SUSE Linux Enterprise Server 9 基本 DNS 伺服器架設

內容大綱：

- 前言
- Linux 名稱解析相關檔案
- DNS 正解原理
- DNS 類型
- DNS 相關套件及設定檔
- 實戰演練：建置 Caching-Only DNS（不指定 forwarder）
- 實戰演練：建置 Caching-Only DNS（指定 forwarder）
- 實戰演練：建置簡易 Master DNS（管理 geeko.idv.tw 網域）
- DNS 反解原理
- 實戰演練：建置 Master DNS（含反解區域）
- named.conf 細項參數
- 實戰演練：建置 Slave DNS
- 基本實用 Master DNS 設定（管理 geeko.idv.tw 網域和 61.219.23.0/24）
- 重點整理

2005/07/11 Alex Lin（林彥明）

前言

如果各位要到 Yahoo 網站競標，我想所有人應該都是在瀏覽器網址列上鍵入 http://tw.yahoo.com，我想沒有人會輸入 http://202.43.195.52。但是其實在 TCP/IP 協定中，每台主機都會有一個 IP Addrsss，兩台主機要互相溝通，必須要告知目的主機的 IP Address。也就是說如果想連到 Yahoo 網站，必須要輸入 http://202.43.195.52（因為 Yahoo 網站的 IP Address 為 202.43.195.52）。不過要人類去記這些數字，真的是太辛苦了。有人就想出一個辦法，在主機上編寫一個檔案，其中幫欲連線的主機取個好記的名字。例如：

IP Address	主機名稱
202.43.195.52	yahoo
202.43.195.52	pchome
202.43.195.52	msn

如此一來，假設需要連線到 202.43.195.52，只需輸入「yahoo」這個名稱即可。早期這個「主機名稱（hostname）」與 IP 的對應表是存放在每部機器的 hosts 這個檔案，當電腦個數不多還好，但是電腦數目一多就會發生問題了；因為電腦數目如果為數十萬台，則在 hosts 檔案中就必須有數十萬筆紀錄，何況現今網路上何只數十萬台機器。而且網路每新增一台主機，就必須修改 hosts 這個檔案。

還有一個麻煩的問題，每台主機的 hosts 檔案不盡相同。所以如果我架設一台網頁伺服器，想讓全世界都知道，還得告訴全世界我的主機 IP Address 61.219.23.88。因為我可能在我自己所管理的主機中全加入下列的設定，但不可能要求全世界的主機皆將這筆紀錄加入他們的 hosts 檔案。

```
61.219.23.88 neohome
```

所以後來發展出 DNS。DNS 全名 Domain Name System，透過 DNS 系統，我們查到某個機器的「主機名稱（hostname）」的「IP Address」；而且也可以 DNS 幫我們查詢某個「IP Address」的「主機名稱（hostname）」為何。

Linux 名稱解析相關檔案

在解釋 DNS 運作原理之前，筆者先探討何謂名稱解析（Name Resolution）及相關設定檔。

什麼是名稱解析（Name Resolution）,就是將主機名稱對應成 IP Adreess（正解）或將 IP Adreess 對應成主機名稱。首先筆者探討在名稱解析中時常出現的幾個名詞：主機名稱（hostname）、short name、FQDN （Fully Qualified Domain Name ）。

讀者在 Windows 2000 會發現電腦名稱（其實就是主機名稱）不能有「.」出現，例如「server1」。但在 Linux 中就可以包含「.」，例如 server1.example.com。一般而言，我們把不帶網域名稱 example.com 的名字稱為「**short name**」或稱「**simple name**」；附帶完整網域名稱的名字稱「**FQDN**」（**Fully Qualified Domain Name**）。

一般作業系統名稱解析的順序，都會設定成先 hosts，若查不到則詢問 DNS。故 hosts 檔目前仍經常使用，因為是直接問本機的檔案，所以速度會比詢問 DNS 快多了，一般建議將時常連線機器的主機名稱與其 IP 寫入此檔案中。

在 Unix-Base 的環境，host 位於 /etc 下，而 Window 的環境下則置於 /windows/system32/drivers/etc 下，其格式為：

IP Address	主機名稱 1	主機名稱 2	主機名稱 3
127.0.0.1	localhost.localdomain	localhost	
192.168.0.254	server1.example.com	server1	www
192.168.0.1	station1.example.com	station1	www1

此檔案還可以簡化鍵盤輸入的長度，如 telnet station1，即會連線到 192.168.0.1，而可不必輸入過長之的 station1.example.com。使用 hosts 雖有不少好處，但有時難免會與 DNS 的資料不同；所以維護 hosts 檔資料的正確性有時還是有必要的，這是不少人常忽略的地方。

對 Linux 而言，如果/etc/hosts 找不到的對應的紀錄，該去問那一台 DNS 呢？欲詢問的 DNS 之 IP 存放於/etc/resolv.conf 。格式應如下：

```
#cat /etc/resolv.conf
nameserver 192.168.0.254
nameserver 168.95.1.1
search example.com     ← 網域搜尋順序
```

其中 **search** 的用途是如果你執行網路的指令是用「**short name**」代表（ 例 telnet server1 ），則此台機器會將 search 後的字串 example.com 附加在其後變成 server1.example.com，然後再去詢問 DNS。

在 Linux 主機上欲找到某個主機名稱所對應的 IP Address，很像我們要查某個朋友的電話，一般都是先找自己的通訊錄（/etc/hosts），如果通訊錄上沒有，就問查號台（DNS）。讀者可能會想可不可以先問 DNS，當然可以！雖然比較沒效率，但可避免因各台主機上 /etc/hosts 不一致所造成的問題。讀者只需修改 **/etc/nsswitch.conf** 約第 33 行，將 dns 移至 files 前面即可。其中 files 的意義就是參考本機的檔案（/etc/hosts）。

```
# vi /etc/nsswitch.conf
33 hosts:     files dns
```

DNS 正解名稱原理

但讀者一定會有疑問，「網路上那麼多機器，到底是如何命名」？「DNS 又是如何幫我們查到某個主機名稱所對應的 IP Address」。第一個 DNS 的規範（RFC1034， RFC1035）是在 1984 年由 Paul Mockapetris 建立。由 DNS 來統一提供相關的資訊，讓不管在那一台機器上查詢網路上「主機名稱」的 IP 都會得到一致的結果（正解，Forward Lookup）。

基本上，DNS 最大的工作就是將主機名稱對應到 IP Address 這個功能（正解，Forward Lookup），不過 DNS 也提供利用 IP Address 來反推「主機名稱」的服務（反解，Reverse Lookup）。DNS 具有以下特性：

- 全球最大的分散式資料庫系統。
- 自己的資料由自己維護，而其他人的資料則分散在全球。
- 沒有一台 DNS Server 會有全部名稱解析的資料。
- 以樹狀結構的方式找到目的位址（每個節點需將被授權）。

「網路上那麼多機器，到底是如何命名」，簡而言之「分門別類」。

目前全球有超過一億部的 DNS Server，以上述的特性運作，可正確且快速的解析到網域名稱與 IP 的對應，這些對應都由 root （“.”） 開始，故其地位相當重要。DNS 系統如同一樹狀結構，每一個分支以 “.” 分隔，其限制最多 127

層，每個分支最長 63 字元 （a-z， 0-9， -），總長 255 字元。

DNS 是一個分層級的分散式名稱對應系統，在最頂端的是一個「root」server。root server 是由 1993 年美國 NSF 建立 InterNIC（Network Information Center）組織負責維護，InterNIC 負責美國以及全球相關的網域名稱登記及系統管理事項，它是全球唯一的網域名稱系統最高管理者。管理 root server 的組織權力最大，他們可以決定有幾個 TLD。

接下來是 **TLD** （ **T**op **L**evel **D**omain ），TLD 又分為 gTLD （ generic TLD ）如「.com」、「.org」、「.net」、「.edu」、「.gov」、「.mil」、「.int」、「.arpa」及 ccTLD （ country code TLD ）如「.tw」（台灣）、「.jp」（日本）、「.uk」（英國）…（ ISO-3166 所定義的 2 個 byte 國碼 ）。

表 1：常見的 TLD 列表

名稱	代表意義
com	公司、行號、企業
org	組織、機構
edu	教育單位
gov	政府單位
net	網路、通訊
mil	軍事單位
arpa	用來將 IP Address 轉換為 FQDN，例如 **# host 130.57.4.27** **27.4.57.130.in-addr.arpa domain name pointer www.novell.com.**

目前「**.com**」及「**.net**」由 networksolutions（已被 verisign 買下）公司所經營，「**.com**」、「**.gov**」、「**.mil**」分別為美國的企業單位、政府單位、軍事單位，「**.int**」為一些國際間的需求（如 internet fax）使用，「**.arpa**」原本為 arpanet （internet 的前身）單位所使用，現為 DNS 反解使用。

接下來我們來探討當 DNS 收到查詢的需求時，到底如何得到對應的 IP。例如有台機器向你的 DNS 詢問 www.novell.com 的 IP 為何？DNS 到底如何運作而得到 www.novell.com 的 IP？

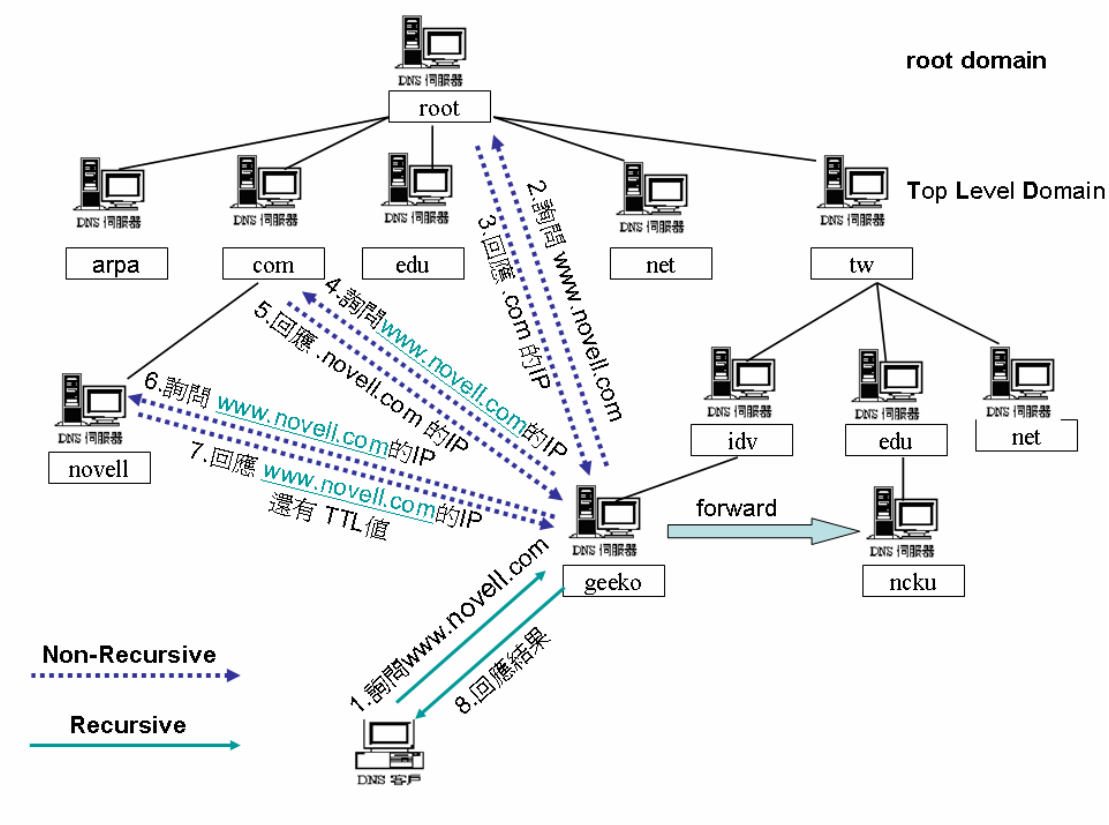

圖 1：DNS 正解名稱解析流程圖

步驟一：

DNS 用戶端向指定的 DNS 伺服器查詢網際網路上某台主機名稱（www.novell.com），若您的 DNS 是管理這個網域名稱（novell.com）的 DNS，有人向你查詢這個主機名稱的資料，可以直接做出回答，則您的 DNS Server 稱為權威（Authoritative）主機，而回應的結果稱為權威回答（Authoritative Answer）。

如果所查詢的主機名稱屬於其它 Domain 的話，則會檢查快取暫存區（Cache），看看有沒有相關資料，在每一個 DNS Server 中都有一個快取暫存區（Cache）。這個快取暫存區的主要目的，是將該名稱伺服器所查詢出來的名稱及相對的 IP 位址紀錄在快取暫存區中。這樣當下一次還有另外一個用戶端到 DNS 上去查詢相同的名稱時，就可直接可以從暫存區中找到該筆名稱紀錄資料，傳回給用戶端，加速用戶端對名稱查詢的速度。

如果名稱伺服器在資料紀錄查不到且快取暫存區中也沒有時，伺服器才會「向 root Server 查詢」或將「查詢需求 forward 另一台 DNS」，麻煩另一台 DNS 幫忙查到對應的結果。就 DNS Client 而言，它所送出的查詢是所謂「Recursive

遞迴查詢」，使用者只送出一個查詢，由 DNS Server 完成其他所需的查詢後回應。

步驟二～三：

當一部 DNS server 剛啟動時，快取暫存區中空空如也，，所以一開始一定要向 root server 發出查詢請求。一般而言，一個 authority 的 DNS 只會告訴查詢者下一站（另一台 DNS Server）到那查詢，而不會主動到外面將結果查回來給查詢者，這種 Query 稱為「**Non-Recursive Query**」。而這種回應的型式我們稱為 “referral”（即回應 NS 紀錄）。

註：DNS server 會有一個檔案紀錄 root server 的 IP

為什麼不提供「**Recursive Query**」？其原因在於遞迴主機會暫存別人的資料，這可能會造成 DNS 欺騙的問題，如果你的 DNS 主機被欺騙了，對你可能沒有什麼關係，但對上層（root、.com、.net ...）等，因為大家都會用到，所以會設成非遞迴（**Non-Recursive**），因為非遞迴不會暫存（不幫別人代查就不會暫存結果），只會回應自己所負責的 Domain 的資料及 root server 清單，故能有較高的安全性。

另外，非遞迴因為只有收到、回應自己 Domain 的資料，不幫別人代查，也可減輕自己的負擔。所以 root server 並不會幫 geeko.idv.tw 的 DNS 查到 www.novell.com 的 IP，而只會告訴它負責.com 的 DNS IP Address，整個流程如圖 2 所示。

- **Recursive Query**：在上面介紹的過程中，DNS client 端只丟出一個詢問給 local DNS server，然後 local DNS 就會不斷地查到答案出來為止，最後把結果傳回來給 client，這種查詢稱為 Recursive Query。
- **Non-Recursive Query**（**iterative query**）：前面的介紹中，local DNS 對其它 DNS 發出的詢問，都只是知道一個更進一步的線索，然後發問者（local DNS）根據線索再去進一步找答案，這種詢問方式稱為 Non-Recursive Query（iterative query）。

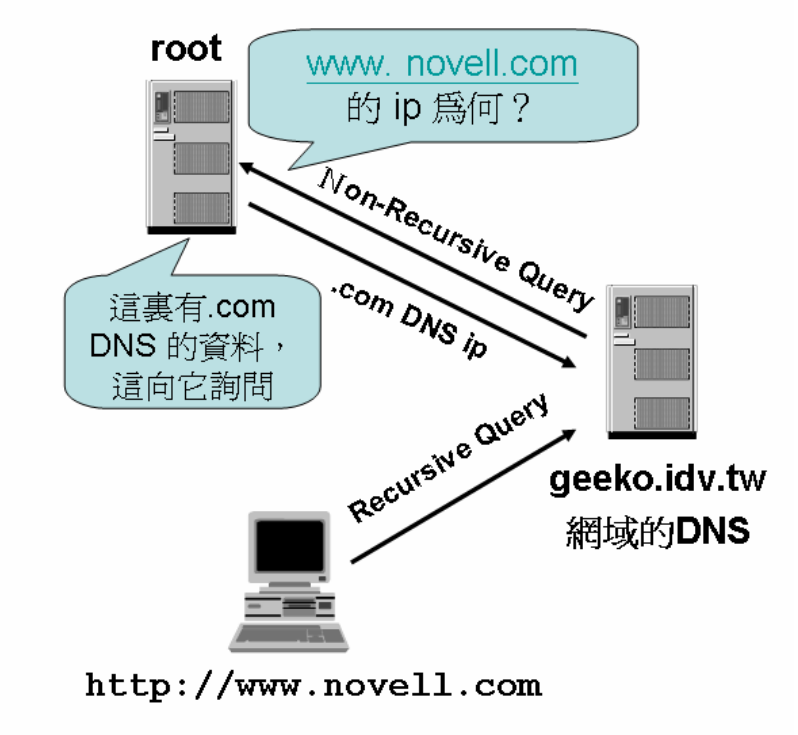

圖 2：Recurive 和 Non-Recursive Query 的比較

步驟四~五：同樣的.com 的 DNS 也只是將 novell.com.的 DNS 在那告訴查詢者。

步驟六~七：novell.com 的 DNS 回應 www.novell.com 的 IP 為何，而且還會附加「TTL（Time To Live ）」值，通常是 86400 秒；TTL 的作用是告訴 geeko. 的 DNS 這個查詢結果可以保存（Caching）多久。

筆者時常形容「**TTL 是 DNS 查詢結果的保存期限**」，下次再有主機詢問 geeko.idv.tw 的 DNS www.novell.com 的 IP 為何，在保存期限內，它就直接回應 www.novell.com 的 IP 為何，而不會依照之前的流程重新詢問一次。

DNS 類型

DNS server 的類型可以分為以下三種：

Master DNS：本身含有 Domain 的資料庫（Zone），此資料庫其實就是包含正解紀錄或者是反解紀錄的文字檔（Zone File）。

Master DNS 的相關設定檔及 Zone File, 通常需有 1 個 DNS 的主要組態檔及 5

個 Zone File（假設其管理 1 個正解網域、1 個反解網段）。

DNS 的主要設定檔（/etc/named.conf：最主要記錄所管理的網域及網段）
網域正解檔（自行撰寫）
網域反解檔（自行撰寫）
本機正解檔（原本需自行撰寫，但現大多由 DNS 套件提供）
本機反解檔（原本需自行撰寫，但現大多由 DNS 套件提供）
root server 資訊檔（本應由 InterNIC 組織下載，但現大多由 DNS 套件提供）

Slave DNS：這種類型的 DNS 功能最主要為備份 Master DNS 的資料庫，並提供名稱解析的功能。它本身也有網域的 Zone File，不過它的 Zone File 是向 Master DNS 複製（Zone Transfer）而來的。

Slave DNS 的相關設定檔及 Zone File 通常也是有 1 個 DNS 的主要組態檔及 5 個 Zone File（假設其備份 1 個正解網域、1 個反解網段的資料）。

DNS 的主要設定檔（/etc/named.conf：最主要記錄所備份的網域及網段）
網域正解檔（向 Mater DNS 做 Zone Transfer 而來）
網域反解檔（向 Mater DNS 做 Zone Transfer 而來）
本機正解檔（原本需自行撰寫，但現大多由 DNS 套件提供）
本機反解檔（原本需自行撰寫，但現大多由 DNS 套件提供）
root server 資訊檔（本應由 InterNIC 組織下載，但現大多由 DNS 套件提供）

Caching-only DNS：Caching-Only DNS 沒有 Domain 資料庫，單純僅幫助 Client 端向外部的 DNS 主機要求資料，然後再保留查詢結果至快取暫存區（Cache）。則下次 Client 再提出名稱查詢的需求，若 TTL 還未過期，就直接檢查快取暫存區（Cache），不用再去詢問另一台 DNS。

Caching-Only DNS 因為沒有管理網域及網段，所以不需有網域正解及網域反解檔，所以只需有 1 個 DNS 的主要組態檔及 3 個 Zone File。

DNS 的主要設定檔（/etc/named.conf：最主要記錄外部的 DNS 的資訊）
本機正解檔（原本需自行撰寫，但現大多由 DNS 套件提供）
本機反解檔（原本需自行撰寫，但現大多由 DNS 套件提供）
root server 資訊檔（本應由 InterNIC 組織下載，但現大多由 DNS 套件提供）

註：本機正解檔、本機反解檔、root server 資訊檔不管那一類型的 DNS 內容均相同。

DNS 相關套件及設定檔

DNS 的相關套件及設定檔如下：

Daemon:named
Daemon 類別：System V standalone daemon
所需套件：
➢bind-*.rpm：為 BIND DNS 伺服器主要程式及設定檔
➢bind-devel：DNS 所需的 library 及相關文件。
➢ bind-utils：內為 host、dig、nslookup 等 DNS 查詢必備工具。
啟動 Script：/etc/init.d/named
Port：53
設定檔：/etc/named.conf，/var/lib/named/*
Log 檔：/var/log/messages

實戰演練：建置 Caching-Only DNS（不指定 forwarder）

因為 Caching-Only DNS 沒有 Domain 資料庫，僅單純幫助 Client 端把需求丟給外部的 DNS 主機，得到結果後便將其暫存，以便下次再有同樣名稱解析的需求時，無需再向外部的 DNS 查詢。所以我們無需去申請 Domain，便可架設 caching-only DNS，這是最容易架設的 DNS。

實作環境:SLES 9+SP1

步驟一：安裝相關套件

利用「YaST 新增或移除軟體」模組安裝 DNS 相關套件，可利用下列方法啟動：

【方法一】圖形介面：點選工具列「開始」→系統→YaST→系統→新增或移除軟體
【方法二】圖形介面：於命令列輸入「yast2」→系統→新增或移除軟體
【方法三】圖形介面：於命令列輸入「yast2 sw_single」
【方法四】文字介面：輸入「yast」→Software→Install and Remove Software
【方法五】文字介面：輸入「yast sw_single」

利用「YaST 新增或移除軟體」模組的搜尋功能，鍵入關鍵字「bind」出現如圖 3 的畫面，勾選 DNS 相關套件。

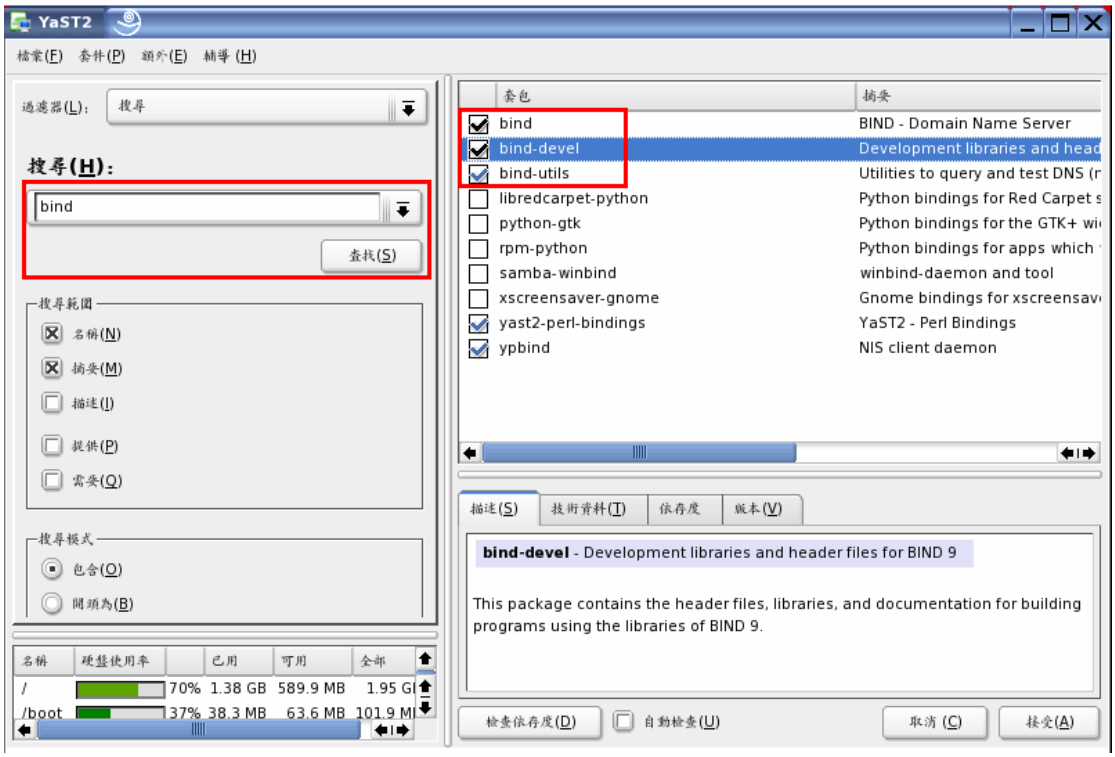

圖 3：安裝 DNS 相關套件畫面

跟 DNS 相關套件如下：

- bind：為 BIND DNS 伺服器主要程式。
- bind-devel：DNS 所需的 library 及相關文件。
- bind-utils：內為 host、dig、nslookup 等 DNS 查詢必備工具。

勾選所需套件後，按下「接受（A）」，會出現提示放入光碟的訊息，放入適當的光碟片，便可順利安裝。

步驟二：查詢 DNS 相關設定檔

由 4 節可得知 Caching-Only DNS 因為沒有管理網域及網段，所以不需有網域正解及網域反解檔，所以只需有 1 個 DNS 的主要組態檔及 3 個 Zone File

DNS 的主要設定檔（/etc/named.conf：最主要記錄外部的 DNS 的資訊）
本機正解檔（原本需自行撰寫，但現大多由 DNS 套件提供）
本機反解檔（原本需自行撰寫，但現大多由 DNS 套件提供）
root server 資訊檔（本應由 InterNIC 組織下載，但現大多由 DNS 套件提供）

而這些 Zone File 會由 DNS 的相關套件提供，我們利用「rpm –ql」指令查詢是否真的已包含在 bind 套件內。

```
sles9:~ # rpm -ql bind
/etc/init.d/named
/etc/named.conf ← DNS 的主要設定檔
/etc/named.conf.include
/var/lib/named ← 預設存放 Zone File 的目錄
/var/lib/named/127.0.0.zone ←本機反解檔
/var/lib/named/dev
/var/lib/named/dev/log
/var/lib/named/dev/null
/var/lib/named/dev/random
/var/lib/named/dyn
/var/lib/named/etc
/var/lib/named/etc/localtime
/var/lib/named/etc/named.conf.include
/var/lib/named/etc/named.d
/var/lib/named/etc/named.d/rndc.access.conf
/var/lib/named/localhost.zone ←本機正解檔
/var/lib/named/log
/var/lib/named/master ← 存放 Master DNS Zone File 的目錄
/var/lib/named/root.hint ←root server 所在資訊檔
/var/lib/named/slave← 存放 Slave DNS Zone File 的目錄
/var/lib/named/var
/var/lib/named/var/lib
/var/lib/named/var/lib/named
/var/lib/named/var/log
/var/lib/named/var/run
/var/lib/named/var/run/named
/var/run/named
```

bind rpm 提供其中的主要組態檔、本機正解檔、本機反解檔、root server 資訊檔可幫助使用者快速建置 Caching-Only DNS。

■DNS 的主要組態檔：/etc/named.conf

■本機正解檔：/var/lib/named/localhost.zone

■本機反解檔：/var/lib/named/127.0.0.zone

■root server 所在資訊檔：/var/lib/named/root.hint

其實 SLES 9 DNS9 Daemon 啟動時，並不是參考上述的位置的設定檔及 Zone File。因為 SLES 9 DNS9 Daemon 預設是啟動 chroot 機制，可參考 /etc/sysconfig/named 內的設定：

sles9:~ #**vi /etc/sysconfig/named**

```
 8 # Shall the DNS server 'named' run in the chroot jail /var/lib/named/?
 9 #
10 # Each time you start 'named' with the init script, /etc/named.conf,
11 # /etc/named.conf.include, /etc/rndc.key, and all files listed in
12 # NAMED_CONF_INCLUDE_FILES will be copied relative to /var/lib/named/.
13 #
14 # The pid file will be in /var/lib/named/var/run/named/named.pid.
15 #
16 NAMED_RUN_CHROOTED="yes" ← 預設啟用 chroot 機制
17
```

所謂 DNS chroot 機制，預設是將/var/lib/named 當成/ 根目錄，所以本來應該置於/etc/目錄的 named.conf 變成應該置至/var/lib/named/etc 目錄下。所以啟動 chroot 機制下，所有的相關檔案應該如下：

■DNS 的主要組態檔：**/var/lib/named**/etc/named.conf
■本機正解檔：**/var/lib/named**/var/lib/named/localhost.zone
■本機反解檔：**/var/lib/named**/var/lib/named/127.0.0.zone
■root server 所在資訊檔：**/var/lib/named**/var/lib/named/root.hint

讀者看到這裏，可能會想“好複雜”，乾脆把 chroot 機制關掉。其實 chroot 機制是為了增加系統的安全性，因為啟動了 chroot 後，DNS 的相關 process 只能讀取到/var/lib/named 目錄下的檔案，因為 DNS 的相關 process 所看到的根目錄已變成/var/lib/named。SLES 9 為了系統的安全性，所以預設啟動 chroot，而且怕因此造成使用者不便。所以當啟動 DNS 時，會自動將相關檔案複製至 /var/lib/named 目錄下，所以讀者在設定時，還是跟沒有 chroot 時設定的方式相同。

步驟三：啟動 DNS

利用 rcnamed start 指令便可立即啟動 DNS，若希望一開機就啟動 named，可執行「chkconfig named on」指令。

```
sles9:~ # ls -l /var/lib/named/etc/named.conf ← 原本無此檔案
/bin/ls: /var/lib/named/etc/named.conf: No such file or directory
sles9:~ # rcnamed start ←立即啟動 DNS
Starting name server BIND                                          done
sles9:~ # ls -l /var/lib/named/etc/named.conf ← 產生 named.conf
-rw-r-----    1 root named 3902 Jul    3 15:44 /var/lib/named/etc/named.conf
sles9:~ # chkconfig named on ←開機後自動啟動 DNS
```

這樣便完成 Caching-Only DNS 的建置，讀者此時可能會納悶，Caching-Only 不是應該設定將名稱查詢的需求 forward 給另一台 DNS，怎麼我們都沒有設定？其實若是未在 /etc/named.conf 設定名稱查詢的需求該 forward 給那台 DNS，Caching-Only DNS 會將需求導至 root server，根據圖 1 的流程去得到結果，並保留在快取暫存區。

步驟四：測試

讀者可以用另一台 Windows 機器，將網路設定中的 DNS 指向 Caching-Only DNS 的 IP Address。或是我們將 Caching-Only DNS 中的 /etc/resolv.conf 指向自己的 IP 來測試是否正常運作亦可。

```
sles9:~ #ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:02:B3:9B:C6:B7
          inet addr:61.219.23.88  Bcast:61.219.23.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:8421 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8196 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3356301 （3.2 Mb）  TX bytes:1086222 （1.0 Mb）
          Interrupt:5 Base address:0xd800 Memory:ef020000-ef020038

sles9:~ #cat /etc/resolv.conf
nameserver 61.219.23.88 ← 自己本機的 IP
sles9:~ # host dns.hinet.net
dns.hinet.net has address 168.95.1.1 ← 可正確解析 dns.hinetnet，代表 Caching-Only DNS
```

實戰演練：建置 Caching-Only DNS（指定 forwarder）

所謂的 forwarder，就是當某一台 DNS 遇到非本機負責的 Zone 之查詢請求的時候，將不直接向 root Server 查詢，而把請求轉交給指定的另一台 DNS 主機（forwarder）代為查詢。

其實就是將自己扮成一個 DNS Client，向 forwarder 送出同樣的請求，然後等待查詢結果；而逐級往下查詢的動作，則交由 forwarder 負責，自己本身就輕鬆多了。但無論這個結果是自己直接查詢得來的，還是 forwarder 送回來的，DNS 都會保存一份資料在 cache 暫存區中。這樣，其後的相同查詢就快多了，這對於 DNS 所服務的 client 而言更是有效率得多。

Caching-Only DNS 若不指定 forwarder，則將需求 forward 給 root server。此演練筆者將指定 forwarder，而且由於 Caching-Only DNS 沒有負責的網域，所以通常會加上 **forward only**；這樣代表這台 DNS 只會把查詢需求 forward 給另一台 DNS。

由於之後的演練必須修改/etc/named.conf，所以我們先來看看 SLES 9 預設的 /etc/named.conf 的長相。

sles9:~ # vi /etc/named.conf

```
#
# Author: Frank Bodammer, Lars Mueller <lmuelle@suse.de>
#
# /etc/named.conf
#
# This is a sample configuration file for the name server BIND 9.  It works as
        2 # All rights reserved.
        3 #
        4 # Author: Frank Bodammer, Lars Mueller <lmuelle@suse.de>
        5 #
        6 # /etc/named.conf
        7 #
        8 # This is a sample configuration file for the name server BIND 9.  It works as
```

```
 9 # a caching only name server without modification.
10 #
11 # A sample configuration for setting up your own domain can be found in
12 # /usr/share/doc/packages/bind/sample-config.
13 #
14 # A description of all available options can be found in
15 # /usr/share/doc/packages/bind/misc/options.
16
17 options {
18
19              # The directory statement defines the name server's working directory
20
21              directory "/var/lib/named";
22              # Write dump and statistics file to the log subdirectory.    The
23              # pathenames are relative to the chroot jail.
24
25              dump-file "/var/log/named_dump.db";
26              statistics-file "/var/log/named.stats";
27
28              # The forwarders record contains a list of servers to which queries
29              # should be forwarded.    Enable this line and modify the IP address to
30              # your provider's name server.    Up to three servers may be listed.
31
32              #forwarders { 192.0.2.1; 192.0.2.2; };
33
34              # Enable the next entry to prefer usage of the name server declared in
35              # the forwarders section.
36
37              #forward first;
38
39              # The listen-on record contains a list of local network interfaces to
40              # listen on. Optionally the port can be specified.    Default is to
41              # listen on all interfaces found on your system.    The default port is
42              # 53.
43
44              #listen-on port 53 { 127.0.0.1; };
45
```

```
46              # The listen-on-v6 record enables or disables listening on IPv6
47              # interfaces.    Allowed values are 'any' and 'none' or a list of
48              # addresses.
49
50              listen-on-v6 { any; };
51
52              # The next three statements may be needed if a firewall stands between
53              # the local server and the internet.
54
55              #query-source address * port 53;
56              #transfer-source * port 53;
57              #notify-source * port 53;
58
59              # The allow-query record contains a list of networks or IP addresses
60              # to accept and deny queries from. The default is to allow queries
61              # from all hosts.
62
63              #allow-query { 127.0.0.1; };
64
65              # If notify is set to yes (default), notify messages are sent to other
66             # name servers when the the zone data is changed. Instead of setting
67              # a global 'notify' statement in the 'options' section, a separate
68              # 'notify' can be added to each zone definition.
69
70              notify no;
71 };
72
73 # To configure named's logging remove the leading '#' characters of the
74 # following examples.
75 #logging {
76 #           # Log queries to a file limited to a size of 100 MB.
77 #           channel query_logging {
78 #                       file "/var/log/named_querylog"
79 #                                   versions 3 size 100M;
80 #                       print-time yes;                         // timestamp log entries
81 #           };
82 #           category queries {
```

```
 83 #                    query_logging;
 84 #            };
 85 #
 86 #            # Or log this kind alternatively to syslog.
 87 #            channel syslog_queries {
 88 #                    syslog user;
 89 #                    severity info;
 90 #            };
 91 #            category queries { syslog_queries; };
 92 #
 93 #            # Log general name server errors to syslog.
 94 #            channel syslog_errors {
 95 #                    syslog user;
 96 #                    severity error;
 97 #            };
 98 #            category default { syslog_errors;    };
 99 #
100 #            # Don't log lame server messages.
101 #            category lame-servers { null; };
102 #};
103
104 # The following zone definitions don't need any modification.    The first one
105 # is the definition of the root name servers.    The second one defines
106 # localhost while the third defines the reverse lookup for localhost.
107
108 zone "." in {
109              type hint;
110              file "root.hint";
111 };
112
113 zone "localhost" in {
114              type master;
115              file "localhost.zone";
116 };
117
118 zone "0.0.127.in-addr.arpa" in {
119              type master;
```

```
120             file "127.0.0.zone";
121 };
122
123 # Include the meta include file generated by createNamedConfInclude. This
124 # includes all files as configured in NAMED_CONF_INCLUDE_FILES from
125 # /etc/sysconfig/named
126
127 include "/etc/named.conf.include";
128
129 # You can insert further zone records for your own domains below or create
130 # single files in /etc/named.d/ and add the file names to
131 # NAMED_CONF_INCLUDE_FILES.
132 # See /usr/share/doc/packages/bind/README.SuSE for more details.
```

看到這個檔案，我想很多人一定會昏倒，不然就會想筆者是不是灌水灌太兇了。其實若個各位仔細觀看，會發現 Novell 真的是超用心的，一一說明常用的選項，不過對非英語體系的國家，乍看之可能會覺得看到一堆蝌蚪文。筆者先介紹如何實作指定 forwarder 的 Caching-Only DNS，待會再對這個檔案略處理。

實作指定 forwarder 的 Caching-Only DNS 的方法，只須將原第 31~33 行的內容

```
31
32              #forwarders { 192.0.2.1; 192.0.2.2; };
33
```

改為下列：

```
31
32          forwarders { 168.95.1.1; };
33          forward only;
```

然後重新啟動 named 即可，這樣一來，當這台 DNS 收到名稱查詢的需求時，便會 forward 給 Hinet 的 DNS 168.95.1.1，由 Hinet 的 DNS 代為處理。這樣速度會丟給 root server 快，但這種方式有個缺點，若是 Hinet 的 DNS 的快取暫存區資料更新出現問題，會造成你的 DNS 老是查到舊的資料。筆者就曾遇過客戶將 forwarders 指向 Hinet 的 DNS，但因 Hinet 的快取暫存區資料的保存期限（TTL）因不明原因一直未過期，造成客戶未能查到正確的資料。

```
Sles9:~ # rcnamed restart
```

```
Shutting down name server BIND                                  done
Starting name server BIND                                       done
```

接著筆者將/etc/named.conf 稍作整理如下：

```
 17 options { ←DNS 選項設定區段
 21             directory "/var/lib/named";
 25             dump-file "/var/log/named_dump.db";
 26             statistics-file "/var/log/named.stats";
 32             forwarders { 192.0.2.1; 192.0.2.2; };
 33             forward only;
 50             listen-on-v6 { any; };
 70             notify no;
 71 };

108 zone "." in {
109             type hint;
110             file "root.hint";
111 };
112
113 zone "localhost" in {
114             type master;
115             file "localhost.zone";
116 };
117
118 zone "0.0.127.in-addr.arpa" in { ←
119             type master;
120             file "127.0.0.zone";
121 };
127 include "/etc/named.conf.include";
```

這樣是不是清爽多了，從整理過的 named.conf，應可看出其架構為：

```
options {
    DNS 選項設定區段
      };

zone "." in {
    root server 資訊檔設定區段
```

```
};

zone "localhost" in {
    本機正解檔設定區段
};

zone "0.0.127.in-addr.arpa" in {
    本機反解檔設定區段
};
include "/etc/named.conf.include";
```

包含了 root server 資訊檔、本機正解檔、本機反解檔，跟之前介紹的原理是不是完全吻合呢，套句筆者最常講的話「**指令和設定檔只不過是觀念的實作**」。

實戰演練：建置簡易 Master DNS（管理 geeko.idv.tw 網域）

Master DNS 本身含有 Domain 的資料庫（Zone），就是包含正解紀錄或者是反解紀錄的文字檔（Zone File），筆者在這實例演練中利用真實的案例來說明建置 Master DNS 所需的步驟。

註：筆者特別去申請了 geeko.idv.tw 網域，geeko 是 SUSE Linux 的吉祥物－綠色變色龍，對喜歡這個網域名稱的讀者說聲「歹勢」，被筆者捷足先登了！

步驟一：申請網域

因為 Master DNS 本身含有 Domain 的資料庫（Zone），所以要建置 Master DNS 首先得向上一層 DNS 註冊，即麻煩它做所謂授權（Delegation）的動作，將某個網域名稱的資料委派給你的機器管理。

筆者簡介在 seednet 申請在 http://rs.seed.net.tw 申請「**geeko.idv.tw**」網域的過程，至於為什麼在 seednet 申請，不是筆者特別偏好 seednet。只是筆者希望可利用信用卡付費，找來找去發現 seednet 可以線上刷卡，就選 seednet 了！

圖 4：seednet 網域申請畫面

Seednet 個人網域申請流程還算親和，主要填入申請的網域名稱及個人資料，如身份證字號、E-mail，資料填寫完畢後便可直接線上刷卡。

圖 5：填入申請的網域名稱

成功扣款後，你應該會收到一封 mail，內容如下，只要完成確認動作，便可到 http://rs.seed.net.tw 管理剛申請的網域。重新連到 http://rs.seed.net.tw，點選左上角的「我的域名管理」(圖 6)。

XXX 您好：

您所申請之網域名稱：geeko.idv.tw 已完成網域名稱申請動作。請您
於2005/07/09前完成網域申請客戶確認程序。請點選下列連結進行確認作業：
https://service.seed.net.tw/register-cgi/rgconfirm?FUNC=CONFIRMDN&ID=N120061403&TransNo=T0993871&Key=1Zvyy1A3fd0%253D
若您在2005/07/09以前未完成網域申請客戶確認程序，系統將自動取消此網域名稱，
無法再行確認且需依申請流程重新登錄網域名稱，做請務必準時完成確
認程序若您已完成確認作業，請忽略本通知勿需作任何處理。
若您有任何問題，請隨時與我們連絡。

Seednet 網域名稱申請小組
(Domain Name Register Center)

圖 6：Seednet 域名管理畫面

點選「我的域名管理」，輸入身份證號及密碼後會出現圖 7 的畫面，點選畫面右方的「2.英文-DNS(HOST/IP 指定)，便可指定 geeko.idv.tw 的」DNS 為何？


目前域名數量：1
目前轉址數量：0

1.用戶密碼[異] 2.發票資料[異] 3.交易資料[查]

個人網域名稱

網域名稱	狀態	到期日	查詢/異動項目
geeko.idv.tw	使用中	2006/07/04	1.基本資料[異] 2.英文-DNS(HOST/IP指定) 3.繳費 4.WhoIS查詢開關 5.異動網域名稱

[查]：表查詢功能 [異]：表異動功能 ([查] or [異]視網域名稱狀態而定)
基本資料：任何狀態下皆爲[異]。
英文-DNS(Host/IP指定)：使用中才提供功能。
繳費：通繳中、轉入通繳中、使用中才提供功能。

圖 7：「我的域名管理」畫面

接著出現圖 8 的畫面，什麼是「主機模式」？什麼是「DNS 模式」？
所謂的主機模式，就是你自己不用架設 DNS，由 ISP 幫你負責這個網域的相關紀錄。也就是圖上所列出的 vdns1.seed.net.tw 及 vdns1.seed.net.tw 這兩台 DNS 會幫你管理 geeko.idv.tw 網域的資料，通常 ISP 允許你可設定 3~5 筆的資料。

那「DNS 模式」又是什麼，DNS 模式代表你有架設 DNS，你要自行管理 geeko.idv.tw 網域的資料。也就是麻煩上一層 DNS 做所謂授權（Delegation）的動作，將 geeko.idv.tw 網域名稱的資料委派給你的 DNS 管理。

授權（Delegation）的動作便是在上一層 DNS 設定一筆 NS Record，表示負責「**geeko.idv.tw**」網域 DNS 主機名稱為 dns.geeko.idv.tw，和一筆 A Record 告知 dns.geeko.idv.tw 的 IP 為 61.219.23.88。

通常 ISP 會要求你填入二台以上的 DNS（Slave DNS），筆者因為只有一台 DNS，所以將第二及第三台 DNS 也指向同一台機器（圖 7）。

圖 8：「網域名稱異動」畫面

步驟二：修改/etc/named.conf

接下來，你必須修改 /etc/named.conf，設定此台 DNS 所負責的網域為「geeko.idv.tw」及網域正解檔（Domain Zone File）的存放位置。

sles9:~ # vi /etc/named.conf

```
17 options { ←DNS 選項設定區段
21          directory "/var/lib/named";
25          dump-file "/var/log/named_dump.db";
26          statistics-file "/var/log/named.stats";
32          forwarders { 192.0.2.1; 192.0.2.2; };
33          forward only;      將這兩行註解掉或刪除
50          listen-on-v6 { any; };
70          notify no;
71 };

108 zone "." in {
109         type hint;
110         file "root.hint";
111 };
112
113 zone "localhost" in {
114         type master;
```

```
115             file "localhost.zone";
116 };
117
118 zone "0.0.127.in-addr.arpa" in { ←
119             type master;
120             file "127.0.0.zone";
121 };
122 #加入以下這段文字
123 zone "geeko.idv.tw" in {
124   type master; ← 扮演 Master DNS
125   file "master/geeko.idv.tw.zone"; ← Zone File 所在位置
126 };
132 include "/etc/named.conf.include";
```

步驟三：新增網域正解 Zone File（/var/named/master/gooke.idv.tw.zone）

Master DNS 維護網域的資料，所以必須新增包含正解紀錄文字檔（Zone File），內容如下：

sles9:~ # vi /var/lib/named/master/gooke.idv.tw.zone

```
$TTL 2d ←TTL 值定為 2 天
@                   IN SOA          dns.geeko.idv.tw.     neo.geeko.idv.tw. (
                                    2005070200          ; serial
                                    3h                  ; refresh
                                    1h                  ; retry
                                    1w                  ; expiry
                                    1d )                ; minimum
geeko.idv.tw.               IN NS           dns.geeko.idv.tw.
dns.geeko.idv.tw.           IN A            61.219.23.88
```

這個 Zone File 含 1 個 TTL 值的定義及 3 筆 Resource Record，TTL 在第 2 節原理已提過，指的是別台的 DNS 向我這台 DNS 詢問資料時，告知對方的 DNS 我所回覆的結果，可以保存多久（單位預設是秒），在 TTL 時間未到時，對方的 DNS 對同樣的名稱解析需求是不會重新查詢。

什麼是所有 Resource Record 呢？DNS 的資料庫稱之為「Zone」，而資料庫內的是一筆一筆的紀錄便是要 Resource Record（RR）。

【Resource Record 的語法】

Resource Record 的語法如下：

[domain]	[ttl]	[class]	<type>	<resource_record_data>

■ **domain**：代表要對應的名稱。

■ **ttl**：代表這筆 record 的 TTL（Time To Live），意思是當其它的 DNS server cache 這筆 record 時，最長不應該超過的時間，這個參數可以不寫。

■ **class**：目前只能填 IN，代表 internet。

■ **type** ：resource record type 有很多種，較常用的 resource record type 如下：

SOA：Start Of Authority，Zone File 中的第一筆 Resource Record 定是 SOA Record，這筆 Record 中記錄 DNS 內容的版本（是內容的版本不是指 DNS 版本，例如用 2005070601 代表是 2005 年 7 月 6 日第一次修正的版本），還有 Master DNS 和 Slave DNS 溝通的條件。
NS：name server，定義某個 domain 是由哪個 name server 負責。
A：address，定義某個主機名稱（FQDN）所對應的 IP。
PTR：pointer，定義某個 IP 對應的主機名稱（FQDN）。
CNAME：canonical name，定義一個別名及其真正對應到的 record。
MX：mail exchanger，定義某部機器的 mail exchanger，所有要送往那部機器的 mail 都要經過 mail exchanger 轉送。

筆者在 Zone File 中加入 3 筆 Resource Record，這是扮演 Master DNS 最基本的三筆 Record。其中 SOA Record 的設定較複雜，說明如下：

【SOA Record】

```
@                IN SOA    dns.geeko.idv.tw.    neo.geeko.idv.tw. (
                           2005070200           ; serial
                           3h                   ; refresh（3 小時）
                           1h                   ; retry（1 小時）
                           1w                   ; expiry（1 週）
                           1d )                 ; minimum（1 天）
```

SOA record 其中@這個符號是縮寫，代表 named.conf 中這個 zone file 所對應的 zone。以這個例子來說就是 geeko.idv.tw。

SOA 後面的兩個參數是指這個 zone file 是在哪部主機定義的，以及這個 zone file 的負責人（注意格式為 neo.geeko.idv.tw.而不是 neo@geeko.idv.tw.，因為@已有其他的意義）然後是用括號括起來的 5 個參數，分別說明如下：

serial：代表這個 zone file 內容的版本（通常格式為“年月日修正”，例如用 2005070601 代表是 2005 年 7 月 6 日第一次修正的版本）），每當 zone file 內容有變動，DNS server 管理者就應該增加這個號碼，因為每隔一段時間（refresh）,，slave DNS 會本身 serial number 與 Master DNS 的 serial number 比對，若是 Master DNS 的 serial number 數字較大，即會再 copy Master DNS 的資料（即進行 zone transfer）。

refresh：slave server 每隔這段時間（預設單位：秒），就會檢查 master server 上的 serial number。

retry：當 slave server 無法和 master 進行 serial check 時，要每隔多久（預設單位：秒）重新嘗試（retry ）一次。

expire：當時間超過 Expire 所定的時間（預設單位：秒）而 slave server 都無法和 master 取得連絡，那麼 slave 將停止提供名稱解析的服務。

minimum （minimum ttl for negative answer）：對於查無此名稱的結果保存期限，例如筆者執行 host www.geeko.idv.tw 指令，但.geeko.idv.tw 的 DNS 回答 not found，這種答案稱為 negative answer，針對這種查詢結果，要求提出查詢的 DNS 暫存多久（預設單位：秒）。

```
sles9:~ # host www.geeko.idv.tw
Host www.geeko.idv.tw not found: 3(NXDOMAIN) ← negative answer
```

【NS Record 及 A Record】

至於其他兩筆 Resource Record 就比較單純，NS Record 指定 geeko.idv.tw 的 DNS 為 dns. geeko.idv.tw，而 A Record 的目的即設定 dns.blue-linux.com 的 IP 為 61.219.23.88。

geeko.idv.tw.	IN **NS**	dns.geeko.idv.tw.
dns.geeko.idv.tw.	IN **A**	61.219.23.88

步驟四：重新讀取 DNS 設定檔

然後要求 named 重新讀取設定檔即可，並利用「hostdns.geeko.idv.tw」.檢查是否能正確解析。

```
sles9:~ # rcnamed reload
Reloading name server BIND                    done
sles9:~ # host dns.geeko.idv.tw
dns.geeko.idv.tw has address    61.219.23.88
```

步驟五：新增其他的 Resource Record

筆者在建置 DNS 時，通常不會一次設定所有的 Resource Record，這樣萬一出現問題時，很難去找到那筆 Resource Record 設定錯誤。所以筆者都先將最基本的設定確認無誤後，再新增其他的 Record。如下面範例筆者新增 A、CNAME、MX 等相關 Record，並利用 host 指令測試是否正確？

sles9:~ # vi /var/lib/named/master/gooke.idv.tw.zone

```
$TTL 2d ←TTL 值定為 2 天
@                   IN SOA          dns.geeko.idv.tw.     neo.geeko.idv.tw. (
                                    2005070200          ; serial
                                    3h                  ; refresh
                                    1h                  ; retry
                                    1w                  ; expiry
                                    1d )                ; minimum
geeko.idv.tw.               IN      NS              dns.geeko.idv.tw.
dns.geeko.idv.tw.           IN      A               61.219.23.88
;加上下列的 Resource Record，注意註解一行要用「;」不可用「#」
mail1.geeko.idv.tw..        IN      A               61.219.23.89
mail2.geeko.idv.tw.         IN      A               61.219.23.90
geeko.idv.tw                IN      MX      10      mail1
geeko.idv.tw.               IN      MX      20      mail2
;當有信件要寄送至 geeko.idv.tw.（即 email 格式為 user@geeko.idv.tw）時，
;會將信件送至優先權較高（數值較小）的 mail1；當 mail1 當掉時，信件才會
;送至 mail2。所以通常 mail2 為備援的 Mail Server

station1                    IN      A               61.219.23.101
```

```
station2                IN      A       61.219.23.102
station3                IN      A       61.219.23.103
station4                IN      A       61.219.23.104
station5                IN      A       61.219.23.105
;上面 5 行具有規則性，可利用$GENERATE 變數將定檔簡化為下面的格式
;$GENERATE 1-5   station$       A       61.219.23.10$
;station $ 即表示 station1-5，將會自動展開成 5 行的 A Record
;Class 欄位不可寫入 IN
```

```
sles9:~ # rcnamed reload
Reloading name server BIND                                  done
```

```
sles9:~ # host mail1.geeko.idv.tw
mail1.geeko.idv.tw has address 61.219.23.89
sles9:~ # host mail2.geeko.idv.tw
mail2.geeko.idv.tw has address 61.219.23.90
sles9:~ # host ftp.geeko.idv.tw
ftp.geeko.idv.tw is an alias for dns.geeko.idv.tw.
dns.geeko.idv.tw has address 192.168.1.254
sles9:~ # host -t mx geeko.idv.tw
geeko.idv.tw mail is handled by 10 mail1.geeko.idv.tw.
geeko.idv.tw mail is handled by 20 mail2.geeko.idv.tw.
l sles9:~ # host station1.geeko.idv.tw
station1.geeko.idv.tw has address 61.219.23.101
sles9:~# host station2.geeko.idv.tw
station2.geeko.idv.tw has address 61.219.23.102
```

DNS 反解原理

DNS 除了提供查詢網路上「主機名稱」所對應的 IP Address 正解（Forward Lookup）的功能外，也具備將 IP Address 反推「主機名稱」的反解服務（Reverse Lookup）。

圖 9 為 DNS 反解的流程圖，假設想要知道 IP Address 211.21.98.10 的名稱為何？反解（使用 PTR Record）查詢過程跟正解流程雷同，DNS 會先檢查此 IP

是否是本身所負責的網段？快取暫存區中是否有相關之紀錄？若皆不是則將此需求傳給 root server。

root server 將需求往下丟給 arpa server（讀者可以把 arpa server 想成就是一台 DNS），然後再往下給 in-addr server，再到負責所有 IP Address 211 開頭的伺服器，依此步驟最後到負責 211.21.98 網段的伺服器，由其反解 Zone 得知 211.21.98.10 所對應的主機名稱為 www.test.com.tw。

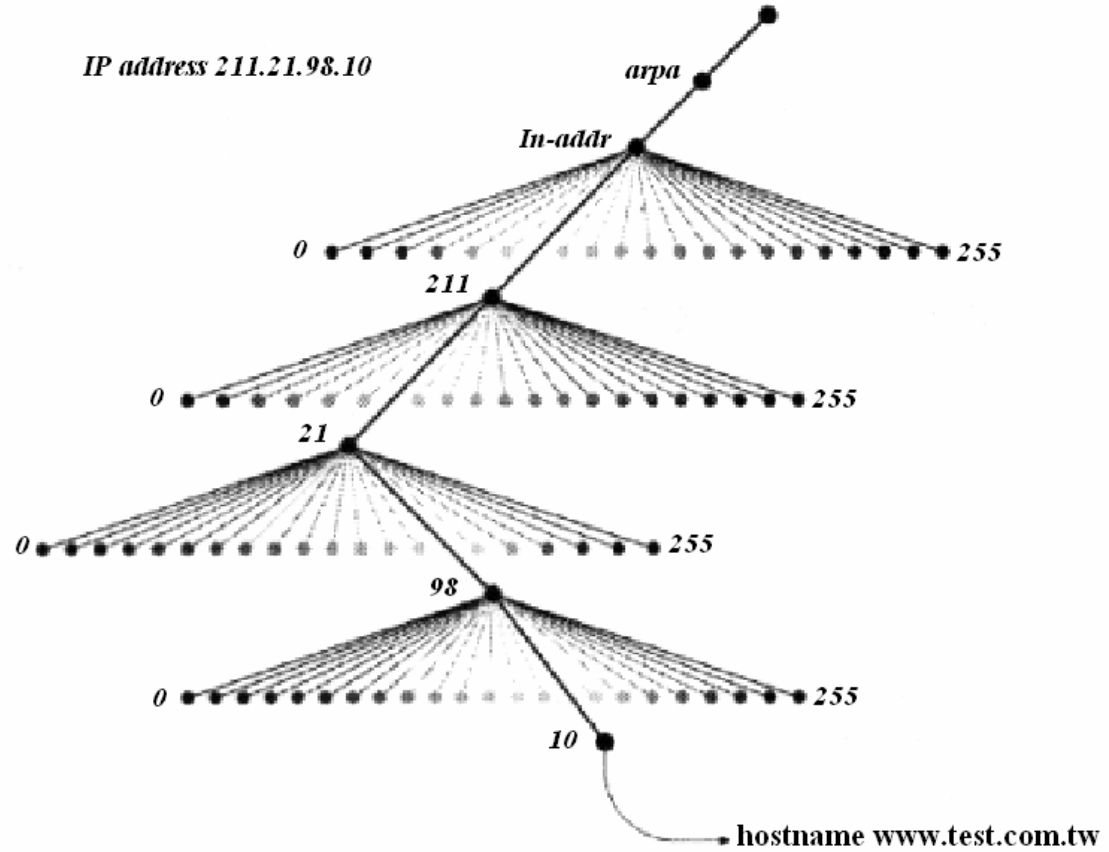

圖 9：DNS 反解（Reverse Lookup）原理圖

讀者可能會好奇，什麼時候會用到反解，電腦會作反解的行為主要有下列兩個目的：

1、讓使用者或管理者容易了解連線情形；如在畫面或 Log 中顯示主機名稱（FQDN）會比顯示 IP Address 讓人更容易了解連線對象是那個單位

2、安全考量；如某些 server 會以連線對象的 IP Address 查詢主機名稱（FQDN），再檢查主機名稱（FQDN）對應的 IP Address 是否一致。

根據 TWNIC 的調查，國內 IP 的反解比率約四成，而中國大陸不到 5%，造就國內高反解比率(與國際相較)主要因為 TANET 早期的推動及近來 ISP 的處理方式。

國內的系統較不嚴謹，通常不會檢查正反解的一致性，但國外蠻高比例的系統都會進行這個部分的確認；由來源 IP 查反解名稱，依結果再查正解，並檢驗其結果是否一致。例如，有部分的 Mail Server 也會使用正反解確認的機制來減少 SPAM 的問題。

很多人會認為網路上的服務正解的需求高於反解的需求，其實不盡然。根據調查 DNS Query 正解部份約佔 40%，反解佔 60%。原因是多數的服務皆會進行 IP 來源的反查所致，例如 telnet、ftp、WWW、MAIL、Firewall …等。

註：為什麼執行 telnet 或 ftp IP Address，會經過很久才出現登入畫面？
這個問題的通常是 /etc/resolv.conf 或 /etc/hosts 設定錯誤或 DNS 異常所造成。原因是：許多網路服務會檢查反解是否設立，通常不成立也一樣可以建立連線。但是會因為等待 DNS 回應，但 DNS 可能沒有運作，等待 timeout 而造成很久才出現登入畫面。解決方法除了在 DNS 上設定正確反解紀錄外，亦可於 hosts 加上對應紀錄，因為 hosts 也可提供反解的功能。

實戰演練：建置 Master DNS（含反解區域）

在第 7 節已實作 Master DNS，但不含反解區域。此演練假設此 DNS 亦提供反解功能，且負責的網段為 61.219.23.0/24。

步驟一：修改/etc/named.conf 加入負責網段 61.219.23.0/24 之設定

sles9:~ # vi /etc/named.conf

```
#第 7 節加入以下這段文字
zone "geeko.idv.tw" in {
        type master; ← 扮演 Master DNS
        file "master/geeko.idv.tw.zone"; ← Zone File 所在位置
};
/* 此次演練再加入下面這段文字代表負責的網段為 61.219.23.0/24，反解 Zone File 為 61.219.23.zone */
zone "23.219.61.in-addr.arpa" in {
```

```
        type master;
        file "master/61.219.23.zone";
};
include "/etc/named.conf.include";
```

步驟二：編寫反解 Zone File

在/var/lib/named/master 目錄下除了之前撰寫的 geeko.idv.tw.zone 外，再編寫名為的 61.219.23.zone 的反解 Zone File。內容如下：

sles9:~ # vi /var/lib/named/master/61.219.23.zone

```
$TTL 2d ←TTL 值定為 2 天
@                   IN SOA          dns.geeko.idv.tw.    neo.geeko.idv.tw. (
                                    2005070200          ; serial
                                    3h                  ; refresh
                                    1h                  ; retry
                                    1w                  ; expiry
                                    1d )                ; minimum
                              IN    NS      dns.geeko.idv.tw.
88.23.219.61.in-addr.arpa.    IN    PTR     dns.geeko.idv.tw.
89.23.219.61.in-addr.arpa.    IN    PTR     dns2.geeko.idv.tw.
90.23.219.61.in-addr.arpa.    IN    PTR     mail1.geeko.idv.tw.
91.23.219.61.in-addr.arpa.    IN    PTR     mail2.geeko.idv.tw.

101                           IN    PTR     61-2123-101.geeko.idv.tw.
102                           IN    PTR     61-2123-102.geeko.idv.tw.
103                           IN    PTR     61-2123-103.geeko.idv.tw.
104                           IN    PTR     61-2123-104.geeko.idv.tw.
105                           IN    PTR     61-2123-105.geeko.idv.tw.
;如果第一個欄位沒有以 . 結束的話，會自動加上.23.219.61.in-addr.arpa.
;上面 5 行具有規則性，可利用$GENERATE 變數將定檔簡化為下面的格式
;$GENERATE     1-5  10$          PTR     61-2123-10$.geeko.idv.tw.
; $ 即代表數字 1-5，上行將會自動展開成 5 行的 PTR Record
; Class 欄位不可寫入 IN
```

步驟三：重新讀取設定檔及測試反解結果

```
sles9:~ # rcnamed reload
```

```
Reloading name server BIND                                    done
sles9:~# host 61.219.23.88
88.23.219.61.in-addr.arpa domain name pointer dns.geeko.idv.tw.
sles9:~# host 61.219.23.89
89.23.219.61.in-addr.arpa domain name pointer dns2.geeko.idv.tw.
sles9:~# host 61.219.23.101
101.23.219.61.in-addr.arpa domain name pointer 61-2123-101.geeko.idv.tw.
sles9:~# host 61.219.23.102
102.23.219.61.in-addr.arpa domain name pointer 61-2123-102.geeko.idv.tw.
```

雖然在伺服器本身測試反解結果正確，此時讀者一定會有疑問？由圖 9 來看，不是應該向上一層反解 DNS 註冊，請它把 61.219.23.0/24 授權給你管理？沒錯！

所以剛剛實作的反解的部份，若上一層 DNS 未授權 61.219.23.0/24 給此台 DNS，則網路上的 DNS 是不會來詢問你這台 DNS 有關 61.219.23.0/24 反解的資訊。但在現今 IP 不敷使用的情況下，要申請到一個 Class C 談何容易！難道沒有一個 Class C，就無法設定反解嗎？

在台灣未滿一個 Class C，必須麻煩 ISP 代為設定反解紀錄，例如 HiNet 的用戶可到下列網址 http://hidomain.hinet.net/hidns.html 直接填寫相關設定，此網址並有提供申請反請範例，可參考圖 2（http://hidomain.hinet.net/hidns.html）。

圖 10：用戶申請由 HiNet 建立領域反解範例

named.conf 細項參數

接下來探討/etc/named.conf 這個 DNS 主要設定的細項參數及利用 ACL 讓 named.conf 設定更加清楚有效率。

10-1　Options 全域項目設定

options {
[directory path_name;]
Zone file 的預設存放位置（預設為/etc）
[named-xfer path_name;]
slave AXFR 存放的位置（預設為/etc）
[dump-file path_name;]
core dump 預設位置（預設為/etc）
[pid-file path_name;]
named 的 PID 存放位置
[auth-nxdomain yes_or_no;]
是否保留負面資料（預設為 no），即不正確資訊的狀況是否做快取（Cache）

[fake-iquery yes_or_no;]
作假DNS server的反解（預設為no）
[fetch-glue yes_or_no;]
不做任何的cache（預設為no）
[multiple-cnames yes_or_no;]
一個 FQDN 可否做IN CNAME 多次
[notify yes_or_no;]
Zone 變更通知（預設為yes）
[recursion yes_or_no;]
遞迴查詢，回應問的人去哪裏查（預設為yes）
[forward (only | first);]
Only代表只使用Forward first 則先使用Forward
[forwarders { [in_addr ; [in_addr ; ...]] };]
找不到使資料都往該 IP（另一台 DNS）送， 若此項有值則上一個項目預設為 first。
[check-names (master | slave | response) (warn | fail | ignore);]
檢查 FQDN 名稱不合法性於 type 為 (master | slave | 查詢要求)就(警告|失敗|忽略)
[allow-query { address_match_list };]
允許從那些 IP 查詢，可使用 ACL Name
[allow-transfer { address_match_list };]
允許從 IP AXFR（Zone Transfer），可使用 ACL Name
[listen-on [port ip_port] {address_match_list };]
DNS Listen port 為何，IP為何，不建議更改
[query-source [address(ip_addr|*)][port(ip_port|*)];]
查詢外部的 DNS(IP|*)時使用 ip_port，不建議更改
[max-transfer-time-in number;]
AXFR（Zone Transfer）的最大分鐘（預設為 120m）
[transfer-format(one-answer|many-answers);]
AXFR（Zone Transfer）時一次幾筆 RR（預設為 one-answer）
[transfers-in number;]
同時間最大的 AXFR（Zone Transfer）in 數目（預設為=10）
[transfers-out number;]
同時間最大的 AXFR（Zone Transfer）out 數目（預設為=10）
[transfers-per-ns number;]
每部 NS 同時間 AXFR（Zone Transfer）為 N 個
[version “version_string”;]

版本說明，隱藏版本有助於系統安全
[use-id-pool yes_or_no;]
每個查詢都保持一份 query ID（預設為 no），會增加系統負擔但能增加安全性
[blackhole { address_match_list };]
來自這些 IP 的查詢將不必處理
[lame-ttl number;]
不良的委任資料紀錄要保留多久秒（0~18000 不留，預設為 600）
[max-ncache-ttl number;]
負面資料的快取秒數（預設為 10800(3H) ， N<7D）

10-2 Zone 轄區設定

■Master Zone

```
zone “domain_name” {
type master; 類別為主要Master;表示權威主機
file path_name; Zone File的檔名
[ check-names ( warn | fail | ignore ); ]
[ allow-update { address_match_list }; ] 允許來自IP 的動態動新(使用
nsupdate 指令)
[ allow-query { address_match_list }; ]
[ allow-transfer { address_match_list }; ]
[ notify yes_or_no; ] 轄區資料變更是否同通知Slave主機(NS RR)
[ also-notify { ip_addr; [ ip_addr; ... ] };轄區資料變更時通知那些DNS主機( IP )
};
```

■Slave Zone

```
zone “domain_name” {
type slave;
[ file path_name; ]
masters [ port ip_port ]{ ip_addr;[ip_addr;..]};
[ check-names ( warn | fail | ignore ); ]
[ allow-update { address_match_list }; ]
[ allow-query { address_match_list }; ]
[ allow-transfer { address_match_list }; ]
[ notify yes_or_no; ]
[ also-notify { ip_addr; [ ip_addr; ... ] };
};
```

■**Forward Zone**

```
zone “domain_name” {
type forward;
[ forward ( only | first ); ]
[ forwarders { [ ip_addr ; [ ip_addr ; ... ] ] }; ]
[ check-names ( warn | fail | ignore ); ]
};
```

10-3 ACL 存取控制列表

主要在定義存取的列表，供其他”參數”所使用

```
acl acl_name {
IP; IP 位址 IP/[netmask]
DN; 網域名稱 *.blue-linux.com
path_name; 檔案名稱， 內存 ACL
CIDR; IP 段 61.219.23.0/24
None; 沒有任何 IP
Any; 任何 IP
Localhost; localhost ( 127.0.0.1)
Localnets; 網卡的IP/Netmask（即相連的網路）
```

例如:

```
acl Internal {192.168.0.0/24; 61.219.23.0/24;}
```

在別的“參數”再定義其行為

```
allow-transfer { Internal;};
allow-query { Internal;};
```

實戰演練：建置 Slave DNS

在介紹完 named.conf 的細項設定後，就可以實作 Slave DNS，因為建置 Slave DNS 的最主要工作在於修改 named.conf。

Slave DNS 功能最主要為備份 Master DNS 的資料庫，並提供名稱解析的功能。它本身也有網域的 Zone File，不過它的 Zone File 是向 Master DNS 複製（Zone Transfer）而來的。所以在 Slave DNS 的設定中最主要有兩件事：1.設定 DNS Type 為 Slave。2.指定 Master DNS 的 IP。

實作環境：

作業系統：SLES 9+SP1

Master DNS：dns.geeko.idv.tw（61.219.23.88）

Slave DNS：dns2.geeko.idv.tw（61.219.23.89）

實作步驟：

步驟一：Mastr DNS /etc/named.conf

經過前面的實例演練後，我們已成功架設一個含反解功能的 Master DNS，筆者將/etc/named.conf 的說明文字去除後，內容如下：

```
options {
        directory "/var/lib/named";
        dump-file "/var/log/named_dump.db";
        statistics-file "/var/log/named.stats";
        listen-on-v6 { any; };
        notify no;
};

zone "." in {
        type hint;
        file "root.hint";
};

zone "localhost" in {
        type master;
        file "localhost.zone";
};

zone "0.0.127.in-addr.arpa" in {
        type master;
        file "127.0.0.zone";
};

zone "geeko.idv.tw" in {
        type master;
        file "master/geeko.idv.tw.zone";
};
```

```
zone "23.219.61.in-addr.arpa" in {
        type master;
        file "master/61.219.23.zone";
};
include "/etc/named.conf.include";
```

步驟二：

從 Master DNS copy 其/etc/named.conf 至 Slave DNS，然後進行修改。將 type 由 master 改為 slave，並指定 masters 為 61.219.23.88。

```
#dns2:~ # scp 61.219.23.88:/etc/named.conf /etc
```

```
options {
    directory "/var/lib/named";
};

zone "." in {
    type hint;
    file "root.hint";
};

zone "localhost" in {
    type master;
    file "localhost.zone";
};

zone "0.0.127.in-addr.arpa" in {
    type master;
    file "127.0.0.zone";
};

zone "geeko.idv.tw" in {
    ~~type master;~~
    type slave;
    masters { 61.219.23.88;};
```

```
      file "master/geeko.idv.tw.zone";
      file "slave/geeko.idv.tw.zone-slave";
};

zone "23.219.61.in-addr.arpa" in {
            type master;
            type slave;
            masters { 61.219.23.88;};
            file "master/61.219.23.zone";
            file "salve/61.219.23.zone-slave";
};
include "/etc/named.conf.include";
```

步驟三：重新啟動 named 並檢查 Zone Transfer 是否成功

```
dns2:~ # ls -l /var/lib/named/var/lib/named/slave/←原本此目錄下無任何檔案
total 0
drwxr-xr-x    2 named named    48 Jul    6 00:59 .
drwxr-xr-x    9 root    root    312 Jul    6 00:54 ..
dns2:~ # rcnamed restart
Shutting down name server BIND           done
Starting name server BIND       done
dns2:~ # ls -l /var/lib/named/var/lib/named/slave/←多出 2 個 Zone File
total 8
drwxr-xr-x    2 named named 128 Jul    6 01:00 .
drwxr-xr-x    9 root    root    312 Jul    6 00:54 ..
-rw-------    1 named named 593 Jul    6 01:00 61.219.23.zone-slave
-rw-------    1 named named 454 Jul    6 01:00 geeko.idv.tw.zone-slave
```

/var/lib/named/var/lib/named/slave/目錄下多了兩個 Zone File，分別為向 Master DNS 做 Zone Transfer 而得的網域正解資料存為「**geeko.idv.tw.zone-slave** 」及網域反解則存為「**61.219.23.zone-slave**」。

基本實用 Master DNS 設定

介紹那麼多 DNS 相關內容及檔案，我想讀者可能有點眼花瞭亂了，你可能會想

說我只是要架個 DNS 伺服器，有需要搞得這麼複雜；可不可以利用可愛的 YaST 來完成上述的工作。筆者有些想法跟大家分享，很多接觸 Linux 的使用者鮮少第一個接觸的網路伺服器便是「DNS」。但筆者覺得 DNS 是所有網路伺服器的基礎。

讀者一定會這樣想：「不會吧！怎麼會是基礎？光是要讓它可以運作就這麼麻煩！隨便一個網路伺服器都比 DNS 容易設定」。其實筆者認為 DNS 是所有網路伺服器的基礎，不是指它的設定簡單，相反地 DNS 的原理與設定有一定的難度。筆者之所以稱「DNS 是所有網路伺服器的基礎」。仍是因為很多網路伺服器皆需有 DNS 的存在才能正常運作。甚者，像 Mail Server 可能會因為 DNS 異常，造成信件始終無法正確地送達目的地。

至於為什麼沒有介紹利用 YaST 來設定，經筆者測試，雖說 YaST 是個強大的管理工具，不過在設定網路相關服務時，如果你按照順序，完全沒有出錯，是可以很快可以設定完成。但若是中間稍出差錯，例如設定到一半，發現網域名稱打錯，回上一步修改，可能會造成資料不一致的問題。而且這些網路服務不論在 SUSE Linux、RedHat Linux、IBM AIX 文字檔設定的方式幾乎一模一樣，筆者希望讀者看完這些網路伺服器的章節，到別的平台仍能“吾道一以貫之”，所以網路服務部份，筆者會偏向介紹利用手動編輯設定檔的方式來建置伺服器。

最後，筆者整理架設一個基本的 Master DNS 的相關設定檔，並在其中利 ACL 的機制來讓 named.conf 更有彈性，並加入一些 DNS 安全機制的控制項。

Master DNS 設定檔及 Zone File（管理 geeko.idv.tw 網域和 61.219.23.0/24）

設定檔：
/etc/named.conf
Zone File：
網域正解檔（自行撰寫）：/var/lib/named/master/geeko.idv.tw.zone
網域反解檔（自行撰寫）：/var/lib/named/master/61.219.23.zone
本機正解檔（bind 套件提供）
本機反解檔（bind 套件提供）
root server 資訊檔（bind 套件提供）

■ /etc/named.conf 基本範例

```
// 定義 ACL(s)
acl geekoNetwork { 61.219.23.0/24; };
```

```
acl internal    { 127.0.0.1; 192.168.0.0/24; 192.168.1.0/24; };

options {
            // zone files 存放位置
      directory "/var/lib/named";

      //只接受公司內部 IP 的名稱查詢要求,
      //不過如果是外部 DNS 通常都採預設值,
      //接受所有 hosts 的名稱查詢要求。
      #allow-query { internal; };

      //只幫公司內部 IP 做 recursion 查詢
      allow-recursion { internal; };

      //只允許 geekoNetwork 內的主機可以做 Zone Transfer
      allow-transfer { geekoNetwork; };
};

//////////////////////////////////////////////////////////////
//                提供 root nameservers 的資訊                //
//////////////////////////////////////////////////////////////
zone "." in {
      type hint;
      file "root.hint";
};

//////////////////////////////////////////////////////////////
//                      本機正解/反解檔                       //
//////////////////////////////////////////////////////////////
zone "localhost" in {
      type master;
      file "localhost.zone";
};
```

■ **網域正解檔：/var/lib/named/master/geeko.idv.tw.zone 基本範例**

```
$TTL 2d ←TTL 值定為 2 天
```

```
@                   IN SOA          dns.geeko.idv.tw.    neo.geeko.idv.tw. (
                                    2005070201          ; serial
                                    3h                  ; refresh
                                    1h                  ; retry
                                    1w                  ; expiry
                                    1d )                ; minimum
geeko.idv.tw.               IN NS           dns.geeko.idv.tw.
dns.geeko.idv.tw.           IN A            61.219.23.88
```

■ **網域反解檔：/var/lib/named/master/61.219.23.zone 基本範例**

```
$TTL 2d
@                   IN SOA          dns.geeko.idv.tw.    neo.geeko.idv.tw. (
                                    2005070201          ; serial
                                    3h                  ; refresh
                                    1h                  ; retry
                                    1w                  ; expiry
                                    1d )                ; minimum
                            IN  NS      dns.geeko.idv.tw.
88.23.219.61.in-addr.arpa.  IN  PTR     dns.geeko.idv.tw.
```

重點整理

■ 簡述 DNS Server 的用途及特性

DNS 的工作就是提供將主機名稱對應到 IP Address（正解，Forward Lookup）及利用 IP Address 來反推「主機名稱」的服務（反解，Reverse Lookup）。DNS 具有以下特性：

1.全球最大的分散式資料庫系統。

2.自己的資料由自己維護，而其他人的資料則分散在全球。

3.沒有一台 DNS Server 會有全部名稱解析的資料。

4.以樹狀結構的方式找到目的位址（每個節點需將被授權）。

■ 簡述 DNS Server 的類型

DNS Server 的類型可以分為以下三種：

Master DNS：

本身含有 Domain 的資料庫（Zone），此資料庫其實就是包含正解紀錄或者是反解紀錄的文字檔（Zone File）。

Slave DNS：

這種類型的 DNS 功能最主要為備份 Master DNS 的資料庫，並提供名稱解析的功能。它本身也有網域的 Zone File，不過它的 Zone File 是向 Master DNS 複製（Zone Transfer）而來的。

Caching-only DNS：

Caching-Only DNS 沒有 Domain 資料庫，單純僅幫助 Client 端向外部的 DNS 主機要求資料，然後再保留查詢結果至快取暫存區（Cache）。則下次 Client 再提出名稱查詢的需求，若 TTL 還未過期，就直接檢查快取暫存區（Cache），不用再去詢問另一台 DNS。

■ 簡述 Resource Record 種類

SOA（Start Of Authority）Resource Record：

Zone File 中的第一筆 Resource Record 定是 SOA d，這筆 Record 中記錄 DNS 內容的版本（是內容的版本不是指 DNS 版本，例如用 2005070601 代表是 2005 年 7 月 6 日第一次修正的版本），還有 Master DNS 和 Slave DNS 溝通的條件。

NS（Name Server）Resource Record：
定義某個 domain 是由哪個 name server 負責。

A（Address）Resource Record：
定義某個主機名稱（FQDN）所對應的 IP。

PTR（Pointer）Resource Record：
定義某個 IP 對應的主機名稱（FQDN）。

CNAME（Canonical NAME）Resource Record：
定義一個別名及其真正對應到的 record。

MX（Mail eXchanger）Resource Record：
定義某部機器的 mail exchanger，所有要送往那部機器的 mail 都要經過 mail exchanger 轉送。

■ 為什麼實務上架設反解 DNS 通常沒有作用

在台灣除非申請到 Class C 的網域，反則若上一層的反解 DNS 並不會未授權給此你的 DNS 管理反解區域，所以自行架設的反解 DNS 是不會作用的。在台灣未滿一個 Class C，可以麻煩 ISP 代為設定反解紀錄！

彥明有感：
「和自己比賽，不要有求勝的心」在某片電影看到這句話，算是老生常談的一句話了！當時電影快到片尾時看到男主角的父親用這句話鼓勵他時，那時覺得編劇了無新意。沒想到隔天在個 Linux 教學場合中，回收的問卷中，大部份學生可接受我的授課方式，不過有個學生認為仍「有待改進」，比不上另一位教師。心中有點沮喪，覺得沒能讓所有人均接受自己的授課方式，也對那位學生不好意思，他花了那麼多時間，卻未得到他想要的收獲。

走在台北東區，想著那位學生期待的教法是如何？另一位老師有那些地方值得學習？此時，腦中竟浮現「和自己比賽，不要有求勝的心」這句話，思索著到底自己要成為每個人都稱道的人，還是應該「和自己比賽」！

雖然長輩時常告訴我們不要跟別人比較，應和自己比較。但你會發現有時現實的

環境不由得你不和他人一較高下。要不和人比較，很難！要讓眾人皆認同，很難！要做只和自己比賽的人，也很難！

批評的話，雖然難聽，但表示自己有需要改進的地方，不是嗎？別人的批評，固然要放在心上，不過也要記住要「和自己比賽」而不是和別人求勝。筆者這樣認為，你呢？

2005/07/11 Alex Lin（林彥明）