Enterprise Linux 實戰講座

Domain Name System　網域名稱伺服器（一）

筆者最近參加 IBM 2004『Speed-start Linux 應用程式優勢技術研討會』，會後有些想法。Linux 作業平台對現今企業來說，是一個比微軟更優越且符合成本效益的選擇，但如何從 Windows 環境轉換 Linux 環境，對很多資訊人員卻是一大難題。所以筆者未來的文章，打算介紹在 Windows 常見的 Server，例如 DNS、Mail Server、File Server（SAMBA）、Directory Service（LDAP）…等，學著如何在 Linux 上建置這些 Server。

前言

DNS 全名是 Domain Name System，透過 DNS 系統，我們可以由一部機器的「主機名稱（hostname）」查其「IP Address」，也可以由機器的「IP Address」反查它的「主機名稱（hostname）」，除此之外 DNS 還可與 Mail System 結合，提供 Mail routing 的功能。

早期這個「主機名稱（hostname）」與 IP 的對應表是記在每部機器的/etc/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；因為用 hosts 記錄有問題，所以後來發展出 DNS。

Linux 名稱解析相關檔案

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「**.**」（**註：Linux 的主機名稱存放於 /etc/sysconfig/network**），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**」。附帶完整網域名稱的名字稱 「**FQDN**」。

註：在 RHEL 中，建議主機名稱使用 FQDN 格式

一般作業系統名稱解析的順序，都會設成先讀 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 IP 寫入此檔案中。

在 Unix-Base 的環境，hosts 位於 /et 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；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。

那麼對 Linux 而言，如果/etc/hosts 找不到的對應的記錄，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。

在某台 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，當然也可以，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 38 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
37 #hosts:      db files nisplus nis dns
38 hosts:       files dns
```

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 timeout，而造成很久才出現登入畫面。解決方法除了在 DNS 上設定正確反解記錄外，亦可於 hosts 加上對應記錄，因為 hosts 也可提供反解的功能。

DNS 正解名稱原理

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，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。
- 沒有一台電腦會有全部的 DNS 資料。
- 以樹狀結構的方式找到目的位址（每個結點需將被授權）。

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」，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 # **host 209.132.177.50** **50.177.132.209.in-addr.arpa domain name pointer www.redhat.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**等使用。

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.redhat.com 的 IP 為何？DNS 到底如何運作而得到 www.redhat.com 的 IP？

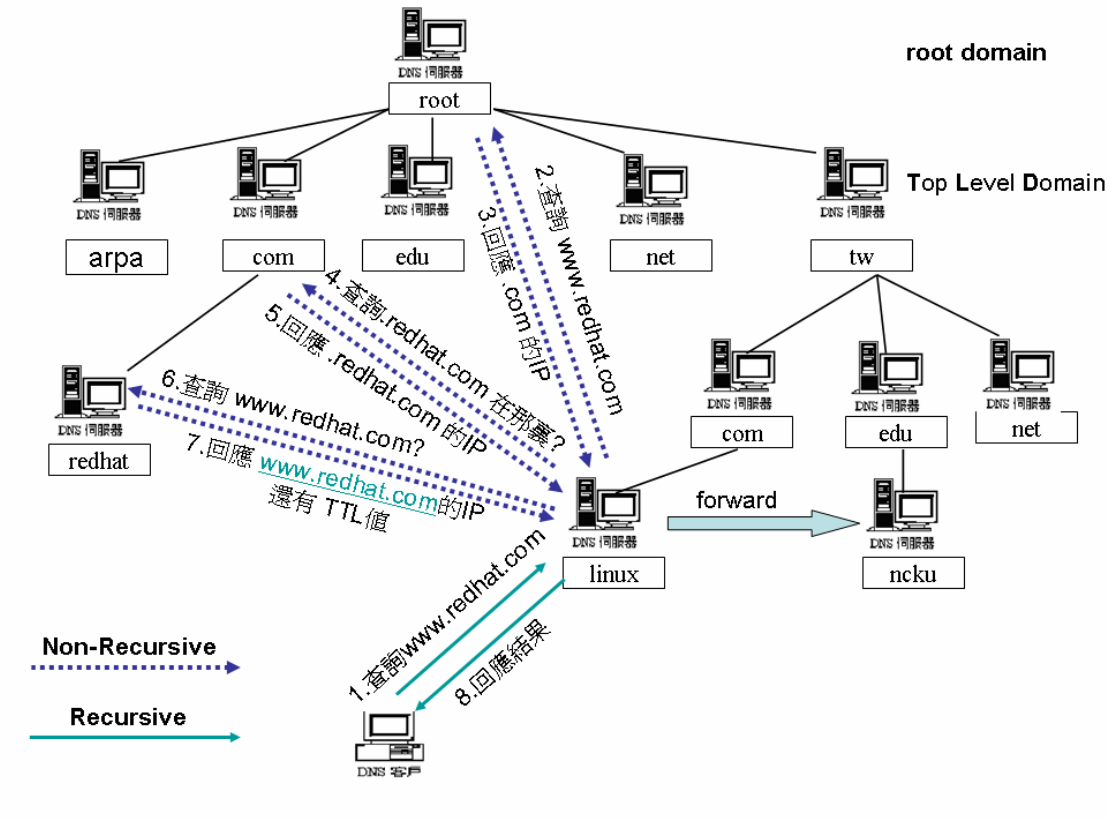

圖 1：DNS 正解解析流程圖

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.redhat.com ），若您的 DNS 是管理這個網域名稱（ redhat.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（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 位址記錄在快取暫存區中。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記錄資料，傳回給用戶端，加速用戶端對名稱查詢的速度，而您的 FQDN 資料要被暫存多久，則是由您的記錄上的 TTL 欄位決定。

如果名稱伺服器在資料記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 name server 剛啟動時，cache 中是空的，除了/var/named/named.ca 檔案中所定義的十三部 root server 外沒有其他的資料，所以一開始一定要向這十三部 root server 之一發出查詢請求。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 記錄）。

為什麼不提供「**Recursive Query**」？其原因在於遞迴主機會 Cache 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 Cache（不會代查別人的就不會 cache），只會回應自己的 Domain 的資料及 root server list，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 linux.com.tw 的 DNS 查到 www.redhat.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）。

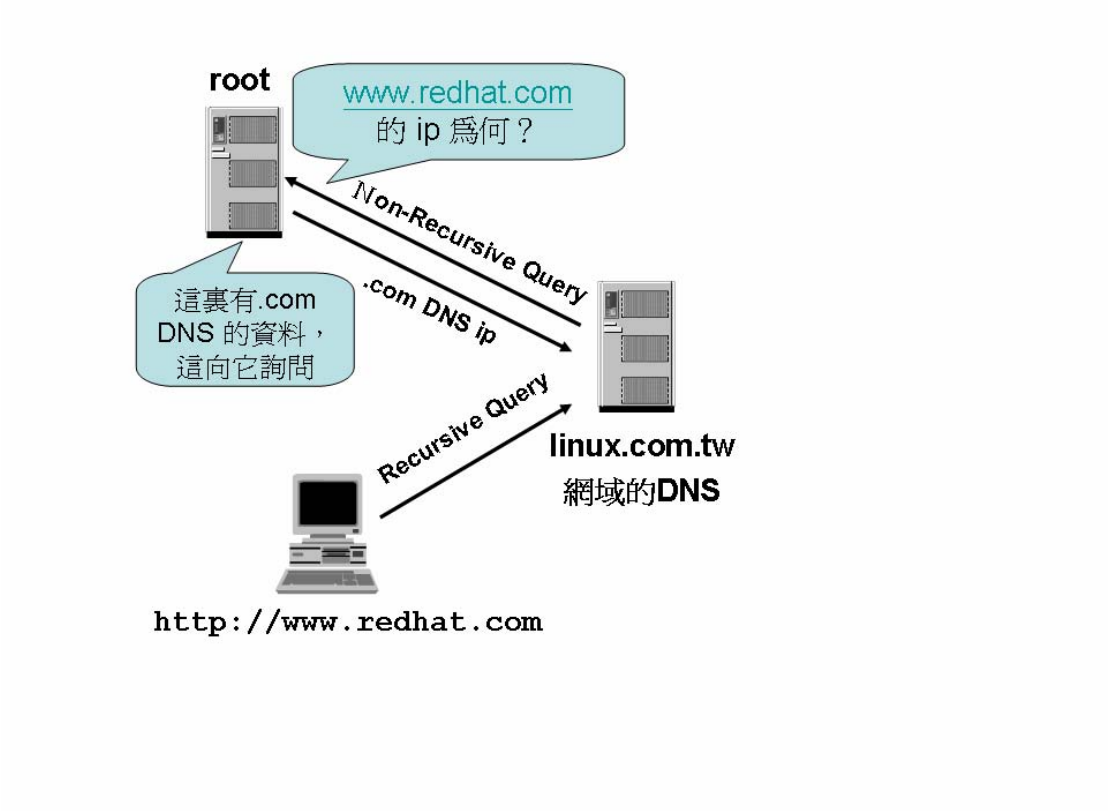

圖 2：Recurive 和 Non-Recursive Query

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

步驟六~七：redhat.com 的 DNS 回應 www.redhat.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 linux.com.tw 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 linux.com.tw 的 DNS www.redhat.com 的 IP 為何，在保存期限內，它就直接回應 www.redhat.com 的 IP 為何，而不會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）。
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。

實戰演練一：建置 Caching-Only DNS（不指定 forwarder）

因為 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，所以我們無需去申請 Domain，便可架設 caching-only DNS。

實作環境:RHEL 3

註：RHEL 4 若關掉 chroot 機制，則作步驟亦同 RHEL 3
#vi /etc/sysconfig/named
在 ROOTDIR=/var/named/chroot 前加上#，然後#service named restart 即可。若不關閉此功能，則所有設定檔皆需放在/var/named/chroot 目錄下，就是把/var/named/chroot 目錄想成/ 目錄。所以在 RHEL 3 上要修改/etc/named.conf；在 RHEL 4 上就得修改/var/named/chroot/etc/named.conf。

步驟一：安裝 bind

跟 DNS 相關套件如下：

bind-utils-9.2.2-21
bind-9.2.2-21
redhat-config-bind-2.0.0-14

caching-nameserver-7.2-7

■ bind-utils-9.2.2-21 內為 host、dig、nslookup 等 DNS 查詢必備工具及 DNS 所需的 library。
■ bind-9.2.2-21 內為 BIND 9.2.2 主要程式。
■ redhat-config-bind-2.0.0-14 為圖形化的 DNS 設定工具。
■ caching-nameserver-7.2-7 提供設定 Caching-Only DNS 所需的設定檔。

讀者可利用「**rpm –ivh 套件檔案名稱**」指令安裝 DNS 必要套件「bind-utils-9.2.2-21」及「bind-9.2.2-21」，或是執行「**redhat-config-packages**」勾選「名稱伺服器」（如圖 3），安裝相關套件。

```
[root@dns RPMS]# rpm -ivh bind-utils-9.2.2-21.i386.rpm
warning: bind-utils-9.2.2-21.i386.rpm: V3 DSA signature: NOKEY， key ID db42a60e
Preparing...                ########################################### [100%]
   1:bind-utils             ########################################### [100%]
[root@dns RPMS]# rpm -ivh bind-9.2.2-21.i386.rpm
warning: bind-9.2.2-21.i386.rpm: V3 DSA signature: NOKEY， key ID db42a60e
Preparing...                ########################################### [100%]
   1:bind                   ########################################### [100%]
```

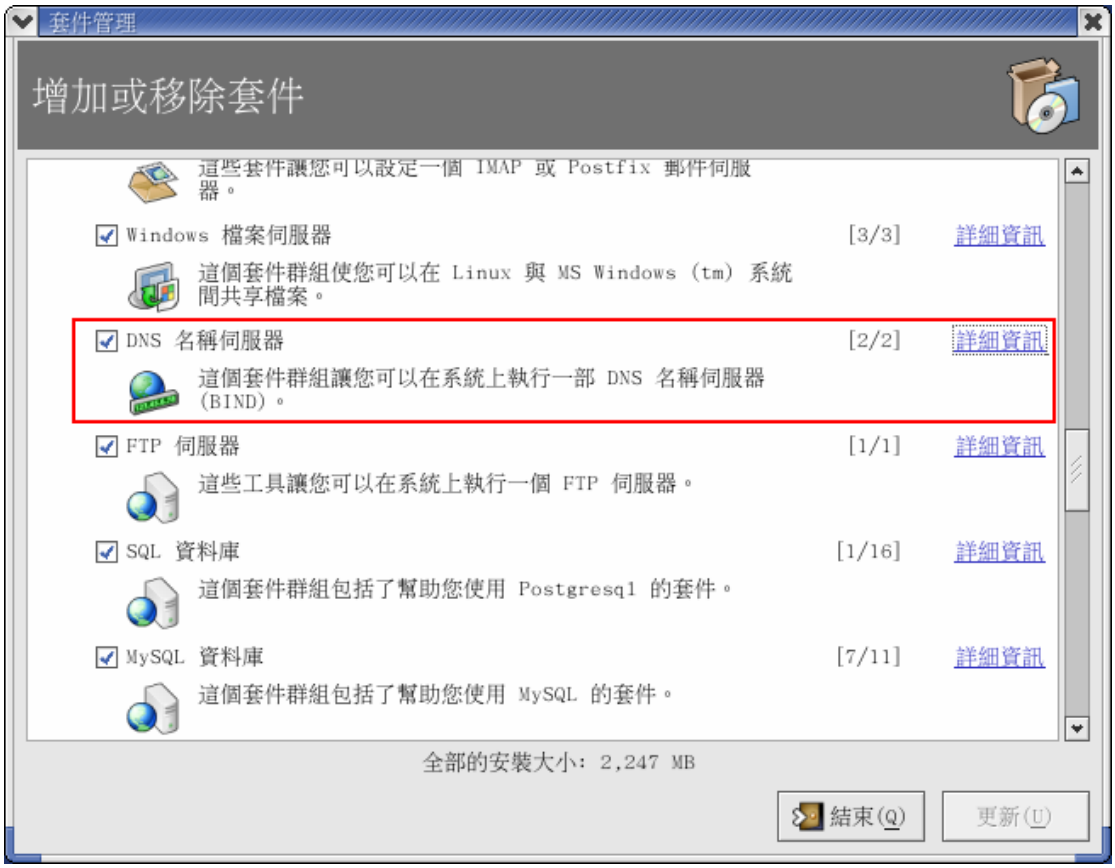

圖 3：redhat-config-packages 設定畫面

步驟二：安裝 caching-nameserver rpm

檢查 caching-nameserver-7.2-7.noarch.rpm 提供那些檔案

```
[root@dns RPMS]# rpm -qlp caching-nameserver-7.2-7.noarch.rpm
warning: caching-nameserver-7.2-7.noarch.rpm: V3 DSA signature: NOKEY， key ID db42a60e
/etc/named.conf
/usr/share/doc/caching-nameserver-7.2
/usr/share/doc/caching-nameserver-7.2/Copyright
/var/named/localhost.zone
/var/named/named.ca
/var/named/named.local
```

安裝 caching-nameserver-7.2-7.noarch.rpm

```
[root@dns RPMS]# rpm -ivh caching-nameserver-7.2-7.noarch.rpm
warning: caching-nameserver-7.2-7.noarch.rpm: V3 DSA signature: NOKEY， key ID db42a60e
Preparing...                ########################################### [100%]
   1:caching-nameserver     ########################################### [100%]
```

通常Master DNS通常需有6個設定檔,1個DNS的主要組態檔及5個Zone File。

```
/etc/named.conf：DNS 的主要組態檔
/var/named/db.正解：網域正解檔
/var/named/db.反解：網域反解檔
/var/named/localhost.zone：本機正解檔
/var/named/named.local：本機反解檔
/var/named/named.ca ：root server 資訊檔
```

但預設安裝「bind-utils-9.2.2-21」及「bind-9.2.2-21」並不會有這些檔案，以前這些檔案都必須自行撰寫，但現在只要安裝「caching-nameserver」rpm。它會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。因為這個 rpm 的目的就是為了幫助使用者快速建置 Caching-Only DNS。

```
/etc/named.conf：DNS 的主要組態檔
/var/named/localhost.zone：本機正解檔
/var/named/named.local：本機反解檔
/var/named/named.ca ：root server 資訊檔
```

步驟三：啟動 DNS

利用 service named start 指令便可立即啟動 DNS。

```
[root@dns root]# service named start
啟動 named:                                                [  確定  ]
```

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。

```
[root@dns RPMS]# ifconfig eth1
```

```
eth1      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

[root@dns RPMS]# cat /etc/resolv.conf
nameserver 61.219.23.88 ← 自己本機的 IP
[root@dns RPMS]# 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 正常運作
```

實戰演練二：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 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。修改方式如下：

```
[root@dns root]# vi /etc/named.conf
// generated by named-bootconf.pl

options {
```

```
        directory "/var/named";
        forwarders {168.95.1.1;};
        forward only;只要加上這兩行，其餘皆不需修改
        /*
         * If there is a firewall between you and nameservers you want
         * to talk to， you might need to uncomment the query-source
         * directive below.  Previous versions of BIND always asked
         * questions using port 53， but BIND 8.1 uses an unprivileged
         * port by default.
         */
        // query-source address * port 53;
};

//
// a caching only nameserver config
//
controls {
        inet 127.0.0.1 allow { localhost; } keys { rndckey; };
};
zone "." IN {
        type hint;
        file "named.ca";
};

zone "localhost" IN {
        type master;
        file "localhost.zone";
        allow-update { none; };
};

zone "0.0.127.in-addr.arpa" IN {
        type master;
        file "named.local";
        allow-update { none; };
};

include "/etc/rndc.key";
```

然後要求 named 重新讀取組態檔 /etc/named.conf 即可。

```
[root@dns root]# service named reload
重新載入 named:                                            [  確定  ]
[root@dns root]# host www.redhat.com
www.redhat.com has address 209.132.177.50
```

實戰演練三：建置 Master DNS

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。

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。例如筆者在 http://www.danow.com 申請「**blue-linux.com**」網域。

其實所謂註冊的動作，便是在上一層 DNS 設定一筆 NS Record，表示負責「**blue-linux.com**」網域的 DNS 為 dns.blue-linux.com 及一筆 A Record 內容為 dns.blue-linux.com 其 IP 為 61.219.23.88。通常會要求你得設定兩台 DNS，筆者因為只有一台 DNS，所以將第二台 DNS 也指向同一台機器。（圖 4）

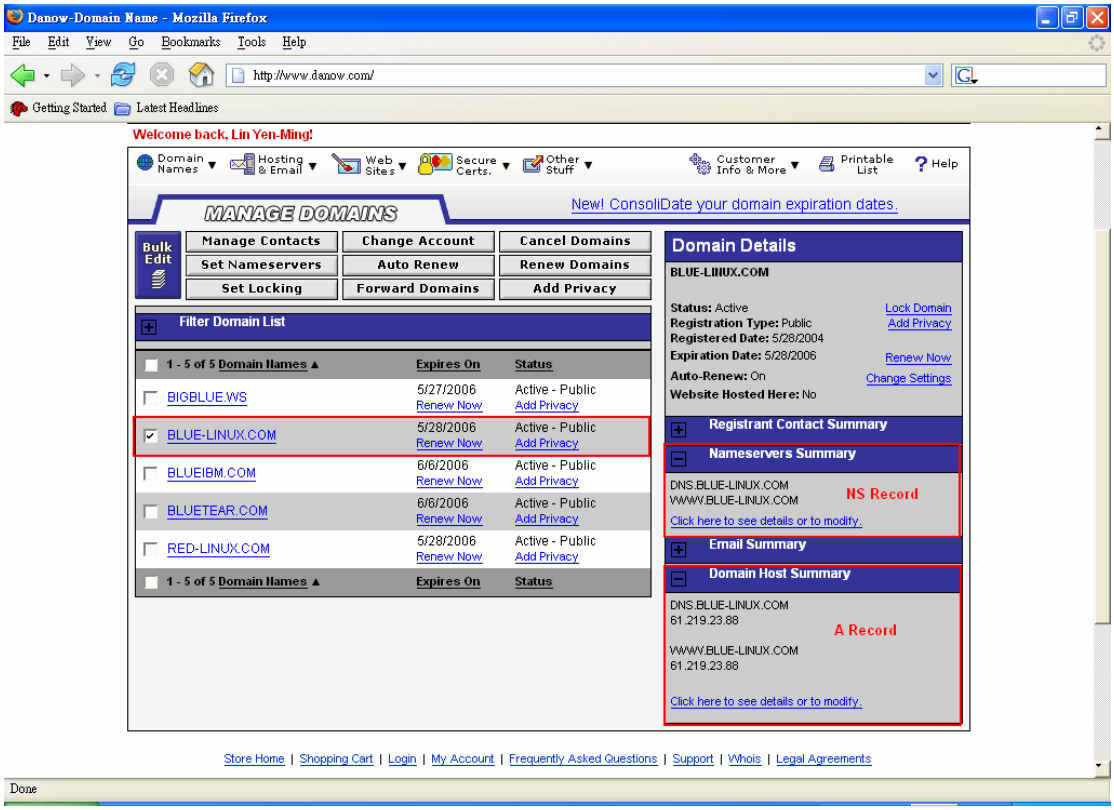

圖 4：www.danow.com 設定畫面

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「blue-linux.com」及網域正解檔（Domain Zone File）的存放位置。

```
[root@dns root]# vi /etc/named.conf
// generated by named-bootconf.pl

options {
        directory "/var/named";
        forwarders {168.95.1.1;};
        forward only;將這兩行刪除
        /*
         * If there is a firewall between you and nameservers you want
         * to talk to， you might need to uncomment the query-source
         * directive below.  Previous versions of BIND always asked
         * questions using port 53， but BIND 8.1 uses an unprivileged
         * port by default.
         */
        // query-source address * port 53;
```

```
};

//
// a caching only nameserver config
//
controls {
        inet 127.0.0.1 allow { localhost; } keys { rndckey; };
};
zone "." IN {
        type hint;
        file "named.ca";
};

zone "localhost" IN {
        type master;
        file "localhost.zone";
        allow-update { none; };
};

zone "0.0.127.in-addr.arpa" IN {
        type master;
        file "named.local";
        allow-update { none; };
};
```

加入以下這段文字

```
zone "blue-linux.com" {
        type master;
        file "db.blue-linux";
};

include "/etc/rndc.key";
```

步驟三：新增網域正解 Zone File（/var/named/db.blue-linux）

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：

```
[root@dns root]# vi /var/named/db.blue-linux
$TTL 86400
@ IN    SOA dns.blue-linux.com. alex.blue-linux.com.  （
```

```
                                2001101100      ; serial number
                                10800           ; refresh
                                3600            ; retry query
                                604800          ; expire
                                0 )             ; minimum TTL

blue-linux.com.         IN      NS      dns.blue-linux.com.
dns.blue-linux.com.     IN      A       61.219.23.88
```

所有要被查詢的資料都是在 Zone File，Zone File 的內容除了少數的控制命令外，都是一筆一筆的 Resource Record（有時也稱 RR）。

Resource Record 的表示法為

```
[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，這種 record 放在 zone file 一開始的地方，描述這個 zone 負責的 name server、 version number、maintainer 資料，以及當 slave server 要備份這個 zone 時的一些參數。
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 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 blue-linux.com。

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（注意是寫成 alex.blue-linux.com 不是 alex@blue-linux.com）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，每當 zone file 內容有變動，name server 管理者就應該增加這個號碼，因為 slave 會將這個號碼與其 copy 的那份比對，以便決定是否要再 copy 一次（即進行 zone transfer）。

refresh

slave server 每隔這段時間（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，要每隔幾秒 retry 一次。

expire

當時間超過 Expire 所定的秒數而 slave server 都無法和 master 取得連絡，那麼 slave 會刪除自己的這份 copy。

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

```
bule-linux.com.        IN    NS    dns.blue-linux.com.
dns.blue-linux.com.    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 host dns.blue-linux.com 檢查是否能正確解析。

```
[root@dns root]# service named reload
重新載入 named:                                    [  確定  ]
[root@dns root]# host dns.blue-linux.com
dns.blue-linux.com 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 R.R.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？

```
root@dns root]# vi /var/named/db.blue-linux

$TTL 86400
@ IN    SOA dns.blue-linux.com. alex.blue-linux.com.  （
                                        2001101100          ; serial number
                                        10800               ; refresh
                                        3600                ; retry query
                                        604800              ; expire
                                        0 ）                 ; TTL

blue-linux.com.   IN   NS   dns.blue-linux.com.
;加上下列的 Resource Record
dns.blue-linux.com.     IN      A       61.219.23.88
mail1.blue-linux.com.   IN      A       61.219.23.89
mail2.blue-linux.com.   IN      A       61.219.23.90
ftp                     IN      CNAME   dns
;如果第一個欄位沒有以 . 結束的話，會自動加上這個網域名稱；所以 ftp 相當於 ftp.blue-linux.com.
blue-linux.com.         IN      MX      10  mail1
blue-linux.com.         IN      MX      20  mail2
;當有信件要寄送至 blue-linux.com.(即 email 格式為 user@blue-linux.com)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
```

```
[root@dns root]# service named reload
重新載入 named:                                            [  確定  ]
```

```
[root@dns root]# host mail1.blue-linux.com
mail1.blue-linux.com has address 61.219.23.89
[root@dns root]# host mail2.blue-linux.com
mail2.blue-linux.com has address 61.219.23.90
[root@dns root]# host ftp.blue-linux.com
ftp.blue-linux.com is an alias for dns.blue-linux.com.
dns.blue-linux.com has address 61.219.23.88
[root@dns root]# host -t mx blue-linux.com
blue-linux.com mail is handled by 10 mail1.blue-linux.com.
```

blue-linux.com mail is handled by 20 mail2.blue-linux.com.

作者簡介

林彥明 （ Alex Lin ）： RedHat 技術顧問，現任職於 IBM Taiwan 技術支援中心，負責 Linux、AIX、WebSphere 相關技術支援工作，具有 RHCX （RedHat 認證主考官）、RHCE、NCLP、LPIC、IBM AIX Expert、MQ、SCJP、SCWCD 國際認證，曾參予建置臺灣第一套商業用 IBM 1350 Linux 叢集系統。