國立暨南國際大學

通訊工程研究所

972 網際網路通訊協定第六版
期末報告

BBS Everywhere

97321526 王筱婷

97321538 李霓雅

97321539 黃信富

中 華 民 國 九 十 八 年 六 月 二 十 五 日

一、 動機

隨著可上網設備的增加，若提供每個設備一個至多個 IP，那麼 IPv4 的位置即將不敷使用，也因此 IPv6 在這樣的背景下誕生。而目前國內的 Bulletin Board System(以下簡稱 BBS) Service，大多僅支援 IPv4 服務，只有少數的 BBS 站台，如銘傳資工(bbs.mcu.edu.tw)支援 IPv6 的連線，最後終歸都會支援 IPv6 的連線。我們期末專題的目的是要讓現有的 BBS Client 端軟體(PCManX)可以跟上 BBS Server 端的腳步，提供 IPv6 連線的機制。這樣一來，Client 跟 Server 就可以用 IPv6 Address 連線，即使 IPv4 位置枯竭了，對於我們索取 BBS 的服務也不會因此受到中斷。

二、 實驗環境

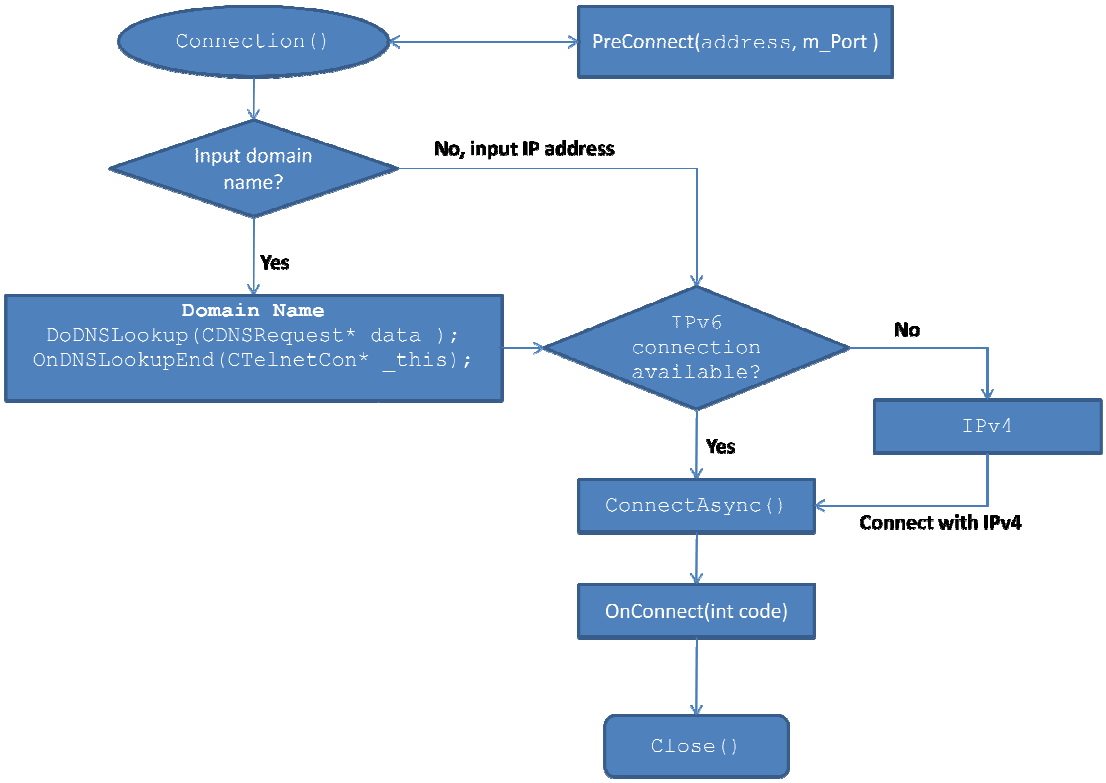
我們所要修改的 BBS 軟體是 PCManX-0.3.8，PCManX 是 PCMAN 在 Linux 的版本，由於它是 Open Source 並且允許我們對內部的程式做修改、散佈，所以我們修改它，讓它可以同時支援 IPv4 與 IPv6，不管是 IPv4 或 IPv6 Address 皆可以成功連線。

作業系統是採用 Linux 眾多發行版之一的 CentOS 5.2。

編譯環境需求:

- GTK+ 2.4 以上
- Xft 2.x
- Intltool
- Automake
- libtool

三、 流程圖


Connection()
PreConnect(address, m_Port)
Input domain name?
No, input IP address
Yes
Domain Name
DoDNSLookup(CDNSRequest* data);
OnDNSLookupEnd(CTelnetCon* _this);
IPv6 connection available?
No
IPv4
Yes
ConnectAsync()
Connect with IPv4
OnConnect(int code)
Close()

四、 程式修改

黑色粗體字為說明程式之註解。

符號說明： “+“為新增之程式碼； “-“為被刪除掉之程式碼。

1. src/view/telnetcon.h

```
class CTelnetCon : public CTermData
{
        …
protected:
        …
        string m_LoginPrompt;
        string m_PasswdPrompt;
        static int m_SocketTimeout;
-       in_addr m_InAddr;
+       in6_addr m_InAddr;          // v6 Address Structure
+       in_addr m4_InAddr;
        unsigned short m_Port;
        void PreConnect(string& address, unsigned short& port);
        void CheckAutoLogin(int row);
};
```

2. src/view/telnetcon.cpp

```
// 新增全域變數
+       bool isIPv6 = 0;  // 是否有 IPv6 Address。有為”1”，沒有為”0”。
+       bool IPv6Crash = 0;   // IPv6 Address 是否連線失敗。失敗為”1”。

void CTelnetCon::PreConnect(string& address, unsigned short& port)
{
        m_Duration = 0;
        m_IdleTime = 0;
        m_State = TS_CONNECTING;

// 字串分割處理。原本 IP Address(Domain Name)與 Port Number 是以”:”符號做區分，如 127.0.0.1:23；在
這裡為了可以與 v6 Address 做個分別，我們以”@”符號來做區分，如 2001:da8:8000:6003::161@23。
```

```
-       int p = m_Site.m_URL.find(':',true);
+       int p = m_Site.m_URL.find('@',true);
        if( p >=0 )         // use port other then 23;
        {
                port = (unsigned short)atoi(m_Site.m_URL.c_str()+p+1);
                address = m_Site.m_URL.substr(0, p);
        }
        else
                address = m_Site.m_URL;
}

bool CTelnetCon::Connect()
{
        …
        if ( m_Port == 23 && m_Site.m_UseExternalTelnet )
        {
                …
        }
        /* external ssh */
        else if ( m_Port == 22 && m_Site.m_UseExternalSSH )
        {
                …
        }
        else // Use built-in telnet command handler
#endif
        {
// (1) 如果 address 字串符合 IPv6 Address 格式，則直接使用 IPv6 Address 連結。
// (2) 如果 address 字串符合 IPv4 Address 格式，則直接使用 IPv4 Address 連結。
// (3) 其他類型的字串，則視為 DNS 字串，丟給處理 DNS 的函數做處理。
-               if( m_InAddr.s_addr != INADDR_NONE || inet_aton(address.c_str(), &m_InAddr) )
-                       ConnectAsync();
+               if( inet_pton(AF_INET6 ,address.c_str(),&m_InAddr)){
+                       isIPv6 = 1;
+                       ConnectAsync();
+               }else if( m4_InAddr.s_addr != INADDR_NONE || inet_aton(address.c_str(), &m4_InAddr) ){
+                       isIPv6 = 0;
+                       ConnectAsync();
+               }
```

```
            else // It's a domain name, DNS lookup needed.
            {
                g_mutex_lock(m_DNSMutex);
                CDNSRequest* dns_request = new CDNSRequest(this, address, m_Port);
                m_DNSQueue.push_back( dns_request );
                if( !m_DNSThread ) // There isn't any runnung thread.
                    m_DNSThread = g_thread_create( (GThreadFunc)&CTelnetCon::ProcessDNSQueue,
NULL, true, NULL);
                g_mutex_unlock(m_DNSMutex);
            }
        }

    return true;
}

void CTelnetCon::DoDNSLookup( CDNSRequest* data )
{
-       in_addr addr;
-       addr.s_addr = INADDR_NONE;
+       in6_addr addr;          // v6 address structure
+       in_addr addr4;          // v4 address structure
+       addr4.s_addr = INADDR_NONE;

//  Because of the usage of thread pool, all DNS requests are queued
//  and be executed one by one.  So no mutex lock is needed anymore.
-       if( ! inet_aton(data->m_Address.c_str(), &addr) )
-       {
//  gethostbyname is not a thread-safe socket API. -       if( host )
-               addr = *(in_addr*)host->h_addr_list[0];
-       }
+       hostent *host, *host4;
        // 藉由 Domain Name 查詢 IPv6 Address 。
+       if( host=gethostbyname2(data->m_Address.c_str(), AF_INET6) ){
            // 取得之後，儲存 IPv6 Address，並將 isIPv6 設為 1，代表此 Domain Name 有 IPv6 位置。
+           addr = *(in6_addr*)host->h_addr_list[0];
+           isIPv6 = 1;
            // 為了防止無法經由 IPv6 Address 連線，我們也查詢了 IPv4 位置。IPv6Crash 設為 1，則代表如果
            IPv6 連線不成功，還有 IPv4 可以支援連線。
```

```
+           if(host4 = gethostbyname(data->m_Address.c_str())){
+                   addr4 = *(in_addr*)host4->h_addr_list[0];
+                   IPv6Crash = 1;
+           }
        // 若此 Domain Name 查不到 IPv6 Address，則我們查詢 IPv4 Address。
+       }else if( host4 = gethostbyname(data->m_Address.c_str())){
+           addr4 = *(in_addr*)host4->h_addr_list[0];
+           isIPv6 = IPv6Crash = 0;
+       }

        g_mutex_lock(m_DNSMutex);
        if( data && data->m_pCon)
        {
-           data->m_pCon->m_InAddr = addr;
            // 若有 IPv6 Address，則用 IPv6 Address 連線；若 IPv6 Address 連線失敗(IPv6Crash==1)，那麼
            我們就使用 IPv4 Address 來連線。
+           if(isIPv6){
+               data->m_pCon->m_InAddr = addr;
+               if(IPv6Crash) data->m_pCon->m4_InAddr = addr4;
            // 沒有 IPv6 Address，則使用 IPv4 Address 連線。
+           }else{
+               data->m_pCon->m4_InAddr = addr4;
+           }
            g_idle_add((GSourceFunc)OnDNSLookupEnd, data->m_pCon);
        }
        g_mutex_unlock(m_DNSMutex);
}


gboolean CTelnetCon::OnDNSLookupEnd(CTelnetCon* _this)
{
       INFO("CTelnetCon::OnDNSLookupEnd");
       g_mutex_lock(m_DNSMutex);
-      if( _this->m_InAddr.s_addr != INADDR_NONE )
+      if( isIPv6 || (_this->m4_InAddr.s_addr != INADDR_NONE) ){
           _this->ConnectAsync();
       g_mutex_unlock(m_DNSMutex);
       return false;
```

```
}
void CTelnetCon::ConnectAsync()
{
	int err;
-	sockaddr_in sock_addr;
-	sock_addr.sin_addr = m_InAddr;
-	sock_addr.sin_family = AF_INET;
-	sock_addr.sin_port = htons(m_Port);
+	sockaddr_in6 sock_addr6;
+	sockaddr_in sock_addr;

+	if( isIPv6 ){
+		sock_addr6.sin6_addr = m_InAddr;
+		sock_addr6.sin6_family = AF_INET6;
+		sock_addr6.sin6_port = htons(m_Port);
+	}else{
+		sock_addr.sin_addr = m4_InAddr;
+		sock_addr.sin_family = AF_INET;
+		sock_addr.sin_port = htons(m_Port);
+	}

#ifdef USE_PROXY
	if ( m_Site.m_ProxyType == PROXY_NONE ) // don't use proxy server
	{
#endif
-		m_SockFD = socket(PF_INET, SOCK_STREAM, 0);
+		if( isIPv6 ){
+			m_SockFD = socket(PF_INET6, SOCK_STREAM, 0);
+		}else{
+			m_SockFD = socket(PF_INET, SOCK_STREAM, 0);
+		}
		int sock_flags = fcntl(m_SockFD, F_GETFL, 0);
		fcntl(m_SockFD, F_SETFL, sock_flags | O_NONBLOCK);

		…

		setsockopt(m_SockFD, IPPROTO_TCP, TCP_NODELAY, (char *)&sock_flags, sizeof(sock_flags));
-		err = connect( m_SockFD, (sockaddr*)&sock_addr, sizeof(sockaddr_in) );
```

```
+           if( isIPv6 ){
+               err = connect( m_SockFD, (sockaddr*)&sock_addr6, sizeof(sockaddr_in6) );
+           }else{
+               err = connect( m_SockFD, (sockaddr*)&sock_addr, sizeof(sockaddr_in) );
+           }
            fcntl(m_SockFD, F_SETFL, sock_flags );

    …
}


void CTelnetCon::OnConnect(int code)
{
    if( 0 == code )
        …
    }
    else
    {
// 使用 IPv6 Address 連線失敗，並且此 Domain Name 有 IPv4 Address，則使用 IPv4 Address 重新連線。
+           if(isIPv6 && IPv6Crash){
+               isIPv6 = 0;
+               IPv6Crash = 0;
+               ConnectAsync();
+           }else{
                m_State = TS_CLOSED;
                Close();

                …

                if( GetView()->GetParentFrame()->GetCurView() == m_pView )
                {
                    for( unsigned int col = 0; col < sizeof(failed_msg); )
                        col += m_pView->DrawChar( 0, col );
                }
#endif
+           }
    }
}
```