第三章

Hesiod / BIND server 的建置與應用

戴基峰

國立中央大學資訊管理系三年級

im800998@im.mgt.ncu.edu.tw

邱顯茂

國立中央大學資訊管理系四年級

im790874@im.mgt.ncu.edu.tw

3、1 節 簡介

Hesiod 是麻省理工學院(M.I.T)在 Project Athena 中所發展出來的一項 naming service。就像是 BIND (Berkeley Internet Name Domain) 提供 Internet 上機器 domain name service 一樣，Hesiod 也是提供 naming service，但 Hesiod 所提供的 naming service 所涵蓋的範圍更大、使用起來更具彈性。

在 M.I.T Athena 環境中，原先是以 UNIX 這類分時系統 (timing-sharing system)爲主體所構成的，但隨著計劃的不斷進行，陸陸續續加入了許多 file server以及開放給大家所使用的工作站；而機器與使用者的人數也成呈 2-3倍的成長。一般而言，在 UNIX 中，對於一些資訊的管理都是以 TEXT 檔的格式儲存在每臺機器上(譬如：利用 /etc/host 檔存放 host - IP address 對應資料；/etc/passwd 存放 username - password 對應資料)；但

隨著機器與使用者人數不斷成長，要維護這數以百計機器上的資料成爲一項相當沉重的負擔。因此，在 Athena Project 中就摹仿 Domain Name System 的方式，將整個環境的管理資訊分散由幾個 name server 集中管理，以節省個別機器維護的時間與成本。

由於 Hesiod 是在這種背景下所設計出來的，因此爲了節省發展的時間與精力，很自然的就希望在現有的 name service 基礎下繼續發展。BIND，是目前 Internet 上最有名的 name service 之一，且又有原始程式碼，因此 Project Athena 就在 BIND 的基礎上發展 Hesiod。這也是我們不稱 Hesiod Service 而稱 Hesiod / BIND Service 的緣故。

3、2 節 Hesiod / BIND Service 工作原理

由於 Hesiod Service 是建立在 BIND Service 的基礎之上，因此在說明 Hesiod Service 的工作原理之前，我們必須花些篇幅介紹 BIND Service 的運作原理。

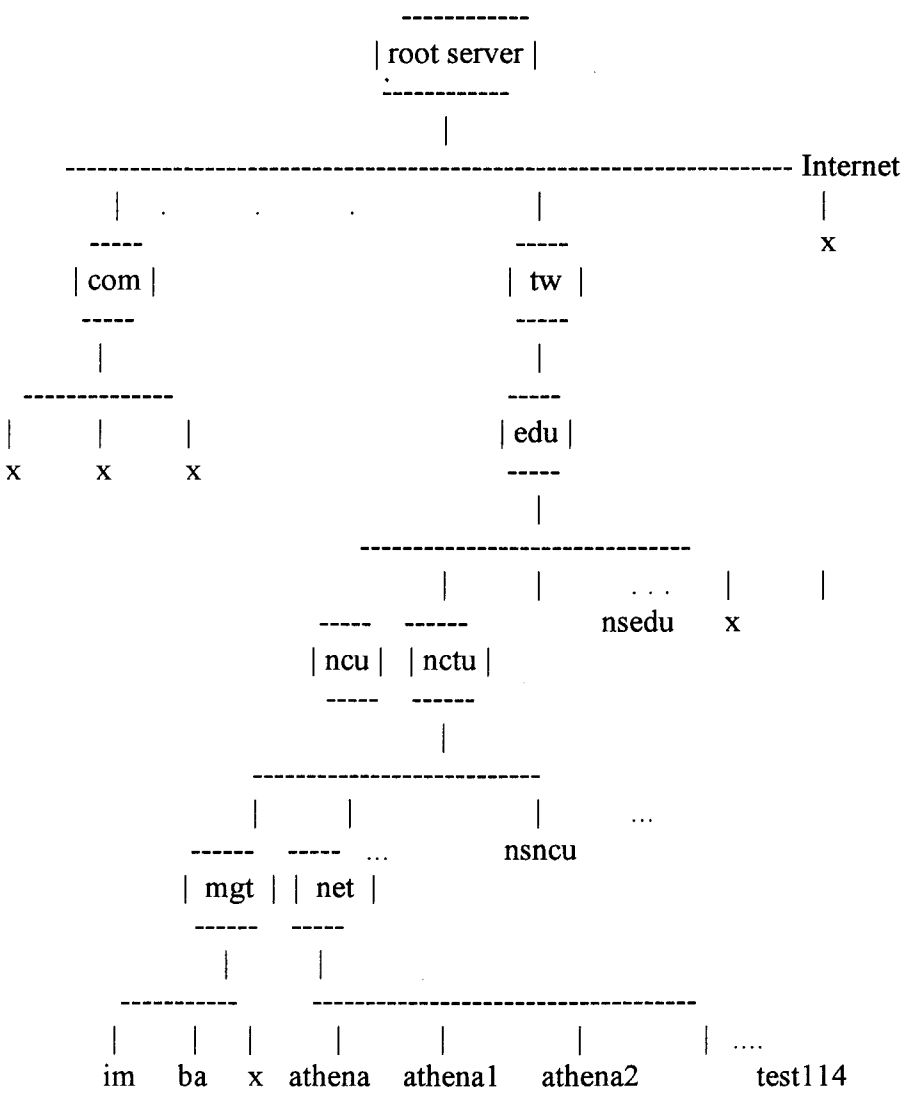
3.2.1 架構討論

如圖一所示，在 TCP/IP Internet 之下，網路上機器名稱的命名方式是使用階層式的架構加以組成的，稱作 DNS〔Domain Name System〕。在此種命名的架構下，Internet 網路下所有的機器都可以一個個 domain 的方式來加以命名及管理。以圖一來看，root server 是整個 Internet 網路上命名架構上最高的層級，其下可分出許多 sub-domain；每個 sub-domain 下包括了許多的機器，而這些機器又可以合成一個個 sub-domain 加以命名，依此類推就形成了樹狀結構的命名方式。

爲了管理網路上這麼多各式各樣的命名階層，不可能由一個 root server 來記錄所有網路上機器命名所對應 IP Address 的資訊，因此在 Internet 上，利用一個個 domain 爲基礎來加以管理。在 Internet 的命名階層上，由高到低每個 domain 都擁有至少一臺 name server，負責解答該 domain 內的機器名眞實地址。以圖一來說，假設 test114.net.ncu.edu.tw 發出 nctuccca.edu.tw 的位址求解的查詢時，該 domain 的 name server〔譬如是 athena〕就必需負責解答。若 athena 無法解決這個機器名求解時，則必需向它上一層的 domain name server 求解，若再不行的話則再往上一層送，直到 root server 或求解成功爲止。

BIND Service 就是提供符合如上述此種階層式命名架構的 naming and resolving service。

D27



註：　代表一個 sub-domain

　　x　代表機器名

圖　一

3.2.2 Name Service 的原理

基本上，BIND Service 在架構上可分成兩部分；在 Server 部份稱爲 Name Server，在 Client 部份則稱爲 Name Resolver。

我們先前說過，在每一個 domain 下至少要有一個以上的 name server 負責解答該 domain 的 name resolution，在 name server 上所執行的程式叫作 named(name daemon)；而不是該 domain name server 的其它機器則必需有 name resolver。name resolver 是一組 library call，提供網路上其它機器向 domain name server 查詢的介面。

一般而言，domain name server 可以分爲四個種類：

master server：顧名思義，primary server 是主要負責該 domain name service 的 server。master server 又可分作 primary server 及 secondary server。其中，primary server 負責整個 domain datafile maintain 及 name resolution 工作；secondary server 則扮演 backup server 的角色，但若獲得 primary server 授權亦可負起 name service 的工作。

cache server：只有 cache 功能，並無自行維護資料。

forward server (forwarder)：能夠連上 Internet 網路的 name server。由於 forwarder 能夠存取到 Internet 上其它的節點，因此當所要查詢的形式是屬於 recursive requests 時，則透過 Internet 上 root server 可以減少詢問其它 name server 的次數。一般而言，forwarder 是與 slave server 配合使用的。

slave server：slave server 無法連上 Internet，因此，將不屬於 local domain

的 name resolution request 送給 forwarder 求解，並將傳回的資訊存在 cache 裏。

有了上述的背景知識後，我們來解釋 name service 的運作方式。譬如在上頁圖一中，athena 是 net.ncu.edu.tw 的 name server；nsncu 是 ncu.edu.tw 的 name server；nsedu 是 edu.tw 的 name server。當 athena1.net.ncu.edu.tw 發出 nctuccca.edu.tw 的 name resolution request 時；首先，該 domain 的 nameserver athena 必需檢查所詢問的地址是否位於該 domain，若是：則將資料庫中的 IP address 傳回；否則，(1) 當此查詢爲 recursive resolution，則 athena 會向 nsncu、nsncu 向 nsedu 查詢，解得 nctuccca.edu.tw 的 IP address 後，傳回給 athena，而 athena 則回覆 athena1 的查詢。(2) 若爲 iteration resolution 時，則 athena 得到的是 nsedu 的 IP address，並傳給 athena1 作爲 client下次求解時的 name server 地址。

3、3 節 Name Service Implementation

3.3.1 資料檔的管理

先前我們提過，在每個 domain 下我們至少會建立一台 name server 來負責管理解答該 domain 的 resolution request，因此每台 name server 上就要維護許多的資料檔(data file)。這些資料檔的內容是由許多資源記錄所組成的，稱作 Resource Records。Resource Records 的內容包括了許多欄位〔field〕，以下是一般 Resource Record 的欄位格式：

name *ttl* *address-class* *entry-type* *entry-specific-data*

說明：

name　　domain 的命名。

ttl　　Time-To-Live 的簡寫。指資料被該 Domain Resolver Cache 的時間。

address-class　　可分成三種值：

IN：Internet Address Class

HS：Hesiod naming service data

ANY：All other types of network address

entry-type　　指明在 Resource Records 中的資料類別。包括：

A　- address

CNAME - canonical name

HINFO - host information

MB　- mail box

MG　- mail group

MINFO　- mailbox information

MR　- mail rename

MX　- mail exchanger

NS　- name server

PTR　- domain name pointer

SOA　- start of authority

TXT　- Hesiod data or text

WKS　- well know services

entry-specific-data 相應於上述 class 與 entry-type 的資料。

在 server 端，根據不同的資訊，維護不同檔案，但所有的資料檔都是由上面提到的 Resource Record 所組成的。

在了解了 domain name server 如何管理及維護該 domain 內各個主機的資訊後，我們來討論 Hesiod Service 與 BIND Service 的關係，進而解釋 Hesiod 與 BIND 如何相互配合完成資料的查詢。

3.3.2 Hesiod 與 BIND Service 的關係

先前說過，Hesiod 是架在 BIND 之上所發展出的一個 Service，而這種關係的關鍵就在於其 Resource Records 欄位內容。在 Resource Records 的欄位中有一個是 entry-type，用來指明在 Resource Records 中的資料類別。因此，由於在 Hesiod 中查詢的資料都是以字串的方式寫在的 nameserver 的資料檔中，而 Resource Record 的 entry-type 中也有 TXT 的類別，因此 Hesiod 很容易用 TXT 的 entry-type 來指明其資料內容的型別。於是乎，Hesiod 就架在 BIND 之上了。

以下就以實例來說明上面的敘述：

named.hosts example file

```
$ORIGIN      net.ncu.edu.tw.
; name      ttl     address-class   ertry-type          entry-specific-data

athena                IN                  A           140.115.10.122
athena1               IN                  A           140.115.10.123
athena2               IN                  A           140.115.10.121
```

passwd.db example file

```
$ORIGIN        passwd.net.ncu.edu.tw.
; name  ttl  address-class  ertry-type              entry-specific-data
benjamin       HS             TXT
    "benjamin:F4j8PmyA8Uubk:100:100::/home/benjamin:"
muller         HS             TXT
    "muller:4q3HB0ABRJuiE:104:100:muller:/home/mulle"
fong           HS             TXT
    "fong:c0jNirs509QbE:106:100:fong:/home/fong:/bin"
```

在上述兩個檔案中，第一行 $origin 代表目前的 domain name ，因此資料檔 passwd.db 的 name field 的資料其實會被 BIND 加上 domain name 後得到的。譬如：named.hosts 檔中 athena 其實是代表 athena.net.ncu.edu.tw ，而 passwd.db 內的 benjamin 則是 benjamin.passwd.net.ncu.edu.tw 。

當我們使用 nslookup 來查詢 athena 的 IP address 時，首先 nslookup 將 athena 加上 domain name ，成爲標準的 BIND 查詢格式，隨後透過 resolver 的介面向 name server 查詢。BIND 在查詢使用者資料時，會到 named.hosts 檔中所列的資料檔去查詢是否存在有 athena.net.ncu.edu.tw 這筆資料，若有則將其資料欄的資料傳回詢問者端，並完成查詢；否則查詢失敗。

相同的道理，當我們使用 Hesiod Service 來查詢網路上的某些資訊，

譬如 password 好了，我們必需在 named.boot 檔中告訴 hesiod server 關於 password 的資料檔在何處(在 named.boot 檔所指定的資料檔 athena.db 中，加入$INCLUDE /etc/namedb/passwd.db)，並且將資料檔的內容設定正確(如上所列的範例)。當使用 Hesiod Service 時，譬如我們利用 hesinfo (Hesiod Service 所附的測試程式) 查詢 muller 這個使用者關於密碼的資訊時（指令是：hesinfo muller passwd），Hesiod 會先呼叫他本身所提供的函數 hes_to_bind () 將 muller、passwd、及 hesiod.conf 中的 lhs、rhs 欄位連接成一個 BIND 格式的字串：muller.passwd.net.ncu.edu.tw. 。然後，透過 resolver 介面向 name server 求解。

由此，我們很明顯的看出 Hesiod Service 與 BIND 所提供的 naming service 之間的關係了！

3、4 節 Hesiod / BIND Service 的架設與安裝

3.4.1 來源

我們可以到教育部的 anonymous ftp server： moers2.edu.tw (IP：140.111.2.22) 的 /pub/uunet/packages/athena 目錄下找到 hesiod.tar.Z。這個版本的 Hesiod 是 version 1.2，是在 1990 年所公佈的。

3.4.2 系統環境

1. 硬體：

   PC 486DX-50、16M RAM、550M HD

2. 軟體：

   Operating System：FreeBSD version 4.3 Release 1.0e

Naming service：Hesiod distribution verison 1.2

BIND version 4.8

3.環境

domain：net.ncu.edu.tw.

name server： athena.net.ncu.edu.tw （140.115.10.122）

host： athena1.net.ncu.edu.tw （140.115.10.123）

domain：subnet.net.ncu.edu.tw.

name server： test114.subnet.net.ncu.edu.tw （140.115.10.114）

3.4.3 安裝過程

1.首先，我們要利用 ftp 去抓 hesiod.tar.Z。

```
ftp moers2.edu.tw
;       鍵入 username 爲 anonymous，及 E-Mail address 爲
;       password
cd /pub/uunet/packages/athena
bin
get hesiod.tar.Z
```

假設我們在 ~userhome/ 下建了一個名爲 hesiod 的子目錄，並將 hesiod.tar.Z 放入。

2.將 hesiod.tar.Z 解壓縮及還原。

```
uncompress hesiod.tar.Z
```

tar -xpvf hesiod.tar

則產生了下列目錄及檔案：

README - 說明檔，內容說明架設 Hesiod Name Server 的需求及解開此壓縮檔後應有的目錄及檔案。

INSTALL - 安裝程序說明。

doc/ - 關於 Hesiod 的文件。

examples/ - 架設 Server 時所需資料檔 (data file) 的範例。

hesiod/ - Hesiod 所提供的程式庫、測試程式及 manual page 的原始程式檔。

include - 所需用到的新版 include file。

named - 新版的 BIND 4.8 的 named 程式。

res - 新版的 Resolver C 程式庫。

3.將新版的 ~userhome/hesiod/include/arpa/nameser.h 搬至 /usr/include/arpa/ 目錄下。

cp ~userhome/hesiod/include/arpa/nameser.h /usr/include/arpa/.

注意：在蓋掉原先的檔案之前，最好確定原先的檔案是否是舊版的，最簡單的判斷方法就是比較兩個檔案的大小。

4.建立及安裝新版的 named。

注意：若系統已有安裝 BIND 4.8 版的 named，則此步驟可省略。

cd ~userhome/hesiod/named; make depend

cd ~userhome/hesiod/named; *make*

cd ~userhome/hesiod/named; *make xfer*

cd ~userhome/hesiod/named; *make install*

TroubleShooting：

在進行 make 的動作時可能會發生下列 error message，以下列出其訊息及解決方法。

· db_load.c: In function `free_sort_list':

db_load.c:924: `fnettab' undeclared (first use this function)

db_load.c:924: (Each undeclared identifier is reported only once

db_load.c:924: for each function it appears in.)

原因：程式中沒有宣告 fnettab

解決方法：在 db_load.c 檔第 922 行，加入 extern struct netinfo *fnettab;

· ns_init.c: In function `free_forwarders':

ns_init.c:602: `fwdtab' undeclared (first use this function)

ns_init.c:602: (Each undeclared identifier is reported only once

ns_init.c:602: for each function it appears in.)

原因：程式中沒有宣告 fnettab

解決方法：在 ns_init.c 檔第 922 行，加入 extern struct fwdinfo *fwdtab;

· cd res; make

cd: can't cd to res

原因：在編譯過程中，並沒有產生上述的目錄，可能是 Makefile 寫錯。

解決方法：在 Makefile 中，將 cd res; make 這行刪掉。

- db_dump.o: Undefined symbol _p_type referenced from text segment
  db_dump.o: Undefined symbol _p_class referenced from text segment

原因：沒有含入標題檔

解決方法：在 db_dump.c 中，加入 #include <resolv.h> 這行即可。

- ns_forw.o: Undefined symbol _fp_query referenced from text segment
ns_forw.o: Undefined symbol _fp_query referenced from text segment
ns_forw.o: Undefined symbol _fp_query referenced from text segment
ns_init.o: Undefined symbol _dn_skipname referenced from text segment
ns_init.o: Undefined symbol _dn_skipname referenced from text segment
ns_req.o: Undefined symbol _fp_query referenced from text segment
ns_req.o: Undefined symbol _dn_skipname referenced from text segment
ns_req.o: Undefined symbol _fp_query referenced from text segment
ns_resp.o: Undefined symbol _dn_skipname referenced from text segment
ns_resp.o: Undefined symbol _dn_skipname referenced from text segment
ns_resp.o: Undefined symbol _dn_skipname referenced from text segment
ns_resp.o: Undefined symbol _dn_skipname referenced from text segment
ns_resp.o: Undefined symbol _fp_query referenced from text segment
ns_resp.o: Undefined symbol _fp_query referenced from text segment
ns_resp.o: Undefined symbol _fp_query referenced from text segment
ns_sort.o: Undefined symbol _dn_skipname referenced from text segment

原因：沒有含入標題檔

解決方法：在 ns_forw.c、ns_init.c、ns_req.c、ns_resp.c、ns_sort.c 中，加入 #include <resolv.h> 這行即可。

· ns_init.o: Undefined symbol _nsaddr_list referenced from text segment

原因： 程式誤用變數

解決方法：在 ns_init.c

第 558 行加入： struct state temp;

第 576 行： fwdaddr.sin_port = nsaddr.sin_port;

改成

fwdaddr.sin_port = temp.nsaddr.sin_port;

5.編譯 Hesiod 及測試程式。

cd ~userhome/hesiod/hesiod; make

TroubleShooting：

· hespwnam.c: In function `hes_getpwnam':

hespwnam.c:55: structure has no member named `pw_quota'

hespwnam.c:56: structure has no member named `pw_comment'

原因：在 include file 中所定義的 structure 並沒有包含這兩個欄位

解決方法：將 hespwnam.c 的第 55、56 行刪掉。

6.編輯 nameserver 所需的資料檔及設備檔。

資料檔的編輯方式在第四部份將會詳細解說。

7.重新啓動 named。

四、資料檔及設備檔的建立

在建立 Hesiod/BIND Service 所需的檔案時，我們可分作 Server 與 Client 兩部份加以討論。在 Client 方面，需要建立的檔案有：/etc/hesiod.conf、/etc/resolv.conf 、/etc/hosts 。在 Server 部分則有 /etc/hesiod.conf 、/etc/resolv.conf 、/etc/hosts 、athena.db 、/etc/namedb/named.boot 、/etc/namedb/named.local、/etc/namedb/named.ca、/etc/namedb/named.hosts、/etc/namedb/named.rev、/etc/namedb/passwd.db、/etc/namedb/localhost.rev。

以下將一一介紹如何建立這些檔案。

1.resolv.conf

以下是一個範例：

```
domain          net.ncu.edu.tw
nameserver  140.115.10.122
```

在這個範例中，domain 與 nameserver 都是識別字。 domain 後必須寫上所在的 domain name ，而 nameserver 後則寫入 nameserver 所在 IP address。

2.hesiod.conf

以下是一個範例：

rhs=.ncu.edu.tw

lhs=.net

在這個範例中，rhs 代表 Right Hand Side，lhs 則代表 Left Hand Side。在前面說明 Hesiod 與 BIND Service 的關係時我們提過：當 Hesiod 要將 Hesiod Name Service query 轉成 BIND format query 時，會去參考 resolv.conf 的 rhs 與 lhs，將 query 的名字 (HesiodName)、型別 (HesiodName Type)、與 lhs、rhs 連成一個 BIND Format 的字串，然後呼叫 resolver 去向 name server 查詢。舉例來說，當我們下指令：hesinfo muller passwd 時，Hesiod 會將它轉爲 BIND 格式的字串如下所示：muller.passwd.net.ncu.edu.tw。

至於 lhs 與 rhs 要如何設定並無一定規則，但根據我們的經驗只要使得產生的 BIND 格式字串在 nameserver 的資料檔中能夠查詢到便可以了。

3.hosts

以下是一個範例：

```
#
# Host Database
# This file should contain the addresses and aliases
# for local hosts that share this file.
# It is used only for "ifconfig" and other operations
```

```
# before the nameserver is started.
#
#
127.1          localhost localhost.my.domain
#
# Imaginary network.
140.115.10.122  athena.net.ncu.edu.tw athena
140.115.10.123  athena1.net.ncu.edu.tw athena1
```

一般而言，/etc/hosts 檔早就在作業系統安裝時就設定好了，在此是提醒大家將 name server 的 hostname 及 IP address 加入此檔中。

以下開始將介紹如何建立資料檔。在下列資料檔中常常會出現 SOA 這個 entry-type (前面提過)，意指 START OF AUTHORITY（指一個 zone 開始的地方，zone 是指一台 name server 所管轄的範圍）。其範例如下：

```
@     IN    SOA    athena.net.ncu.edu.tw. root.athena.net.ncu.edu.tw. (
                   1            ; serial
                   300          ; Refresh - 5
                   60           ; Retry 1 minute
                   1209600      ; Expire 2 weeks
                   43200 )      ; Minimum 12 hours
```

在 SOA 後接的是 nameserver、管理者的 E-Mail address 及 SOA 型別的資料。其資料的欄位名及意義如下：

serial#　　指出此資料檔的版本。

refresh　　指出多少秒後，Secondary Server 要向 Primary Server 查詢是否應更新所抓回的資料檔。

retry　　指出多少秒後，Secondary Server 要重試更新檔案的動作。

expire　　指出多少秒內，Secondary Server 在還未將所抓回來的資料檔丟棄或更新前使用其 cache 內的資料。

min　　指出所抓回來的資料檔的預設存活時間。

注意：Secondary Server 會在 Primary Server 的資料檔的 SOA 的 #serial 這欄改變後，才會去將抓回來的資料更新。

4.named.boot

以下是一個範例：

```
;
;       named.boot file for the Primary server
;format：
;type        domain/zone              source data file/IP address
;
directory     /etc/namedb
;
primary       net.ncu.edu.tw.              athena.db
;
primary       10.115.140.in-addr.arpa.     hosts.rev
;
```

```
primary       0.0.127.in-addr.arpa.          named.local
;
secondary     subnet.net.ncu.edu.tw.         140.115.10.114    named.hosts.subnet
;
secondary     passwd.subnet.net.ncu.edu.tw.  140.115.10.114    named.passwd
;
cache         net.ncu.edu.tw.                named.ca
```

這是一個很重要的檔案，告訴 name server 在啓動時的一些必要資訊。首先，要聲明的是，所有的資料檔其格式都如同前面第參部份所介紹的形式，同時在資料檔的內容內是不管大小寫的差別的。在這個範例檔中，我們看到了幾個識別字 (Identifier)：directory、primary、secondary、cache 等等。directory 是告訴 name server 以下所列的資料檔是在哪個目錄下。primary、secondary、cache 則是指名本身是屬於何種類型的 name server。以上述的內容來看，說明了本身當 net.ncu.edu.tw. 這個 domain 的 primary server，同時資料檔放在 /etc/namedb/athena.db 及 /etc/namedb/hosts.rev；另外，當 net.ncu.edu.tw. 的 cache server，資料檔放在 /etc/namedb/named.ca。另外，這台機器本身也當 subnet.net.ncu.edu.tw. 這個 domain 的 secondary server，資料檔放在 [140.115.10.114] 這臺機器上的 /etc/namedb/named.hosts.subnet 及 /etc/namedb/passwd.db。由於 secondary server 在重新啓動 named 時會根據 named.boot 的設定自動到 primary server 處抓取資料，因此我們將 domain、data file 的欄位設定好即可獲得其他 domain primary server 的資料了。

5.named.local

以下是一個範例：

```
;
;          ncu zone named.local for server sun1
;
$ORIGIN      net.ncu.edu.tw.
@     IN     SOA    athena.net.ncu.edu.tw. root.athena.net.ncu.edu.tw. (
                    1            ; serial
                    300          ; Refresh - 5
                    60           ; Retry 1 minute
                    1209600      ; Expire 2 weeks
                    43200 )      ; Minimum 12 hours
      IN     NS     athena.net.ncu.edu.tw.
      IN     PTR    localhost.
localhost.    IN     A      127.0.0.1
```

這個檔案只是說明 local lookback interface 的位址。

6.localhost.rev

以下是一個範例：

```
      @(#)localhost.rev      5.1 (Berkeley) 6/30/90

@     IN     SOA    athena.net.ncu.edu.tw. root.athena.net.ncu.edu.tw. (
                    1            ; serial
                    300          ; Refresh - 5
```

```
                60          ; Retry 1 minute
                1209600     ; Expire 2 weeks
                43200 )     ; Minimum 12 hours
        IN      NS      athena.net.ncu.edu.tw.
1       IN      PTR     localhost.net.ncu.edu.tw.
```

同上，照抄即可。

7.named.ca

以下是一個範例：

```
;
;       start the cache data for root domain server
;
.           99999999    IN      NS      ns.nic.ddn.mil.
            99999999    IN      NS      ns.nasa.gov.
            99999999    IN      NS      terp.umd.edu.
            99999999    IN      NS      kava.nisc.sri.com.
            99999999    IN      NS      aos.brl.mil.
            99999999    IN      NS      c.nyser.net.
            99999999    IN      NS      nic.nordu.net.edu.tw.
            99999999    IN      NS      moevax.edu.tw.
;
;       list the server's address
;
ns.nic.ddn.mil.  99999999    IN      A       192.112.36.4
ns.nasa.gov.     99999999    IN      A       128.102.16.10
```

```
ns.nasa.gov.    99999999    IN    A    192.52.195.10
kava.nisc.sri.com.    99999999    IN    A    192.33.33.24
aos.brl.mil.    99999999    IN    A    192.5.25.82
aos.brl.mil.    99999999    IN    A    26.3.0.29
aos.brl.mil.    99999999    IN    A    128.63.4.82
c.nyser.net.    99999999    IN    A    192.33.4.12
terp.umd.edu.    99999999    IN    A    128.8.10.90
```

關於這個檔案我們通常只是照抄，因爲他是用來指明在較高層 domain 的 cache server。

8.athena.db

以下是一個範例：

```
;
; Authoritative data for athena.net.ncu.edu.tw,    class HS
;
@    IN    SOA    athena.net.ncu.edu.tw. root.athena.net.ncu.edu.tw. (
                1            ; serial
                300          ; Refresh - 5
                60           ; Retry 1 minute
                1209600      ; Expire 2 weeks
                43200 )      ; Minimum 12 hours
     IN    NS    athena.net.ncu.edu.tw.
     HS    NS    athena.net.ncu.edu.tw.

$INCLUDE /etc/namedb/named.hosts
```

$INCLUDE /etc/namedb/passwd.db

$INCLUDE /etc/namedb/named.rev

在這個檔案中，有一個識別字：$INCLUDE，是類似 C 語言裡 #include 的功能，只不過在此是將資料檔包含進來的意思。注意，我們在 named.boot 中指明 domain ： net.ncu.edu.tw. 的資料檔是 athena.db ，其餘譬如： named.hosts 、 named.rev 、 passwd.db 等我們想要加入的資料檔都在 athena.db 這個檔裡用 $INCLUDE 這個識別字包含進來了。

9.named.hosts

以下是一個範例：

```
;
;          net zone hosts file for server athena
;
$ORIGIN    net.ncu.edu.tw.
@          IN    SOA    athena.net.ncu.edu.tw. root.athena.net.ncu.edu.t
                  11.01    ;Serial
                  360000   ;Refresh
                  300      ;Retry
                  3600000  ;Expire
                  3600 )   ;Minimum
           IN    NS     athena.net.ncu.edu.tw.
           IN    A      140.115.10.122
localhost  IN    A      127.0.0.1
```

```
athena      IN   A    140.115.10.122
athena1     IN   A    140.115.10.123
athena2     IN   A    140.115.10.121
subnet      IN   NS   test114.subnet.net.ncu.edu.tw
            IN   A    140.115.10.114
$ORIGIN     ncu.edu.tw.
@           IN   NS   rs540.ncu.edu.tw.
            IN   NS   sun1.ncu.edu.tw.
*.dnet      IN   MX   10   ncu865.ncu.edu.tw.
rs540       IN   A    140.115.19.42
sun1        IN   A    140.115.1.31
ncu865      IN   A    140.115.1.101
twncu865    IN   A    140.115.1.101
ncu-gate    IN   A    140.115.1.254
```

這個檔案主要是用來查詢 hostname => IP address 之用。

10.named.rev

以下是一個範例：

```
@       IN   SOA    athena.net.ncu.edu.tw. root.athena.net.ncu.edu.tw. (
                 11.01     ;Serial
                 360000    ;Refresh
                 300       ;Retry
                 3600000   ;Expire
                 3600 )    ;Minimum
```

```
        IN    NS    athena.net.ncu.edu.tw.
122     IN    PTR   athena.net.ncu.edu.tw.
123     IN    PTR   athena1.net.ncu.edu.tw.
```

這個檔案主要是用來查詢 IP address => hostname 之用。

10.passwd.db

以下是一個範例：

```
; This file contains /etc/passwd entries for lookup in the name server.
; Passwords are not stored here; they are handled by the Kerberos
; authentication system.
;       modified by fong
$origin passwd.net.ncu.edu.tw.
@       IN      SOA     athena.net.ncu.edu.tw. root.athena.net.ncu.edu.tw. (
                        1               ; serial
                        300             ; Refresh - 5
                        60              ; Retry 1 minute
                        1209600         ; Expire 2 weeks
                        43200 )         ; Minimum 12 hours
        HS      NS      athena.net.ncu.edu.tw.
benjamin    HS      TXT
        "benjamin:F4j8PmyA8Uubk:100:100::/home/benjamin:
muller      HS      TXT
        "muller:4q3HB0ABRJuiE:104:100:muller:/home/mulle
fong        HS      TXT
```

"fong:c0jNirs509QbE:106:100:fong:/home/fong:/bin

這個檔案主要是用來查詢 user password 之用。

3、4 節 Hesiod/BIND Name Service 的應用

在本文一開始有提過，Hesiod Service 所提供的是一個比 BIND 更具彈性、更廣泛的查詢內容的服務，但到底如何彈性呢？請看下文！

除了基本 BIND 所有的查詢內容外，Hesiod還提供下列預設的資料查詢形態：

HesiodName	HesiodNameType	Used By Info Returned
workstation name	"cluster"	get cluster workstation；cluster
filesystem name	"filsys"	RVD and NFS file system
username	"pobox"	location and type of mailbox；mailbox
username	"passwd"	login Athena-wide；/etc/passwd
uid (ASCII)	"uid"	getpwent()；Athena-wide UID to username
group name	"group"	getgrent()；Athena-wide /etc/group
group name	"grplist"	getgrent()；Athena-wide group list
gid (ASCII)	"gid"	getgrent()；Athena-wide GID to group
printer name	"pcap"	getent()；Athena-wide /etc/printcap entry

除此之外，應用程式還可以自行設定 HesiodNameType，這才是我在上面所提到 Hesiod 最吸引人的彈性。舉個例說，我們定義一個 test 這個 HesiodNameType，只需要做下列動作就可以查詢到相關的資料。

1.建立 test.db。

test.db sample file

```
$origin test.net.ncu.edu.tw.
@     IN    SOA    athena.net.ncu.edu.tw. root.athena.net.ncu.edu.tw. (
                   1            ; serial
                   300          ; Refresh - 5
                   60           ; Retry 1 minute
                   1209600      ; Expire 2 weeks
                   43200 )      ; Minimum 12 hours
      HS    NS     athena.net.ncu.edu.tw.
test1       HS     TXT   "This is test1 data"
test2       HS     TXT   "This is test2 data"
```

2.在 athena.db 內用 $include 將此檔案包含進去。

```
$include    /etc/namedb/test.db
```

做好上述動作後，我們可利用 hesinfo 來測試新的設定。
當我們下 hesinfo test1 test 後，我們可獲得回答 This is test1 data。

至於若應用程式要自行查詢 nameserver，則可使用 hes_to_bind() 及 hes_resolv() 這兩個程式呼叫。

3、5 節 測試 -- hesinfo 與 nslookup 的使用

在我們建立 Hesiod/BIND Service 的時候，常常不知道我們的設定對不對，或是不知道錯誤出在何處。直到我們開始使用 nslookup 與 hesinfo 時，我們才開始能掌握我們的設定結果。下列將介紹這兩個有用的工具程式，相信對初次建立 Hesiod/BIND Service 的人有很大的幫助。

hesinfo：

Usage: hesinfo [-bl] identifier type

-l selects long format

-b also does hes_to_bind conversion

Example: ***hesinfo -bl muller passwd***

hesinfo test1 test

nslookup：

Usage: 請用 man nslookup 查詢。在這裡要說明的是：當要查詢 Hesiod 類型的詢問時，須先做下列設定

athena>***nslookup***

> ***set cl=hs*** ; 設定 Address Class = HS

> ***set q=txt*** ; 設定 Query Type = TXT

> ***set debug*** ; 設定爲除錯狀態，對於我們除錯有相當相當大的幫助

3、6 節 如何利用Hesiod Server達成共通帳號

3.6.1 傳統登錄的方法

由於Hesiod 可以提供各種網路上的各種物件資訊服務，當然包括網路使用者的簽入資料，因此我們可以利用它來發展共通帳號的服務。

在ＵＮＩＸ系統中，有一個檔案專門記錄所有使用者身份資料，它是/etc/passwd。當使用者要login 系統時，login 的程式就會到這個檔案中查看有無此人，並且檢查密碼是否正確，如果完全正確系統就會載入一個使用者指定的shell，並將它的home directory指到預設的地方完成login 的動作。可見得/etc/passwd決定一個人是否能進入到系統中。

但是在網路發達的現在，分散式計算已蔚爲風潮，使用者往往有很多的帳號和密碼，這對於使用者來說是不方便的，因爲要記憶這麼多的密碼和帳號實在是非常累人的一件事，而且當一位使用者要申請一個帳號時，管理者就要在每一部ＵＮＩＸ系統上的/etc/passwd建立使用者的資料，當使用者有所更動時，亦要每一部都更改，要維持資料的一致性非常困難。因此若能將這張passwd表集中管理，讓每一部機器都向它查詢使用者的登錄資料，如此一來，當一個使用者來申請帳號時只要在這張集中管理的passwd加入使用者資料即可。使用者亦只要記一個帳號和密碼就可以同時使用數部機器了。

在傳統的ＵＮＩＸ系統中的login程式是利用一個system call 'getpwnam()'來到/etc/passwd中找使用者的資料，並將使用者的資料分割爲各個欄位，再填入一結構變數struct passwd中，再將指向這變數的指標傳回呼叫程式;如果找不到則傳回ＮＵＬＬ。這個system cal的傳入值是使用者帳號的字串，傳回是一指向struct passwd的指標。struct passwd的結構如下:

```
struct passwd {
    char    *pw_name;
```

```
    char    *pw_passwd;
    int     pw_uid;
    int     pw_gid;
    char    *pw_age;
    char    *pw_comment;
    char    *pw_gecos;
    char    *pw_dir;
    char    *pw_shell;
};
```

剛好就是/etc/passwd內每筆記錄各欄的資料。

3.6.2 修改login.c程式

因此要使數部機器共用同一張的passwd，就要改變傳統login 程式向local 的/etc/passwd詢問資料，轉而向遠端的Hesiod Server詢問即可。

你可以拿兩種不同的版本的login程式來修改：一是用FreeBSD UNIX所附的source code來修改，它是在/usr/src/usr.bin/login這個目錄下，但是如果你的網路環境有Kerberos的話，則要使用Kerberos所附的login應用程式來修改它是位於../kerberos/obj/appl/bsd這個路徑下

在Compiler Hesiod之後它提供了一個library叫做hesiod.a，如果你已經完整地安裝Hesiod，那你可以在/usr/lib這個目錄下找到。裡面有一個函數叫hes_getpwnam()，它可以向遠端的Hesiod server詢問使用者的登入資料並傳回一指向struct passwd的指標，如果查無此人則傳回ＮＵＬＬ，完全和系統原本的system call 'getpwnam()'一樣，只不過getpwnam()是向本機的/etc/passwd詢問，而hes_getpwnam()則是向遠端的Hesiod server詢問，從呼叫程式面來看，兩者是完全相同的，因此若要使login 程式不要向本機的/etc/passwd查詢使用者登錄資料，轉而向遠端的Hesiod server詢問，只要將login.c中呼叫的getpwnam()改爲hes_getpwnam()即可。作法如下:

a). 使用FreeBSD UNIX 所附的login 原始程式來修改

1.login的原始成所在的目錄爲/usr/src/usr.bin/login

athena>/usr/src/usr.bin/login 2# ls

Makefile login.1 obj@

klogin.c login.c pathnames.h

2.vi login.c 將呼叫getpwnam()的地方改爲hes_getpwnam()

事實上login.c中只有一個地方呼叫到getpwnam() （第222行），你可以用vi所提供的尋找功能來找（按ESC鍵使vi確定在命令狀態，再鍵入/getpwnam它就會幫你找到）它的原始敘述如下：

```
if (pwd = getpwnam(username))
        salt = pwd->pw_passwd;
    else
        salt = "xx";
```

我們並不建議你直接將getpwnam()替換爲 hes_getpwnam()，因爲如果你這樣做的話，任何人要login到這部機器上時，它只會向Hesiod server詢問，將來如果Hesiod server當機後，它便因爲詢問不到任一使用者的登錄資料而使得任何一人都無法進入系統，這樣做實在太危險了。因此我們建議你將上面這一片段的原始程式改爲：

```
if (pwd = getpwnam(username))  <--先向本機的/etc/passwd查詢
    salt = pwd->pw_passwd;
else
    if (pwd=hes_getpwnam(username))    <--如果本機找不
```

salt = pwd->pw_passwd; 到再向Hesiod查詢

else

salt = "xx"; <--無此使用者

3.接下來就要重新來編譯login.c了，我們不需要將整個UNIX的source code重新make一次，只要將login.c單獨編譯就可以了。在重新編譯時，要將你在安裝Hesiod時所產生的 hesiod.a指定給編譯器，在這裡我的hesiod.a是位於/usr/lib之內，要下的命令是cc -o login login.c /usr/lib/hesiod.a -lutil；以下是我的實作：

athena>/usr/src/usr.bin/login 34# ***cc -o login login.c /usr/lib/hesiod.a -lutil***

login.c: In function `main':

login.c:225: warning: assignment makes pointer from integer without a cast

athena>/usr/src/usr.bin/login 35#ls

Makefile login* login.c obj@

klogin.c login.1 login.c.org pathnames.h

login* 就是我們所要的。這樣就完成重新編譯的工作了。

4.將login chmod 成 4555

athena> ***chmod 4555 login***

5.將login拷貝到/usr/bin下。

athena> ***cp login /usr/bin/login***

6.確定/etc/reslov.conf檔案之中的nameserver是否指到你這個domain的Hesiod server。例如我的Hesiod server是在140.115.10.122，我所在的domain是net.ncu.edu.tw，那我的/etc/resolv.conf就要這樣設定：

athena>/usr/bin 48# ***more /etc/resolv.conf***

domain net.ncu.edu.tw

nameserver 140.115.10.122

7.現在你就可以重新啓動你的系統，讓系統使用我們新編譯的login程式。當系統重新啓動後，原本在本機註冊的使用者可以照常使用，而且它也可以讓在Hesiod server註冊過的使用者登錄。

以上是利用ＢＳＤ ＵＮＩＸ所附的login source code來改寫，使它具有向遠端Hesiod server查詢的能力。如果你的環境有Kerberos server則建議你使用Kerberos所附的應用軟體login來修改。Kerberos所附的login原始程式所在目錄爲../../kerberos/obj/appl/bsd修改方法亦是將login.c程式中呼叫getpwnam()改爲hes_getpwnam()，再重新compile一次即可。此login除了會向Hesiod server詢問passwd之外亦會向kerberos做認證的動作並取得ticket。方法如下

b). 使用kerberos所附的login應用程式來修改

1.我的kerberos的login應用程式位於/mit/kerberos/obj/appl/bsd

athena>cd /mit/kerberos/obj/appl/bsd

2.vi login.c將程式中所有呼叫getpwnam()的地方改爲hes_getpwnam()

跟前面修改過的ＢＳＤ version的login.c一樣，我們不能只是將所有的getpwnam()直接替換爲hes_getpwnam()而已，還要略做修改。在kerberos所提供的應用程式login.c中，呼叫getpwnam()的地方共有三處（第353,825,894行），你也可以用vi的搜尋功能（先按ＥＳＣ鍵確定爲命令狀態，再鍵入/getpwnam它就會找到，並將指標停在getpwnam的字首，如要

找下一個則按N鍵）這三處的原始敘述爲

```
.
.
if (pwd = getpwnam(username))                    <--Line 353
        salt = pwd->pw_passwd;
    else
        salt = "xx";
.
.
pwd = getpwnam(username);                        <--Line 825
    if (pwd == NULL)
        return(-1);
.
.
  pwd = getpwnam(lusername);                     <--Line 894
    if (pwd == NULL) {
        pwd = NULL;
        return(-1);
    }
.
.
```

我們要將上面三個敘述修改爲

```
.
.
if (pwd = getpwnam(username))
        salt = pwd->pw_passwd;
    else
```

```
                    if (pwd = hes_getpwnam(username))
                            salt = pwd->pw_passwd;
                    else
                salt = "xx";


pwd = getpwnam(username);
    if (pwd == NULL){
            pwd = hes_getpwnam(username);
            if (pwd == NULL)
          return(-1);
    }


  pwd = getpwnam(lusername);
    if (pwd == NULL) {
              pwd = hes_getpwnam(lusername);
              if (pwd == NULL){
          pwd = NULL;
          return(-1);
              }
    }
```

3.***vi Makefile*** 在第58行處加入一行變數HES_LIB=/usr/lib/hesiod.a <--要設定爲你的hesiod.a所在的路徑：

NOENCFLAG=

HES_LIB=/usr/lib/hesiod.a <--加入這一行

DES_LIB=$(BUILDTOP)/lib/des/libdes.a

DES_LIBDEP=$(DES_LIB)

DES_LINTLIB=$(BUILDTOP)/lib/des/llib-ldes.ln

，並在220行要編譯login.krb的地方，在最後面加上${HES_LIB}：

login.krb: ${KRB_LIBDEP} ${DES_LIBDEP} login.o

$(CC) $(CFLAGS) -o $@ login.o ${KRB_LIB} ${DES_LIB} **${HES_LIB}**

^^^^^^^^^^

新加入的變數

4.make

athena#/mit/kerberos/obj/appl/bsd>***make*** <enter>

5.完成之後你會得到一個可執行的login.krb你可以將之rename成login並chmod 成4555，再拷貝到/usr/bin下即可。

6.同樣的，你現在可以重新啓動你的系統了。或者你可以試驗一下login程式是否正常。你可以下命令 ./login 來執行，注意！一定要指定是要執行目前目錄下的login而非系統的login。並且要在可以執行login的shell下執行才可以。（第一次由系統登錄時所載入的shell，不可以執行過su的指令，如果有執行過，請先退出(Ctrl d)再執行）。

3.6.3 管理方法

當你安裝了Hesiod/Bind server以及修改好login.c的程式後，你必需注意下列事項：

1.將編譯好的login可執行檔放到/usr/bin並且chmod至4555。

2.確定/etc/resolv.conf中的nameserver是指向你的Hesiod/Bind server。

檢查這兩項事情後，接下來所碰到的事是如何爲你的使用者建立一個新的帳號，步驟如下：

1.帳號統一建在Hesiod/Bind server中的資料庫，這個資料庫是etc/namedb/passwd.db

2.假設一位新的使用者叫muller密碼是vga256，管理者首先要用vipw將這位使用者建入系統，其目的只是要看密碼vga256被系統編密後的結果是什麼假設是Wd0rDnx7得到這一串密文後就可以將使用者的所有資料填入Hesiod/Bind server中的資料庫/etc/namedb/passwd.db中。步驟如下：

a).管理者用vipw將使用者（假設是muller）加入系統中。

athena# ***vipw***

cym:l1.Ezzsv.o0vs:107:100::0:0:cym:/home/cym:/bin/tcsh

hilbert:sWx0M9120nQbU:108:100::0:0:alias of cym:/home/cym:/bin/tcsh

pass:WBhdmESuWCG6s:110:100::0:0:pass:/home/pass:/bin/csh

muller::106:100::0:0:muller:/home/muller:/bin/csh

~

~

！注意：**muller**的第二個欄位是空的（沒有斜體字的密碼密文）這表示

muller第一次login時不須要密碼就可以進入。

b).用muller進入系統（不需要密碼），進入後用passwd這個更改命令的指令來設定密碼。

FreeBSD (athena) (ttyp4)

login:***muller***

%passwd

passwd

Changing local password for muller

New password:***vga256***

Retype new password:***vga256***

Passwd: rebuilding the database...

Passwd: done

c).系統管理者再用vipw這個指令去看muller的密碼vga256被編密後的結果是什麼。

athena# ***vipw***

cym:l1.Ezzsv.o0vs:107:100::0:0:cym:/home/cym:/bin/tcsh

hilbert:sWx0M9120nQbU:108:100::0:0:alias of cym:/home/cym:/bin/tcsh

pass:WBhdmESuWCG6s:110:100::0:0:pass:/home/pass:/bin/csh

muller:**G2l07f9tkV.2U**:106:100::0:0:muller:/home/muller:/bin/csh

~

~

由此可知muller的密碼vga256被系統編密後的密文爲**G2l07f9tkV.2U**

d).將使用者的資料正式加入Hesiod server的資料庫中

athena# />***vi /etc/namedb/passwd.db***

muller HS TXT

"muller:**G2l07f9tkV.2U**:1008:999:muller:/home/muller:/bin/csh"

^ ^ ^ ^

tab tab tab 將密文抄到這裡

注意！當你建入一位新的使用者到/etc/namedb/passwd.db中之後，並不代表這位使用者就已完成而可以登錄了，而必須重新啓動Hesiod/Bind server讓它rebuild資料庫才會有作用。要重新啓動Hesiod/Bind server

athena# /> ***cd /etc***

athena# /etc> ***./named /etc/namedb/named.boot***

3.如果你有Kerberos server，而且你的login是用Kerberos所附的應用程式login.c所修改的，則一位新的使用者要建立時，除了要在Hesiod/Bind server註冊外，還外還要向kerberos server註冊並且要使用相同的username及passwd。如何向kerberos server 註冊，請看第四章kerberos的建置。

也許你會感到要建立一位使用者竟是那麼麻煩，要有那麼多的步驟，尤其是要先將使用者的密碼先編密後，再將密文抄到眞正的資料庫中，這是由於原本這些繁雜的工作應由一個Morra的應用軟體來做的，你只要輸入一次使用者的資料morra就會將密碼編密後寫到Hesiod/Bind server和Kerberos的資料庫中，但是目前我們暫不使用morra而用人工的方式來建立。

七、參考資料

1. Guide to Kerberos

June 1990 ULTRIX Version 4.0 or higher

2. READMEs and Document in Hesiod Distribution

~username/hesiod/INSTALL

~username/hesiod/README

~username/hesiod/doc/hesiod.PS