```
olcLogFile: /var/log/slapd.log
```

`olcReferral`

`olcReferral` 選項允許您指定一個伺服器的 URL 來處理請求，命令如下：

```
olcReferral: URL
```

默認不使用

例如：使用 `olcReferral` 命令

```
olcReferral: ldap://root.openldap.org
```

`olcWriteTimeout`

`olcWriteTimeout` 選項允許您指定在關閉一個未完成的寫請求連接的時候，可以等待的時間。格式如下：

```
olcWriteTimeout
```

默認不開啟（意思是值為 0）

例如：使用 `olcWriteTimeout` 命令

```
olcWriteTimeout: 180
```

#### 6.2.3.2. 修改特定資料庫配置

默認的，OpenLDAP 伺服器使用 Berkeley DB(BDB)作為後端資料庫。該



資料庫配置存儲在 `/etc/openldap/slapd.d/cn=config/olcDatabase={1}bdb.ldif` 檔。以下為配置資料庫命令：

`olcReadOnly`

`olcReadOnly` 命令允許您使用只讀模式使用資料庫，使用如下：

```
olcReadOnly: boolean
```

接收參數：TRUE（只讀模式）或 FALSE（可修改模式），默認為 FALSE

例如：使用 `olcReadOnly` 命令

```
olcReadOnly: TRUE
```

`olcRootDN`

`olcRootDN` 命令可以為 LDAP 目錄指定超級用戶。使用如下：

```
olcRootDN: distinguished_name
```

接收 DN。默認值為 `cn=Manager,dn=my-domain,dc=com`。

例如：使用 `olcRootDN` 命令

```
olcRootDN: cn=root, dn=example, dn=com
```

`olcRootPW`

`olcRootPW` 命令允許您為用戶設置密碼，使用如下：

```
olcRootPW: password
```

可以接收沒有加密的文本字串，或者是哈希值，在 shell 終端，使用如下命令生成哈希值。

```
#slappaswd  
New password:  
Re-enter new password:  
{SSHA}WczWsyPEnMchFf1GRTweq2q7XJcvmSxD
```

例如：使用 `olcRootPW` 命令

```
olcRootPW:  
{SSHA}WczWsyPEnMchFf1GRTweq2q7XJcvmSxD
```

`olcSuffix`

`olcSuffix` 命令允許您指定提供資訊的域，使用如下：

```
olcSuffix: domain_name
```

接收 FQDN，默認是 `dc=my-domain, dc=com`。

例如：使用 `olcSuffix` 命令

```
olcSuffix: dc=example, dc=com
```

### 6.2.3.3. 架構擴展

自從 OpenLDAP2.3 以來，`/etc/openldap/slapd.d/`目錄包含了之前存放在`/etc/openldap/schema/`目錄下的一些 LDAP 定義。OpenLDAP 使用這些可以擴展方案，支持額外的 `attribute` 類型和對象類使用缺省架構檔。關於這個方面更多的資訊，可以參考 <http://www.openldap.org/doc/admin/schema.html>。

#### 6.2.3.4. 建立安全連接

OpenLDAP 客戶端和服務器端使用 Transport Layer Security (TLS) 框架來保證安全。TLS 是提供網路通信安全的加密協議。上面提到的，在銀河麒麟高級伺服器操作系統中 OpenLDAP 使用 Mozilla NSS 作為 TLS 的安裝實現。

使用 TLS 建立安全連接，怎麼通過 Mozilla NSS 使用 TLS/SSL，鏈接 <http://www.openldap.org/faq/data/cache/1514.html>。因此，需要在客戶端和服務器端進行相關配置。至少，伺服器端需要配置 CA 證書和它本身的伺服器證書和私鑰。客戶端需要配置包含可信任的 CA 證書檔。

典型的，伺服器端需要指定一個 CA 證書，客戶端想要連接到一個安全的伺服器端，因此需要在其配置檔裏面指定可信任的 CA。

#### 伺服器配置：

該章節列出了在 OpenLDAP 伺服器中使用 TLS，需要對 slapd 命令進行全局配置，在 `/etc/openldap/slapd.d/cn=config.ldif` 配置檔中配置。

老版本的配置使用單獨的配置檔，通常是 `/usr/local/etc/openldap/slapd.conf`，新版的使用 slapd 後端資料庫保存配置，保存目錄 `/usr/local/etc/openldap/slapd.d/`。

以下命令對於確立 SSL 來說也是有效的，除了 TLS 命令外，您需要在伺服器端給 SSL 打開一個端口，一般來說是 636 端口。編輯 `/etc/sysconfig/slapd` 檔，添加 `ldaps:///` 字串，指定 `SLAPD_URLS` 命令 URLs。

#### **olcTLSCACertificateFile**

`olcTLSCACertificateFile` 命令指定了 Privacy-Enhanced Mail (PEM) 編碼的檔，包含可信的 CA 證書。使用如下

```
olcTLSCACertificateFile: path
```

`path` 為包含 CA 證書的檔，或者如果使用 Mozilla NSS 的話，可以是證書名稱。

### **olcTLSCACertificatePath**

`olcTLSCACertificatePath` 命令指定在不同檔中包含單獨 CA 證書存放的目錄。該目錄必須由 OpenSSL `c_rehash` 進行管理，生成指向實際證書檔的鏈接，一般而言，常使用 `olcTLSCACertificateFile` 作為替代。

假如 Mozilla NSS 被使用，`olcTLSCACertificatePath` 接受 Mozilla NSS 資料庫路徑（就像下例所說）。在這種情況下，`c_rehash` 是不需要的。

使用如下

```
olcTLSCACertificatePath: path
```

例如：使用 `olcTLSCACertificatePath` Mozilla NSS。

通過 Mozilla NSS，`olcTLSCACertificatePath` 指定目錄路徑，該目錄包含 NSS 證書和數據庫檔：

```
olcTLSCACertificatePath: sql:/home/nssdb/sharednssdb
```

`certutil` 命令用來給 NSS 資料庫檔添加 CA 證書：

```
certutil -d sql:/home/nssdb/sharednssdb -A -n  
"CA_certificate" -t CT,, -a -i certificate.pem
```

以上的命令添加了一個 CA 證書，以 PEM-formatted 格式保存，名字為

certificate.pem。-d 選項指定資料庫目錄，該目錄包含 NSS 證書和數據庫檔，-n 選項設置證書名稱，-t CT,,意思是證書是可信任的，在 TLS 客戶端和服務器端使用。-A 添加已存在的證書至證書資料庫中，-a 允許使用 ASCII 格式作為輸入輸出，-i 將 certificate.pem 輸入傳遞給命令。

### **olcTLSCertificateFile**

olcTLSCertificateFile 命令指定包含 slapd 伺服器證書的檔。

```
olcTLSCertificateFile: path
```

path 為包含 slapd 伺服器證書的檔，如果使用 Mozilla NSS，改為證書名稱。

例如：通過 Mozilla NSS 使用 olcTLSCertificateFile

當使用 Mozilla NSS 和證書關鍵資料庫檔和 olcTLSCACertificatePath 命令，olcTLSCACertificatePath 用來指定證書的名稱。首先，列出 NSS 資料庫檔中可用的證書。

```
certutil -d sql:/home/nssdb/sharednssdb -L
```

選擇一個證書，將它的名稱傳遞給 olcTLSCertificateFile

```
olcTLSCertificateFile slapd_cert
```

### **olcTLSCertificateKeyFile**

olcTLSCertificateKeyFile 命令指定檔，該檔包含私鑰，私鑰和存放在 olcTLSCertificateFile 的證書是匹配的。當前不支持加密的私鑰，因此該檔必

須被有效的保護。

```
olcTLSCertificateKeyFile: path
```

當使用 PEM 證書時，`path` 替換為私鑰檔路徑。當使用 Mozilla NSS 時，`path` 替換為一個檔的名稱，該檔包含 `olcTLSCertificateFile` 命令指定證書的密碼（參考下例使用 `olcTLSCertificateKeyFile` 和 Mozilla NSS）。

例如：使用 `olcTLSCertificateKeyFile` 和 Mozilla NSS

當使用 Mozilla NSS 時，該命令指定一個檔的名稱，該檔包含 `olcTLSCertificateFile` 指定的證書的 `key` 的密碼。

```
olcTLSCertificateKeyFile: slapd_cert_key
```

`modutil` 命令能夠用來轉變密碼保護或改變 NSS 資料庫檔密碼。

```
modutil -dbdir sql:/home/nssdb/sharednssdb -change pw
```

## 客戶端配置

配置檔 `/etc/openldap/ldap.conf` 在系統中是全局的，也有單獨的用戶在 `~/.ldaprc` 配置中進行覆蓋配置。

相同的指令可以創建一個 SSL 連接。在 OpenLDAP 命令比如 `ldapsearch`，`ldaps://` 字串必須替代 `ldap://`。這些命令使用伺服器端默認的 SSL 端口 636。

## TLS\_CACERT

`TLS_CACERT` 命令指定一個檔，包含客戶端識別的所有的證書。該功能等同於伺服器的 `olcTLSCACertificateFile` 命令。`TLS_CACERT` 應該在檔

/etc/openldap/ldap.conf 中 TLS\_CACERTDIR 選項前被指定。

TLS\_CACERT path

path: CA 證書檔路徑

### **TLS\_CACERTDIR**

TLS\_CACERTDIR 命令指定在不同檔中包含單獨 CA 證書存放的目錄。跟 olcTLSCACertificatePath 命令類似。該目錄必須由 OpenSSL c\_rehash 進行管理，接受 Mozilla NSS 資料庫檔路徑，在這種情況下，c\_rehash 是不需要的。

TLS\_CACERTDIR directory

directory: 包含 CA 證書的目錄路徑。使用 Mozilla NSS 時，為證書或關鍵資料庫檔路徑。

### **TLS\_CERT**

TLS\_CERT 指定檔，包含客戶端證書。該命令只有在用戶 ~/.ldaprc 檔中被指定。在 Mozilla NSS 下，該命令指定的是證書的名字，由 TLS\_CACERTDIR 命令指定。

TLS\_CERT path

path: 客戶端證書檔路徑，或者是 NSS 資料庫證書的名稱

### **TLS\_KEY**

TLS\_KEY 指定包含私鑰的檔，私鑰是匹配存放在 TLS\_CERT 指定的證書。該功能和服務器 olcTLSCertificateFile 類似，加密的檔是不支持的，所以該檔需要被小心保護，該選項只能在用戶的 ~/.ldaprc 檔指定。

當使用 Mozilla NSS 時，`TLS_KEY` 指定一個檔，包含私鑰的密碼，用來保護 `TLS_CERT` 指定的證書。功能和 `olcTLSCertificateKeyFile` 類似，您可以使用 `modutil` 命令管理密碼。

`TLS_KEY` 使用如下

```
TLS_KEY path
```

`path`: 客戶端證書檔路徑，或者是 NSS 資料庫中的密碼檔案名稱。

#### 6.2.3.5. 設置備份

備份是一個拷貝更新的進程，從一個 LDAP 伺服器（`provider`）至一個或多個其他的伺服器或客戶端（`consumer`）。一個 `provider` 備份目錄更新至 `consumers`，接收到的更新能夠被 `consumer` 傳播至其他的伺服器端，因此一個 `consumers` 可以被當做一個 `provider`。一個 `consumers` 可以不是一個 LDAP 伺服器端，它可以僅僅是一個個客戶端。在 OpenLDAP，有多種備份模式，常見的有 `mirror` 和 `sync`。

為了使用選擇好的備份模式，需要在 `provider` 和 `consumers` 的 `/etc/openldap/slapd.d/` 選擇以下其中一種模式：

##### **olcMirrorMode**

`olcMirrorMode` 使用 `mirror` 備份模式

```
olcMirrorMode on
```

##### **olcSyncrepl**

`olcSyncrepl` 使用 `sync` 備份模式



```
olcSyncrepl on
```

#### 6.2.3.6. 加載模組和後端

您可以使用動態加載模組用來增強 `slapd` 服務。在配置 `slapd` 的時候，可以使用 `--enable-modules` 選項來配置那些可以支持的模組。模組保存在 `.la` 結尾的檔中。

```
module_name.la
```

後端存儲和數據檢索應對 LDAP 請求。後端靜態的編譯至 `slapd` 或者當模組是被支持的，它們能夠被動態的加載。對於後者，以下為命名約定

```
back_backend_name.la
```

為了加載模組和後端，在 `/etc/openldap/slapd.d/` 選擇：

#### **olcModuleLoad**

`olcModuleLoad` 指定動態加載的模組

```
olcModuleLoad: module
```

`module`：一個檔包含模組或後端，將要被加載。

#### 6.2.4. 使用 LDAP 應用的 SELinux 策略

SELinux 是一個在 linux 內核中實現的一個強制訪問控制機制。默認的，SELinux 會組織應用訪問 OpenLDAP 伺服器。為了允許應用通過 LDAP 驗證，SELinux 的 `allow_yppbind` 必須被設為可用的。某些應用也需要

authlogin\_nsswitch\_use\_ldap 是被允許的。以下是啟動命令：

```
#setsebool -P allow_ybind=1
#setsebool -P authlogin_nsswitch_use_ldap=1
```

-P 選項是這次設置在系統重啟一直保持有效。

### 6.2.5. 運行 OpenLDAP 服務

該章節描述了怎樣啟動、停止、重啟和檢查當前 LDAP 服務的狀態。

#### 6.2.5.1. 啟動服務

使用 root 許可權，開啟 slapd 服務，命令如下：

```
#systemctl start slapd.service
```

使用 root 許可權，設置開機啟動 slapd 服務，命令如下：

```
#systemctl enable slapd.service
Created symlink /etc/systemd/system/openldap.service →
/usr/lib/systemd/system/slapd.service.
Created symlink
/etc/systemd/system/multi-user.target.wants/slapd.service →
/usr/lib/systemd/system/slapd.service.
```

#### 6.2.5.2. 停止服務

停止服務，命令如下：

```
#systemctl stop slapd.service
```

設置開機不啟動服務，命令如下：

```
#systemctl disable slapd.service
Removed
/etc/systemd/system/multi-user.target.wants/slapd.service.
Removed /etc/systemd/system/openldap.service.
```

#### 6.2.5.3. 重啟服務

重啟服務，命令如下：

```
#systemctl restart slapd.service
```

該命令會先停止服務，馬上再重啟服務。使用該命令重新加載配置。

#### 6.2.5.4. 檢查運行狀態

檢查 slapd 服務運行狀態，命令如下：

```
#systemctl is-active slapd.service
active
```

### 6.2.6. 配置系統使用 OpenLDAP 作為驗證

為了配置系統使用 OpenLDAP 作為驗證，確保所有的安裝包在 LDAP 伺服器 and 客戶端機器上已經安裝。怎麼安裝請參考章節 6.2.2 安裝 OpenLDAP 組件和 6.2.3 配置 OpenLDAP 伺服器。在客戶端上，輸入命令如下：

```
#dnf install openldap openldap-clients nss-pam-ldapd
```

### 6.2.6.1. 遷移舊的驗證資訊至 LDAP 格式

遷移工具包提供了許多 shell 和 perl 腳本，可以幫助遷移舊的驗證資訊至 LDAP 格式。安裝這些包，命令如下：

```
#dnf install migrationtools
```

安裝完後，腳本存放目錄：`/usr/share/migrationtools/`。編輯 `/usr/share/migrationtools/migrate_common.ph` 檔，修改以下行內容用來映射當前域。

```
#Default DNS domain
$DEFAULT_MAIL_DOMAIN = "example.com";
#Default base
$DEFAULT_BASE = "dc=example,dc=com";
```

或者，使用命令行指定環境變數。例如，運行 `migrate_all_online.sh` 腳本，使用默認的基準，設置 `dc=example,dc=com`

```
# DEFAULT_BASE="dc=example,dc=com" \
/usr/share/migrationtools/migrate_all_online.sh
```

決定使用哪個腳本來運行遷移用戶數據庫，參考表 6-10 常用的 LDAP 遷移腳本。

表 6-10 常用的 LDAP 遷移腳本

已存在的服務	LDAP 是否運行	使用的腳本
/etc flat files	yes	migrate_all_online.sh
/etc flat files	no	migrate_all_offline.sh

NetInfo	yes	migrate_all_netinfo_online.sh
NetInfo	no	migrate_all_netinfo_offline.sh
NIS (YP)	yes	migrate_all_nis_online.sh
NIS (YP)	no	migrate_all_nis_offline.sh

怎麼使用這些腳本，參考 README 和 migration-tools.txt 檔，存放在 /usr/share/doc/migrationtools-47/目錄下。

### 6.3. 檔和列印伺服器

這個章節主要介紹 Samba 的安裝和配置，一個 Server Message Block (SMB)和 common Internet file system (CIFS)協議的開源實現，vsftpd，銀河麒麟高級伺服器操作系統的 FTP 伺服器。此外，還介紹了怎麼使用列印伺服器工具去配置印表機。

#### 6.3.1. Samba

Samba 是標準 linux 開源 windows 套件專案。它實現了 SMB 和 CIFS 協議。它允許不同的系統包括 Microsoft Windows®,Linux,UNIX 等，訪問基於 windows 的檔和印表機共用。Samba 的 SMB 使用類似 windows 伺服器與 windows 客戶端一樣。

samba 安裝

```
#dnf install samba
```

##### 6.3.1.1. Samba 介紹

Samba 是一個很重要的組件，作為無縫集成 Linux 伺服器和桌面至 Active Directory(AD)環境。它可以作為一個域控制器，或者是常規的域成員。

Samba 能夠做什麼：

- 服務目錄樹和 Linux, Unix 和 windows 客戶端的印表機;
- 協助網路流覽;
- Windows 域進行身份驗證登錄;
- 提供 Windows Internet Name Service (WINS)名稱服務解決方案;
- Windows NT®-style Primary Domain Controller (PDC);
- Backup Domain Controller (BDC) for a Samba-based PDC;
- 功能變數名稱伺服器成員;
- 加入 Windows NT/2000/2003/2008 PDC/Windows Server 2012

Samba 不能做什麼:

- 作為 BDC 替代 windows PDC;
- 作為功能變數名稱伺服器控制器。

#### 6.3.1.2. Samba 以及相關服務

Samba 是由三個守護進程組成 (smbd, nmbd 和 winbind)。三個服務 (smb, nmb 和 winbind) 控制著守護進程的啟動, 停止和其他相關服務功能。這些服務作為不同的 init 腳本。以下詳細介紹每一個守護進程。

##### smbd

伺服器端 smbd 守護進程給 windows 客戶端提供檔共用和列印服務。另外, 它負責用戶身份驗證, 資源鎖定, 數據共用通過 SMB 協議。默認的, 伺服器監控 SMB 傳輸端口為 TCP 端口 139 和 445。smbd 守護進程是被 smb 服務控制。

##### nmbd

nmbd 守護進程對於 NetBIOS 名稱服務的請求 (SMB/CIFS in

Windows-based systems ) 進行理解和回應。這些系統包括 Windows 95/98/ME, Windows NT, Windows 2000, Windows XP, 和 LanManager 客戶端。默認伺服器監控 NMB 傳輸端口是 UDP 端口 137。

#### winbindd

winbind 服務解決了用戶和組資訊從運行在 Windows NT, 2000, 2003, Windows Server 2008, or Windows Server 2012 伺服器上的接收問題。這使得 windows 用戶和組資訊能夠被 UNIX 平臺識別。這是被 Microsoft RPC calls, Pluggable Authentication Modules (PAM) 和 the Name Service Switch (NSS) 所實現。這允許 Windows NT 域和 Active Directory 用戶被當做 UNIX 用戶進行操作。雖然和 Samba 進行了捆綁, 但是 winbind 服務是和 smb 分開進行控制的。

#### 連接 Samba 共用

您可以使用 **Nautilus** 或命令行連接可用的 Samba 共用。

#### 掛載共用

有時候, 需要對 Samba 共用進行掛載操作, 需要使用 root 用戶登錄, 命令如下:

```
mount -t cifs //servername/sharename /mnt/point/ -o  
username=username,password=password
```

該命令將 sharename 掛載至本地 /mnt/point/ 目錄。

### 6.3.1.3. 配置 Samba 伺服器

默認的配置檔/etc/samba/smb.conf 允許用戶訪問它們的 home 目錄作為 samba 共用。它可以共用所有配置好的印表機作為 samba 共用印表機。您可以在系統中附加一個印表機並且通過 windows 機器列印東西。

#### 圖形配置

配置 Samba 使用圖形介面，可以參考 <http://www.samba.org/samba/GUI/>。

#### 命令行配置

Samba 使用/etc/samba/smb.conf 作為配置檔。修改這個配置檔後，需要重啟 Samba 才能夠生效

```
#systemctl restart smb.service
```

指定 windows 工作組和 Samba 伺服器簡短的描述，編輯 /etc/samba/smb.conf 檔。

```
workgroup = WORKGROUPNAME  
server string = BRIEF COMMENT ABOUT SERVER
```

在 linux 系統上創建 Samba 共用目錄，在/etc/samba/smb.conf 檔中添加以下內容：

例如：Samba 伺服器的配置

```
[sharename]  
comment = Insert a comment here  
path = /home/share/
```



```
valid users = tfox carole  
writable = yes  
create mask = 0765
```

該例子允許用戶 **tfox** 和 **carole** 通過 **Samba** 客戶端對 **Samba** 伺服器 **/home/share/**目錄進行讀寫操作

加密密碼

加密密碼是可以被使用的，因為加密的密碼會更安全。創建一個使用加密密碼的用戶，命令如下：

```
smbpasswd -a username
```

#### 6.3.1.4. 啟動和關閉 Samba

啟動命令如下：

```
#systemctl start smb.service
```

關閉命令如下：

```
#systemctl stop smb.service
```

重啟

```
#systemctl restart smb.service
```

**condrestart**（特定條件下重啟）在當前正在運行的情況下才會啟動 **smb**。這個選項對腳本是非常有用的，當守護進程沒有運行的時候，將不會被啟動。

```
#systemctl try-restart smb.service
```

重新載入/etc/samba/smb.conf 配置檔，不需要進行服務的重啟，命令如下：

```
#systemctl reload smb.service
```

開機自啟動，命令如下：

```
#systemctl enable smb.service
```

#### 6.3.1.5. Samba 安全模式

Samba 有兩種安全模式，share-level 和 user-level。share-level 已經被棄用了，User-level 可以有三種不同的實現方式，這些方式被稱作安全模式。

##### User-Level 安全

User-level security 是 Samba 默認推薦的配置。即使 security = user 沒有在/etc/samba/smb.conf 配置檔中列出，它照樣會被使用。當伺服器接收了客戶端的用戶名和密碼，客戶端在掛載共用的時候就能夠不需要指定密碼。

Samba 也能夠接收到基於用戶名和密碼的請求。客戶端通過使用一個單獨的 UID 維持已驗證的狀態。

/etc/samba/smb.conf 檔中, security = user 設置 user-level security。

```
[GLOBAL]
...
security = user
...
```

## Samba Guest 共用

上面已經提到過,share-level security 模式已經被放棄。怎麼配置 Samba guest 共用, 不使用 security = share 選項。參考以下步驟:

### 實例: 配置 Samba Guest 共用

- 1) 創建用戶映射檔, 例如/etc/samba/smbusers, 添加以下行:

```
nobody = guest
```

- 2) 修改/etc/samba/smb.conf, 不要使用 valid users

```
[GLOBAL]
...
security = user
map to guest = Bad User
username map = /etc/samba/smbusers
...
```

- 3) 修改/etc/samba/smb.conf, 不要使用 valid users

```
[SHARE]
...
guest ok = yes
...
```

下麵將會介紹其他 user-level security 實現方式。

## Domain Security Mode (User-Level Security)

在這種方式下, Samba 伺服器有一個機器帳號 (domain security 信任

的帳號)，所有驗證的請求都需要通過域控制器。Samba 伺服器要作為域成員，需要配置/etc/samba/smb.conf。

```
[GLOBAL]
...
security = domain
workgroup = MARKETING
...
```

### Active Directory Security Mode (User-Level Security)

假如您有一個 Active Directory 環境，加入域作為一個本地成員是可能的。即使安全策略限制使用 NT-compatible 驗證協議，Samba 伺服器可以通過使用 Kerberos 加入 ADS。Samba 在 Active Directory member 模式下可以接受 Kerberos tickets。

修改/etc/samba/smb.conf

```
[GLOBAL]
...
security = ADS
realm = EXAMPLE.COM
password server = kerberos.example.com
...
```

### Share-Level 安全

在該模式下，伺服器從客戶端那僅接收一個沒有明確用戶名的密碼。伺服器期望一個密碼可以給所有的共用，不依賴用戶名。許多報告指出，Microsoft Windows 客戶端相容 share-level security 伺服器。該模式已經被 Samba 拋棄。配置項 security = share 必須被升級成 user-level security。如果想使

用，請參考 User-Level 安全中的例子：配置 Samba Guest 共用。

#### 6.3.1.6. Samba 網路流覽

網路流覽使得 Windows 和 Samba 伺服器出現在 Windows Network Neighborhood 中，在 Network Neighborhood 裏面，伺服器有自己的圖示，打開圖示，伺服器可用的共用和印表機可以被看到。

網路流覽需要 NetBIOS 通過 TCP/IP。NetBIOS-based 網路使用 UDP 發送消息完成流覽列表管理。沒有 NetBIOS 和 WINS 作為 TCP/IP 主機名稱解決方案基本的方法，其他的例如靜態檔（/etc/hosts）或 DNS 是必須要使用的。

一個域的主流覽伺服器從所有子網中的本地主流覽伺服器收集核對所有的流覽列表，因此在組和子網間可以相互流覽。

##### 域流覽

默認的，一個 Windows 伺服器 PDC 作為一個域，也是該域的主流覽伺服器。Samba 伺服器不應該被作為一個主流覽伺服器。

在許多子網中，不包含 Windows 伺服器 PDC，因此，Samba 伺服器可以作為一個本地流覽伺服器。配置/etc/samba/smb.conf 檔作為一個本地主流覽伺服器（或不作為），配置過程和組配置類似（參見 6.3.1.3 配置 Samba 伺服器。）

##### WINS（Windows Internet Name Server）

Samba 和 Windows NT 伺服器可以被作為一個 WINS 伺服器。當 WINS 伺服器和 NetBIOS 一起使用時，UDP 傳播能夠被轉發在網路中允許使用名稱轉換。沒有 WINS 伺服器時，UDP 傳播只能在當前子網傳播不能夠被轉發至其他

子網，組和域。假如需要 WINS 備份功能，不要使用 Samba 作為您的 WINS 伺服器，因為 Samba 不支持 WINS 備份。

在一個 NT/2000/2003/2008 伺服器和 Samba 混合環境中，推薦使用 Microsoft WINS。在一個只有 Samba 環境中，推薦使用 Samba 作為 WINS。

下麵是一個使用 Samba 作為 WINS 伺服器的例子。

例如：配置 WINS 伺服器

```
[global]
wins support = yes
```

#### 6.3.1.7. Samba Distribution Programs

net 命令：

```
net <protocol> <function> <misc_options>
<target_options>
```

net 命令的使用和在 Windows 和 MS-DOS 系統中類似。第一個參數指定命令使用的協議。protocol 選項可以是 ads, rap 或 rpc。Active Directory 使用 ads, Win9x/NT3 使用 rap, Windows NT4/2000/2003/2008 使用 rpc。假如 protocol 選項沒指定，net 會自動嘗試確定它。

下麵例子顯示了主機名為 wakko 的可用的共用：

```
#net -l share -S wakko
Password:
```

下麵例子顯示了 wakko 主機可用的 Samba 用戶列表：

```
#net -l user -S wakko  
root password:
```

nmblookup 命令：

```
nmblookup <options> <netbios_name>
```

nmblookup 可以解決 NetBIOS 名稱轉換為 IP 地址。該專案在子網中會廣播查詢直到有目標主機回應。

下麵的例子是顯示 NetBIOS 名稱 trek 的 IP 地址：

```
#nmblookup trek  
querying trek on 10.1.59.255  
10.1.56.45 trek<00>
```

pdbedit 命令：

```
pdbedit <options>
```

pdbedit 專案管理 SAM 資料庫存儲的帳戶。所有的後端包括支持 smbpasswd, LDAP 和 tdb 資料庫。

下麵的例子是添加，刪除和列出用戶：

```
#pdbedit -a kristin  
new password:  
retype new password:  
#pdbedit -x joe  
#pdbedit -L  
andriusb:505: lisa:504: kristin:506:
```

rpcclient 命令：

```
rpcclient <server> <options>
```

rpcclient 專案問題管理命令使用 Microsoft RPCs，這個是給知道 Microsoft RPCs 複雜性用戶使用的。

smbcacls 命令：

```
smbcacls <\\server/share> <filename> <options>
```

smbcacls 專案修改 Windows ACLs 檔和 Samba 共用的目錄或者是 Windows 伺服器。

smbclient 命令：

```
smbclient <\\server/share> <password> <options>
```

smbclient 是 UNIX 系統通用的客戶端，功能和 ftp 類似。

smbcontrol 命令：

```
smbcontrol -i <options>  
smbcontrol <options> <destination> <messagetype>  
<parameters>
```

smbcontrol 專案發送控制消息來運行 smbd, nmbd 或 winbindd 守護進程。smbcontrol -i 互動式運行，直到出現空行或者‘q’被輸入。

smbpasswd 命令：

```
smbpasswd <options> <username> <password>
```



`smbpasswd` 專案管理加密密碼。該專案能夠被超級用戶修改所有的用戶密碼還有普通用戶修改自己的 Samba 密碼。

`smbspool` 命令：

```
smbspool <job> <user> <title> <copies> <options>
<filename>
```

`smbspool` 專案是 Samba 一個 CUPS 相容的列印介面。雖然被設置為 CUPS 印表機，`smbspool` 可以和非 CUPS 印表機一起使用。

`smbstatus` 命令：

```
smbstatus <options>
```

`smbstatus` 專案顯示當前 Samba 連接狀態

`smbtar` 命令：

```
smbtar <options>
```

`smbtar` 程式執行基於 Windows 共用檔和目錄的備份和恢復。雖然和 `tar` 命令相似，兩者不相容。

`testparm` 命令：

```
testparm <options> <filename> <hostname IP_address>
```

`testparm` 程式檢查 `/etc/samba/smb.conf` 檔的語法，假如您的 `smb.conf` 存儲在默認位置（`/etc/samba/smb.conf`），則不需要指定。指定主機名和 IP 地址給 `testparm` 程式檢查 `hosts.allow` 和 `host.deny` 檔是否配置正確。

testparm 程式可以顯示 smb.conf 檔和服務器規則在測試後。在調試的時候可以給豐富經驗的管理員提供有用的資訊。例如：

```
#testparm
Load smb config files from /etc/samba/smb.conf
Processing section "[homes]"
Processing section "[printers]"
Processing section "[tmp]"
Processing section "[html]"
Loaded services file OK.
Server role: ROLE_STANDALONE
Press enter to see a dump of your service definitions
<enter>
#Global parameters
[global]
workgroup = MYGROUP
server string = Samba Server
security = SHARE
log file = /var/log/samba/%m.log
max log size = 50
socket options = TCP_NODELAY SO_RCVBUF=8192
SO_SNDBUF=8192
dns proxy = no
[homes]
comment = Home Directories
read only = no
browseable = no
[printers]
comment = All Printers
path = /var/spool/samba
printable = yes
browseable = no
[tmp]
```

```
comment = Wakko tmp
path = /tmp
guest only = yes
[html]
comment = Wakko www
path = /var/www/html
force user = andriusb
force group = users
read only = no
guest only = yes
```

wbinfo 命令：

```
wbinfo <options>
```

wbinfo 程式顯示 winbindd 守護進程資訊。winbindd 進程必須是運行的。

### 6.3.1.8. 其他資源

安裝文檔

`/usr/share/doc/samba-<version-number>/`

可以查看 man 手冊

- smb.conf(5)
- samba(7)
- smbd(8)
- nmbd(8)
- winbindd(8)

有用的網站

- <http://www.samba.org/>
- [https://wiki.samba.org/index.php/User\\_Documentation](https://wiki.samba.org/index.php/User_Documentation)
- <http://samba.org/samba/archives.html>

### 6.3.2. FTP

該章節將會介紹 FTP 協議以及 vsftpd (Linux 系統 FTP 伺服器)

FTP 使用客戶端-服務端架構模式來傳輸檔，使用 TCP 網路協議。因為 FTP 是一個較早的協議，他不支持加密用戶和密碼進行驗證。基於這個原因，FTP 是被認為不安全的傳輸協議，除非特殊情況下不建議使用。因為 FTP 在網路上是非常流行的，它經常被用來分享檔。因此，作為一個管理員，需要知道它的獨特特性。

這個章節描述了怎麼配置 vsftpd 來建立安全連接通過 TLS 和怎麼通過 SELinux 來加強 FTP 的安全性。FTP 的一個更好的取代品是 sftp，是由 OpenSSH 組件工具提供。要想獲取更多 OpenSSH 配置以及 SSH 協議，請參考 5.2OpenSSH。

不同於其他協議，FTP 需要多個端口才能正常工作。當 FTP 客戶端建立一個到 FTP 伺服器端的連接，它會打開端口 21 在伺服器端——控制端口。這個端口被用來發送命令至伺服器。所有從伺服器端到客戶端的數據請求是由專門的數據端口傳輸。數據連接端口，以及數據連接的初始化依賴於客戶端請求數據方式：active 或 passive 模式。

#### **active mode**

Active mode 是 FTP 協議傳輸數據協議使用到的最原始的模式。當一個 active-mode 數據傳輸被 FTP 客戶端初始化，伺服器端會打開一個 20 端口給 IP 地址，和一個隨機無特權的端口（大於 1024）。這樣的安排意味著客戶端機器必須被允許接受任何超過 1024 的端口連接。隨著不安全網路的增長，使用防

火牆來保護客戶端機器現在是普遍的。因為這些客戶端防火牆常常否定從主動模式傳入的連接，所以 **passive mode** 被設計出來。

### **passive mode**

像 **active mode** 一樣，**Passive mode** 是被 **FTP** 客戶端初始化的。當從伺服器請求數據，**FTP** 客戶端表明它想在 **Passive mode** 下訪問數據，伺服器在伺服器上會提供一個 **IP** 地址和一個隨機、無特權的端口(大於 **1024**)。然後客戶端連接到該端口在伺服器上下載請求的資訊。

雖然 **Passive mode** 確實解決為客戶端防火牆干擾數據連接問題，它可以使管理伺服器端防火牆。您可以在一個伺服器上通過限制的範圍無特權的端口在 **FTP** 伺服器上減少開放端口的數量。這也簡化了伺服器配置防火牆規則的過程。

#### 6.3.2.1. vsftpd 伺服器

**vsftpd** 設計為快、穩定和安全。**vsftpd** 是銀河麒麟高級伺服器操作系統中的 **FTP** 伺服器。是因為其能夠處理大量的連接數和安全性。

**vsftpd** 所使用的安全模型有三個主要方面：

- 強大的特權和非特權分離過程——獨立的進程處理不同的任務，每一個進程運行的任務只需要最小許可權；
- 需要提升許可權的任務可以使用最小許可權進行處理——利用 **libcap** 庫相容性的特性，需要 **root** 特權許可權的任務可以用很小特權的進程執行；
- 大多數進程運行在 **chroot** 下——只要可能，進程可以改變程式執行時參考的根目錄位置為當前共用的目錄。這個目錄被認為是 **chroot** 目錄。

例如，假如 **/va/ftp/** 目錄是共用目錄，**vsftpd** 指定 **/va/ftp/** 為新的 **root**

目錄，作為/。這會減少駭客的攻擊。

使用這些安全措施會對 **vsftpd** 如何處理請求產生影響，如下：

- 父進程運行時僅需最少的特權——父進程能夠動態的計算最小特權等級將風險最小化。子進程處理直接與客戶端交互和盡可能使用無權限運行。所有的需要特權的操作被一個小的父進程處理——就像 **Apache HTTP Server** 一樣，**vsftpd** 分發無特權的子進程來處理傳入的連接。這允許特權，父進程盡可能小處理任務。
- 所有從非特權子進程來的請求將會被父進程懷疑——和子進程進行對話是通過 **socket**，任何來自子進程的資訊將會被檢查；
- 大部分和 **FTP** 客戶端互動式操作是在 **chroot** 環境下——因為子進程是非特權的，它們只有在共用目錄下有登入許可權，只允許攻擊者能夠訪問共用目錄。

啟動和停止 **vsftpd**

以 **root** 用戶登錄

啟動命令：

```
#systemctl start vsftpd.service
```

停止命令：

```
#systemctl stop vsftpd.service
```

重啟命令：

```
#systemctl restart vsftpd.service
```

有條件重啟，當它已經在運行的時候，才能夠執行：

```
#systemctl try-restart vsftpd.service
```

設置開機自啟動：

```
#systemctl enable vsftpd.service  
Created symlink  
/etc/systemd/system/multi-user.target.wants/vsftpd.service →  
/usr/lib/systemd/system/vsftpd.service.
```

開啟 vsftpd 多路拷貝

有時候，一臺電腦會被用來作為多路 FTP 域。這種技術叫做多歸屬。vsftpd 的多歸屬的用法是運行多個進程的拷貝，每一個進程擁有自己獨立的配置檔。

首先，給所有的網路設備分配好 IP。

再次，FTP 的域的 DNS 伺服器必須對應正確的機器。

當 vsftpd 回應不同 IP 的請求時，進程的多路拷貝必須是運行的。為了分發 vsftpd 進程的多路實例，一個特殊的服務 systemd service unit (vsftpd@.service) 是被 vsftpd 包提供的。

為了使用該服務，一個單獨的 vsftpd 配置檔為各個 FTP 伺服器實例是需要的，保存在 /etc/vsftpd/ 目錄。這些配置檔必須擁有獨立的名稱（例如 /etc/vsftpd/vsftpd-site-2.conf），並且擁有 root 的讀寫許可權。

每一個配置檔，要有如下參數作為監聽 IPv 網路：

```
listen_address=N.N.N.N
```

N.N.N.N 為 FTP 站點的 IP 地址。假如使用的是 IPv6，則使用 `listen_address6`。

假如配置檔已經存放在 `/etc/vsftpd/` 目錄，單獨的 `vsftpd` 進程，可以由以下命令啟動：

```
#systemctl start vsftpd@configuration-file-name.service
```

在以上的命令中，修改 `configuration-file-name` 為需要的配置檔案名稱。

例如 `vsftpd-site-2`，注意 `.conf` 尾碼是不需要添加進來的。

如果想一次性啟動多個 `vsftpd` 進程，命令如下：

```
#systemctl start vsftpd.target
#systemctl enable vsftpd.target
Created symlink
/etc/systemd/system/multi-user.target.wants/vsftpd.target →
/usr/lib/systemd/system/vsftpd.target.
```

使用 TLS 加密 `vsftpd` 連接

為了對抗 FTP 固有的不安全性(使用明文傳輸)。`vsftpd` 進程可以使用 TLS 來對連接和傳輸進行加密。FTP 客戶端必須支持 TLS。

在配置檔中 `vsftpd.conf` 設置 `ssl_enable` 為 YES 打開 TLS 的支持。

例如：配置 `vsftpd` 使用 TLS

修改 `vsftpd.conf` 配置檔；

```
ssl_enable=YES
```



```
ssl_tlsv1=YES  
ssl_sslv2=NO  
ssl_sslv3=NO
```

重啟 vsftpd service 生效；

```
#systemctl restart vsftpd.service
```

可以查看 vsftpd.conf(5)的 man 手冊瞭解更多相關資訊。

### vsftpd 的 SELinux 策略

SELinux 策略管理 vsftpd 守護進程(以及其他 ftpd 流程),定義了一個強制訪問控制,為了允許 FTP 守護進程訪問特定檔或目錄,需要分配給他們適當的標籤。

例如,為了能夠匿名共用檔,public\_content\_t 標籤必須分配給共用的檔和目錄。您可以使用 chcon 命令。

```
#chcon -R -t public_content_t /path/to/directory
```

/path/to/directory 是您想分配標籤的目錄路徑,如果您想建立一個上傳檔的目錄,您必須分配一個特使的標籤 public\_content\_rw\_t。除此之外,allow\_ftpd\_anon\_write 選項必須設置為 1,使用 setsebool 命令設置。

```
#setsebool -P allow_ftpd_anon_write=1
```

假如您想讓本地用戶通過 FTP 訪問 home 目錄,在銀河麒麟高級伺服器操作系統中這個是缺省的設置,ftp\_home\_dir 選項必須設置為 1。假如 vsftpd

允許以獨立模式運行，在銀河麒麟高級伺服器操作系統中這個是缺省的設置，`ftpd_is_daemon` 必須設置為 1。

### 6.3.3. 列印設置

列印設置工具用來印表機的配置，維護印表機配置檔，列印排隊目錄以及列印過濾和印表機管理類。

該工具是基於通用 Unix 印刷系統(CUPS)。如果你升級的系統是先前的銀河麒麟高級伺服器操作系統版本，使用的是 CUPS，在升級過程將會保存印表機資訊。

啟動印表機配置工具

從命令行啟動印表機配置工具，輸入 `system-config-printer`，印表機配置工具啟動。或者是開始->控制面板->印表機，印表機配置工具啟動如下圖。



圖 6-1 列印設置

## 6.4. 使用 **chrony** 套件配置 NTP

在 IT 行業，保持精確的時間是非常重要的，這有很多原因。例如，在網路上，包分發和日誌是需要精確的時間戳的。在 linux 系統中，NTP 協議由守護進程運行在用戶空間實現。

用戶空間的守護進程更新運行在內核空間的系統時鐘。系統時鐘能夠使用多種時鐘資源保持時間準確。通常的使用 Time Stamp Counter (TSC)。TSC 是一個 CPU 寄存器用來計算週期數。它非常快,高解析度,沒有中斷。

有兩個守護進程的選擇，`ntpd` 和 `chrony`，來自 `ntpd` 和 `chrony` 包。本節描述和 `chrony` 套件的使用的實用程式來更新系統時鐘系統,不符合傳統的永久網路化。

### 6.4.1. **chrony** 套件介紹

`Chrony` 是運行在用戶空間的守護進程的命令行程序，系統如果經常開關機，會花費很多時間通過 `ntpd` 校對系統時間。

#### 6.4.1.1. `ntpd` 和 `chronyd` 的差異

最主要的差異是用於控制電腦的時鐘的演算法。`chronyd` 能夠比 `ntpd` 做的更好。

- 當外部基準時間只能夠間歇性的訪問時 `chronyd` 能夠工作的很好。`ntpd` 需要可持續查詢時間基準才能夠工作的好；
- `chronyd` 能夠在網路擁堵的時候工作的好；
- `chronyd` 通常可以同步時鐘更快和更好的時間精度；

- **chronyd** 能夠在時鐘的速度突然變化時迅速適應,例如,由於晶體振盪器的溫度的變化,而 **ntpd** 可能需要很長時間才能再次適應下來;
- 在默認配置中,在系統啟動後時間已經同步,**chronyd** 從未設置時間,為了不打亂其他運行程式。**ntpd** 也可以配置為不同步時間,但它必須使用不同的方式調整時鐘,這會有一些缺點;
- 在 linux 操作系統上,**chronyd** 能夠在一個很大的範圍調整時間速率。這允許它在一個損壞或不穩定的機器上進行操作。例如,在一些虛擬機上。  
**chronyd** 能夠做一些 **ntpd** 不能做的事情;
- **chronyd** 支持孤立網路時間校正的唯一方法是手動輸入。例如,通過管理員看時鐘。**chronyd** 能夠在不同的更新中檢查出錯誤資訊,評估電腦過多或丟失的時間速率;
- **chronyd** 提供支持工作的收益或損失的速度即時時鐘,硬體時鐘,維護電腦是關閉的時候。它可以使用這些數據在系統啟動時設置系統時間使用調整後的即時時鐘的時間的價值。寫作時,僅可在 Linux。  
**ntpd** 可以做 **chronyd** 不能做的事情。
- **ntpd** 支持 NTP 版本 4 (RFC5905),包括廣播,多播,manycast 客戶端和服務器,和孤兒模式。它還支持額外的身份驗證方案基於公鑰加密 (RFC5906)。**chronyd** 使用 NTP 版本 3 (RFC1305),相容版本 4;
- **ntpd** 包括許多參考時鐘的驅動而 **chronyd** 依賴於其他專案,例如 **gpsd**,訪問數據的參考時鐘。

#### 6.4.1.2. NTP 守護進程的選擇

- Chrony 可以考慮在這些系統中使用，這些系統會經常暫停或間歇地斷開連接和重新連接網路，例如移動和虛擬機系統；
- NTP 守護進程（ntpd）應該考慮用在經常保持不變的系統上。系統需要使用廣播或多播 IP，或執行身份驗證數據包的 Autokey 協議，應該考慮使用 ntpd。chrony 僅僅支持對稱密鑰身份驗證，使用 MD5 消息身份驗證代碼 (MAC), SHA1 或者更強的哈希演算法。而 ntpd 還支持 Autokey 認證協議，該協議可以利用 PKI 系統。Autokey 在 RFC5906 中有描述。

### 6.4.2. 理解 CHRONY 及其配置

#### 6.4.2.1. 理解 chronyd

chronyd 是 chrony 的守護進程，運行在用戶空間，調整運行在內核的系統時鐘。它通過諮詢外部時間源，使用 NTP 協議來進行調整。當外部引用並不可用，chronyd 將使用最後被計算存儲在檔的數據。它還可以被 chronyc 手動進行修改。

#### 6.4.2.2. 理解 chronyc

chronyd 是 chrony 的守護進程，可以被命令行組件 chronyc 控制。這個工具提供了一個命令提示符輸入一些命令可以對 chronyd 進行更改。默認的配置 chronyd 僅僅接收本地的 chronyc 命令，chronyc 可以配置為 chronyd 可以接收外部控制。chronyc 可以遠程運行後第一個配置 chronyd 接受遠程連接。IP 地址允許連接到 chronyd 應嚴格控制。

理解 chrony 配置命令

`chronyd` 默認的配置檔`/etc/chrony.conf`，`-f` 選項可以指定一個配置檔的路徑。

### 注釋

注釋前應以`#`、`%`或`!` 開頭。

### **allow**

可選擇的，指定一個主機，子網或者網路連接一臺可以作為 NTP 伺服器的機器。默認是不允許連接。

例如：

```
allow server1.example.com
```

通過主機名，指定一個主機，運行連接

```
allow 192.0.2.0/24
```

指定一個網路允許連接

```
allow 2001:db8::/32
```

指定一個 IPv6 地址

### **cmdallow**

該命令和 `allow` 命令相似，除了它允許特定的子網或主機的 control 接入（而不是 NTP 客戶端接入）。（control 接入，意思是 `chronyc` 能夠運行在其他主機上並且能夠通過本地電腦連接到 `chronyd`）他的語法是一樣的。`cmddeny all` 命令和 `cmdallow all` 命令類似。

## **dumpdir**

保存 chronyd 服務重啟的測量歷史目錄路徑。

## **dumponexit**

假如該命令是運行的，它表明 chronyd 必須為所有的時間源保存測量的歷史。

## **local**

local 關鍵字是允許 chronyd 通過客戶端推送輸入進行時間同步，即使它沒有同步源。該選項經常被用來在 master 主機上使用，在一個獨立的網路中，許多電腦需要同步，master 主機需要通過手動輸入保持準確時間。

例如：

```
local stratum 10
```

## **log**

log 命令表明某些資訊將要被記錄日記。它接收以下選項：

### **measurements**

該選項記錄了測量以及相關資訊，生成 measurements.log 檔。

### **statistics**

該選項記錄了回歸處理相關資訊，生成 statistics.log 檔。

### **tracking**

該選項記錄了系統的收益或損失的估計，生成 tracking.log 檔。

### **rtc**

這個選項記錄了系統的即時時鐘資訊。

### **refclocks**

這個選項記錄了原始和過濾參考時鐘測量，生成 refclocks.log。

### **tempcomp**

這個選項記錄溫度測量和系統補償速率，生成 tempcomp.log 檔。

log 日誌記錄檔存放在 logdir 指定的目錄

```
log measurements statistics tracking
```

### **logdir**

指定日誌檔存放的目錄

```
logdir /var/log/chrony
```

### **makestep**

通常，chronyd 將根據需求通過減慢或加速時鐘，使得系統逐步糾正所有時間偏差。在某些特定情況下，系統時鐘可能會漂移過快，導致該調整過程消耗很長的時間來糾正系統時鐘。該指令強制 chronyd 在調整期大於某個閾值時步進調整系統時鐘，但只有在因為 chronyd 啟動時間超過指定限制（可使用負值來禁用限制），沒有更多時鐘更新時才生效。

```
makestep 1000 10
```

### **maxchange**

該指令將指定最大修正偏移量，當偏移量大於指定的閾值，chronyd 將會放棄並且退出，將消息發送給 syslog。

```
maxchange 1000 1 2
```

第一個時鐘更新後，chronyd 將會檢查每一個時鐘偏移量的更新，將會忽略 2 次大於閾值的偏移量修正。

### **maxupdateskew**



`chronyd` 的任務之一就是要相對於其源而言電腦的時鐘運行的快或慢的程度。`maxupdateskew` 參數是確定估計是否是否可靠的閾值，缺省情況下，閾值是 1000ppm，語法格式如下：

```
maxupdateskew skew-in-ppm
```

### **noclientlog**

這個命令沒有參數，客戶端不記錄 log 日誌。

### **reselectdist**

當 `chronyd` 選擇同步源可用的來源，它將更喜歡最小的同步距離。然而，為了避免頻繁的重新選擇有來源相似的距離時，一個固定的距離被添加。默認情況下，距離是 100 微秒。

格式如下

```
reselectdist dist-in-seconds
```

### **stratumweight**

`stratumweight` 指令設置當 `chronyd` 從可用源中選擇同步源時，每個層應該添加多少距離到同步距離。默認情況下，CentOS 中設置為 0，讓 `chronyd` 在選擇源時忽略源的層級。

格式如下

```
stratumweight dist-in-seconds
```

默認情況下，`dist-in-seconds` 為 1 秒

### **rtcfile**

`rtcfile` 指令定義了一個檔的名稱，該檔，`chronyd` 可以保存跟蹤系統的即

時時鐘的準確性(RTC)參數。語法的格式是：

```
rtcfile /var/lib/chrony/rtc
```

### **rtcsync**

`rtcsync` 指令將啟用一個內核模式，在該模式中，系統時間每 11 分鐘會拷貝到即時時鐘（RTC）。

### **chronyc 的安全性**

和 Network Time Protocol (NTP)相比，PTP 最主要的優點是硬體的支持，包括許多的 network interface controllers (NIC)和 network switches。極大的提高了時間同步的準確性。

## **6.4.3. 使用 chrony**

### **6.4.3.1. 安裝 chrony**

使用 root 用戶登錄

```
#dnf install chrony
```

默認的 `chrony` 進程位置 `/usr/sbin/chronyd`。命令行組件安裝在 `/usr/bin/chronyc`。

### **6.4.3.2. 檢查 chronyd 狀態**

```
#systemctl status chronyd
```

#### 6.4.3.3. 啟動 chronyd

啟動命令：

```
#systemctl start chronyd
```

設置開機自啟動命令：

```
#systemctl enable chronyd
```

#### 6.4.3.4. 關閉 chronyd

關閉命令：

```
#systemctl stop chronyd
```

取消開機自啟動命令：

```
#systemctl disable chronyd
```

#### 6.4.3.5. 檢查 chrony 是否同步

為了檢查 chrony 是否同步，使用 tracking, sources 和 sourcestats 命令。

檢查 chrony Tracking

命令如下：

```
#chronyc tracking
```

字段如下：

**Reference ID**

與之進行同步的 ntp 伺服器的參考 ID（一串 16 進制數字）和名稱（或 ip 地址）。假如是 127.127.1.1，表示該電腦是沒有跟外部源進行同步，您正在使用 local 模式操作。

### **Stratum**

擁有的層級數。

### **Ref time**

最後一次同步後測量的 UTC 時間。

### **System time**

系統時間。

### **Last offset**

最後一次預估的偏移量。

### **RMS offset**

偏移量的長期平均值。

### **Frequency**

如果 chronyd 沒有進行糾錯，系統時間出錯的速率，例如：1ppm 意思是系統時間 1 秒，可能實際時間是 1.000001 秒。

### **Residual freq**

顯示當前選中源的剩餘頻率。

### **Skew**

頻率上的估計誤差。

### **Root delay**

這是網路路徑延遲到最終同步電腦時的 stratum-1 電腦的總和。在某些極端的情況下，此值可以為負。

### **Root dispersion**

通過所有經過的電腦，回到最終同步的 stratum-1 電腦累積的總彌散。下

麵的公式給出了電腦時鐘精度的絕對界( 假設 stratum-1 電腦的時間是正確的 ):

$$\text{clock\_error} \leq |\text{system\_time\_offset}| + \text{root\_dispersion} + (0.5 * \text{root\_delay}).$$

### Update interval

最後兩次時鐘更新的時間間隔。

### Leap status

可能值為 Normal、Insert second、Delete second、Not synchronized。

檢查 chrony 來源

命令如下:

```
#chronyc sources
```

### M

表示源的模式。^表示一個伺服器，=表示一個 peer，#表示本地連接的參考時鐘。

### S

表示源的狀態。\*表示正在同步，+表示已連接，-表示已被排除，? 表示連接已丟失。“x”表示一個時鐘 chronyd 認為是 falseticker(與大多數其他來源的時間是不一致的)，“~”表示一個源的時間似乎有太多的變化。

### Name/IP address

源的 IP 地址或名稱。

### Stratum

源的層，層 1 表明電腦附帶本地參考時鐘，與層 1 同步的電腦在層 2，與層 2 同步的電腦在層 3，以此類推。

### Poll

顯示源被 **polled** 的速率, 例如值為 **6** 的時候, 就表示每 **64s** 進行一次測量。

### Reach

顯示最後一個收到的源的時間。一般是在幾秒鐘之間。字母 **m h d** 或 **y** 顯示分鐘, 小時, 幾天或幾年。 **10** 年的值表示沒有收到這個源。

### LastRx

顯示最後一個收到的源的時間。一般是在幾秒鐘之間。字母 **m h d** 或 **y** 顯示分鐘, 小時, 幾天或幾年。 **10** 年的值表示沒有收到這個源。

### Last sample

此列顯示本地時鐘與最新的測量數據源之間的偏移量。

### 檢查 chrony 來源統計

**sourcestats** 命令顯示當前被檢查的源的漂移速度和偏移量的資訊。 **-v** 參數能夠被指定, 意思是冗長的。在這種情況下, 額外的行顯示為一個提醒列。

```
#chronyc sourcestats
```

### Name/IP address

NTP 伺服器的地址。

### NP

這是樣本點的數量目前為伺服器保留。漂移速率和電流偏移估計通過這些點進行線性回歸。

### NR

這是運行的 **residuals** 具有相同回歸符號的數量。

### Span

這是最古老的和最新的樣本之間的時間間隔。如果沒有顯示單元的值, 則表示是在幾秒鐘內。在這個例子中, 間隔是 **46** 分鐘。

## Frequency

估計的剩餘頻率，在這種情況下，電腦的時鐘估計是  $1/10^9$  相對於伺服器來說。

## Freq Skew

估計的誤差界限。

## Offset

估計的偏移量。

## Std Dev

這是估計樣本標準差。

### 6.4.3.6. 手動調整系統時鐘

命令如下：

```
#chronyc
chrony> password commandkey-password
200 OK
chrony> makestep
200 OK
```

## 6.4.4. 為不同的環境設置 chrony

### 6.4.4.1. 為一個很少連接的系統設置 chrony

這個例子是用於系統使用 dial-on-demand 連接。正常的配置應滿足移動和虛擬設備連接斷斷續續。首先,審查和確認/etc/chrony.默認設置配置類似於以下幾點:

```
driftfile /var/lib/chrony/drift
commandkey 1
```

```
keyfile /etc/chrony.keys
```

command key ID 是在安裝時候生成，應符合 commandkey 值，  
/etc/chrony.keys。

添加 NTP 伺服器

```
server 0.pool.ntp.org offline  
server 1.pool.ntp.org offline  
server 2.pool.ntp.org offline  
server 3.pool.ntp.org offline
```

#### 6.4.4.2. 為一個獨立網路系統設置 chrony

在一個從來不和外部網路進行連接的獨立的網路中，可以選擇一臺電腦作為時間主機，其他主機作為主機的客戶端。

在 master 主機上，編輯/etc/chrony.conf

```
driftfile /var/lib/chrony/drift  
commandkey 1  
keyfile /etc/chrony.keys  
initstepslew 10 client1 client3 client6  
local stratum 8  
manual  
allow 192.0.2.0
```

192.0.2.0 是可以允許連接的子網地址

客戶端機器上，編輯/etc/chrony.conf

```
server master  
driftfile /var/lib/chrony/drift
```



```
logdir /var/log/chrony
log measurements statistics tracking
keyfile /etc/chrony.keys
commandkey 24
local stratum 10
initstepslew 20 master
allow 192.0.2.123
```

192.0.2.123 是 master 主機的地址。

### 6.4.5. 使用 **chronyc**

#### 6.4.5.1. 使用 **chronyc** 控制 **chronyd**

進入 **chronyc** 的交互介面

```
#chronyc -a
```

**chronyc** 必須以 **root** 用戶執行。**-a** 選項是使用本地 **keys** 自動登錄。

**chronyc** 命令行介面：

```
chronyc>
```

您可以使用 **help** 顯示所有命令。

也可以使用非交互介面進行輸入：

```
chronyc command
```

#### 6.4.5.2. 使用 **chronyc** 進行遠程管理

配置 **chrony** 連接到遠程 **chronyd**，格式如下：

```
#chronyc -h hostname
```

指定端口：

```
#chronyc -h hostname -p port
```

port 是用來控制和監控遠程 chronyd。

第一條命令必須輸入密碼：

```
chronyc> password password  
200 OK
```

密碼不允許有空格。

假如密碼不是 MD5 哈希，哈希密碼必須被 authhash 命令在前。

```
chronyc> authhash SHA1  
chronyc> password  
HEX:A6CFC50C9C93AB6E5A19754C246242FC5471BCDF  
200 OK
```

## 6.5. 配置 NTP 使用 NTPD

### 6.5.1. NTP 介紹

NTP 是網路時間協議(Network Time Protocol)，它是用來同步網絡中各個電腦的時間的協議。

### 6.5.2. NTP 分層

NTP 分層如下：

**Stratum 0:**

原子鐘和信號廣播電臺和 GPS

- GPS (Global Positioning System)
- Mobile Phone Systems
- Low Frequency Radio Broadcasts WWVB (Colorado, USA.), JJY-40 and JJY-60 (Japan), DCF77 (Germany), and MSF (United Kingdom)

這些信號可以被專用的設備接收，並經常被 RS-232 連接到系統作為一個組織或站點範圍內的時間伺服器。

### **Stratum 1:**

電腦附加了無線時鐘、GPS 時鐘或原子鐘。

### **Stratum 2:**

從 Stratum 1 讀取，作為更底層的伺服器。

### **Stratum 3:**

從 Stratum 2 讀取，作為更底層的伺服器。

### **Stratum n+1:**

從 Stratum n 讀取，作為更底層的伺服器。

### **Stratum 15:**

從 Stratum 14 讀取，作為更底層的伺服器。

### **6.5.3. 理解 NTP**

銀河麒麟高級伺服器操作系統使用的 NTP 版本，在 RFC 1305 Network Time Protocol (Version 3) Specification, Implementation and Analysis 和 RFC 5905 Network Time Protocol Version 4: Protocol and Algorithms Specification 有詳細描述。

#### 6.5.4. 理解 drift 檔

drift 檔記錄保存著系統時間頻率與 UTC 時鐘源頻率的偏移量。

#### 6.5.5. UTC, TIMEZONES 和 DST

NTP 完全在 UTC(Universal Time, Coordinated) 中，Timezones 和 DST(Daylight Saving Time) 被本地系統應用。/etc/localtime 是 /usr/share/zoneinfo 的拷貝或鏈接。RTC 可能在本地時間或者 UTC 中，在 /etc/adjtime 第三行指定。這個檔可以知道 RTC 怎麼被設置。用戶可以很方便的使用 System Clock Uses UTC 在 Date and Time 圖形配置介面進行配置的修改。

#### 6.5.6. NTP 身份驗證選項

在當前網路，攻擊者可以通過發送 NTP 包進行攻擊。在使用公共的 NTP 源時，為了減少風險，在/etc/ntp.conf 檔中，必須有 3 個公共源。假如只有一個源，ntpd 將會忽略該源。根據應用的風險需要，可以選擇開啟或不開啟身份驗證。

默認的組播和廣播是需要驗證的。假如您決定關閉身份驗證，可以使用 disable auth 在 ntp.conf 檔中。

#### 6.5.7. 在虛擬機中管理時間

虛擬機不能使用真實的 hardware clock 並且虛擬機時間不夠穩定。在銀河麒麟高級伺服器操作系統中，kvm 默認使用 kvm-clock。

#### 6.5.8. 理解閏秒

閏秒，是指為保持協調世界時接近於世界時時刻，由國際計量局統一規定在

年底或年中（也可能在季末）對協調世界時增加或減少 1 秒的調整。由於地球自轉的不均勻性和長期變慢性（主要由潮汐摩擦引起的），會使世界時（民用時）和原子時之間相差超過到 $\pm 0.9$  秒時，就把世界時向前撥 1 秒（負閏秒，最後一分鐘為 59 秒）或向後撥 1 秒（正閏秒，最後一分鐘為 61 秒）；閏秒一般加在西曆年末或西曆六月末。

### 6.5.9. 理解 ntpd 配置檔

守護進程 ntpd 在系統啟動或重啟時讀取配置檔/etc/ntp.conf，您可以通過下麵命令查看配置檔。

```
#less /etc/ntp.conf
```

下麵介紹默認的配置項

#### The driftfile entry

drift 檔路徑：

```
driftfile /var/lib/ntp/drift
```

#### The access control entries

以下行設置默認訪問控制限制：

```
restrict default nomodify notrap nopeer noquery
```

nomodify 選項：阻止修改配置檔；

notrap 選項：防止 ntpdc 控制消息協議陷阱；

nopeer 選項：防止 peer 聯合的形成；

noquery 選項：防止 ntpq 和 ntpdc 查詢,但沒有時間查詢。

有時候各種進程和應用需要 127.0.0.0/8 地址段。默認 restrict 行阻止所有非允許的接入：

```
#the administrative functions.  
restrict 127.0.0.1  
restrict ::1
```

如果特別需要被另一個應用程式，可以在下面添加地址。

主機在本地網路是不允許的，因為默認的 restrict。為了進行修改，例如，允許 192.0.2.0/24 網路進行時間的查詢：

```
restrict 192.0.2.0 mask 255.255.255.0 nomodify notrap  
nopeer
```

如果只是允許某一臺主機，例如 192.0.2.250/32

```
restrict 192.0.2.250
```

如果沒有指定 mask，255.255.255.255 將會被指定。

### **The public servers entry**

默認的，ntp.conf 配置檔包含四個公共伺服器：

```
server 0.rhel.pool.ntp.org iburst  
server 1.rhel.pool.ntp.org iburst  
server 2.rhel.pool.ntp.org iburst  
server 3.rhel.pool.ntp.org iburst
```

### **The broadcast multicast servers entry**

默認的，ntp.conf 包含了很多被注釋掉的例子，這些在很大程度上是自我

解釋。

### 6.5.10. 理解 ntpd 的 sysconfig 檔

這個檔將會在 ntpd 啟動服務初始化腳本的時候讀取。默認的內容如下：

```
#Command line options for ntpd
OPTIONS="-g"
```

-g 選項可以使 ntpd 忽略 1000s 的偏移量限制，嘗試同步時間即使偏移量大於 1000s，但是僅僅在系統啟動的時候。當離開這個選項的時候，ntpd 在偏移量大於 1000s 的時候自動退出。

### 6.5.11. 禁止 chrony

命令如下：

```
#systemctl stop chronyd
```

取消 chrony 開機啟動選項：

```
#systemctl disable chronyd
```

查看狀態：

```
#systemctl status chronyd
```

### 6.5.12. 檢查 NTP 守護進程是否安裝

命令如下：

```
#dnf install ntp
```

### 6.5.13. ntpd 的安裝

```
#dnf install ntp
```

設置 ntpd 開機啟動：

```
#systemctl enable ntpd
```

### 6.5.14. 檢查 ntp 的狀態

檢查 ntpd 是否運行：

```
#systemctl status ntpd
```

獲取 ntpd 的狀態報告：

```
#ntpstat
```

### 6.5.15. 配置防火牆允許 ntp 包進入

ntp 使用 UDP 包，端口 123。必須要能夠允許通過防火牆。

檢查防火牆時候允許 ntp 傳輸，使用圖形介面 **Firewall Configuration** 工具。

啟動 **firewall-config**，點擊 **Super** 鍵進入，輸入 **firewall** 並且按回車鍵，防火牆配置窗口被打開。輸入用戶密碼。

命令行進入：



```
#firewall-config
```

尋找“連接”一詞在左下角。這表明 `firewall-config` 工具連接到用戶空間守護進程, `firewalld`。

#### 6.5.15.1. 修改 firewall 配置

為了能夠讓配置生效,確保下拉菜單 **Configuration** 是設置為 `Runtime`。或者修改配置檔在下次啟動生效,在下拉菜單中選擇 `Permanent`。

#### 6.5.15.2. 打開防火牆端口

打開 `firewall-config` 工具,選擇網路,選擇端口標籤,點擊“添加”按鈕。`Port and Protocol` 窗口被打開,輸入端口 `123`,下拉菜單選擇 `udp`。

### 6.5.16. 配置 ntpdate 伺服器

`ntpdate` 服務是為了設置時間在系統啟動的時候。

檢查 `ntpdate` 服務是否運行:

```
#systemctl status ntpdate
```

設置開機啟動:

```
#systemctl enable ntpdate
```

在銀河麒麟高級伺服器操作系統系列,默認 `/etc/ntp/step-tickers` 檔包含 `0.rhel.pool.ntp.org`。添加額外的 `ntpdate` 伺服器,編輯 `/etc/ntp/step-tickers`。伺服器的數量是不重要的,因為 `ntpdate` 只是使用這個獲取一次時間資訊。假如您有一個外網的時間伺服器,在第一行輸入該伺服器的地址。在第二行輸入備份的伺服器。

## 6.5.17. 配置 ntp

修改默認的 ntp 伺服器配置，編輯/etc/ntp.conf 檔。該檔是和 ntpd 一起被安裝。

### 6.5.17.1. 配置 NTP 服務訪問控制

編輯 ntp.conf，使用 restrict 命令

```
#Hosts on local network are less restricted.  
#restrict 192.168.1.0 mask 255.255.255.0 nomodify  
notrap
```

restrict 使用

```
restrict option
```

option：可以是一個和多個

- ignore——所有的數據包將被忽略,包括 ntpq 和 ntpdc 查詢;
- kod——一個“Kiss-o'-death”包發送以減少不必要的查詢;
- limited——如果數據包違反了速率限制，不回應服務請求。ntpq 和 ntpdc 查詢不受影響;
- lowpriotrap——低優先順序匹配主機設定的陷阱;
- nomodify——阻止修改配置;
- noquery——阻止 ntpq 和 ntpdc 查詢，不包括時間查詢;
- nopeer——阻止 peer 聯合形成;
- noserve——除了 ntpq 和 ntpdc 查詢，拒絕所有的包;
- notrap——防止 ntpdc 控制消息協議陷阱;

- **notrust**——拒絕沒有密碼身份驗證數據包；
- **ntpport**——修改演算法，只匹配 NTP UDP port123；
- **version**——拒絕不匹配當前 NTP 版本的數據包。

為了不相應所有的查詢，配置速率限制，**restrict** 可以使用 **limited** 選項。

假如 **ntpd** 必須回應 **KoD** 數據包，**restrict** 必須使用 **limited** 和 **kod** 選項。

**ntpq** 和 **ntpd** 查詢可用於放大的攻擊，不要移除 **noquery** 選項。

#### 6.5.17.2. 配置連接 NTP 伺服器的速率限制

給 **restrict** 加上 **limited** 選項，如果您不想使用默認放棄的參數，可以使用 **discard** 命令。

命令格式如下：

```
discard [average value] [minimum value] [monitor value]
```

- **average**——指定允許的最小平均數據包間隔，它接受一個參數 **log2** 秒。  
默認值是 3（2 的三次方，相當於 8）；
- **minimum**——指定允許的最低包間距，在 **log2** 秒它接受一個參數。默認值是 1（2 的一次方，相當於 2 秒）；
- **monitor**——指定數據包的丟棄概率，一旦允許利率已經超過限制。默認值為 3000 秒。這個選項適用於伺服器每秒獲得 1000 或更多請求。

**discard** 命令例子

```
discard average 4
```

```
discard average 4 minimum 2
```

### 6.5.17.3. 添加 peer 地址

在 `ntp.conf` 配置檔添加 `peer` 命令：

```
peer address
```

`address` 是一個 IP 地址或 DNS 可識別的名稱。`address` 必須是在同一層的系統。`Peers` 必須擁有至少一個不同的時間源。

### 6.5.17.4. 添加伺服器地址

添加一個伺服器地址，該伺服器是運行 NTP 服務且處在更高層。在 `ntp.conf` 檔，使用 `server` 命令：

```
server address
```

`address` 是可識別的 IP 地址或 DNS 識別的名稱。

### 6.5.17.5. 添加一個廣播或多播伺服器地址

添加一個廣播或多播發送地址，也就是說，廣播或多播 NTP 包到達的地址。

在 `ntp.conf` 檔中，使用 `broadcast`。

使用如下：

```
broadcast address
```

### 6.5.17.6. 添加一個 Manycast 客戶地址

添加 `manycast` 客戶端地址，在 `ntp.conf` 檔中，添加 `manycastclient` 命令。

格式如下：

```
manycastclient address
```

該命令配置系統作為一個 NTP 客戶端。系統可以同時為客戶端和服務器端。

#### 6.5.17.7. 添加 Broadcast 客戶端地址

添加 broadcast 客戶端地址，在 ntp.conf 檔中，添加 broadcastclient 命令。

格式如下：

```
broadcastclient
```

該命令配置系統作為一個 NTP 客戶端。系統可以同時為客戶端和服務器端。

#### 6.5.17.8. 添加一個 Manycast 伺服器地址

添加 manycast 伺服器地址，在 ntp.conf 檔中，添加 manycastserver 命令。

格式如下：

```
manycastserver address
```

該命令配置系統作為一個 NTP 伺服器。系統可以同時為客戶端和服務器端。

#### 6.5.17.9. 添加一個 Multicast 伺服器地址

添加 multicast 伺服器地址，在 ntp.conf 檔中，添加 multicastclient 命令。

格式如下：

```
multicastclient address
```

該命令配置系統作為一個 NTP 伺服器。系統可以同時為客戶端和服務器端。

#### 6.5.17.10. 配置 Burst 選項

不要和公共 NTP 伺服器一起使用該選項，該選項適用於在自己組織裏的應用程式。

在伺服器命令結尾添加該選項：

```
burst
```

#### 6.5.17.11. 配置 iburst 選項

在伺服器命令結尾添加該選項：

```
iburst
```

#### 6.5.17.12. 使用 key 配置 Symmetric Authentication

在伺服器或 peer 命令後，添加該選項：

```
key number
```

number 是一個 1 到 65534 的數。該選項可以在數據包中使用 message authentication code (MAC)。該選項和 peer,server,broadcast, 和 manycastclient 命令使用。

/etc/ntp.conf，格式如下：

```
server 192.168.1.1 key 10  
broadcast 192.168.1.255 key 20
```

```
manycastclient 239.255.254.254 key 30
```

#### 6.5.17.13. 配置 Poll Interval

修改默認的 poll interval，在伺服器或 peer 命令後，添加該選項：

```
minpoll value and maxpoll value
```

#### 6.5.17.14. 配置伺服器優先順序

指定一個高優先順序的伺服器，在 server 或 peer 命令後添加該選項：

```
prefer
```

#### 6.5.17.15. 為 NTP 數據包配置 Time-to-Live

在 server 或 peer 命令後添加該選項：

```
ttl value
```

#### 6.5.17.16. 配置 NTP 使用版本

在 server 或 peer 命令後添加該選項：

```
version value
```

### 6.5.18. 配置硬體時鐘更新

配置系統時間更新硬體時鐘（RTC），在/etc/sysconfig/ntpdate 添加以下選項：

```
SYNC_HWCLOCK=yes
```

命令使用如下：

```
#hwclock --systohc
```

### 6.5.19. 配置時鐘源

在系統列出可用的時鐘源：

```
#cd /sys/devices/system/clocksource/clocksource0/  
#cat available_clocksource  
#cat current_clocksource
```

覆蓋默認的時鐘源，在內核的 GRUB 裏添加 clocksource，例如：

```
#grubby --args=clocksource=tsc  
--update-kernel=DEFAULT
```

## 6.6. 使用 ptp4l 配置 PTP

### 6.6.1. PTP 介紹

Precision Time Protocol (PTP)是用於網路中時鐘同步的協議。使用時結合硬體支持，要比普通的 NTP 更快。PTP 的支持被劃分為內核和用戶空間。銀河麒麟高級伺服器操作系統 linux 內核支持 PTP 時鐘。linuxptp 是該協議的實現。

這個 linuxptp 包包含 ptp4l 和 phc2sys 時鐘同步專案。ptp4l 程式實現了 PTP 邊界時鐘和普通時鐘。與硬體時間戳，它用於 PTP 硬體時鐘與主時鐘同步，與軟體時間戳，它用於系統時鐘與主時鐘同步。phc2sys 程式只需與硬體時間戳，將系統時鐘同步到 PTP 硬體時鐘網路介面卡(NIC)。



### 6.6.1.1. PTP 的優點

和 Network Time Protocol (NTP)相比, PTP 最主要的優點是硬體的支持, 包括許多的 network interface controllers (NIC)和 network switches。極大的提高了時間同步的準確性。

## 6.6.2. 使用 PTP

為了使用 PTP, 內核網路驅動必須支持軟體或硬體時間戳。

### 6.6.2.1. 檢查驅動和硬體支持

使用 ethtool 命令查詢:

```
#ethtool -T eth3
```

eth3 是您想檢查的介面

如果支持軟時間戳, 必須包含以下參數

- > SOF\_TIMESTAMPING\_SOFTWARE
- > SOF\_TIMESTAMPING\_TX\_SOFTWARE
- > SOF\_TIMESTAMPING\_RX\_SOFTWARE

如果支持硬體時間戳, 必須包含以下參數

- > SOF\_TIMESTAMPING\_RAW\_HARDWARE
- > SOF\_TIMESTAMPING\_TX\_HARDWARE
- > SOF\_TIMESTAMPING\_RX\_HARDWARE

### 6.6.2.2. 安裝 PTP

```
#dnf install linuxptp
```

這將會安裝 ptp4l 和 phc2sys。

### 6.6.2.3. 啟動 ptp4l

ptp4l 可以通過命令行或者作為服務啟動。當 ptp4l 作為一個服務時，可以在 `/etc/sysconfig/ptp4l` 指定選項。不管是作為服務或是命令行啟動，必須在 `/etc/ptp4l.conf` 指定選項。`/etc/sysconfig/ptp4l` 檔包含 `-f/etc/ptp4l.conf` 行，這會讓 ptp4l 讀取 `/etc/ptp4l.conf` 檔。

作為服務啟動 ptp4l:

```
#systemctl start ptp4l
```

使用命令行運行 ptp4l:

```
#ptp4l -i eth3 -m
```

輸出結果如下:

```
ptp4l[81921.804]: selected /dev/ptp0 as PTP clock
ptp4l[81921.805]: driver changed our HWTSTAMP options
ptp4l[81921.806]: tx_type 1 not 1
ptp4l[81921.806]: rx_filter 1 not 12
ptp4l[81921.806]: port 1: INITIALIZING to LISTENING on
INIT_COMPLETE
ptp4l[81921.806]: port 0: INITIALIZING to LISTENING on
INIT_COMPLETE
ptp4l[81929.604]: port 1: LISTENING to MASTER on
ANNOUNCE_RECEIPT_TIMEOUT_EXPIRES
```

記錄 ptp4l 日誌

默認情況下，日誌檔是發送到 `/var/log/messages`。 `-m` 選項打開日誌的標準輸出，能夠為 debugging 提供資訊。

打開軟時間戳， `-s` 選項，使用如下:

```
#ptp4l -i eth3 -m -S
```

選擇一個延遲測量機制

有 2 種不同的延遲測量機制，能夠通過不同的選項進行選擇：

-P

-P 選擇 peer-to-peer(P2P) 延遲測量機制；

P2P 機制是首選，因為它對網路拓撲的變化更快，可能比其他機制更準確測量延遲。

-E

-E 選擇 end-to-end(E2E)延遲測量機制。這是默認的選項；

E2E 也被認為是 request-response 延遲機制。

-A

-A 打開延遲機制的自動選擇；

自動選項打開的是 ptp4l 的 E2E 模式。如有需要，可以改成 P2P 模式。

### 6.6.3. 和多個介面使用 PTP

在不同網路的多介面中使用 PTP，有必要改變反向路徑轉發模式為鬆散模式。

sysctl 組件用來對內核進行讀寫。對正在運行的系統，可以使用命令行或者是修改/etc/sysctl.conf 檔。

修改為 loose 模式，命令如下：

```
#sysctl -w net.ipv4.conf.default.rp_filter=2  
#sysctl -w net.ipv4.conf.all.rp_filter=2
```

為了對每一個介面修改反向路徑轉發模式，使用

net.ipv4.interface.rp\_filter 命令行，例如，em1 網路介面：

```
#sysctl -w net.ipv4.conf.em1.rp_filter=2
```

為了讓修改的配置永久有效，修改/etc/sysctl.conf 檔。例如：為了修改所有的網卡的模式，修改/etc/sysctl.conf，如下所示：

```
net.ipv4.conf.all.rp_filter=2
```

如果是修改單一的某個網卡模式，如下所示：

```
net.ipv4.conf.interface.rp_filter=2
```

#### 6.6.4. 指定一個配置檔

沒有默認的配置檔，所以需要在運行的時候指定，使用-f 選項：

```
#ptp4l -f /etc/ptp4l.conf
```

一個配置檔內容，相當於-i eth3 -m -S 選項，如下所示：

```
#cat /etc/ptp4l.conf
```

#### 6.6.5. 使用 PTP 管理客戶端

PTP 的管理客戶端，pmc 可以用來獲取額外 ptp4l 資訊。

```
#pmc -u -b 0 'GET CURRENT_DATA_SET'
```

```
#pmc -u -b 0 'GET TIME_STATUS_NP'
```

查看 pmc 全部命令：

```
#pmc --help
```

### 6.6.6. 同步時鐘

phc2sys 程式用來將系統時間同步到 PTP 的 hardware clock (PHC)。

phc2sys 服務配置檔/etc/sysconfig/phc2sys。默認配置如下：

```
OPTIONS="-a -r"
```

-a 選項使得 phc2sys 能夠同步來自 ptp4l 應用的時鐘。它將跟隨 PTP 端口狀態的變化，相應調整網卡的硬體之間的同步時鐘，系統時鐘將不會被同步，除非-r 選項被指定。如果您想讓系統時鐘成為一個源，指定-r 選項兩次。

修改/etc/sysconfig/phc2sys 後，重啟 phc2sys 服務：

```
#systemctl restart phc2sys
```

正常情況下，使用 systemctl 命令來啟動，停止和重啟 phc2sys 服務。

如果您不想使用服務啟動 phc2sys，您可以通過命令行來啟動：

```
#phc2sys -a -r
```

-a 選項使得 phc2sys 能夠同步來自 ptp4l 應用的時鐘。如果您想讓系統時鐘成為一個源，指定-r 選項兩次。

使用-s 選項同步系統時鐘至特定的 PTP 硬體時鐘，例如：

```
#phc2sys -s eth3 -w
```

-w 選項等待運行的 ptp4l 應用同步 PTP 時鐘，恢復 TAI 至 UTC 偏移量

正常情況下，PTP 操作在 International Atomic Time(TAI)，系統時間保持在 Coordinated Universal Time(UTC)。當前 TAI 和 UTC 的偏移量是 35s。當閏秒插入或刪除的時候，偏移量會進行改變，這個變化每幾年會發生一次。當

-w 選項沒有使用時，可以使用-O 選項來進行偏移量的設置：

```
#phc2sys -s eth3 -O -35
```

當 phc2sys servo 處於鎖狀態時，時鐘將不會走，除非-S 選項能夠被使用。這意味著 phc2sys 程式必須在 ptp4l 程式以及和 PTP 硬體時鐘同步後啟動。使用-w 選項，phc2sys 不用在 ptp4l 後啟動，因為它會等待，直到 ptp4l 已經同步。

phc2sys 可以作為服務啟動：

```
#systemctl start phc2sys
```

當以服務進行啟動，可以將選項在/etc/sysconfig/phc2sys 檔指定。

### 6.6.7. 驗證時間同步

當 PTP 時間同步工作正常的時候，新的頻率偏移量以及調整消息將會列印 ptp4l 和 phc2sys。這些資訊可以在/var/log/messages 檔查看。

```
ptp4l[352.359]: selected /dev/ptp0 as PTP clock
ptp4l[352.361]: port 1: INITIALIZING to LISTENING on
INITIALIZE
ptp4l[352.361]: port 0: INITIALIZING to LISTENING on
INITIALIZE
.....
```

phc2sys 輸出的例子

```
phc2sys[526.527]: Waiting for ptp4l...
phc2sys[527.528]: Waiting for ptp4l...
phc2sys[528.528]: phc offset      55341 s0 freq      +0
delay  2729
```

```
phc2sys[529.528]: phc offset      54658 s1 freq  -37690
delay    2725
.....
```

ptp4l 有一個命令 `summary_interval`，可以減少輸出，默認情況下，每一秒會列印一條資訊。可以修改為每 1024s 列印一次，在 `/etc/ptp4l.conf` 檔添加以下行：

```
summary_interval 10
```

一個 ptp4l 輸出的例子，使用 `summary_interval 6`

```
ptp4l: [615.253] selected /dev/ptp0 as PTP clock
ptp4l: [615.255] port 1: INITIALIZING to LISTENING on
INITIALIZE
.....
```

減少從 `phc2sys` 的輸出，可以使用 `-u` 選項

```
#phc2sys -u summary-updates
```

`summary-updates` 是時鐘更新數，例子如下：

```
#phc2sys -s eth3 -w -m -u 60
```

### 6.6.8. 使用 NTP 服務 PTP 時間

`ntpd` 守護進程可以配置從系統時間分發時間，系統時間是由 `ptp4l` 和 `phc2sys` 使用 LOCAL 時間驅動進行同步。為了防止 `ntpd` 調整系統時鐘，`ntp.conf` 檔必須不能夠指定任何 NTP 伺服器。以下是一個例子：

```
#cat /etc/ntp.conf
server 127.127.1.0
fudge 127.127.1.0 stratum 0
```

### 6.6.9. 使用 PTP 服務 NTP 時間

NTP 至 PTP 同步也是可能的。當 ntpd 是用來同步系統時間，ptp4l 可以使用 priority1 選項配置為 grandmaster 時鐘，通過 PTP 系統時鐘來分發時間。

```
#cat /etc/ptp4l.conf
[global]
priority1 127
[eth3]
#ptp4l -f /etc/ptp4l.conf
```

使用硬體時間戳，需要使用 phc2sys 來同步 PTP 硬體時鐘至系統時鐘。假如 phc2sys 是一個服務，編輯/etc/sysconfig/phc2sys 檔，默認的配置如下

```
OPTIONS="-a -r"
```

使用 root 用戶，編輯

```
#vi /etc/sysconfig/phc2sys
OPTIONS="-a -r -r"
```

在這裏，-r 選項被指定 2 次，允許 PTP 網卡硬體時鐘從系統時鐘進行同步。

重啟 phc2sys:

```
#systemctl restart phc2sys
```

防止 PTP 時鐘頻率快速的變化，可以通過設置更小的 P (proportional) 和

I (integral):

```
#phc2sys -a -r -r -P 0.01 -I 0.0001
```



### 6.6.10. 使用 **timemaster** 同步 **PTP** 或 **NTP** 時間

可以使用 **timemaster** 程式讓系統時間跟可用的時間源進行同步。PTP 時間是由 **phc2sys** 和 **ptp4l** 提供，通過使用 **shared memory driver**。NTP 守護進程能夠比較所有的源，選擇最優的源進行同步。

啟動時，**timemaster** 會讀取一個配置檔，該配置檔會指定 NTP 和 PTP 時間源，檢查哪些網路介面有自己或共用的 PTP 硬體時鐘，生成 **ptp4l** 和 **chronyd** 或 **ntpd** 配置檔，啟動 **ptp4l**，**phc2sys** 和 **phc2sys** 或 **ntpd**。

#### 6.6.10.1. 作為一個服務啟動 **timemaster**

使用 **root** 用戶執行如下命令：

```
#systemctl start timemaster
```

啟動時，會讀取 **/etc/timemaster.conf** 配置檔。

#### 6.6.10.2. 理解 **timemaster** 配置檔

默認的配置檔如下所示：

```
#less /etc/timemaster.conf
#Configuration file for timemaster

#[ntp_server ntp-server.local]
#minpoll 4
#maxpoll 4

#[ptp_domain 0]
#interfaces eth0
```

```
[timemaster]
ntp_program chronyd

[chrony.conf]
include /etc/chrony.conf

[ntp.conf]
includefile /etc/ntp.conf

[ptp4l.conf]

[chronyd]
path /usr/sbin/chronyd
options -u chrony

[ntpd]
path /usr/sbin/ntpd
options -u ntp:ntp -g

[phc2sys]
path /usr/sbin/phc2sys

[ptp4l]
path /usr/sbin/ptp4l
```

注意到部分命名如下：

```
[ntp_server address]
```

這是一個 NTP 伺服器區域的例子，ntp-server.local 是一個本地 LAN 的 NTP 伺服器。

注意到部分命名如下：

```
[ptp_domain number]
```

PTP domain 是一組 PTP 時鐘，它們能夠相互同步。它們可以或者不可以和其他域進行同步。擁有相同域的數字組成一個域。這包括一個 PTP 的 grandmaster 時鐘。

注意到部分命名如下：

```
[timemaster]
```

默認的 timemaster 配置檔包含了 ntpd 和 chrony 配置（/etc/ntp.conf 或者/etc/chronyd.conf），為了能夠包含訪問限制的配置和認證密鑰。這意味著任何指定的 NTP 伺服器能夠和 timemaster 一起被使用。

### 6.6.10.3. 配置 timemaster 選項

#### 實例：編輯 timemaster 配置檔

- 1) 打開/etc/timemaster.conf 配置檔；
- 2) 對於每一個 NTP 伺服器，您想控制使用 timemaster，創建[ntp\_server address]字段；
- 3) 添加要在域中使用的網卡，編輯#[ptp\_domain 0]字段並添加網卡，例子如下：

```
[ptp_domain 0]
    interfaces eth0

[ptp_domain 1]
    interfaces eth1
```

- 4) 假如需要使用 ntpd 作為 NTP 守護進程，在[timemaster]字段修改默認的

entry, chronyd 改為 ntpd;

- 5) 假如使用 chronyd 作為 NTP 伺服器，在[chrony.conf]字段添加額外的選項，在默認的 include /etc/chrony.confentry 下；
- 6) 假如使用 ntpd 作為 NTP 伺服器，在[chrony.conf]字段添加額外的選項，在默認的 include /etc/ntp.confentry 下；
- 7) 在[ptp4l.conf]字段，添加任何將要拷貝到 ptp4l 生成的配置檔的選項；
- 8) 在[chronyd]字段，添加任何命令行，當被 timemaster 訪問時該命令需要傳遞至 chronyd；
- 9) 在[ntpd]字段，添加任何命令行，當被 timemaster 訪問時該命令需要傳遞至 ntpd；
- 10)在[phc2sys]字段，添加任何命令行，當被 timemaster 訪問時該命令需要傳遞至 phc2sys；
- 11)在[ptp4l]字段，添加任何命令行，當被 timemaster 訪問時該命令需要傳遞至 ptp4l；
- 12)保存配置檔，重啟 timemaster。

```
#systemctl restart timemaster
```

#### 6.6.11. 提高準確性

ptp4l 和 phc2sys 應用能夠配置為使用新的 adaptive servo。對於 PI servo 來說，它的優勢在於不需要配置 PI 優化配置檔。在 ptp4l 中使用，需要在/etc/ptp4l.conf 配置檔添加以下行：

```
clock_servo linreg
```

重啟 ptp4l 服務：

```
#systemctl restart ptp4l
```

在 phc2sys 中使用，需要在/etc/sysconfig/phc2sys 配置檔添加以下行：

```
-E linreg
```

重啟 phc2sys 服務：

```
#systemctl restart phc2sys
```

## 第七章 監控和自動化

### 7.1. 系統監控工具

在配置系統之外，掌握收集基本的系統資訊的方法也很重要。譬如，您應該知道如何找出空閒記憶體的数量、可用硬碟空間，硬碟分區方案，以及正在運行進程的資訊等等。本節將說明如何使用幾個簡單程式來從您的銀河麒麟伺服器操作系統中檢索這類資訊。

#### 7.1.1. 查看系統進程

##### 7.1.1.1. 使用 ps 命令

Ps 命令顯示系統運行進程的資訊。它會生成一個靜態列表，列表裏的進程是當您執行命令時系統運行的進程的一個快照。如果您想持續更新即時進程列表，您需要使用 top 命令和系統監控應用。

如果想列出當前系統中的所有進程，可以在 shell 裏使用如下命令：

```
ps ax
```

對於每一個列出的進程，ps ax 命令會顯示進程的 PID，進程依附的終端，進程狀態，進程佔用的 CPU 時間，可執行檔的名字。例如：

```
[root@localhost ~]# ps ax
  PID TTY          STAT       TIME COMMAND
    1 ?           Ss        0:00 /usr/lib/systemd/systemd --switched-root
    2 ?           S          0:00 [kthreadd]
    3 ?           I<        0:00 [rcu_gp]
    4 ?           I<        0:00 [rcu_par_gp]
    6 ?           I<        0:00 [kworker/0:0H-kblockd]
    8 ?           I<        0:00 [mm_percpu_wq]
   10 ?           S          0:00 [ksoftirqd/0]
   11 ?           I          0:00 [rcu_sched]
   12 ?           I          0:00 [rcu_bh]
   13 ?           S          0:00 [migration/0]
   14 ?           S          0:00 [cpuhp/0]
   16 ?           S          0:00 [kdevtmpfs]
   17 ?           I<        0:00 [netns]
```

如果想顯示進程的所有者，使用如下命令：

```
ps aux
```

除了 `ps ax` 命令提供的資訊外，`ps aux` 還顯示進程擁有者的名字，進程佔用 CPU 和記憶體的比例，以位元組為單位顯示虛擬記憶體大小和未交換物理記憶體的大小，進程開始運行的時間。

您可以使用 `ps` 命令和 `grep` 命令的組合來查看某進程是否在運行。譬如，要判定 Emacs 是否在運行，使用下面這個命令：

```
#ps ax | grep emacs
```

有關可用命令行選項的完整列表，查看 `ps(1) man` 手冊。

#### 7.1.1.2. 使用 top 命令

`Top` 命令顯示系統運行中進程的即時列表。它還會顯示附加資訊包括系統更新時間，當前 CPU 和記憶體的使用率，運行的進程總數。這樣您可以對進程做更進一步的操作，例如可以給進程排序或者殺死一個進程。

運行 `top` 命令，在 `shell` 裏輸入如下命令：

```
top
```

對於列出的進程，`top` 命令會顯示進程的 `PID`，進程所有者的名字，進程優先順序，進程 `NICE` 值，進程使用的虛擬記憶體，進程使用的未交換的物理記憶體，進程使用的共用記憶體，進程的狀態，進程使用記憶體和 CPU 的百分比，進程佔用 CPU 的時間和可執行檔的名字。

下表可以和 `top` 一起使用的互動命令：

表 7-1 交互 top 命令

命令	描述
[Space]	立即刷新顯示。
[h]	顯示幫助螢幕。
[k]	殺死某進程。您會被提示輸入進程 ID 以及要發送給它的信號。
[n]	改變要顯示的進程數量。您會被提示輸入數量。
[u]	按用戶排序。
[M]	按記憶體用量排序。
[P]	按 CPU 使用率排序。
[q]	退出 top。

### 7.1.1.3. 使用系統監控工具

用戶可以在系統監控工具的圖形介面上查看和查找進程，改變進程的優先順序以及殺死進程。開始->所有程式->系統工具->系統監視器點擊打開，如下圖：



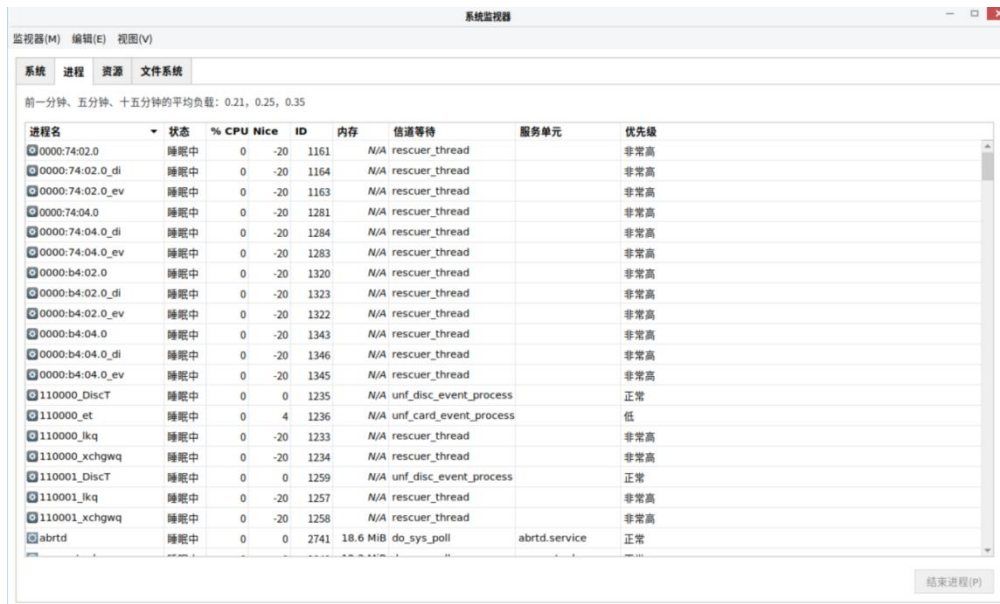


圖 7-1 進程

對於列出的進程，系統監控工具會顯示進程的名字、狀態、進程 NICE 值、進程使用記憶體和 CPU 的百分比、進程的 PID、進程等待的通道和進程會話的其他細節。通過點擊進程名字欄可以將進程顯示的資訊進行昇冪或降冪排列。

默認系統監控工具會顯示當前登錄用戶的進程列表，通過選擇查看菜單下的不同選項，您可以做如下事情：

- 查看活動的進程；
- 查看所有進程；
- 查看當前登錄用戶的進程；
- 查看進程依賴；

另外，有兩個按鈕有如下作用：

- 刷新進程列表；
- 選中進程後殺死進程。

## 7.1.2. 查看記憶體使用情況

### 7.1.2.1. 使用 free 命令

使用 **free** 命令可以查看系統空間和已經使用的記憶體，在 **shell** 下輸入如下命令：

```
free
```

**Free** 命令可以提供物理記憶體和交換空間的資訊。它可以顯示記憶體總量，使用的記憶體大小，空間記憶體大小，共用記憶體大小，**buff** 的大小和 **cache** 的大小以及是否可用。例子如下：

```
#free
```

默認 **free** 以位元組顯示大小，如果想以兆為單位可以加 **-m** 選項，如下：

```
#free -m
```

### 7.1.2.2. 使用系統監控工具

系統監控工具的資源標籤頁可以查看系統中空閒的記憶體大小和已經使用的大小，點擊資源標籤頁查看系統記憶體使用情況。

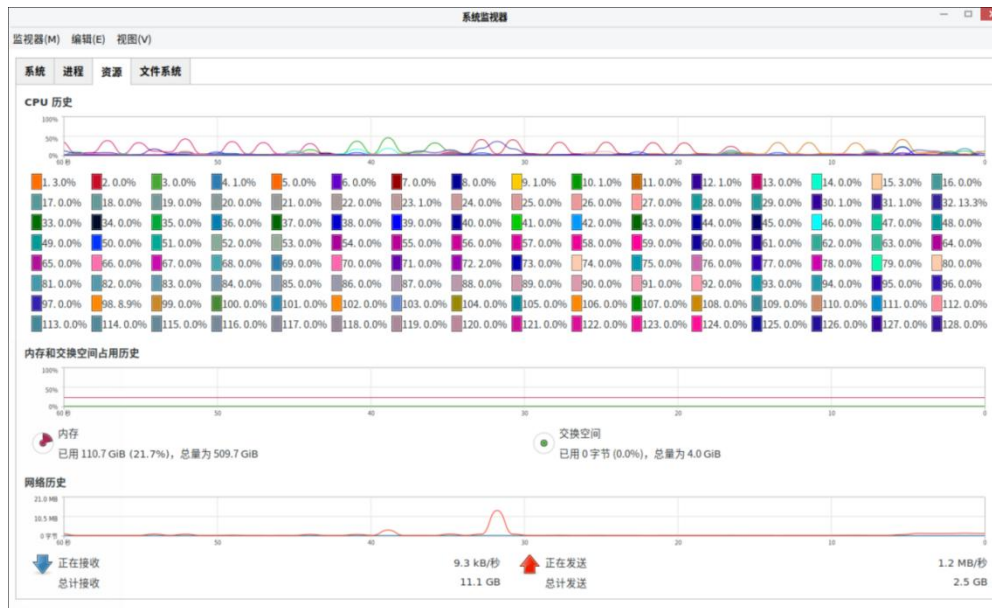


圖 7-2 資源

在內存和交換分區章節，系統監控工具顯示記憶體和交換分區的歷史使用圖表和物理記憶體和交換分區的總大小以及使用率。

### 7.1.3. 查看 CPU 使用

#### 7.1.3.1. 使用系統監控工具

點擊資源標籤頁查看系統的 CPU 使用情況，在 CPU 章節，系統監控工具顯示 CPU 的歷史使用圖表和 CPU 當前使用率的圖表查看塊設備和文件系統。

#### 7.1.3.2. 使用 lsblk 命令

lsblk 命令可以顯示系統可以使用的塊設備。它比 blkid 命令提供更多的資訊和輸出格式控制。它從 udev 讀取資訊，因此非 root 用戶都可以使用。顯示塊設備列表需要在 shell 輸入如下命令：

```
lsblk
```

每一塊輸出的塊設備，lsblk 都顯示設備名，主次設備號，設備是否可以刪除，設備檔大小，設備是否是只讀，設備類型，設備掛載路徑。例子如下：

```
#lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sr0 11:0 1 1024M 0 rom
vda 252:0 0 20G 0 rom
|-vda1 252:1 0 500M 0 part /boot
`-vda2 252:2 0 19.5G 0 part
|-vg_kvm-lv_root (dm-0) 253:0 0 18G 0 lvm /
`-vg_kvm-lv_swap (dm-1) 253:1 0 1.5G 0 lvm [SWAP]
```

默認 `lsblk` 命令以樹形結構輸出塊設備，如果想獲取更多的設備資訊添加 `-l` 參數，例如：

```
#lsblk -l
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sr0 11:0 1 1024M 0 rom
vda 252:0 0 20G 0 rom
vda1 252:1 0 500M 0 part /boot
vda2 252:2 0 19.5G 0 part
vg_kvm-lv_root (dm-0) 253:0 0 18G 0 lvm /
vg_kvm-lv_swap (dm-1) 253:1 0 1.5G 0 lvm [SWAP]
```

有關可用命令行選項的完整列表，查看 `lsblk(8) man` 手冊。

### 7.1.3.3. 使用 `blkid` 命令

`blkid` 命令可以顯示可用塊設備底層資訊。它需要 `root` 許可權，因此非 `root` 用戶只能使用 `lsblk` 命令，以 `root` 身份在 `shell` 輸入如下命令：

```
blkid
```

每一個列出的塊設備，`blkid` 會顯示它的可用屬性，例如 `UUID`、檔系統類型、卷標籤。例子如下：

```
#blkid
/dev/vda1: UUID="7fa9c421-0054-4555-b0ca-b470a97a3d84"
```

```
TYPE="ext4"  
  /dev/vda2: UUID="7lvYzk-TnnK-oPjf-ipdD-cofz-DXaj-gPdgBW"  
  TYPE="LVM2_member"  
  /dev/mapper/vg_kvm-lv_root:  
  UUID="a07b967c-71a0-4925-ab02-aebcad2ae824"  
  TYPE="ext4"  
  /dev/mapper/vg_kvm-lv_swap:  
  UUID="d7ef54ca-9c41-4de4-ac1b-4193b0c1ddb6"  
  TYPE="swap"
```

默認 `blkid` 命令會顯示所有可用的塊設備。如果想顯示某一塊設備，需要在命令後面加設備名：

```
blkid device_name
```

例如，想顯示設備 `/dev/vda1` 的資訊，以 `root` 用戶許可權輸入命令：

```
#blkid /dev/vda1  
/dev/vda1: UUID="7fa9c421-0054-4555-b0ca-b470a97a3d84"  
TYPE="ext4"
```

在上面的命令中加入 `-p` 和 `-o` 選項可以獲取更多細節資訊，命令同樣需要 `root` 許可權。

```
blkid -po udev /dev/vda1
```

例如：

```
#blkid -po udev /dev/vda1  
ID_FS_UUID=7fa9c421-0054-4555-b0ca-b470a97a3d84  
ID_FS_UUID_ENC=7fa9c421-0054-4555-b0ca-b470a97a3d84  
ID_FS_VERSION=1.0  
ID_FS_TYPE=ext4  
ID_FS_USAGE=filesystem
```

有關可用命令行選項的完整列表，查看 `blkid(8) man` 手冊。

### 7.1.3.4. 使用 findmnt 命令

findmnt 命令顯示系統當前掛載的檔系統，在 shell 裏輸入如下命令：

```
findmnt
```

每一個列出的檔系統，findmnt 命令顯示掛載點，原設備，檔系統類型和有關的掛載選項。

例如：

```
~]$ findmnt
TARGET                                SOURCE                                FSTYPE
OPTIONS
/                                       /dev/mapper/rhel-root                xfs
rw,relatime,seclabel,attr2,inode64,noquota
├-/proc                                proc                                  proc
rw,nosuid,nodev,noexec,relatime
├-/proc/sys/fs/binfmt_misc            systemd-1                             autofs
rw,relatime,fd=32,pgrp=1,timeout=300,minproto=5,maxproto=5,direct
├-/proc/fs/nfsd                        sunrpc                                  nfsd
rw,relatime

├-/sys                                  sysfs                                  sysfs
rw,nosuid,nodev,noexec,relatime,seclabel
├-/sys/kernel/security                securityfs                              securityfs
rw,nosuid,nodev,noexec,relatime
├-/sys/fs/cgroup                       tmpfs                                    tmpfs
rw,nosuid,nodev,noexec,seclabel,mode=755
[output truncated]
```

默認 findmnt 以樹型結構輸出檔系統，如果想獲取更多的設備資訊添加-l

參數，例如：

```
#findmnt -l
```

```
TARGET      SOURCE      FSTYPE  OPTIONS
/sys        sysfs       sysfs   rw,nosuid,nodev,noexec,relatime
/proc       proc        proc    rw,nosuid,nodev,noexec,relatime
/dev        devtmpfs    devtmpf rw,nosuid,size=997904k,nr_inodes=2
/sys/kernel/security securityfs  securit rw,nosuid,nodev,noexec,relatime
/dev/shm    tmpfs       tmpfs   rw,nosuid,nodev
/dev/pts    devpts      devpts  rw,nosuid,noexec,relatime,gid=5,mo
```

您可以選擇只輸出某種類型的檔系統，使用-t 選項，後面寫檔系統類型，例

如：

```
findmnt -t {type}
```

例如，顯示所有的 xfs 檔系統：

```
[root@localhost ~]# findmnt -t xfs
TARGET SOURCE FSTYPE OPTIONS
/ /dev/mapper/klas-root xfs rw,relatime,attr2,inode64,noquota
└─/boot /dev/sda1 xfs rw,relatime,attr2,inode64,noquota
```

有關可用命令行選項的完整列表，查看 `findmnt(8) man` 手冊。

### 7.1.3.5. 使用 df 命令

`df` 命令輸出系統磁片空間的使用報告，在 `shell` 裏輸入如下命令：

```
df
```

每一個列出的檔系統，`df` 命令都會顯示檔系統的名字，大小（以 K 位元組為單位），磁片空間使用率和使用大小，磁片空間剩餘大小和文件掛載點。例子如下：

```
#df
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/mapper/vg_kvm-lv_root 18618236 4357360 13315112
25% /
tmpfs 380376 288 380088 1% /dev/shm
/dev/vda1 495844 77029 393215 17% /boot
```

默認 `df` 命令顯示分區大小（以 K 位元組為單位），磁片空間使用總量，磁片剩餘空間可用量（以 K 位元組為單位）。想以 M 位元組和 G 位元組顯示需要使用 `-h` 選項，該選項可以以易讀方式的格式顯示磁片空間大小。

例如：

```
#df -h
Filesystem Size Used Avail Use% Mounted on
/dev/mapper/vg_kvm-lv_root 18G 4.2G 13G 25% /
```

```
tmpfs 372M 288K 372M 1% /dev/shm
/dev/vda1 485M 76M 384M 17% /boot
```

有關可用命令行選項的完整列表，查看 `df(1) man` 手冊。

### 7.1.3.6. 使用 du 命令

`du` 命令可以查看檔的大小。`du` 命令不加參數可以顯示每個子目錄中文件的大小。例如：

```
#du
14972 ./Downloads
4 ./mozilla/extensions
4 ./mozilla/plugins
12 ./mozilla
15004 .
```

默認情況下，`du` 命令以千位元組為單位顯示磁片的使用情況。如果想以易讀方式（MB 和 GB）顯示大小，可以添加參數 `-h`。例如：

```
#du -h
15M ./Downloads
4.0K ./mozilla/extensions
4.0K ./mozilla/plugins
12K ./mozilla
15M .
```

在列表末尾 `du` 命令會顯示出當前目錄所有檔大小的和，如果只顯示目錄中文件的總大小可以添加參數 `-s`，例如：

```
#du -sh
15M .
```

有關可用命令行選項的完整列表，查看 `du(1) man` 手冊。



### 7.1.3.7. 使用系統監控工具

在系統監控工具的檔系統標籤頁可以以圖表的方式查看檔系統和磁片空間的使用情況。

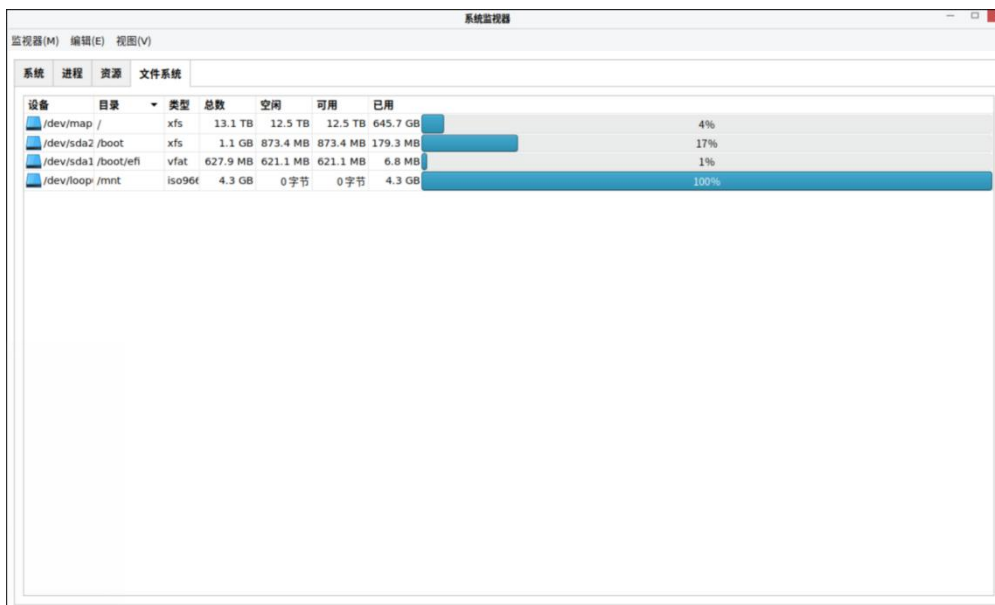


圖 7-3 檔系統

每一個列出的檔系統，系統監控工具都會顯示原設備，目標掛載點，檔系統類型和大小，磁片空間使用大小和未使用大小。

### 7.1.4. 查看硬體資訊

#### 7.1.4.1. 使用 lspci 命令

lspci 命令可以顯示 PCI 匯流排上的設備資訊，顯示所有的 PCI 設備在 shell 裏輸入如下命令：

```
#lspci
```

您可以使用 -v 選項輸出設備詳細資訊，使用 -vv 選項輸出設備更詳細的資訊。

```
#lspci -v
```

有關可用命令行選項的完整列表，查看 `lspci(8) man` 手冊。

#### 7.1.4.2. 使用 `lsusb` 命令

`lsusb` 命令可以顯示 `usb` 設備資訊。顯示所有的 `usb` 設備資訊在 `shell` 裏輸入如下命令：

```
#lsusb
```

您可以添加 `-v` 參數顯示設備的詳細資訊。

```
#lsusb -v
```

有關可用命令行選項的完整列表，查看 `lsusb(8) man` 手冊。

#### 7.1.4.3. 使用 `lscpu` 命令

`lscpu` 命令顯示當前系統的 `cpu` 資訊，資訊包括 `cpu` 數目，架構，型號，家族，模式，`cpu` 高速緩存等等，在 `shell` 輸入命令：

```
#lscpu
```

有關可用命令行選項的完整列表，查看 `lscpu(1) man` 手冊。

#### 7.1.5. 檢查硬體錯誤

銀河麒麟高級伺服器操作系統引入了新的硬體事件報告機制。這個機制會為 `DIMMs` 收集記憶體錯誤和錯誤檢查和糾錯機制報告的錯誤，並且把錯誤向用戶空間報告。用戶空間的守護進程 `rasdaemon` 會捕獲和處理這些由內核機制跟蹤的有可靠性可用性可維護性的錯誤事件，並記錄它們。這個功能以前由 `edac-utils` 提供，現在由守護進程 `rasdaemon` 提供。

安裝 `rasdaemon` 需要 `root` 許可權。

```
#dnf install rasdaemon
```

啟動服務命令如下

```
#systemctl start rasdaemon
```

輸入下麵的命令查看命令的各種選項

```
#rasdaemon --help
```

命令在 `rasdaemon(8) man` 手冊裏也有描述。

`ras-mc-ctl` 工具提供了一種可以使用 `EDAC` 驅動的方法，輸入如下命令可以查看命令的選項。

```
#ras-mc-ctl --help
```

命令在 `ras-mc-ctl(8) man` 手冊裏也有描述。

### 7.1.6. 使用 **Net-SNMP** 監控性能

銀河麒麟高級伺服器操作系統包括 `Net-SNMP` 軟體套件，其中包括一個靈活的可擴展的簡單網路管理協議（`SNMP`）代理。該代理及其關聯的實用程式可以被用於從大量的系統中提供性能數據給一系列工具，這些工具都支持 `SNMP` 協議。

本節提供了關於配置 `Net-SNMP` 代理通過網路安全地提供性能數據，使用 `SNMP` 協議檢索數據和擴展 `SNMP` 代理以提供自定義的性能指標。

#### 7.1.6.1. 安裝 `Net-SNMP`

`Net-SNMP` 軟體套件可作為銀河麒麟高級伺服器操作系統 V10 發行版的一

組 RPM 軟體包。表 7-2 可用 Net-SNMP 軟體包概述了每個包和它們的內容。

**表 7-2 可用 Net-SNMP 軟體包**

軟體包	提供商
net-snmp	SNMP 代理守護進程和文檔。輸出性能數據需要這個包。
net-snmp-libs	netsnmp 庫和捆綁的管理資訊庫（MIBs）。輸出性能數據需要這個包。
net-snmp-utils	SNMP 客戶端例如 snmpget 和 snmpwalk。需要該軟體包以查詢 SNMP 系統的性能數據。
net-snmp-perl	mib2c 程式和 NetSNMP Perl 模組。此包是由可選通道提供的。
net-snmp-python	Python 的 SNMP 客戶端庫。此包是由可選通道提供的。

安裝任何一個軟體包按如下方式使用 dnf 命令：

```
dnf install package...
```

例如，安裝在本節的其餘部分中使用的 SNMP 代理守護進程和 SNMP 客戶端，以 root 身份在 shell 鍵入如下命令：

```
#dnf install net-snmp net-snmp-libs net-snmp-utils
```

#### 7.1.6.2. 運行 Net-SNMP 守護進程

net-snmp 軟體包中包含 SNMP 代理守護進程 snmpd。本節提供有關如何啟動，停止和重新啟動 snmpd 服務的資訊。有關在銀河麒麟高級伺服器操作系統 V10 系統的管理服務的詳細資訊請參閱 5.1 使用 systemd 管理系統服務。

## 啟動服務

運行 snmpd 服務以 root 身份在 shell 鍵入如下命令：

```
#systemctl start snmpd.service
```

配置服務使其在系統啟動後自動開始運行使用如下命令：

```
#systemctl enable snmpd.service
```

## 停止服務

停止 snmpd 服務以 root 身份在 shell 鍵入如下命令：

```
#systemctl stop snmpd.service
```

配置服務使其在系統啟動後並不開始運行，使用如下命令：

```
#systemctl disable snmpd.service
```

## 重啟服務

重啟 snmpd 服務以 root 身份在 shell 鍵入如下命令：

```
#systemctl restart snmpd.service
```

該命令停止服務並快速重啟服務。要僅重新加載配置，而無需停止服務，運行以下命令：

```
#systemctl reload snmpd.service
```

這會運行 snmp 服務去重新加載配置。

### 7.1.6.3. 配置 Net-SNMP

要修改 Net-SNMP 代理守護進程的配置，編輯/etc/snmp/snmpd.conf 配置檔。銀河麒麟高級伺服器操作系統 V10 包含的默認的 snmpd.conf 檔被大量

注釋，並可以作為一個很好的代理配置檔的起點。

本節主要介紹兩種常見的任務：設置系統資訊並配置認證。有關可用的配置指令的詳細資訊，請參閱 `snmpd.conf(5)` 手冊頁。此外，在 `net-snmp` 軟體包裏有一個名為 `snmpd.conf` 的實用程式，可以用來以交互方式生成代理配置。

需要注意的是 `net-snmp-util` 軟體包必須被安裝才能使用本節所述的 `snmpwalk` 實用程式。

```
#systemctl reload snmpd.service
```

設置系統資訊

`Net-SNMP` 通過系統樹提供了一些基本的系統資訊。例如，下麵的 `snmpwalk` 的命令顯示具有默認代理配置系統樹。

```
#snmpwalk -v2c -c public localhost system
```

缺省情況下，`sysName` 對象被設置為主機名。`sysLocation` 和 `sysContact` 對象可以在 `/etc/snmp/snmpd.conf` 檔通過改變 `syslocation` 和 `syscontact` 的值被配置。例如：

```
syslocation Datacenter, Row 4, Rack 3
syscontact UNIX Admin <admin@ example.com>
```

更改配置檔後，重新加載配置，並通過再次運行 `snmpwalk` 命令測試以下配置是否生效。

```
#systemctl reload snmp.service
#snmpwalk -v2c -c public localhost system
```

配置許可權

`Net-SNMP` 代理守護程式支持所有三個版本的 `SNMP` 協議。前兩個版本（1

和 2c) 通過使用字串提供簡單的身份驗證。該字串是代理人 and 任何客戶端實用程式之間共用的秘密。該字串在網路上用明文傳遞，這認為是不安全的。SNMP 協議第 3 版支持使用多種協議的用戶認證和資訊加密。Net-SNMP 代理還支持 SSH 通道，使用 X.509 證書的 TLS 認證和 Kerberos 認證。

### 配置 SNMP 版本 2c

要配置 SNMP 版本 2c 的社區，在 `/etc/snmp/snmpd.conf` 配置檔下使用 RO 社區或 RW 社區指令。指令的格式是如下：

```
directive community [source [OID]]
```

其中，community 要使用社區字串，source 是 IP 地址或子網，OID 對 SNMP 樹提供訪問屬性。例如，下麵的指令對客戶端提供只讀的系統樹訪問，該客戶端使用本地電腦上的社區字串“kylin”。

```
rocommunity kylin 127.0.0.1 .1.3.6.1.2.1.1
```

使用 `snmpwalk` 命令並添加 `-v` 和 `-c` 參數可以測試配置。

```
#snmpwalk -v2c -c kylin localhost system
```

### 配置 SNMP 版本 3

要配置 SNMP 版本 3，使用 `net-snmp-create-v3-user` 命令。此命令添加條目到 `/var/lib/net-snmp/snmpd.conf` 和 `/etc/snmp/snmpd.conf` 檔，這兩個檔可以創建用戶並授權訪問。注意，`net-snmp-create-v3-user` 命令只有代理沒有運行時才能運行。下麵的示例創建了“admin”用戶，密碼“kylinsnp”：

```
#systemctl stop snmpd.service  
#net-snmp-create-v3-user
```

`net-snmp-create-v3-user` 命令添加 `rwuser` 指令（或當提供的 `-ro` 命令行選項使用 `rouser`）到 `/etc/snmp/snmpd.conf` 中和添加 `rwcommunity` 和 `rocommunity` 的格式相同：

```
directive user [no auth|auth|priv] [OID]
```

其中，`user` 是用戶名，`OID` 是 SNMP 樹提供訪問。默認情況下，Net-SNMP 代理僅允許已通過授權的請求（`auth` 選項）。`noauth` 選項可以允許未經授權的請求，`priv` 選項強制使用加密，`authpriv` 選項指定的請求必須通過授權和回復被加密。

例如，下麵一行表示授予用戶“`admin`”讀寫訪問整個樹：

```
rwuser admin authpriv .1
```

為了測試配置，在您的用戶目錄下創建一個 `snmp` 目錄，在該目錄下創建 `snmp.conf` 檔（`~/snmp/snmp.conf`），使用如下命令：

```
defVersion 3
defSecurityLevel authPriv
defSecurityName admin
defPassphrase kylin snmp
```

`snmpwalk` 命令在查詢代理時會使用這些授權設置。

```
#snmpwalk -v3 localhost system
SNMPv2-MIB::sysDescr.0 = STRING: Linux
localhost.localdomain 3.10.0-
123.el7.x86_64 #1 SMP Mon May 5 11:16:57 EDT 2014 x86_64
[output truncated]
```

#### 7.1.6.4. 檢索數據性能

銀河麒麟高級服务器操作系統的 Net-SNMP 代理在 SNMP 協議上提供了多



種性能資訊。此外，可以通過代理查詢系統中安裝 RPM 包的列表，系統當前運行的進程列表和系統的網路配置。

本節提供了關於 OIDs 在 SNMP 上的性能概述。它假定系統已安裝 net-snmp-utils 軟體包，並且用戶被授予訪問 SNMP 樹，如 7.1.7.3 配置許可權中描述的。

#### 硬體配置

包括 Net-SNMP 的 Host Resources MIB 提供了主機客戶端程式的硬體和軟體配置。表 7-3 可用的 OIDs 總結了在 MIB 下可以獲取的不同的 OID。

**表 7-3 可用的 OIDs**

OID	描述
HOST-RESOURCES-MIB::hrSystem	包含一般系統資訊，如運行時間，用戶的數目，和運行的進程的數目。
HOST-RESOURCES-MIB::hrStorage	包含記憶體和文件系統使用情況數據。
HOST-RESOURCES-MIB::hrDevices	包含所有的處理器，網路設備和文件系統的列表。
HOST-RESOURCES-MIB::hrSWRun	包含所有正在運行的進程的列表。
HOST-RESOURCES-MIB::hrSWRunPerf	包含進程表中的記憶體和 CPU 統計資訊，進程表來自於 HOST-RESOURCES-MIB::hrSWRun。

HOST-RESOURCES-MIB::hrSWInstalled	包含 RPM 資料庫的列表。
-----------------------------------	----------------

在 Host Resources MIB 中,還有一些可用於檢索的可用資訊的 SNMP 表。

下麵的例子列出了 HOST-RESOURCES-MIB::hrFSTable:

```
~]$ snmptable -Cb localhost HOST-RESOURCES-MIB::hrFSTable
SNMP table: HOST-RESOURCES-MIB::hrFSTable

Index MountPoint RemoteMountPoint                               Type
Access Bootable StorageIndex LastFullBackupDate LastPartialBackupDate
1      "/"                               "" HOST-RESOURCES-TYPES::hrFSLinuxExt2
readWrite true                 31  0-1-1,0:0:0.0      0-1-1,0:0:0.0
5 "/dev/shm"                               "" HOST-RESOURCES-TYPES::hrFSOther
readWrite false                35  0-1-1,0:0:0.0      0-1-1,0:0:0.0
6      "/boot"                             "" HOST-RESOURCES-TYPES::hrFSLinuxExt2
readWrite false                36  0-1-1,0:0:0.0      0-1-1,0:0:0.0
```

關於 HOST-RESOURCES-MIB 的更多資訊,可以查看 `/usr/share/snmp/mibs/HOST-RESOURCES-MIB.txt` 檔。

CPU 和記憶體資訊

UCD-SNMP-MIB 的大多數系統性能數據是可用的。systemStats OID 提供了一些處理器的使用計數器:

```
#snmpwalk localhost UCD-SNMP -MIB::systemStats
```

特別是, `ssCpuRawUser`,`ssCpuRawSystem`,`ssCpuRawWait` 和 `ssCpuRawIdle` OIDs 提供的計數器在確定系統是否要花費了大量的處理器時間在內核空間或用戶空間或 I/O 時非常有用。`ssRawSwapIn` 和 `ssRawSwapOut` 可確定系統記憶體是否耗盡時非常有用。

UCD-SNMP-MIB::memory OID 下很多可用的記憶體資訊和 `free` 命令類似:

```
#snmpwalk localhost UCD-SNMP-MIB::memory
```

UCD-SNMP-MIB 的負載均衡是可用的。SNMP 的

UCD-SNMP-MIB::laTable 會列出 1 分鐘，5 分鐘和 15 分鐘的負載均衡值。

```
~]$ snmptable localhost UCD-SNMP-MIB::laTable
SNMP table: UCD-SNMP-MIB::laTable

 laIndex laNames laLoad laConfig laLoadInt laLoadFloat laErrorFlag
 laErrMessage
 1 Load-1 0.00 12.00 0 0.000000 noError
 2 Load-5 0.00 12.00 0 0.000000 noError
 3 Load-15 0.00 12.00 0 0.000000 noError
```

檔系統和磁片資訊

Host Resources MIB 提供檔系統的大小和使用資訊。每個檔系統（以及各記憶體池）在 HOST-RESOURCES-MIB::hrStorageTable 表中有一個入口：

```
~]$ snmptable -Cb localhost HOST-RESOURCES-MIB::hrStorageTable
SNMP table: HOST-RESOURCES-MIB::hrStorageTable

 Index          Type          Descr
 AllocationUnits Size Used AllocationFailures
 1              HOST-RESOURCES-TYPES::hrStorageRam Physical memory
 1024 Bytes 1021588 388064 ?
 3              HOST-RESOURCES-TYPES::hrStorageVirtualMemory Virtual memory
 1024 Bytes 2045580 388064 ?
 6              HOST-RESOURCES-TYPES::hrStorageOther Memory buffers
 1024 Bytes 1021588 31048 ?
 7              HOST-RESOURCES-TYPES::hrStorageOther Cached memory
 1024 Bytes 216604 216604 ?
 10             HOST-RESOURCES-TYPES::hrStorageVirtualMemory Swap space
 1024 Bytes 1023992 0 ?
 31             HOST-RESOURCES-TYPES::hrStorageFixedDisk /
 4096 Bytes 2277614 250391 ?
 35             HOST-RESOURCES-TYPES::hrStorageFixedDisk /dev/shm
 4096 Bytes 127698 0 ?
 36             HOST-RESOURCES-TYPES::hrStorageFixedDisk /boot
 1024 Bytes 198337 26694 ?
```

HOST-RESOURCES-MIB::hrStorageSize 和

HOST-RESOURCESMIB::hrStorageUsed 中的 ODI 被用來計算掛載檔系統的剩餘容量。

I/O 數據在 UCD-SNMP-MIB::systemStats 和 UCD-DISKIO-MIB::diskIOTable 表中都是可用的。後者提供了更詳細的數據，後者的 OID 有 diskIONReadX 和 diskIONWrittenX 其提供計數器，用於在系統啟動時記錄讀取和寫入塊設備的位元組數。

```

~]$ snmptable -Cb localhost UCD-DISKIO-MIB::diskIOTable
SNMP table: UCD-DISKIO-MIB::diskIOTable

  Index Device      NRead  NWritten Reads Writes LA1 LA5 LA15  NReadX
  NWrittenX
  ...
    25   sda 216886272 139109376 16409   4894   ?   ?   ? 216886272
139109376
    26   sda1 2455552    5120    613     2   ?   ?   ? 2455552
5120
    27   sda2 1486848      0     332     0   ?   ?   ? 1486848
0
    28   sda3 212321280 139104256 15312   4871   ?   ?   ? 212321280
139104256

```

## 網路資訊

Interfaces MIB 提供網路設備資訊。IF-MIB::ifTable 提供了一個 SNMP 表，系統上每個介面及其配置和各種數據包計數器在這個表裏都一個條目。下麵的例子顯示 ifTable 的部分列，它顯示了系統上的兩個物理網路介面：

```
#snmptable -Cb localhost IF-MIB::ifTable
```

網路流量是根據 IF-MIB::ifOutOctets 和 IF-MIB::ifInOctets 提供。下麵 SNMP 查詢將檢索此系統上的每一個介面的網路流量。

```
#snmpwalk localhost IF-MIB::ifDescr
```

## 擴展的 Net-SNMP

Net-SNMP 代理可以擴展到提供除原生系統度量的應用度量。這使得容量規劃和性能問題的故障排除。例如，在測試中知道郵件系統每 15 分鐘有 5 分鐘處在負載均衡狀態是有幫助的，但更有用的是知道郵件系統在每秒處理 8000 個消息時它 15 分鐘負載均衡是多少。當應用程式負載的度量可通過相同的介面作為系統的度量，這也允許在不同的負載影響的情況下對系統性能可視化（例如，每增加 10,000 條資訊會增加平均負載線至 100,000）。

包括銀河麒麟高級伺服器操作系統 V10 的一些應用程式通過 SNMP 擴展

Net-SNMP 代理以提供應用度量。有幾種方法來擴展代理定制應用程式。本節介紹通過 shell 腳本和可選的 Perl 插件來擴展代理。假設系統中 net-snmp-utils 和 net-snmp-perl 包已經安裝，同時用戶被授權訪問 SNMP 樹。

### 使用 shell 腳本擴展 Net-SNMP

在 Net-SNMP 代理提供了一個擴展 MIB（NET-SNMP-EXTEND-MIB），可以用來查詢任意的 shell 腳本。要指定的 shell 腳本來運行，可以在 /etc/snmp/snmpd.conf 檔使用 extend 指令。一旦定義，代理將提供退出碼和在 SNMP 上命令的任何輸出。下麵的例子演示了這種機制，該腳本確定在進程表中 httpd 進程的數量。

#### 注意：

Net-SNMP 代理提供了一個內建的機制通過 proc 命令檢查進程表。更多的資訊可以查看 snmpd.conf(5)手冊頁。

下麵 shell 腳本的退出碼是在給定時間點的系統上運行的 httpd 進程數。

```
#!/bin/sh
NUMPIDS=`pgrep httpd | wc -l`
exit $NUMPIDS
```

為了使這個腳本在 SNMP 上可用，將腳本複製到系統路徑上的某個位置，設置可執行位，並添加 extend 指令到/etc/snmp/snmp.conf 檔。添加 extend 指令的格式如下：

```
extend name prog args
```

name 是標識 extend 的字串，prog 是要運行的程式，args 是傳入程式的參數。例如，如果上面的 shell 腳本複製到/usr/local/bin/check\_apache.sh，

下麵的指令會添加腳本到 SNMP 樹。

```
extend httpd_pids /bin/sh /usr/local/bin/check_apache.sh
```

腳本可以在 NET-SNMP-EXTEND-MIB::nsExtendObjects 中查詢到。

```
~]$ snmpwalk localhost NET-SNMP-EXTEND-MIB::nsExtendObjects
NET-SNMP-EXTEND-MIB::nsExtendNumEntries.0 = INTEGER: 1
NET-SNMP-EXTEND-MIB::nsExtendCommand."httpd_pids" = STRING: /bin/sh
NET-SNMP-EXTEND-MIB::nsExtendArgs."httpd_pids" = STRING:
/usr/local/bin/check_apache.sh
NET-SNMP-EXTEND-MIB::nsExtendInput."httpd_pids" = STRING:
NET-SNMP-EXTEND-MIB::nsExtendCacheTime."httpd_pids" = INTEGER: 5
NET-SNMP-EXTEND-MIB::nsExtendExecType."httpd_pids" = INTEGER: exec(1)
NET-SNMP-EXTEND-MIB::nsExtendRunType."httpd_pids" = INTEGER: run-on-
read(1)
NET-SNMP-EXTEND-MIB::nsExtendStorage."httpd_pids" = INTEGER:
permanent(4)
NET-SNMP-EXTEND-MIB::nsExtendStatus."httpd_pids" = INTEGER: active(1)
NET-SNMP-EXTEND-MIB::nsExtendOutput1Line."httpd_pids" = STRING:
NET-SNMP-EXTEND-MIB::nsExtendOutputFull."httpd_pids" = STRING:
NET-SNMP-EXTEND-MIB::nsExtendOutNumLines."httpd_pids" = INTEGER: 1
NET-SNMP-EXTEND-MIB::nsExtendResult."httpd_pids" = INTEGER: 8
NET-SNMP-EXTEND-MIB::nsExtendOutLine."httpd_pids".1 = STRING:
```

請注意，退出碼設置為整型（示例中退出碼為 8），其他任何輸出為字串類型。為了顯示整數的多重度量，**extend** 指令提供了不同的參數。例如，下麵的腳本可以被用於確定匹配任意字串的進程數，並且也將輸出一個文本字串來表示進程數：

```
#!/bin/sh
PATTERN=$1
NUMPIDS=`pgrep $PATTERN | wc -l`
echo "There are $NUMPIDS $PATTERN processes."
exit $NUMPIDS
```

當上面的腳本被拷貝到 /usr/local/bin/check\_proc.sh 下，執行下麵 /etc/snmp/snmpd.conf 檔中的指令會顯示出 httpd 和 snmpd 的進程數。

```
extend httpd_pids /bin/sh /usr/local/bin/check_proc.sh httpd
extend snmpd_pids /bin/sh /usr/local/bin/check_proc.sh snmpd
```

下麵的例子顯示 snmpwalk 命令對 nsExtendObjects OID 的輸出：

```
~]$ snmpwalk localhost NET-SNMP-EXTEND-MIB::nsExtendObjects
NET-SNMP-EXTEND-MIB::nsExtendNumEntries.0 = INTEGER: 2
NET-SNMP-EXTEND-MIB::nsExtendCommand."httpd_pids" = STRING: /bin/sh
NET-SNMP-EXTEND-MIB::nsExtendCommand."snmpd_pids" = STRING: /bin/sh
NET-SNMP-EXTEND-MIB::nsExtendArgs."httpd_pids" = STRING:
/usr/local/bin/check_proc.sh httpd
NET-SNMP-EXTEND-MIB::nsExtendArgs."snmpd_pids" = STRING:
/usr/local/bin/check_proc.sh snmpd
NET-SNMP-EXTEND-MIB::nsExtendInput."httpd_pids" = STRING:
NET-SNMP-EXTEND-MIB::nsExtendInput."snmpd_pids" = STRING:
...
NET-SNMP-EXTEND-MIB::nsExtendResult."httpd_pids" = INTEGER: 8
NET-SNMP-EXTEND-MIB::nsExtendResult."snmpd_pids" = INTEGER: 1
NET-SNMP-EXTEND-MIB::nsExtendOutLine."httpd_pids".1 = STRING: There are 8
httpd processes.
NET-SNMP-EXTEND-MIB::nsExtendOutLine."snmpd_pids".1 = STRING: There are 1
snmpd processes.
```

### 警告：

整數退出碼被限制在 0-255 的範圍。對於超過 256 的值，既可以使用腳本的標準輸出（被認為是字串）或擴展代理的不同方法。

## 7.2. 查看和管理日誌檔

日誌檔是包含系統資訊的檔，包括內核，服務及其上運行的應用程式。不同的日誌檔包含不同的資訊，例如，有一種默認系統日誌檔，一種只是用於安全資訊的日誌檔，還有用於後臺任務的日誌檔。

當試圖解決系統問題的時候，如試圖載入內核驅動程式或尋找未授權的登錄系統時，日誌檔是非常有用的。本章討論在哪里可以找到日誌檔，如何查看日誌檔，以及在日誌檔中查看什麼。

一些日誌檔被一個名為 `rsyslogd` 的守護進程控制。該 `rsyslogd` 守護進程是替代 `sysklogd` 的一種增強型進程，並提供了擴展的篩檢程式，消息加密保護，各種配置選項，輸入輸出模組，通過 TCP 或 UDP 協議進行傳輸。需要注意的是 `rsyslogd` 與 `sysklogd` 是相容的。

日誌檔還可以由 `journald` 守護進程進行管理，該守護進程是 `systemd` 的

組件。journald 守護進程捕獲系統日誌消息，內核日誌消息，初始 RAM 磁片和早期啟動消息和寫到標準輸出的消息，以及所有服務的標準錯誤輸出，把這些消息進行索引，並提供給用戶。本地日誌檔格式，它是一種結構化和索引二進位檔，改進了搜索，並提供更快的操作，而且它也存儲像時間戳或用戶 ID 的元數據資訊。由 journald 生成的日誌檔在默認情況下不是一直存在的，日誌檔只保存在內存或/run/log/journal/目錄下的環形緩衝區。記錄的數據量取決於可用記憶體，當您達到容量極限時，最早的記錄將被刪除。但是，設置可以被改變-請參閱 7.5.5 使能持續存儲。有關期刊詳細資訊，請參閱 7.5 使用日誌。

默認情況下，這兩個記錄工具並存在您的系統。journald 守護進程是解決問題的主要工具。它也提供了必要的用於創建結構化日誌消息的附加數據。通過 journald 獲取的數據被轉發到/run/systemd/journal/syslog 套接字，可以由 rsyslogd 進一步處理數據。然而，rsyslogd 默認通過 imjournal 輸入模組，從而避免了上述的套接字。您還可以使用 omjournal 模組從 rsyslogd 到 journald 在相反方向上傳數據。參見 7.2.7 Syslogd 服務和日誌的交互。集成使基於文本的日誌檔使用一致的格式方便用來維護，以確保可能的應用程式或配置依賴於 rsyslogd 的相容性。另外，您可以用結構一致化的格式維護 rsyslogd 日誌。（參見 7.3 Syslogd 日誌結構）

### 7.2.1. 日誌檔的位置

許多由 rsyslogd 維護的日誌檔在/etc/rsyslog.conf 配置檔裏記錄。大多數的日誌檔在/var/log/目錄下。一些應用程式例如 httpd 和 samba 在/var/log/目錄下有自己的日誌檔。



您可能會注意到在`/var/log` 目錄下有多個日誌檔其名字後的數字不同(例如, `cron-20100906`)。這些數字代表添加輪替日誌檔裏的時間戳。日誌檔被輪替, 所以其大小不會太大。Logrotate 包括了一個後臺任務, 它可以根據 `/etc/logrotate.conf` 配置來自動輪替日誌檔, 該配置檔在`/etc/logrotate.d/`目錄下。

### 7.2.2. Rsyslog 的基本配置

Rsyslog 的主要配置檔是`/etc/rsyslog.conf`。在這裏, 您可以指定全局指令, 模組和篩檢程式的規則和動作部分。此外, 您還可以以文本形式評論添加解釋(以`#`號開頭)。

#### 7.2.2.1. 篩檢程式

規則是篩檢程式的一部分, 它選擇系統日誌消息的一個子集, 以及行動的一部分, 它指定對選中的消息做什麼。要在您的`/etc/rsyslog.conf` 的配置檔定義規則, 在一行同時定義篩檢程式和動作, 用一個或多個空格或跳位字元分隔。

根據所選屬性, rsyslog 現在提供了不同的方式過濾系統日誌消息。可用的過濾方法, 可分為設置/基於優先順序的, 基於屬性的, 和基於正則運算式的篩檢程式。

#### 設置/基於優先順序的篩檢程式

最常用的和眾所周知的方式來過濾系統日誌消息是使用基於設置/基於優先順序的篩檢程式, 其過濾 `syslog` 消息基於兩個條件: 由點號分隔設置和優先順序。要創建一個選擇器, 請使用以下語法

```
FACILITY.PRIORITY
```

➤ **FACILITY** 指定一個特定的子系統來生成日誌消息。例如，郵件子系統處理所有與郵件相關的系統日誌消息。**FACILITY** 可以由以下關鍵字之一（或由數字代碼）來表示：

kern(0),user(1),mail(2),daemon(3),auth(4),syslog(5),lpr(6),news(7),uucp(8),cron(9),authpriv(10),ftp(11),和 local0 through local7 (16-23).

➤ **PRIORITY** 指定系統日誌消息的優先順序。**PRIORITY** 可以由以下關鍵字之一（或由數字代碼）來表示：

debug(7),info(6),notice(5),warning(4),err(3),crit(2),alert(1),和 emerg(0).

上述語法根據定義或更高優先順序選擇系統日誌消息。通過比較等號（=）兩邊的優先順序，您指定的優先順序的系統日誌消息將被選中，所有其他優先順序將被忽略。相反，優先關鍵字之前有一個（!），選擇除了該優先順序的所有系統日誌消息。

除上述指定的關鍵字，您也可以使用星號（\*）來定義所有設置或優先順序（這取決於您在哪里放置星號，逗號之前或之後）。沒有給出優先順序設備的優先順序指定其關鍵字為 **none**。設置和優先條件是不區分大小寫。

要定義多個設置和優先順序，用逗號（,）將它們分開。要在一行中定義多個選擇，用分號（;）分隔它們。注意，在選擇器在其字段能夠覆蓋前面的部分，它可以排除來自模式一些優先次序。

### **實例：設置/基於優先順序的篩檢程式**

以下是可在 `/etc/rsyslog.conf` 中指定設置/基於優先順序的篩檢程式的幾個簡單例子。要選擇所有優先順序的所有內核系統日誌消息，添加下麵的文字到

配置檔：

```
kern.*
```

選擇所有的郵件系統日誌消息，使用優先順序為 **crit** 或更高，使用如下形式：

```
mail.crit
```

選擇所有的 **cron** 日誌消息除了優先順序為 **info** 和 **debug** 的。使用如下格式：

```
cron.!info,!debug
```

### 基於屬性的篩檢程式

基於屬性的篩檢程式可讓您通過任何屬性過濾系統日誌消息，如 **timegenerated** 或 **syslogtag**。您可以將每個指定屬性和值比較，這些值和說明參見表 7-4 基於屬性的比較操作。屬性和比較操作都是區分大小寫的。

基於屬性的篩檢程式以冒號（:）開始，定義該篩檢程式，使用如下語法：

```
:PROPERTY,[!]COMPARE_OPERATION,"STRING"
```

- **PROPERTY** 定義需要過濾的屬性；
- 可選的感嘆號（!）反向輸出比較操作。基於屬性的篩檢程式目前不支持其他布爾運算符；

- **COMPARE\_OPERATION** 屬性定義了比較操作，參考表 7-4 基於屬性的比較操作基於屬性的比較操作；

- 字串屬性指定比較值，其由屬性的文字字串提供。該值必須用引號括起來。為了使用某些字元的字串中（例如一個引號（"）），用反斜杠字元（\）。

表 7-4 基於屬性的比較操作

Compare-operation	description
Contains	檢查提供的字串是否有任何部分和所提供的文本屬性字串相匹配。要執行大小寫敏感的比較，使用 <code>contains_i</code> 。
isequal	比較提供的字串和文本提供的屬性字串。這兩個值必須匹配。
startswith	檢查提供字串和文本的屬性字串開頭的匹配。要執行不區分大小寫的比較，使用 <code>startswith_i</code> 。
regex	比較提供的 POSIX BRE（基本正則運算式）和文本所提供的屬性字串。
ereregex	比較提供的 POSIX ERE（擴展正則運算式）和文本所提供的屬性字串。
isempty	檢查該屬性是否為空。值被丟棄。在使用歸一化的數據時，某一些字段可能被填充，這個操作則特別有用。

**實例：基於屬性的篩檢程式**

以下是可在 `/etc/rsyslog.conf` 指定基於屬性的篩檢程式的幾個例子。在消息文本裏要選擇包含 `error` 字串的日誌，如下所示：

```
:msg, contains, "error"
```

下麵的篩檢程式從主機名為 `host1` 條件選擇日誌消息。

```
:hostname,isequal,"host1"
```

選擇的日誌消息不包含 `fatal` 和 `error` 已經其之間的文本。

```
:msg,!regex,"fatal .*error"
```

### 基於運算式的篩檢程式

基於運算式篩檢程式根據定義的算術，布爾或字串操作選擇系統日誌消息。

基於運算式篩檢程式使用 `rsyslog` 自己的腳本語言調用 `RainerScript` 腳本，可以構建複雜的篩檢程式。

基本的基於運算式的篩檢程式使用如下：

```
if EXPRESSION then ACTION else ACTION
```

➤ **EXPRESSION** 屬性表示要計算的運算式。例如：下麵的運算式

```
$msg startswith 'DEVNAME' or $syslogfacility-text == ' local0'
```

您

可以定義多個運算式，使用 `and` 或者 `or` 操作符；

➤ **ACTION** 屬性代表如果運算式返回為真要執行的動作。這可以是一個單一的動作，或包含在大括弧的任意複雜的腳本；

➤ 在一個新行的開始，基於運算式篩檢程式由關鍵字 `if` 指示。關鍵字 `then` 分開 **EXPRESSION** 和 **ACTION**。或者，也可以用關鍵字 `else` 來指定哪些動作是如果條件得不到滿足將要執行。

基於運算式的篩檢程式，可以使用大括弧嵌套使用條件就像例如：基於運算式的篩檢程式。該腳本可以讓您在運算式中使用設備/基於優先順序的篩檢程式。另一方面，基於屬性的篩檢程式，不建議在這裏。`RainerScript` 支持特定函數 `re_match()` 和 `re_extract()` 的正則運算式。

### 實例：基於運算式的篩檢程式

下麵的運算式包含兩個嵌套條件。`prog1` 程式創建的日誌檔被分成基於消

息中的“test”字串的兩個檔。

```
if $programname == 'prog1' then {
  action(type="omfile" file="/var/log/prog1.log")
  if $msg contains 'test' then
    action(type="omfile" file="/var/log/prog1test.log")
  else
    action(type="omfile" file="/var/log/prog1notest.log")
}
```

更多的關於基於運算式的篩檢程式參考線上文檔。RainerScript 腳本是 rsyslog 新配置格式的基礎，請參考 7.2.3 使用新的配置格式。

#### 7.2.2.2. 行為

行為指定對已經過濾的消息做什麼。下麵是一些可以在您的規則中定義的操作：

##### 存儲 **syslog** 消息到日誌檔

主要行為指明 **syslog** 消息被保存哪個日誌檔。這是由您指定檔路徑完成。

**FILTER PATH**

**FILTER** 代表著用戶選擇的目標檔的路徑。例如，下麵的規則會選擇所有的 **corn syslog** 消息然後將其存儲到 **/var/log/cron** 檔中。

**cron.\* /var/log/cron**

默認情況下，日誌檔每次生成一個系統日誌消息的時間同步。使用破折號標記 (-) 作為檔路徑的首碼指定省略同步：

**FILTER -PATH**

請注意，您可能會失去資訊，如果系統終止發生在寫之前。但是，這種設置可以提高性能，特別是如果您運行的程式會產生非常詳細的日誌資訊。

您指定的檔路徑可以是靜態或動態的。靜態檔由一個固定檔路徑，正如上所

示的例子中表示的。動態檔路徑根據所接收的消息是不同的。動態檔路徑是由一個問號（?）的首碼代表表示：

```
FILTER ?DynamicFile
```

`DynamicFile` 是一個名稱。相當於預定義的範本其用於修改輸出路徑。您可以使用破折號做首碼（-）來禁用同步，您也可以使用一個冒號分隔的多個範本（;）。

當您使用的是 X Window 系統的時候，如果您指定的檔是存在的終端或 `/dev/console` 設備，系統日誌消息發送到標準輸出（使用特殊的終端處理）或控制臺（使用特殊的 `/dev/console` 的-處理）。

### 通過網路發生syslog消息

`Rsyslog` 允許您可以通過網路發送和接收系統日誌消息。此功能允許您在一臺機器管理多個主機的系统日誌消息。要轉發 `syslog` 消息到遠程電腦，請使用以下語法：

```
@[(zNUMBER)]HOST:[PORT]
```

- “@”符號表示 `syslog` 消息被轉發到使用 UDP 協議的主機；
- “@@”符號表示 `syslog` 消息被轉發到使用 TCP 協議的主機；
- 可選項 `zNUMBER` 設置對系統日誌消息的 `zlib` 壓縮。該 `NUMBER` 屬性指定的壓縮級別（從 1-最低到 9-最大）。壓縮增益自動由 `rsyslogd` 檢查，如果有任何壓縮增益消息會被壓縮，如果消息小於 60 位元組則不會被壓縮；
- `HOST` 屬性定義哪個主機選擇 `syslog` 消息；
- `PORT` 屬性定義主機端口；

- 當主機定義 IPV6 地址，用方括號括地址（[]）。

### 實例：通過網路發送 **syslog** 消息

以下是該通過網路（請注意，所有操作都前面帶有一個選擇器，其選擇任何優先順序的所有消息）轉發系統日誌資訊的操作的一些例子。要通過 UDP 協議，將消息轉發到 192.168.0.1。

```
*.* @192.168.0.1
```

使用 6514 端口和 TCP 協議將消息轉發到 example.com：

```
*.* @@example.com:6514
```

下麵壓縮資訊以 zlib（9 級壓縮），並將其轉發給[2001:db8::1]，傳輸使用 UDP 協議。

```
*.* @(z9)[2001:db8::1]
```

### 輸出通道

輸出通道主要用於指定日誌檔可以增長到的最大大小。這對日誌檔輪替非常有用的（有關詳細資訊，請參見 7.2.2.4 日誌輪替）。輸出通道基本上是關於輸出操作的資訊的集合。輸出通道由 `$outchannel` 指令定義。要定義 `/etc/rsyslog.conf` 輸出通道，請使用以下語法：

```
$outchannel NAME,FILE_NAME,MAX_SIZE,ACTION
```

- `NAME` 屬性指定了輸出通道的名稱；
- `FILE_NAME` 屬性指定了輸出檔案名。輸出通道只能寫入到檔中，沒有管道，終端，或其他類型的輸出；
- `MAX_SIZE` 屬性指定了檔可以增長的最大大小，以位元組為單位；



- **ACTION** 屬性定義當檔到最大大小的行為；
- 要使用定義的通道作為規則裏的一個行為，如下：

```
FILTER:omfile:$NAME
```

### 實例：輸出通道日誌輪替

下面通過使用一個輸出通道顯示了一個簡單的日誌輪替。輸出通道經由 `$outchannel` 指令定義：

```
$outchannel log_rotation,/var/log/test_log.log,104857600,  
/home/joe/log_rotation_script
```

使用規則選擇所有優先順序的所有 **syslog** 消息，獲取消息後執行預先定義的輸出通道。

```
*.* :omfile:$log_rotation
```

一旦限制（在本例中 100MB）被達到，`/home/joe/log_rotation_script` 會被執行。該腳本可以包含任何從檔移動到其他檔夾內容，編輯具體的內容，或者乾脆刪除它。

### 發送 **syslog** 消息給特定的用戶

**rsyslog** 通過指定用戶名將 **syslog** 消息發送到指定的用戶（如本章實例：指定多行為）。要指定多個用戶，用逗號（,）分隔每個用戶名。將消息發送給當前登錄的所有用戶，使用星號（\*）。

### 執行一個程式

**rsyslog** 可以為選定的系統日誌消息執行程式，並使用 `system()` 調用在 `shell` 中執行的程式。要指定一個程式來執行，用（^）字元首碼該命令。因此，

指定一個範本其接收的消息的格式，並將其傳遞到指定的可執行檔作為一個行參數。

```
FILTER ^EXECUTABLE; TEMPLATE
```

這裏篩檢程式狀態的輸出通過由 **EXECUTABLE** 指定的程式進行處理。這個程式可以是任何有效的可執行檔。用格式範本的名稱替換 **TEMPLATE**。

### 實例：執行程式

在下面的例子中，被選擇的任何優先順序的任何系統日誌消息，其格式經過與 **template** 範本匹配並作為參數傳遞給 **test-program** 程式。

```
*.* ^test-program;template
```

### 警告：

從任何主機接收消息時以及使用 **shell** 執行操作，可能會受到命令注入。攻擊者可以嘗試將命令注入到您指定的行為要執行的程式。為了避免任何可能的安全威脅，充分考慮使用的 **shell** 執行行為。

### 存儲 **syslog** 消息到資料庫

通過使用資料庫寫操作，被選擇的系統日誌消息可以直接寫入到資料庫表。資料庫寫入使用的語法如下：

```
:PLUGIN:DB_HOST,DB_NAME,DB_USER,DB_PASSWORD;[TEMPLATE]
```

➤ **PLUGIN** 調用特定的插件，這些插件可以處理資料庫寫操作；（例如，**ommysql** 插件）。

➤ **DB\_HOST** 屬性指定資料庫的主機名；

➤ **DB\_NAME** 屬性指定資料庫名；

- DB\_USER 屬性指定資料庫用戶；
- DB\_PASSWORD 屬性指定資料庫用戶的密碼；
- TEMPLATE 屬性指定一個修改 syslog 資訊的範本。

### 重要：

目前，rsyslog 只支持 MySQL 和 PostgreSQL 資料庫。為了使用 MySQL 和 PostgreSQL 資料庫寫入功能，安裝 rsyslog-mysql 和 rsyslog-pgsql 包。

此外，請確保您在您的/etc/rsyslog.conf 配置檔加載相應的模組：

```
$ModLoad ommysql #Output module for MySQL support
$ModLoad ompgsql #Output module for PostgreSQL
support
```

更多關於 rsyslog 的資訊參考 7.2.6 使用 Rsyslog 模組。

另外，您也可以使用由 omlibdb 模組提供通用的資料庫介面（支持：Firebird/Interbase,MS SQL,Sybase,SQLite,Ingres,Oracle,mSQL）。

### 丟棄 syslog 消息

想丟棄選擇好的消息使用波形符（~）。

```
FILTER ~
```

丟棄行為主要是用來在進行任何進一步的處理之前，以篩選出的消息。如果您想省略一些重複的消息，這是非常有效的，否則重複消息將填充日誌檔。丟棄操作的結果取決於所在的配置檔中的指定配置的位置，為得到最好的結果把這些行為放到行為列表的前面。請注意，一旦消息被丟棄在後面的配置檔中的行則沒有辦法來檢索它。

例如，下麵的規則丟棄了任何 cron 的 syslog 消息。

```
cron.* ~
```

### 定義多行為

對於每一個選擇，您被允許指定多個行為。對一個選擇要指定多個操作，每一個行為寫在單獨一行，並用符號它前面（&）字元：

```
FILTER ACTION  
& ACTION  
& ACTION
```

指定多個操作提高了所希望結果的整體性能，因此選擇器需要被重新評估。

#### 7.2.2.3. 全局指令

全局指令是適用於 `rsyslogd` 守護進程的配置選項。他們通常會指定特定預定義變數的值，該值會影響 `rsyslogd` 守護進程的行為或遵循的規則。所有全局指令必須以美元符號（\$）。只有一個指令是每行指定。下麵是一個全局指令的例子，其指定系統日誌消息佇列的最大大小：

```
$MainMsgQueueSize 50000
```

這個指令默認大小是 50000 個消息，可以修改數字重新定義。

您可以在 `/etc/rsyslog.conf` 配置檔中定義多個指令。一個指令影響的所有配置選項的行為，直到同一個指令的另一事件被檢測到。全局指令可以用來配置行為、佇列和調試。所有可用的配置指令的完整列表可以參考線上文檔。目前，一種新的配置格式已經開發出其可以替代 \$ 基礎的語法（7.2.3 使用新的配置格式）。然而，經典的全局指令仍然為舊格式提供支持。

#### 7.2.2.4. 日誌輪替

下麵是一個簡單的/etc/logrotate.conf 配置檔例子：

```
#rotate log files weekly
weekly
#keep 4 weeks worth of backlogs
rotate 4
#uncomment this if you want your log files compressed
compress
```

示例配置檔中所有的行的定義全局選項，其適用於每一個日誌檔。在我們的例子，日誌檔每週輪替，輪替的日誌檔保留四個星期，所有的輪替日誌檔由 gzip 壓縮的成.gz 的格式。以#號開頭的任何行是注釋，不會被處理。

您可以為特定的日誌檔定義配置選項，並將其放置在全局選項的下麵。然而，最好是在/etc/logrotate.d/directory 目錄下為特定的日誌檔創建單獨的配置檔，並在配置檔裏配置選項。

下麵是一個在/etc/logrotate.d/目錄下配置檔的例子：

```
/var/log/messages {
rotate 5
weekly
postrotate
/usr/bin/killall -HUP syslogd
endscript
}
```

該檔中的配置選項只為/var/log/messages 日誌檔配置。在可能的情況下，在此指定的設置將覆蓋全局設置。因此，輪替的/var/log/messages 日誌檔將保留五周，而不是全局選項定義的四周。

下麵是一些指令的列表，您可以在您的日誌輪替配置檔裏修改。

- > **weekly**-指定日誌輪替的週期為每週，類似的命令還包括如下：
  - daily**
  - monthly**
  - yearly**
- > **compress**-使能輪替檔的壓縮功能，類似的命令還包括如下：
  - nocompress**
  - compresscmd** -指定用於壓縮的命令。
  - uncompresscmd**
  - compressext** -指定用於壓縮的擴展。
  - compressoptions** -指定用於傳給程式的壓縮選項。
  - delaycompress** -退出壓縮直到下一個輪替週期。
- > **rotate INTEGER** -指定了一個日誌檔被輪替的最大數目，超過數目的日誌檔被刪除或郵寄到一個特定的地址。如果指定值 **0**，舊的日誌檔被刪除，而不會輪替。
- > **mail ADDRESS** -此選項，已輪替多次的日誌檔的郵件地址。類似的指令包括：
  - nomail**
  - maifirst**-指定只是輪替的日誌檔被郵寄，到期的日誌檔不被郵寄。
  - Maillast**-指定到期的日誌檔被郵寄，輪替的日誌檔不郵寄。當郵寄功能打開時，此選項是默認選項。

關於個配置選項的完整的指令列表參考 **logrotate(5)** 手冊頁。

### 7.2.3. 使用新的配置格式

銀河麒麟高級伺服器操作系統安裝的 **rsyslog** 包默認情況下是第 **8** 版，介

紹其新的配置語法。這種新的配置格式的目標是更強大，更直觀，通過不允許某些無效的結構防止常見的錯誤。語法增強的使能是通過 RainerScript 腳本配置處理器完成。遺留格式仍然得到支持，它默認在 `/etc/rsyslog.conf` 配置檔中使用。

RainerScript 是一種腳本語言，旨在用於處理網路事件和配置事件處理器，例如 rsyslog。RainerScript 最早是用來定義基於運算式的篩檢程式，見 7.2.2.1 篩檢程式中有關基於運算式篩檢程式的章節。RainerScript 在 rsyslog7 的版本中實現了 `input()` 和 `ruleset()`，其允許所述 `/etc/rsyslog.conf` 配置檔被寫入新的語法。新的語法不同之處主要在於，它是更結構化的；參數作為參數傳遞給語句，如輸入，行為，範本，模組加載。選項的範圍由塊限制。這增強可讀性，並降低所造成由錯誤配置的錯誤的數量。還有一個顯著的性能優勢。有些功能在新舊語法都可以用，有的只在新語法可以用。

與舊風格的參數配置比較：

```
$InputFileName /tmp/inputfile
$InputFileTag tag1:
$InputStateFile inputfile-state
$InputRunFileMonitor
```

同樣的配置使用新的配置語句如下：

```
input(type="imfile" file="/tmp/inputfile" tag="tag1:"
statefile="inputfile-state")
```

這顯著減少在配置中使用的參數的數目，提高了可讀性，並且還提供了更高的運行速度。有關 RainerScript 語句和參數的詳細資訊，請參見線上文檔。

### 7.2.3.1. 規則集

除特殊指令之外，`rsyslog` 處理的消息由規則定義。規則由過濾條件和如果條件為真要執行的操作組成。在 `/etc/rsyslog.conf` 檔中的舊規則，所有的規則以每個輸入消息的出場順序來評估。此過程開始於第一規則，並繼續，直到所有的規則都已經被處理或直到消息被其中某一規則丟棄。

然而，規則可被分成不同序列稱為規則集。在規則集中，您可以限制某些規則的影響，只選擇輸入或通過定義一組綁定到特定的輸入的行為提升 `rsyslog` 的性能。換句話說，某些類型的消息不可避免的被評估為假的，類似這樣的過濾條件可以被跳過。在 `/etc/rsyslog.conf` 中舊規則集定義如下所示：

```
$RuleSet rulesetname
rule
rule2
```

當另一個規則被定義，上一個規則就結束了，或者默認規則集被調用，如下：

```
$RuleSet RSYSLOG_DefaultRuleset
```

`rsyslog 8` 的新的配置格式，`input ( )` 和 `ruleset ( )` 語句執行被保留。新

格式的規則集在 `/etc/rsyslog.conf` 中定義，如下所示：

```
ruleset(name="rulesetname") {
rule
rule2
call rulesetname2
...
}
```

用您規則集的識別字替換 `rulesetname`。該規則集名稱不能以 `RSYSLOG_` 開始，因為這個名稱空間是保留的，供 `rsyslog` 使用。



`RSYSLOG_DefaultRuleset` 定義缺省設置的規則集，如果消息沒有分配其他規則集將被執行。在上面提到的 `rule` 和 `rule2` 下，您可以定義過濾-操作的格式規則。對於調用參數，您可以嵌套規則集由內部或者規則集塊調用它們。

創建規則集後，您需要指定它適用於什麼輸入：

```
input(type="input_type" port="port_num"
ruleset="rulesetname");
```

這裏可以通過 `input_type` 識別輸入消息，這是所收集的資訊，或者通過 `port_num` - 端口號。其他參數，如 `file` 或 `tag` 可以用於指定 `input()`。用規則集的名稱替換 `rulesetname`。萬一某個輸入消息未明確在規則集中指定，默認規則集會被觸發。

#### 7.2.3.2. Sysklogd 服務的相容性

`rsyslog` 第 5 版通過 `-c` 選項指定相容模式，但在第 8 版不支持，同樣，`Sysklogd` 風格的命令行選項已過時，應避免通過這些命令行選項配置 `rsyslog`。但是，您可以使用多個範本和指令配置 `rsyslogd` 來模仿像 `sysklogd` 似的行為。

更多關於不同的 `rsyslogd` 選項的資訊參考 `rsyslogd(8)` 手冊頁。

#### 7.2.4. 使用 **Rsyslog** 佇列

佇列用於在 `rsyslog` 的組件之間傳遞內容，傳遞的主要內容是系統日誌消息。在佇列機制下，`rsyslog` 能夠同時處理多個消息，並可以用多個行為對單個消息立刻反應。`rsyslog` 的數據流說明如下：

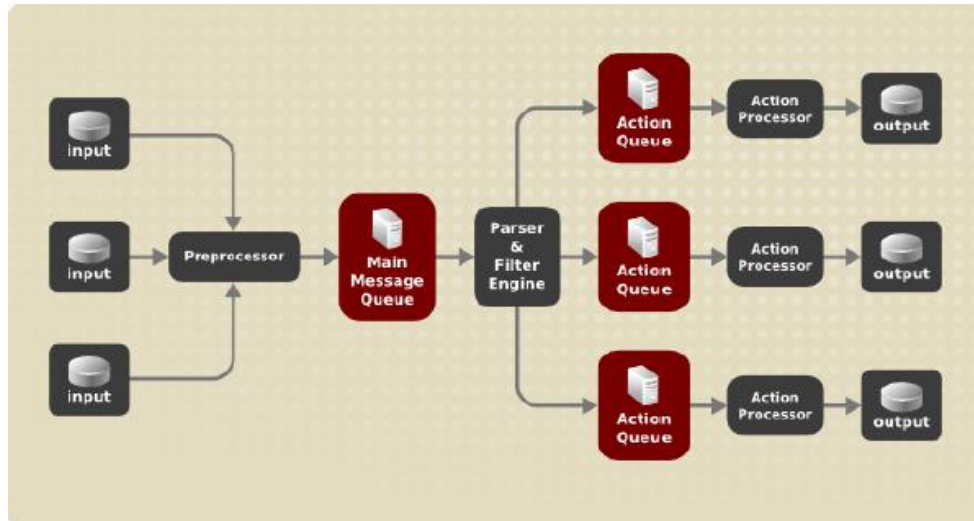


圖 7-4Rsyslog 中消息流

無論 rsyslog 什麼時候收到消息，它都把這個消息給預處理，然後將其放置到主消息佇列。消息等在那裏待出隊並傳遞到規則處理器。

規則處理器是一個解析和過濾引擎。在此，在 `/etc/rsyslog.conf` 定義的規則被應用。基於這些規則，該規則處理器評估哪些操作被執行。每個行為都有自己的執行佇列。消息通過佇列傳遞到行為處理器最終產生相應的輸出。注意，在這一點上，幾個行為可以同時對一個消息運行。為了這個目的，一個消息被複製並傳遞到多個行為處理器。

只有每個行為一個佇列是可能的。根據配置的不同，該消息可以正確發送到行為處理器而不通過執行佇列。這是直接佇列（見下文）的行為。在輸出操作失敗的情況下，行為處理器通知執行佇列，一段時間間隔後，行為處理器又會去取未處理的消息，再次嘗試。

綜上所述，rsyslog 佇列有兩個位置：無論是對與規則處理器作為一個主消息佇列的前面或在各種類型的輸出行為作為動作佇列的前面。佇列有兩個主要優點既都會提高消息處理性能：

- 它們充當緩衝器，在 `rsyslog` 的結構中分離生產者和消費者；
- 它們允許並行執行消息。

除此之外，佇列可以用幾種不同的指令被配置來為系統提供最佳的性能。這些配置選項在以下各節介紹。

#### 警告：

如果輸出插件無法傳送一個消息，它被存儲在前面的消息佇列。如果在佇列滿時，輸入阻塞，直到它不再滿。這將阻止新的消息進入被阻塞的消息佇列。在沒有單獨的執行佇列的情況下，這會產生嚴重的後果，例如阻止 `SSH` 日誌記錄，這反過來又阻止 `SSH` 訪問。因此，建議為那些通過網路轉發到資料庫的輸出使用專用的行為的佇列。

#### 7.2.4.1. 定義佇列

根據消息在哪里存儲，分為幾種類型的佇列：`direct`，`in-memory`，`disk` 和 `disk-assisted`，最廣泛使用的佇列是 `in-memory`。您可以選擇這些類型的其中之一作為主要消息佇列，也可以作為執行佇列。將下麵的語句添加到 `/etc/rsyslog.conf` 檔：

```
$objectQueueType queue_type
```

在這裏，您可以為主消息佇列（用 `MainMsg` 替換 `object`）或執行佇列（用 `action` 代替 `object`）申請設置。用 `direct`，`linkedlist` 或 `fixedarray`（`in-memory` 佇列），或 `disk` 更換 `queue_type`。

默認設置主消息佇列是 `FixedArray` 佇列，最大存儲 10000 個消息。執行佇列默認設置為 `Direct` 佇列。

## Direct 佇列

對於許多簡單的操作，例如，輸出寫入到本地檔，在執行前建立一個佇列是不需要的。為了避免排隊，使用如下：

```
$objectQueueType Direct
```

使用 `MainMsg` 或者 `Action` 替代 `object`，這個選項可以用在主消息佇列或者執行消息佇列。對於 `direct` 佇列，消息直接傳送，它會很快從生產者傳送到消費者。

## Disk 佇列

`disk` 佇列存儲消息到硬碟驅動器，這使得它們高度可靠的，但也是所有可能的排隊模式中最慢的。此模式可用於防止高度重要的日誌數據的丟失。但是，大多數用例中不建議使用 `disk` 佇列。要設置 `disk` 佇列，在 `/etc/rsyslog.conf` 中鍵入以下內容：

```
$objectQueueType Disk
```

使用 `MainMsg` 或者 `Action` 替代 `object`，這個選項可以用在主消息佇列或者執行消息佇列。`disk` 佇列寫入部分，默認大小為 `10MB`。此默認大小可以通過下麵的配置指令進行修改：

```
$objectQueueMaxFileSize size
```

其中，`size` 表示 `disk` 佇列的大小。定義的大小限制不是強制性的，`rsyslog` 總是寫入一個完整的佇列條目，即使違反的大小限制。`disk` 佇列的每個部分有單獨的檔相匹配。這些檔的命名指令如下所示：

```
$objectQueueFilename name
```

這為要設置的檔設置了檔案名首碼。其檔案名以 7 個數字開始，每增加一個檔數字也增加。

### **In-memory 佇列**

在 in-memory 佇列中，排隊的消息被保存在內存中，這使得這個過程非常快。如果電腦重新加電或關機，則排隊的數據將丟失。但是，您可以使用 `$ActionQueueSaveOnShutdown` 設置關機前保存數據。有兩種類型的 in-memory 佇列：

- `FixedArray queue`–默認的主要消息佇列，最大 10,000 個元素。

這種類型的佇列的使用固定預分配的數組保存佇列元素的指針。由於這些指針，即使佇列為空也會消耗一定量的記憶體。然而，`FixedArray` 提供了最佳的運行時性能，當您期望排隊的消息相對較少和高性能，此佇列是最佳的；

- `LinkedList queue`–這裏，一個鏈表裏所有結構都是動態分配的，因此，記憶體僅在需要時分配。`LinkedList` 的佇列處理偶爾的突發消息非常好。

在一般情況下，當使用 `LinkedList` 佇列相比於 `FixedArray` 佇列，它消耗更少的記憶體，並降低了處理開銷。

配置 in-memory 佇列使用下麵的語法：

```
$objectQueueType LinkedList  
$objectQueueType FixedArray
```

使用 `MainMsg` 或者 `Action` 替代 `object`，這個選項可以用在主消息佇列或者執行消息佇列。

## disk-assisted in-memory 佇列

disk 和 in-memory 佇列都有各自的優點,rsyslog 讓您可以將其合併為 disk-assisted in-memory 佇列。為此,配置一個正常的 in-memory 佇列然後添加 `$objectQueueFileName` 指令可定義為磁片援助的檔案名。這個佇列就變成 disk-assisted, 這意味著 in-memory 佇列與 disk 佇列通力協作。

如果 in-memory 佇列已滿或者需要關機後消息不丟的 disk 佇列被啟動。在 disk-assisted 佇列中,可以同時設置 disk-specific 或 in-memory specific 配置參數。這種類型的佇列可能是最常用的,它對可能長時間運行的和不可靠的操作特別有用。

指定 disk-assisted in-memory 佇列使用 so-called 浮水印:

```
$objectQueueHighWatermark number  
$objectQueueLowWatermark number
```

使用 `MainMsg` 或者 `Action` 替代 `object`,這個選項可以用在主消息佇列或者執行消息佇列。使用允許入隊的最大消息數替代 `number`。當 in-memory 佇列達到了高水位定義的數字,它開始將消息寫入磁片,並一直持續到 in-memory 佇列大小下降到與低浮水印定義的數量。正確設置浮水印儘量減少不必要的磁片寫操作,同時也留下了消息的記憶體空間,因為寫入磁片檔是相當慢的。因此,高水位必須比整個佇列容量設置 `$objectQueueSize` 低。高水位和整體佇列大小之間的差是專供消息突發而備用的記憶體緩衝區。在另一方面,設定高水位過低會不必要的頻繁的打開磁片援助。

### 7.2.4.2. 管理佇列

所有類型的佇列可以進一步配置以滿足您的要求。您可以使用幾種不同的指

令來修改執行佇列和主消息佇列。目前，有 20 多個佇列參數可用，見線上文檔。其中的一些設置是常見的，其他諸如工作線程管理，提供對佇列行為多的控制，保留給高級用戶使用。在高級設置中，您可以優化 `rsyslog` 的性能，調度排隊，或修改系統關閉佇列的行為。

### 限制佇列大小

您可以限制佇列可以包含消息的數量，使用如下設置：

```
$objectQueueHighWatermark number
```

使用 `MainMsg` 或者 `Action` 替代 `object`，這個選項可以用在主消息佇列或者執行消息佇列。用佇列可以包含的消息數目替代 `number`。可以設置佇列的大小，而不是作為它們實際記憶體的大小。主要消息佇列和規則集佇列默認佇列大小為 10000，執行佇列默認大小為 1000。

`Disk assisted` 佇列在默認情況下是無限制的，不能用指令來強制大小，但可以以位元組為單位保留他們的物理磁片空間，使用以下設置：

```
$objectQueueMaxDiscSpace number
```

使用 `MainMsg` 或者 `Action` 替代 `object`。當佇列被填滿時，新消息被丟棄，直到佇列釋放了足夠的空間。

### 丟棄消息

當佇列消息達到一定的數量，為了節省空間，您可以在佇列中丟棄不太重要的資訊，保留空間給優先順序更高的消息。啟動丟棄處理的閾值可設定 `discard mark`：

```
$objectQueueDiscardMark number
```

使用 `MainMsg` 或者 `Action` 替代 `object`, 這個選項可以用在主消息佇列或者執行消息佇列。這裏, `number` 表示要啟動丟棄消息進程的消息數目。定義哪些消息要被丟棄, 使用如下:

```
$objectQueueDiscardSeverity priority
```

用下麵的關鍵字之一 (或一些) 替代 `priority`: `debug` (7), `info` (6), `notice` (5), `warning` (4), `err` (3), `crit` (2), `alert` (1) 和 `emerg` (0)。使用此設置, 當達到丟棄數目後, 比定義優先順序較低的新來的和已排隊消息從佇列中刪除。

### 使用時限

可以配置 `rsyslog` 在一個特定的時間段處理佇列。有了這個選項, 您可以轉移部分工作到非高峰時段。要定義一個時間範圍, 請使用以下語法:

```
$objectQueueDequeueTimeBegin hour  
$objectQueueDequeueTimeEnd hour
```

您可以指定綁定時間的框架, 以小時為單位。使用 24 小時制, 不支持設置分。

### 配置工作線程

工作線程執行消息入隊的指定操作。例如, 在主消息佇列, 一個工作線程的任務將濾波器邏輯應用到每個輸入消息, 並將其排隊到相關執行佇列。當消息到達時, 一個工作線程將自動啟動。當消息的數量達到一定數量時, 另一個工作線程被啟動。要指定此數, 使用如下:

```
$objectQueueWorkerThreadMinimumMessages number
```

用消息的數量替代 `number` 將觸發補充工作線程。例如, `number` 置為 100,



當 100 多個消息到達時，一個新的工作線程開始。當超過 200 個消息到達時，第三個工作線程啟動等。然而，並行運行太多的工作線程是低效的，所以您可以限制它們的最大數量，如下所示：

```
$objectQueueWorkerThreads number
```

`number` 代表可並行運行的工作線程的最大數目。對於主消息佇列，默認限制為 1 個線程。一旦一個線程工作已經開始，它會一直運行直到超時出現。

要設置超時時長，如下所示：

```
$objectQueueWorkerTimeoutThreadShutdown time
```

用持續時間（以毫秒為單位）替換 `time`。如果沒有設置 `time`，零超時會被應用，當它沒有消息可以處理時，工作線程立即終止。如果您指定的時間為 -1，沒有線程將被關閉。

### 批量出佇列

為了提高性能，您可以配置 `rsyslog` 同時多佇列的消息數。設定出佇列消息數目的上限，則使用如下命令：

```
$objectQueueDequeueBatchSize number
```

用可被同時出佇列的消息的最大數目替換 `number`。請注意，出佇列消息數目較高的設置和很多的工作線程同時運行會導致較大的記憶體消耗高。

### 終止佇列

當要終止的佇列中仍然包含資訊，您可以嘗試通過指定時間間隔使工作線程來完成佇列處理，以減少數據丟失：

**\$objectQueueTimeoutShutdown time**

指定的 **time** 以毫秒為單位。如果一段時間後仍然有一些排隊的消息，工作線程完成當前的數據處理，然後終止佇列。因此，未處理的消息都將丟失。另一個時間間隔可以讓工作線程完成最後的數據處理：

**\$objectQueueTimeoutActionCompletion time**

在設置的 **time** 超時的情況下，任何工作線程都被關閉。在關機時要保存數據，可以使用：

**\$objectQueueTimeoutSaveOnShutdown time**

如果按上面那樣設置，所有佇列的數據在 **rsyslogd** 終止前都會被存儲到磁片。

#### 7.2.4.3. 使用 **rsyslog** 佇列新語法

從 **rsyslog 7** 開始 **rsyslog** 提供了新的語法，佇列對象在 **action ( )** 內部定義，既可以單獨使用或在 **/etc/rsyslog.conf** 中一個規則集使用。執行佇列的格式如下：

```
action(type="action_type" queue.size="queue_size"  
queue.type="queue_type" queue.filename="file_name")
```

用要執行的操作的模組名稱替換 **action\_type**，用消息佇列可以容納的最大數目替代的 **queue\_size**。對於 **queue\_type**，可以選擇 **disk** 佇列或從 **in-memory** 佇列中選擇一個：**direct**，**LinkedList** 的或 **fixedarray**。對於 **file\_name** 僅指定一個檔案名，而不是路徑。請注意，如果創建一個新的目錄來保存日誌檔，**SELinux** 必須被設置。

### 7.2.5. 在日誌伺服器上配置 rsyslog

rsyslog 服務提供的設施可用來運行日誌伺服器 and 配置單個系統來發送日誌檔到日誌伺服器。請參見實例“可靠的轉發日誌消息到伺服器”，查看客戶端 rsyslog 配置資訊。

rsyslog 服務必須安裝在您打算作為日誌伺服器的系統上，並且將所有的系統配置為將日誌發送給日誌伺服器。銀河麒麟高級伺服器操作系統 V10 默認情況下會安裝 rsyslog，如果需要，以確保系統確實安裝了 rsyslog，以 root 身份輸入以下命令：

```
#dnf install rsyslog
```

Syslog 流量默認的協議和端口是 UDP 和 514，其在/etc/services 檔中列出。然而，rsyslog 默認使用 TCP 在端口 514。在配置檔/etc/rsyslog.conf 中，TCP 是由@@聲明。

其他的端口在示例有時會用到，然而 SELinux 的默認配置為僅允許發送和接收在下面列出的端口中：

```
#semanage port -l | grep syslog
```

semanage 實用程式由 policycoreutils-python-utils 包的一部分提供。如果需要的話，安裝包如下：

```
#dnf install policycoreutils-python-utils
```

此外，在默認情況下 SELinux 的 rsyslog 類型是 rsyslogd\_t，其被配置為允許發送和接收類型是 rsh\_port\_t 的遠程 shell（RSH）端口，其 TCP 默認端口是 514。因此，這是沒有必要使用 semanage 的明確的允許 TCP 端口為 514。

例如，要檢查 SELinux 允許什麼程式在端口 514 通信，輸入命令如下：

```
#semanage port -l | grep 514
```

請在您打算做日誌伺服器的系統上，使用下列過程中的步驟。在這些過程中的所有步驟必須以 root 身份來執行命令。

#### 7.2.5.1. 在日誌伺服器上使用範本

rsyslog 具有多種不同的範本樣式。字串範本最接近舊格式。用字符串格式生成上面例子中的範本，如下所示：

```
template(name="TmplAuthpriv" type="string"
string="/var/log/remote/auth/%HOSTNAME%/PROGRAMNAME:::s
ecpathreplace%.log")
template(name="TmplMsg" type="string"
string="/var/log/remote/msg/%HOSTNAME%/PROGRAMNAME:::s
ecpathreplace%.log")
```

這些範本也可以寫成下麵這樣的格式列表：

```
template(name="TmplAuthpriv" type="list") {
constant(value="/var/log/remote/auth/")
property(name="hostname")
constant(value="/")
property(name="programname" SecurePath="replace")
constant(value=".log")
}
template(name="TmplMsg" type="list") {
constant(value="/var/log/remote/msg/")
property(name="hostname")
constant(value="/")
property(name="programname" SecurePath="replace")
constant(value=".log")
}
```

對這些 **rsyslog** 新規則來說，這個範本文本格式可能會更容易閱讀，因此可以更容易應用。

要完成新語法的更改，我們需要重新生成模組加載命令，添加規則集，然後綁定協議，端口和規則集的規則：

```
module(load="imtcp")
ruleset(name="remote1"){
  authpriv.* action(type="omfile" DynaFile="TmplAuthpriv")
  *.info;mail.none;authpriv.none;cron.none
action(type="omfile"
  DynaFile="TmplMsg")
}
input(type="imtcp" port="10514" ruleset="remote1")
```

#### 7.2.6. 使用 **Rsyslog** 模組

由於採用模組化設計，**rsyslog** 提供各種模組，這些模組提供額外的功能。需要注意的是模組可以由第三方開發。大多數模組提供額外的輸入（見下文輸入模組）或輸出（見下文輸出模組）。其他模組提供具體到每個模組的特定功能。加載一個模組後，其可提供可用的附加配置指令。要加載模組，請使用以下語法：

```
$ModLoad MODULE
```

其中 **\$ModLoad** 是加載指定的模組的全局指令。**MODULE** 表示您所需的模組。例如，如果您要加載的文本檔輸入模組（**imfile**），使 **rsyslog** 將任何標準的文本檔轉換成系統日誌資訊，在 **/etc/rsyslog.conf** 配置檔中指定以下行：

```
$ModLoad imfile
```

**rsyslog** 提供了許多模組，這些模組被分為以下類別：

- 輸入模組-輸入模組收集來自各種來源的消息。輸入模組的名稱總是

以 `im` 首碼開始，如 `imfile` 和 `imjournal`。

➤ 輸出模組-輸出模組提供便利給各種目標發出消息，如通過網路發送，存儲在資料庫中，或者加密等消息。輸出模組的名稱總是以 `om` 首碼，如 `omsnmp`，`omrelp` 等等。

➤ 解析模組-這些模組在創建自定義解析規則，或者解析異常的消息是有用的。使用 C 語言的中間層，您可以創建自己的消息解析器。解析器模組的名稱總是以 `pm` 首碼開始，如 `pmrfc5424`，`pmrfc3164`，等。

➤ 消息修改模組-消息修改模組修改系統日誌消息的內容。這些模組的名稱以 `mm` 的首碼開始。消息修改的模組如 `mmanon`，`mmnormalize`，或 `mmjsonparse` 其用於匿名或消息的正常化。

➤ 字串生成模組-串生成模組生成基於消息內容的字串，並與 `rsyslog` 提供的範本功能相互協同。串生成模組的名稱總是以 `sm` 首碼開始，如 `smfile` 或 `smtrad` 檔。

➤ 庫模組-庫模組為其他可加載模組提供功能。這些模組在 `rsyslog` 需要時自動被加載，不能由用戶配置。

所有可用的模組和它們的詳細描述的完整列表可以在 [http://www.rsyslog.com/doc/rsyslog\\_conf\\_modules.html](http://www.rsyslog.com/doc/rsyslog_conf_modules.html) 找到。

警告：

需要注意的是 `rsyslog` 加載的模組，這使他們可以訪問自己的函數和數據。這會帶來潛在的安全威脅。為了最大限度地減少安全風險，僅使用可信賴模組。

### 7.2.6.1. 導入 text 檔

文本檔輸入模組，簡稱 `imfile`，使 `rsyslog` 將任意文本檔轉化成系統日誌消息流。您可以使用 `imfile` 從創建自己的文本檔日誌的應用程式中導入日誌消息。

要加載 `imfile`，在 `/etc/rsyslog.conf` 添加以下內容：

```
$ModLoad imfile
$InputFilePollInterval int
```

它足以加載 `imfile` 一次，導入多個檔時也是如此。`$InputFilePollInterval` 全局指令指定 `rsyslog` 檢查被連接的文本檔變化的頻率。默認的時間間隔為 10 秒，要改變它，用以秒為單位的時間間隔替換 `int`。

為了確定文本檔導入，在 `/etc/rsyslog.conf` 檔中使用的語法如下：

```
#File 1
$InputFileName path_to_file
$InputFileTag tag:
$InputStateFile state_file_name
$InputFileSeverity severity
$InputFileFacility facility
$InputRunFileMonitor
#File 2
$InputFileName path_to_file2
...
```

需要 4 個設置來指定一個輸入檔：

- 用文本檔路徑替代 `path_to_file`
- 用消息的 `tag` 名替代 `tag`
- 用唯一的狀態檔案名替代 `state_file_name`。狀態檔都存儲在

`rsyslog` 的工作目錄，為監控的檔保持標記，標記已經被處理完的分區。如果刪除它們，整個檔將被重新讀入。請確保您指定了一個不存在的名稱。

➤ 添加 `$InputRunFileMonitor` 指令使能檔監控。沒有這個設置，文本檔會被忽略。

除了所要求的指令，存在可以應用到文本輸入的幾個其他設置。通過用適當的關鍵字替換關鍵字 `severity` 來設置導入消息的嚴重性。用關鍵字替換 `facility` 來定義生成消息的子系統。

#### 7.2.6.2. 從資料庫導出消息

在資料庫中，處理日誌數據比處理文本檔可以更快，更方便。基於所使用的 DBMS 的類型，選擇各種輸出模組，例如 `ommysql`，`ompgsql`，`omoracle`，或 `ommongodb`。作為替代，使用依賴於 `libdbi` 庫的通用 `omlibdbi` 輸出模組。`omlibdbi` 模組支持的資料庫系統有 Firebird/Interbase，MS SQL，Sybase，SQLite，ingres，甲骨文，mSQL，MySQL 和 PostgreSQL。

#### 7.2.6.3. 使能加密傳輸

網路傳輸的機密性和完整性可以通過 TLS 或 GSSAPI 加密協議來提供。

傳輸層安全性（TLS）是旨在提供通過網路通信的安全性的加密協議。當使用 TLS，`rsyslog` 消息被發送之前加密，發送者和接收者之間的需要相互認證。

通用安全服務 API（GSSAPI）是一種為程式訪問安全服務的應用程式編程介面。為了在與 `rsyslog` 的連接中使用，您必須有一個正常的 Kerberos 環境。

### 7.2.7. Syslogd 服務和日誌的交互

如上所述，`rsyslog` 和 `journal`，您系統上的兩個日誌應用程式，有幾個鮮明的特點，使它們適用於特定的用例。在很多情況下（見 7.3 Syslogd 日誌結構），



它們可以相互合作，比如創建結構化的資訊，並將它們存儲在一個檔資料庫中。需要這種合作的一個通信介面是由輸入和輸出模組提供的，這些模組由 **Rsyslog** 和 **journal** 的通信 **socket** 支持。

默認情況下，**rsyslogd** 使用 **imjournal** 模組作為日誌檔的默認的輸入模式。使用這個模組，您不僅可以導入消息，也可以導入由 **Journald** 提供的結構化數據。此外，舊的數據可以從 **journald** 導入（除非禁止用 **#ImjournalIgnorePreviousMessages** 指令）。參考 7.3.1 從日誌中導入數據部分關於 **imjournal** 的基本配置。

作為替代，配置 **rsyslogd** 來讀取由 **journal** 提供的套接字作為一個基於 **syslog** 應用的輸出。套接字的路徑 **/run/systemd/journal/syslog**。當您想維護 **rsyslog** 消息，請使用此選項。相比 **imjournal**，套接字輸入提供更多的功能，如綁定規則集或篩檢程式。要通過套接字導入日誌數據，在 **/etc/rsyslog.conf** 中使用以下配置：

```
$ModLoad imuxsock
$OmitLocalLogging off
```

上述語法加載 **imuxsock** 模組並關閉 **\$OmitLocalLogging** 指令，使能通過系統套接字導入。套接字的路徑分別在 **/etc/rsyslog.d/listen.conf** 中指定，如下：

```
$SystemLogSocketName /run/systemd/journal/syslog
```

您也可以從 **Rsyslog** 輸出消息到使用 **omjournal** 模組的 **journal**。在 **/etc/rsyslog.conf** 中配置輸出如下：

```
$ModLoad omjournal
*.* :omjournal:
```

例如，下麵的配置轉發所有收到的資訊通過 TCP 端口 10514 到 journal：

```
$ModLoad imtcp
$ModLoad omjournal
$RuleSet remote
*.* :omjournal:
$InputTCPServerBindRuleset remote
$InputTCPServerRun 10514
```

### 7.3. Syslogd 日誌結構

在產生大量的日誌數據的系統上，一個結構化格式の日誌資訊可以方便地被維護。通過結構化的資訊，很容易搜索特定資訊，生成統計資訊和處理消息的改變和不一致。rsyslog 使用 JSON（JavaScript 對象符號）格式提供日誌資訊的結構化。

比較下麵非結構化的日誌消息：

```
Oct 25 10:20:37 localhost anacron[1395]: Jobs will be executed
sequentially
```

下麵是結構化的日誌消息：

```
{"timestamp":"2020-2-7 T10:20:37", "host":"localhost",
  "program":"anacron", "pid":"1395", "msg":"Jobs will be
executed
sequentially"}
```

使用鍵值對搜索結構化數據比使用正則運算式搜索文本檔的速度更快更精確。結構化消息還可以讓您搜索各種應用程式生成的相同的消息。另外，JSON 檔可以存儲在文檔資料庫，如 MongoDB，它提供了額外的性能和分析功能。另一方面，一個結構化的消息需要比非結構化的消息更多的磁片空間。

在 `rsyslog`，帶有元數據的日誌資訊被使用 `imjournal` 模組的 `journal` 推出。使用 `mmjsonparse` 模組，可以解析來自 `journal` 和其他來源導入的數據，並進一步對其進行處理，例如，作為一個資料庫輸出。為了使解析成功，`mmjsonparse` 要求輸入消息是結構化的，向其在 `lumberjack` 專案定義的那樣。

`Lmberjack` 專案的目的是以向後相容的方式為 `rsyslog` 增加結構化日誌記錄。要確定一個結構化的消息，`Lmberjack` 指定 `@cee:` 字串，其前置 `JSON` 結構。另外，`Lmberjack` 定義了應該用在 `JSON` 串標準字段名稱的列表。有關 `Lmberjack` 的更多資訊，請參見線上文檔。

下麵是 `lumberjack` 格式消息的例子：

```
@    cee:    {"pid":17055,    "uid":1000,    "gid":1000,
"appname":"logger",
"msg":"Message text."}
```

要在 `Rsyslog` 中構建這個結構，一個範本會被使用，請參見 7.3.2 過濾結構化消息。應用程式和服務器可以採用 `libumberlog` 庫來生成 `lumberjack` 相容形式的消息。有關 `libumberlog` 的更多資訊，請參見線上文檔。

### 7.3.1. 從日誌中導入數據

`imjournal` 模組是 `rsyslog` 的輸入模組用來讀取日誌檔(見 7.2.7 `SSyslogd` 服務和日誌的交互。作為其他的 `rsyslog` 消息，日誌消息以文本格式被記錄。然而，隨著進一步的處理，其能夠將由 `journal` 提供的元數據翻譯成結構化的消息。

為了從 `Journal` 導入數據到 `Rsyslog`，在 `/etc/rsyslog.conf` 檔中使用下麵配置：

```
$ModLoad imjournal
$imjournalPersistStateInterval number_of_messages
```

```
$imjournalStateFile path
$imjournalRatelimitInterval seconds
$imjournalRatelimitBurst burst_number
$ImjournalIgnorePreviousMessages off/on
```

- 使用 `number_of_messages`，您可以指定保存日誌數據的頻率。

每當達到 `number_of_messages` 指定的消息數時，就會觸發數據保存；

- 用狀態檔的路徑替代 `path`。此檔跟蹤 `journal` 記錄，其是處理的最後一個記錄；

- 使用 `seconds`，設置的限制速率的時間間隔。此間隔期間處理的消息的數量不能超過在 `burst_number` 指定的值。默認設置為每 600 秒 20000 的消息。Rsyslog 會丟棄在規定的時限內超過最大 `burst_number` 值的消息；

- 使用 `#ImjournalIgnorePreviousMessages` 可以忽略當前在 `journal` 的消息和只導入新的消息，這些消息當未指定狀態檔時被使用。默認設置為關閉。請注意，如果該設置是關閉的，沒有狀態檔，所有 `journal` 的消息會被處理，即使他們已經在 `rsyslog` 以前的會話中被處理過。

注意：

您可以同時使用 `imjournal` 和 `imuxsock` 模組，`imuxsock` 模組是舊的系統日誌輸入。然而，為了避免消息重複，您必須阻止 `imuxsock` 讀取 `journal` 的系統套接字。要做到這一點，使用 `$OmitLocalLogging` 指令：

```
$ModLoad imuxsock
$ModLoad imjournal
$OmitLocalLogging on
$AddUnixListenSocket /run/systemd/journal/syslog
```

您可以通過將存儲在 `journal` 的數據和元數據翻譯成結構化消息。其中的一些元資料項目在本章實例“`journalctl` 的 `verbose` 輸出”中列出，想獲取 `journal` 域的完整列表，請參閱 `systemd.journal-fields` (7) 手冊頁。例如，可以將重點放在內核 `journal` 字段，該域被內核初始生成的消息使用。

### 7.3.2. 過濾結構化消息

要創建一個 `rsyslog` 解析模組需要的 `lumberjack` 格式的消息，請使用以下範本：

```
template(name="CEETemplate" type="string"
string="%TIMESTAMP% %HOSTNAME%
%syslogtag% @ cee: %$!all-json%\n")
```

這個範本預先考慮 `@cee:` 可以應用的 `JSON` 字串，例如，當創建使用 `omfile` 模組的輸出檔時。要訪問 `JSON` 字段名稱，使用 `$!` 首碼。例如，搜索符合下麵的篩選條件的消息，指定了消息的主機名和 `UID`。

```
($!hostname == "hostname" && $!UID == "UID")
```

### 7.3.3. 解析 `JSON`

`mmjsonparse` 模組用於解析結構化消息。這些資訊可以來自 `journal` 或其他輸入源，並且必須由 `lumberjack` 專案中定義的方式進行格式化。這些消息是由 `@cee:` 字串表示而被識別的。然後，`mmjsonparse` 會檢查 `JSON` 結構是否有效的，然後該消息被解析。

為了用 `mmjsonparse` 模組解析 `lumberjack` 格式的 `JSON` 消息，在 `/etc/rsyslog.conf` 檔下使用下麵的配置：

```
$ModLoad mmjsonparse
*.* :mmjsonparse:
```

在這個例子中, `mmjsonparse` 模組在第一行被加載, 所有的消息轉發給它。

目前, 沒有可用的配置參數用於 `mmjsonparse`。

#### 7.3.4. 向 MongoDB 中存儲消息

`rsyslog` 支持通過 `ommongodb` 輸出模組在 MongoDB 的文檔型資料庫中存儲 JSON 日誌。

要轉發日誌消息到 MongoDB 中, 在 `/etc/rsyslog.conf` 使用以下語法 ( `ommongodb` 配置參數只適用於新配置格式; 見 7.2.3 使用新的配置格式 )

```
$ModLoad ommongodb
*. *      action(type="ommongodb"      server="DB_server"
serverport="port"
      db="DB_name"      collection="collection_name"      uid="UID"
      pwd="password")
```

> 用 MongoDB 伺服器的地址或者名字替代 `DB_server`. 配置端口需要從 MongoDB 的伺服器選擇一個非標準的端口。默認端口值是 0, 並且通常沒有必要改變該參數;

> 使用 `DB_NAME`, 確定您想直接輸出到 MongoDB 的伺服器上的哪個資料庫。用這個資料庫集合的名稱替換 `collection_name`。在 MongoDB 中, 集合是一組文檔, 它和一個 RDBMS 表是等效的;

> 您通過設置 `UID` 和 `password` 來輸入您的細節資訊。

您可以使用範本來塑造最終資料庫的輸出形式。默認情況下, `rsyslog` 使用的範本基於標準的 `lumberjack` 字段名稱。

## 7.4. 調試 Rsyslog

在 `debug` 模式運行 `rsyslogd`, 使用如下命令:

```
rsyslogd -dn
```

使用此命令，`rsyslogd` 產生調試資訊並列印到標準輸出。`-n` 表示“no fork”。您可以修改調試環境變數，例如，可以在日誌檔中存儲調試輸出資訊。在啟動 `rsyslogd` 之前，在命令行上執行以下操作：

```
export RSYSLOG_DEBUGLOG="path"  
export RSYSLOG_DEBUG="Debug"
```

用所需記錄調試資訊檔的位置替換 `path`。對於可用於 `RSYSLOG_DEBUG` 變數選項的完整列表，請參閱 `rsyslogd (8)` 手冊的相關章節。

檢查在 `/etc/rsyslog.conf` 檔中使用的語法是否有效：

```
rsyslogd -N 1
```

其中，`1` 表示輸出消息的詳細程度的級別。這是前向相容性選項，因為目前，只有一個級被提供。但是，您必須添加此參數來運行驗證。

## 7.5. 使用日誌

`journal` 是 `systemd` 的一個組件，它負責查看和管理日誌檔。它可並行使用或者代替一個舊的 `syslog` 守護進程，如 `rsyslogd`。`journal` 的開發是為了解決與舊的日誌記錄有關的問題。它緊密地與系統的其餘部分集成，支持多種日誌記錄技術和訪問管理日誌檔。

記錄數據由 `journald` 服務收集，存儲，並處理。它創建和維護名為 `journals` 的二進位檔，這些記錄的資訊來自內核，用戶進程，標準輸出，標準錯誤輸出，或通過原生 API 的標準錯誤輸出。這些 `journals` 被結構化並被索引，它提供了比較快的查詢道時間。`journal` 條目有一個唯一的識別字。`journald` 服務收集每

個日誌消息眾多的元數據字段。日誌檔是安全的，不能被手動編輯。

### 7.5.1. 查看日誌檔

為了訪問 journal 日誌，使用 journalctl 工具。以 root 身份查看日誌類型的基本資訊：

```
journalctl
```

此命令的輸出是系統所有日誌檔的列表，包括系統組件和用戶產生的消息。

該輸出的結構類似於 /var/log/messages 檔中消息結構，但有一定的改進：

- 記錄優先順序的標記是可見的。錯誤的優先順序和更高的優先順序使用了紅色，通知和警告優先線使用加粗的字體；
- 時間戳轉換為系統的本地時區；
- 所有記錄數據都會顯示，包括輪替的日誌；
- 引導的開始將被標上專用線。

### 7.5.2. 訪問控制

默認情況下，沒有 root 許可權的 journal 用戶，只能查看由它們產生的日誌檔。系統管理員可以添加選定用戶到 adm 組，授予他們訪問完整日誌檔的許可權。要做到這一點，以 root 用戶輸入如下：

```
usermod -a -G adm username
```

用要添加到 adm 組的用戶名替代 username。該用戶隨後接收 journalctl 命令的輸出同 root 用戶獲取的輸出一樣。請注意，訪問控制只有當為 journal 啟用了持久存儲才會工作。



### 7.5.3. 使用 Live view

當不帶參數調用，`journalctl` 從收集到的最早的條目開始，顯示條目的完整列表。有了 `live view`，您可以監控即時的日誌消息，因為新的條目出現就會被列印出來。要開啟 `journalctl` 的 `live view` 模式，輸入：

```
journalctl -f
```

該命令返回十個最新的日誌行列表。`journalctl` 工具將保持運行，等待新的消息並立即顯示他們。

### 7.5.4. 過濾消息

不帶參數執行 `journalctl` 命令的輸出資訊是很多的，因此您可以使用各種過濾方法來提取資訊，以滿足您的需求。

#### 通過優先順序過濾

日誌消息通常用於跟蹤系統上錯誤的行為。要查看選定的或更高優先順序消息，請使用以下語法：

```
journalctl -p priority
```

在這裏，用下麵的關鍵字之一（或數字）替換優先順序：`debug`（7），`info`（6），`notice`（5），`warning`（4），`err`（3），`crit`（2），`alert`（1），和 `emerg`（0）。

#### 實例：通過優先順序過濾

查看優先順序為 `error` 或更高的消息，使用：

```
journalctl -p err
```

## 通過時間過濾

要查看僅從當前引導的日誌條目：

```
journalctl -b
```

如果您偶爾會重新啟動系統，**-b** 不會顯著減少 `journalctl` 的輸出。在這樣的情況下，基於時間的濾波是更有幫助：

```
journalctl --since=value --until=value
```

使用 `--since` 和 `--until` 選項。您可以查看指定時間範圍內創建的日誌資訊。您可以以 `time` 的格式或者 `data` 的格式或者兩者都可傳遞 `value` 給這兩個選項，如以下示例。

### 實例：通過優先順序和時間過濾

根據具體的要求過濾選項可以組合，以減少的結果輸出。例如，從某個時間點，查看警告或更高優先順序的消息，如下：

```
#journalctl -p warning --since="2020-3-5 23:59:59"
```

## 高級過濾

在本章實例“`journalctl` 的 `verbose` 輸出”列出指定日誌條目的一組字段，其都可以用於過濾。對於 `systemd` 可存儲的元數據的完整說明，參見 `systemd.journal-fields (7)` 手冊頁。每個日誌資訊的元數據都被收集，而無需用戶干預。值通常是基於文本的，但可以採取二進位和大的值；雖然不是很常見的，但是字段可以分配多個值。

要查看發生在一個特定領域產生的唯一值的列表，請使用以下語法：

```
journalctl -F fieldname
```

用您想要的字段名替代 `fieldname`。

要顯示出滿足特定條件日誌條目，請使用以下語法：

```
journalctl fieldname=value
```

用字段名和該字段包含的值替代 `fieldname` 和 `value`。結果，符合這個條件的行會輸出。

注意：

如通過 `systemd` 存儲的元數據字段的數量是相當大的，它很容易忘記感興趣的字段的確切名稱。如果不能確定，請鍵入：

```
journalctl
```

按兩次 `tab` 鍵。這會顯示出可用的字段名。`Tab` 鍵名稱補齊是基於該字段的上下文選項完成的，您可以輸入一部分字母，然後按 `Tab` 鍵來自動完成這個名字。同樣，您可以從一個字段中列出唯一值。如下：

```
journalctl fieldname=
```

按 `tab` 鍵兩次，就會像輸入 `journalctl -F fieldname` 一樣。

您可以在一個字段自定多個值，如下：

```
journalctl fieldname=value1 fieldname=value2 ...
```

同一個字段指定兩個可以匹配的值，像邏輯 `or` 一樣匹配。符合匹配值 1 或值 2 的條目會顯示。

當然，您可以指定多個 `field-value` 對來進一步減少輸出：

```
journalctl fieldname1=value fieldname2=value ...
```

如果指定了兩個不同的字段名的匹配，它們將與邏輯 **AND** 組合。必須匹配這兩個條件的條目才會顯示。

使用+號，您可以設置邏輯 **or** 組合匹配多字段：

```
journalctl fieldname1=value + fieldname2=value ...
```

該命令返回至少匹配一個條件的條目，不僅是滿足所有條件的條目。

### 實例：高級篩檢程式設置

要顯示通過 `avahi-daemon` 或 `crond.service` 創建的條目。且該條目的用戶 UID 為 70，請使用以下命令：

```
journalctl _UID=70 _SYSTEMD_UNIT=avahi-daemon.service  
_SYSTEMD_UNIT=crond.service
```

由於 `_SYSTEMD_UNIT` 字段可以設置兩個值，匹配兩者的結果將顯示，但 `_UID= 70` 條件必須被滿足。這可以簡單地表示為（`UID=70 and (avahi or cron)`）。

您可以應用到前面提到的篩檢程式，其以 `live-view` 模式來跟蹤選定組的日誌消息的變化：

```
journalctl -f fieldname=value ...
```

### 7.5.5. 使能持續存儲

默認情況下，`journal` 只在內存中或 `/run/log/journal/` 目錄下的小環形緩衝區中存儲日誌檔。使用 `journalctl` 顯示最近歷史日誌就足夠了。該目錄有易失性，日誌數據不會永久保存。在默認配置下，`syslog` 讀取日誌記錄，並將其存

儲在 `/var/log/` 目錄下。若持續性日誌記錄開啟，日誌檔存儲在 `/var/log/journal`，這意味著重新啟動後他們仍然存在。對一些用戶，`journal` 能代替 `rsyslog`。

使能持續性存儲有下麵的優點：

- 較長時間的可用來解決問題的更豐富的數據會被記錄；
- 對於需要立刻解決的問題，重啟後可以獲取更豐富的可用的數據；
- 伺服器控制臺當前從日誌中讀取數據，而不是日誌檔。

持續性存儲也有某些缺點：

- 持續性存儲所存儲的數據量取決於空間的記憶體，不保證可以覆蓋特定的時間跨度；
- 需要更多的磁片空間存儲日誌檔。

`journal` 啟用持續性存儲，按下麵所示的例子手動創建 `journal` 目錄，以 `root` 用戶輸入：

```
mkdir -p /var/log/journal
```

重啟 `journald` 來是設置生效。

```
systemctl restart systemd-journald
```

## 7.6. 自動化系統任務

任務，也被稱為工作，可以通過配置，在一個指定的日期，在一個指定的時間週期內，或者當系統的平均負載低於 0.8 時，自動地運行。

銀河麒麟高級伺服器操作系統預先配置了運行一些重要的系統任務，以保持系統的更新。例如，`locate` 命令使用的 `slocate` 資料庫每天都會更新。系統管理員可以使用自動任務來執行週期性的備份、監控系統、運行自定義腳本等等。

銀河麒麟高級伺服器操作系統提供了以下自動任務工具：`cron`、`anacron`、`at` 和 `batch`。

每一種工具都是為了調度不同的任務類型而被設計的，`Cron` 和 `Anacron` 調度重複發生的任務，`At` 和 `Batch` 調度一次性的任務（請分別參考 7.6.1 `Cron` 和 `Anacron` 和 7.6.8 `At` 和 `Batch`）。

銀河麒麟高級伺服器操作系統 V10 支持使用 `systemd.timer` 在一個指定的時間執行一個任務。參見 `systemd.timer(5)` 用戶手冊頁面瞭解更多資訊。

### 7.6.1. `Cron` 和 `Anacron`

`Cron` 和 `Anacron` 都是能夠調度重複性任務在一個確定的時間點執行的守護進程，時間點是通過準確的時間、月份中的某天、月份、一周中的某天、周來定義的。

`Cron` 的任務可以每分鐘都運行。但是，這個工具假定系統是持續運行的，如果在任務被安排的時間系統並沒有運行，則任務不會被執行。

另一方面，如果系統在任務被安排的時間被沒有運行，`Anacron` 會記住這個已經被安排的任務。一旦系統開始運行，這個任務將被馬上執行。然而，`Anacron` 對於一個任務一天只能運行一次。

### 7.6.2. 安裝 `Cron` 和 `Anacron`

要安裝 `Cron` 和 `Anacron`，您需要安裝 `Cron` 的 `cronie` 包和 `Anacron` 的 `cronie-anacron` 包（`cronie-anacron` 是 `cronie` 的一個子包）。

要確定您的系統上是否已經安裝了相應的包，可以執行以下命令：

```
rpm -q cronie cronie-anacron
```

如果已經安裝了，上述命令將返回 `cronie` 包和 `cronie-anacron` 包的完整名稱，否則將通知您相應的包不可用。

要安裝這些包，以 `root` 用戶按照下麵的格式來使用 `dnf` 命令：

```
dnf install package
```

例如，要同時安裝 `Cron` 和 `Anacron`，可以在命令行提示符下輸入以下命令：

```
#dnf install cronie cronie-anacron
```

要瞭解如何在銀河麒麟高級伺服器操作系統中安裝新的包，請參見 4.2.4 安裝軟體包。

### 7.6.3. 運行 `Cron` 服務

`cron` 和 `anacron` 任務都是由 `crond` 服務來控制的。本節將說明如何啟動、停止和重啟 `crond` 服務，以及如何配置讓其開機自啟動。要瞭解更多有關在銀河麒麟高級伺服器操作系統 V10 中如何管理服務的通用方法，請參見 5.1 使用 `systemd` 管理系統服務。

#### 7.6.3.1. 啟動 `Cron` 服務

要確定服務是否正在運行，請使用以下命令：

```
systemctl status crond.service
```

要在當前會話中運行 `crond` 服務，請以 `root` 用戶在命令行提示符下輸入以下命令：

```
systemctl start crond.service
```

要配置服務開機自啟動，請以 root 用戶執行以下命令：

```
systemctl enable crond.service
```

#### 7.6.3.2. 停止 Cron 服務

要在當前會話中停止 crond 服務，請以 root 用戶在命令行提示符下輸入以下命令：

```
systemctl stop crond.service
```

要禁止服務開機自啟動，請以 root 用戶執行以下命令：

```
systemctl disable crond.service
```

#### 7.6.3.3. 重啟 Cron 服務

要重啟 crond 服務，請以 root 用戶在命令行提示符下輸入以下命令：

```
systemctl restart crond.service
```

該命令停止服務後將很快地再次啟動服務。

#### 7.6.4. 配置 Anacron 任務

調度任務的主要配置檔是/etc/anacrontab 檔，它只能通過 root 用戶進行訪問。該檔包含以下內容：

```
SHELL=/bin/sh
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
#the maximal random delay added to the base delay of the
jobs
RANDOM_DELAY=45
#the jobs will be started during the following hours only
START_HOURS_RANGE=3-22
#period in days    delay in minutes    job-identifier
```



```

command
1      5      cron.daily      nice run-parts
/etc/cron.daily
7      25     cron.weekly     nice run-parts
/etc/cron.weekly
@monthly 45     cron.monthly    nice run-parts
/etc/cron.monthly

```

前三行定義用來配置 **anacron** 任務運行環境的相關變數：

- **SHELL**——用來運行任務的 **shell** 環境（示例中是 **Bash shell**）；
- **PATH**——可執行程式的路徑；
- **MAILTO**——通過電子郵件接收 **anacron** 任務輸出的用戶的名稱；

如果沒有定義 **MAILTO** 變數（**MAILTO=**），則不會發送郵件。

接下來的兩個變數用來修改已定義任務的調度時間：

- **RANDOM\_DELAY**——將會加到為每個任務指定的 **delay in minutes**

變數上的最大分鐘數；

默認情況下，最小延遲的值被設置為 **6** 分鐘。

例如，如果 **RANDOM\_DELAY** 被設置為 **12**，那麼這個特定的 **anacrontab** 中的每一個任務的 **delay in minutes** 值都將被加上 **6** 到 **12** 分鐘。

**RANDOM\_DELAY** 的值也可以設置為 **6** 以下，包括 **0**。當設置為 **0** 的時候，則不會增加隨機延遲。隨機延遲被證明是很有用的，例如，當多臺共用一個網路連接的電腦需要每天都下載相同的數據時。

- **START\_HOURS\_RANGE**——計畫任務可以運行的時間段，以小時為單位；

萬一錯過了這個時間段，例如，由於停電等原因，則當天的計畫任務不會被執行。

`/etc/anacrontab` 檔的剩餘行代表計畫任務，並且遵從以下格式：

```
period in days    delay in minutes  job-identifier
command
```

- `period in days`——按天數計的任務執行頻率；

該屬性值可以被定義為一個整數或者一個宏（`@daily`、`@weekly` 或者 `@monthly`），`@daily` 和整數 1 表示的是同一個值，`@weekly` 和整數 7 一樣，而 `@monthly` 則表示任務一個月只運行一次，不管這個月的天數是多少。

- `delay in minutes`——在執行任務前，`anacron` 等待的分鐘數；

該屬性值可以被定義為一個整數。如果該值被設置為 0，則表示沒有延遲。

- `job-identifier`——用於日誌檔中關聯一個特定任務的唯一名稱；
- `command`——將被執行的命令；

這裏的命令既可以是一個類似 `ls /proc >>/tmp/proc` 的命令，也可以是一個執行自定義腳本的命令。

以井號（`#`）開頭的所有行都是注釋，不會被處理。

#### 7.6.4.1. Anacron 任務示例

以下是一個簡單的 `/etc/anacrontab` 檔的示例：

```
SHELL=/bin/sh
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root

#the maximal random delay added to the base delay of the
```

```
jobs
RANDOM_DELAY=30
#the jobs will be started during the following hours only
START_HOURS_RANGE=16-20

#period in days    delay in minutes    job-identifier
command
1      20      dailyjob      nice run-parts
/etc/cron.daily
7      25      weeklyjob
/etc/weeklyjob.bash
@monthly 45      monthlyjob      ls /proc >>
/tmp/proc
```

在這個 `anacrontab` 檔中定義的所有任務，都將隨機的延遲 6 到 30 分鐘，並且將在 16:00 到 20:00 之間被運行。

第一個被定義的任務將在每天 16:26 到 16:50 間被觸發（`RANDOM_DELAY` 是在 6 到 30 分鐘之間；`delay in minutes` 屬性值為 20 分鐘）。該任務所指定的命令將使用 `run-parts` 腳本（`run-parts` 腳本接受一個目錄作為命令行參數，並順序地執行目錄中的每一個程式）執行 `/etc/cron.daily/` 目錄下的所有程式。參見 `run-parts` 用戶手冊頁面瞭解更多有關 `run-parts` 腳本的資訊。

第二個任務每週執行一次 `/etc/` 目錄下的 `weeklyjob.bash` 腳本。

第三個任務運行一個命令，該命令把 `/proc` 的內容寫到 `/tmp/proc` 檔裏，每個月一次（`ls /proc >> /tmp/proc`）。

### 7.6.5. 配置 Cron 任務

Cron 任務的配置檔是 `/etc/crontab` 檔，它只能通過 `root` 用戶進行訪問。

該檔包含以下內容：

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/
#For details see man 4 crontabs

#Example of job definition:
#----- minute (0 - 59)
#| .----- hour (0 - 23)
#| | .----- day of month (1 - 31)
#| | | .----- month (1 - 12) OR jan,feb,mar,apr ...
#| | | | .---- day of week (0 - 6) (Sunday=0 or 7) OR
sun,mon,tue,wed,thu,fri,sat
#| | | | |
#* * * * * user-name  command to be executed
```

前三行包含了和 `anacrontab` 檔一樣的變數定義：`SHELL`、`PATH` 和 `MAILTO`。

要瞭解更多有關這幾個變數的資訊，請參見 7.6.4 配置 Anacron 任務。

此外，該檔還可以定義 `HOME` 變數。`HOME` 變數定義一個目錄，該目錄在任務執行命令或腳本時將被作為家目錄。

`/etc/crontab` 檔的剩餘行代表計畫任務，並遵從以下格式：

```
minute    hour    day    month    day of week
username  command
```

以下各項定義了任務將要運行的時間：

- `minute`——從 0 到 59 的任意整數；

- **hour**——從 0 到 23 的任意整數；
- **day**——從 1 到 31 的任意整數（如果指定了月份，則這裏的日期必須是相對於該月有效的日期）；
- **month**——從 1 到 12 的任意整數（或者月份的簡短名稱，例如 **jan** 或者 **feb**）；
- **day of week**——從 0 到 7 的任意整數，0 或 7 代表星期日（或者使用每週各天的簡短名稱，例如 **sun** 或者 **mon**）；

以下各項定義了任務的其他屬性：

- **username**——指定執行任務的用戶；
- **command**——將要執行的命令。

這裏的命令既可以是一個類似 `ls /proc >>/tmp/proc` 的命令，也可以是一個執行自定義腳本命令

對於以上的各項屬性值，星號（\*）可以用來表示所有的有效值。例如，如果您將 **month** 的值定義為星號，則該任務將在其他條件的約束下每個月都執行。

整數之間的連字型大小（-）代表了一個整數範圍。例如，**1-4** 表示整數 1、2、3、4。

用逗號（,）分隔的值列表指定了一個列表。例如，**3,4,6,8** 就確實指明了這四個整數。

斜杠（/）可以用來指定步長值。指定了步長值的項將按步長所定義的整數來增長。例如，**minute** 項的值定義為 **0-59/2**，則表示每隔一分鐘。步長值也可以和星號一起使用。例如，如果 **month** 項的值被定義為 **\*/3**，則任務將每三


個月執行一次（每隔兩個月）。

以井號（#）開頭的所有行都是注釋，不會被處理。

非 root 用戶可以使用 `crontab` 工具來配置 `cron` 任務。用戶定義的 `crontabs` 被保存在 `/var/spool/cron/` 目錄下，並且以創建它們的用戶來執行。

要以一個特定用戶來創建 `crontab`，請以該用戶進行登錄，並且輸入命令 `crontab -e` 來使用 `VISUAL` 或者 `EDITOR` 環境變數所指定的編輯器對用戶的 `crontab` 進行編輯。該檔使用和 `/etc/crontab` 完全相同的格式。當用戶對 `crontab` 的修改被保存時，`crontab` 按照用戶名來保存，並寫入 `/var/spool/cron/username` 檔中。要列出當前用戶的 `crontab` 檔的內容，使用 `crontab -l` 命令。

`/etc/cron.d/` 目錄下包含的檔和 `/etc/crontab` 檔具有相同的語法。只有 root 用戶被允許在該目錄下創建和修改檔。

 `cron` 守護進程會每分鐘檢查 `/etc/anacrontab` 檔、`/etc/crontab` 檔、`/etc/cron.d/` 目錄和 `/var/spool/cron/` 目錄的變化，並把檢測到的修改加載到記憶體中。因此，當一個 `anacrontab` 或 `crontab` 檔被修改後，並不需要重啟守護進程。

#### 7.6.6. 控制對 Cron 的訪問

要限制對 `Cron` 的訪問，您可以使用 `/etc/cron.allow` 和 `/etc/cron.deny` 檔。這些訪問控制檔使用相同的格式，都是每一行一個用戶名。記住，兩個檔都不允許使用空格。

如果存在 `cron.allow` 檔，只有在該檔中列出的用戶才能使用 `cron`，並且

`cron.deny` 檔將被忽略。

如果 `cron.allow` 檔不存在，則 `cron.deny` 檔中列出的用戶將被禁止使用 Cron。

如果訪問控制檔被修改了，不必重啟 Cron 守護進程（`crond`）。每次用戶試圖添加或刪除一個 `cron` 任務時，都會檢查訪問控制檔。

不管訪問控制檔中的用戶名是什麼，`root` 用戶都始終能夠使用 `cron`。

您也可以通過插入式認證模組（`Pluggable Authentication Modules, PAM`）來控制訪問。相應的設置保存在 `/etc/security/access.conf` 檔中。例如，在檔中添加如下一行之後，將只有 `root` 用戶能夠創建 `crontabs`：

```
 -:ALL EXCEPT root :cron
```

被禁止的任務將被記錄在適當的日誌檔中，或者，當使用 `crontab -e` 命令時，返回到標準輸出裏。要瞭解更多資訊，請參考 `access.conf.5` 用戶手冊頁面。

#### 7.6.7. Cron 任務的黑白名單

任務的黑白名單用來定義任務的某部分不需要被執行。當在一個 Cron 目錄裏（例如，`/etc/cron.daily/`）調用 `run-parts` 腳本時這很有用：如果用戶把這個目錄下的程式加入了黑名單，`run-parts` 腳本將不會執行這些程式。

要創建一個黑名單，請在 `run-parts` 腳本將要執行的目錄中創建一個 `jobs.deny` 檔。例如，如果您需要從 `/etc/cron.daily/` 目錄中忽略一個特定的程式，請創建 `/etc/cron.daily/jobs.deny` 檔。在這個檔中，指定需要在執行時被忽略的程式的名稱（只有位於同一個目錄下的程式能夠加入列表）。如果一項任

務執行一個命令，該命令從 `/etc/cron.daily/` 目錄執行程式，例如 `run-parts /ect/cron.daily`，則 `jobs.deny` 檔中定義的程式不會被執行。

要定義一個白名單，請創建 `jobs.allow` 檔。

`jobs.deny` 和 `jobs.allow` 的使用規則和 8.7.6 控制對 Cron 的訪問中描述的 `cron.deny` 和 `cron.allow` 的使用規則一樣。

### 7.6.8. At 和 Batch

Cron 被用來調度重複性任務，At 工具被用來在一個指定的時間調度一次性任務，Batch 工具被用來調度將在系統平均負載低於 0.8 時被執行的一次性任務。

#### 7.6.8.1. 安裝 At 和 Batch

要確定您的系統上是否已經安裝了 `at` 包，請執行以下命令：

```
rpm -q at
```

如果已經安裝了，上述命令將返回 `at` 包的完整名稱，否則將通知您包不可用。

要安裝程式包，以 `root` 用戶按照下麵的格式來使用 `dnf` 命令：

```
dnf install package
```

例如，要同時安裝 `At` 和 `Batch`，可以在命令行提示符下輸入以下命令：

```
#dnf install at
```

#### 7.6.8.2. 運行 At 服務

`At` 和 `Batch` 任務都是由 `atd` 服務來控制的。本節將說明如何啟動、停止和



重啟 **atd** 服務，以及如何配置讓其開機自啟動。

### 啟動 **At** 服務

要確定服務是否正在運行，請使用以下命令：

```
systemctl status atd.service
```


要在當前會話中運行 **atd** 服務，請以 **root** 用戶在命令行提示符下輸入以下

命令：

```
systemctl start atd.service
```

要配置服務開機自啟動，請以 **root** 用戶執行以下命令：

```
systemctl enable atd.service
```

 建議您將 **atd** 服務在您的系統中配置為開機自啟動。

### 停止 **At** 服務

要在當前會話中停止 **atd** 服務，請以 **root** 用戶在命令行提示符下輸入以下

命令：

```
systemctl stop atd.service
```

要禁止服務開機自啟動，請以 **root** 用戶執行以下命令：

```
systemctl disable atd.service
```

### 重啟 **At** 服務

要重啟 **atd** 服務，請以 **root** 用戶在命令行提示符下輸入以下命令：

```
systemctl restart atd.service
```

該命令停止服務後將很快地再次啟動服務。

### 7.6.8.3. 配置 At 任務

要使用 At 工具調度一次性任務在指定時間執行，請按以下步驟操作：

1. 在命令行中輸入命令 `at TIME`，這裏的 `TIME` 表示命令執行的時間。

`TIME` 參數可以使用以下任意一種格式定義：

- `HH:MM`：指定確切的小時和分鐘，例如，`04:00` 表示上午 4 點；
- `midnight`：指定為午夜 12 點；
- `noon`：指定為中午 12 點；
- `teatime`：指定為下午 4 點；
- `MONTHDAYYEAR` 格式：例如，`January 15 2020` 表示 2020 年 1 月

15 日，年份的值是可選的；

- `MMDDYY`、`MM/DD/YY` 或者 `MM.DD.YY` 格式：例如，`011520` 表示 2020 年 1 月 15 日；

- `now + TIME`：這裏的 `TIME` 由一個整數和 `minutes`、`hours`、`days` 或者 `weeks` 幾個類型值來定義。例如，`now+5days` 表示命令將在從現在起五天後的同一時間被執行。

必須首先指定時間，其後可以指定可選的日期。要瞭解更多關於時間格式的資訊，請參考 `/usr/share/doc/at-<version>/timespec` 文本檔。

如果指定的時間已經過了，則任務將在明天的同一時間被執行。

2. 在顯示出的 `at>` 命令行提示符下，定義任務命令：

A. 輸入任務應該執行的命令，並按下回車鍵。可選地，可以重複此步驟，提供多個命令。

B. 在命令行提示符下輸入一個 **shell** 腳本，並在腳本的每一行之後按下回車鍵。

任務將使用用戶的 **SHELL** 環境變數所設置的 **shell**、用戶的登錄 **shell**，或者 **/bin/sh**（無論先找到哪一個）。

3. 一旦輸入結束，請在一個空行中按下 **Ctrl+D** 組合鍵，退出命令行提示符。

如果這一系列命令或者腳本試圖在標準輸出中顯示資訊，則輸出將會以電子郵件的形式發送給用戶。

要查看等待執行的任務列表，請使用 **atq** 命令。請參見 7.6.8.5 查看等待執行的任務瞭解更多資訊。

您也可以限制 **at** 命令的使用。要瞭解更多資訊，請參見 7.6.8.7 控制對 **At** 和 **Batch** 的訪問。

#### 7.6.8.4. 配置 **Batch** 任務

**Batch** 程式在系統平均負載降低到 **0.8** 以下的時候執行已定義的一次性任務。

要定義 **Batch** 任務，請按以下步驟進行操作：

1. 在命令行中輸入 **batch** 命令。
2. 在顯示出的 **at>** 命令行提示符下，定義任務命令：
  - A. 輸入任務應該執行的命令，並按下回車鍵。可選地，可以重複此步驟，提供多個命令。
  - B. 在命令行提示符下輸入一個 **shell** 腳本，並在腳本的每一行之後按

下回車鍵。

任務將使用用戶的 SHELL 環境變數所設置的 shell、用戶的登錄 shell，或者/bin/sh（無論先找到哪一個）。

3. 一旦輸入結束，請在一個空行中按下 Ctrl+D 組合鍵，退出命令行提示符。

如果這一系列命令或者腳本試圖在標準輸出中顯示資訊，則輸出將會以電子郵件的形式發送給用戶。

要查看等待執行的任務列表，請使用 atq 命令。請參見 7.6.8.5 查看等待執行的任務瞭解更多資訊。

您也可以限制 batch 命令的使用。要瞭解更多資訊，請參見 7.6.8.7 控制對 At 和 Batch 的訪問。

#### 7.6.8.5. 查看等待執行的任務

要查看等待執行的 At 和 Batch 任務，可以執行 atq 命令。atq 命令將顯示一個等待執行的任務的列表，每個任務單獨顯示為一行。每一行的格式為任務編號、日期、任務類型和用戶名稱。用戶只能查看他們自己的任務。如果 root 用戶執行 atq 命令，則會顯示所有用戶的所有任務。

#### 7.6.8.6. 額外的命令行選項

at 和 batch 命令的額外命令行選項如下所示：

**表 7-5at 和 batch 命令行選項**

選項	描述
----	----

選項	描述
-f	從一個檔中讀取命令或 shell 腳本，而不是在命令行提示符下輸入命令或腳本。
-m	當任務完成後向用戶發送電子郵件。
-v	顯示任務被執行的時間。

#### 7.6.8.7. 控制對 At 和 Batch 的訪問

您可以使用 `/etc/at.allow` 和 `/etc/at.deny` 檔來限制對 `at` 和 `batch` 命令的訪問。這些訪問控制檔使用相同的格式，都是每一行一個用戶名。記住，兩個檔都不允許使用空格。

如果存在 `at.allow` 檔，只有在該檔中列出的用戶才能使用 `at` 或者 `batch`，並且 `at.deny` 檔將被忽略。

如果 `at.allow` 檔不存在，則 `at.deny` 檔中列出的用戶將被禁止使用 `at` 或者 `batch`。

如果訪問控制檔被修改了，不必重啟 `at` 守護進程（`atd`）。每次用戶試圖執行 `at` 或 `batch` 命令時，都會讀取訪問控制檔。

不管訪問控制檔中的內容是什麼，`root` 用戶都始終能夠執行 `at` 和 `batch` 命令。

## 第八章 系統安全

### 8.1. 安全基礎服務

#### 8.1.1. 防火牆

防火牆是系統的主要的安全工具，可以提供基本的安全防護。`firewalld` 支持網路/防火牆區域(zone)定義網路鏈接、是介面安全等級的動態防火牆管理工具。它支持 IPv4,IPv6 防火牆設置以及以太網橋接，並且擁有運行時配置和永久配置選項。它也支持允許服務或者應用程式直接添加防火牆規則的介面且可以動態管理防火牆。

##### 8.1.1.1. 主要功能

- 1) 實現動態管理，對於規則的更改不再需要重新創建整個防火牆；
- 2) 提供 `firewall-cmd` 命令行介面進行管理及配置工作；
- 3) 實現 `firewall-config` 圖形化配置工具；
- 4) 實現系統全局及用戶進程的防火牆規則配置管理；
- 5) 區域支持。

##### 8.1.1.2. 基本命令

安裝命令如下：

```
dnf install firewalld
```

啟動命令如下：

```
systemctl enable firewalld.service  
systemctl start firewalld.service
```

關閉命令如下：

```
systemctl stop firewalld
systemctl disable firewalld
```

查看狀態命令如下：

```
systemctl status firewalld
```

### 8.1.1.3. 區域管理

通過將網路劃分成不同的區域（通常情況下稱為 **zones**），制定出不同區域之間的訪問控制策略來控制其間傳送的數據流。例如互聯網是不可信任的區域，而內部網路是高度信任的區域，以避免安全策略中禁止的一些通信。典型信任的區域包括互聯網（一個沒有信任的區域）和一個內部網路（一個高信任的區域）。最終目標是提供受控連通性在不同水準的信任區域通過安全政策的運行和連通性模型之間根據最少特權原則。例如：公共 **WIFI** 網路連接應該不信任，而家庭有線網路連接就應該完全信任。網路安全模型可以在安裝、初次啟動和首次建立網路連接時選擇初始化。該模型描述了主機所聯的整個網路環境的可信級別，並定義了新連接的處理方式。在 `/etc/firewalld/` 的區域設定是一系列可以被快速執行到網路介面的預設定。有幾種不同的初始化區域：

#### drop（丟棄）

任何接收的網路數據包都被丟棄，沒有任何回復。僅能有發送出去的網路連接。

#### block（限制）

任何接收的網路連接都被 IPv4 的 `icmp-host-prohibited` 資訊和 IPv6 的 `icmp6-adm-prohibited` 資訊所拒絕。

#### public（公共）

在公共區域內使用，不能相信網路內的其他電腦，只能接收經過選取的連接。

#### **external**（外部）

特別是為路由器啟用了偽裝功能的外部網。您不能信任來自網路的其他電腦，只能接收經過選擇的連接。

#### **dmz**（非軍事區）

用於您的非軍事區內的電腦，此區域內可公開訪問，可以有限地進入您的內部網路，僅僅接收經過選擇的連接。

#### **work**（工作）

用於工作區。您可以基本相信網路內的其他電腦不會危害您的電腦。僅僅接收經過選擇的連接。

#### **home**（家庭）

用於家庭網路。您可以基本信任網路內的其他電腦不會危害您的電腦。僅僅接收經過選擇的連接。

#### **internal**（內部）

用於內部網路。您可以基本上信任網路內的其他電腦不會威脅您的電腦。僅僅接受經過選擇的連接。

#### **trusted**（信任）

可接受所有的網路連接。

說明：**firewalld** 的缺省區域是 **public**。

顯示支持的區域列表

```
firewall-cmd --get-zones
```



設置為家庭區域

```
firewall-cmd --set-default-zone=home
```

查看當前的區域

```
firewall-cmd --get-active-zones
```

設置當前的區域的介面

```
firewall-cmd --get-zone-of-interface=enp03s
```

顯示所有公共區域 ( public )

```
firewall-cmd --zone=public --list-all
```

臨時修改網路介面 enp0s3 為內部區域 ( internal )

```
firewall-cmd --zone=internal --change-interface=enp03s
```

永久修改網路介面 enp0s3 為內部區域 ( internal )

```
firewall-cmd --permanent --zone=internal  
--change-interface=enp03s
```

#### 8.1.1.4. 服務管理

amanda、ftp、samba 和 tftp 等重要的服務已被 firewalld 提供相應的服務，可以使用命令查看：

```
firewall-cmd --get-services
```

顯示當前服務

```
firewall-cmd --list-services
```

添加 http 服務到內部區域 ( internal )

```
firewall-cmd --permanent --zone=internal
--add-service=http
firewall-cmd --reload
```

將一個服務加入到分區

要把一個服務加入到分區，例如允許 SMTP 接入工作區：

```
firewall-cmd --zone=work --add-service=smtp
firewall-cmd --reload
```

要從分區移除服務，比如從工作區移除 SMTP：

```
firewall-cmd --zone=work --remove-service=smtp
firewall-cmd --reload
```

#### 8.1.1.5. 端口管理

打開端口

打開 443/tcp 端口在內部區域（internal）：

```
firewall-cmd --zone=internal --add-port=443/tcp
firewall-cmd --reload
```

端口轉發

```
firewall-cmd --zone=external --add-masquerade
firewall-cmd --zone=external
--add-forward-port=port=22:proto=tcp:toport=3777
```

上面的兩個命令的意思是，首先啟用偽裝（masquerade），然後把外部區域（external）的 22 端口轉發到 3777。

#### 8.1.1.6. 直接介面

firewalld 有一個被稱為“direct interface”（直接介面），它可以直接通過 iptables、ip6tables 和 ebtables 的規則。它適用於應用程式，而不是用戶。firewalld 保持對所增加專案的追蹤，所以它還能質詢 firewalld 和發現由使用

直接端口模式的程式造成的更改。直接端口由增加--direct 選項使用。直接端口模式適用於服務或者程式，以便在運行時間內增加特定的防火牆規則。這些規則不是永久性的。例如添加端口 tcp 9000 端口。

```
firewall-cmd --direct --add-rule ipv4 filter INPUT 0 -p tcp --dport
9000 -j ACCEPT
firewall-cmd --reload
```

#### 8.1.1.7. 富規則 (Rich Language)

通過“rich language”語法，可以用比直接介面方式更易理解的方法建立複雜防火牆規則。此外還能永久保留設置。這種語言可以用來配置分區，也仍然支持現行的配置方式。所有命令都必須以 root 用戶身份運行。增加一項規則的命令格式如下：

```
#firewall-cmd [--zone=zone] --add-rich-rule='rule' [--timeout
9=seconds]
  移除一項規則：
  firewall-cmd [--zone=zone] --remove-rich-rule='rule'
  檢查一項規則是否存在：
  firewall-cmd [--zone=zone] --query-rich-rule='rule'
```

一個具體例子，假設在一個 IP 地址 (192.168.0.0) 的伺服器配置防火牆

允許如下服務 http,https,vnc-server,PostgreSQL。

```
firewall-cmd --add-rich-rule 'rule family="ipv4" source
address="192.168.0.0/24" service name="http" accept'
firewall-cmd --add-rich-rule 'rule family="ipv4"
source address="192.168.0.0/24" service name="http" accept'
--permanent
firewall-cmd --add-rich-rule 'rule family="ipv4" source
address="192.168.0.0/24" service name="https" accept'
firewall-cmd --add-rich-rule 'rule family="ipv4" source
```

```
address="192.168.0.0/24" service name="https" accept'  
--permanent  
firewall-cmd --add-rich-rule 'rule family="ipv4" source  
address="192.168.0.0/24" service name="vnc-server" accept'  
firewall-cmd --add-rich-rule 'rule family="ipv4" source  
address="192.168.0.0/24" service name="vnc-server" accept'  
--permanent  
firewall-cmd --add-rich-rule 'rule family="ipv4" source  
address="192.168.0.0/24" service name="postgresql" accept'  
firewall-cmd --add-rich-rule 'rule family="ipv4" source  
address="192.168.0.0/24" service name="postgresql" accept'  
--permanent  
firewall-cmd --reload
```

#### 8.1.1.8. 創建服務

首先需要建立的服務是 RTMP( RTMP 是 Real Time Messaging Protocol (即時消息傳輸協議) 的首字母縮寫。該協議基於 TCP ) 端口號 1935。在 `/etc/firewalld/services/` 目錄中，利用現有的配置檔如 `nfs.xml` 作為範本。

```
cd /etc/firewalld/services/
```

說明：該目錄中存放的是定義好的網路服務和端口參數，只用於參考，不能修改。這個目錄中只定義了一部分通用網路服務。在該目錄中沒有定義的網路服務，也不必再增加相關 xml 定義，後續通過管理命令可以直接增加。

```
cp /usr/lib/firewalld/services/nfs.xml /etc/firewalld/services/
```

說明：從上面目錄中將需要使用的服務的 xml 檔拷至這個目錄中，如果端口有變化則可以修改檔中的數值。

```
#cd /etc/firewalld/services/  
下麵修改 nfs.xml 為 rtmp.xml
```

```
#mv nfs.xml rtmp.xml
```

下面使用 vi 編輯器修改 rtmp.xml 檔為如下內容：

```
<?xml version="1.0" encoding="utf-8"?>
<service>
<short>rtmp </short>
<description>RTMP Stream </description>
<port protocol="tcp" port="1935"/>
</service>
```

每一個服務定義都需要一個簡短的名字、描述和端口網路用於指定需要使用的協議、端口和模組名。然後把此服務加入防火牆規則中。

```
systemctl restart firewalld.service
firewall-cmd --add-service=rtmp
firewall-cmd --add-service=rtmp --permanent
firewall-cmd --reload
```

#### 8.1.1.9. firewall-config

firewall-config 支持防火牆的所有特性。管理員可以用它來改變系統或用戶策略。通過 firewall-config 用戶可以配置防火牆允許通過的服務、端口、偽裝、端口轉發、和 ICMP 篩檢程式和調整 zone（區域）設置等功能以使防火牆設置更加自由、安全和強健。

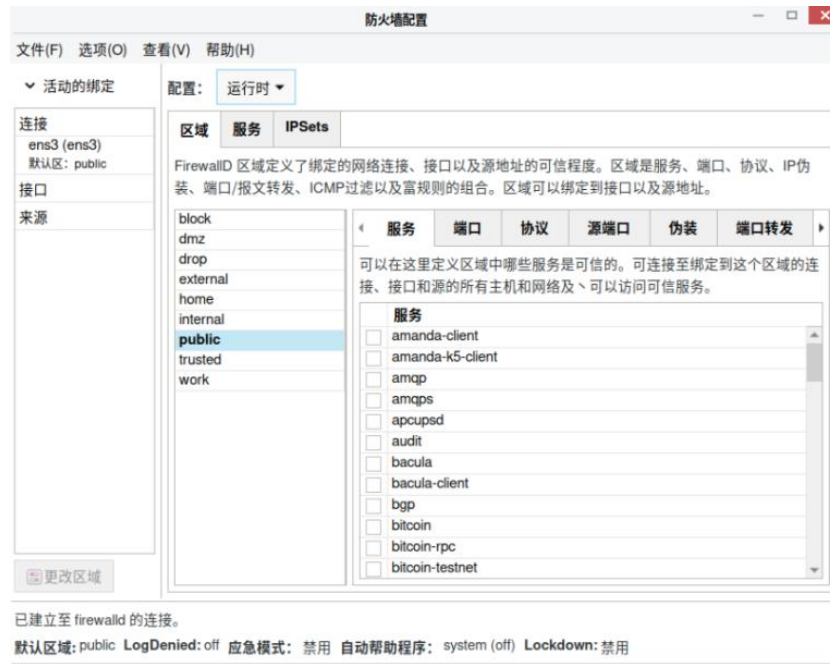


圖 8-1 防火牆配置

firewall-config 工作介面分成三個部分：上面是主菜單，中間是配置選項卡。下麵是區域、服務、ICMP 端口、白名單等設置選項卡。

說明：在左下方角落尋找“連接”字元，這標誌著 firewall-config 工具已經連接到用戶區後臺程式 firewalld。注意，ICMP 類型、直接配置（Direct Configuration）和鎖定白名單（Lockdown Whitlist）標籤只在從查看下拉菜單中選擇之後才能看見。

### 1) firewall-config 主菜單

firewall-config 主菜單包括四個選項：檔，選項，查看，幫助。其中選項子菜單是最主要的，它包括幾個部分：

**重載防火牆：**重載防火牆規則。例如所有現在運行的配置規則如果沒有在永久配置中操作，那麼系統重載後會丟失。

**更改連接區域：**更改網路連接的默認區域。

改變默認區域：更改網路連接的所屬區域和介面。

應急模式：應急模式意味著丟棄所有的數據包。

鎖定：鎖定可以對防火牆配置進行加鎖，只允許白名單上的應用程式進行改動。鎖定特性為 `firewalld` 增加了鎖定本地應用或者服務配置的簡單配置方式。它是一種羽量級的應用程式策略。

## 2) 配置選項卡

`firewall-config` 配置選項卡包括：運行時和永久。

運行時：運行時配置為當前使用的配置規則。運行時配置並非永久有效，在重新加載時可以被恢復，而系統或者服務重啟、停止時，這些選項將會丟失。

永久：永久配置規則在系統或者服務重啟的時候使用。永久配置存儲在配置檔種，每次機器重啟或者服務重啟、重新加載時將自動恢復。

## 3) 區域選項卡

區域選項卡是一個主要設置介面：

網路或者防火牆區域定義了連接的可信程度。`firewalld` 提供了幾種預定義的區域。這裏的區域是服務、端口、協議、偽裝、ICMP 過濾等組合的意思。區域可以綁定到介面和源地址。服務子選項卡定義哪些區域的服務是可信的。可信的服務可以綁定該區的任意連接、介面、和源地址；

### i. 服務選項卡

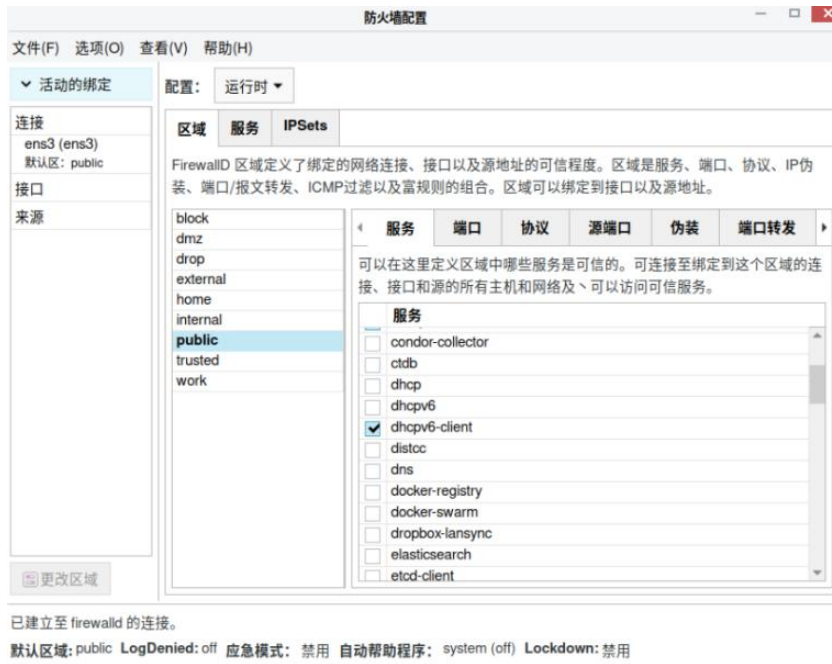


圖 8-2 服務選項卡

ii. 端口子選項卡

端口子選項卡用來設置允許主機或者網路訪問的端口範圍。

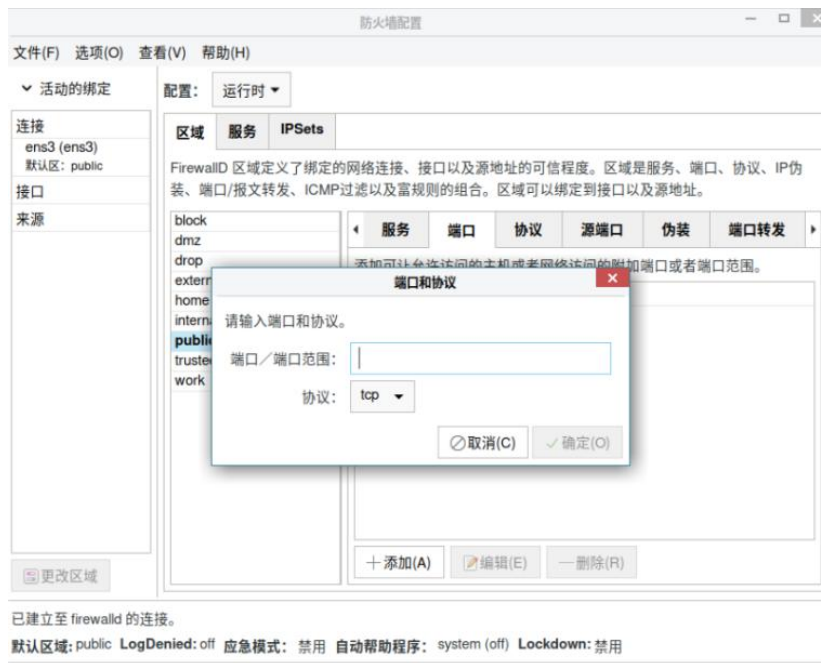


圖 8-3 端口子選項卡

要允許流量通過防火牆到達某個端口，則啟動 `firewall-config` 並選擇您想



更改設定的網路區域。選擇端口圖示並點擊右邊的添加按鈕, Port and Protocol 就打開了。

輸入端口數量或者端口號範圍,獲得許可。從下拉菜單中選擇 tcp 或者 udp。

### iii. 偽裝子選項卡

偽裝子選項卡用來把私有網路地址可以被映射到公開的 IP 地址

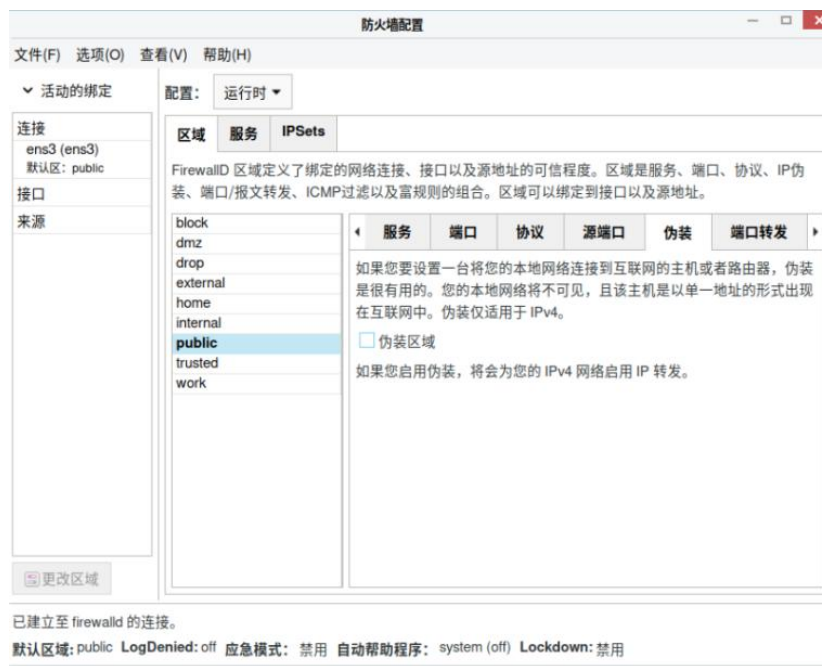


圖 8-4 偽裝子選項卡

要將 IPv4 地址轉換為一個單一的外部地址,則啟動 firewall-config 工具並選擇需要轉換地址的網路區域。選擇偽裝標籤和複選框以便把 IPv4 地址轉換成一個單一的地址。

### iv. 端口轉發子選項卡

端口轉發可以映射到另一個端口以及/或者其他主機;

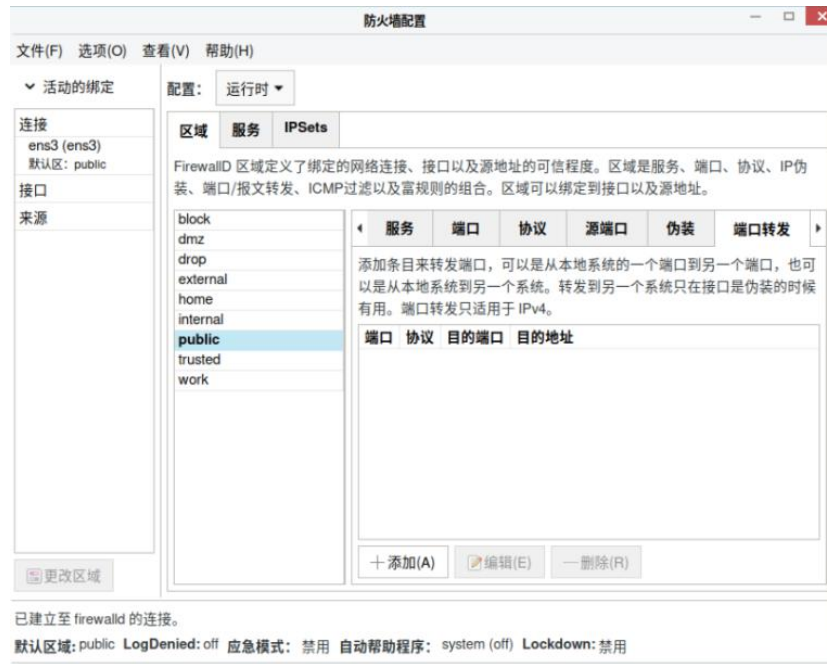


圖 8-5 端口轉發子選項卡

為一個特定端口轉發入站網路流量或“packets”到一個內部地址或者替代端口，首先啟動偽裝 IP 地址，然後選擇端口轉發標籤。在窗口靠上部分選擇入站流量協議和端口或者端口範圍。靠下部分是用於設置目的端口細節的。

要轉發流量到一個本地端口即同一系統上的端口，需選擇本地轉發複選框，輸入要轉發的流量的本地端口或者端口值範圍。要轉發流量到其他的 IPv4 地址，則選擇轉發到另一個端口複選框，輸入目的地 IP 地址和端口或者端口範圍。如果端口位置空缺則默認發送到同一個端口。點擊確定按鈕執行更改。

#### v. ICMP 篩檢程式子選項卡

ICMP 篩檢程式可以選擇 Internet 控制報文協議的報文。這些報文可以是資訊請求亦可是對資訊請求或錯誤條件創建的回應；

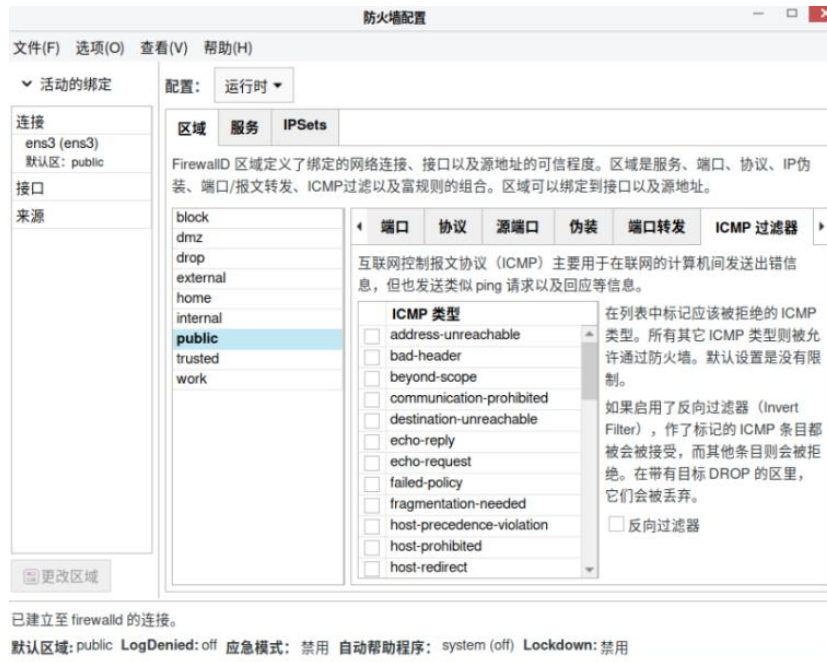


圖 8-6 ICMP 篩檢程式子選項卡

要使用或者禁用一個 ICMP 過濾，則啟動 firewall-config 工具並選擇要過濾其資訊的網路區域。選擇 ICMP Filter 圖示並選擇每種您需要過濾的 ICMP 資訊類型的複選框。清除複選框以禁用過濾。這種設定是單向的，默認允許全部。

#### vi. 直接配置選項卡

直接配置選項卡包括三個子選項卡：鏈、規則和穿通。說明以下這個選項卡主要用於服務或者應用程式增加特定的防火牆規則。這些規則並非永久有效，並且在收到 firewalld 通過 D-Bus 傳遞的啟動、重啟、重載信號後需要重新應用。



圖 8-7 直接配置選項卡

#### 4) 改變防火牆設置

要立刻改變現在的防火牆設置，須確定當前視圖設定在運行時或者從下拉菜單中選擇永久（Permanent），編輯下次啟動系統或者防火牆重新加載時執行的設定。

在運行時（Runtime）模式下更改防火牆的設定時，一旦您啟動或者清除連接伺服器的複選框，選擇立即生效。在 Permanent 模式下更改防火牆的設定，僅僅在重新加載防火牆或者系統重啟之後生效。可以使用檔菜單下的重新加載圖示，或者點擊選項菜單，選擇重新加載防火牆。



圖 8-8 改變防火牆設置

### 5) 修改默認分區

要設定一個將要被分配新介面的分區作為默認值，則啟動 `firewall-config`，從菜單欄選擇選項卡，由下拉菜單中選擇修改默認區域，從給出的列表中選擇您需要用的分區作為默認分區，點擊確定按鈕即可。

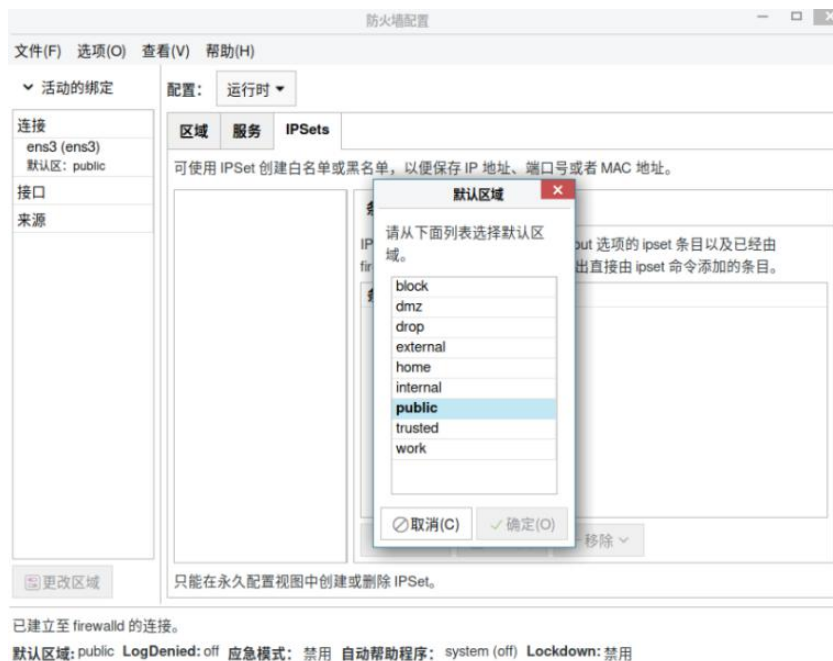


圖 8-9 修改默認分區

### 8.1.2. 審計管理(audit)

#### 8.1.2.1. 簡介

audit 工具可以將審計記錄寫入日誌檔。包括記錄系統調用和文件訪問。管理員可以檢查這些日誌，確定是否存在安全漏洞（如多次失敗的登錄嘗試，或者用戶對系統檔失敗訪問）。

#### 8.1.2.2. 架構及原理分析

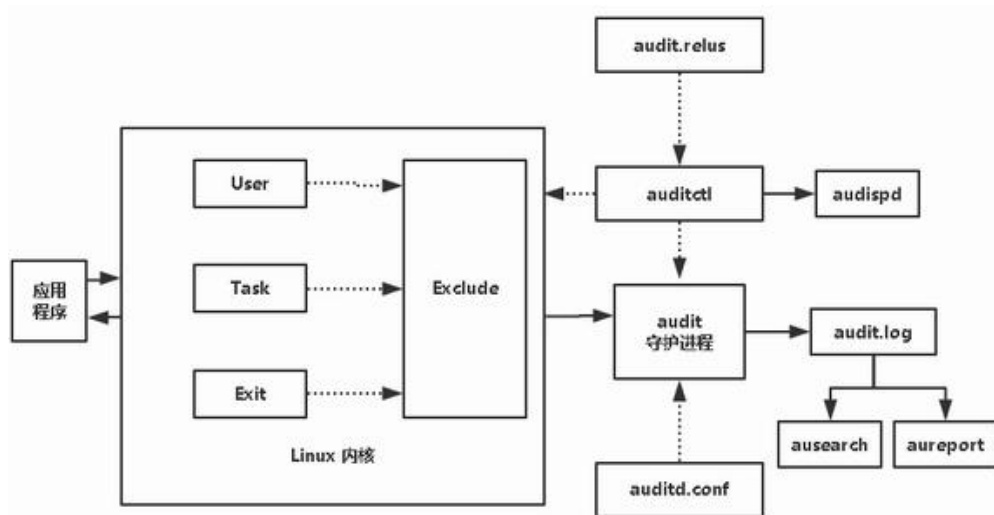


圖 8-10 架構示意圖

說明：實線代表數據流，虛線代表組件之間的控制關係。包括有兩大部分：中間的是內核中的幾種系統調用（user,task,exit,exclude），右側是一系列應用程式（auditd、audispd、auditctl、autrace、ausearch 和 aureport 等）。

Linux 內核中的幾種系統調用是：

**User:**記錄用戶空間中產生的事件；它的作用是過濾消息的，內核傳遞給審計後臺進程之前先查詢它。

**Task:** 跟蹤應用程式的子進程（fork）；當一個任務被創建時，也就是父進程通過 fork 和克隆創建子進程時記錄該事件；

**Exit:** 當一個系統調用結束時判斷是否記錄該調用；

**Exclude:** 刪除不合格事件；Exclude 是用來過濾消息的，也就是不想看到的消息可以在這裏寫規則進行過濾。

audit 是內核中的一個模組，內核的運行情況都會在 audit 中記錄，這個記錄的規則是由超級用戶來設置的。內核的 audit 模組是由應用層的一個應用程式 auditd 來控制的。audit 產生的數據都會傳送到 auditd 中，然後再由 auditd 進行其他操作。auditd.conf 是 auditd 的配置檔。audit.rules 是 audit 的規則檔，確定 audit 的日誌中記錄哪些操作。它通過一個對 audit 進行控制的應用程式 auditctl 進行操作。root 用戶也可以直接調用 auditctl 進行操作。auditd 收到的數據後會有兩個去處。默認的是將日誌保存在 audit.log 檔中，默認路徑 /var/log/audit/audit.log。另一個通過 audispd 將日誌進行分發。

要使用安全審計系統可採用下麵的步驟：首先安裝軟體包。然後設置配置檔、配置常用命令，添加審計規則，然後啟用 audit 守護進程並開始進行日誌記錄，最後通過生成審計報表和搜索日誌來週期性地分析數據。

用戶空間審計系統由 auditd、audispd、auditctl、autrace、ausearch 和 aureport 等應用程式組成。下麵依次說明：

**auditctl:**即時控制審計守護進程的行為的工具，如添加規則等。

**auditd:** audit 守護進程負責把內核產生的資訊寫入到硬碟上，這些資訊由應用程式和系統活動觸發產生。用戶空間審計系統通過 auditd 後臺進程接收內

核審計系統傳送來的審計資訊，將資訊寫入到`/var/log/audit/audit.log`。

`aureport`:查看和生成審計報告的工具。

`ausearch`:查找審計事件的工具

`auditspd`:轉發事件通知給其他應用程式，而不是寫入到審計日誌檔中。

`autrace`:一個用於跟蹤進程的命令。類似於 `strace`，跟蹤某一個進程，並將跟蹤的結果寫入日誌檔之中。

### 8.1.2.3. 安裝及配置

#### 1) 使用 dnf 工具安裝軟體包

```
dnf install audit -y
```

#### 2) audit 配置檔

`audit` 安裝後會生成 2 個配置檔：`/etc/audit/auditd.conf` 和 `/etc/audit/audit.rules`。`/etc/audit/auditd.conf` 是守護程式的默認配置檔。`/etc/audit/audit.rules` 是記錄審計規則的檔。首次安裝 `audit` 後，審計規則檔是空的。

`/etc/audit/auditd.conf` 守護程式的默認配置檔是其關鍵檔。

下面簡單設置一個 `/etc/audit/auditd.conf`

```
#vi /etc/audit/auditd.conf  
  
#設置日誌檔  
  
log_file = /var/log/audit/audit.log  
  
#設置日誌檔輪詢的數目，它是 0~99 之間的數。如果設置為小於 2，則不會迴圈日誌。如果沒有設置 num_logs 值，它就默認為 0，意味著從來不迴圈
```



日誌檔

```

num_logs = 5
#設置日誌檔是否使用主機名稱

name_format = NONE
#設置日誌檔大小,以兆位元組表示的最大日誌檔容量。當達到這個容量時,
會執行 max_log_file_action 指定的動作

max_log_file = 6
#設置日誌檔到達最大值後的動作,這裏選擇 ROTATE (輪詢)

max_log_file_action = ROTATE
    
```

8.1.2.4. auditctl 命令簡介

auditctl 命令格式如下：

```
auditctl [選項] filter,action -S syscall -F condition -k label
```

表 8-1 auditctl 命令選項

專案	可選參數	說明
filter	user,exit,task,exclude	filter 詳細說明哪個內核規則匹配篩檢程式應用在事件中。篩檢程式:task、exit、user 以及 exclude
action	always,never	是否審核事件 (always 表示是) (never 表示否)
syscall	all,2,open 等	所有的系統調用都可以在 /usr/include/asm/unistd_64.h 檔中找到。

condition	euclid=0,arch=b64	詳細說明其他選項，進一步修改規則來與特定架構、組 ID、進程 ID 和其他內容為基礎的事件相匹配
label	任意文字	標記審核事件並檢索日誌

-S 表示系統調用號或名字；

-F 表示規則域；

-k 表示設置審核規則上的過濾關鍵。

#### 8.1.2.5. audit 審核規則

audit 審核規則分成三個部分：

1) 控制規則：這些規則用於更改審核系統本身的配置和設置。

控制規則可以在 `/etc/audit/audit.rules` 中設置。主要包括：

`-D#`刪除所有當前裝載的審核規則#

`-b 8192#`在內核中設定最大數量的已存在的審核緩衝區為 8Mb#

`-e 2 #`鎖定審核配置#

2) 檔系統規則：這些是檔或目錄監視。使用這些規則，我們可以審核對

特定檔或目錄的任何類型的訪問。

可以通過 `auditctl` 命令設置。監控檔系統行為（依靠檔、目錄的許可權屬性來識別）

規則格式：

`-w` 路徑；

`-p` 許可權；

-k 關鍵字；

其中-p 許可權的動作分為四種：

r — 讀取檔或者目錄；

w — 寫入檔或者目錄；

x — 運行檔或者目錄；

a — 改變在檔或者目錄中的屬性。

例如要監控/etc/passwd 檔的修改行為，可以使用這個命令：

```
auditctl -w /etc/passwd -p wa
```

也可以自己將上述內容加入到檔/etc/audit/rules.d/audit.rules 中即可實現對該檔的監視。

下麵審核規則記錄了每次讀取或者修改/etc/hosts 檔的嘗試

```
auditctl -w /etc/hosts -p wa -k hosts_change
```

3) 系統調用規則：這些規則用於監視由任何進程或特定用戶進行的系統調用。

監控系統調用可能會引起高負荷的日誌活動，這會讓內核承受更大的負荷。所以要慎重衡量哪些系統調用需要放到 audit.rules 中。如果審計的是目錄的話，只能對該目錄本身的屬性進行審計。如果想審計下麵的檔，需要一一列出。

系統調用的監控：

-a 添加一條系統調用監控規則；

-S 顯示需要監測的系統調用的名稱。

顯示規則和刪除規則：

- D 刪除所有規則；
- d 刪除一條規則和-a 對應；
- w 寫入檔或者目錄；
- W 刪除一條規則和-w 對應；
- l 列出所有規則。

舉例:定義記錄有哪些檔，特定用戶（UID 為 10001）訪問和標籤的日誌條

目的規則：

```
auditctl -a always,exit -F arch=b64 -F auid=10001 -S  
open -k userfile
```

說明：**userfile** 是用戶自己設置的一個規則名字，以上的都設置完畢了，就可以生成報告了。另外通過 **auditctl** 命令添加的規則不是永久有效的。為了讓他們在重新啟動後有效的，可以將其添加到檔 **/etc/audit/rules.d/audit.rules** 中。

#### 8.1.2.6. 守護進程

啟動：

```
systemctl start auditd
```

自動啟動：

```
systemctl enable auditd
```

停止：

```
service auditd stop
```

重啟：

```
service auditd restart
```

在/var/log/audit/目錄中旋轉日誌檔：

```
service auditd rotate
```

推遲審核事件日誌之後重新開始，例如存在沒有足夠的磁片分區空間來保存。

審核日誌檔情況：

```
service auditd resume
```

顯示運行狀態：

```
service auditd status
```

列出所有活動的規則和觀察器：

```
auditctl -l
```

配置 auditd 的日誌檔到遠程主機。

假設 audit 伺服器名稱是 kylin1,ip 地址 192.168.0.1;audit 客戶端名稱是 kylin2, ip 地址 192.168.0.2。

修改伺服器端配置檔：

```
vi /etc/audit/auditd.conf  
#設置監聽端口為 60  
tcp_listen_port =60  
  
重啟服務  
#service auditd restart
```

客戶端安裝相關軟體包並且配置檔：

```
#dnf -y install audispd-plugins  
修改配置檔
```

```
#vi /etc/audit/plugins.d/au-remote.conf
#把審核日誌發送到日誌伺服器
active =yes

#vi /etc/audit/audisp-remote.conf
remote_server =kylin1

#設置監聽端口為 60
port = 60

#vi /etc/audit/auditd.conf
log_format =NOLOG

然後重啟服務

#service auditd restart
```

#### 8.1.2.7. aureport

要生成審計消息的報表，可使用 `aureport` 命令。如果執行 `aureport` 時沒有使用任何選項，則會顯示如匯總報表。下麵是幾個例子：

生成一段特定時間內的報告：

```
aureport -ts 8:00 -te 17:30 -f -i
```

生成所有用戶失敗事件的總結報告：

```
aureport -u --failed --summary -i
```

生成系統調用事件報告：

```
aureport -s -i --summary
```

#### 8.1.2.8. ausearch

使用 `ausearch`，您可以過濾和搜索事件類型。它還可以通過將數值轉換

為更加直觀的值(如系統調用或用戶名)來解釋事件。以 root 用戶執行 `ausearch` 命令，當顯示結果時，每個記錄用 4 條虛線組成的一行隔開，每個記錄前均顯示時間標記。

示例：`ausearch -f /etc/passwd`

```
[root@localhost audit]# ausearch -f /etc/passwd | less
----
time->Fri Mar 20 11:54:34 2020
type=PROCTITLE msg=audit(1584676474.800:1468): proctitle="whoami"
type=PATH msg=audit(1584676474.800:1468): item=0 name="/etc/passwd" inode=1049428 dev=
=fd:00 mode=0100644 ouid=0 ogid=0 rdev=00:00 obj=system_u:object_r:passwd_file_t:s0 no
ametype=NORMAL cap_fp=0000000000000000 cap_fi=0000000000000000 cap_fe=0 cap_fver=0
type=CWD msg=audit(1584676474.800:1468): cwd="/root"
type=SYSCALL msg=audit(1584676474.800:1468): arch=c000003e syscall=257 success=yes ex
it=3 a0=ffffff9c a1=7f830dd3b15e a2=80000 a3=0 items=1 ppid=42011 pid=42536 auid=0 ui
d=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts0 ses=34 comm="whoami" e
xe="/usr/bin/whoami" subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 key="
passwd_changes"
----
time->Fri Mar 20 11:54:34 2020
type=PROCTITLE msg=audit(1584676474.803:1469): proctitle=6964002D757200726F6F74
type=PATH msg=audit(1584676474.803:1469): item=0 name="/etc/passwd" inode=1049428 dev=
=fd:00 mode=0100644 ouid=0 ogid=0 rdev=00:00 obj=system_u:object_r:passwd_file_t:s0 no
ametype=NORMAL cap_fp=0000000000000000 cap_fi=0000000000000000 cap_fe=0 cap_fver=0
type=CWD msg=audit(1584676474.803:1469): cwd="/root"
type=SYSCALL msg=audit(1584676474.803:1469): arch=c000003e syscall=257 success=yes ex
it=3 a0=ffffff9c a1=7f4b43e7615e a2=80000 a3=0 items=1 ppid=42011 pid=42537 auid=0 ui
d=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts0 ses=34 comm="id" exe="
/usr/bin/id" subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 key="passwd_c
hanges"
type=SYSCALL msg=audit(1584676474.803:1470): arch=c000003e syscall=257 success=yes ex
```

圖 8-11 輸出結果

輸出結果介紹：

`time`:審計時間；

`name`:審計對象；

`cwd`:當前路徑；

`syscall`:相關的系統調用；

`auid`:審計用戶 ID；

`uid` 和 `gid`:訪問檔的用戶 ID 和用戶組 ID；

`comm`:用戶訪問檔的命令；

**exe**:上面命令的可執行檔路徑。

下麵是幾個例子：

搜索系統登錄失敗,使用如下命令：

```
ausearch --message USER_LOGIN --success no -interpret
```

搜索所有的帳戶，群組，角色變更，使用以下命令：

```
ausearch -m ADD_USER -m DEL_USER -m ADD_GROUP -m  
USER_CHAUTHOK -m DEL_GROUP -m CHGRP_ID -m  
ROLE_ASSIGN -m ROLE_REMOVE -i
```

搜索從制定時間段的失敗的系統調用，使用以下命令：

```
ausearch --start yesterday --end now -m SYSCALL -sv no -i
```

使用關鍵字（**key**）搜索審計事件記錄。

**auditd** 生成的日誌檔內容比較多，如果沒有設置關鍵字，要查找搜索審計事件記錄,使用關鍵字（**key**）可以提高效率。

首先使用 **auditctl** 命令建立一個規則：

```
auditctl -w /etc/passwd -p rwa -k passwd_changes
```

上面這個規則的意思是記錄訪問或修改/etc/passwd 用戶帳戶數據庫的任何嘗試。一旦有人修改了資料庫檔，可以使用如下關鍵字（**key**）搜索審計事件記錄。

使用關鍵字（**key**）搜索審計事件記錄：

```
ausearch -k passwd_changes | less
```

上面的命令顯示訪問或修改/etc/passwd 檔的日誌資訊，其他資訊忽略。



## 8.2. 安全增強組件

在配置系統之外，掌握收集基本的系統資訊的方法也很重要。譬如，您應該知道如何找出空閒記憶體的数量、可用硬碟空間，硬碟分區方案，以及正在運行進程的資訊等等。本節將說明如何使用幾個簡單程式來從您的銀河麒麟伺服器操作系统中檢索這類資訊。

### 8.2.1. KYSEC 安全機制

銀河麒麟高級伺服器操作系统 V10 結合國產操作系统特點和現有國內外操作系统的安全機制，提出了基於標記的軟體執行控制機制，實現對系統應用程式標記識別和執行約束，確保應用來源的可靠性和應用本身的完整性。執行控制機制控制檔執行、模組加載和共用庫使用，分為系統檔、第三方應用程式，其中只允許具有合法標記的檔執行，任何網路下載、拷貝等外來軟體均被禁止執行。

#### 8.2.1.1. 安全狀態設置

KYSEC 共有兩種狀態；強制模式（normal）、軟模式（softmode）。強制模式下不能執行非法應用程式，即禁止用戶執行沒有合法標記的應用程式，而軟模式下，只是記錄非法應用程式的執行行為，不禁止用戶操作。系統默認 KYSEC 安全狀態為關閉狀態，且執行控制、檔保護、內核保護狀態都處於關閉狀態，三權分立關閉。

查看 kysec 安全狀態：

```
#getstatus
KySec status: Normal
exec control: on
```

```
file protect: on
kmod protect: on
three admin : off
```

設置 kysec 安全狀態為軟模式：

```
#sudo setstatus softmode
```

設置 kysec 安全狀態為強制模式：

```
#sudo setstatus normal
```

### 8.2.1.2. 執行控制

銀河麒麟高級伺服器操作系統 V10 添加了對檔執行、模組加載和共用庫使用的控制，分為系統檔、第三方應用程式，其中只允許具有合法標記的檔執行，任何網路下載、拷貝等外來軟體均被禁止執行。本來可運行的可執行檔在刪除、修改、拷貝、移動、重命名等操作後，會被禁止執行。

#### (1) 可執行檔的執行控制

如下情況的/tmp/ls 則執行失敗。

```
#cp /bin/ls /tmp
#/tmp/ls
```

但管理員設置/tmp/ls 的 kysec 標記後，用戶則可執行/tmp/ls 程式：

```
#sudo kysec_set -n exectl -v original /tmp/ls
#/tmp/ls
```

篡改/tmp/ls 程式內容後，該程式不能再執行：

```
#echo " " >> /tmp/ls
```

```
#!/tmp/ls
```

### (2) 可執行腳本的執行控制

創建腳本檔/tmp/test.sh 並添加執行許可權,內容如下:

```
#!/bin/bash  
echo "test"  
#chmod +x /tmp/test.sh
```

但用戶使用如下命令,都無法執行程式:

```
#!/tmp/test.sh  
#bash /tmp/test.sh
```

管理員用戶設置該檔的 kysec 標記後,程式就能正常執行。

```
#sudo kysec_set -n exectl -v original /tmp/test.sh
```

### (3) 內核模組的執行控制

用戶將可加載的內核模組複製到主目錄,再嘗試加載拷貝的內核模組,因為 KYSEC 功能,則加載失敗:

```
#cp /lib/modules/`uname  
-r`/kernel/net/netfilter/nf_conntrack_ftp.ko ~  
#sudo insmod ~/nf_conntrack_ftp.ko
```

修改拷貝的內核模組標記,則可以加載成功。

```
#sudo kysec_set -n exectl -v original ~/nf_conntrack_ftp.ko  
#sudo insmod ~/nf_conntrack_ftp.ko
```

### 8.2.1.3. 檔保護功能

檔保護功能開啟時，受保護的檔不能被修改、重命名、刪除。這樣能保護系統重要的配置檔不被誤操作修改。

給檔設置檔保護的標記：

```
#sudo kysec_set -n protect -v readonly /tmp/testfile
```

給檔移除檔保護的標記：

```
#sudo kysec_set -xn protect /tmp/testfile
```

給檔 /tmp/testfile 設置檔保護的標記後，則禁止修改、重命名、刪除 /tmp/testfile 檔。如下操作則會提示失敗。

```
#echo "test1" >> /tmp/testfile  
#mv /tmp/testfile /tmp/abc  
#rm -f /tmp/testfile
```

### 8.2.2. 數據隔離保護機制

銀河麒麟高級伺服器操作系統 V10 實現用戶私有數據的機密性隔離存儲；並基於該保護機制，研製了檔保護箱功能，確保每個用戶的檔保護箱相互隔離，即使非法用戶獲得管理員許可權也無法獲取其他用戶的私有數據；同時也提供檔保護箱的數據共用機制。

- 隔離隱藏，用戶私有數據僅對自己或共用的系統用戶可見，對沒有授權的用戶實現隱藏隔離；
- 數據共用，每個用戶均可對自己的數據進行共用配置，可設置賦予其他用戶讀、寫許可權，實現數據共用。

### 8.2.2.1. 隔離隱藏

數據隔離保護機制的操作目錄為/box。用戶使用 `boxadm` 創建一個麒麟檔保護箱，其中的檔、目錄等就不能被別的用戶訪問。

`test` 用戶創建一個檔保護箱 `testbox`，並新建一個檔：

```
#boxadm -c testbox
#touch /box/test/testbox/testfile
```

`test` 用戶查看/box/test 的目錄及檔：

```
#ls -l /box/test
testbox
#ls -l /box/test/testbox
testfile
```

但 `root` 用戶查看/box 下的子目錄，如：

(1) `ls -l /box/test` 命令輸出中沒有/box/test/testbox，提示“總用量 0”；

(2) `ls -ld /box/test/testbox` 命令提示“No such file or directory”；

(3) `ls -l /box/test/testbox/testfile` 命令提示” No such file or directory”

```
#ls -l /box/test
#ls -ld /box/test/testbox
#ls -l /box/test/testbox/testfile
```

### 8.2.2.2. 數據共用

用戶使用 `boxadm` 設置檔保護箱中的目錄共用許可權，如只讀共用許可

權、讀寫共用許可權，被授權的用戶就能訪問這個目錄。

test 用戶設置 root 對/box/test/testbox 目錄的只讀共用許可權後，root 用戶才能訪問/box/test/testbox 目錄：

```
#boxadm -s -u root -p rx testbox
```

root 用戶執行如下命令，則可以看到/box/test 目錄下存在 testbox 目錄、/box/test/testbox 目錄下存在 testfile 檔。

```
#ls -l /box/test/  
#ls -l /box/test/testbox
```

### 8.2.2.3. boxadm 的使用方法

test 用戶創建一個麒麟檔保護箱 testbox，會生成目錄/box/test/testbox：

```
#boxadm -c testbox
```

設置 root 對/box/test/testbox 目錄的只讀共用許可權後，root 用戶才能訪問/box/test/testbox 目錄：

```
#boxadm -s -u root -p rx testbox
```

設置 root 對/box/test/testbox 目錄的讀寫共用許可權後，root 用戶才能讀寫/box/test/testbox 目錄：

```
#boxadm -s -u root -p rwx testbox
```

查看/box/test/testbox 目錄的 Box 資訊：

```
#boxadm -i testbox
```

```
Access:user:root:rwx
```

移除 root 用戶對/box/test/testbox 目錄的共用許可權：

```
#boxadm -s -u root -X testbox
```

刪除/box/test/testbox 目錄：

```
#boxadm -r testbox
```

### 8.2.3. 強制訪問控制

在電腦安全領域指一種由操作系統約束的訪問控制，目標是限制主體或發起者訪問或對對象或目標執行某種操作的能力。在實踐中，主體通常是一個進程或線程，對象可能是檔、目錄、TCP/UDP 端口、共用記憶體段、I/O 設備等。主體和對象各自具有一組安全屬性。每當主體嘗試訪問對象時，都會由操作系統內核強制施行授權規則——檢查安全屬性並決定是否可進行訪問。任何主體對任何客體的任何操作都將根據一組授權規則（也稱策略）進行測試，決定操作是否允許。

**主體和客體：**主體指的是訪問者（一般為一個進程），客體指的是被訪問的資源。根據資源的類型不同，在策略配置語言中被分為不同的客體，如 **dir**、**file**、**process** 等。

**安全上下文：**指標記在所有事物上的擴展屬性，包括 SELinux 用戶、角色、域或類型，如果使用 **MLS**（多級安全，以下會有介紹）策略還包括安全級。

**類型和域：**是分配給一個對象並決定誰可以訪問這個對象，域對應進程，類型對應其他對象。

**域轉換：**進程以給定的進程類型運行的能力稱為域轉換，需滿足三個轉換條

件：一個進程新的域類型有對一個可執行檔類型的 **entrypoint** 訪問許可權，進程的當前（或以前）域類型對入口點檔類型擁有 **execute** 訪問許可權，進程的當前域類型對新的域類型擁有 **transition** 訪問許可權。

角色：和一些域相關聯，決定哪些域可以使用。

安全策略用戶：和標準 **linux** 用戶對應，影響哪個域可以進入。

### 8.2.3.1. SELinux 基礎策略

銀河麒麟高級服務器操作系統 V10 根據系統安全需求，定制了系統圖形登錄功能策略、三權分立功能策略、審計服務策略、執行控制功能策略、白名單功能策略、kvm/lxc 等系統功能策略、系統使用修訂桌面常用工具策略、系統啟動時自動標記腳本功能。根據不同應用場景，定制了 **ukmls**、**ukmcs** 和 **target** 三套 SE 基礎策略。

### 8.2.3.2. 安全策略模式切換

系統在正常啟動後就進入了允許模式。允許模式是為了方便我們服務器配置，即使我們的安全策略沒有允許這樣的操作仍可以執行，但會被審計下來；強制模式是只要沒有這樣的規則操作就不被允許。我們的安全服務器系統在普通模式下使用的就是允許模式的策略，在默認模式下則使用強制模式的策略來加固我們服務器的安全。

#### （1）當前安全策略模式查詢

我們提供了一些工具命令可以查詢當前的安全策略模式，如 **/usr/sbin/getenforce** 和 **/usr/sbin/sestatus**。所有用戶都可以通過這些命令進行查詢。



getenforce 命令輸出若為 Permissive 則表示當前處於允許模式，若為 Enforcing 則表示當前處於強制模式。

sestatus 命令的輸出可能為以下內容：

```
#sestatus
SELinux status:                enabled
SELinuxfs mount:              /sys/fs/selinux
SELinux root directory:       /etc/selinux
Loaded policy name:           ukmcs
Current mode:                  permissive
Mode from config file:        enforcing
Policy MLS status:           enabled
Policy deny_unknown status:   allowed
Memory protection checking:   actual (secure)
Max kernel policy version:    30
```

它不僅反應了當前安全策略的模式，同時也可以得到其他安全策略的資訊：安全策略開啟狀態、安全策略檔系統掛載位置、配置檔中的默認模式、安全策略版本和安全策略目錄名稱。

## （2）安全策略模式修改

為了安全考慮所以系統一啟動都會進入到強制模式，但是也提供了可以修改當前安全策略模式的命令。

**setenforce 1** 設置為強制模式；

**setenforce 0** 設置為允許模式。

系統三個管理員中只有安全管理員可以使用 **setenforce 1** 進入到強制模式，

也只有安全管理員可以使用 `setenforce 0` 進入到允許模式。

### (3) 禁用安全策略

在系統出現故障時我們也可以先禁用我們的安全策略進入到維護模式進行修復。進入方法如下：

a) 系統啟動時進入 `grub` 菜單。

b) 輸入 `grub` 密碼修改內核啟動參數，將參數 `selinux=1 enforcing=1` 修改為 `selinux=1 enforcing=0`。

### (4) 啟動系統。

這樣系統就直接進入到維護模式，由 `root` 用戶對系統進行修復。

#### 8.2.3.3. 策略配置命令

為了方便用戶對安全策略進行配置，我們也提供了以下命令對當前安全策略進行配置：

##### 8.2.3.3.1. 查看標記命令

系統中一些常用命令的 `-Z` 選項都提供了查看檔或進程安全上下文的功能，如：`ls -Z id -Z` 和 `ps -Z`，這些命令加了 `-Z` 選項後就會顯示該檔或進程的安全上下文，如 `ls -Z` 一個檔，輸出以下內容：

```
-rw-r--r--. root root sysadm_u:object_r:admin_home_t:s0
```

其中 `sysadm_u` 就表示該檔屬於 `sysadm_u` 安全策略用戶，`object_r` 表示該檔屬於客體類型的角色，`admin_home_t` 表示該檔的類型，我們的策略就是根據這些標記來制定規則進行訪問控制的。

### 8.2.3.3.2. 修改標記命令

安全管理員可以通過一些命令對系統中所有檔的安全上下文進行操作，如：  
chcon、setfiles 和 restorecon 命令。

/bin/lis 命令為 bin\_t 類型，安全管理員可以輸入 chcon -t sbin\_t /bin/lis 將 bin\_t 修改為 sbin\_t 類型。

chcon 命令的 -u 參數可以修改該檔安全上下文中的安全策略用戶，-r 參數可以修改它的角色，-l 參數可以修改它的安全級別，其他具體使用可以查看它的幫助手冊。

setfiles 命令可以根據我們的安全策略的標記檔對系統中所有檔進行標記。  
如：

```
setfiles /etc/selinux/targeted/contexts/files/file_contexts /
```

restorecon 命令可以恢復該檔的默認安全上下文，如：restorecon -r filename。

在對檔的安全上下文標記進行操作時首先都會檢查這個標記的正確性，然後再去進行操作，具體使用可以查看用戶手冊。

### 8.2.3.3.3. 用戶、角色和域的關係

登錄程式如 (login, sshd) 負責映射 Linux 用戶到安全策略用戶，在登錄時，如果安全策略用戶識別字恰好和一個 Linux 用戶識別字完全相同，匹配的安全策略用戶識別字成為初始 shell 進程安全上下文的用戶識別字。安全管理員也可使用 semanage 命令完成映射，如：semanage login -a -s charlie\_u charlie，表示 Linux 用戶 charlie 作為安全策略用戶 charlie\_u 登錄系統。

使用 `semanage login -l` 命令可以查看，如下圖：

```
#semanage login -l
```

登錄名	SELinux 用戶	MLS/MCS 範圍	服務
__default__	user_u	s0-s0:c0.c1023	*
auditadm	auditadm_u	s0-s0:c0.c1023	*
root	root	s0-s0:c0.c1023	*
secadm	secadm_u	s0-s0:c0.c1023	*

一個用戶可以對應多個角色，但在同一時刻只能作為其中一個角色，可以通過策略管理工具 `semanage` 輸入 `semanage user -a -R mgr_r -R charlie_r -P user charlie_u` 來創建一個可以作為角色 `mgr_r` 和 `charlie_r` 的安全策略用戶 `charlie_u`。

可通過 `semanage user -l` 進行查看，如下圖：

```
#semanage user -l
```

SELinux 用戶	標記中 首碼	MLS/ MCS 級別	MLS/ MCS 範圍	SELinux 角色
auditadm_u	audadm	s0	s0-s0:c0.c1023	auditadm_r
guest_u	user	s0	s0	guest_r
root	sysadm	s0	s0-s0:c0.c1023	sysadm_r
secadm_u	secadm	s0	s0-s0:c0.c1023	secadm_r

角色僅僅是一套域類型的集合，方便與用戶建立聯繫，具體的訪問控制都是通過進程的類型和客體的類型根據制定的規則來進行控制。

#### 8.2.3.3.4. 端口配置

系統中所有端口同樣也被標記了類型，安全管理員可以通過：

```
semanage port -a -t vnc_port_t -p udp 222
```

來添加一個標記了類型的端口。

可通過 `semanage port -l` 進行查看，如下圖：

```
#semanage port -l
SELinux 端口類型          協議      端口號
afs3_callback_port_t    tcp       7001
afs3_callback_port_t    udp       7001
afs_bos_port_t          udp       7007
afs_fs_port_t           tcp       2040
afs_fs_port_t           udp       7000, 7005
afs_ka_port_t           udp       7004
```

#### 8.2.4. 三權分立機制

三權分立系統在滿足安全系統標準要求的同時，儘量減少對系統的改動，大大的提高了系統的可用性和引用性。

主要設計思想是，將傳統意義上超級管理員 `root` 的權能進行劃分，分別為 `root(uid=0)`、`secadm(uid=600)`和 `auditadm(uid=700)`分別作為超級用戶的別名存在，它們分別對應 `selinux` 三權分立角色中的一個角色，從而保持傳統系統用戶身份鑒別系統結構的同時，達到三權分立的目的。

三權分立系統是對 `linux` 現有用戶身份認證機制的擴展，增加部分功能主要是為解決傳統認證機制無法支持三權分立的用戶身份的鑒別。三權系統使用傳統 `linux` 用戶身份鑒別機制的認證系統之外可以額外採用雙因數驗證方式進行身份

鑒別，這套系統是專門定制開發，且符合公安部安全操作系統相關標準。

#### 8.2.4.1. 功能

三權系統功能包括兩部分：

1、實現對從一個 UID 為 0 的超級用戶衍生出的三個管理員用戶的身份認證功能，這個認證系統要支持包括登錄系統和用戶身份切換等所有可能涉及身份許可權切換的情況，同時本三權系統對其他非管理員用戶是透明的。

2、實現在每次切換管理員身份時，可以按照不同管理員指派不同的 SELinux 角色給用戶。

上面兩個功能實現的前提是，本三權系統不能影響到系統中傳統身份鑒別系統對非管理員用戶的所有驗證操作，即新三權系統只對三個管理員用戶有作用，對任何其他用戶是透明的。

#### 8.2.4.2. 示例

只有系統管理員具有“分區管理功能”。該功能使系統中只有系統管理員可對系統磁片進行分區管理。當三權系統的安全狀態為 strict 模式時。系統管理員可在圖形介面下，點擊應用程式->工具->磁片。系統管理員可以成功打開分區編輯器。安全管理員、審計管理員的開始菜單中沒有分區編輯器的圖示顯示。

#### 8.2.4.3. 三權用戶命令集

表 8-2 三權用戶命令集

用戶	命令	備註
系統	reboot	系統重啟命令。
管理	init	管理系統運行模式。

員	ifconfig	網路介面管理。
	rpm	軟體包管理（但不能刪除我們的策略包）。
	useradd	添加用戶。如：useradd abc
	passwd	修改普通用戶密碼。如：passwd abc
	userdel	刪除用戶。如：userdel -r abc
	insmod	插入模組。
	rmmod	刪除模組。
	iptables	防火牆相關。
	system-config-date	配置系統時間（配置網路時間用到 ntpd 服務需要在允許模式下進行）。
	system-config-keyboard	鍵盤配置。
	system-config-language	語言配置。
	system-config-network system-config-network- cmd system-config-network- tui	網路配置命令。
	system-config-printer system-config-printer-a pplet	印表機配置。
	setup	整體配置（其中有關服務配置需要切換到允許模式下進行）。
	mount	分區掛載。

	<b>baobab</b>	磁片分析工具。
安全 管理 員	<b>setenforce</b>	開啟和關閉 <b>selinux</b> 的強制模式  如： <b>setenforce 1</b> 開啟強制模式 (其他管理員都可執行)；  <b>setenforce 0</b> 關閉強制模式 (只有安全管理員才能執行)。
	<b>checkpolicy</b>	將可讀策略檔編譯生成二進位策略檔，該檔和所在檔夾必須為 <b>policy_src_t</b> 類型。  如： <b>checkpolicy -M policy.conf -o policy.24</b>
	<b>load_policy</b>	如： <b>load_policy -bq(伺服器)</b> 更換內核中的安全策略，保持使用當前的 <b>Boolean</b> 值。
	<b>chcon</b>	修改檔和文件夾的安全上下文。如： <b>chcon -t bin_t filename</b>
	<b>chcat</b>	修改檔和文件夾的敏感級。如： <b>chcat s0:c0.c255 filename</b>
	<b>fixfiles</b>	檢查修復檔安全上下文。如： <b>fixfiles -F restore filename</b>
	<b>setfiles</b>	初始化檔標記，檢查標記正確性。  如： <b>setfiles /etc/selinux/targeted/contexts/files/f</b>



		ile_contexts /
	restorecon	恢復默認檔安全上下文。 如： restorecon -r filename
	semanage	SELinux 管理工具命令 如： semanage user -l semanage login -l semanage user -m -R secadm_r secadm_u
	semodule	管理 SELinux 策略模組。 如： semodule -l semodule -i modulename
	setsebool	修改 SELinux 中的 bool 值來修改策略。 如： setsebool xdm_sysadm_login = 1
審計 管理 員	audit2allow	顯示訪問被拒絕後應添加的允許規則。如： audit2allow -a audit2allow -i /var/log/dmesg
	audit2why	根據訪問被拒絕的審計資訊顯示被拒絕的原因。 如 : audit2why < /var/log/audit/audit.log
	aureport	根據審計資訊檔統計登錄情況, avc, pid, file, event, config 等眾多資訊。 如： aureport -au

		<pre>aureport -a aureport -p</pre>
	ausearch	<p>搜索審計檔中的資訊。</p> <p>如：ausearch -gi 0</p>
	auditd	<p>啟動審計服務。</p> <p>如：auditd -s enable</p>
	auditctl	<p>顯示服務狀態，審計規則相關。</p> <p>如：auditctl -s</p>
	audispd	<p>審計調度器。</p> <p>如：audispd</p>

#### 8.2.4.4. 啟用與關閉

採用如下方式開啟三權分立安全功能：

- 1) 執行下述命令開啟三權分立安全功能：

```
security-switch --set strict
```

- 2) 重啟系統，使得修改生效

3) 通過 `security-switch --get` 命令查看當前修改後的增強安全狀態中“三權分立”為啟用，命令如下：

```
security-switch --get
```

顯示當前安全級別資訊：

```
當前安全級別:strict
-----
```

安全模組	當前狀態	默認狀態
SELinux	啟用(Enforcing)	啟用(Enforcing)
三權分立	啟用	啟用
執行控制	啟用(Normal)	啟用(Normal)

採用切換其他安全模式可以關閉三權分立安全功能：

```
security-switch --set default
```

### 8.2.5. 核外安全功能及配置

#### 8.2.5.1. 用戶 UID 唯一性

操作系統中用戶名和用戶標識（下稱 UID）是存在對應關係，一般介面調用中也是通過 UID 判斷對應用戶，通常情況下 UID 在用戶創建時會自動分配，也可以在創建用戶時手動設置未被使用的 UID，該 UID 可以是已刪除用戶使用過的。而本功能提供對用戶 UID 的唯一性檢查，確保在整個操作系統生命週期內用戶 UID 數字僅且只能使用一次。

若系統已經存在 kylin\_1 用戶，且該用戶下的 UID（可通過命令：`id kylin_1` 查看 kylin\_1 用戶的 uid）為 kylin\_1-UID，進行如下操作：

```
1.root 刪除用戶 kylin_1
#userdel kylin_1

2.root 創建 kylin_2 用戶且指定 uid 為 kylin_1-UID
#adduser kylin_2 -u kylin_1-UID
```

若沒有用戶 UID 唯一性檢查，則添加 kylin\_2 用戶操作可以成功；該功能

使用後步驟 2 中的操作將失敗，提示：“adduser：無效的用戶 ID”異常資訊。

### 8.2.5.2. 安全切換工具

`security-switch` 是系統下提供的一個用於安全配置的工具，通過該工具可進行如下 3 種安全模式的切換：

- **default** 模式：啟用系統 kysec 安全機制；
- **strict** 模式：啟用 kysec、selinux、三權分立，且 selinux 為 ukmcs 策略；
- **custom** 模式：用戶自定義安全模式。

#### 8.2.5.2.1. default 模式

##### 設置 **default** 模式

1.kylin 登錄系統，執行：

```
$sudo security-switch --set default
```

2.重啟系統

模式狀態：

安全模組	當前狀態	默認狀態
執行控制	用(Normal)	啟用(Normal)

#### 8.2.5.2.2. strict 模式

##### 設置 **strict** 模式

1.kylin 登錄系統，執行：

```
#sudo security-switch --set strict
```

（然後分別設置 root,secadm,auditadm 三個管理員用戶的密碼）

2.重啟系統

### 模式狀態

安全模組	當前狀態	默認狀態
SELinux	啟用(Enforcing)	啟用(Enforcing)
三權分立	啟用	啟用
執行控制	啟用(Normal)	啟用(Normal)

#### 8.2.5.2.3. 自定義模式

##### 設置自定義模式

1.kylin 登錄系統，設置自定義安全機制，僅啟用 selinux 安全機制：

```
#sudo security-switch --set custom -list selinux
```

2.重啟系統；

### 模式狀態

安全模組	當前狀態	默認狀態
SELinux	啟用(Enforcing)	啟用(Enforcing)

#### 8.2.5.3. 審計系統許可權管理

基於基礎安全審計系統功能，麒麟操作系統根據三權分立要求，實現管理員

分權機制，修訂只允許審計管理員具有審計服務管理權限和審計規則修改許可權。根據其系統安全機制要求，增加審計服務容錯機制，當審計服務出錯時，審計服務將自動重啟，確保審計服務正常運行；移除 SU 到審計管理員的許可權，限制系統管理員通過 SU 命令執行 auditctl 等審計管理工具。同時，添加了不同等級的審計規則、修復在啟用或不啟用三權分立時對用戶的判斷等功能。

審計系統許可權需要在安全狀態為 strict 模式時相應功能才會開啟，另外要確保系統的審計服務處於開啟狀態（systemctl status auditd）。

#### 8.2.5.3.1. 登錄審計

審計用戶登錄時，無論成功與否，均會在審計日誌中有資訊輸出。審計日誌包括事件類型、時間、用戶、事件成功與否、身份鑒別請求的源等，如：

```
Authentication Report
=====
=====
#date time acct host term exe success event
=====
=====
2020 年 02 月 09 日 13:21:21 test localhost localdomain
/dev/tty1 /usr/libexec/gdm-session-worker yes 309
```

#### 8.2.5.3.2. 行為審計

添加用戶審計規則後，用戶的行為會被系統審計。

示例（如對 uid 為 600 的用戶添加行為審計規則）：

1. 審計管理員添加審計規則，對 uid=600 的用戶行為進行審計：

```
#auditctl -a exit,always -S all -F uid=600
```

2.安全管理員執行操作：

```
#cat /etc/uid_list
```

3.審計管理員以 uid 進行審計資訊查看

```
#ausearch -ui 600 | grep uid_list
```

此時可在在審計日誌中查看到針對步驟 2 的審計資訊。

#### 8.2.5.3.3. 帳號管理審計

所有用戶帳號的增加、刪除（`useradd`、`passwd`、`userdel`）等行為都會被系統審計。

示例如下，在審計用戶下通過`$auseport -m` 可以查看到用戶帳戶操作的審計日誌。

1.root 創建一個用戶：

```
#useradd testuser
```

2.root 修改用戶密碼：

```
#passwd testuser
```

3.root 刪除用戶

```
#userdel testuser
```

#### 8.2.5.3.4. 檔、目錄監視審計

添加檔、目錄監視規則後，任何對檔、目錄的操作都會被系統審計。

示例如下：

存在 kylin 用戶創建的/tmp/file1 檔及/tmp/test.dir 目錄：

1. 審計管理員登錄系統，添加審計規則：

```
$auditctl -w /tmp/file1 -k file
```

```
$auditctl -w /tmp/test.dir -k test_dir
```

2. kylin 用戶 cat 查看/tmp/file1 檔

3. kylin 用戶使用 rm 刪除/tmp/file1 檔

4. kylin 用戶對/tmp/test.dir 進行讀操作

5. kylin 用戶對/tmp/test.dir 進行寫操作

6. 審計管理員登錄系統，刪除步驟 1 中審計規則：

```
$auditctl -W /tmp/file1 -k file
```

```
$auditctl -W /tmp/test.dir -k test_dir
```

步驟 1 添加規則後，步驟 2、3、4、5 等對於檔及目錄的操作均會在審計日誌中體現；步驟 6 刪除規則後不再記錄相關操作審計日誌。

#### 8.2.5.3.5. 規則配置

添加排除規則後可以排除指定的審計資訊。

#### 8.2.5.3.6. 審計日誌保護

只有審計管理員才可以查看審計日誌。

#### 8.2.5.3.7. 審計日誌轉儲

當審計日誌達到設置的最大值時，會覆蓋所存儲的最早的審計記錄。該部分需要審計管理員對配置檔/etc/audit/auditd.conf 進行手動編輯。

示例：

1. 審計管理員編輯/etc/audit/auditd.conf 檔，使 max\_log\_file=1，



```
num_logs = 3 即日志檔最大為 1MB，備份檔數為 3
```

## 2. 審計管理員重啟服務

```
#systemctl restart auditd
```

## 3. 審計管理員添加審計規則，

```
$auditctl -a exit,always -S all -Fsys
```

## 4. 安全管理員隨便執行任何操作，以便產生大量日誌

```
$find /
```

## 5. 審計管理員查看審計日誌檔大小和數目

```
$ls -lh /var/log/audit
```

經步驟 5 查看，系統審計日誌有：audit.log、audit.log.1、audit.log.2，其中 audit.log.1、audit.log.2 檔大小都為 1M 左右。

### 8.2.5.3.8. 審計日誌超閾值警報

當審計跟蹤的磁片空間已到達極限時，有審計報警日誌提醒。對於磁片空間極限值的設定，需要審計用戶手動調整/etc/audit/auditd.conf。

示例，假設系統日誌分區總大小 50G，將 30G 設置為剩餘空間閾值，即當用系統日誌佔用 20G 以上時會有審計日誌產生：

```
1. 審計管理員編輯 /etc/audit/auditd.conf 檔，使  
space_left=30000，即日志磁片空間閾值為 30G；  
space_left_action=SYSLOG，即向日誌中寫入一條報警日誌；（具  
體磁片空間閾值根據測試系統實際日誌磁片空間確定）
```

## 2. 審計管理員重啟服務

```
#systemctl restart auditd
```

步驟 2 後，如果日誌空間剩餘容量小於 30G，則審計日誌中將產生審計日誌資訊。

#### 8.2.5.4. 支持安全機制審計日誌類別

目前審計服務會根據系統中不同的安全機制功能寫入不同類型的審計日誌類型。

如 kysec 類的審計日誌類型為：

```
type=KYSEC_STATUS   msg=audit(1502938410.303:182384):
status=2  old_status=4   loginuid=0  session=4294967295
id=1502938410.303:182384

type=KYSEC_AVC      msg=audit(1502886638.943:132300):
denied { execute } for pid=15325  comm='bash'
name='/secadm/ls'  ssid=0x01  osid=0x00  loginuid=0
session=4294967295                               kysec_status=4
id=1502886638.943:132300
```

### 8.3. 麒麟安全管理工具-安全中心

安全中心是一款基於麒麟安全框架 KYSEC 的管理工具，提供系統安全加固、帳戶保護配置、聯網控制、應用執行控制和應用防護等功能，保障系統運行環境的安全和穩定。

安全中心集安全加固、帳戶保護、網路保護和應用保護等功能於一體，全面保障系統運行環境的安全。

安全加固：提供安全服務、內核參數、安全網絡、系統命令、系統審計、系統設置、潛在危險、檔許可權、風險帳戶、磁片檢查、密碼強度、帳戶鎖定、系統安全、系統維護、資源分配等多維度的掃描與一鍵加固，及時發現並處理系統

安全隱患。

**帳戶保護：**提供系統帳戶密碼強度檢查和帳戶鎖定機制，實現對系統帳戶的統一管控，提升系統帳戶安全防禦能力，有效防止密碼被暴力破解。

**網路保護：**提供應用聯網控制功能，即時防護未知應用網路行為，阻斷主動外聯及其它異常網路活動，提高網路訪問安全性。

**應用控制：**提供應用程式執行控制功能，阻止未知軟體、應用程式的惡意執行，避免木馬病毒攻擊，保障系統運行環境的安全性、可靠性。

**應用防護：**提供進程防殺死、內核模組防卸載和文件防篡改功能，保護系統關鍵檔完整性，阻止系統關鍵應用服務異常中斷。

詳細內容請參考《銀河麒麟安全中心用戶手冊》。

#### **8.4. 麒麟檔保護箱**

麒麟檔保護箱是基於內核級數據隔離機制的保護工具，提供用戶間數據隔離和加密保護功能，支持國密演算法，實現一箱一密、一文一密的細粒度控制，保障用戶數據安全。

具有如下特點：

**多重防護：**支持用戶間數據隔離以及細粒度的許可權控制，保障數據安全。

**安全加密：**支持一箱一密、一文一密的透明加密機制，且對密鑰進行安全管理，能夠滿足政企和金融級客戶的核心安全訴求。

**豐富演算法：**支持標準國際演算法、國密演算法和硬體級加密演算法，能夠滿足不同安全等級的加密應用場景。

高兼容性：支持保護箱版本相容機制，用戶升級適配無感知，保證用戶數據安全存儲、永不丟失。

簡單易用：支持內置檔管理器，實現統一管理，操作簡單、易於上手。

詳細內容請參考《銀河麒麟檔保護箱用戶手冊》。

## 第九章 FAQ

### 9.1. 版本查詢方法

字元終端輸入“nkvers”命令即可查詢，輸出資訊即為版本資訊，如下所示：

```
#nkvers
#####Kylin Linux Version #####
Release:
Kylin Linux Advanced Server release V10 (Halberd)
Kernel:
4.19.90-{內核小版本號}.x86_64
Build:
Kylin Linux Advanced Server
Kylin Linux Advanced Server
release V10 ( SP4 )/(Halberd)-x86_64-Buildxx/YYYYMMDD
#####
```

### 9.2. 字體安裝方法

在安裝字體之前首先需要查詢系統中已經安裝的字體，可以使用 `fc-list` 命令進行查詢，如果系統中沒有該命令需要先安裝軟體包。

```
yum install -y fontconfig mkfontscale
```

執行 `fc-list` 命令查詢已安裝的字體，如果發現有缺失相關字體檔，可以將

已拿到的字體檔（.ttf、otf 等）放入/usr/share/fonts/目錄下，然後使用如下命令建立字體索引資訊進行字體安裝。

```
cd /usr/share/fonts/  
mkfontscale  
mkfontdir  
fc-cache
```

字體安裝完畢之後使用 `fc-list` 查看安裝字體資訊，查看字體是否安裝成功。

### 9.3. 詳細包資訊查詢

使用 `rpm -pqi` 包名，可以查詢包詳細資訊。

```
[root@localhost 桌面]#rpm -pqi  
tigervnc-1.10.1-3.p02.ky10.x86_64.rpm  
Name       : tigervnc  
Version    : 1.10.1  
Release    : 3.p02.ky10  
Architecture: x86_64  
Install Date: (not installed)  
Group      : Unspecified  
Size       : 882530  
License    : GPLv2+  
Signature  : RSA/SHA1, 2020年12月06日 星期日 07時23分  
33秒, Key ID 41f8aebe7a486d9f  
Source RPM : tigervnc-1.10.1-3.p02.ky10.src.rpm  
Build Date : 2020年11月13日 星期五 15時40分20秒  
Build Host  : localhost.localdomain  
Packager    : Kylin Linux  
Vendor      : KylinSoft  
URL         : http://www.tigervnc.com  
Summary     : A TigerVNC remote display system
```

**Description :**

This package provides client for Virtual Network Computing (VNC), with which you can access any other desktops running a VNC server.

**9.4. 檢查包是否被篡改**

使用 rpm -Kv+包名可以查詢包是否被篡改，如下都為“OK”

```
[root@localhost      桌      面      ]#rpm      -Kv
tigervnc-1.10.1-3.p02.ky10.x86_64.rpm
tigervnc-1.10.1-3.p02.ky10.x86_64.rpm:
    頭 V3 RSA/SHA1 Signature, 密鑰 ID 7a486d9f: OK
    頭 SHA256 digest: OK
    頭 SHA1 digest: OK
    Payload SHA256 digest: OK
    V3 RSA/SHA1 Signature, 密鑰 ID 7a486d9f: OK
    MD5 digest: OK
```